KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Science

Development of a dedicated laser-polarization beamline for ISOLDE-CERN

Wouter Gins

Supervisor:
Prof. dr. G. Neyens

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Science (PhD): Physics

January 2019

Development of a dedicated laser-polarization beamline for ISOLDE-CERN

Wouter GINS

Examination committee:
Prof. dr. E. Janssens, chair
Prof. dr. G. Neyens, supervisor
Prof. dr. N. Severijns
Prof. dr. R. Raabe
Prof. dr. T. E. Cocolios

Dr. M. L. Bissell
(The University of Manchester)
Dr. M. Kowalska
(CERN)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Science (PhD): Physics

January 2019

Uitgegeven in eigen beheer, Wouter Gins, Celestijnenlaan 200D - 2418, B-3001 Leuven (Belgium)

Dankwoord

De laatste loodjes wegen het zwaarst, maar als de laatste stap van het schrijven van een thesis het dankwoord is, is de last toch net iets minder groot.

Eerst en vooral: Gerda, bedankt voor de begeleiding die reeds tijdens mijn master begon. Door de uitdaging die je mij hebt overtuigd vol te houden ben ik volledig veranderd in de laatste vier jaar, ten goede. Merci!

I also thank all the members of my examination committee (in no particular order): Gerda, Nathal, Riccardo, Ewald, Thomas, Magda and Mark. I truly enjoyed the critical questions and discussions, although it might not have looked like it at certain points...

I am thankful for all the members of the Nuclear Moments group: Hanne, Ivan, Jasna, Ronald, Kara, Ruben, Agi, Xiaofei, Xu, Marieke, Tassos and Fredrik. Sharing offices/shifts with all of you has been a pleasure!

Of course, being part of several collaborations, I have had the opportunity to work with a lot of wonderful colleagues at the COLLAPS, CRIS and VITO setups. Each and every one of you has taught me something new and provided warm memories. Some people that stand out: Rob, thanks for putting problems "over there, with the rest of the fire" and applying "a bit of leverage" together with me! Mark, I learned a lot from you and our discussions on physics, math, programming and (oddly enough) cars and life in general. Frank, without you I would know a *lot* less about Prussia than I do now. Hanne, just thanks for everything!

I spent the last four years having fun with everyone working at IKS. The "Moete train" always provided some quirky topics of conversation during lunchtime, which was a welcome distraction from work. Thanks to my office mates (Hanne, Ivan, Xu, Kristof, João and Tassos) for putting up with my chatting and making the office a fun place! My fellow finishing PhD's (Greg, Tiago, Leendert, Peter and Koen), thanks for all the opportunities for a drink after work and *goeie babbels* during work.

Natuurlijk kan het IKS niet zonder de ruimere groep die steun biedt. Isabelle, Danielle and Fabienne, bedankt voor alle hulp met administratie allerhande. Bert en Luc, de IT ondersteuning is verschillende keren van onschatbare waarde gebleken. Willy, Johan, Bart en Dries, bedankt voor het overlopen van alle zotte technische tekeningen die ik produceerde en mij te helpen met het ontwerpen van realistische stukken.

Buiten de werkcollega's zijn er uiteraard ook de andere vriendenkringen die het leven interessant houden. Beide D&D groepen waaraan ik meedoe, bedankt voor de fijne avonden *waarop het zachtjes begon te regenen.* Ook een speciale vermelding voor Jonas, Anne, en Roxanne voor de movie nights, en het spelen van klankbord als ik iets kwijt moest.

Last but certainly not least, de familie. In september 2009 begon ik aan een avontuur in Leuven dat nu afloopt. Mama, papa, Geert, Anja en Yaro, bedankt voor de steun, de aanmoediging en het vertrouwen bij het begin, midden en einde ervan!

– Wouter

Abstract

This thesis presents measurements of the β-asymmetry of spin-polarized ^{35}Ar in different crystals and at different temperatures. These measurements form the first phase of a research project that aims to measure the β-asymmetry parameter with a relative precision of 0.5%. This thesis details technical developments of the beamline that was used to perform these measurements. These developments include the specifications of the different magnetic fields and the design and construction of the different coils. Furthermore, an existing rate equation formalism was extended to permit calculations in which an arbitrary number of levels and lasers interact. This extension proved crucial in optimizing the optical pumping scheme of ^{35}Ar and achieving maximal polarization. In addition, analysis techniques based on likelihood maximization and Monte-Carlo probing of the resulting parameter distribution are described here.

After implantation of the spin-polarized ^{35}Ar into NaCl and KCl crystals, an asymmetry of about 1% was observed at low temperatures (≈10 K). This result is compared to theoretical estimates, taking into account several processes which were investigated numerically: the optical pumping process, the rotation of the nuclear spin in the applied magnetic fields and the path of the emitted β-particles. This comparison shows agreement between the data and the simulations, although the β-scattering simulations do come with a large systematic uncertainty due to problems with the simulation software. Furthermore, a growth of an impurity layer on top of the crystal surface limited the measurement time: after every hour of measurement, a two hour cycle of heating and cooling down the crystal was required to restore the observed signal.

Future technical challenges will include improving the vacuum system to extend the useful measurement time at cryogenic temperatures and reduce the duty cycle loss. Preliminary studies of techniques to enhance the polarization indicate that laser re-ionization would be most interesting for the future of the project.

Beknopte samenvatting

In deze thesis wordt de β-asymmetrie van spin-gepolariseerd ^{35}Ar in verschillende kristallen en bij verschillende temperaturen opgemeten. Deze metingen vormen de eerste fase van een onderzoeksproject met als doel het bepalen van de β-asymmetrie-parameter met een relatieve precisie van 0.5%. Een eerste deel van de thesis bespreekt het technisch ontwerp van de bundellijn waar deze metingen zijn uitgevoerd. De beschrijving bevat de specificaties van de verschillende magnetische velden en het ontwerp van de spoelen. In een tweede deel van de thesis is het bestaand formalisme van *rate equations* uitgebreid om een systeem van een arbitrair aantal interagerende energieniveau's en lasers theoretisch te beschrijven. Deze extensie was cruciaal in het optimaliseren van lasersysteem voor het optisch pompen van ^{35}Ar om de kernpolarisatie te maximaliseren. De aangewende analysetechnieken, die gebruik maken van *likelihood-maximalisatie* en de daaruitvolgende parameter-distributies opmeten door Monte Carlo-technieken, worden ook besproken in deze thesis.

Bij het implanteren van spin-gepolariseerd ^{35}Ar in NaCl- en KCl-kristallen, is een asymmetrie van ongeveer 1% opgemeten bij lage temperaturen ($\approx 10\,\mathrm{K}$). Dit resultaat is vergeleken met theoretische verwachtingen, waar met verschillende processen numeriek rekening is gehouden: (*i*) het optisch pompen, (*ii*) de rotatie van de kernspin in de magnetische velden van de opstelling, en (*iii*) het pad van de uitgestraalde β-deeltjes. Deze vergelijking toont een goede overeenkomst tussen de experimentele resultaten en de theoretische simulaties, hoewel er een significante onzekerheid verbonden is aan de β-verstrooingssimulaties omwille van de gebruikte software. Bovendien is ook vastgesteld dat een opbouw van onzuiverheden op het oppervlak van de NaCl- en KCl-kristallen de beschikbare meettijd beperkt: na elk uur meten aan koude temperaturen moet een cyclus van opwarmen en afkoelen met een totale duur van twee uur doorlopen worden om opnieuw dezelfde asymmetrie te observeren.

Verdere technische uitdagingen zijn o.a. het verbeteren van het vacuüm in de opstelling, waardoor de opbouw van onzuiverheden wordt vertraagd. Hierdoor zal langer gemeten kunnen worden vooraleer een opwarm-afkoel cyclus vereist is. Een preliminaire vergelijking van verscheidene technieken om het meetsignaal te verbeteren duidt het gebruik van lasers om de bundel opnieuw te ioniseren aan als de meest interessante optie.

Contents

List of Figures

List of Tables

Chapter 0

Preface

This thesis describes the development of a laser-polarization beamline located at ISOLDE, CERN and the results of polarization tests of ^{35}Ar performed using the same beamline. The beamline has been designed with the goal of producing polarized beams using the technique of *optical pumping*, where circularly polarized lasers are used to spin-polarize the nuclear spin of an ensemble via resonant excitation. Such a spin-polarized ensemble, when it undergoes β-decay, will emit the β-particles with an asymmetric angular distribution. After spin-polarizing radioactive ^{35}Ar atoms using the optical pumping technique, the subsequent β-decay asymmetry was measured as a function of time and for different laser frequencies. These measurements were used to characterize the optical pumping process. Simulations of the spin-rotation and scattering of the β-particles were performed in order to investigate the experimental measurements more deeply. The result of these measurements and simulations provide input for further phases of the project. This project aims to perform high precision measurements for weak interaction studies.

This work is divided into seven chapters. Chapter 1 provides the context and motivation to build a beamline dedicated to laser-polarizing a nuclear ensemble and the role of β-asymmetry measurements of ^{35}Ar could play in the context of weak interaction studies. Chapter 2 presents a theoretical framework for performing rate equations to calculate nuclear spin-polarization as a result of laser-particle interactions, which were generalized to an arbitrary amount of levels and lasers for the first time. A description of the application of these rate equations to the ^{35}Ar system is provided in Chapter 3. The optimal laser scheme, which uses multi-frequency pumping, is deduced. The impact of the fragmentation of the population across the atomic levels of Ar after charge exchange between the Ar ions and a K vapor is also studied in this chapter.

Chapter 4 details the experimental setup, including the development of the beamline as well as additional considerations on the magnetic fields and a crystal holder that was cooled to cryogenic temperatures. This is mainly done through the inclusion of an article, which gives an overview of the new beamline including beam transport simulations and results on the newly applied multi-frequency pumping technique:

- **A new beamline for laser spin-polarization at ISOLDE**
  W. Gins, R. D. Harding, M. Baranowski, M. L. Bissell, R. F. Garcia Ruiz, M. Kowalska, G. Neyens, S. Pallada, N. Severijns, Ph. Velten, F. Wienholtz, Z. Y. Xu, X. F. Yang, D. Zakoucky
  Submitted to *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*

The details of the experimental conditions are given in Chapter 5, where the method of extracting the β-asymmetry and relaxation times in different crystals are also discussed. Included in this chapter is the analysis method, presented in the article detailing the development of the SATLAS analysis package which was used throughout the thesis:

- **Analysis of counting data: Development of the SATLAS Python package**
  W. Gins, R. P. de Groote, M. L. Bissell, C. Granados Buitrago, R. Ferrer, K. M. Lynch, G. Neyens, S. Sels
  Computer Physics Communications 222 (2018) 286-294

The measured β-asymmetry is studied in more detail in Chapter 6, where additional simulations of the spin-rotation and the scattering of β-particles are also given. Chapter 7 concludes this thesis and summarizes the observed β-asymmetry behavior of ^{35}Ar and gives an outlook for the future of the project.

Chapter 1

Introduction

Time has come and we must be on our way
—Twilight Force, *To The Stars*

As nuclear physics research grows, more specialized measurement techniques are continuously developed. This leads to research facilities installing permanent setups in order to perform specific types of measurements. The chapter describes the motivation behind both the development of a permanent beamline dedicated to laser-polarization at ISOLDE, CERN is given, as well as the β-asymmetry measurements on ^{35}Ar.

1.1 Laser-polarization beamline

ISOLDE (Isotope Separator OnLine DEvice) is a facility, located in the CERN Meyrin site, where ion beams of exotic radioactive species have been produced since 1967 [1]. These radioactive beams have allowed studies ranging from nuclear structure [2, 3] and nuclear astrophysics [4] to fundamental interactions [5] and solid state physics [6]. This wide range of research topics can be covered due to the availability of more than 1300 radioactive species at ISOLDE at energies ranging from 30 keV up to 10 MeV/u, and production of new isotopes is an ever ongoing research topic [7].

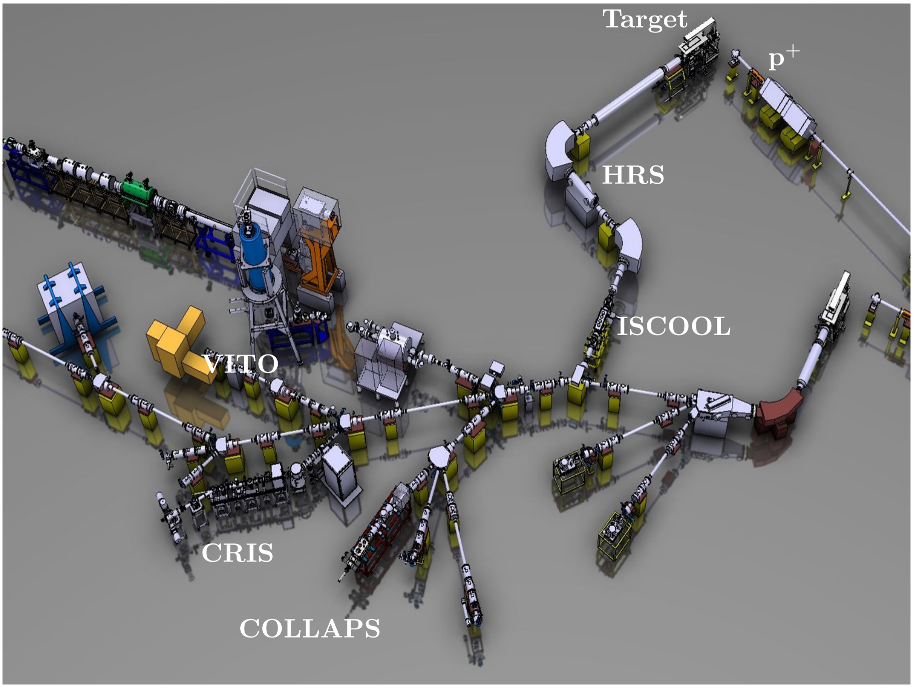


Figure 1.1: ISOLDE layout, with the location of the collinear laser spectroscopy beamlines indicated. Also indicated are the proton line (p^+), the target, the High Resolution Separator (HRS) and the ISOLDE cooler-buncher (ISCOOL). Figure taken from the ISOLDE website.

The produced radio-isotopes can be guided to different measurement stations and studied by applying various techniques, several of which rely on polarizing the nuclear spin. When a nuclear spin-polarized ensemble decays, the β-particle has an angular distribution (relative to an orientation axis) of [8]

$$W(\theta) \sim 1 + A\frac{\langle I\rangle}{I}\frac{p_e c}{E_e}\cos(\theta) = 1 + AP\frac{v}{c}\cos(\theta), \qquad (1.1)$$

where I is the nuclear spin, $\langle I\rangle / I$ is the nuclear polarization P, A is the β-asymmetry parameter and v is the velocity of the emitted β-particle. Several setups have made use of nuclear spin-polarized beams to perform measurements, ranging from semi-temporary beamline modifications to permanent setups. In order to generate the spin-polarization, methods such as optical pumping [9], low temperature orientation [10] and passing the beam through a tilted foil

[11] have all been used at ISOLDE. Of these techniques, optical pumping has been used in a temporary setup at the COLLAPS beamline in order to perform β-NMR measurements, allowing the study of the nuclear moments of exotic radioactive isotopes [9, 12, 13]. When an optical pumping cycle is available in an atomic or ionic system, the advantage of optical pumping is the high degree of polarization that can be achieved, which is typically higher than can be generated by passage through tilted foils [14].

As spin-polarized beams can be used in nuclear structure research [15], fundamental symmetries [16] and biomedical/biochemical research on liquid samples [17, 18], the demand for polarized beams is very high and construction of a dedicated beamline was suggested as the Versatile Ion Techniques Online (VITO) beamline [19, 20] (location in the experimental hall given in Figure 1.1). One of the proposed projects at this new beamline focuses on a high precision measurement of the β-asymmetry parameter of ^{35}Ar for fundamental studies [21]. This thesis investigates the feasibility of this experiment and specifically focuses on the necessary conditions related to the production of a polarized ensemble of ^{35}Ar atoms and how long this polarization can be maintained after implantation in a suitable host crystal.

1.2 β-asymmetry of ^{35}Ar

One of the tests of the Standard Model is the probing of the unitarity of the Cabibbo-Kobayashi-Maskawa matrix, which describes the mixing of the quark states. One of the main elements in this matrix is the mixing between the up and down quark, described by the matrix element V_{ud}. This matrix element is typically deduced from ft-values from various types of β-decay, of which the superallowed Fermi decays give the most accurate determination [22]. One of these types is β-decay of mirror nuclei, where the number of protons and neutrons are switched after decay. Based on the superallowed Fermi decays, V_{ud} is evaluated to be [23]

$$|V_{ud}| = 0.97420(21). \tag{1.2}$$

Combining this result with the accepted result for $|V_{us}| = 0.2243(5)$ for the mixing with the strange quark ($|V_{ub}|$ for the mixing with the bottom quark is negligibly small), the unitarity test results in $|V_{ud}|^2 + |V_{us}|^2 = 0.9994(5)$ which is consistent with the expected value of 1. The uncertainty on this test is dominated by the uncertainty on $|V_{ud}|^2$, which is a factor 2 larger than the uncertainty contribution of $|V_{us}|^2$.

The extraction of V_{ud} from mirror nuclei requires a measurement of either the β-asymmetry parameter A or the β-ν correlation parameter a. The latter of these parameters describes the angular distribution between the emitted positron and neutrino in the same way that A describes the angular distribution of the emitted positron and the nuclear spin direction (pointed along an applied static magnetic field). Previous measurements of the β-asymmetry parameter of ^{35}Ar [24, 25] used polarized protons impacting a gaseous target of Cl_2, producing polarized ^{35}Ar isotopes via a $^{35}\mathrm{Cl}(p,n)$ reaction. From these studies, an asymmetry parameter of respectively 0.49(10) and 0.427(23) was measured, with the same technique as will be described in the following subsection, namely β-γ coincidence measurements. A recent survey [26] shows that improving the precision on this β-asymmetry parameter could result in one of the most sensitive single measurements of $|V_{ud}|$ to date. To this end, a proposal was written to carry out high-precision β-asymmetry measurements on ^{35}Ar isotopes at ISOLDE [21].

In what follows, the measurement principle as given in the proposal will be slightly expanded. Furthermore, the results of a preliminary study on the experimental requirements will be discussed.

1.2.1 Measurement principle

In $\beta^{\mp}$-decay, the emitted electron/positron and antineutrino/neutrino can have their spins either parallel to each other or anti-parallel. Therefore, transitions proceed via a combination of two different types of decay [27]:

1. Fermi decay, where the emitted electron and neutrino have anti-parallel spin vectors. This implies the total angular momentum of the nucleus cannot change:
$$\Delta I = 0 \tag{1.3}$$
2. Gamow-Teller decay, where the emitted electron and neutrino have parallel spin vectors. Here, the total angular momentum of the nucleus can change by a maximum of 1:
$$\Delta I = 0, \pm 1 \tag{1.4}$$

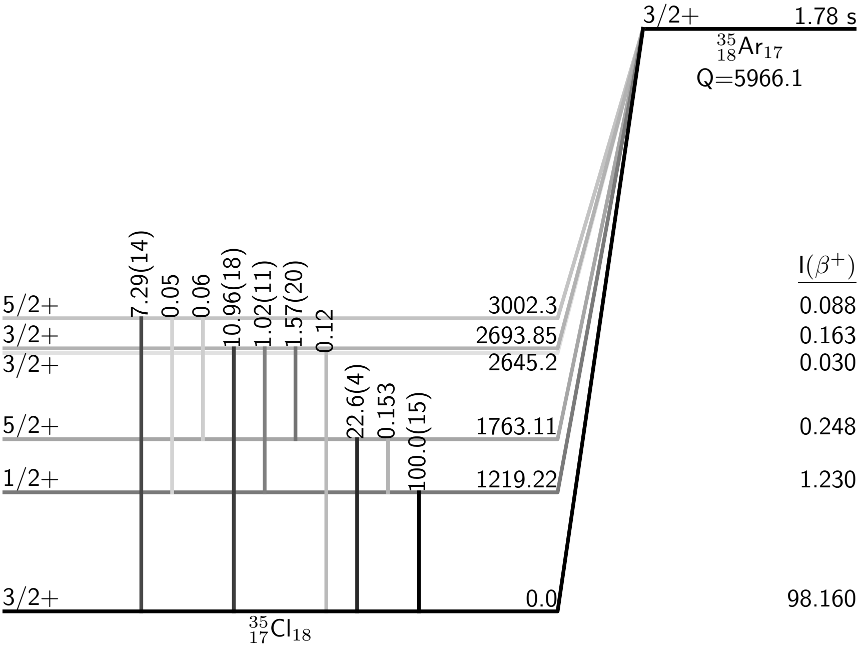


Figure 1.2: Decay scheme of ^{35}Ar, decay branches with a branching ratio $< 0.01\%$ have been omitted for clarity. Darker levels/transitions correspond to a greater branching ratio/γ intensity. Data obtained from Ref. [28].

In addition, Gamow-Teller decay cannot be present in a $0^+ \rightarrow 0^+$ transition.

As shown in Figure 1.2, the decay of ^{35}Ar to ^{35}Cl proceeds mainly via β^+ decay to the ground state (98.16% branching ratio) and the first excited state (1.23% branching ratio). Electron capture decay is also present, but is on the order of 0.07% and 0.002% respectively. Taking the nuclear spins of these states into account, the decay to the first excited state can only proceed via Gamow-Teller decay, since $I_f - I_i = 1/2 - 3/2 = -1$. As the ground state of ^{35}Cl has the same spin as the ground state of ^{35}Ar, this decay will be a mix of Fermi and Gamow-Teller decay. As explained in the following section, it is the mixing ratio between the Gamow-Teller and Fermi decays which is used to calculate V_{ud}. The mixing ratio can be accessed by measurement of the β-asymmetry A, as introduced in Eq. (1.1).

The decay scheme also shows that decay to the first excited state is followed by the emission of a 1219.22 keV γ. By performing a β-γ coincidence measurement, this γ-line can be used to identify β-decays to the excited state and hence allows measurement of the separate β-asymmetries of the two branches. The ratio of the asymmetries is given by the ratio of the experimental asymmetries $\mathcal{A}$:

$$\frac{A_{gs}}{A_{ex}} = \frac{\left\langle \frac{v}{c} \cos\left(\theta\right) \right\rangle_{ex} \mathcal{A}_{gs}}{\left\langle \frac{v}{c} \cos\left(\theta\right) \right\rangle_{gs} \mathcal{A}_{ex}}, \tag{1.5}$$

where $\left\langle \frac{v}{c} \cos\left(\theta\right) \right\rangle$ is a correction factor for the phase space and solid angle. By measuring the ratio, the absolute polarization P which occurs in both the ground and excited state asymmetry terms can be factored out of the equation. Additionally, the ratio of the correction factors is easier to calculate to high precision than the absolute correction factors, reducing systematic uncertainties.

1.2.2 β-decay asymmetry to V_{ud}

The mixing ratio of the decay is the ratio of the Gamow-Teller to the Fermi matrix element of the decay [22]:

$$\rho = \frac{C_A M_{GT}}{C_V M_F}. \tag{1.6}$$

This mixing ratio has an impact on the associated asymmetry parameter. Assuming no Beyond Standard Model physics, the asymmetry parameter A_{SM} is given by [26]

$$A_{SM}^{\beta\mp} = \frac{\mp\lambda_{I_f,I_i}\rho^2 - 2\delta_{I_f,I_i}\sqrt{\frac{I_i}{I_i+1}}\rho}{1+\rho^2}, \tag{1.7}$$

with

$$\lambda_{I_f,I_i} = \begin{cases} 1 & \text{for } I_f - I_i = -1 \\ \frac{1}{I_i+1} & \text{for } I_f - I_i = 0 \\ -\frac{I_i}{I_i+1} & \text{for } I_f - I_i = 1 \end{cases}. \tag{1.8}$$

In the case of a pure Fermi decay ($\rho \to 0$) the asymmetry parameter is 0, while for pure Gamow-Teller decay ($\rho \to \infty$) the asymmetry parameter is given by $\lambda_{I_f I_i}$. As the decay of ^{35}Ar to the first excited state of ^{35}Cl is a pure Gamow-Teller decay, the associated asymmetry parameter is $\lambda_{1/2,3/2} = 1$. Consequently, the β-asymmetry parameter of the decay to the ground state can be extracted from the measured ratio of experimental asymmetries using Eq. (1.5).

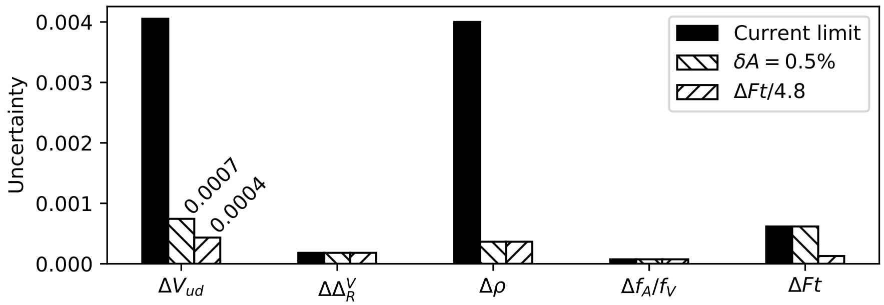


Figure 1.3: The error budget on the extracted value of V_{ud} from the β-asymmetry parameter of ^{35}Ar, along with updated budgets that could be achieved with improved measurements.

Once the value of the β-asymmetry of the ground state decay branch is measured, Eq. (1.7) can be inverted numerically to yield the mixing ratio of the decay. The literature value for the β-asymmetry parameter of ^{35}Ar (dominated by the ground state decay) is 0.430(22), from which a mixing ratio of −0.279(16) is calculated [29]. This mixing ratio can then be used to calculate the corrected ft-value [26]

$$\mathcal{F}t_0 \equiv \mathcal{F}t\left(1+\frac{f_A}{f_V}\rho^2\right) = \frac{K}{G_F^2 V_{ud}^2\left(1+\Delta_R^V\right)}, \tag{1.9}$$

where f_A/f_V is a statistical rate correction, K is a constant and G is the Fermi coupling constant, and Δ_R^V is a transition-independent radiative correction with a value of 2.361(38)% [26].

This equation can then be used to extract $|V_{ud}|$ and calculate the contribution of each experimental value to the uncertainty. Figure 1.3 shows the uncertainty on V_{ud} calculated from the ^{35}Ar mirror decay using the currently accepted values (black bars), assuming a new measurement of the β-asymmetry (top left-bottom

Table 1.1: Values used in calculation of the V_{ud} error budget. Data for the current limit taken from [29].

	f_A/f_V [−]	$\mathcal{F}t$ [s]	ρ [−]	$G/(\hbar c)^3$ [GeV^{-2}]	$K/(\hbar c)^6$ [GeV^{-4}s]
Current limit	0.9894(21)	5688.6(72)	−0.279(16)	$1.166\,37(1)\times10^{-5}$	$8120.278(4)\times10^{-10}$
$\delta A = 0.5\%$	-	-	−0.2790(16)	-	-
$\Delta\mathcal{F}t/4.8$	-	5688.6(15)	-	-	-

right hashes), and with an additional improved measurement of the $\mathcal{F}t$ value (bottom left-top right hashes). The values used are given in Table 1.1.

The current uncertainty is dominated by the mixing ratio, which can be reduced to below the contribution of the $\mathcal{F}t$ value by measuring the β-asymmetry with a relative precision of 0.5%. A further improvement of the $\mathcal{F}t$ precision by a factor 4.8 would result in an uncertainty on V_{ud} of 4×10^{-4} [26]. This error is within a factor 2 of the current uncertainty on V_{ud}.

1.2.3 Required experimental conditions

A report has been written by Philippe Velten [30] detailing preliminary estimates of the experimental requirements to measure the asymmetry parameter to a relative precision of 0.5%. The main points raised in this document of relevance for this thesis will be summarized here.

In this report, the time (in seconds) estimated to reach a certain statistical precision on $\mathcal{A}_{ex}$ is deduced to be

$$t = \frac{4}{I_{dcy}\epsilon_{coinc}} \left(1 + \frac{2}{SB}\right) \frac{R}{(R-1)^2} \left(\frac{\Delta\mathcal{A}_{ex}}{\mathcal{A}_{ex}}\right)^{-2} \tag{1.10}$$

where I_{dcy} is the implanted activity, ϵ_{coinc} is the coincident β-γ detection efficiency, SB is the signal-to-background ratio and R is given by $\frac{-1-\langle\cos(\theta)v/c\rangle P A_{ex}}{\langle\cos(\theta)v/c\rangle P A_{ex}-1}$.

In this expression, ϵ_{coinc}, SB and $\langle\cos(\theta)\, v/c\rangle$ have been evaluated for a preliminary setup geometry using Monte-Carlo calculations in FLUKA. Values of respectively 2.3×10^{-5}, 2.25 and 0.8036 were found and used in this expression. The remaining parameters are then the activity I_{dcy} and the nuclear polarization P. The measurement times calculated for two different polarizations and

Table 1.2: Parameters and estimated measurement time for two different estimates of experimental conditions. Data taken from Ref. [30].

Parameter	Situation 1	Situation 2
I_{dcy}	1×10^6	0.5×10^6
P	30%	20%
t	16 hours	3 days

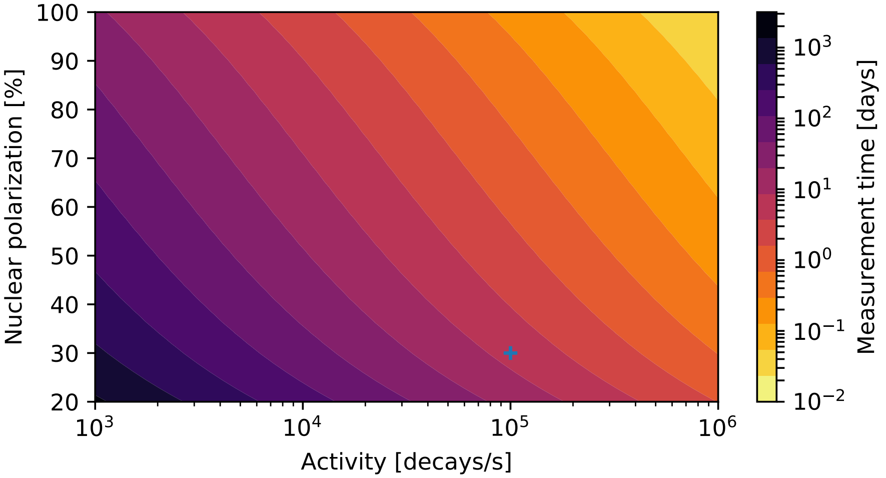


Figure 1.4: Estimated measurement time with the setup as described in Ref. [21] as a function of activity and nuclear polarization.

activities are given in Table 1.2. The impact of the polarization here is clearly seen, with even a decrease from 30% to 20% increasing the measurement time by almost a factor 3. The remaining factor of 2 is caused by the decrease in activity. Figure 1.4 shows the estimated measurement time to reach 0.5% relative precision as a function of the activity and the nuclear polarization.

As shown in both Table 1.2 and Figure 1.4, the nuclear polarization will be the key element in determining the measurement time needed to achieve the relative precision of 0.5%. Indeed, if an ensemble polarization of 30% could be reached, then with an activity of $1 \times 10^5\,\mathrm{s}^{-1}$ (a typical production rate at ISOLDE for ^{35}Ar), an experiment of 5-10 days would allow to reach the required precision (indicated with a plus on Figure 1.4). This thesis will therefore focus on the first phase of the project, where the optical pumping scheme for polarizing ^{35}Ar is characterized and crystal hosts are tested in order to deliver an ensemble with both maximal polarization and relaxation time.

1.3 Experimental overview

The first phase of the experiment, as stated above, will characterize the optimal optical pumping scheme to polarize a ^{35}Ar atom beam and measurements will be performed to identify the most suitable crystal host for maintaining the ^{35}Ar ensemble polarization after implantation. A short overview of the employed experimental method is given here to provide a reference frame for the following chapters.

The experiment will employ the method of optical pumping to spin-polarize the beam of ^{35}Ar. The only available closed cycle in the fine structure of Ar, required for performing efficient optical pumping, is found in atomic Ar; therefore, the ion beam will need to be neutralized via a charge exchange process. Due to the high level density in the fine structure of atomic Ar, the final population of Ar atoms will be distributed among several levels and this imposes a first limitation of the maximum polarization that can be achieved in the ensemble. In addition, while the desired fine structure transition frequency is well known, the isotope shift of ^{35}Ar compared to the stable ^{34,36,40}Ar isotopes in this line has never been measured. The experiment will therefore start by measuring the isotope shifts of the three stable isotopes of Ar using the well-known method of fluorescence detection after resonance excitation [3]. Using this information, the isotope shift of ^{35}Ar can be predicted and the frequency range to scan for the hyperfine spectrum of ^{35}Ar deduced. For the latter isotope, the yield is too low to allow for an optical detection of the hyperfine spectrum. Only the more sensitive method of β-asymmetry using a polarized beam can be applied [13, 31].

The charge exchange process takes place in the Charge Exchange Cell (CEC, Figure 3 of Article I), after which the atom beam immediately interacts with the laser beam. If the frequency of the laser beam is at resonance, the optical pumping process starts. Previous experiments using atomic/ionic laser transitions involving a $J = 1/2$ level for laser polarization have utilized the formalism of rate equations very successfully to describe the measured β-asymmetry spectra as a function of laser frequency [9, 32]. However, this formalism needs to be extended to include decoupling of the nuclear and electron spins in the case where the electron spin is not $J = 1/2$. In the case of ^{35}Ar, where $J = 2$, the fine structure level in the optical pumping cycle splits in four hyperfine levels. Therefore, Acousto-Optic Modulators (AOM) will be used to excite multiple hyperfine transitions simultaneously. This will enhance the generated polarization in the ensemble, but simulations have to be performed to determine the gain. For these simulations, interaction with multiple frequencies has to be added to the rate equation formalism.

In contrast to previous experiments where the β-asymmetry parameter was determined by production (and stopping) of a polarized ^{35}Ar beam in a gas [24, 25], the spin-polarized beam will be implanted in a solid crystal host in this study. Implantation in a gas is not possible in our case, because of the low (50 keV) beam energy, which prevents the atoms from passing through a gas-cell window. The implantation crystals are chosen to have a cubic lattice structure, minimizing the impact of quadrupole interactions which lead to a breakdown of the spin-polarization [33]. Also, the spin-relaxation time has to be investigated, as this is crystal dependent and predictions of the relaxation time are in many cases impossible (except for metals where it is known that Korringa relaxation dominates [34]). Only one previous experiment was performed by implanting polarized Ar in a crystal host, namely KBr [35]. After cooling down the crystal below 50 K, a β-asymmetry of 1% was observed. The polarization was provided by the reaction mechanism and the beam energy was on the order of 3.5 GeV. In order to identify a crystal which allows maintaining the polarization, various crystals will be tested in this work, over a wide temperature range. For this, measurements of the spin-lattice relaxation time will be performed at different temperatures.

With the end goal being an accurate determination of the β-asymmetry parameter, the possible causes for loss of ensemble polarization, from the moment of laser-induced polarization production to the time of observing β-asymmetry after implantation, have to be evaluated carefully. Aside from the aforementioned population distribution after charge exchange, two other processes that can result in loss of polarization need to be evaluated: the rotation and decoupling of the (coupled) nuclear and electron spin in the magnetic fields of the setup, and the effect of scattered β particles in the setup on the observed asymmetry.

Chapter 2

Spin-polarization through optical pumping with laser light

Make use of a computation; Tap into the digits to write; A set of rules to form the universe
—EPICA, *The Cosmic Algorithm*

Spin-polarization can be achieved through different mechanisms, e.g. momentum selection of ion beams at fragment separator facilities [36] or using low temperature orientation [10]. One technique that can produce highly polarized ensembles is optical pumping using circularly polarized laser light. The laser-particle interaction can be modeled using rate equations, which will be developed in this chapter. This development requires a brief introduction into the hyperfine structure and its effect on the energy levels of the atomic fine structure.

2.1 Atomic fine and hyperfine structure

The energy structure of atomic systems can be explained by considering both the electron-electron and the electron-nucleus interaction. The inclusion of several relativistic effects leads to the splitting of the electron energy levels and is called the fine structure. Up to this point, the nucleus is considered to be a point charge and the charge density is represented as a delta function. Considering the nucleus as an extended, finite-size system leads to a further correction to the electron energy, leading to the hyperfine structure.

When expanding the operators associated with the hyperfine interaction [37], the parity of the operators can be used to reduce the number of non-zero contributions. The result is a Hamiltonian with the terms

$$\mathcal{H} = \mathcal{H}_{E0} + \mathcal{H}_{M1} + \mathcal{H}_{E2} + \ldots . \tag{2.1}$$

where E stands for the electric matrix elements and M for the magnetic matrix elements. Only the terms going up to the electric quadrupole moment have been included, since these are the leading order corrections needed to reproduce almost all known hyperfine spectra within typical experimental precision. For specific isotopes, the magnetic octupole moment has been measured and reported [38], but this term will not be treated here.

2.1.1 Isotope shift

The transition frequency between two levels is isotope dependent and this variation is called the isotope shift. In relation to a reference isotope A, the isotope shift of an isotope A' is calculated as

$$\delta\nu^{A,A'} = \nu_0^{A'} - \nu_0^{A}. \tag{2.2}$$

This shift is the result of two effects:

- Isotopes may have a different charge radius, so the overlap with the electron wave function will be slightly different. This contribution is called the *field shift*.
- The isotopes differ in mass, modifying the center of mass about which the nucleus and electron cloud orbit, resulting in a slight energy difference. This shift is called the *mass shift*.

Combining these contributions, the isotope shift of a specific transition i is given by

$$\delta\nu_i^{A,A'} = F_i \delta \left\langle r^2 \right\rangle^{A,A'} + M_i \frac{m_{A'} - m_A}{m_{A'} m_A}, \tag{2.3}$$

where F_i and M_i are respectively the field and mass shift factors, $\delta \left\langle r^2 \right\rangle^{A,A'}$ is the change in mean square charge radius and m is the mass of the isotope.

2.1.2 Energy splittings

The main contribution to the hyperfine splitting is generated by the interaction between the magnetic field generated by the electron cloud and the magnetic dipole moment of the nucleus. The associated term in the Hamiltonian is given by

$$\mathcal{H}_{M1} = -\int_V \boldsymbol{B} \mathrm{d}\boldsymbol{\mu}. \tag{2.4}$$

Assuming a constant magnetic field over the nuclear volume results in

$$\mathcal{H}_{M1} = -\boldsymbol{\mu} \cdot \boldsymbol{B}, \tag{2.5}$$

which can be expanded to

$$\mathcal{H}_{M1} = -\frac{\mu B_0}{\hbar^2 I J} \boldsymbol{I} \cdot \boldsymbol{J}, \tag{2.6}$$

where $\boldsymbol{I}$ is the nuclear spin, $\boldsymbol{J}$ the spin associated with the electron cloud and μ and B_0 the nuclear magnetic moment and magnetic field generated by the electrons respectively. When no external magnetic field is present, the nuclear and electron spin couple together to the total atomic spin $\boldsymbol{F} = \boldsymbol{I} + \boldsymbol{J}$. Evaluation of the energy change associated with this term, converted to a frequency, gives [39]

$$\Delta\nu_{M1} = -\frac{1}{2} \overbrace{\frac{\mu B_0}{h I J}}^{A} \overbrace{\left(F\left(F+1\right) - I\left(I+1\right) - J\left(J+1\right)\right)}^{K}. \tag{2.7}$$

The energy difference is thus defined in terms of the hyperfine A parameter.

The next term in the nuclear-electronic interaction Hamiltonian is the quadrupole interaction term. This is an interaction between the spectroscopic quadrupole moment of the nucleus and the electric field gradient induced by the electron cloud. For this interaction to occur, both the nuclear spin $\boldsymbol{I}$ and the electron spin $\boldsymbol{J}$ have to be at least 1. The resulting shift in frequency of a specific hyperfine level F is given by

$$\Delta\nu_{E2} = -\overbrace{eQV_{zz}}^{B}\frac{3K\left(K+1\right)-4I\left(I+1\right)J\left(J+1\right)}{8I\left(2I-1\right)J\left(2J-1\right)}, \tag{2.8}$$

where the electron charge e, the spectroscopic quadrupole moment Q and the electric field gradient V_{zz} form the hyperfine B parameter.

2.1.3 Combined systems

Combining both the magnetic dipole and electric quadrupole splitting, each hyperfine level F undergoes a shift given by

$$\Delta\nu\left(J,F\right) = -\frac{AK}{2} - B\frac{3K\left(K+1\right)-4I\left(I+1\right)J\left(J+1\right)}{8I\left(2I-1\right)J\left(2J-1\right)}. \tag{2.9}$$

This results in a modified transition frequency between two hyperfine levels (F_j and F_i) of two electronic fine structure levels i and j (with electron spins J_j and J_i). Starting from the fine structure transition frequency ν_0 and taking the isotope shift into account, the transition frequency between two levels is given by

$$\nu_{F_i,F_j} = \nu_0 + \Delta\nu\left(A_j,B_j,I,J_j,F_j\right) - \Delta\nu\left(A_i,B_i,I,J_i,F_i\right) + \delta\nu_{i,j}^{A,A'}, \tag{2.10}$$

where j/i is the higher/lower hyperfine level. The strength of the transition is given by the transition coefficient, which will be calculated in Section 2.2.3.

The measurement of these transition frequencies allows the extraction of the nuclear moments that appear in the A and B factors. This is independent of any nuclear model, but it does require knowledge of the magnetic hyperfine field B that appears in the A-factor and the electric field gradient V_{zz} which appears in the B-factor. These magnetic and electric field contributions are different for the upper and lower levels. Although a very prominent topic in current nuclear structure research, this thesis will not extract nuclear information from the hyperfine spectrum. Rather, the nuclear moments and hyperfine splittings are already known and will be used in the rate equations to predict the hyperfine spectrum.

2.2 Rate equations for optical pumping

The term *rate equations* covers a broad range of models, mostly connected with describing the dynamics of (chemical) reactions [40]. The mathematical principles behind the formulation of these equations can also be applied to laser-particle interactions. This provides a classical model in the form of a set of differential equations. As this idea can be traced back to Albert Einstein [41], the model and coefficients are called the Einstein model and Einstein coefficients.

Based on the population distribution, the rate equations can be used to calculate either the nuclear polarization or the photon emittance as a function of applied laser frequency. Both possibilities are discussed at the end of this section.

2.2.1 Einstein model

The derivation of the equations is based on the work found in Refs. [42, 43].

The Einstein model describes the processes in a multi-level system interacting with coherent radiation of frequency ν. In this section, we will work in a three level system to more easily show the generalization. There are three interactions to be accounted for:

1. Spontaneous decay (coefficient denoted with A).
2. Excitation (coefficient denoted with B).
3. Stimulated emission (also denoted with B).

Of these phenomena, only the first is independent of the presence of, and interaction with, photons. Both excitation and stimulated emission are driven by the presence of the photon field.

Consider a generic three level system as presented in Figure 2.1. We assume that the probability for an atom in one of the three states to absorb or emit a photon is proportional to the population of those states and the photon density. In this case, the differential equations are written as

$$\frac{\mathrm{d}N}{\mathrm{d}t} = \mathrm{d}\begin{bmatrix}N_f\\N_j\\N_i\end{bmatrix}\Big/\mathrm{d}t = A\cdot N + B\rho\left(\nu\right)\cdot N, \tag{2.11}$$

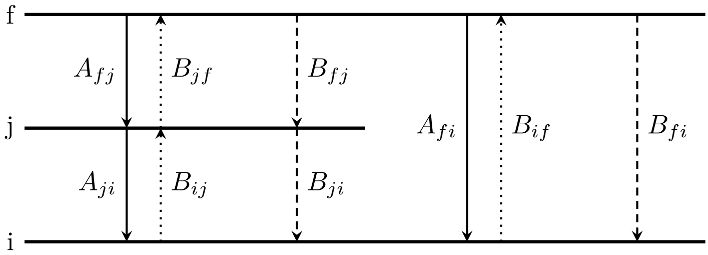


Figure 2.1: Generic three level system with levels i, j and f. The spontaneous decay (solid), excitation (dotted) and stimulated emission (dashed) paths are indicated.

where

$$A = \begin{bmatrix} -(A_{fi} + A_{fj}) & 0 & 0 \\ A_{fj} & -A_{ji} & 0 \\ A_{fi} & A_{ji} & 0 \end{bmatrix}, \tag{2.12}$$

$$B\rho\left(\nu\right) = \begin{bmatrix} -\left(B_{fj} + B_{fi}\right) & B_{jf} & B_{if} \\ B_{fj} & -\left(B_{ji} + B_{jf}\right) & B_{ij} \\ B_{fi} & B_{ji} & -\left(B_{ij} + B_{if}\right) \end{bmatrix} \rho\left(\nu\right). \tag{2.13}$$

The matrix A contains the spontaneous decay dynamics, B contains the probability of stimulated emission and laser excitation and $\rho\left(\nu\right)$ represents the spectral energy density for a specific frequency ν.

The Einstein A-coefficients are experimentally measured values and are known as the transition strengths. They are given in units of s^{-1} and are also related to the natural linewidth Γ of the transition via the Heisenberg uncertainty relation:

$$\Delta E \Delta t = \hbar \rightarrow \Gamma = \frac{\Delta E}{h} = \frac{1}{2\pi\Delta t} = \frac{A}{2\pi} \tag{2.14}$$

where the lifetime Δt of the excited state is given by the inverse of the Einstein A-coefficient.

For the calculation of $B\rho\left(\nu\right)$, consider first the B-coefficients. Assuming time-reversal symmetry, we see that the coefficient for stimulated emission should be the same as for excitation for non-degenerate levels. As we will continue to work with non-degenerate levels, we can assume the equality

$$B_{if} = B_{fi}. \tag{2.15}$$

To rewrite $B\rho\left(\nu\right)$, consider the interaction cross section $\sigma\left(\nu\right)$. This can be defined as the absorbed energy divided by the total incoming energy density I:

$$\sigma\left(\nu\right) = \frac{B_{ij}\rho\left(\nu\right)h\nu}{I}. \tag{2.16}$$

with I the energy density given in W/m^2. Ref. [42, pages 137-140] describes how an expression for the absorption cross section can be derived from the optical Bloch equations and linked to the Einstein A-parameter. The result is

$$\sigma\left(\nu\right) = \frac{A_{ji}c^2}{8\pi\nu_0^2}\mathcal{L}\left(\nu\right), \tag{2.17}$$

with $\mathcal{L}\left(\nu\right)$ the frequency dependent line shape function. In the ideal case, this is a Lorentzian profile with the natural linewidth Γ as Full Width Half Maximum (FWHM):

$$\mathcal{L}\left(\nu\right) = \frac{1}{\pi}\frac{\Gamma/2}{\left(\nu-\nu_0\right)^2 + \left(\Gamma/2\right)^2}. \tag{2.18}$$

The lineshape is centered at ν_o, the difference in energy between levels i and j converted to frequency. The full expression is then

$$B_{ij}\rho\left(\nu\right) = \frac{I\sigma\left(\nu\right)}{h\nu} = \frac{A_{ij}Ic^2}{8\pi h\nu_0^2\nu}\mathcal{L}\left(\nu\right). \tag{2.19}$$

The Lorentzian function $\mathcal{L}$ can be replaced to account for different broadening mechanisms. The only requirement is that

$$\int_{-\infty}^{+\infty}\mathcal{L}\left(\nu\right)\mathrm{d}\nu = 1. \tag{2.20}$$

In order to incorporate broadenings with a Lorentzian and Gaussian result, the lineshape is exchanged for a Voigt profile, which is defined as the convolution of these profiles and more easily numerically evaluated with the Faddeeva function [44, Eq. (7.4.13)]:

$$\mathcal{V}\left(\nu;\nu_i,\sigma,\gamma\right) = \int_{-\infty}^{+\infty}\mathcal{G}\left(\nu';\nu_i,\sigma\right)\mathcal{L}\left(\nu-\nu';\nu_i,\gamma\right)\mathrm{d}\nu' = \frac{\mathrm{Re}\left[w\left(z\right)\right]}{\sigma\sqrt{2\pi}} \tag{2.21}$$

where $\mathrm{Re}\left[w\left(z\right)\right]$ is the real part of the Faddeeva function with the argument $z = \dfrac{\nu - \nu_i + \mathrm{i}\gamma}{\sigma\sqrt{2}}$.

Inserting these relations in Eq. (2.11) gives the time dynamics for a three level system.

2.2.2 Generalized n-level system

In order to go to an n-level system, it is sufficient to note that the sum of every column of A and $B\rho\left(\nu\right)$ is zero. The diagonal elements can therefore be written as the sum of all the other elements in that column, with a minus sign. Writing down the generalization of the matrices results in

$$A = \begin{bmatrix} -\sum_{j<n} A_{nj} & 0 & \cdots & 0 & 0 \\ A_{n(n-1)} & -\sum_{j<n-1} A_{(n-1)j} & & & \\ \vdots & & \ddots & \vdots & \vdots \\ \vdots & & & -A_{21} & \\ A_{n1} & \cdots & \cdots & A_{21} & 0 \end{bmatrix}, \tag{2.22}$$

$$B\rho\left(\nu\right) = \begin{bmatrix} -\sum_{j\neq n} B_{nj} & B_{(n-1)n} & \cdots & B_{1n} \\ B_{n(n-1)} & -\sum_{j\neq n-1} B_{(n-1)j} & & \vdots \\ \vdots & & \ddots & \vdots \\ B_{n1} & B_{(n-1)1} & \cdots & -\sum_{j\neq 1} B_{1j} \end{bmatrix} \rho\left(\nu\right), \tag{2.23}$$

where the levels have been numbered 1 to n in increasing energy.

Including interactions with more than one laser is done by including extra $B\rho\left(\nu\right)$ matrices. The total interaction matrix for m lasers is then

$$B\rho\left(\nu_1, \ldots, \nu_m\right) = \sum_{i=1\rightarrow m} B\rho\left(\nu_i\right). \tag{2.24}$$

The differential equation governing the time dynamics of the generalized system is still given by

$$\frac{\mathrm{d}N}{\mathrm{d}t} = A \cdot N + B\rho\left(\nu_1, \ldots, \nu_m\right) \cdot N. \tag{2.25}$$

2.2.3 Transition intensities

The experimentally measured A coefficients correspond to the total decay rate from all hyperfine sub-levels. The partial decay rate for each of the hyperfine sub-levels therefore has to be calculated. This is given by [45]:

$$\mathcal{A}_{F_jF_im_{F_j}m_{F_i}} = \mathcal{C}\nu^3 \left|\left\langle J_iIF_im_{F_i}\middle|\hat{D}_\mu\middle|J_jIF_jm_{F_j}\right\rangle\right|^2, \tag{2.26}$$

with $\hat{D}_\mu$ the electric dipole operator with polarization μ and $\mathcal{C} = 4\left(2\pi\right)^3 / \left(3c^3\hbar\right)$. This is the partial decay rate between levels j (upper) and i (lower), with total spin $\boldsymbol{F_j}$ and $\boldsymbol{F_i}$ respectively. The m quantum number refers to the z-component of this spin. Application of the Wigner-Eckart theorem [46]

$$\langle jm|T^k_q|j'm'\rangle = (-1)^{j-m}\begin{pmatrix} j' & j & k \\ m' & -m & q \end{pmatrix}\langle j||T^k||j'\rangle \tag{2.27}$$

results in

$$\begin{aligned}\left|\left\langle J_iIF_im_{F_i}\middle|\hat{D}_\mu\middle|J_jIF_jm_{F_j}\right\rangle\right|^2 = &\begin{pmatrix} F_i & F_j & 1 \\ -m_{F_i} & m_{F_j} & \mu \end{pmatrix}^2 \\ &\times\left|\left\langle J_iIF_i\middle|\middle|\hat{D}\middle|\middle|J_jIF_j\right\rangle\right|^2 .\end{aligned} \tag{2.28}$$

The reduced matrix element for a coupled system (j_1 coupling with j_2 to form a total spin J) can be written as a function of the matrix elements in the individual subsystems, provided the tensor operator can be considered as a tensor product of operators acting on two independent subsystems [47], for which the general result is

$$\begin{aligned}\langle j_1j_2J||T^k||j_1'j_2'J'\rangle = \sqrt{2J+1}\sqrt{2J'+1}\sqrt{2k+1}&\begin{Bmatrix} j_1 & j_2 & J \\ j_1' & j_2' & J' \\ k_1 & k_2 & k \end{Bmatrix} \\ &\times\langle j_1||T^{k_1}||j_1'\rangle\langle j_2||T^{k_2}||j_2'\rangle\end{aligned} \tag{2.29}$$

Applied to the previous equation:

$$\left|\left\langle J_i I F_i \middle\| \hat{D} \middle\| J_j I F_j \right\rangle\right|^2 = \left(2F_i+1\right)\left(2F_j+1\right) \times \begin{Bmatrix} J_i & F_i & I \\ F_j & J_j & 1 \end{Bmatrix}^2 \left|\left\langle J_i \middle\| \hat{D} \middle\| J_j \right\rangle\right|^2 . \tag{2.30}$$

The reduction of the $9j$ to the $6j$ symbol is due to the dipole operator acting only on the electron spin subspace. The identity operator acting on the nuclear spin subspace results in $k_2 = 0$, which makes the $9j$ symbol proportional to a $6j$ symbol through the reduction rules of a $9j$ symbol.

Substituting this back into Equation (2.26) gives

$$\mathcal{A}_{F_j F_i m_{F_j} m_{F_i}} = \mathcal{C}\nu^3 \left|\left\langle J_i \middle\| \hat{D} \middle\| J_j \right\rangle\right|^2 \left(2F_i+1\right)\left(2F_j+1\right) \times \begin{pmatrix} F_i & F_j & 1 \\ -m_{F_i} & m_{F_j} & \mu \end{pmatrix}^2 \begin{Bmatrix} J_i & F_j & I \\ F_j & J_j & 1 \end{Bmatrix}^2 . \tag{2.31}$$

Here, $\mathcal{C}\nu^3 \left|\left\langle J_i \middle\| \hat{D} \middle\| J_j \right\rangle\right|^2$ can be related to the total decay rate. Specifically, summing over all polarizations μ and states in level i, we get

$$A_{ji} = \sum_{\mu m_{F_i} F_i} \mathcal{A}_{F_j F_i m_{F_j} m_{F_i}} = \frac{\mathcal{C}\nu^3 \left|\left\langle J_i \middle\| \hat{D} \middle\| J_j \right\rangle\right|^2}{2J_j+1} . \tag{2.32}$$

The final result is that

$$\mathcal{A}_{F_j F_i m_{F_j} m_{F_i}} = A_{ji}\left(2J_j+1\right)\left(2F_i+1\right)\left(2F_j+1\right) \times \begin{pmatrix} F_i & F_j & 1 \\ -m_{F_i} & m_{F_j} & \mu \end{pmatrix}^2 \begin{Bmatrix} J_i & F_i & I \\ F_j & J_j & 1 \end{Bmatrix}^2 , \tag{2.33}$$

where the factor after A_{ji} is to be interpreted as the relative intensity of the hyperfine transition. This relative intensity is also often colloquially referred to as the Racah intensity. The Wigner $3j$- and $6j$-symbols also contain all the symmetries that must be obeyed for a transition to be possible. The restrictions are that:

- $\mu = m_{F_i} - m_{F_j}$.

- $|F_j - F_i| \leq 1$.
- F_j and F_i are not both 0.

In building the matrices describing the rate equations for a set of hyperfine split levels, Eq. (2.33) is used to calculate the A-coefficient. In addition, the entries in the matrices as given in Equations (2.22) and (2.23) are now replaced by submatrices. In the case of A, every element A_{ji} is replaced by the following matrix:

$$A_{ji} \rightarrow \begin{bmatrix} \mathcal{A}_{\lceil F_j \rceil \lceil F_i \rceil \lceil m_{F_j} \rceil \lceil m_{F_i} \rceil} & \cdots & \mathcal{A}_{\lfloor F_j \rfloor \lceil F_i \rceil \lfloor m_{F_j} \rfloor \lceil m_{F_i} \rceil} \\ \vdots & \ddots & \vdots \\ \mathcal{A}_{\lceil F_j \rceil \lfloor F_i \rfloor \lceil m_{F_j} \rceil \lfloor m_{F_i} \rfloor} & \cdots & \mathcal{A}_{\lfloor F_j \rfloor \lfloor F_i \rfloor \lfloor m_{F_j} \rfloor \lfloor m_{F_i} \rfloor} \end{bmatrix}, \qquad (2.34)$$

where the notation $\lceil\rceil$ and $\lfloor\rfloor$ is used to denote the maximal and minimal value respectively, and F_j, F_i, m_{F_j}, and m_{F_j} are the total spin and its projection for the states j and i.

The matrix B is expanded in a similar way, and the energy difference between the levels now includes the shift due to the hyperfine interaction. However, in replacing the decay rate A by the partial decay rate $\mathcal{A}$ in Eq. (2.19), the polarization parameter μ has to be replaced by the polarization of the laser. In the case of a right- or left-handed circularly polarized laser (σ^+ and σ^- respectively), this is $+1$ and -1. A linear polarization π corresponds with a value of 0.

With the inclusion of the hyperfine splitting, the matrices and Eq (2.25) can be used to track the population of the F, m_F states in time. This is done by numerically solving the rate equations. The Runge-Kutta (RK) methods are a standard choice [43] for numerical solving, but they can become unstable when the system of equations is stiff. The analysis code used in this thesis generates the matrices as given in this section and uses the Livermore Solver for Ordinary Differential Equations (LSODA/LSODE) [48]. The SciPy interface is used to access this method in the Python programming language, which in turn uses a Fortran implementation [49].

Developed for use with both stiff and non-stiff systems of equations, the numerically estimated Jacobian is used to determine the stiffness of the system. A Backward Differentiation Formula (BDF) method is used in the case of a stiff system, and an Adams method (in the same family of solvers as the RK

Table 2.1: Using the interface provided in SciPy, the computation time per frequency point is the same order of magnitude as for the RK methods implemented in ROOT (code provided by D. Yordanov).

LSODA with Jacobian	LSODA	RK4 (ROOT)	Adaptive RK (ROOT)
0.4 ms	0.6 ms	0.1 ms	0.08 ms

method) when the system is evaluated as non-stiff. Note however that the Jacobian of the system of equations is the matrix $A + B\rho\,(\nu)$. By supplying the algorithm with this exact Jacobian, a speedup of 50% is achieved (Table 2.1).

2.3 Simulations of hyperfine structure spectra

2.3.1 Nuclear polarization calculation

The rate equations as developed in the previous section allow the evaluation of the population of the F, m_F states for an arbitrary amount of levels interacting with an arbitrary number of lasers. For simulations of β-asymmetry spectra, the asymmetry is calculated as the product of the nuclear polarization P and the β-asymmetry parameter A. This nuclear polarization can be extracted from the populations calculated using the rate equations. The nuclear polarization of an ensemble of different F, m_F states is defined as

$$P = \frac{\langle m_I \rangle}{I} = \sum_{F,m_F} \frac{w(F, m_F) m_I\,(F, m_F)}{I}, \tag{2.35}$$

where $m_I\,(F, m_F)$ is the projection of I in a specific F, m_F state and $w\,(F, m_F)$ is the population probability of the F, m_F state. A transformation to go from F, m_F quantum numbers to m_I, m_J is therefore required. In the limit of a strong external magnetic field, where the hyperfine interaction can be considered a perturbation effect, a simple mapping between the two sets of quantum numbers is possible with the assumption that the magnetic field evolves slowly enough such that the population of each eigenstate does not change [43].

The Hamiltonian describing both the hyperfine interaction and the magnetic interaction with the electrons (Zeeman interaction) for a specific fine structure

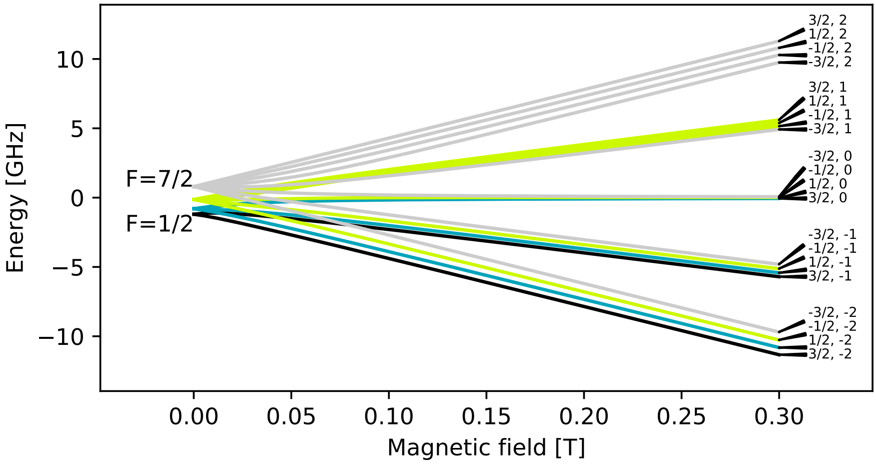


Figure 2.2: Breit-Rabi diagram for meta-stable ^{35}Ar. The quantum numbers on the right are the (m_I, m_J) numbers. The colors group sub-levels originating from the same F state. The atomic and nuclear parameters used in this calculation are given in

level is of the form

$$\mathcal{H} = A\boldsymbol{I} \cdot \boldsymbol{J} + g_L \mu_B \boldsymbol{B} \cdot \boldsymbol{J} \tag{2.36}$$

Table 2.2: Projection table of (m_I, m_J) for the case of the metastable state of ^{35}Ar. From spin-projection, it follows that $m_F = m_I + m_J$ and is the number calculated in the array. The groupings of different F states for $A > 0$ are given by L shapes, and are colored to show the groupings for $F = 7/2$ (purple), $F = 5/2$ (green), $F = 3/2$ (white) and $F = 1/2$ (red).

$\mathbf{m_J}$ \ $\mathbf{m_I}$	$-3/2$	$-1/2$	$1/2$	$3/2$
-2	$-7/2$	$-5/2$	$-3/2$	$-1/2$
-1	$-5/2$	$-3/2$	$-1/2$	$1/2$
0	$-3/2$	$-1/2$	$1/2$	$3/2$
1	$-1/2$	$1/2$	$3/2$	$5/2$
2	$1/2$	$3/2$	$5/2$	$7/2$

where g_L is the Landé g-factor for the level under consideration and μ_B is the Bohr magneton. In the strong field limit ($\|\boldsymbol{B}\| \gg |A| / (g_L \mu_B)$), the hyperfine interaction can be neglected and only the Zeeman interaction is present. This leads to the grouping of levels with the same m_J, as seen in Figure 2.2. When creating a projection table as in Table 2.2, the rows therefore represent levels of the approximately same energy. The quantum number m_F can be assigned by considering the conservation of spin-projection that guarantees $m_F = m_I + m_J$. Only the quantum number F now needs to be assigned to the (m_I, m_J) states in order to have a mapping between the two sets.

If the hyperfine A parameter is positive, the states with the highest F quantum number are also highest in energy at $\|\boldsymbol{B}\| = 0$. As levels with the same m_F will repulse each other, the relative ordering of these levels is conserved. The bottom row of Table 2.2 therefore corresponds with the sub-states with maximal F. The second row from the bottom will correspond with the $F - 1$ sub-states, except for the minimal projection on that row in the leftmost column. This state is missing from the group of maximal F sub-states, and is therefore grouped with them. Continuing in this fashion up the table shows that the grouping forms L -shapes. This completely defines the one-to-one mapping between the two sets of quantum numbers. When the hyperfine A parameter is negative, the same reasoning leads to groupings in ⅂-shapes. Table 2.2 has been annotated with colored boxes that illustrate the grouping.

This mapping has been verified by determining the eigenvalues and eigenstates as a function of magnetic field with the matrix representation of Eq. (2.36). With the eigenstates, the spin projections can be explicitly calculated and compared to the expected result from the constructed mapping.

Applied to the ^{35}Ar system in the meta-stable state ($I = 3/2$, $J = 2$), the resulting Breit-Rabi diagram is given in Figure 2.2. The (m_I, m_J) quantum numbers for a high magnetic field are given on the right. Comparing to the mapping, the calculated quantum numbers match the mapping exactly.

2.3.2 Photon rate calculation

From the rate equations, the photon emission rate can also be calculated. As only moving population from an upper level to a lower one is accompanied by photon emission, only parts of the A and $B\rho(\nu)$ matrices are needed to calculate the photon emission rate.

The decays to a lower lying energy state are selected by only considering the lower triangle of the matrices A and B. The strictly lower triangular form of the matrix is considered, also setting the elements on the main diagonal to zero. By summing each column to a single number and reducing the matrix to a $1 \times n$ vector, the total photon decay rate from each level is calculated. The total photon emission rate per particle $\mathcal{R}$ is then given by the matrix multiplication with the populations of the levels:

$$\mathcal{R} = \begin{bmatrix} \sum_{j<n} A_{nj} + B_{nj} & \cdots & 0 \end{bmatrix} \cdot \begin{bmatrix} N_n \\ \vdots \\ N_1 \end{bmatrix} = \sum_n \sum_{j<n} \left(A_{nj} + B_{nj} \right) N_n. \qquad (2.37)$$

Calculating how long the particle beam is in front of photo-multiplier tubes (PMTs) and integrating the photon rate over this timespan gives the total amount of photons per particle to be observed. Note that Eq. (2.25) is still used to determine the dynamics of the populations.

2.3.3 Spectrum calculation

The result of the rate equations can now be converted from a population into either a polarization, or an amount of emitted photons per particle. These calculations are carried out over a range of laser frequencies and the result is either an optical or β-asymmetry spectrum.

Both these methods have been applied in Figure 2.3. In these simulations, a laser power density of 80 W/m^2 has been used to calculate the polarization spectrum and 2 W/m^2 for the optical spectra. These different power densities are used since the goal in the polarization spectrum is to generate many pumping cycles and thereby maximizing the polarization and resulting β-asymmetry signal. When a photon spectrum is measured, the goal is to see the different hyperfine peaks with their Racah intensities and not too much power broadening. A lower laser power is therefore used. The interaction time for both types of spectra has been calculated for a 50 kV beam. For the polarization spectrum, the beam passes through the full setup as described in Chapter 4. The optical signal is obtained by integrating the photon rate in front of the optical detection region. This results in interaction times of approximately 4 and 0.4 μs respectively.

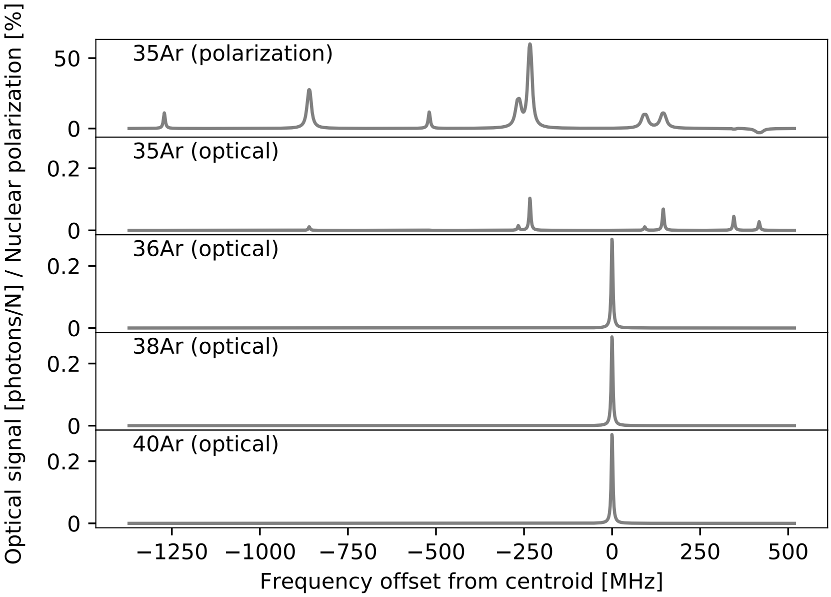


Figure 2.3: Predicted spectra for the experimental setup as described in Chapter 4. Isotope shifts and differences in yield have not been included. The total intensity in the optical spectrum of ^{35}Ar is spread over several hyperfine transitions, reducing the intensity per peak significantly. Peaks seen in the polarization spectrum at $\approx -1250\,$MHz and $-500\,$MHz are also present in the optical spectrum, but with a very small intensity. This illustrates the higher sensitivity that can be achieved via β-asymmetry detection.

To fit the optical spectra, the full set of rate equations will rarely be used. Instead, a lineshape (typically the Voigt profile) is placed at each calculated transition frequency (based on Eq. (2.10)). The intensity of each profile a is either left as a free parameter or restricted to the relative intensities given by Eq. (2.33). The resulting spectrum S is then the sum of the individual profiles:

$$S(\nu) = \sum_i a_i \cdot \mathcal{V}(\nu; \nu_i, \sigma_i, \gamma_i)\,. \tag{2.38}$$

The advantage of this method is the speed of calculation compared to solving the rate equations. This speedup is crucial in fitting routines. The fitting of optical spectra has been implemented as a standard feature in a user-friendly analysis package developed in the course of this work, which is presented in Chapter 5.

For the polarization spectra, no such simplification can be made and the rate equations must be used to fit to experimental data. The rate equations have been implemented in the same analysis package, but are not part of the standard distribution.

Chapter 3

Spin-polarizing ^{35}Ar through optical pumping

> *I, I go my own way; I swim against the stream*
>
> —Arch Enemy, *The Eagle Flies Alone*

This chapter will discuss the selection of a suitable optical pumping transition for the polarization of ^{35}Ar. Furthermore, a method to maximize the polarization in ^{35}Ar using multi-frequency pumping is described.

3.1 Laser-polarizing a ^{35}Ar beam

In order to laser-polarize a beam of ^{35}Ar, a suitable optical pumping cycle must be found. In the ion, no suitable transition can be found due to all involved wavelengths being below 100 nm.

A viable transition is available in the atomic spectrum. Figure 3.1(a) shows a Grotrian diagram (data taken from [50]) detailing the known levels and transitions in neutral Ar. Since the first excited state is so high in energy that no laser light of a suitable frequency can be generated, the pumping cycle will have to use one of the metastable levels. The metastable levels of atomic Ar

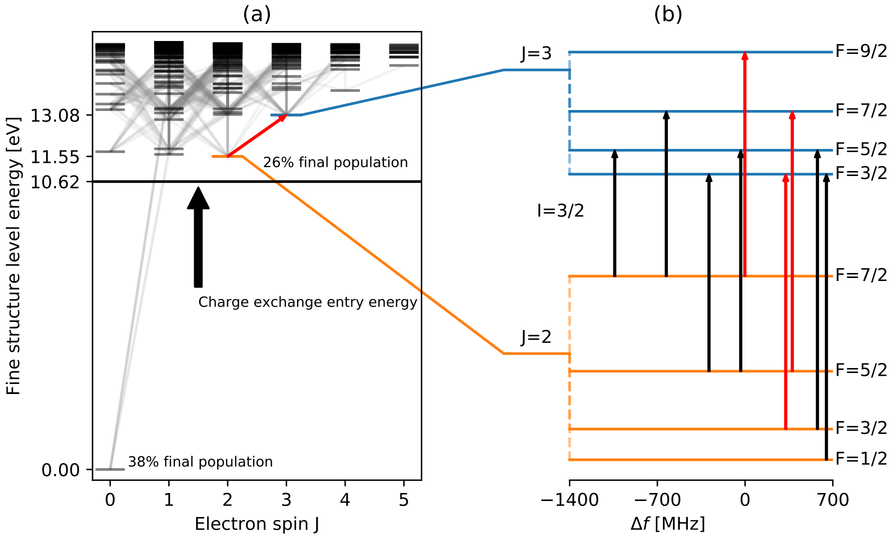


Figure 3.1: Level structure for Ar I. (a) Overview of the entire fine structure of neutral Ar, with details for the charge exchange process annotated. (b) Hyperfine structure of the selected transition, with the hyperfine transitions that yield the highest polarization indicated in red.

are populated by a charge exchange process with a K vapor (see Section 3.3 for more details).

Three metastable levels are present in neutral Ar: one with an electron spin of 0, one with 2 and one with 4. Of these levels, the $J = 0$ state does not have the required hyperfine interaction through which the laser-induced polarization is transferred to the nuclear system and oriented. The population of the spin 4 state is spread across many F-states, hindering efficient polarization. The $J = 2$ state $(3s^2 3p^5 (^2P^{\circ}_{3/2}) 4s)$ has an optical pumping cycle with an 811.754 24 nm transition to a $J = 3$ $(3s^2 3p^5 (^2P^{\circ}_{3/2}) 4p)$ state with an Einstein A-coefficient of $3.31 \times 10^7\,\mathrm{s}^{-1}$ and a natural linewidth of 5 MHz [50]. Furthermore, as this transition has already been studied in ^{39}Ar [51] and the nuclear moments of ^{35}Ar have been measured by other methods [35, 52], the hyperfine parameters for this transition can be calculated precisely enough ahead of time and do not require online measurements.

In Figure 3.1(b), the hyperfine levels for this transition are shown, along with the allowed laser transitions, with those leading to the highest nuclear spin polarization indicated in red.

3.2 Simulations of ^{35}Ar spin-polarization through optical pumping

The hyperfine structure parameters are calculated by taking the measured hyperfine parameters for ^{39}Ar in the same transition [51] and scaling them with the ratio of the g-factors for the A parameter, and the quadrupole moment ratio for the B parameter:

$$A\left(^{35}Ar\right) = \frac{g\left(^{35}Ar\right)}{g\left(^{39}Ar\right)} A\left(^{39}Ar\right) \tag{3.1}$$

$$B\left(^{35}Ar\right) = \frac{Q\left(^{35}Ar\right)}{Q\left(^{39}Ar\right)} B\left(^{39}Ar\right) \tag{3.2}$$

Table 3.1 gives an overview of the parameters that are used in the simulations.

The rate equations as described in Chapter 2 were formulated for ^{35}Ar, and the resulting polarization for the entire hyperfine structure was calculated as a function of the applied laser frequency. A maximal polarization of approximately 65-75% can be achieved by pumping on a single transition (depending on the laser power), as visible in Figure 3.2. This limit is due to the fraction of

Table 3.1: Hyperfine parameters for the $3s^2 3p^5(^2P^{\circ}_{3/2})4s \rightarrow 3s^2 3p^5(^2P^{\circ}_{3/2})4p$ transition in ^{39}Ar taken from Ref. [51]. The magnetic and quadrupole moments for ^{35,39}Ar are taken from [53], to calculate the hyperfine parameters for ^{35}Ar. The uncertainty on the calculated values for ^{35}Ar is the result of uncertainty propagation from all experimental values.

	A [MHZ]	B [MHz]	g [-]	Q [b]
^{39}Ar: $4s$	-287.15(14)	119.3(15)	-0.454(4)	-0.12(2)
$4p$	-135.16(12)	113.6(19)		
^{35}Ar: $4s$	266.6(24)	84(20)	0.42147(13)	-0.084(15)
$4p$	125.5(1.1)	80(19)		

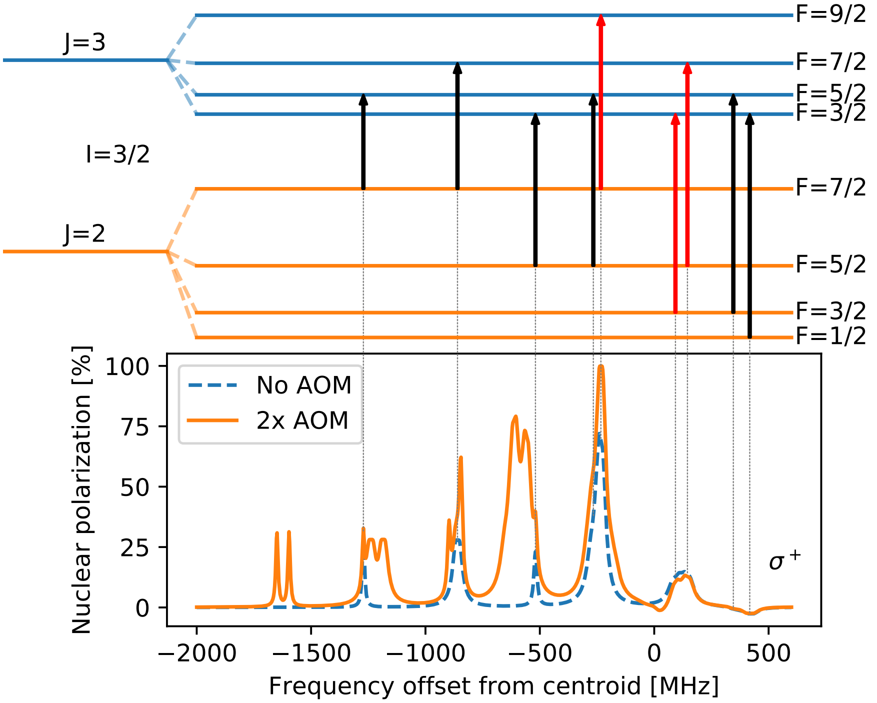


Figure 3.2: Top: the hyperfine splitting of the two selected levels in ^{35}Ar. The transitions indicated in red are used to provide maximal polarization. Bottom: the calculated nuclear polarization over the entire range of the hyperfine spectrum when a single laser beam (No AOM) is present, and when the two additional laser beams, with respectively +325 and +378 MHz are also applied using two AOM's when scanning the frequency of the main beam (2x AOM). The gray dotted lines connect the hyperfine transitions with their location in the spectrum without AOMs.

the population in a single hyperfine level. Note that this polarization is only achievable if the transition is fully saturated.

To increase the maximal polarization, simultaneous pumping of several hyperfine transitions can be used. From simulations, it is shown that simultaneous pumping on the $7/2 \rightarrow 9/2$, $5/2 \rightarrow 7/2$ and $3/2 \rightarrow 3/2$ transitions, indicated in red in Figure 3.1(b), gives a near-perfect (100%) nuclear polarization. To achieve this, two Acousto Optic Modulators (AOM) are needed to generate sidebands to the main laser frequency. The crystals in these AOMs allow interaction between an incoming photon and a phonon generated by applying an RF frequency to the crystal. This shifts the incoming laser frequency by the applied RF frequency. Thus, when scanning the main laser frequency, the frequency of the side bands is scanned simultaneously.

Figure 3.2 shows the result of including 2 AOMs that are tuned to +325 and +378 MHz. These values are the frequency spacings of, respectively, the $3/2 \to 3/2$ and $5/2 \to 7/2$ transitions from the $7/2 \to 9/2$ transition. These frequency spacings are based on the parameters calculated in Table 3.1. The number of peaks in the spectrum is increased due to the presence of these sidebands, and the intensity of most peaks also increases. The strongest component, previously generating $\sim 75\%$, gives now $\sim 100\%$ nuclear polarization.

Additionally, multiple-transition optical pumping reduces the required power to saturate the optical pumping process. A full polarization of the ensemble is therefore easier to achieve.

3.3 Production of an atomic beam of ^{35}Ar in the metastable state using charge exchange

While the previous section has shown the suitability of the 811 nm transition for optical pumping, illustrating that in an ideal two-level system 100% polarization can be achieved, the influence of the population distribution of the hyperfine levels after the charge exchange process will now be investigated.

At the time of the proposal writing [21], it was assumed that $\sim 40\%$ of the ensemble would be in the metastable state after charge exchange with K, based on earlier experiments [52]. This meant that, in the best case, 40% nuclear polarization of the ensemble could be expected. The feasibility and beam time estimates for the experiment were based on this assumption [21].

However, more recent cross section calculations for the charge exchange process based on the formalism of Francis and Rapp [54, 55] suggest a much wider distribution of the ensemble population across the many fine structure levels above the energy entry point (Figure 3.1). The result of these calculations is shown in Figure 3.3, where the distribution of the population across the different fine structure levels is shown, omitting the ground state since this has

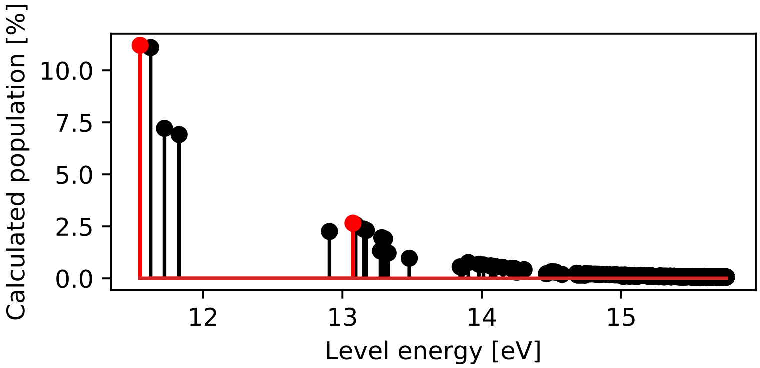


Figure 3.3: Calculated initial population distribution of the fine structure levels in ^{35}Ar. The ground state, with 0% initial population, has been omitted from the plot and the levels in the pumping cycle have been indicated in red.

a 0% population after charge exchange. The population in the metastable level from which the optical pumping starts is calculated to be about 10%, a large deviation from the estimated 40%.

Furthermore, these calculated populations can be used as input in the optical pumping calculations, also using available data on the Einstein A-coefficients from Ref. [50] to include spontaneous decay from all levels. These calculations started by including the first 10 fine structure levels, and were steadily extended by adding more of the populated fine structure levels. The laser was set to include both AOM beams and pump on the strongest polarizing component. The result, going up to 426 included fine structure levels, is given in Figure 3.4.

As more atomic levels are included in the calculations, two effects come into play. Firstly, the relative population available for pumping is reduced which means the entire time dynamics curve is scaled down by a multiplicative factor. Secondly, higher lying levels feed into the meta-stable state at a later time, modifying the rise time of the curve. From these calculations, it is concluded that a maximum of 25% polarization can be achieved, rather than 40%. When this ensemble is implanted in a crystal to study the asymmetry in its β-decay, the large unpolarized fraction of the ensemble will induce a significant reduction in the observed β-decay asymmetry.

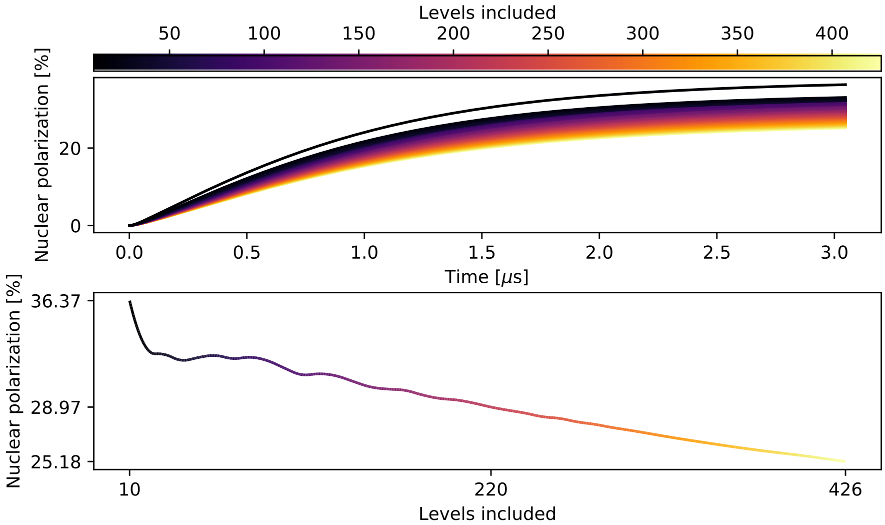
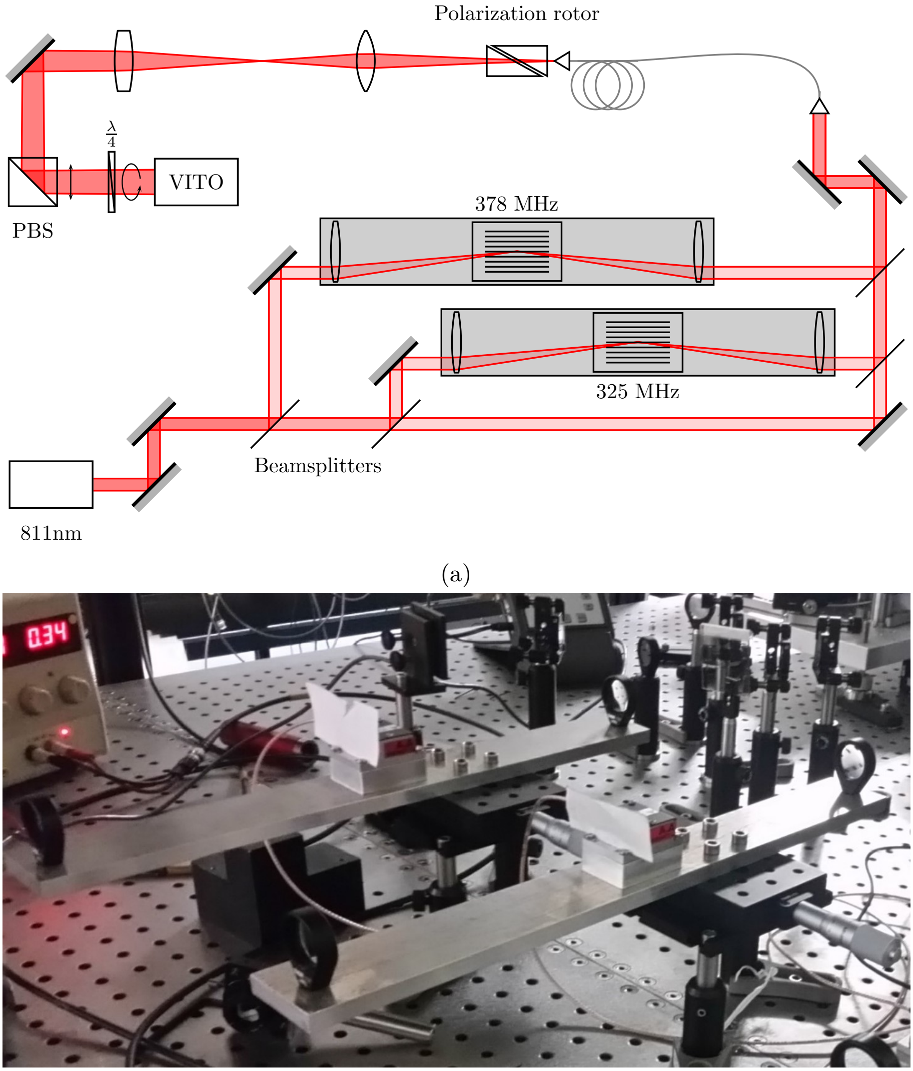


Figure 3.4: Top plot: time dynamics of optical pumping on the strongest component for an increasing number of included levels. Bottom plot: final polarization reached for an increasing number of levels. The line is colored to match the colormap of the top plots. The ticks on the y-axis match the ticks on the x-axis.

With an asymmetry parameter $A = 0.43$ and a polarization $P = 0.25$, a maximal observed β-asymmetry of about 10% is expected, provided a crystal can be identified in which the polarization is maintained long enough.

3.4 Technical realization of a maximally polarized ^{35}Ar beam

Considering the results of the optical pumping simulations, three overlapped beams of the same circular polarization (σ^+) and nearly the same frequency have to be transported to the beamline. Typically, laser beams are overlapped by using Polarizing BeamSplitter (PBS) cubes that reflect or transmit the laser light based on the linear polarization. As this method gives beams of different circular polarization, this method is not suitable. Another way to combine the beams is using mirrors that are transparent for one of the beams. With the



(a)

(b)

Figure 3.5: (a) Schematic overview of the main components in the laser setup and transport down to the VITO beamline. The mounting plates are indicated by gray boxes. (b) The mounting plate for the AOMs and lenses, attached to a translational stage.

frequencies of the beams being so close to each other, such mirrors do not exist. The only remaining option is to use beamsplitter mirrors that are less than 100% reflective. They are used to split the beam before and to recombine the beams after using the AOMs. Figure 3.5 shows a schematic drawing of the laser setup, as well as a picture of the AOMs.

To maximize the efficiency of the AOMs, two mounting plates were designed with space for two lenses and the AOM, indicated by gray boxes in Figure 3.5a. This mounting plate is attached to a translational stage that moves the entire plate perpendicular to the beam path, as visible on Figure 3.5b. By moving the lenses such that the beam enters off-center, the laser light enters the AOM under an angle. When this angle is the Bragg angle of the crystal, the interaction between the photons and the phonons is maximized and the AOM diffracts the beam. The second lens collimates the beam, and also introduces the same angle again to the light. The result is that the beam path after the second lens is unaffected by translating the entire bench.

For these components, the AOMs used are modified MT350-A0.12-800 models from AA Opto-Electronic and the lenses are LBF254-200-B best form lenses from ThorLabs. Best form lenses use two different radii for their circular surfaces, which minimize spherical aberration from the light entering the lens off-center. This ensures the focal point is independent of the location the beam enters the lenses.

Utilizing the setup as described here, 100 mW of laser light was converted to 80 mW of frequency shifted light for both AOMs. This 80% efficiency is also the maximal efficiency as reported by the manufacturer.

After the transition through the AOMs is optimized, beamsplitter mirrors are again used to overlap the beams. A 30 m long optical fiber carries the light from the laser lab down to the beamline, where a telescope is used to shape the beamsize. The final circular polarization is achieved by sending the beam through a polarizing beamsplitter cube followed by a quarter waveplate. A maximum of about 20 mW of power could be delivered to the beamline, distributed uniformly over each laser beam.

Figure 3.5a gives an overview of the entire setup and illustrates the design to tune the AOMs, while Figure 3.5b shows the mounting plates.

Chapter 4

A dedicated beamline for spin-polarizing an atom beam from ISOLDE

Into battle walk in a line
—SABATON, *The Carolean's Prayer*

A new beamline has been constructed at ISOLDE, CERN for producing polarized radioactive beams. In this chapter, we first present a paper describing the new experimental setup. It is followed by more details on the design of the magnetic fields used in the setup. The chapter ends with a description of the cooled crystal holder used in the experiments on ^{35}Ar.

4.1 Article I: The setup for laser spin-polarization at VITO-ISOLDE

The following article has been submitted to *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* (preprint available on arXiv [56])

A new beamline for laser spin-polarization at ISOLDE

W. Gins[a,*], R. D. Harding[b,c], M. Baranowski[d], M. L. Bissell[e], R. F. Garcia Ruiz[a,1], M. Kowalska[b], G. Neyens[a,b], S. Pallada[b], N. Severijns[a], Ph. Velten[a], F. Wienholtz[b], Z. Y. Xu[a], X. F. Yang[a,f], D. Zakoucky[g]

[a] *KU Leuven, Instituut voor Kern- en Stralingsfysica, Leuven, Belgium*
[b] *EP Department, CERN, Switzerland*
[c] *University of York, York, United Kingdom*
[d] *Faculty of Physics, Adam Mickiewicz University, Poznan, Poland*
[e] *School of Physics and Astronomy, The University of Manchester, Manchester, United Kingdom*
[f] *School of Physics and State Key Laboratory of Nuclear Physics and Technology, Peking University, Beijing 100871, China*
[g] *NPI, Czech Academy of Sciences, Rez, Czech Republic*

Abstract

A beamline dedicated to the production of laser-polarized radioactive beams has been constructed at ISOLDE, CERN. We present here different simulations leading to the design and construction of it, as well as technical details of the full setup and examples of the achieved polarizations for several radioisotopes. Beamline simulations show a good transmission through the entire line, in agreement with observations. Simulations of the induced nuclear spin-polarization as a function of atom-laser interaction length are presented for ^{26,28}Na, and for ^{35}Ar, which is studied in this work. Adiabatic spin rotation calculations of the spin-polarized ensemble of atoms, and how this influences the observed nuclear ensemble polarization, are also performed for the same nuclei. For ^{35}Ar, we show that multiple-frequency pumping enhances the ensemble polarization by a factor 1.85, in agreement with predictions from a rate equations model.

Keywords: beamline, laser polarization, β-asymmetry, adiabatic rotation

1. Introduction

Spin-polarized radioactive nuclei have been a staple of nuclear and particle physics research since the discovery of parity violation [1]. With the use of polarized nuclei as a probe in fields ranging from fundamental interactions to material and life sciences [2, 3, 4], an initiative for a dedicated experiment at ISOLDE was started, and a beamline was built and commissioned. Results from the commissioning of the new beamline have been reported in Ref. [5]. The present article documents the technical aspects of this beamline.

Section 2 describes the mechanism of laser polarization through optical pumping and how the induced nuclear polarization can be observed through the asymmetry in the nuclear β-decay. Section 3 reports on the different parts of the beamline, with Section 4 being dedicated to the ion-optical simulations. The generated magnetic fields are discussed in Section 5. Calculations of the adiabatic rotation of the spin-polarized ensembles of ^{26,28}Na and ^{35}Ar in the magnetic fields are presented in Section 6 and compared to the observed asymmetries. The successful use of multiple-frequency optical pumping to achieve higher polarization for ^{35}Ar atoms, to be used in

[*] Corresponding author

Email addresses: `wouter.gins@fys.kuleuven.be` (W. Gins), `robert.harding@cern.ch` (R. D. Harding)

[1] Current affiliation: EP Department, CERN, CH-1211 Geneva 23, Switzerland

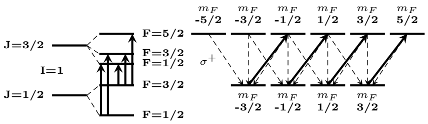


Figure 1: Optical pumping scheme in the D2 line of ^{28}Na. Excitations using σ^+ polarized light are drawn with solid lines, while the dashed lines indicate decay paths through photon emission.

future fundamental interactions studies [4], is described in Section 7 and compared with experimental data. Conclusions are given in Section 8.

2. Optical pumping and β-asymmetry

The hyperfine interaction couples the nuclear spin $\vec{I}$ and the electron spin $\vec{J}$ together to a total atomic spin $\vec{F} = \vec{I} + \vec{J}$, which splits a fine structure level, characterized by the J quantum numbers into several hyperfine levels characterized by the F quantum numbers. Atomic population can be resonantly transferred from one hyperfine level to a radiatively coupled level through interaction with a narrowband laser. Typically, a continuous wave (cw) laser is used. In this process, conservation of angular momentum dictates that the atomic spin $\vec{F}$ can only change by maximally one unit. The left side of Figure 1 illustrates the five allowed hyperfine transitions for the D2 line in ^{28}Na ($3^2S_{1/2} \rightarrow 3^2P_{3/2}$) as solid lines. The term “optical pumping” is used when resonant excitations and decay drive the atomic population towards a specific (magnetic sub)state [6].

When the laser light is circularly polarized (σ^+ or σ^-), conservation of angular momentum further imposes the restriction $\Delta m_F = 1$ or -1 (indicated as solid lines on the right side of Figure 1). The decay back to the lower state is however not bound to this rule and can proceed with $\Delta m_F = 0, \pm 1$ (indicated by dashed lines) [7]. By repeating such excitation/decay cycles many times, the population of a specific F-state is pushed towards substates with either maximal or minimal m_F quantum numbers in the initial state.

The number of excitation/decay cycles can be increased by either increasing the laser photon density or by having a longer interaction time. In a collinear geometry (as used in high-resolution collinear laser spectroscopy experiments [8]), where the particle and cw laser beam are spatially overlapped, such a long interaction time is achieved by choosing an appropriate length of the laser-particle interaction region.

The resulting atomic polarization is then transferred to a polarization of the nuclear spin through the hyperfine interaction, assuming an adiabatic transfer of population between the atomic (coupled) states and the (decoupled) eigenstates in a strong magnetic field. The nuclear polarization in such an ensemble of nuclei is defined as

$$P = \sum_{m_I} \frac{w\left(m_I\right) m_I}{I}, \qquad (1)$$

with $w\left(m_I\right)$ the probability that the $|I, m_I\rangle$ quantum state is populated after the optical pumping process.

The nuclear polarization can be observed by detecting the asymmetry in the β-decay of radioactive isotopes, due to the parity violation in nuclear β-decay [1]. The β-decay of a polarized ensemble has a specific angular distribution that can be approximated to $W\left(\theta\right) \sim 1 + AP\cos\left(\theta\right)$ for allowed β-transitions [9], where θ is the angle between the emitted electron momentum and the nuclear spin orientation. A is the asymmetry parameter of the decay which depends on the initial and final spin of the nuclear states involved in the β-decay, and P is the polarization of the nuclear ensemble with respect to a quantisation axis. The experimental asymmetry (as reported in Section 7) is then defined as

$$A_{exp} = \frac{N\left(0°\right) - N\left(180°\right)}{N\left(0°\right) + N\left(180°\right)} = \epsilon AP, \qquad (2)$$

where ϵ represents depolarization effects which depend on several experimental factors.

For more details on spin-polarization via optical pumping, see Refs. [10, 11].

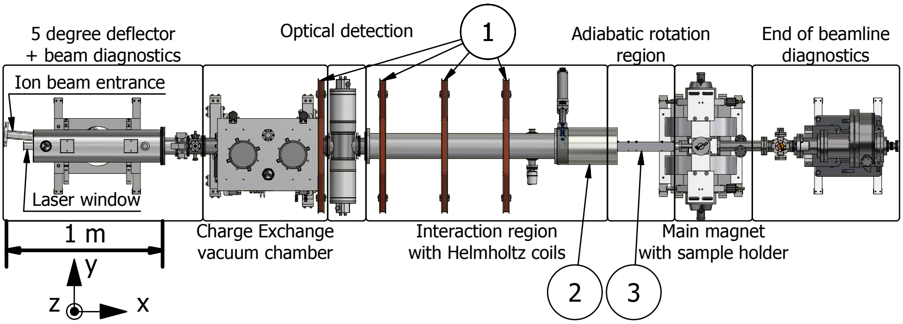


Figure 2: Schematic overview of the VITO beamline, with the main beamline components indicated. The beam from ISOLDE enters from the top left beampipe, and the start of this beampipe is used as the start for the ion-optical simulations. The numbers are used to indicate the coils in Table 1. Indicated in the bottom left is the axis system used throughout this paper.

3. Beamline description

More than 1300 radioactive isotopes of more than 70 elements can be produced at ISOLDE, CERN, via the impact of a 1.4 GeV proton beam on a variety of targets using different types of ion sources [12]. The resulting ion beam is mass-separated in the High Resolution Separator (HRS) after which it can be bunched, if desired, using the ISCOOL cooler-buncher [13]. After mass separation, the ion beam is sent to the laser-polarization setup, which is part of the Versatile Ion Techniques Online (VITO) beamline at ISOLDE, CERN [14]. For ions or atoms having a suitable scheme for laser optical pumping, typically the alkali and earth-alkali elements, the nuclear spin of the isotopes can be polarized through application of optical pumping of an atomic hyperfine transition. The ion beam from ISOLDE is sent into the laser-polarization beamline and overlapped with a circularly polarized laser beam to induce nuclear polarization. After implantation in a suitable host, placed in a strong magnetic field, the change in β-asymmetry is observed as a function of laser frequency, scanned across the hyperfine structure. Although the technique is also applicable to ions [15], the current design is specific for working with atoms. An overview of the layout of the entire optical pumping beamline is shown in Figure 2.

The first element of the beamline is a 5° deflector equipped with a laser window, where the laser beam is overlapped with the pulsed radioactive ion beam. A beam diagnostics box, containing an adjustable iris to define the beam waist and a readout plate for the ion-current, is placed directly after the deflector. Beamline simulations of the 5° deflector are discussed in Section 4.

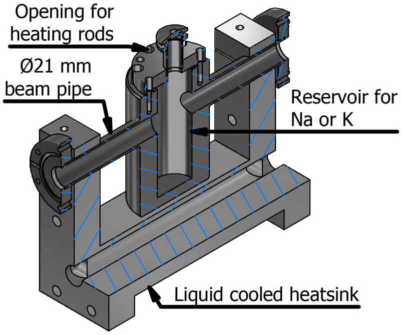


Figure 3: 3/4 view of the CAD drawing of the CEC. The stainless steel reservoir has 6 deep holes to house heating rods. The liquid cooled heatsink clamps onto the beampipe ends.

The Charge Exchange Cell (CEC), housed in the Charge Exchange vacuum chamber of Figure 2 and depicted in Figure 3, is placed after the diagnostic box, where the beam passes through a vapor of Na or K and undergoes charge exchange. It contains a reservoir in the middle where solid Na or K is deposited. The stainless steel reservoir is heated using six RS Components RS-8607016 220 W heating cartridges powered by a DC power supply. The reservoir is heated to approximately 200°C for both Na and K in order to produce a sufficiently dense vapor for charge exchange [16, 17]. Neutralization efficiencies of 100% for a Na$^+$ beam on Na vapor and 50-75% for an Ar$^+$ beam on K vapor have been observed. To minimize diffusion of the Na or K vapor into the rest of the beamline, the ends of the pipe are kept at a lower temperature. This is achieved using a heatsink cooled with circulating Galden® PFPE down to 90°C, which will ensure the ends of the beampipe are kept above the melting point of K (63.2°C) and Na (97.8°C). This minimizes the loss of vapor to the rest of the beamline and ensures a high vapor density in the middle. Non-neutralized ions are deflected from the beam after the CEC using an electrostatic deflector. A specially designed electrode arrangement (referred to as *voltage scanner*, design details in Section 4) is attached to this cell and modifies the kinetic energy of the incoming ion beam. This changes the velocity of the beam, and induces a Doppler shift of the laser frequency. The relation between the labframe ν_{rest} frequency and the frequency ν_{obs} observed by this accelerated (or decelerated) beam is

$$\nu_{obs} = \nu_{rest}\sqrt{\frac{1-\beta}{1+\beta}}, \tag{3}$$

$$\beta = \sqrt{1 - \left[mc^2/\left(mc^2 + qE_{kin}\right)\right]^2}, \tag{4}$$

where m and q are the mass and charge of the particles, c is the speed of light and E_{kin} is the kinetic energy. This allows a fast scanning of the hyperfine structure by changing the acceleration voltage. In Figure 4 we show the electrical diagram of the wiring that enables this voltage scanning. The data acquisition system (DAQ) contains a digital-to-analog converter (DAC) and can provide a controlled voltage of up to ±10 V, which is amplified by a Kepco amplifier (model number BOP 1000M, modified for high inductive loads) by a factor of 100. This voltage of ±1 kV (and typical precision of 0.02 V) is then applied to the secondary windings of an isolating transformer. The insulating transformer supplies the 230 V line voltages on the secondary side to a DC power supply which is then also floated by the ±1 kV. Since the biased electrode of the voltage scanner is connected to the base of the CEC and thus to the lower output of the power supply, both elements are biased to ±1 kV relative to the beamline ground while a constant voltage is applied over the heating rods.

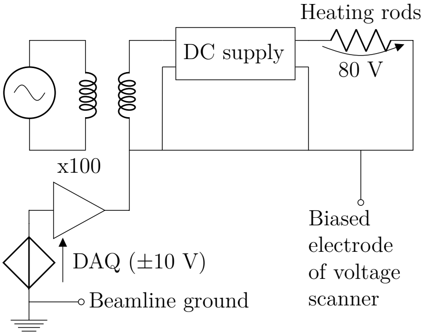


Figure 4: Simplified diagram of the electrical wiring for the voltage scanning. The voltage scanning provides a floating potential for a secondary circuit, where an isolating transformer supplies power to a DC power supply for the heating rods mentioned in Section 3. The data acquisition program can supply ±10V, which is amplified by a factor 100.

Following the CEC is the optical detection region, which is a copy of the light collection region used in the COLLAPS setup [18]. The photomultiplier tubes in this detection area are used for determining the resonant laser frequency through optical detection of the fluorescence decay from a stable isotope of the same element, prior to starting β-asymmetry measurements on the less abundant radioactive species of interest.

In the interaction region, where the optical pumping takes place, Helmholtz coils provide a magnetic field on the order of 2 mT along the beamline axis pointing in the beam direction. This magnetic field defines a quantisation axis and avoids coupling of the atomic spins to possible stray fields in the environment. The minimal length of this interaction region is determined by the time needed for the pumping process. The process of optical pumping with a CW laser can be modelled through the formation of rate equations based on the Einstein formalism [19, 20, 10]. By solving this system of differential equations, the degree of nuclear polarization P (Eq. (1)) can be calculated for any interaction time. The D2 line in Na [5] was used as the case study. Figure 5(a) shows the hyperfine spectrum generated with this method, while in (b) the polarization in the most intense peak is calculated as a function of the laser-atom interaction time (translated into a beam line length assuming a 50 keV beam). In both figures, a laser intensity of 80 mW/cm^2 was used as this is a typically achievable power density. Calculation of the polarization as a function of laser intensity is shown in Figure 5(c), demonstrating that the transition is saturated in the simulation. The difference in maximal polarization for the different nuclei is due to the number of hyperfine levels across which the population is distributed originally. Based on these calculations, a length of 1.6 m was selected as a compromise between achievable polarization and available space in the ISOLDE hall. Although the length needed to fully polarize an ensemble depends on the Einstein A parameter of the transition, 1.6 m will give a sufficiently long interaction time for most strong transitions for which A has a value in the order of $10^7 - 10^8$ Hz. This is illustrated in Figure 5(b) for ^{26,28}Na and ^{35}Ar, the first isotopes that have been polarized with the new set-up.

A series of solenoids and a large electromagnet are placed after the interaction region, with the field of the solenoids acting along the beam direction and the electromagnet generating a field perpendicular to it. The combination of these fields adiabatically rotates the atomic spin in the horizontal plane, orienting it in the same direction as the field of the electromagnet. This field is sufficiently strong to decouple the nuclear from the electron spin. The details of this adiabatic spin

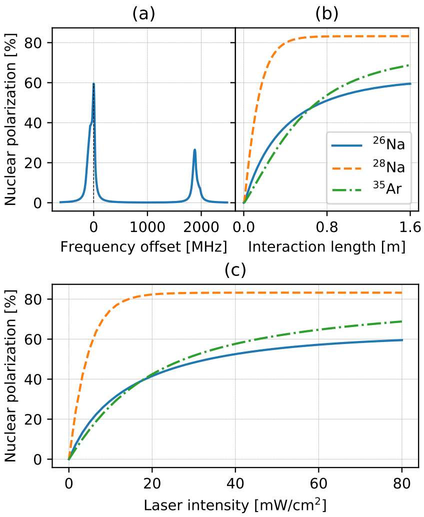


Figure 5: (a) Simulated hyperfine spectrum for pumping ^{26}Na in the D2 line using $\sigma+$ polarized light at a typical laser power of 80 mW/cm^2. The interaction time corresponds to an interaction length of 1.6 m for a 50 keV beam. (b) The calculated polarization in the strongest component of the hyperfine spectrum (indicated with a dashed line in (a)) as a function of interaction length for the different nuclear species discussed in this paper. (c) The laser intensity dependence of the strongest component of the hyperfine spectrum for an interaction length of 1.6 m.

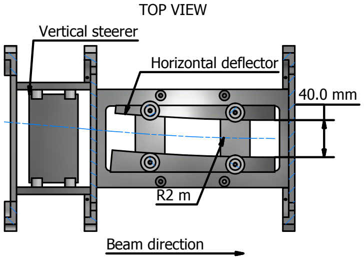
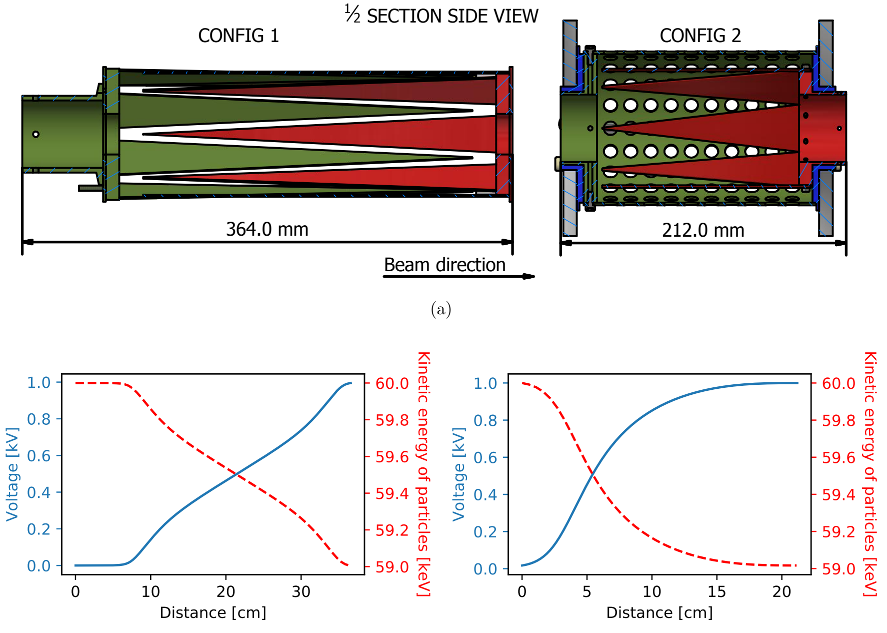


Figure 6: CAD drawing of the 5° ion-deflector. Ion beam coming from ISOLDE (left) is bent 5°s with a radius of 2 m to overlap the ion beam with the laser beam.



(a)

(b)

(c)

Figure 7: (a) 1/2 section view of both configurations for the voltage scanner. The biased electrode is indicated in red, the grounded electrode in green and the teflon insulator in blue (color online). (b) Electric potential (full line) and the kinetic energy (dashed line) of a 60 keV beam of ^{39}K as a function of distance using configuration 1 when 1 kV is applied to the biased electrode. (c) Same as (b), but for configuration 2.

rotation and decoupling of the nuclear and electron spins are discussed in Section 6. A removable sample holder and β-detectors are installed inside the electromagnet, where the polarized ensemble is implanted in a suitable host material. This β-detection setup has been used before in β-NMR studies on Mg isotopes [15].

A diagnostics box containing a wire scanner and Faraday cup, as detailed in [12] (supplemented with a copper plate to detect atomic beams), is installed after this region to provide beam diagnostics.

4. Ion optics

Ion-optical beam transmission simulations were performed to benchmark the effect that the deflector and voltage scanner have on the path of the beam and the transmission that can be expected.

A standard beam of ^{39}K^{+} with 3 π mm mrad beam emittance is generated for the simulations, as reported for the cooler-buncher ISCOOL [13]. The focal point of the beam is, prior to entering the 5° deflector, optimized for maximal transmission using the quadrupole doublet that is installed in front of it. As the doublet is not included in the simulations, the focus is set by optimizing the Twiss parameters in the simulation. For Gaussian

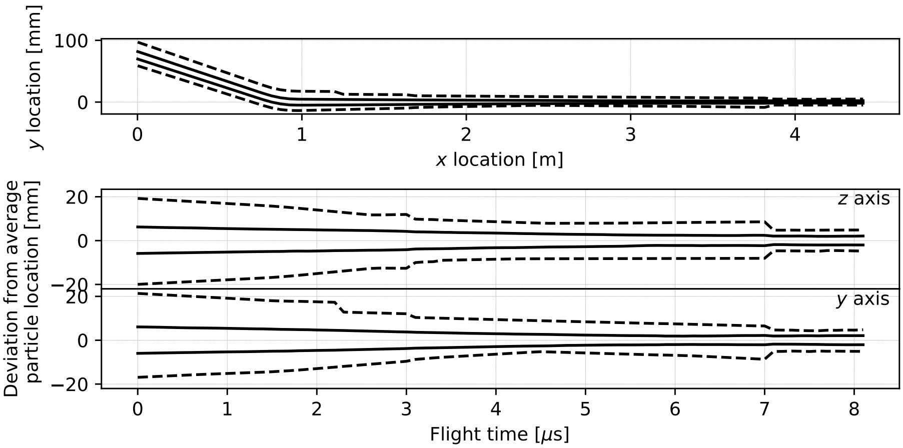


Figure 8: Overview of the particle trajectories as a function of x and y location (top panel), and as a function of flight time and deviation from the center of the beam in the y and z directions (bottom two panels). The solid black lines indicate the area containing $\pm 1\sigma$ of the beam, the dashed lines the location of the outermost particles The focal point corresponds to the location at $x \approx$3.5 m, at a flight time of 6 μs. The axis system is the same one as used in Figure 2.

beams, the distribution in phase space is given by an ellipse, where the orientation and shape is given by the Twiss parameters [21]. Changing these parameters changes both the size of the beam, as well as the focal point. The kinetic energy of the beam is set to 60 keV, which is the maximal beam energy that can be delivered by ISOLDE to the low energy section. For all simulations, the COMSOL multiphysics software was used [22]. The used geometry is a simplified version of the geometry shown in Figure 2. First an overview of all the elements included in the simulations will be given.

The first electrostatic element of the VITO beamline in the simulations is the 5° deflector, which bends the ion beam to overlap it collinearly with the laser light. The 5° deflector has an internal opening of 40 mm and consists of two vertical steerer plates and a pair of electrodes with a machined curve matching 2 m (see Figure 6). After this, a voltage scanner (the design of which will be treated further in this section) adjusts the kinetic energy of the ion beam in a range of ±1 keV. It is mounted inside the vacuum box where the CEC is also mounted and the biased electrode is connected to the CEC. The CEC acts as a long collimator with a 2 cm opening, followed by another collimator with an opening of 1 cm approximately 2 m further downstream. These small collimators guarantee a good overlap between the particle and laser beam. As the charge exchange process neutralizes the charged particles, the CEC is the final electrostatic element considered in the simulations. The tubes and chambers forming the beamline up to this point are also present and are grounded to provide accurate potential fields.

The design for the voltage scanner deviates from a series of ring electrodes connected through a resistor chain [20]. Instead, two specially shaped electrodes define the equipotential electrical field (Figure 7a). Two configurations of this geometrical design of the shaped electrodes have been used.

In the first configuration (Config. 1 in Figure 7a), eight triangular spikes are attached to an octagonal mounting base. Overlapping two of such electrodes gives a gradual and nearly-linear change in potential experienced by the ions, as

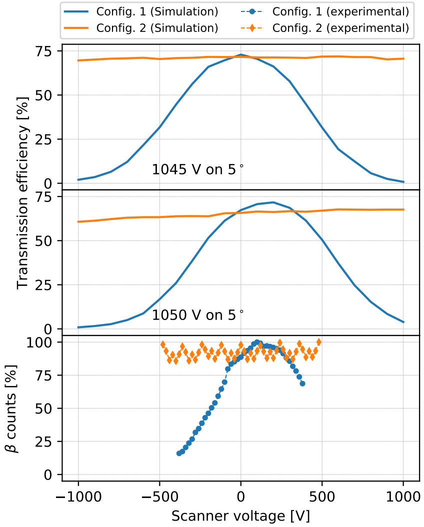


Figure 9: Transmission efficiency through the beamline, simulated for both configurations of the scanning electrodes, as a function of voltage applied to the scanner and for two different voltages on the deflector. When the optimal voltage is applied to the 5° deflector (1045 V, top plot), the beam is centered on the axis of the biased electrode, while a slightly offset voltage (1050 V, middle plot) results in slightly lower transmission since the axes of the beam and the electrode no longer coincide. The experimental transmission profile (bottom plot) agrees with the general trend predicted by the simulations. The β-counts have been normalised to the datapoint with the highest intensity, the staggering in the data is due to the structure of the proton beam supercycle.

shown in Figure 7b. In the second design (Config. 2 in Figure 7a), the grounded electrode is replaced with a cylinder covering the entire scanner, separated from the biased electrode with a teflon insulator. The central beam axis thus is more fully encapsulated and has a better defined ground potential, resulting in a more sigmoidal variation of the electric potential (Figure 7c). To emulate mechanical imperfections related to the construction, the grounding electrode was rotated 0.5° relative to the z-axis as defined in Figure 2. This rotation is implemented in all simulations for both configurations (Figure 7a). The typical flight path of the particles is depicted in Figure 8, where the extent of the particle beam is given. The collimators are visible as the sharp cuts at 3 and 7 μs.

The transmission of the beam through the beam line as a function of scanning voltage has been simulated for the two different designs of the voltage scanner (Figure 9a). A second set of simulations were performed with a slightly detuned 5° deflector (Figure 9b). Both designs have been constructed and used in experiments, such that the transmission simulations can be compared to actual data (Figure 9c).

The transmission data for the first configuration was gathered in the commissioning experiment [5] by means of the total β-counts from a ^{26}Na beam measured as a function of scanning voltage. During an experiment on β-NMR in liquid samples performed in May 2018 [3], the second configuration was used for the first time and the data was gathered in the same way.

The simulations (top two panels of Figure 9) and the online data (bottom panel of Figure 9) show very good agreement. The oscillation in the counts is due to the proton supercycle. Proton pulses are separated by an integer number of 1.2 seconds, but the time between subsequent pulses can vary. A rapid succession of pulses causes build-up and a variation in yield, which can only be averaged away by measuring over many supercycles. In both the measurements and the simulations, the first configuration has a severe beamsteering effect, reducing the transmission efficiency as a function of scanning voltage. The second configuration has no such effect, owing to the better ground potential definition. The simulations with a detuned 5° deflector show that this can result in a slight slope for the second configuration and a shift in the peak location for the first. These features are also present in the data.

Based on comparisons of the beam current on Faraday cups located before and at the end of the beamline, a maximal transmission of ~60% was observed in May 2018. As this transmission is heavily dependent on the exact emittance of the beam, this small deviation from the simulated value of 75% is not unreasonable.

5. Design of magnetic fields

The magnetic field generated in the beamline is separated in three separate sections (see Figure 2), each with their own requirements:

1. the interaction region (labelled 1 and 2 in Figure 2).
2. the transitional field region (labelled 3).
3. the main magnet region.

The interaction region has to provide a weak, uniform magnetic field over the entire beam path that needs to compensate for stray fields in order to maintain the laser-induced atomic spin polarization. The field should be small enough however, not to induce a large splitting of the magnetic substates of the hyperfine levels. A magnetic field of approximately 2 mT fulfills both requirements.

Once the radioactive beam is implanted, the magnetic field has to be strong enough to decouple the nuclear spin from random interactions with potential (defect-associated) electric field gradients in the crystal. The installed electromagnet can generate a field of up to 0.7 T. This value depends on both the current supplied to the magnet as well as the distance between the magnet poles, which can be varied. With a maximal pole distance of 8 cm, different setups for holding samples and placing detectors can be accommodated.

Since the field generated by the electromagnet is perpendicular to the beamline axis (which is also the atomic spin orientation axis), the transitional field region has to be tuned to provide adiabatic rotation of the oriented atomic spins. The field previously designed for the β-NMR setup at COLLAPS (see Ref. [23]) was used as a model for this.

For designing the magnetic field of both the interaction and the transitional region, simulations were made in COMSOL. As a design choice, four octagonal coils arranged in a Helmholtz configuration are used for the interaction region (see Figure 2). The light guides of the β-detection system prevents us from putting a fifth octagonal coil. Therefore, a solenoid with 11.25 cm radius extends the interaction region. Several smaller solenoids directly wound onto a beampipe continue past this point and form the transitional field. The final parameters of all solenoids are given in Table 1.

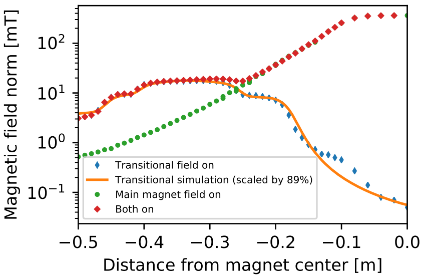


Figure 10: Comparison between the simulation (solid line) and measurements (markers) of the magnetic field. The simulation is scaled down by a global factor to account for a difference between the read-out current and the current applied to the coils. The discrepancy above -0.2 m is due to the residual magnetization of the magnet.

In order to compare the simulated magnetic field profile with reality, magnetic field measurements with a 3D Hall probe were made in three circumstances: the transitional field and the electromagnet on (red wide diamonds in Figure 10), only the main magnet powered (green dots) and only the transitional field powered (blue thin diamonds). Due to the unknown configuration in the main magnet, only the transitional field could be simulated in COMSOL (full line). The good agreement between simulation and measurements suggests a good correspondence between the coil parameters in the simulation and the physical coils.

6. Adiabatic rotation

In the experimental set-up, the atomic spin-polarization axis is along the (laser) beam line. On the other hand, for the β-asymmetry measurements, the nuclear spin-polarization axis should be along the direction of the strong holding field in which the implantation crystal is mounted. This

Table 1: Optimized parameters for the coils as determined with COMSOL simulations. A wire thickness of 0.8 mm was used through all simulations. For the Helmholtz coils, the radius refers to the inscribed circle. The length refers to the dimension of the coil along the beamline axis. The number after the description refers to the labelling in Figure 2.

	Helmholtz coils (1)	Large solenoid (2)	Small solenoids (3)				
			1	2	3	4	5
Current [A]	1.6	1.0	2				
Windings [#]	1000	900	10	65	154	1100	225
Radius [mm]	400	112.5	20.5				
Length [mm]	32	315	40	60	40	154	66

field is perpendicular to the beam line, in order to allow for β-detectors to be mounted at 0 and 180 degrees with respect to the field direction. From the rate equation calculations of the atomic population, the nuclear spin polarization is extracted under the assumption that the decoupling field is oriented along the spin-polarization axis. As we apply a gradually increasing magnetic field in order to rotate (and decouple) the nuclear and electron spins adiabatically into the strong field direction, changes in the nuclear spin polarization due to the adiabtic rotation process are possible. To this end, simulations of this adiabatic spin rotation have been performed. The magnetic field profile in the three directions has been measured and used in these simulations, for which the total field strength along the beam line was shown in Figure 10.

Quantum mechanical calculations, starting from the interaction Hamiltonian including the hyperfine interaction and the two Zeemann interactions with $\vec{I}$ and $\vec{J}$. The used Hamiltonian is of the form

$$\mathcal{H}\,(t) = A\vec{I}\cdot\vec{J} + g_L\mu_b\vec{B}\cdot\vec{J} - g\mu_N\vec{B}\cdot\vec{I}, \qquad (5)$$

where A is the magnetic dipole hyperfine parameter, $\vec{B}$ is the magnetic field vector, $\vec{I}$ and $\vec{J}$ are the nuclear and electron angular momenta, g and g_L are the nuclear and Landé g-factor respectively and μ_B and μ_N are the Bohr and nuclear magnetons.

A similar Hamiltonian has been studied before in the context of particles moving through a magnetic field, resulting in a time-varying field in the reference frame of the particle [24]. The derived differential equations have been solved for the explicit case of a constant magnetic field precessing about the z-axis. Here, the magnetic interaction between the spins $A\vec{I}\cdot\vec{J}$ has been added. Simulation code was written for use with QuTiP [25] which generates and solves the Hamiltonian for arbitrary nuclear species, each having different g-factors, spins and hyperfine parameters. The measured magnetic field is included in this Hamiltonian.

The state vector is initialized in the atomic ground state with populations in the $\vec{F}$ and m_F states as given by the rate equations after the optical pumping process. The interaction strength of the magnetic field represents the changing magnetic field as the particle beam travels through the setup at a certain speed. The measured magnetic field as plotted in Figure 10 is used as input for $\vec{B}$, aside from the first few cm of flight path. Here, the magnetic field components perpendicular to the beamline have been suppressed. Around 0.1 μs, the perpendicular components are no longer suppressed and the spin vectors start precessing around the magnetic field. The varying oscillation periods for the simulated isotopes are mainly due to the different hyperfine interaction strengths. The spin dynamics are then calculated by solving the Schrödinger equation with the previous Hamiltonian. Experimental values for nuclear parameters in these equations were taken from Refs. [26] and [27]. The calculated flight time for the beam from the start of the transitional magnetic field to the implantation host is extended to also include a period where the beam is stopped in the host. The first period (the *dy-*

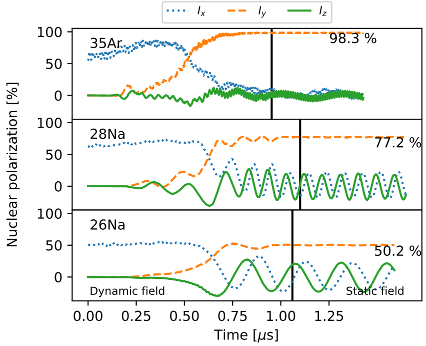


Figure 11: Calculated change in the nuclear spin polarization along the x, y and z directions (dotted, dashed and full lines) due to the spin rotation from along the beam axis (x) to along the main field axis (y). The black full lines indicate the time at which the crystal site is reached by the beam: differences are due to mass and kinetic energy differences.

namic field region) is 2/3 of the total time solved for, while the implanted period (the *static* field region) accounts for 1/3 of the time. More details of the calculation can be found in Ref. [11].

The validity of the simulations has been verified by comparing the observed asymmetries for ^{26,28}Na to the asymmetry calculated on the one hand by the rate equations (neglecting spin rotation) and on the other hand from the quantum mechanical simulations that include the spin rotation process. Figure 11 shows the calculated projection of the nuclear spin as a function of time along the three axes. Averaged over the implanted period, the nuclear spin projection along the main field axis (dashed line) is calculated to be 77% for ^{28}Na and 50% for ^{26}Na.

Table 2: Calculated and observed nuclear spin polarization of ^{26,28}Na. Experimental data taken from Ref. [5]. The quantum mechanical calculation is the average y polarization in Figure 11 after implantation, the rate equation results are from a direct application of the adiabatic approximation to the rate equation populations.

	QM calc.	Rate equation	Experiment
^{28}Na	77%	83%	59%
^{26}Na	50%	58%	39%
Ratio	1.54	1.43	1.51

Table 2 presents the calculated nuclear spin polarization from both the rate equations assuming perfect adiabatic rotation and the quantum mechanical calculations. The observed asymmetry ratio between ^{26}Na and ^{28}Na matches the quantum mechanical calculation, although the absolute number is too high.

Simulations have also been made for the adiabatic rotation of maximally polarized ^{35}Ar (top panel of Figure 11, see next section for details), indicating no loss in nuclear spin polarization from the rotation process.

7. Multiple-frequency pumping

For most elements, it is not possible to achieve 100% nuclear spin polarization using only one laser frequency in the pumping process. E.g. in the case of ^{35}Ar, one of the isotopes envisaged for fundamental interaction studies [4], simulations show that the maximum achievable nuclear spin polarization using a single laser transition is 67% for a laser power density of 80 mW/cm^2. This pumping scheme uses the 811 nm atomic transition ($1s_5 \rightarrow 2p_9$) that starts from the metastable $(1s_5)J = 2$ state which is populated in the charge exchange process [28]. The hyperfine interaction with the nuclear spin of 3/2 leads to the metastable population being distributed among four hyperfine levels (see Figure 12). Thus, when using only one laser transition in the optical pumping process, the maximum achievable atomic (and nuclear) spin-polarization will be limited. This can be overcome by using multi-frequency optical pumping to simultaneously excite more than one hyperfine transitions. The same concept has already been applied at the TRIUMF facility for the polarization of Li, where EOM's were used to induce side-frequencies in the range of ±400 MHz to the main laser beam frequency [29].

In order to estimate the possible gain in nuclear polarization using multiple-frequency pumping, we performed simulations using the rate equations (details to be found in Ref. [11]). In these simulations, the nuclear spin polarization is calculated as a function of laser frequency, assuming optical pumping with two additional laser frequencies. The simulations have been performed for ^{35}Ar, the hyperfine parameters of which have been estimated (Table 3) based on its known nuclear moments [27] and the known hyperfine parameters of ^{39}Ar [30]. Adding two laser frequencies, at +325 MHz and +378 MHz from the main frequency (indicated in red in Figure 12), it was found that by tuning the fundamental beam frequency to the $7/2 \rightarrow 9/2$ transition, nearly 100% nuclear polarization can be achieved, much higher than the 67% estimated for excitation with a single laser frequency.

This was verified experimentally, using a ^{35}Ar beam produced by a 1.4 GeV beam onto a CaO target at ISOLDE. The multi-frequency pumping was realized by using two acousto-optic modulators (AOMs), which can each produce a side band frequency in the required ranges of 300-400 MHz by applying a fixed RF-frequency to the crystal inside the AOM. This shifts the frequency of the incoming laser light by the same value. For details on the setup of the AOMs, see Ref. [11].

The observed hyperfine spectra, using either one laser frequency or all three laser frequencies simultaneously, are shown in Figure 12. The frequencies were varied by scanning the voltage applied to the voltage scanner, thus scanning all incident laser frequencies simultaneously. The induced nuclear spin-polarization is observed by measuring the asymmetry in the radioactive β-decay after implantation in a suitable crystal, as outlined before. The spectra shown in Figure 12 have been recorded by implanting the polarized ^{35}Ar beam into a NaCl crystal kept at a temperature of 15(5) K, where the relaxation time was observed to be longer than the halflife of ^{35}Ar (1.78 s).. The signal height in both spectra is significantly different, as can be seen clearly in e.g. the $7/2 \rightarrow 9/2$ transition, for which the observed asymmetry increases by almost a factor of 2.

Table 3: The hyperfine parameters of ^{35}Ar, deduced from the measured hyperfine parameters of ^{39}Ar in the same laser transition [30] and the known ^{35}Ar nuclear moments [27].

	A [MHz] (calc.)	B [MHz] (calc.)
$1s_5$ $(J=2)$	265.8(28)	83(25)
$2p_9$ $(J=3)$	125.6(12)	80(19)

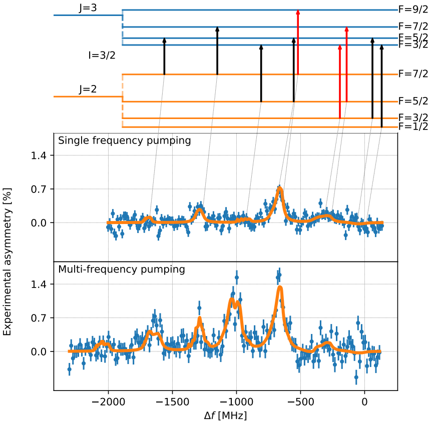


Figure 12: Top: Hyperfine structure of the 811 nm transition with indicated allowed transitions. Using the hyperfine transitions $7/2 \rightarrow 9/2$, $5/2 \rightarrow 7/2$ and $3/2 \rightarrow 3/2$ (marked in red) saturates the polarization of the ensemble. Bottom plots: experimental spectra for both single- and multi-frequency pumping. Both spectra were fitted simultaneously to the rate equations. The laser power in the main beam was set to be a shared fit parameter, while the hyperfine parameters were fixed to the derived value (see Table 3). The transitions in the top are connected with their location in the single-frequency pumping spectrum. The x-axis gives the change in frequency of the fundamental laser beam, resulting in additional peaks in the multi-frequency spectrum at ~ 400 MHz to the left of each peak.

In order to quantify the observed gain in signal strength (and thus spin-polarization), the data have been fitted using the rate equations that were implemented using the SATLAS Python package [31]. In the fitting procedure, the laser power (determining the linewidths), the centroid of the spectrum and the scaling (which the entire spectrum is multiplied with) were left as free parameters, while the A and B factors as given in Table 3 were kept as fixed values. Both spectra were fit simultaneously with the same value for the laser power, thus ensuring the correlations due to shared parameters were propagated correctly when determining the ratio of the signal strengths.

Excellent agreement is found between the observed spectra and the calculated β-asymmetry spectra as a function of the laser frequency, both for the single and triple laser-atom interaction systems. This gives confidence in the predictive power of this simulation package, such that it can be used in the future to optimize the laser polarization experiments. From the fitted signal strengths in both spectra, we find an increase in polarization of a factor 1.85(3).

The maximal asymmetry signal that can be observed requires careful evaluation of the ϵA term in the angular distribution. The asymmetry parameter A is known to be 0.43 [32]. The observed asymmetry is further reduced by several factors, all included in the ϵ. That includes the distribution of the ensemble among the different fine structure levels after charge exchange and the solid angle coverage of the β-detectors. The evaluation of these factors is still ongoing. Note that the observed β-asymmetry in the largest peak amounts to nearly 1.5% using all three laser beams for pumping. This is the largest observed asymmetry for ^{35}Ar until now. Previous studies, using momentum selection in fragmentation reactions to select a polarized beam, reported a maximum of 0.5% in KBr at 20 K [33].

8. Conclusion

To answer the demand for accessible spin-polarized radioactive nuclei, the VITO beamline at ISOLDE was built as a dedicated setup. It is an adaptable beamline delivering highly polarized nuclei to a central detection point.

Beamline simulations agree with the observed transmission efficiency of 60%. The experimentally observed beamsteering effect seen for both designs of the voltage scanner is explained by the simulations.

The series of magnetic fields provide an efficient adiabatic rotation and decoupling of the nuclear spin. Rotation calculations agree with the observed asymmetry ratios. The rotation calculations can be repeated for different species provided the necessary nuclear parameters are available.

Finally, multi-frequency pumping has been established as a viable technique to increase the asymmetry signal that can be expected from the radioactive species. Tests on ^{35}Ar show that the increase and spectrum can be reproduced by the rate equation model.

9. Acknowledgments

This work was supported by the ERC Starting Grant no. 640465, FWO-Vlaanderen (Belgium), GOA 15/010 from KU Leuven, the Science and Technology Facilities Council (UK), the European Union's Horizon 2020 research and innovation programme grant agreement No 654002, and the Ministry of Education of Czech Republic (grant LM2015058).

References

[1] C. S. Wu, E. Ambler, R. W. Hayward, D. D. Hoppes, R. P. Hudson, Phys. Rev. 105 (1957) 1413–1415. doi:10.1103/PhysRev.105.1413.

[2] R. F. Garcia Ruiz, M. L. Bissell, A. Gottberg, M. Stachura, L. Hemmingsen, G. Neyens, N. Severijns, EPJ Web of Conferences 93 (2015) 07004. doi:10.1051/epjconf/20159307004.

[3] M. Kowalska, V. Araujo Escalona, M. Baranowski, J. Croese, L. Cerato, M. Bissell, W. Gins, F. Gustafsson, R. Harding, L. Hemmingsen, H. Heylen, F. Hofmann, A. Kanellakopoulos, V. Kocman, M. Kozak, M. Madurga Flores, G. Neyens, S. Pallada, J. Plavec, K. Szutkowski, M. Walczak, F. Wienholtz, J. Wolak, X. Yang, D. Zakoucky, Interaction of Na+ ions with DNA G-quadruplex structures studied directly with Na beta-NMR spectroscopy, Technical Report CERN-INTC-2018-019. INTC-P-521-ADD-1, CERN, Geneva, 2018.

[4] P. Velten, M. L. Bissell, G. Neyens, N. Severijns, Measurement of the β-asymmetry parameter in ^{35}Ar decay with a laser polarized beam, Technical Report CERN-INTC-2014-062. INTC-P-426, CERN, Geneva, 2014.

[5] M. Kowalska, P. Aschenbrenner, M. Baranowski, M. L. Bissell, W. Gins, R. D. Harding, H. Heylen, G. Neyens, S. Pallada, N. Severijns, P. Velten, M. Walczak, F. Wienholtz, Z. Y. Xu, X. F. Yang, D. Zakoucky, Journal of Physics G: Nuclear and Particle Physics 44 (2017) 084005.

[6] A. Kastler, J. Opt. Soc. Am. 47 (1957) 460–465. doi:10.1364/JOSA.47.000460.

[7] W. Demtröder, Laser Spectroscopy - Basic Concepts and Instrumentation, Springer-Verlag, 1981.

[8] R. Neugart, J. Billowes, M. L. Bissell, K. Blaum, B. Cheal, K. T. Flanagan, G. Neyens, W. Nörtershäuser, D. T. Yordanov, Journal of Physics G: Nuclear and Particle Physics 44 (2017) 064002. doi:10.1088/1361-6471/aa6642.

[9] J. D. Jackson, S. B. Treiman, H. W. Wyld, Phys. Rev. 106 (1957) 517–521. doi:10.1103/PhysRev.106.517.

[10] D. T. Yordanov, From ^{27}Mg to ^{33}Mg: transition to the Island of inversion, Ph.D. thesis, Katholieke Universiteit Leuven, 2007. URL: `https://fys.kuleuven.be/iks/nm/files/thesis/phd-thesis-dtyordanovphd.pdf`.

[11] W. Gins, Development of a dedicated laser-polarization beamline for ISOLDE-CERN, Ph.D. thesis, KULeuven, In preparation.

[12] E. Kugler, Hyperfine Interactions 129 (2000) 23–42. doi:10.1023/A:1012603025802.

[13] E. Mané, J. Billowes, K. Blaum, P. Campbell, B. Cheal, P. Delahaye, K. T. Flanagan, D. H. Forest, H. Franberg, C. Geppert, T. Giles, A. Jokinen, M. Kowalska, R. Neugart, G. Neyens, W. Nörtershäuser, I. Podadera, G. Tungate, P. Vingerhoets, D. T. Yordanov, European Physical Journal A 42 (2009) 503–507. doi:10.1140/epja/i2009-10828-0.

[14] M. Stachura, A. Gottberg, K. Johnston, M. L. Bissell, R. F. Garcia Ruiz, J. Martins Correia, A. R. Granadeiro Costa, M. Dehn, M. Deicher, A. Fenta, L. Hemmingsen, T. E. Mølholt, M. Munch, G. Neyens, S. Pallada, M. R. Silva, D. Zakoucky, Nuclear Instruments and Methods in Physics Research, Section B: Beam Interactions with Materials and Atoms 376 (2016) 369–373. doi:10.1016/j.nimb.2016.02.030.

[15] M. Kowalska, D. T. Yordanov, K. Blaum, P. Himpe, P. Lievens, S. Mallion, R. Neugart, G. Neyens, N. Vermeulen, Physical Review C - Nuclear Physics 77 (2008) 1–11. doi:10.1103/PhysRevC.77.034307.

[16] N. Bendali, H. T. Duong, P. Juncar, J. M. Saint Jalm, J. L. Vialle, Journal of Physics B: Atomic and Molecular Physics 19 (1986) 233–238.

[17] A. C. C. Villari, W. Mittig, Nuclear Instruments and Methods 179 (1981) 195–198.

[18] K. Kreim, M. Bissell, J. Papuga, K. Blaum, M. D. Rydt, R. G. Ruiz, S. Goriely, H. Heylen, M. Kowalska, R. Neugart, G. Neyens, W. Nörtershäuser, M. Rajabali, R. S. Alarcón, H. Stroke, D. Yordanov, Physics Letters B 731 (2014) 97 – 102. doi:https://doi.org/10.1016/j.physletb.2014.02.012.

[19] C. J. Foot, Atomic Physics, Oxford Master Series in Physics, Oxford University Press, 2005.

[20] R. F. Garcia Ruiz, C. Gorges, M. Bissell, K. Blaum, W. Gins, H. Heylen, K. Koenig, S. Kaufmann, M. Kowalska, J. Kramer, P. Lievens, S. Malbrunot-Ettenauer, R. Neugart, G. Neyens, W. Nörtershauser, D. T. Yordanov, X. F. Yang, Journal of Physics G: Nuclear and Particle Physics 44 (2017) 044003. doi:10.1088/1361-6471/aa5a24.

[21] K. Floettmann, Phys. Rev. ST Accel. Beams 6 (2003) 034202. doi:10.1103/PhysRevSTAB.6.034202.

[22] COMSOL, Multiphysics Reference Guide for COMSOL 5.2a, 2017.

[23] M. Keim, Messung der Kernquadrupolmomente neutronenreicher Natriumisotope, Ph.D. thesis, Uni Mainz, 1996.

[24] J. Schwinger, Phys. Rev. 51 (1937) 648–651. doi:10.1103/PhysRev.51.648.

[25] J. Johansson, P. Nation, F. Nori, Computer Physics Communications 184 (2013) 1234 – 1240. doi:https://doi.org/10.1016/j.cpc.2012.11.019.

[26] F. Touchard, J. M. Serre, S. Büttgenbach, P. Guimbal, R. Klapisch, M. de Saint Simon, C. Thibault, H. T. Duong, P. Juncar, S. Liberman, J. Pinard, J. L. Vialle, Phys. Rev. C 25 (1982) 2756–2770. doi:10.1103/PhysRevC.25.2756.

[27] T. Mertzimekis, K. Stamou, A. Psaltis, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 807 (2016) 56–60. doi:http://dx.doi.org/10.1016/j.nima.2015.10.096.

[28] A. Klein, B. Brown, U. Georg, M. Keim, Nuclear Physics A 607 (1996) 1–22.

[29] C. D. P. Levy, M. R. Pearson, G. D. Morris, K. H. Chow, M. D. Hossain, R. F. Kiefl, R. Labbé, J. Lassen, W. A. MacFarlane, T. J. Parolin, H. Saadaoui, M. Smadella, Q. Song, D. Wang, Hyperfine Interactions 196 (2010) 287–294. doi:10.1007/s10751-009-0148-9.

[30] J. Welte, F. Ritterbusch, I. Steinke, M. Henrich, W. Aeschbach-Hertig, M. K. Oberthaler, New Journal of Physics 12 (2010). doi:10.1088/1367-2630/12/6/065031.

[31] W. Gins, R. P. de Groote, M. L. Bissell, C. Granados Buitrago, R. Ferrer, K. M. Lynch, G. Neyens, S. Sels, Computer Physics Communications 222 (2018) 286–294. doi:10.1016/j.cpc.2017.09.012.

[32] N. Severijns, O. Naviliat-Cuncic, Physica Scripta (2013). doi:10.1088/0031-8949/2013/T152/014018.

[33] K. Matsuta, T. Tsubota, C. Ha, T. Miyake, Nuclear Physics A 701 (2002) 383–386.

4.2 Magnetic field and coils design

The sequence of magnetic fields experienced by the particles in the beamline is important to generate and maintain maximal polarization. This section details the design of the various coils and solenoids. The main electromagnet (labeled (4) in Figure 4.1) is not included in any of the designs or simulations.

The magnetic field needs to be homogeneous over the volume traversed by the beam, and assembly of the coils themselves and the beamline must still be possible. The final physical dimensions, achieved after optimization in COMSOL, have already been given in Table 1 in the article. Figure 4.1 sketches the final setup of all coils and solenoids.

The optical pumping region is covered by large coils organized in a Helmholtz construction. This gives a sufficiently homogeneous magnetic center and minimizes the amount of material required. The circular shape was approximated by the circumscribed octagon. With a radius of 400 mm, only 4 coils were needed to introduce a homogeneous field of 2 mT along the 1.2 m optical pumping tube. The large size does not hinder the assembly of the beamline as all pieces fit inside with sufficient extra space. Such a large radius did increase the required number of windings on each coil.

As the original design of the β-detectors included long light guides, the detectors would physically interfere with a fifth coil to complete the required 1.6 m optical pumping region. A long solenoid therefore forms the final piece of the optical

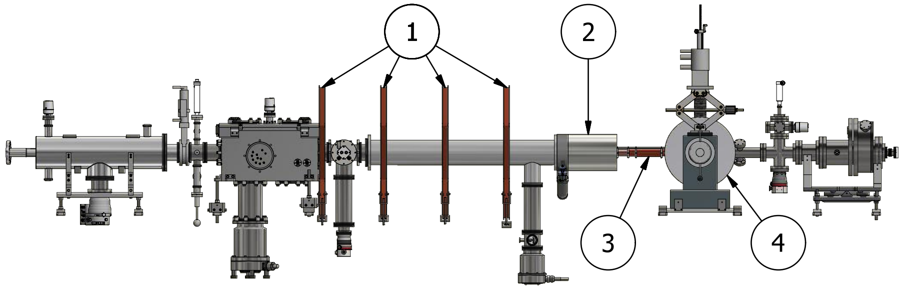


Figure 4.1: The magnetic fields at the VITO beamline are generated by (1) Helmholtz coils surrounding the optical pumping region, (2) a large solenoid to link the optical pumping region and the transitional field region, (3) a series of small solenoids to create the transitional field region for adiabatic spin rotation and (4) the main electromagnet to decouple the nuclear and electron spin.

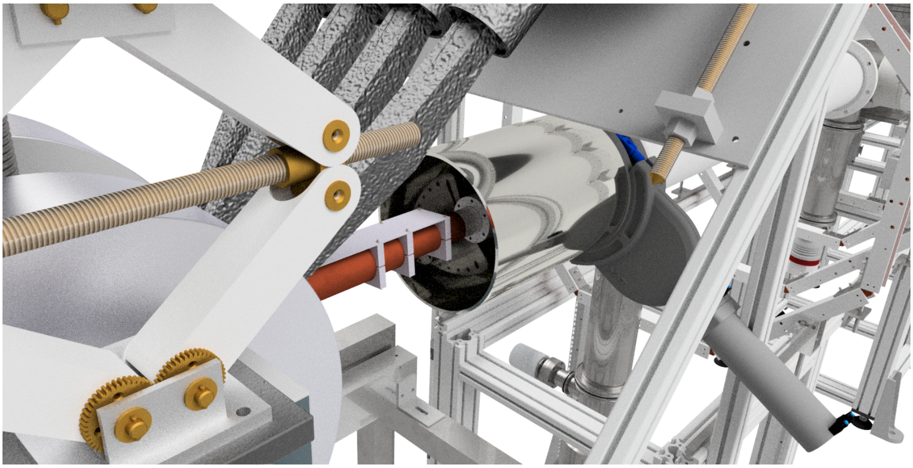


Figure 4.2: CAD rendering of the solenoids. The large solenoid (silver reflective cylinder) has a much larger radius than the beamline and contains one of the small solenoids (copper cylinders) within its volume.

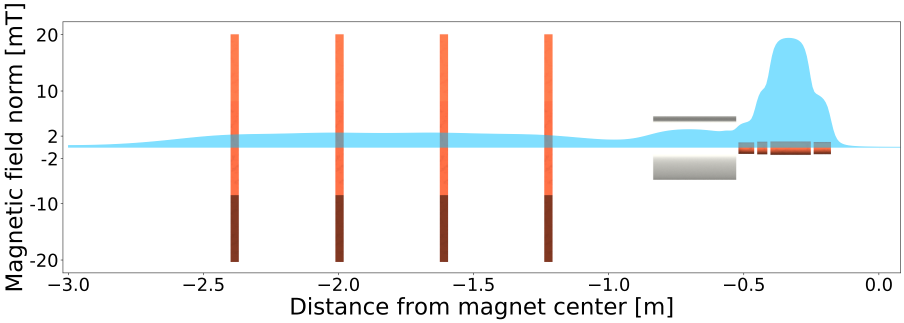
pumping region. Inside the volume of this solenoid, the transitional field solenoids start already (see Figure 4.2).

The design of the transitional field (based on the design as given in Ref. [32]) calls for several sections of solenoids, all of which are used with the same current. Then, the relative intensity of each section is determined by the number of windings only. With a 1 cm spacing in between two sections of solenoids, the number of windings has been optimized to match the field profile generated for the COLLAPS setup. The resulting design can be seen in Figure 4.3a, where the magnetic field is shown as a blue graph on top of a drawing of the coils. Figure 4.3b shows the placement of the Helmholtz coils.

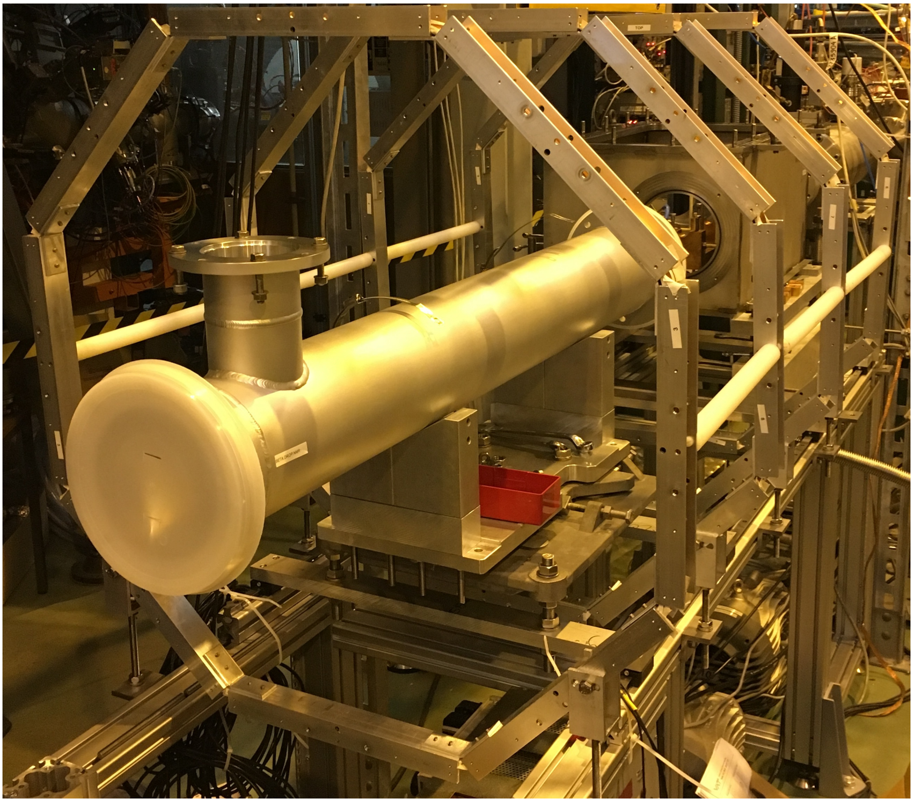
A comparison between the simulated and measured fields was already given in Section 4.1, Figure 9.

4.3 Cooled crystal holder

Measurements on polarized ^{35}Ar performed at NIRS HIMAC [35] showed a strong temperature dependence of the observed β-asymmetry. The normal



(a)



(b)

Figure 4.3: (a) The designed magnetic field, superimposed on a drawing of the coil locations. (b) A picture of the installation of the Helmholtz coils. The aluminium ISO-F 160 beamtube fits comfortably inside the volume of the coils.

crystal holder was therefore replaced by a holder attached to a closed cycle He cold head. The new crystal holder is shown schematically in Figure 4.4.

This cold head provides cooling capability down to approximately 10 K, and an additional heat shield is installed around the crystal holder. As this cold shield is cooled first, it will gather most of the remaining gases in the beamline and delay the onset of condensation on the crystal. Holes are drilled in the front (⌀12 mm) and the side (⌀20 mm) of the cold shield to respectively let the incoming atomic beam pass and let β-particles through.

In order to control the temperature of the crystal, a resistive heating wire is wound around the central rod. A temperature sensor on the central rod near the crystal mount provides input for a PID loop that controls the output of the heating wire. The value of this sensor is used as the temperature of the crystal during the measurements.

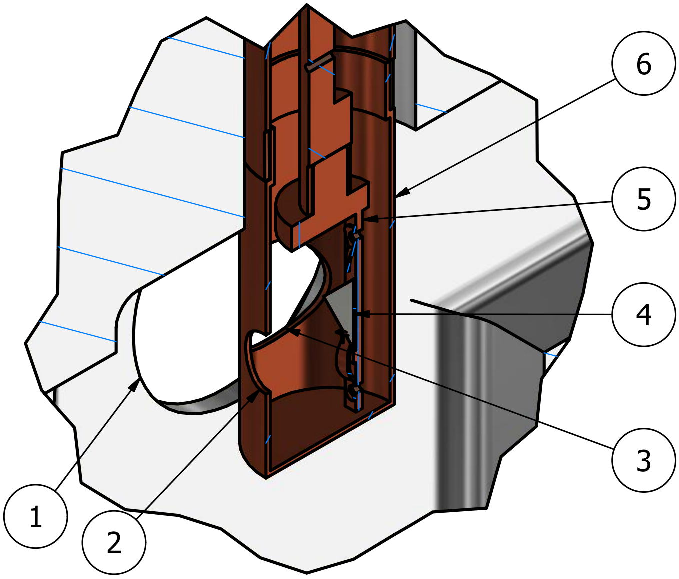


Figure 4.4: Inventor 1/2 view rendering of the custom crystal holder for the Ar beamtimes. Indicated are (1) the ⌀5 cm window in the chamber for the β-particles, (2) the ⌀12 mm opening in the heat shield for the incoming beam, (3) the ⌀20 mm opening in the heat shield for the β-particles, (4) the 1cm x 1cm crystal kept in place with spring-mounted clips, (5) the mounting rod connected to the cold head and (6) the copper heat shield, also connected to the cold head.

Chapter 5

Experimental results

We will not find if we don't seek; We will not know
if we don't peek
—EPICA, *The Quantum Enigma - Kingdom Of*
Heaven Part II

Here, an overview will be given of the experimental details of the two experimental runs on ^{35}Ar at the VITO beam line, performed in April and August 2017. Before detailing the measurements, it is explained how the isotope shift of ^{35}Ar relative to ^{40}Ar is estimated. This is necessary to choose the correct laser frequency for the optical pumping process. The extracted β-asymmetry and relaxation times in different crystals will be discussed in Chapter 6.

5.1 Experimental details

5.1.1 Production of Ar beam

Both runs made use of a plasma ion source with a CaO target to create ^{35}Ar ions, from which a yield of $6 \times 10^6/\mu\text{C}$ was expected according to previous measurements listed in the yield database of ISOLDE [57]. New measurements were performed by the target group in April [58] and in August, giving the same order of magnitude in yield in August but a lower yield in April (see Table 5.1).

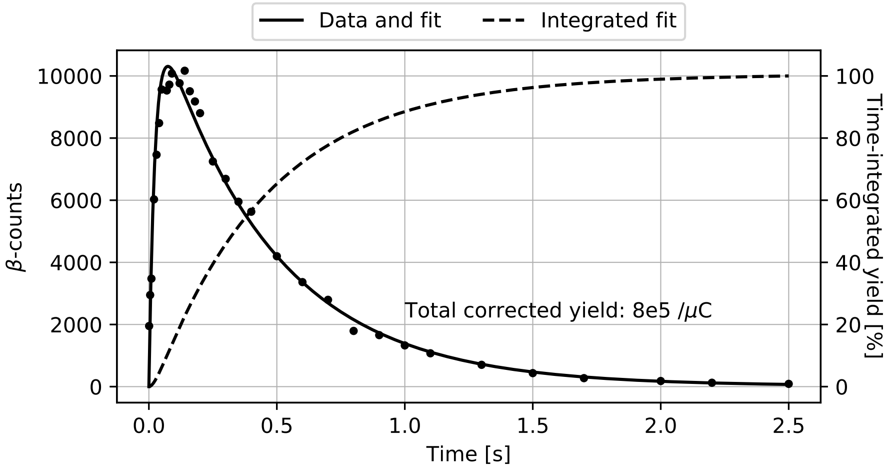


Figure 5.1: β-counts in the tape station as a function of time for ^{35}Ar, with the first 0.6 s containing ≈ 80% of the total yield. The β-counts are not corrected for the half-life and transport time to the tapestation, while the total integrated ion yield is. Data and analysis results courtesy of the target group [58].

A bottle of Ar gas was attached to the ion source, to provide a source of stable isotopes. Natural Ar gas is 99.60% ^{40}Ar, 0.33% ^{36}Ar and 0.06% ^{38}Ar, so all three stable isotopes could be delivered to the setup in sufficient quantities without the use of proton induced production. The HRS separator was used in both instances, allowing the possibility to use continuous as well as bunched beams by using the ISCOOL cooler-buncher.

For both runs, the high-voltage platform was set to 50 kV relative to beamline ground.

Table 5.1: The settings and target details for both experimental campaigns. Yield check for the August campaign was performed by measuring the current on a Faraday Cup (FC), while the April result was done with the tape station.

	Target	HV	Target heating	Line heating	^{35}Ar yield
April	CaO, #596	50 kV	200 A	405 A	$8 \times 10^5/\mu$C
August			215 A		$(5 \times 10^6/\mu$C)

5.1.2 Timing settings

The experiment makes use of several different time signals in order to control different aspects of the setup:

- The first time signal is the arrival of the proton bunch on the target, also called the *proton pulse*. This pulse is provided by ISOLDE equipment and is used to trigger the other time signals which are generated by the VITO experiment. The time between proton pulses to ISOLDE is always an integer amount of 1.2 s.
- A time gate during which ions are extracted from the target, called the *beam gate*. As the release curve in Figure 5.1 shows that the majority of the Ar ions are extracted within the first 0.5 s, the beam gate will be limited to this time in order to suppress the presence of slower released contaminants. Note that the typical time between subsequent proton pulse impacts was restricted to 4.8 s.
- When performing optical measurements, the ISCOOL cooler-buncher [59, 60] was used to accumulate the stable Ar ions and release them in bunches of ≈3 µs FWHM. To this end, a timing signal can switch quickly between the trapping and release potentials of the buncher (see Figure 5.2). This time signal is characterized by an accumulation time t_{acc} and a release time. This release time should be long enough for the bunches to leave ISCOOL.
- Also during the optical measurements, a time gate is applied to the signal from the optical PMT's to remove the background counts registered when the bunches are not in front of the PMT's. This time gate is of the order of the bunch length.

All these timing signals were controlled by the digital delay pulse generator (Quantum Composers 9520 series [61]) installed at COLLAPS.

In the case of optical measurements on stable isotopes, no protons were directed onto the target to induce production. The proton pulse was therefore not used as a trigger, but the pulse generator ran continuously. As the optical measurements used a bunched beam and a time gate on the PMT signals, the combination of the accumulation time t_{acc} in ISCOOL and the length of the time gate t_{gate} gives a background reduction factor of t_{acc}/t_{gate}. For the stable

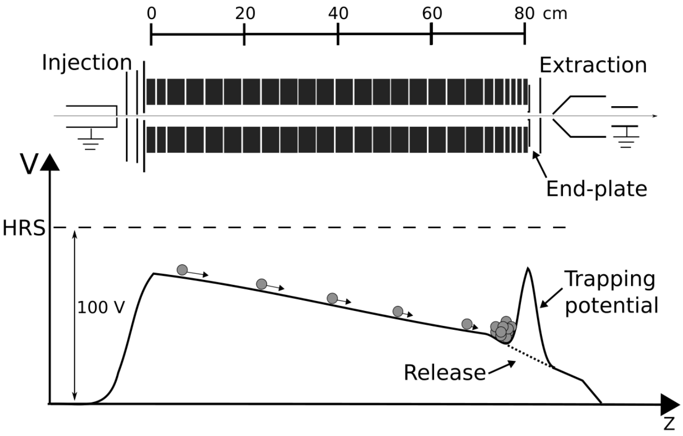


Figure 5.2: ISCOOL consists of segmented electrodes (top), allowing a gradient in the electric potential (bottom). The ions can be trapped and accumulated by applying a trapping potential. Figure taken from Ref. [60].

isotopes (the timing of which is illustrated in Figure 5.3), an accumulation time of 4.9 ms and a gate width of 15 µs was chosen, resulting in a background reduction of ≈ 327.

When performing measurements on radioactive ^{35}Ar, the pulse generator was triggered by the proton pulse arriving at ISOLDE. ISCOOL was not used to bunch the beam and was set to “continuous mode” by not applying the trapping potential. The beamgate now controls how long the beam implants in the crystal. Figure 5.4 shows the timing used for relaxation time measurements in August. The same pattern is used for measurements of the hyperfine spectrum using β-asymmetry, except the beamgate is open for 1 s. In the β-asymmetry spectrum measurements, the total time-integrated number of counts is recorded, while the relaxation measurements integrate the number of counts over smaller timebins. In order to have enough activity to reach sufficient statistics but have minimal effect from the release curve, a beamgate on the order of 0.5 s was chosen for the relaxation measurements, while the longer beamgate for the β-asymmetry spectrum measurements maximizes the activity of ^{35}Ar and suppresses slower released contaminants.

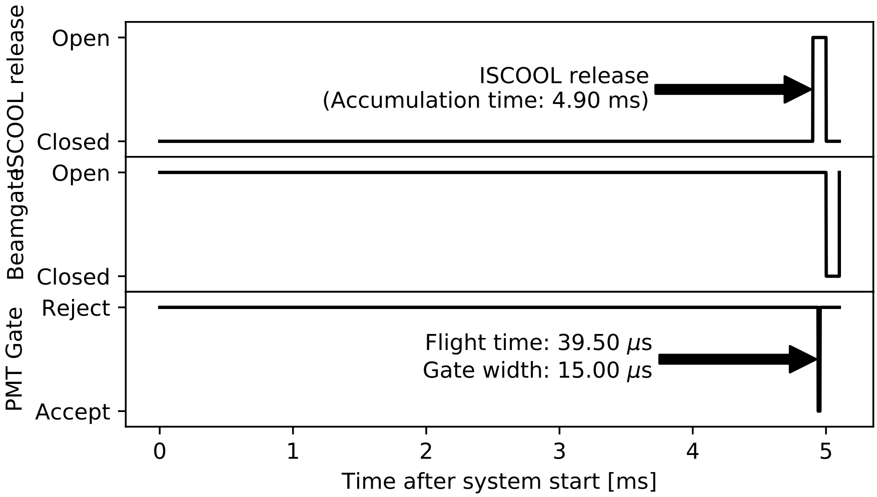


Figure 5.3: For bunching ^{40}Ar, the ions were accumulated in ISCOOL for 4.9 ms and then released. The gate on the signal of the PMTs starts 39.5 µs after release of the bunch and is 15 µs long. After ISCOOL closes, the beamgate briefly closes and the cycle repeats continuously. ^{36,38}Ar used different flight times only.

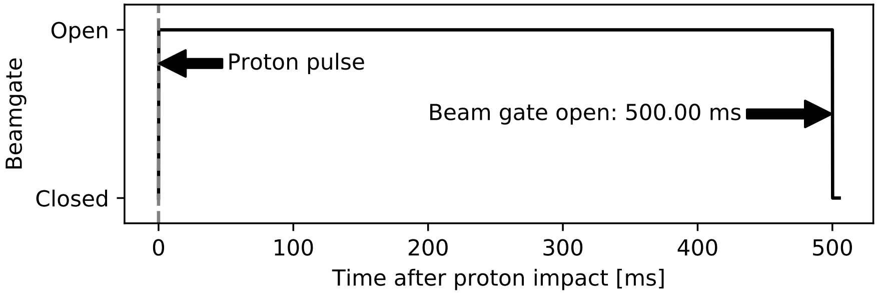


Figure 5.4: For measuring relaxation times of ^{35}Ar, only the beamgate was controlled and opened for 0.5 s after each proton pulse. Scans of the hyperfine spectrum using β-asymmetry opened the beamgate for 1 s instead.

5.2 Analysis method

For the analysis of the different spectra, a combination of χ^2 and Maximum Likelihood Estimation (MLE) routines have been employed. The Python package SATLAS (Statistical Analysis Toolbox for LAser Spectroscopy), developed in the course of this PhD, has been used to perform the analysis.

The development of the library led to a publication describing the software in the journal *Computer Physics Communications* [62], which is included here as article II.

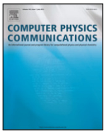
5.2.1 Article II: Analysis of counting data: development of the SATLAS Python package

Computer Physics Communications 222 (2018) 286–294

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc



Analysis of counting data: Development of the SATLAS Python package[☆]

W. Gins [a,*], R.P. de Groote [a], M.L. Bissell [b], C. Granados Buitrago [a], R. Ferrer [a], K.M. Lynch [c], G. Neyens [a], S. Sels [a]

[a] KU Leuven, Instituut voor Kern- en Stralingsfysica, BE-3001 Leuven, Belgium
[b] School of Physics and Astronomy, The University of Manchester, United Kingdom
[c] EP Department, CERN, CH-1211 Geneva 23, Switzerland

ARTICLE INFO

Article history:
Received 16 March 2017
Received in revised form 21 August 2017
Accepted 9 September 2017
Available online 22 September 2017

Keywords:
Data analysis
Counting experiment
Likelihood
Chi-square
Python 3

ABSTRACT

For the analysis of low-statistics counting experiments, a traditional nonlinear least squares minimization routine may not always provide correct parameter and uncertainty estimates due to the assumptions inherent in the algorithm(s). In response to this, a user-friendly Python package (SATLAS) was written to provide an easy interface between the data and a variety of minimization algorithms which are suited for analyzing low, as well as high, statistics data. The advantage of this package is that it allows the user to define their own model function and then compare different minimization routines to determine the optimal parameter values and their respective (correlated) errors. Experimental validation of the different approaches in the package is done through analysis of hyperfine structure data of ^{203}Fr gathered by the CRIS experiment at ISOLDE, CERN.
Source code: https://github.com/woutergins/satlas/
Documentation: https://woutergins.github.io/satlas/

Program summary
Program Title: SATLAS
Program Files doi: http://dx.doi.org/10.17632/3hr8f5nkhb.1
Licensing provisions: MIT
Programming language: Python
External routines/libraries: NumPy, SciPy, LMFIT, Pandas, NumDiffTools
Nature of problem: Fitting data from a counting experiment to extract parameter information.
Solution method: Supply a modular library with fitting routines using pre-implemented goodness-of-fit statistics for counting data under different circumstances.

1. Introduction

Research on exotic nuclei is one of the cornerstones of contemporary nuclear physics [1]. These exotic nuclei are produced with very low yields, resulting in recorded spectra with low count rates (e.g. hyperfine structure spectra, gamma spectra, etc.). In order to extract accurate results from these low statistics datasets, adopting a correct analysis and error determination method is essential.

Traditional data fitting relies on the use of χ^2 minimization techniques. One of the assumptions underlying these techniques is that the distribution of the experimental uncertainties is approximately Gaussian. Due to the Poisson distribution of counting data [2], the validity of this assumption has to be investigated, especially in the low statistics limit of interest. In this article, we present the literature on the equations used to fit data and perform an uncertainty analysis. The suitability of the different approaches is investigated using the custom-written Python package “Statistical Analysis Toolbox for LAser Spectroscopy” (SATLAS). SATLAS started as a project to provide a single package that combines several statistical formulas typically used in the analysis of laser spectroscopic results. It grew into a cohesive unit for fitting data, analyzing results and some convenience functions for visualization, gathering all findings in one location. The selected code architecture allows easy integration of user-defined model functions, so this toolbox is easily extended to other physics problems, such as the analysis of low-statistics gamma-ray spectra. Emphasis lies

[☆] This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).
[*] Corresponding author.
E-mail addresses: wouter.gins@fys.kuleuven.be (W. Gins), ruben.degroote@fys.kuleuven.be (R.P. de Groote).

https://doi.org/10.1016/j.cpc.2017.09.012

on the ease of use and clear documentation. A major advantage of this approach for fitting and model definition is that the results of different experiments can be directly compared.

In the following section, the equation optimized while fitting a model to experimental data is derived from the theory of Maximal Likelihood. Section 3 describes the different components of the SATLAS package which implement the statistical formulas derived in Section 2. Several approaches for calculating uncertainties on the parameters are explored through a toy hyperfine structure dataset, after which the package is tested on recent hyperfine spectra of^{203}Fr data. This illustrates both the capabilities of the package as well as the difference that can arise from different minimization and error analysis approaches. The end result of both investigations is summarized in the conclusion.

2. Fitting of counting experiments

As already argued in 1984 by Baker [3], fitting histograms of counting data requires a different χ^2 formula than for other types of data. Given the continuous push towards the study of weakly produced radioactive isotopes, these results are becoming more relevant. Therefore, we briefly summarize here the results of Baker in order to introduce the different minimization procedures that we integrated into our SATLAS package.

2.1. Cost function

When extracting results from data, the objective is often finding the best fitting *model function*. This function describes the expected response given an input variable and a number of parameters, denoted by $f(x_i)$. To select the best fitting curve, the *cost function* is calculated, which assigns a numerical value to the agreement between the model function with the current set of parameters and the experimental data. The selection of the model function depends on the physics involved in the experiment, and is thus not treated here. In the SATLAS package, the user can insert the model function(s) that is appropriate for the problem being investigated.

The traditional cost function for performing nonlinear least-squares fitting is the weighted χ^2 formula:

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2, \tag{1}$$

where χ^2 is to be minimized. Here, y_i is the experimental value and σ_i the uncertainty. This formula can be derived from the formalism of maximum likelihood [4] by constructing a Gaussian probability distribution around each data point, with the width of the distribution given by the experimental uncertainty. The deviation of the model function from each data point, calculated in units of σ_i, is then optimized. This maximizes the probability that the data is described by the model.

In counting experiments, the uncertainty on each data point is taken as the square root of its value, which is an approximation of the underlying Poisson distribution [2]. The formula then becomes

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{y_i - f(x_i)}{\sqrt{y_i}} \right)^2. \tag{2}$$

However, for spectra with a limited number of counts, approximating σ_i in this way imposes an inherent problem. In such a limit, a data point of 0 counts is likely to be observed which would imply $\sigma_i = 0$, giving rise to infinities in the formula. To avoid this problem, the minimum value of σ is approximated as 1.

Another interpretation of Eq. (1) enables a different way of calculating σ. The model function $f(x_i)$ can be seen as an estimation of the mean value for the distribution from which the data point is drawn. Using this interpretation, σ_i should be given by the square root of $f(x_i)$, which is in general not zero. This results in:

Table 1
Comparison of computational time needed for fitting and calculating the Hessian uncertainty estimate with different methods. The input data is the generated data presented in Section 2.3. The tested methods are available from the LMFIT package, and the time was measured for χ^2 fitting only.

Numerical method	Time [ms]	Relative speed
Levenberg–Marquardt	24	834
SLSQP	370	54
L-BFGS-B	423	47
Powell	487	41
Nelder–Mead	581	34
CG	2 148	9
TNC	2 460	8
COBYLA	20 024	1

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{y_i - f(x_i)}{\sqrt{f(x_i)}} \right)^2. \tag{3}$$

However, in the limit where many data points report 0 counts (such as in a background-free experiment with a low amount of signal counts), neither approach holds. Rather than using cost functions which approximate the Poisson distribution as a Gaussian, a more suitable cost function has to be derived. Calculating the log-likelihood given a Poisson distribution around $f(x_i)$ instead of a Gaussian, the simplified formula is found to be

$$\mathcal{L}(y_i, f(x_i)) = \sum_{i=0}^{N} y_i \log(f(x_i)) - f(x_i). \tag{4}$$

Since most algorithms are developed to find a minimum rather than a maximum, the negative log-likelihood is used in computer programs.

One of the issues with this formula is its form: as it cannot be expressed as a sum of squares, the standard Levenberg–Marquardt algorithm cannot be used [5]. This negatively influences the computation time needed. Indeed, as summarized in Table 1, the LM algorithm is a factor of 15 faster than the follow-up, and a factor of 25 faster than the well known Nelder–Mead method.

Fig. 1 shows the result of Eqs. (2)–(4) when calculating the best fit function to a Gaussian dataset with different statistics. The original χ^2 formula (2) builds an expected distribution around the data point, while the modified formula (3) and the likelihood formula (4) take advantage from the knowledge of the Poissonian nature of a counting experiment. In the dataset with 10 samples, it is clear that the drawn probability distributions each offer a different contribution to the cost function, while the contributions are practically the same in the dataset with 500 samples. This illustrates that low-statistics data need to be treated with non-standard minimization techniques.

2.2. Cost function modifications

Several reasons exist for further modifying the cost function. Two cases will be discussed here: taking uncertainties of the variable into account (errors on the *x*-axis), and adjusting the fit procedure to consider a known literature value for one (or more) parameter(s).

2.2.1. Errors-in-variables

For some datasets, a measurement uncertainty on the variable is non negligible. In the current literature [6,7], the method of effective variance is the most common. Linearizing the response

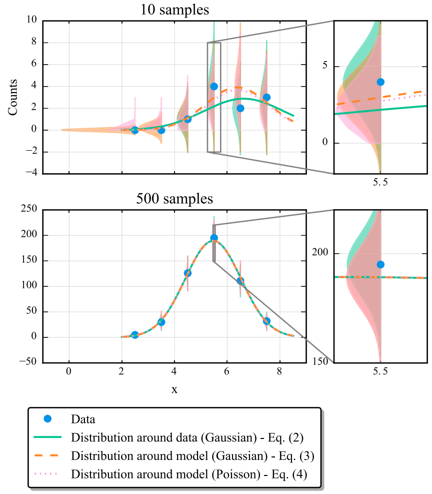


Fig. 1. Fitting of data drawn from a Gaussian distribution ($\mu = 5, \sigma = 1$) with the different methods for 10 samples (top) and 500 samples (bottom). The shaded areas represent the probability distributions, with the line maximizing the probabilities for all datapoints at the same time. Notice the three distinct fits for the top figure in the inset. In the high-statistics dataset in the bottom figure, all of the minimization procedures give the same result.

function and assuming no correlation between the scatter of the variable and the response observable, the *effective variance* is calculated as

$$\sigma_i^2 = \sigma_{y_i}^2 + \left(\frac{\mathrm{d}f\,(x_i)}{\mathrm{d}x_i}\sigma_{x_i}\right)^2 \tag{5}$$

and the weights in the χ^2 formula are adapted to this new calculation.

An alternative description relies on probability theory instead of calculus as the effective variance method does. This approach is based on the work of Hsiao [8]. The joint distribution of variables x and y is given by

$$P_{x_i,y_i}\,(x, y) = P_{y_i}\,(f\,(x))\,P_{x_i}\,(x)\,. \tag{6}$$

In order to convert this expression into a usable fitting formula, a likelihood has to be calculated. Integrating over x and taking the experimental data into account gives

$$L\,(y_i, f\,(x_i)) = \int_{-\infty}^{+\infty} P_y\,(y_i|f\,(x))\,P_{x_i}\,(x)\;\mathrm{d}x. \tag{7}$$

This integral can be converted into a convolution integral of the form

$$L\,(y_i, f\,(x_i)) = \int_{-\infty}^{+\infty} P_y\,(y_i|f\,(x_i+\theta))\,P_\theta\,(\theta)\;\mathrm{d}\theta. \tag{8}$$

This convolution integral is implemented in SATLAS, with the assumption that P_θ is given by a Gaussian distribution. Due to the high computational cost of evaluating this convolution integral, this approach is not used by default.

2.2.2. *Known values of parameters*

In some cases, the value of one of the parameters is already known from other measurements. When this occurs, there is the option of either fixing the parameter in question to the known value (thus reducing the number of parameters) or to include the known value in the cost function. By doing so, the fit takes the previous data into account.

In the χ^2 formula, this is achieved by adding each of the known parameter values as an additional data point to the experimental data:

$$\chi^2_{mod} = \chi^2 + \sum_i^N \left(\frac{a_{lit} - a_{fit}}{\sigma_a}\right)^2, \tag{9}$$

with i running over each parameter a for which the literature value and uncertainty are known. For the likelihood formula, the modification has the same purpose and is of a similar form. The log-likelihood becomes:

$$\mathcal{L}_{mod} = \mathcal{L} - \sum_i^N \frac{\left(a_{lit} - a_{fit}\right)^2}{2\sigma_a^2}. \tag{10}$$

Modifications like these are well known in Bayesian statistics and are referred to as *priors* [9].

2.3. *Uncertainty boundaries*

When fitting the parameters of the model function to data, the uncertainty associated with each parameter has to be calculated. Unless stated otherwise, these uncertainties are typically taken to be the standard deviation σ of a Gaussian distribution around the optimal parameter value. Calculating these uncertainties can be done in several ways [4,10]:

- The uncertainties can be calculated by numerically solving the associated equations (see 2.3.1).
- The Hessian matrix at the optimal point can estimate the uncertainties for a locally Gaussian distributed parameter.
- A histogram of parameter values from a random walk can be generated. The (log-)likelihood value is used as a sampling criterion in this case.

The first two methods are different ways of performing the same calculation, with the estimates merely being a computationally quick approximation of the full numerical solution. Sampling the distribution is a completely different approach. Here, we will discuss all three options.

2.3.1. *Numerical evaluation*

Consider the likelihood function as only depending on the parameter a_j. The values of all other parameters will be changed to achieve the best fit for that particular value of a_j. The assumption is then made that the likelihood is described by a Gaussian shape [4]:

$$L\left(a_j\right) = A\exp\left(-\frac{\left(a_j - a_j'\right)^2}{2\sigma_j^2}\right), \tag{11}$$

where a_j' is the optimal value for the parameter. Going to a χ^2 formulation, this is then given as

$$\chi^2\left(a_j\right) = \left(\frac{a_j - a_j'}{\sigma_j}\right)^2 + \ln A. \tag{12}$$

It is then easy to see that $\Delta\chi^2 = 1$ corresponds to a value of a_j which is exactly one standard deviation away from the optimal value. This can thus be taken as the criterion to find the uncertainty for the determined parameter value.

2.3.2. Hessian approximation

Instead of numerically finding the parameter values which match $\Delta\chi^2 = 1$, it is possible to make a parabolic estimate. From Eq. (12), it follows that

$$\frac{\partial^2\chi^2}{\partial a_j^2} = \frac{2}{\sigma_j^2} \to \sigma_j = \sqrt{2\left(\frac{\partial^2\chi^2}{\partial a_j^2}\right)^{-1}}. \tag{13}$$

Theoretically, the second derivative is thus sufficient to calculate the uncertainty bounds. One pitfall is that it is not just the second derivative of the desired parameter that has to be calculated. Rather, the Hessian matrix is needed:

$$H\left(\chi^2\right) = \begin{pmatrix} \frac{\partial^2\chi^2}{\partial a_1^2} & \frac{\partial^2\chi^2}{\partial a_1\partial a_2} & \cdots \\ \frac{\partial^2\chi^2}{\partial a_2\partial a_1} & \frac{\partial^2\chi^2}{\partial a_2^2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}. \tag{14}$$

Inverting this matrix and using the diagonal elements also takes the correlations between parameters into account.

When using the log-likelihood formula, the factor $-1/2$ which is dropped in the χ^2 formulas has to be reintroduced. This means that the criterion for a 1σ error range from log-likelihood is $\Delta\mathcal{L} = 1/2$ and $\sigma_j = \sqrt{\left(\partial^2\chi^2/\partial a_j^2\right)^{-1}}$.

It is possible that non-linear effects perturb the assumed Gaussian shape. This can result in either a different distribution or an asymmetric Gaussian shape with the discovered σ not being the same for the upper and the lower bound. This can only be known by calculating the likelihood for a series of parameter values and plotting them. In either case, the estimate based on the inverted Hessian matrix is lacking, since this approach does not detect these non-linearities.

2.3.3. Distribution sampling

Instead of assuming a Gaussian distribution for the parameters, a Bayesian look at the problem can be adopted. In this language, we are interested in the *posterior* distribution of the parameters. This is the product of the *probability* distribution of the data given the model parameters, and the *prior* distribution on the parameters, divided by a normalization factor:

$$P\left(model|data\right) = \frac{P\left(data|model\right)P\left(model\right)}{P\left(data\right)}. \tag{15}$$

In the context of this article, the (log-)likelihood function gives the probability distribution and any previous knowledge (such as previous measurements or boundaries) about a parameter is encoded in the prior distribution. For a more complete treatment of Bayesian inference, the reader is referred to Refs. [11,12].

Random steps are made in the high-dimensional parameter space and the value of the log-likelihood is calculated. Depending on the value of the log-likelihood, the step is accepted or rejected according to the criterion used in the sampling algorithm [13,14]. After having sampled sufficiently, a histogram of the accepted steps will show the probability distribution for each parameter. From the properties of this distribution, the final uncertainties on the parameters are estimated.

The advantage of this method is that sampling does not make any assumptions about the details of the parameter probability distribution. The generated data can then be analyzed as preferred: in analogy with the common practice of reporting the 1σ uncertainties, reporting the 16, 50 and 84 percentiles of the parameter histograms is a common choice. The disadvantage is that the computational efforts involved, meaning mostly calculation time and disk space, are significant. Another drawback of this sampling approach is that, a priori, it is not known when the walk has converged, making the estimate of a good walk length a question of both experience and data quality.

2.4. Systematic deviations

As a measure of goodness-of-fit, the *reduced chi-square* (χ^2_{red}) value is often quoted. This is calculated as χ^2/n_d, with n_d the number of degrees of freedom, taken to be the number of datapoints minus the number of parameters. For large n_d, this is expected to be 1 on average, with a variance of $2/n_d$ [15]. This implies an uncertainty on the calculated χ^2_{red} which can be quite significant. In the case of likelihood calculations, there is a statistical measure which will asymptotically be distributed like a χ^2 distribution [3]:

$$\chi^2_{\mathcal{L}} = -2\mathcal{L}\left(y_i, f\left(x_i\right)\right) + 2\mathcal{L}\left(y_i, Y_i\right), \tag{16}$$

where Y_i is the true value at x_i. Since this value is unknown, this is replaced by the measured values:

$$\chi^2_{\mathcal{L}} = -2\mathcal{L}\left(y_i, f\left(x_i\right)\right) + 2\mathcal{L}\left(y_i, y_i\right). \tag{17}$$

From this value, χ^2_{red} can then be calculated.

As a rule of thumb, uncertainties are often multiplied by $\sqrt{\chi^2_{red}}$, with the argument that this accounts for systematic fluctuations not included in the model. Enlarging the error bars on the data points by this factor forces $\chi^2_{red} = 1$. This factor can propagate through the (linear) fit parameters, thereby negating the need for a refit. However, as argued in Refs. [15,16], there are some strong implicit assumptions with this method:

1. The uncertainty distribution on the parameters is Gaussian.
2. The model is linear in all fit parameters.
3. Demanding $\chi^2_{red} = 1$ implies that the used model is the *correct* model underlying the data.

Apart from these requirements, the expected spread on χ^2_{red} makes a value deviating from 1, despite the model and uncertainty distributions matching all requirements, very likely. There appears to be no formal justification for this practice. Taking this into account, a too large or too small value of χ^2_{red} should be investigated for shortcomings of the model instead of adjusting the uncertainties. In the end, it is up to the experimentalist to decide on the goodness-of-fit and to take the comments above into account.

One specific application where this approach is justified is the calculation of weighted averages [12]. Assuming a Gaussian uncertainty distribution on the individual measurements, all three requirements are met by using a constant to fit to the data.

3. SATLAS

For a detailed explanation of the SATLAS package, the user is referred to the online documentation or the docstrings accompanying all classes and methods for a technical description as well as the required input and expected output. What will be presented here is a short overview of the structure and possibilities of the package.

Beyond the scientific Python stack (NumPy and Scipy), SATLAS makes use of the LMFIT package [17] for the implementation of parameter boundaries and expressions linking several parameters, and the emcee package [14] for exploring the likelihood surface via a Monte Carlo walk.

3.1. Available models

In this section, the implementation and design of the models in SATLAS will be explored. These models are used to define the model function as presented in Section 2. The modular design gives the user complete freedom concerning the definition of the response function. This includes convenience functions that allow extension of predefined models, as well as an easy way to define a new model.

3.1.1. General use

The BaseModel is the abstract class used to define common behavior for all models. BaseModel implements the parameter viewing, extracting and restricting, accounting for literature values, and model saving and addition methods. This means that models can be individually defined and the sum (using the standard "+" operator) will return the expected new model.

SumModel and LinkedModel are subclasses of the BaseModel. The input and output for these classes are such that SumModel creates a response function which is the sum of the underlying models. LinkedModel allows simultaneous fitting of different datasets and models with optional links between the parameters of the different models.

Since a polynomial fit is encountered very often in all branches in science, an implementation of such a model is already provided under the name PolynomialModel. Accommodating quick generation of models, MiscModel has to be supplied with a user-defined function and optionally a list of parameter names. The supplied function will then be used by the generated object as a response function. For more involved calculations, where the response function is either too complicated to use a single function or some optimization tricks can be used, the source code of MiscModel provides a minimal example of how BaseModel should be subclassed.

3.1.2. Laser spectroscopy specific

For laser spectroscopic purposes, an HFSModel was implemented. This model calculates the hyperfine spectrum (HFS) of the transition between two atomic levels, given the nuclear and electronic spins involved and the magnetic dipole A and electric quadrupole B parameters. For a system of levels with quantum number F, being a coupling between a nuclear spin I and an electronic spin J, we define the following constants:

$$C(I,J,F) = F(F+1) - I(I+1) - J(J+1) \tag{18}$$

$$D(I,J,F) = \frac{3C(I,J,F)(C(I,J,F)+1) - 4I(I+1)J(J+1)}{2I(2I-1)J(2J-1)}. \tag{19}$$

The resonant transition frequencies between a lower and upper hyperfine level $\nu_{i,j}$ are then dependent on their (frequency) energy difference [18] and an offset labeled 'centroid':

$$\begin{aligned}\nu_{i,j} &= \nu(I, J_{upper}, F_i, A_{upper}, B_{upper}) \\ &\quad - \nu(I, J_{lower}, F_j, A_{lower}, B_{lower}) + \text{centroid},\end{aligned} \tag{20}$$

with the hyperfine level frequency defined as:

$$\nu(I,J,F,A,B) = \frac{AC(I,J,F)}{2} + \frac{BD(I,J,F)}{4}. \tag{21}$$

A higher order correction for the octupole contribution is also available in SATLAS, but is typically not used.

These transition frequencies are automatically calculated by the model object when the parameters are changed and are used when the model function is evaluated. For the lineshapes of the resonances, implementations of Gaussian, Lorentzian, Voigt profiles, a Crystal Ball (Gaussian with low-energy power-law tail) shape [19] and an asymmetric pseudo-Voigt [20] are available. The initial intensity of the resonances is set according to the Racah intensities (I_{Racah}) between the F states, which follow from the coupling of the angular momenta [21]. This intensity can either be fixed to these calculated intensity ratios, left as free parameters or varied between I_{Racah} and I, where I is proportional to $2F_{lower}+1$ [22]:

$$I(s) \sim I_{sat} \cdot \left(\exp\left(\frac{I_{Racah} \cdot s}{I_{sat}}\right) - 1\right). \tag{22}$$

Here, I_{sat} is the saturation intensity of $2F_{lower}+1$.

Additionally, a TransformHFSModel is provided in order to perform model dependent transformations of the x_i input data and the $f(x_i)$ output data. Applications here are, for example, mass dependent doppler shifting from an input voltage to a laser frequency.

3.2. Cost function selection

The fitting routines have been built on top of the LMFIT package. Due to this, boundaries on the parameters and expressions linking them (through the corresponding methods implemented in BaseModel) are accounted for in this package. The behavior of the parameters can be controlled after the creation of a model through the methods:

- `set_expr` to set the value of the parameter depending on the other parameters, e.g. $Ampl_1 = 2 \cdot Ampl_2$
- `set_variation` in order to set the parameter as a constant or a variable,
- `set_boundaries` to restrict the possible values of the parameter,
- `set_lnprior_mapping` to define a prior function for the parameter value and
- `set_literature_values` to define the prior for a parameter with a known literature value.

After setting the desired parameter options, two fitting routines are available: `chisquare_fit` and `likelihood_fit`. Both these functions require a model to be supplied, along with the measurement data. The chosen cost function is then optimized in order to find the best fitting model function and parameters.

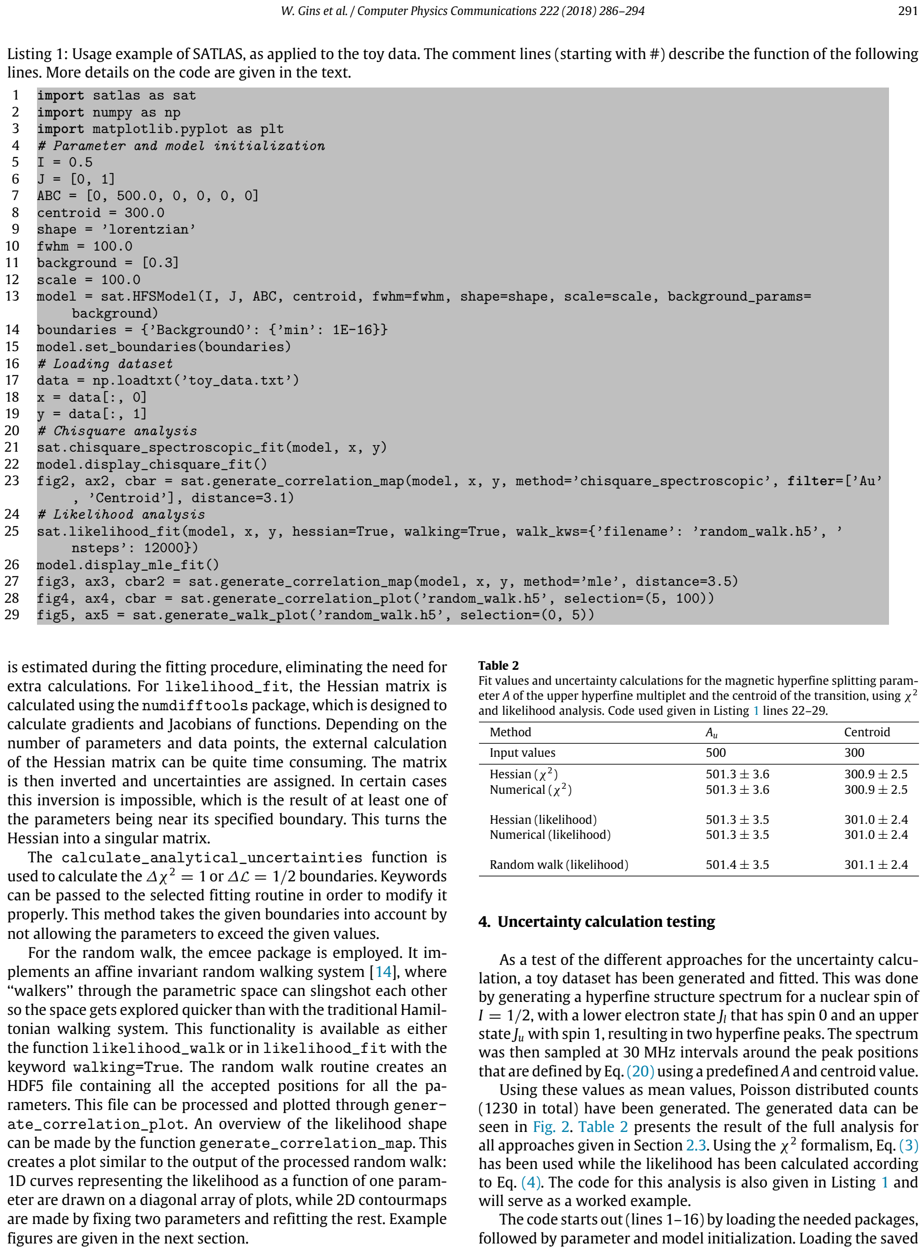
When fitting with the χ^2 routine, either the uncertainties on the data points have to be given, or a function that is applied to $f(x_i)$ in order to get σ_i. When using the likelihood routines, the log-likelihood calculating function has to be supplied. If this is not done, the Poissonian log-likelihood calculation is used. As a convenience function, `chisquare_spectroscopic_fit` automatically ensures that $\sqrt{f(x_i)}$ is used to calculate σ_i. When supplied with the keyword `xerr=value`, these routines automatically transform their respective cost functions in order to incorporate the errors-in-variables with the correct value.

The `loglikelihood` module is used for all the log-likelihood calculations. It also contains a log-likelihood function for Gaussian uncertainties. Through the use of this, normal weighted least squares can benefit from the implementation of the random walk for uncertainty calculation.

Note that the minimization routine for `chisquare_fit` uses the well-documented Levenberg–Marquardt algorithm, which is a quickly converging algorithm that takes advantage of the sum-of-squares form of the cost function. This algorithm cannot be used for `likelihood_fit`, but a selection of other options is available. Convergence will be either not possible or much slower with these algorithms, so some experimentation may be required to find the best algorithm for a given dataset.

3.3. Uncertainty calculators

The Hessian matrix is automatically calculated after fitting such that Eq. (13) is used to estimate the uncertainties. The advantage of the Levenberg–Marquardt algorithm is that the Hessian matrix



Listing 1: Usage example of SATLAS, as applied to the toy data. The comment lines (starting with #) describe the function of the following lines. More details on the code are given in the text.

```
1  import satlas as sat
2  import numpy as np
3  import matplotlib.pyplot as plt
4  # Parameter and model initialization
5  I = 0.5
6  J = [0, 1]
7  ABC = [0, 500.0, 0, 0, 0, 0]
8  centroid = 300.0
9  shape = 'lorentzian'
10 fwhm = 100.0
11 background = [0.3]
12 scale = 100.0
13 model = sat.HFSModel(I, J, ABC, centroid, fwhm=fwhm, shape=shape, scale=scale, background_params=
       background)
14 boundaries = {'Background0': {'min': 1E-16}}
15 model.set_boundaries(boundaries)
16 # Loading dataset
17 data = np.loadtxt('toy_data.txt')
18 x = data[:, 0]
19 y = data[:, 1]
20 # Chisquare analysis
21 sat.chisquare_spectroscopic_fit(model, x, y)
22 model.display_chisquare_fit()
23 fig2, ax2, cbar = sat.generate_correlation_map(model, x, y, method='chisquare_spectroscopic', filter=['Au'
       , 'Centroid'], distance=3.1)
24 # Likelihood analysis
25 sat.likelihood_fit(model, x, y, hessian=True, walking=True, walk_kws={'filename': 'random_walk.h5', '
       nsteps': 12000})
26 model.display_mle_fit()
27 fig3, ax3, cbar2 = sat.generate_correlation_map(model, x, y, method='mle', distance=3.5)
28 fig4, ax4, cbar = sat.generate_correlation_plot('random_walk.h5', selection=(5, 100))
29 fig5, ax5 = sat.generate_walk_plot('random_walk.h5', selection=(0, 5))
```

is estimated during the fitting procedure, eliminating the need for extra calculations. For likelihood_fit, the Hessian matrix is calculated using the numdifftools package, which is designed to calculate gradients and Jacobians of functions. Depending on the number of parameters and data points, the external calculation of the Hessian matrix can be quite time consuming. The matrix is then inverted and uncertainties are assigned. In certain cases this inversion is impossible, which is the result of at least one of the parameters being near its specified boundary. This turns the Hessian into a singular matrix.

The calculate_analytical_uncertainties function is used to calculate the $\Delta\chi^2 = 1$ or $\Delta\mathcal{L} = 1/2$ boundaries. Keywords can be passed to the selected fitting routine in order to modify it properly. This method takes the given boundaries into account by not allowing the parameters to exceed the given values.

For the random walk, the emcee package is employed. It implements an affine invariant random walking system [14], where "walkers" through the parametric space can slingshot each other so the space gets explored quicker than with the traditional Hamiltonian walking system. This functionality is available as either the function likelihood_walk or in likelihood_fit with the keyword walking=True. The random walk routine creates an HDF5 file containing all the accepted positions for all the parameters. This file can be processed and plotted through generate_correlation_plot. An overview of the likelihood shape can be made by the function generate_correlation_map. This creates a plot similar to the output of the processed random walk: 1D curves representing the likelihood as a function of one parameter are drawn on a diagonal array of plots, while 2D contourmaps are made by fixing two parameters and refitting the rest. Example figures are given in the next section.

Table 2
Fit values and uncertainty calculations for the magnetic hyperfine splitting parameter A of the upper hyperfine multiplet and the centroid of the transition, using χ^2 and likelihood analysis. Code used given in Listing 1 lines 22–29.

Method	A_u	Centroid
Input values	500	300
Hessian (χ^2)	501.3 ± 3.6	300.9 ± 2.5
Numerical (χ^2)	501.3 ± 3.6	300.9 ± 2.5
Hessian (likelihood)	501.3 ± 3.5	301.0 ± 2.4
Numerical (likelihood)	501.3 ± 3.5	301.0 ± 2.4
Random walk (likelihood)	501.4 ± 3.5	301.1 ± 2.4

4. Uncertainty calculation testing

As a test of the different approaches for the uncertainty calculation, a toy dataset has been generated and fitted. This was done by generating a hyperfine structure spectrum for a nuclear spin of $I = 1/2$, with a lower electron state J_l that has spin 0 and an upper state J_u with spin 1, resulting in two hyperfine peaks. The spectrum was then sampled at 30 MHz intervals around the peak positions that are defined by Eq. (20) using a predefined A and centroid value.

Using these values as mean values, Poisson distributed counts (1230 in total) have been generated. The generated data can be seen in Fig. 2. Table 2 presents the result of the full analysis for all approaches given in Section 2.3. Using the χ^2 formalism, Eq. (3) has been used while the likelihood has been calculated according to Eq. (4). The code for this analysis is also given in Listing 1 and will serve as a worked example.

The code starts out (lines 1–16) by loading the needed packages, followed by parameter and model initialization. Loading the saved

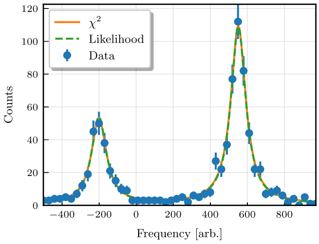


Fig. 2. Toy dataset generated by a hyperfine system with only dipole splitting in the upper state (Au) along with best fits using different cost function calculations.

toy data is done in lines 18–20, and the rest of the code performs the actual analysis. The χ^2 analysis and correlation map generation are done in lines 22 and 24, and the analysis with the likelihood method is performed in lines 26 and 28–30.

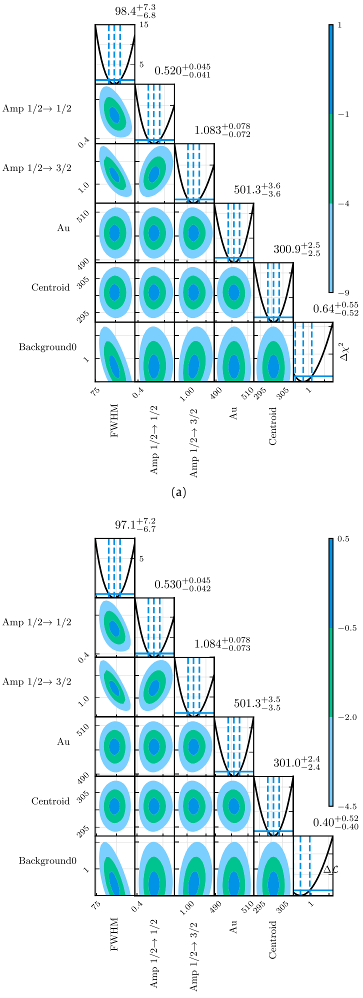
As can be seen in Table 2, all approaches give very comparable results. This shows the equivalence of all approaches under the correct assumptions. If the correlation plots (see lower-left panels in Figs. 3 and 4) show a non-ellipsoidal correlation between parameters or the 1σ-boundaries are very asymmetric, the Hessian estimate will not be justified. In this case, the numerical approach is only valid if left and right sides of the curves on the diagonals in Fig. 3 can be approximated as a parabola. If this is not the case, the assumption of the criterion is meaningless since there is no Gaussian approximation in likelihood space. The only way that uncertainties can then be extracted is through the random walk and characterizing the posterior distribution. Such is the case for the background, amplitude and FWHM parameters in this case, with the background parameter especially presenting a strong deviation from a Gaussian distribution.

Of interest is also Fig. 4(b), where the trajectory of the walkers as generated by the emcee package is plotted. At the plotted scale, it is clear that all walkers start from the same point in parameter space and need some steps in order to start spreading out. This phenomenon is known as burn-in and is a result of the tuning of the sampler settings [14]. The calculated autocorrelation time of the walk can be used to calculate how much burn-in is present and has to be removed. In practice, the easier method by far is plotting the trajectory and judging by eye.

5. Application in low-statistics data: ^{203}Fr

To demonstrate the applicability of the different approaches, data taken by the Collinear Resonance Ionization Spectroscopy (CRIS) team at ISOLDE, CERN in their November 2015 campaign on ^{203}Fr is used [23]. The acquisition at the CRIS setup is of a nearly event-by-event type. This allows later reconstruction of the spectrum for an arbitrary number of scans.

^{203}Fr has a nuclear spin $I = 9/2$, and the atomic transition at 423 nm that couples an $S_{1/2}$ to a $P_{3/2}$ state was probed. For more details on this type of experiment see [24,25]. This gives rise to 6 possible hyperfine transitions between the levels. Due to the large ground state splitting, the spectrum is split into a left and right multiplet. Fig. 5 presents the right multiplet after 3 and 27 scans. The spectra were analyzed with all three cost functions: normal χ^2, the modified formula using $\sqrt{f(x_i)}$ and the Poissonian log-likelihood. The value of the hyperfine parameters was extracted, and compared to the result from the likelihood fit for 27 scans. For the uncertainties, the χ^2 methods made use



(a)

(b)

Fig. 3. Output of `generate_correlation_map` for (a) χ^2 (Listing 1 line 24) and (b) likelihood (Listing 1 line 28) for the toy dataset. The diagonals show the resulting change in χ^2 or likelihood after fixing one parameter and refitting. The 1σ boundaries are also indicated. The 2D contourplot is achieved in the same way with 2 parameters fixed.

of the Hessian matrix approach while the likelihood calculation found the $\Delta\mathcal{L} = 1/2$ edges. Since the calculated uncertainties are comparable, no parameter sampling was performed. Considering

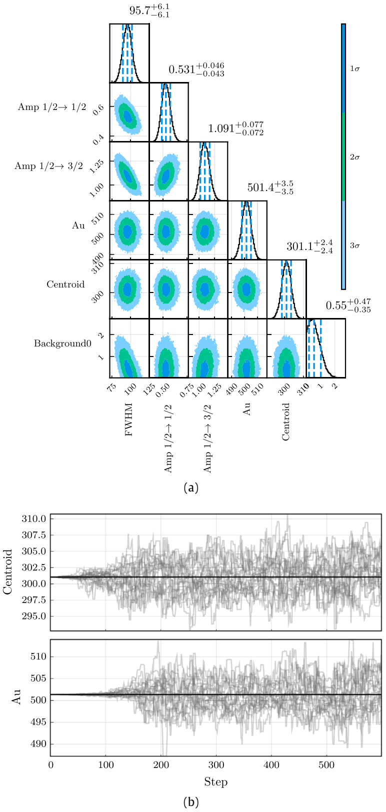


Fig. 4. (a) The result of a walk of 12 000 steps with 20 walkers. The 16, 50 and 84 percentiles are indicated on the diagonal plots and are used as value and uncertainty estimates. The 2D histogram on the lower left shows the same correlation as seen in Fig. 3. (b) The values of the first 600 steps taken by the 20 walkers for the Centroid and Au parameters. A small burn-in is visible in the beginning, where all the walkers start from a central point and only spread out after some steps. This data represents 5% of the total data, and has been removed to generate (a).

systematic deviations as discussed in Section 2.4, the low values of χ^2_{red} were judged to be due to the low statistics of the dataset and invalidity of the Gaussian model for low count rates. Therefore the uncertainties were not enlarged. The result is given in Fig. 6.

Several striking features are immediately present. The uncertainties as calculated by using Eq. (2) are roughly a factor of 2 smaller than those reported by either the modified formula Eq. (3) or the Poissonian approach Eq. (4). Although the same value is reached after 27 scans for all approaches for the B parameter and the centroid, there is a 1σ deviation for the A-parameters when using the normal χ^2 procedure. The normal formula gives an inconsistent picture, and deviates sharply from all other methods below 20 scans for the other parameters. The disagreement between the modified formula and the Poissonian likelihood is much less and much smoother, with the deviation starting only below 10 scans.

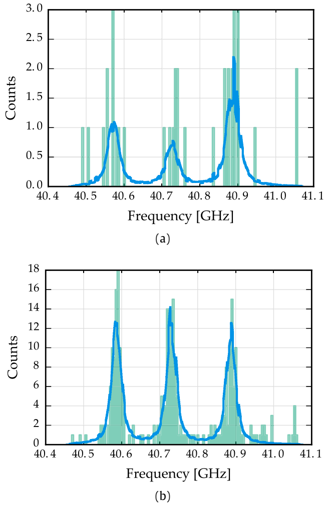


Fig. 5. The right multiplet of ^{203}Fr (green histogram) after (a) 3 scans and (b) 27 scans, along with the likelihood fit (blue line), offset to an arbitrary frequency. The discontinuous shape of the fit is caused by multiplying the model function in each frequency bin by the measurement time, as different times were spent in different bins.

6. Conclusion

Based on the statistical formalism of Maximal Likelihood, the SATLAS package has been written to provide an easy interface with these formulas. The focus on user-friendliness has resulted in easy model definitions and cost function selection. Advanced routines to calculate the uncertainty are available with intuitive interfaces.

SATLAS allows easy and advanced analysis of experimental data with different cost functions, requiring only a minimum of input from the user. The results from the tests show the importance of selecting the correct function as already suggested in 1984 [3], which has an impact on both the value of the parameters extracted and their uncertainty.

From the generated data, we see that the uncertainty calculations are consistent with each other when their individual assumptions are met.

From the CRIS data, it is shown that even at high statistics, the traditional χ^2 formula is incorrect due to the underestimation of the statistical uncertainty on the parameter values. Furthermore, there is a discrepancy between the analysis results obtained for the low and high statistics datasets. This inconsistency is removed when using the modified χ^2 equation, using the model value to estimate the uncertainty on the counted number. At lower statistics, the difference between the modified equation and likelihood is negligible until a very low number of counts is reached. Between these two approaches, the modified equation is therefore preferred

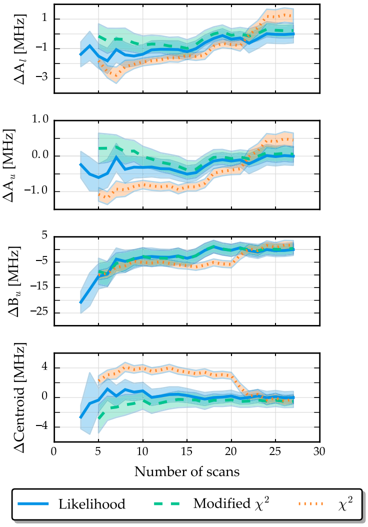


Fig. 6. Overview of the deviation of the fit parameters from the optimal likelihood value as function of the total number of events in the spectrum. The transparent band represents the calculated uncertainty. See text for details.

where feasible due to the lower computational cost. When the spectrum only contains very few counts however, the likelihood equation must be used in the minimization procedure.

Acknowledgments

We thank Professor Thomas Cocolios for fruitful discussions. This work was supported by the BriX Research Program No. P7/12 and FWO-Vlaanderen (Belgium) and GOA 15/010 from KU Leuven and the Science and Technology Facilities Council, UK.

References

[1] NuPECC Long Range Plan 2010: Perspectives of Nuclear Physics in Europe, 2010. available at https://nupecc.org.
[2] G.F. Knoll, Radiation Detection and Measurement, fourth ed., John Wiley & Sons, 2010, p. 830.
[3] S. Baker, Nucl. Instrum. Methods Phys. Res. 221 (2) (1984) 437–442. http://dx.doi.org/10.1016/0167-5087(84)90016-4.
[4] P.R. Bevington, D.K. Robinson, Data Reduction and Error Analysis for the Physical Sciences, third ed., 2003.
[5] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes 3rd Edition: The Art of Scientific Computing, third ed., Cambridge University Press, 2007.
[6] M. Clutton-Brock, Technometrics 9 (2) (1967) 261–269.
[7] R. Brun, F. Rademakers, Nucl. Instrum. Methods Phys. Res. A 389 (1997) 81–86.
[8] C. Hsiao, J. Econometrics 41 (1) (1989) 159–185. http://dx.doi.org/10.1016/0304-4076(89)90047-X.
[9] E.T. Jaynes, IEEE Trans. Syst. Sci. Cybern. 4 (3) (1968) 227–241. http://dx.doi.org/10.1109/TSSC.1968.300117.
[10] F. James, Comput. Phys. Comm. 20 (1) (1980) 29–35. http://dx.doi.org/10.1016/0010-4655(80)90103-4.
[11] B.P. Carlin, T.A. Louis, Bayesian Methods for Data Analysis, third ed., 2008.
[12] J. Beringer, J.F. Arguin, R.M. Barnett, K. Copic, O. Dahl, D.E. Groom, C.J. Lin, J. Lys, H. Murayama, C.G. Wohl, W.M. Yao, P.a. Zyla, C. Amsler, M. Antonelli, D.M. Asner, H. Baer, H.R. Band, T. Basaglia, C.W. Bauer, J.J. Beatty, V.I. Belousov, E. Bergren, G. Bernardi, W. Bertl, S. Bethke, H. Bichsel, O. Biebel, E. Blucher, S. Blusk, G. Brooijmans, O. Buchmueller, R.N. Cahn, M. Carena, a. Ceccucci, D. Chakraborty, M.C. Chen, R.S. Chivukula, G. Cowan, G. D'Ambrosio, T. Damour, D. de Florian, a. de Gouvêa, T. DeGrand, P. de Jong, G. Dissertori, B. Dobrescu, M. Doser, M. Drees, D.a. Edwards, S. Eidelman, J. Erler, V.V. Ezhela, W. Fetscher, B.D. Fields, B. Foster, T.K. Gaisser, L. Garren, H.J. Gerber, G. Gerbier, T. Gherghetta, S. Golwala, M. Goodman, C. Grab, a.V. Gritsan, J.F. Grivaz, M. Grünewald, a. Gurtu, T. Gutsche, H.E. Haber, K. Hagiwara, C. Hagmann, C. Hanhart, S. Hashimoto, K.G. Hayes, M. Heffner, B. Heltsley, J.J. Hernández-Rey, K. Hikasa, a. Höcker, J. Holder, a. Holtkamp, J. Huston, J.D. Jackson, K.F. Johnson, T. Junk, D. Karlen, D. Kirkby, S.R. Klein, E. Klempt, R.V. Kowalewski, F. Krauss, M. Kreps, B. Krusche, Y.V. Kuyanov, Y. Kwon, O. Lahav, J. Laiho, P. Langacker, a. Liddle, Z. Ligeti, T.M. Liss, L. Littenberg, K.S. Lugovsky, S.B. Lugovsky, T. Mannel, a.V. Manohar, W.J. Marciano, a.D. Martin, a. Masoni, J. Matthews, D. Milstead, R. Miquel, K. Mönig, F. Moortgat, K. Nakamura, M. Narain, P. Nason, S. Navas, M. Neubert, P. Nevski, Y. Nir, K.a. Olive, L. Pape, J. Parsons, C. Patrignani, J.a. Peacock, S.T. Petcov, a. Piepke, a. Pomarol, G. Punzi, a. Quadt, S. Raby, G. Raffelt, B.N. Ratcliff, P. Richardson, S. Roesler, S. Rolli, a. Romaniouk, L.J. Rosenberg, J.L. Rosner, C.T. Sachrajda, Y. Sakai, G.P. Salam, S. Sarkar, F. Sauli, O. Schneider, K. Scholberg, D. Scott, W.G. Seligman, M.H. Shaevitz, S.R. Sharpe, M. Silari, T. Sjöstrand, P. Skands, J.G. Smith, G.F. Smoot, S. Spanier, H. Spieler, a. Stahl, T. Stanev, S.L. Stone, T. Sumiyoshi, M.J. Syphers, F. Takahashi, M. Tanabashi, J. Terning, M. Titov, N.P. Tkachenko, N.a. Törnqvist, D. Tovey, G. Valencia, K. van Bibber, G. Venanzoni, M.G. Vincter, P. Vogel, a. Vogt, W. Walkowiak, C.W. Walter, D.R. Ward, T. Watari, G. Weiglein, E.J. Weinberg, L.R. Wiencke, L. Wolfenstein, J. Womersley, C.L. Woody, R.L. Workman, a. Yamamoto, G.P. Zeller, O.V. Zenin, J. Zhang, R.Y. Zhu, G. Harper, V.S. Lugovsky, P. Schaffner, Phys. Rev. D 86 (1) (2012) 010001. http://dx.doi.org/10.1103/PhysRevD.86.010001.
[13] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, J. Chem. Phys. 21 (6) (1953) 1087–1092. http://dx.doi.org/10.1063/1.1699114.
[14] D. Foreman-Mackey, D.W. Hogg, D. Lang, J. Goodman, Publ. Astron. Soc. Pac. 125 (925) (2013) 306. http://stacks.iop.org/1538-3873/125/i=925/a=306.
[15] R. Andrae, T. Schulze-Hartung, P. Melchior, Dos and don'ts of reduced chi-squared (2010) 1–12. http://arxiv.org/abs/1012.3754.
[16] R. Andrae, Error estimation in astronomy: A guide (2010) 1–23 https://arxiv.org/abs/1009.2755.
[17] M. Newville, T. Stensitzki, D.B. Allen, A. Ingargiola, (2014) LMFIT: Non-linear least-square minimization and curve-fitting for python, http://dx.doi.org/10.5281/zenodo.11813.
[18] C. Schwartz, Phys. Rev. 97 (2) (1955) 380–395. http://dx.doi.org/10.1103/PhysRev.97.380.
[19] T. Skwarnicki, A Study of the Radiative Cascade Transitions between the Upsilon-Prime and Upsilon Resonances (Ph.D thesis), University of Cracow, 1986, p. 106.
[20] A.L. Stancik, E.B. Brauns, Vib. Spectrosc. 47 (2008) 66–69. http://dx.doi.org/10.1016/j.vibspec.2008.02.009.
[21] P. Magnante, H. Stroke, J. Opt. Soc. Amer. 59 (7) (1969) 836–841. http://www.osti.gov/scitech/biblio/4781263.
[22] A. Barzakh, Private communication.
[23] S.G. Wilkins, K.M. Lynch, J. Billowes, C.L. Binnersley, M.L. Bissell, T.E. Cocolios, T. Day Goodacre, R.P. de Groote, G.J. Farooq-Smith, K.T. Flanagan, S. Franchoo, R.F. Garcia Ruiz, W. Gins, H. Heylen, Á. Koszorús, G. Neyens, H.H. Stroke, A.R. Vernon, K.D.A. Wendt, X.F. Yang, Phys. Rev. C 96 (2017) 034317. http://dx.doi.org/10.1103/PhysRevC.96.034317.
[24] K.M. Lynch, J. Billowes, M.L. Bissell, I. Budinčević, T.E. Cocolios, R.P. De Groote, S. De Schepper, V.N. Fedosseev, K.T. Flanagan, S. Franchoo, R.F. Garcia Ruiz, H. Heylen, B.A. Marsh, G. Neyens, T.J. Procter, R.E. Rossel, S. Rothe, I. Strashnov, H.H. Stroke, K. D.A. Wendt, Phys. Rev. X 4 (2014) 011055. http://dx.doi.org/10.1103/PhysRevX.4.011055.
[25] R.P. de Groote, I. Budinčević, J. Billowes, M.L. Bissell, T.E. Cocolios, G.J. Farooq-Smith, V.N. Fedosseev, K.T. Flanagan, S. Franchoo, R.F. Garcia Ruiz, H. Heylen, R. Li, K.M. Lynch, B.A. Marsh, G. Neyens, R.E. Rossel, S. Rothe, H.H. Stroke, K.D.A. Wendt, S.G. Wilkins, X. Yang, Phys. Rev. Lett. 115 (2015) 132501. http://dx.doi.org/10.1103/PhysRevLett.115.132501.

5.2.2 Rate equation implementation

The rate equations, as described in Chapter 2, can be used in a straightforward fashion if the only goal is simulation. However, when using the rate equations to analyze data and extract parameters, their implementation must allow easy access to several of the values used in the calculations. To this end, an extension of the SATLAS BaseModel was made which allows direct access to several parameters.

In the implementation (source code given in Appendix A.1), the following values in the rate equations can be adjusted during fitting routines:

- For the splittings, each level has three hyperfine parameters (A, B and C, the latter is the octupole interaction and is fixed to zero) and a centroid to control the location. The magnetic field in the interaction region, which results in a small Zeeman splitting, is also included.
- The lineshape of the laser interaction cross section is chosen to be either a Voigt shape or a Lorentzian shape. In the first case, two parameters control the Lorentzian and Gaussian widths, in the latter case only a single parameter is used.
- The laser intensity for each interacting laser can also be used as a parameter. A single purity factor, describing how much of an admixture of σ^+ and σ^- light is present, is used for all lasers. Considering the laser interaction matrix $B\rho\left(\nu\right)$ as introduced in Chapter 2, the admixture is used to replace this matrix by the weighted average of the matrices for σ^+ and σ^- light:

$$aB\rho\left(\nu\right)^{\sigma^+} + \left(1-a\right)B\rho\left(\nu\right)^{\sigma^-} \tag{5.1}$$

- The whole spectrum is scaled by a general multiplicative factor, which incorporates both changes in unit (e.g. processing the data to give an asymmetry between −100% and 100% instead of between −1 and 1) and general losses.
- Added to the spectrum is the modeling of the background, which is done by adding a first order polynomial.

In order to reach acceptable computation times for the fitting routine, as well as handle the correlations between several parameters such as the interaction time and the scaling, only a few parameters were left free. Others were either restricted to their literature/measured value or expressions were given to define a relationship between several parameters. The following restrictions were implemented:

- The hyperfine parameters were fixed to the literature values to fix the distance between the peaks.
- The interaction length of 1.6 m and the kinetic energy of the beam defines the interaction time, which is also kept fixed.
- When spectra were analyzed where the AOM beams were present, the ratio of the intensity to the main beam was known from measurements with a power meter. This ratio was then used to constrain the value of the intensity of the additional laser beams.
- The lineshape was chosen to be Lorentzian with the linewidth fixed to the natural linewidth. Broadening of the profiles due to a high laser intensity will be used to match the observed linewidths.

This leaves 6 parameters for the fit routine to optimize: the centroid of the spectrum, the laser intensity and purity of the applied laser polarization, the scaling of the entire spectrum and the two parameters for the linear background model. As this approach produces fits that match the experimental spectra and the information of interest is contained in the scaling parameter, this provided the optimal combination of computational efficiency and accuracy.

5.2.3 Weighted average

After parameter estimates have been made for multiple spectra, the resulting weighted average is calculated as

$$\langle x \rangle = \frac{\sum_i \frac{x_i}{\sigma_i^2}}{\sum_i \frac{1}{\sigma_i^2}}. \tag{5.2}$$

The statistical uncertainty on this estimated mean is given by

$$\sigma_{stat}^2 = \frac{1}{\sum_i \frac{1}{\sigma_i^2}}. \tag{5.3}$$

This uncertainty can be subject to over- or underestimation due to an incorrect estimate of the uncertainty on the data points. As argued in the preceding paper, such a correction is made by enlarging the uncertainties such that $\chi_{red}^2 = 1$. Instead of enlarging the individual uncertainties, the statistical uncertainty can be enlarged:

$$\sigma^2 = \sigma_{stat}^2\chi_{red}^2 = \frac{1}{\sum_i \frac{1}{\sigma_i^2}} \frac{1}{N-1} \sum_i \left(\frac{x_i - \langle x \rangle}{\sigma_i}\right)^2 . \tag{5.4}$$

This correction factor is only applied when $\chi_{red}^2 > 1$, which is the most conservative choice in the calculated uncertainty. The weighted average implementation available in SATLAS uses the conservative choice.

5.3 Isotope shifts

5.3.1 Predicting the isotope shift of ^{35}Ar

Prior to the β-asymmetry measurements on ^{35}Ar, an estimate of the isotope shift $\delta\nu_{811}^{40,35}$ had to be made, in order to determine the location of the ^{35}Ar hyperfine spectrum in frequency space. This is possible by using the measured isotope shifts of ^{35,36,38,40}Ar in the 764 nm transition [52]. With the *King plot method*, a prediction can then be made for the isotope shift $\delta\nu_{811}^{40,35}$ in the 811 nm line used in this study.

The King plot method plots modified isotope shifts of different transitions against each other. The isotope shift (Eq. (2.3)) can be rewritten to give the mean square charge radius:

$$\delta\left\langle r_n^2\right\rangle^{A,A'} = \frac{1}{F_i}\left(\delta\nu_i^{A,A'} - M_i\frac{m_{A'} - m_A}{m_{A'}m_A}\right) = \frac{1}{F_i}\left(\delta\nu_i^{A,A'} - \frac{M_i}{\mu^{A,A'}}\right) \tag{5.5}$$

where the subscript refers to a specific transition. Since this holds for all transitions, the isotope shifts in another line are given by:

$$\mu^{A,A'}\delta\nu_j^{A,A'} = \frac{F_j}{F_i}\mu^{A,A'}\delta\nu_i^{A,A'} + M_j - \frac{F_j}{F_i}M_i, \tag{5.6}$$

so there is a linear relationship between the scaled isotope shifts of two transitions.

By measuring the isotope shifts $\delta\nu_{811}^{40,36}$ and $\delta\nu_{811}^{40,38}$ and plotting against the value measured in the 764 nm line, the linear relationship is obtained. From this relationship, $\delta\nu_{764}^{40,35}$ can be used to estimate $\delta\nu_{811}^{40,35}$.

5.3.2 Isotope shift measurements for ^{36,38}Ar relative to ^{40}Ar

Optical spectra of the stable isotopes were obtained by performing bunched beam collinear laser spectroscopy. The optical detection region is a copy of the design for COLLAPS [63]. In the laser polarization line, two (instead of four) 9658B photomultiplier tubes (PMTs) from Electron Tubes were installed opposite to each other in the horizontal plane of the beamline. These PMTs are used to detect infrared photons emitted from the resonantly excited Ar atoms. They have a quantum efficiency of approximately 2% for photons of 811 nm. As they measure infrared signals, thermal emissions generate a high dark count rate of $25 \times 10^3\,\mathrm{s}^{-1}$. This is reduced by almost a factor 100 by cooling the PMTs to −10 °C using a circulator filled with ethanol, resulting in a dark count rate of approximately $300\,\mathrm{s}^{-1}$.

The 811 nm light was produced by a Matisse TS2 actively stabilized Titanium:Sapphire (Ti:Sa) laser. Part of the light output was coupled to a HighFinesse WSU wavemeter. The readout of this wavemeter was used to lock the frequency of the light to a certain setpoint through a Proportional-Integral-Derivative (PID) loop.

Figure 5.5 shows representative spectra of all three isotopes, where an additional feature compared to the simulations in Chapter 2 is immediately visible. The spectra contain asymmetric peaks, which is the result of collisions in the K vapor used to neutralize the ion beam [64]. These collisions lower the kinetic energy of the particle beam, inducing a shift towards a lower frequency. The probability of such a collision occurring is very small, yet there are a lot of

Table 5.2: Overview of laser setpoints and amount of spectra available for isotope shift determination for the different isotopes.

	^{40}Ar	^{38}Ar	^{36}Ar
PID setpoint [cm^{-1}]	12 339.207 240	12 339.725 393	12 340.284 316
Independent spectra	6	5	7
Independent couples	-	5	3

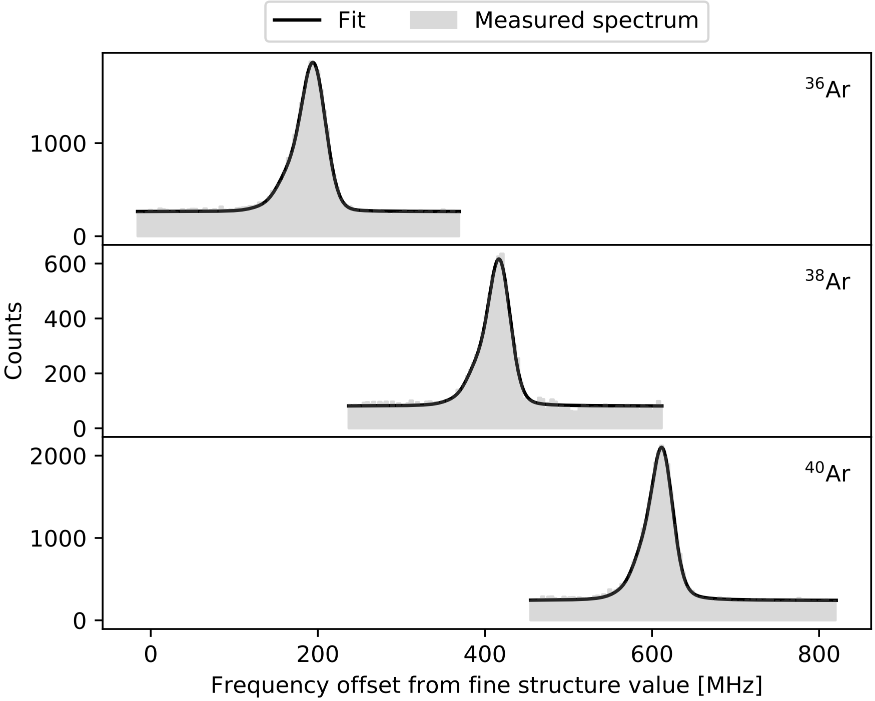


Figure 5.5: Representative measured optical spectra of the stable isotopes, relative to the fine structure frequency given in literature.

particles that could undergo collisions. Therefore, the probability of undergoing k collisions is modeled as a Poisson distribution:

$$P(k) = \frac{\lambda^k \exp(-\lambda)}{k!} \tag{5.7}$$

In the model function defined in SATLAS, this is implemented by including N additional sidepeaks with both the constant spacing between peaks and the intensity as parameters. The relative intensity of the kth sidepeak is calculated as

$$\frac{P(k)}{P(0)} = \frac{\lambda^k \exp(-\lambda)}{k!} \frac{0!}{\lambda^0 \exp(-\lambda)} = \frac{\lambda^k}{k!}. \tag{5.8}$$

Comparing fits with different numbers of sidepeaks (see Figure 5.6), the minimal χ^2_{red} is reached with 4 sidepeaks and no improvements were found with more.

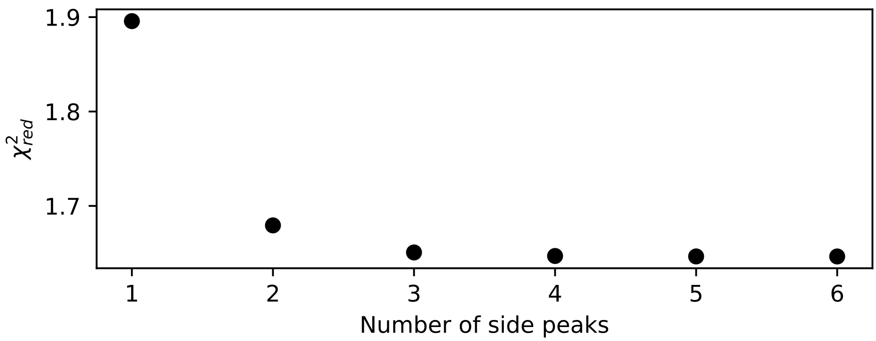


Figure 5.6: Fitting the spectra of ^{40}Ar with a different number of sidepeaks shows a saturated χ^2_{red} value reached at 4 sidepeaks.

Isotope shift

The optical spectra were analyzed by using a χ^2 minimization routine with a hyperfine spectrum as the model function, where the centroid is one of the fit parameters. This was done using the standard implementations for a hyperfine spectrum and χ^2 calculation as defined in the SATLAS fitting library. The results are presented in Table 5.3.

In order to calculate the isotope shift relative to ^{40}Ar, each centroid of ^{36,38}Ar needs to be coupled to a ^{40}Ar centroid. To avoid time related drifts in the centroids, only the ^{40}Ar centroids closest in time to the ^{36,38}Ar centroids are

Table 5.3: The extracted centroids for the stable isotopes, chronologically ordered. The coupling is presented graphically in Figure 5.7.

Measurement number	Centroid		
	^{36}Ar [MHz]	^{38}Ar [MHz]	^{40}Ar [MHz]
1	193.7(9)	420.6(9)	615.7(15)
2	193.8(3)	415.1(5)	610.9(4)
3	195.5(6)	417.3(6)	610.7(3)
4	194.7(4)	415.7(4)	611.4(3)
5	194.8(7)	414.6(5)	612.4(5)
6	192.0(2)	-	611.7(3)
7	194.7(4)	-	-

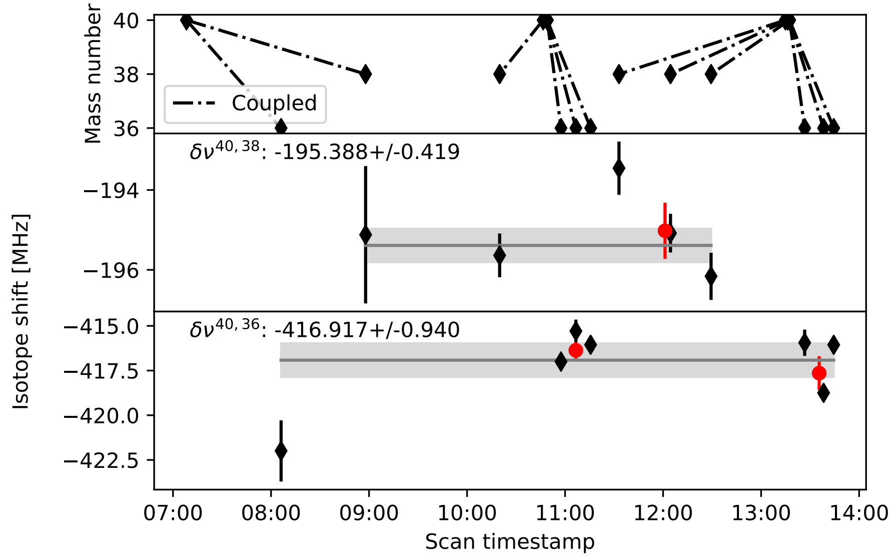


Figure 5.7: The coupling of the individual spectra (top plot) allows correlations between isotope shifts sharing a reference spectrum. This is compensated by calculating the weighted average of the isotope shifts that share the same reference value before calculating the global weighted average. The red dot represents the weighted average of the measurements around that time sharing the same reference value. The resulting isotope shifts (bottom plots) agree with constant values. Note that there is one ^{40}Ar spectrum around 7:00, 2 around 11:00 and three around 13:30.

Table 5.4: The deduced isotope shifts, ordered chronologically. The final row is the weighted average of the numbers above.

$\delta\nu_{811}^{40,36}$ [MHz]	$\delta\nu_{811}^{40,38}$ [MHz]
$-422(2)$	$-195.6(5)$
(red dot) $-417.6(9)$	$-195.1(17)$
(red dot) $-416.4(5)$	(red dot) $-195.0(7)$
$-416.9(9)$	$-196.0(5)$

used and reported in Table 5.3. Figure 5.7 shows the isotope shifts as a function of time. The upper panel shows at which time a spectrum of each isotope is taken, and the coupling is indicated with dashed lines. The middle and lower panels shows the calculated isotope shifts and weighted average values for respectively ^{38}Ar and ^{36}Ar. The results are also given in Table 5.4.

5.3.3 ^{35}Ar isotope shift from King plot

The King plot method is then used with the data from Ref. [52] to predict $\delta\nu^{40,35}$. In Figure 5.8 the King plot is shown, with the two correlated isotope shifts. Since a straight line fit through two data points gives a uniquely defined solution, conventional methods cannot be used to estimate the uncertainty on the prediction. This was circumvented by performing a Markov Chain Monte Carlo random walk through the parameter space and using those estimated distributions of slopes and intercepts to calculate the distribution of the scaled isotope shift for ^{35}Ar (see Figure 5.8). Note that the uncertainties on $\delta\nu_{763}$ and μ have been ignored.

The estimated isotope shift $\delta\nu_{811}^{40,35}$ is then calculated to be $-530(1)$ MHz, corresponding to a laser wavenumber of 12 340.583 43 cm^{-1}. This value has been used in the subsequent β-asymmetry measurements of the hyperfine spectrum of ^{35}Ar (see Section 5.4 for details). From these spectra, the experimental isotope shift $\delta\nu_{811}^{40,35}$ relative to the optical hyperfine spectra of ^{40}Ar can be extracted. The result is also shown in Figure 5.8, but will be discussed in Section 5.4.5.

5.4 Measurement of β-asymmetry hyperfine spectra

β-asymmetry spectra are gathered by implanting a radioactive ensemble in a crystal and measuring the subsequent radioactive β-decay as a function of the acceleration voltage. As this acceleration voltage changes the kinetic energy of the radioactive beam, the laser frequency as observed by the radioactive beam is also changed through a different Doppler shift. The conversion from voltage to frequency is done by calculating

$$\nu_{obs} = \nu_{rest}\sqrt{\frac{1-\beta}{1+\beta}} \tag{5.9}$$

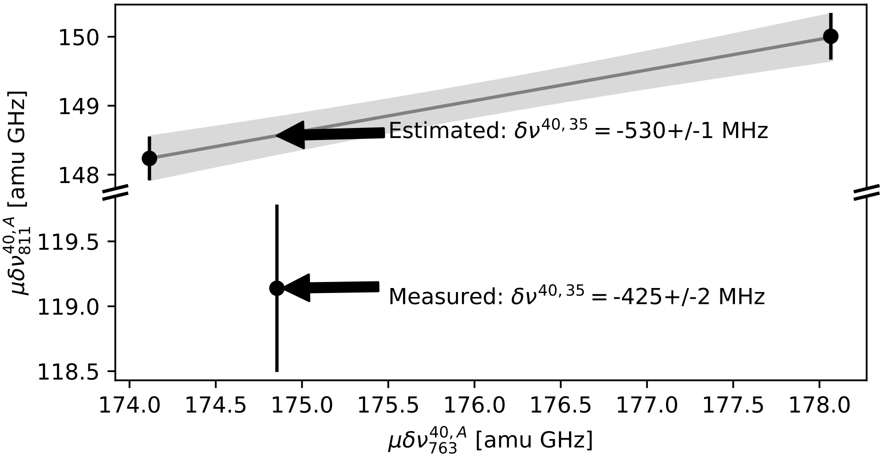


Figure 5.8: The Kingplot of both transitions predicts an isotope shift for ^{35}Ar with a precision on the order of 1 MHz. The measured value however differs significantly (note the broken y-axis), most probably due to retuning of the ISCOOL platform voltage.

$$\beta = \sqrt{1 - \left(\frac{mc^2}{mc^2 + qE_{kin}}\right)^2} \tag{5.10}$$

In the plotting of the spectra, the frequency offset relative to the value of the fine structure transition frequency is used.

The experimental asymmetry is calculated as

$$A_{exp} = \frac{N\left(0°\right) - N\left(180°\right)}{N\left(0°\right) + N\left(180°\right)} = \epsilon AP, \tag{5.11}$$

where N denotes the number of counts detected parallel and antiparallel to the magnetic field (and polarization) axis. This experimental asymmetry corresponds to the product of the asymmetry parameter A and the degree of polarization P up to an efficiency factor ϵ. This experimental asymmetry as a function of laser frequency is then compared to the calculated hyperfine polarization spectra as described in Chapter 3. All spectra shown here have been recorded using a σ^+ polarization of the laser light.

The 0° and 180° detectors consisted of an inner and an outer scintillator, with the aim to record coincidence counts in order to reduce the effect of dark counts. Unfortunately, only the signal of the inner detectors could be used for analysis, as the outer scintillator on the 0° side registered a number of counts that was an order of magnitude less than the other detectors. Thus no coincidence counts could be used for the analysis. Instead, the detector thresholds were set to cut the low-energy background part of the β-spectrum.

Unlike the optical spectra, ISCOOL was not used to bunch the beam as the resulting instantaneous decay rate could saturate the β-detectors. Instead, as the release curve in Figure 5.1 shows a release of ^{35}Ar over a period of about 0.7 s, the radioactive decay is observed for 1 s after proton impact. Thus, there is no signal loss originating from the high instantaneous rate. Additionally, as the halflife of ^{35}Ar is 1.78 s, the spacing between two consecutive proton pulses was kept to a minimum of 4.8 s. This reduced the influence from the ^{35}Ar decay from a previous proton pulse.

The ensemble of ions that is stopped in the crystal will experience interactions with the crystal environment, leading to a relaxation of the spin-polarization to the equilibrium value through spin-lattice interactions, mainly dipole and quadrupole interactions [65, 66]. A suitable crystal host minimizes these interactions and maintains the spin-polarization for a long time.

Six crystals have been tested for suitability: KBr, NaCl, KCl, NaF, Si and Pt. Of these, only KBr (cooled below 20 K) has been used as a host for polarized ^{35}Ar before [35]. NaCl and KCl have been tested based on the results of Ref. [67], where the spin-relaxation times of ^{35}Cl have been reported to be 5.2(5) s and 8.5(9) s respectively. Since ^{35}Cl and ^{35}Ar have similar nuclear properties (same spin-parity, magnetic moments of 0.82 and 0.63 μ_N and quadrupole moments of -0.082 b and -0.084 b respectively), similar relaxation times were expected. Si has been selected due to the use of this crystal by the Osaka team for β-NMR experiments in this mass region, while NaF has been used as a catcher for ^{22}Ne, also a noble gas like Ar and thus likely showing similar implantation behaviour [68]. As the host will be cooled, Pt was used to see if the Korringa relaxation, known to be the major relaxation mechanism in metals, was sufficiently long at low temperatures.

SRIM simulations of a 50 keV ^{35}Ar beam (see Table 5.5) show that the implantation depths range from 19 nm (Pt) up to 130 nm (NaF). As these depths are quite small, the surface quality of the crystal might have a large influence on the observed β-asymmetry, as well as on the relaxation time. Prior to the measurements, no particular cleaning procedure has been applied, but all crystals were ordered with dimensions of 10x10x0.5 mm with one side "inspection polished".

5.4.1 β-asymmetry spectra in different crystals

In order to determine which crystals maintain the polarization, the polarized ^{35}Ar beam has been implanted in different crystals. Because previous studies for KBr revealed that only below 50 K the polarization is maintained, measurements have been made at temperatures ranging from room temperature down to $T \approx 10$ K. Typical hyperfine spectra are shown in Figure 5.9. In all crystals, an asymmetry was observed at room temperature, except for KBr. Here, we show the spectrum for 10 K.

A closed cycle liquid helium cryocooler was used to vary the temperature from room temperature down to 10 K. For this, a resistive heater was used to control the temperature of the crystal with a PID loop.

Once the crystals in which the asymmetry is maintained had been identified, several studies were performed. We have investigated the influence of the AOM's on the signal strength (Section 5.4.2), measured hyperfine spectra with different laser powers in order to study the laser polarization saturation curve (Section 5.4.3) and for two different crystals (KBr and NaCl) we have varied the external magnetic field strength to construct a decoupling curve (Section 5.4.4).

Furthermore, from all the measured hyperfine spectra, we can also deduce the isotope shift of ^{35}Ar relative to ^{40}Ar. This is discussed in Section 5.4.5.

Table 5.5: SRIM simulations for the implantation depth of a 50 keV ^{35}Ar beam.

	NaF	KCl	NaCl	KBr	Si	Pt
Average depth [nm]	130	122	109	89	60	19

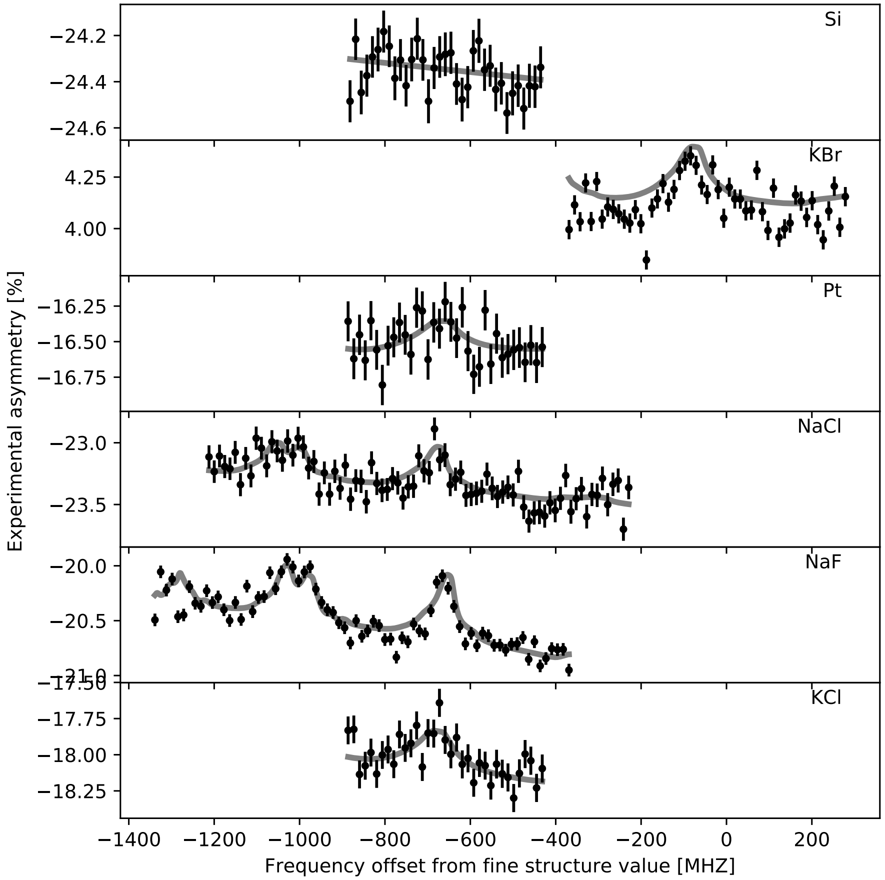


Figure 5.9: Experimental asymmetry as a function of laser frequency for all crystals. Spectra were measured at room temperature in August, except for KBr which was measured at 10 K in April.

5.4.2 Signal gain from multi-frequency pumping

Based on the simulations, the use of the AOMs will result in extra peaks in the spectrum and a higher maximal polarization. From the simulations in Chapter 3 it is seen that an increase of the polarization in the largest peak from 75% to 100% can be achieved by using two AOM's, a gain of a factor 1.33. This has been tested experimentally, by comparing the hyperfine spectrum measured using three correlated laser frequencies induced by the two AOM's (the standard

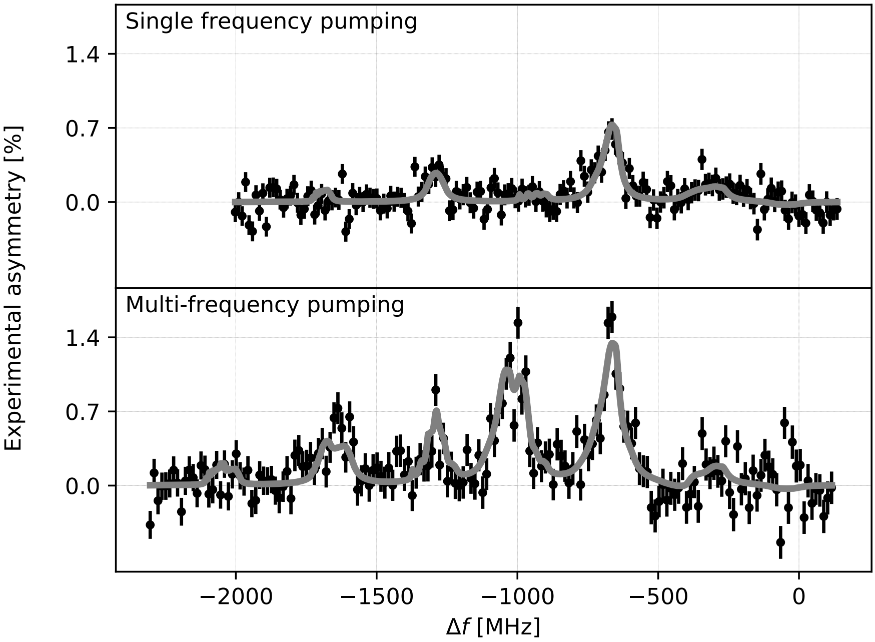


Figure 5.10: β-asymmetry spectra induced by multi-frequency pumping (bottom panel) show additional peaks compared to the spectrum where single-frequency pumping was used (top panel). The fit result from the rate equations (gray line) shows a good agreement to the data for both the single- and multi-frequency pumping. The sloping background as determined by the fit was subtracted in order to get spectra on the same baseline.

measurement method) and then by blocking the two beams coming from the AOM's.

The data are taken for a NaCl crystal at 15 K. The laser power in the single frequency scan and for the three separate laser beams in the multi-frequency scan are restricted to the same value, as measurements with a power meter gave the same power for all three beams at the beamline entrance.

The spectra are shown in Figure 5.10. It is clear that the ratio between the main peaks is close to 2, which is more than the factor 1.33 expected from the simulation. In order to quantify this gain, both datasets were simultaneously fitted with the rate equation model as described in Section 5.2.2. The fit

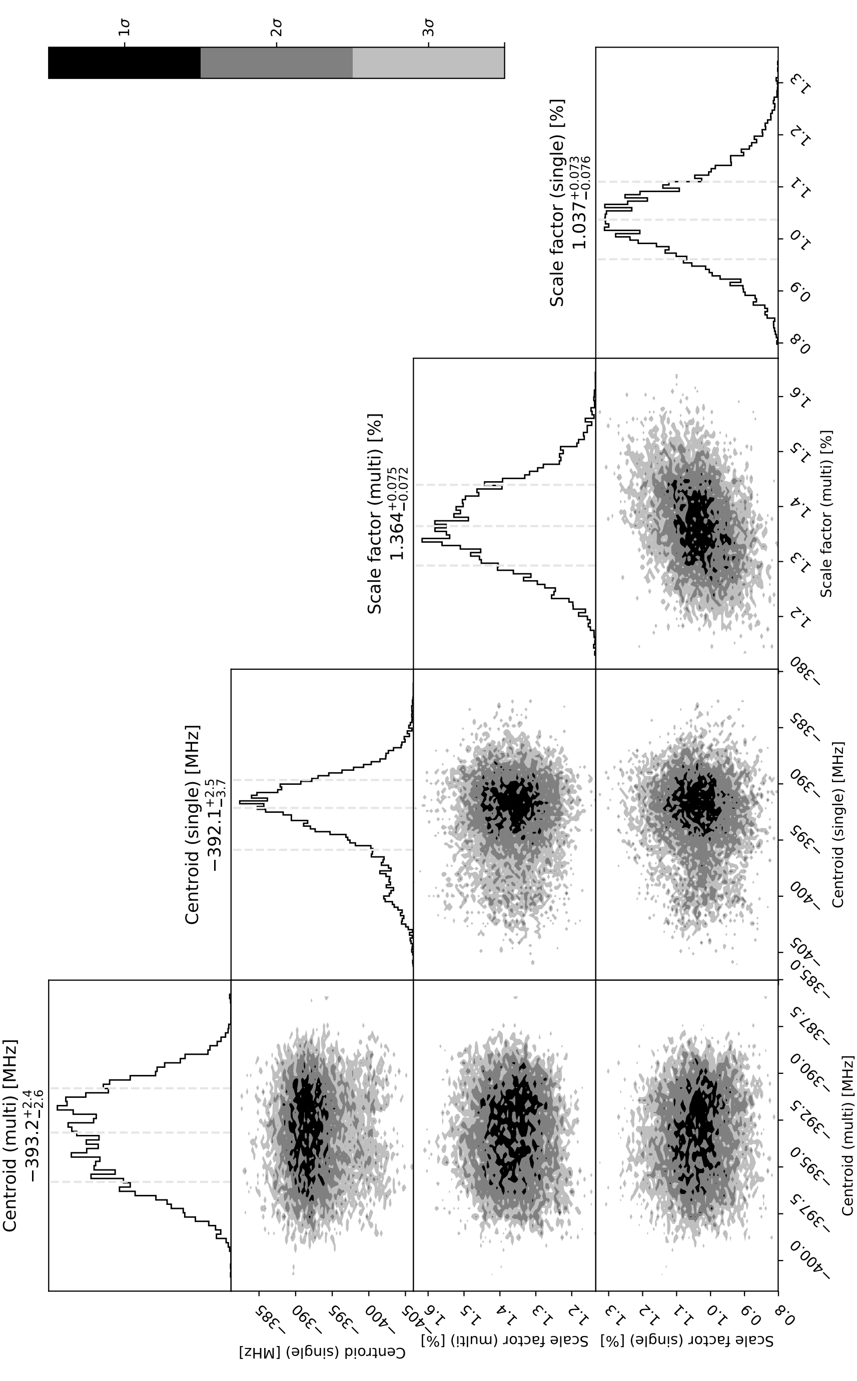


Figure 5.11: The parameter distributions from the simultaneous fit show approximately Gaussian distributions for the centroids and signal heights.

parameters that can reproduce the amplitude of the spectrum are the laser power and two scaling factors, one for each spectrum. These scaling factors should be the same for both spectra (which is not the case, as shown in Figure 5.11), as the laser power is restricted to the same value in all laser beams.

The result of the simultaneous fitting is presented in Figure 5.10. The multi-frequency pumping validates the approach of multiple $B\rho$ matrices in the rate equations, as the shape is well reproduced. The parameter distribution of the simultaneous fit is probed with a random walk. The centroid and the scale factor are fitted individually for the two spectra, as illustrated in Figure 5.11. The scale factor for the multi-frequency scan is larger than 1, which means that the observed gain in polarization is larger than the theoretically predicted value.

The total signal gain can be extracted from the random walk data. For each random walk sample, the asymmetry in the highest peak is calculated for both spectra and the ratio of these values is taken. As each step of the random walk gives correlated values for the parameters, all correlations are propagated correctly by calculating the ratio in this fashion. The distribution quantifies the signal gain as 1.85(3). Compared to the expected gain of 1.33 by going from 75% to 100% polarization, the higher gain suggests that the single-frequency pumping did not fully saturate the transition. Unfortunately, no saturation curve measurements were made for the single frequency pumping to prove this. Note that straightforward calculation with the Gaussian distributions would yield a signal increase of 1.85(17), overestimating the uncertainty due to the unpropagated correlations.

5.4.3 Saturation curve

During the experiment in August, asymmetry spectra with multi-frequency pumping in NaF at room temperature were taken at three different laser powers. The spectra, shown in the insets of Figure 5.12, were analyzed using the random walk method in order to determine the asymmetry in the largest peak. The asymmetry as a function of laser power was then fitted to the function [69]

$$f\left(I;A,S\right)=\frac{AI}{S+I} \tag{5.12}$$

with amplitude A and saturation point S, where I is the laser power delivered into the beamline as measured by the power meter. The laser power that leads to full saturation ($\approx$ 20 mW for the three beam simultaneously) was used for

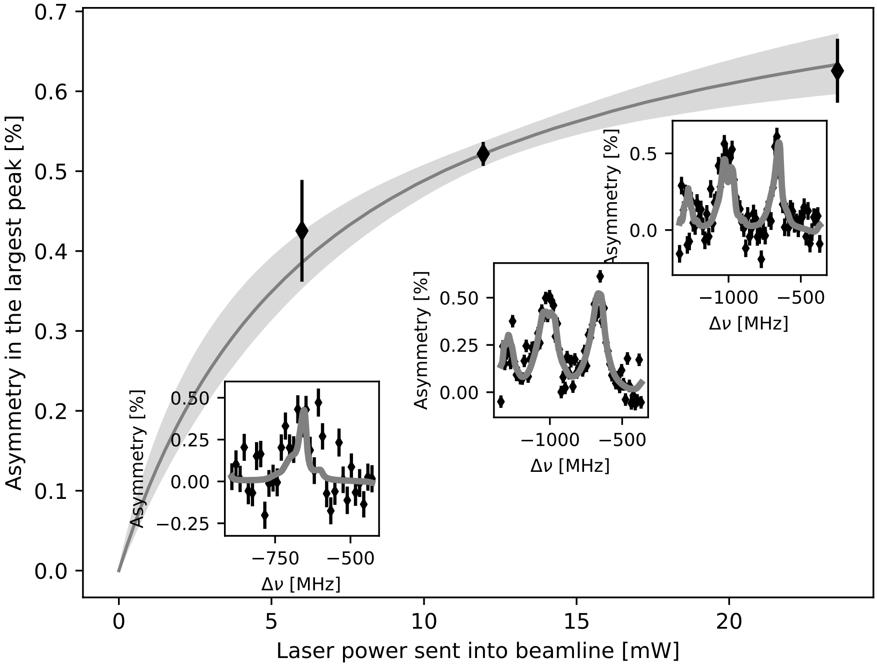


Figure 5.12: The saturation curve (data points extracted from the spectra shown underneath as insets), along with the 1σ uncertainty band, for the 811 nm line.

all other measurements. This means that the power in each individual laser beam was of the order of 7 mW.

The parameter distribution is given in Figure 5.13. The estimated maximal asymmetry is $0.81^{+0.14}_{-0.11}$ % and the saturation point is reached at $6.6^{+3.4}_{-2.6}$ mW. These measurements indicate a nearly saturated transition for the case of multi-frequency pumping. In subsequent measurements, maximal laser power was applied.

5.4.4 Magnetic decoupling curve

For the KBr and NaCl crystals, β-asymmetry spectra with different magnetic fields were taken at a temperature of 12 K. By varying the strength of the field, a “decoupling” curve [70] can be constructed. From the fits to the hyperfine

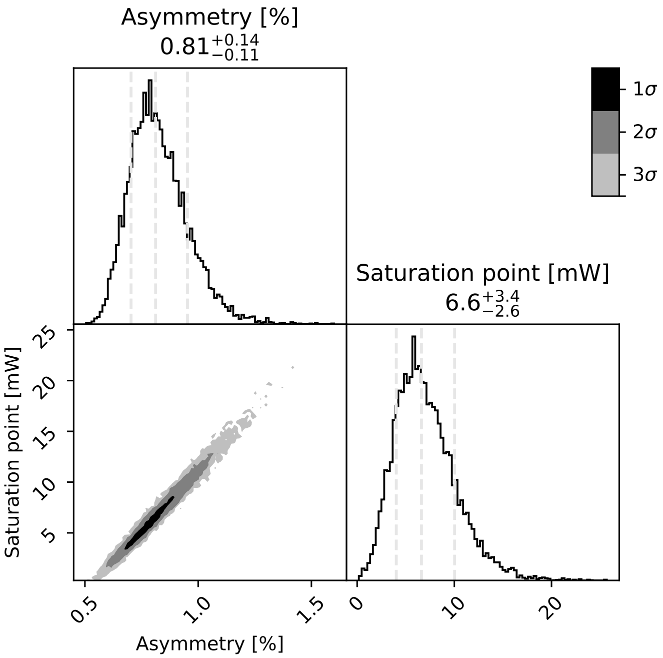


Figure 5.13: The parameters for the saturation curve form highly correlated, non-Gaussian distributions.

spectra, the signal height is extracted as outlined before, and then plotted as a function of applied magnetic field (Figure 5.14). The trend line in these decoupling curves is fitted using a sigmoid function

$$f\left(x; A, s, \mu, b\right) = \frac{A}{1 + \exp\left(-s\left(x - \mu\right)\right)} + b \tag{5.13}$$

with amplitude A, steepness s, location μ and background b. In the analysis, an artificial data point is added to force the function to pass through the origin. Figure 5.14 shows the decoupling behaviour for both crystals.

The data seem to suggest a fully decoupled signal for NaCl already at the lowest applied field strength of 0.2 T, while decoupling is not yet reached for KBr at that point. The sigmoid function is an approximation of the real decoupling curve. The decoupling curve depends on the relative strength of the magnetic and quadrupole interactions that the implanted ^{35}Ar experiences in the host. Even though both crystals have a cubic crystal structure, so no quadrupole

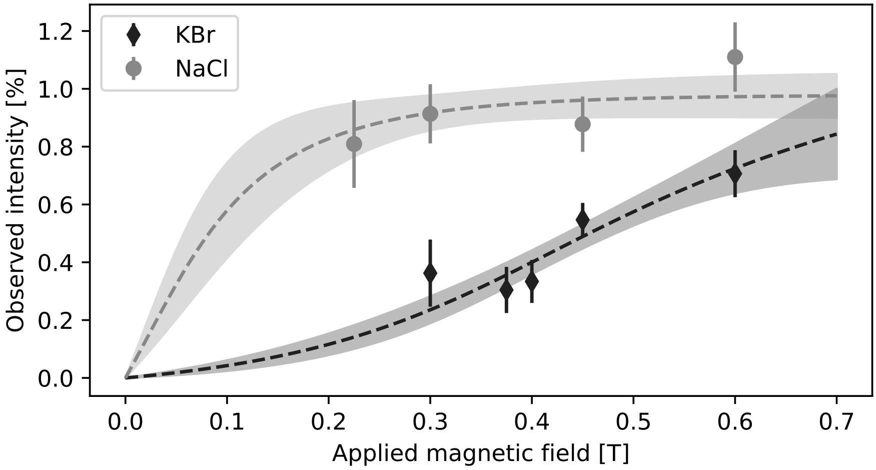


Figure 5.14: Decoupling curves for KBr and NaCl at 12 K.

interaction (leading to a loss of polarization) should be observed, it is known that often small electric field gradients are experienced after ion implantation. Such (defect-associated) quadrupole interaction is different for each host crystal, thus a different static magnetic field strength is needed to decouple the nuclear spin from the quadrupole interaction, resulting in different decoupling curves for both crystals.

The data does suggest that in the case of KBr, a slight gain in the maximal polarization could be achieved by applying a higher field. For all measurements in this work, a field of approximately 0.6 T was applied.

5.4.5 Isotope shift measurement of 35Ar

From the β-asymmetry spectra measured in NaF and KBr (typical spectra shown in Figure 5.9), the centroid value can be used to determine the isotope shift $\delta\nu_{811}^{40,35}$ by measuring the optical spectra of ^{40}Ar. As the hyperfine parameters were fixed to the literature values, even a single peak is sufficient to determine the centroid. As the peak position does not depend on temperature, spectra gathered at temperatures ranging from 12 K up to room temperature have been used. However, the optical measurements use a lot less laser power than the β-asymmetry measurements, as the PMTs would otherwise saturate from

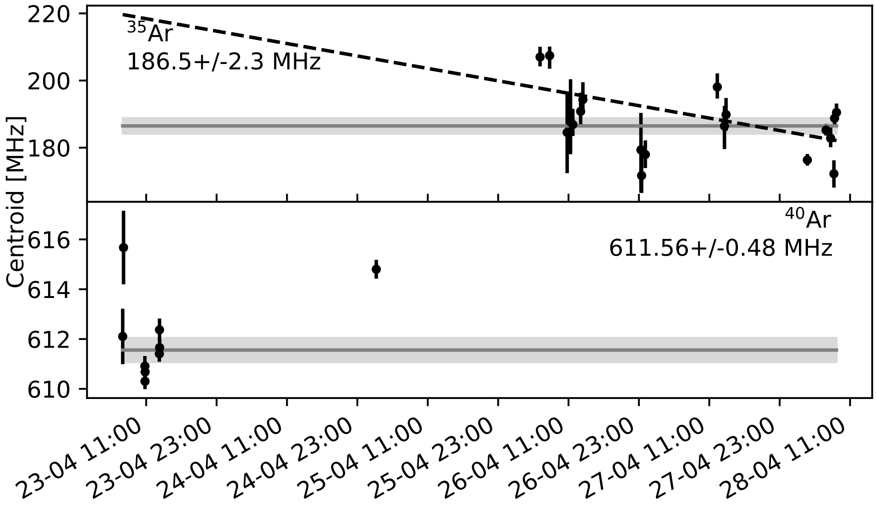


Figure 5.15: The weighted average of the centroid of all the ^{35}Ar (top panel) and ^{40}Ar (bottom panel) spectra. The ^{40}Ar centroids have been measured close together in time and show little drift, while the ^{35}Ar centroids have been taken over a period of three days and show a significant drift of about 30 MHz. One remeasured ^{40}Ar centroid is visible after 24-04 23:00.

scattered light. Therefore, in order to not change the laser setup during the measurements of ^{35}Ar, the measurements of ^{40}Ar were not done very close in time to the ^{35}Ar measurements. The time drifts of the centroids can therefore not be corrected for by coupling different spectra, as was done in Section 5.3.2. Instead, the difference of the weighted average of the centroids for the ^{40}Ar (611.6(5) MHz) and the ^{35}Ar (187(2) MHz) spectra is taken, resulting in an isotope shift of $-425(2)$ MHz.

Another effect that was not corrected for is the change in beam energy. As the potential of the high-voltage platform was modified during the evening of 24-04, the beam energy is different before and after the 24-01 23:00 timestamp in Figure 5.15. In principle, this can be corrected for by recording the platform voltage for every measurement. However, as the extraction of the isotope shift of ^{35}Ar in this transition was not the goal of the experiment, these voltages were not recorded and this effect cannot be corrected for.

From Figure 5.15, where the individual measurements for both ^{35}Ar and ^{40}Ar are shown, the drift is approximated by fitting a straight line through the centroid values of ^{35}Ar. Although the variation of the centroid values is not necessarily linear [71], a linear estimate is the best guess with such irregularly spaced data points. Extrapolation to the time where the measurements of ^{40}Ar were performed gives an estimated systematic uncertainty of 35 MHz. Additional recordings of the voltage could have reduced the slope of the fit and lowered the systematic uncertainty.

When comparing to the estimated value of $-530(1)$ MHz, the difference is almost 100 MHz. However, by applying an additional voltage of 16.5 V to the Doppler transform of the ^{35}Ar spectra, the isotope shift matches the prediction.

This apparent voltage shift can be explained by a combination of two factors:

1. Only one value for the ISCOOL voltage was used in the analysis. The communication with the Agilent digital voltmeter normally used to record this value for every scan was not working, resulting in only one value being noted down. Typical ISCOOL drifts are on the order of 1 V in amplitude for a voltage of 30 kV.

2. The beam tune, including the potential of the high-voltage platform, was changed multiple times during the experiment. The setpoint did change by 10 V between the time when ^{40}Ar and ^{35}Ar were measured, and the readback value from the power supply changed by 14 V.

As the combination of these two effects gives a result of approximately 15 V, it is a possible cause of the observed discrepancy. This illustrates the need for regular reference measurements when using collinear laser spectroscopy to accurately measure isotope shifts.

5.5 Measurement of β-asymmetry as a function of time

Relaxation curves were measured by fixing the acceleration voltage, i.e. fixing the laser frequency in the atoms' frame of reference, such that maximal polarization was generated. As with the β-asymmetry hyperfine spectrum measurements, the beam was not bunched by using ISCOOL but the beamgate was opened for

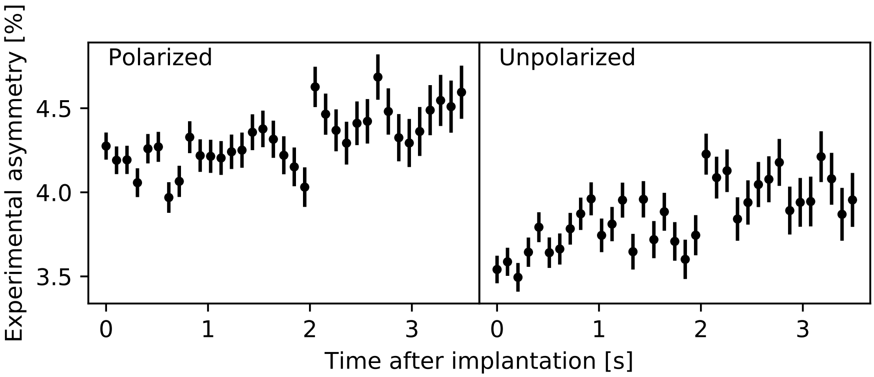


Figure 5.16: Relaxation curve in KBr at a temperature of 50 K. The data on the right represent the baseline asymmetry, which should be a constant at the background level of the left panel.

a short time. Unlike the β-asymmetry hyperfine spectrum measurements, the beamgate was only opened for 0.5 s. The β-activity was recorded as a function of time by integrating the counts in each detector in small timebins of the order of 20 ms. For plotting purposes, this data has been rebinned to have timebins of 200 ms. In the processing of the data, measured activity during the implantation time is ignored and the time after closing the beamgate is used to measure the relaxation curves during the subsequent 4.2 s.

The experimental asymmetry can vary as a function of time due to fluctuations of the beam position, buildup of an impurity layer on the crystal surface, etc. Therefore, in order to determine the laser-induced asymmetry, it is necessary to regularly perform a reference measurement of the experimental non-laser induced asymmetry, in order to measure the background asymmetry level. This is done by performing a measurement of the asymmetry as a function of time, using an unpolarized beam (lasers blocked). Two different methods have been used for this.

For the measurements in April, background asymmetry measurements were made separately after a full measurement of the relaxation curve. Analysis of these data (Figure 5.16) shows that the baseline asymmetry measurements cannot be used to determine the background asymmetry since all of the unpolarized measurements show a slope, artificially boosting the signal measured in the unpolarized spectrum. Therefore, the relaxation time curves from the April

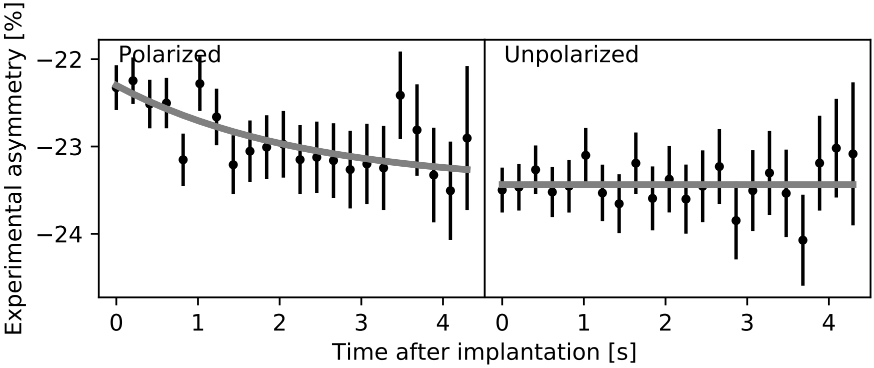


Figure 5.17: Representative relaxation curve in NaCl at a temperature of 60 K. The curve on the right was taken with the laser light blocked and establishes the baseline, while the curve on the left does not have the laser light blocked.

experiment unfortunately did not reveal any consistent results and will not be discussed.

For the data gathered in August, a remote-controlled laser shutter was activated after each second proton pulse to block the laser light. Thus, every second proton pulse, the asymmetry is that of a non-polarized ensemble. This sequence of polarized-unpolarized-polarized implantations allows quasi-simultaneous measurement of the background (non-polarized) β-asymmetry signal. An example of the resulting curves is given in Figure 5.17.

5.5.1 Relaxation time curves as a function of temperature

For NaCl and KCl, the data was saved for each proton pulse as the crystal was being cooled (see Figure 5.18 for the recorded temperature as a function of time) instead of measuring for multiple proton pulses while the crystal is kept at one specific temperature. In order to compensate for the limited statistics generated by a single proton pulse, the results of several proton pulses were summed together to form a single measurement curve. For both crystals, the summation resulted in an averaging over a temperature range of approximately 20 K.

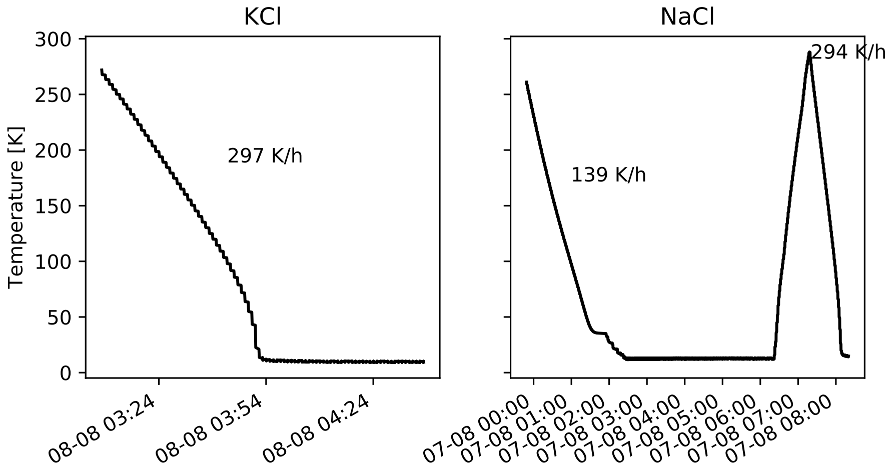


Figure 5.18: The measured temperature from a temperature sensor located near the crystal. The cooling rate was approximately $300\,\mathrm{K\,h^{-1}}$ except for the first cooling curve of NaCl where some thermal contact was made with the environment, resulting in a cooling rate of $139\,\mathrm{K\,h^{-1}}$.

These experimental curves were then fitted to the equation

$$\mathcal{A}\left(t\right) = amp \cdot \exp\left(-\frac{t}{T_1}\ln\left(2\right)\right) + back \tag{5.14}$$

where amp is the implanted asymmetry at $t = 0$, T_1 is the spin-relaxation time and $back$ is the background parameter. The parameter distribution of amp is then used as the extracted asymmetry.

Figure 5.19 shows both the asymmetry extracted from the relaxation curves in NaCl (top panel) and the temperature (bottom panel) as a function of time. The signal height increases as the temperature is lowered. It is also clear that, after an hour at 10 K, the asymmetry in NaCl starts to decrease at a rate of about 0.3% per hour. This can be attributed to a buildup of impurity layers on top of the crystal surface. Since this buildup is comprised of the remaining gas in the beamline, this layer will not form the appropriate crystal structure to maintain the ensemble polarization and asymmetry will therefore be lost when implanting.

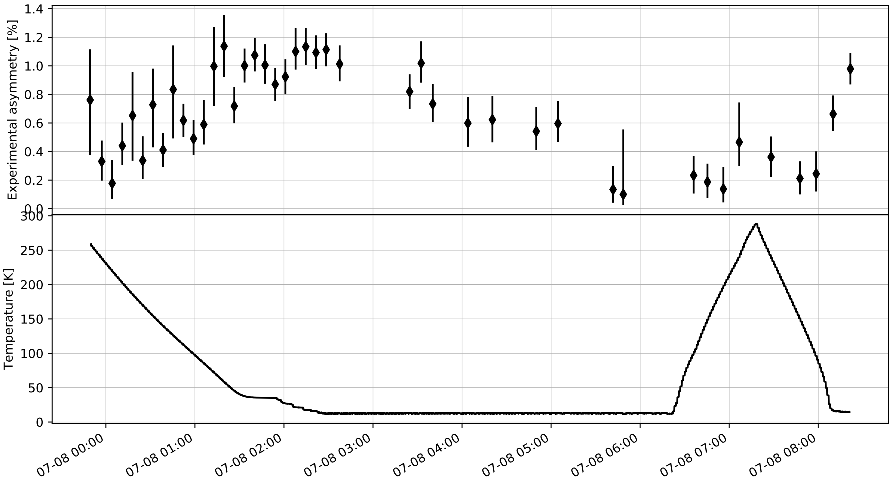


Figure 5.19: The extracted asymmetry from the NaCl relaxation curves as a function of time, along with the temperature of the crystal. After 3:00, when the lowest temperature has been reached for an hour, the asymmetry signal starts degrading. The heating and cooling cycle between 6:00 and 8:00 restores the observed asymmetry signal.

However, the asymmetry can be restored to the previous value by heating the crystal up and cooling it down again. This temperature cycle evaporates the condensed layer on top of the crystal, restoring the surface to the implantation conditions necessary for measuring again. As approximately one hour of measurements can be performed before the signal starts degrading and the heating/cooling cycle takes two hours, a duty cycle of about 30% is present when measuring at cold temperatures.

5.6 Diffusion of ^{35}Ar

Instead of calculating the asymmetry in the two detectors, the counts in both detectors at $t = 0$ can be summed together. This β-intensity gives a measure of the implanted activity. This β-intensity, when plotted as a function of crystal temperature, shows a decrease for higher temperatures. The evolution of the data is generally described by a modified Arrhenius relation:

$$f\left(T; A, Q, b\right) = -A \exp\left(-\frac{Q}{8.617\,330\,3 \times 10^{-5}\,\mathrm{eV\,mol^{-1}\,K^{-1}} \cdot T}\right) + b \qquad (5.15)$$

where A is the amplitude of the curve, b is the saturated value and Q is the activation energy in kJ mol^{-1}. Here, the Arrhenius relation is used to account for the amount of missing β-intensity as a function of temperature, hence the modified equation. As Arrhenius curves of this form are commonly used to model diffusion [72], agreement would suggest that the Ar nuclei are diffusing out of the crystals, which has also been raised as a possible explanation for the low polarization observed in Ref. [35].

For both NaCl and KCl, Eq. (5.15) was used to fit to the observed data, and the results are shown in Figures 5.20 and 5.21 respectively. An activation energy of approximately 55 meV is found for both crystals.

Since radiogenic Ar content is used as a dating method in geology, diffusion of Ar in several rock types has been studied and have been found to have a low activation energy (about 1.3 eV for sodalite ($Na_8(Al_6Si_6O_{24})Cl_2$)), causing significant diffusion losses even at ambient temperatures [73, 74]. Self diffusion in solid argon has an even lower activation energy, which was measured to be 179 meV with theory predicting 165 meV [75], which is due to the even smaller interaction strength of the Ar with the crystal lattice.

The factor of 3 difference between the self diffusion and the obtained results could be explained by taking into account that the Ar-NaCl/KCl diffusion mechanism is different from the Ar-Ar mechanism. For example, Ar experiencing a chemical potential in NaCl/KCl could enhance the diffusion process.

Regardless of the actual mechanism for the loss, room temperature measurements contain only about 80% of the β-counts compared to measurements at low temperatures.

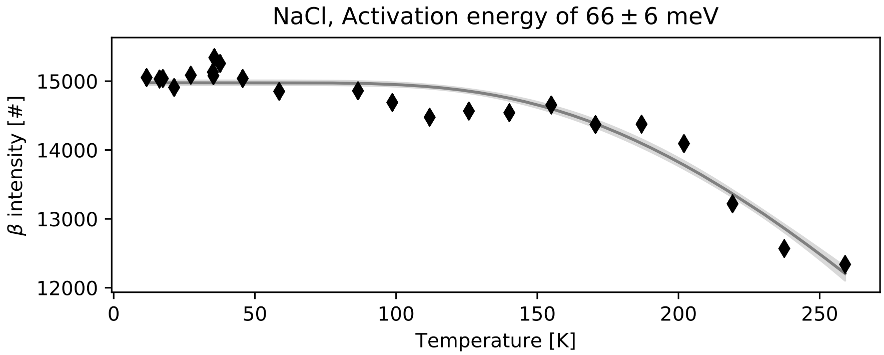


Figure 5.20: The observed activity in the NaCl crystal, plotted as a function of temperature, agrees well with the Arrhenius curve (solid line with 1σ uncertainty band).

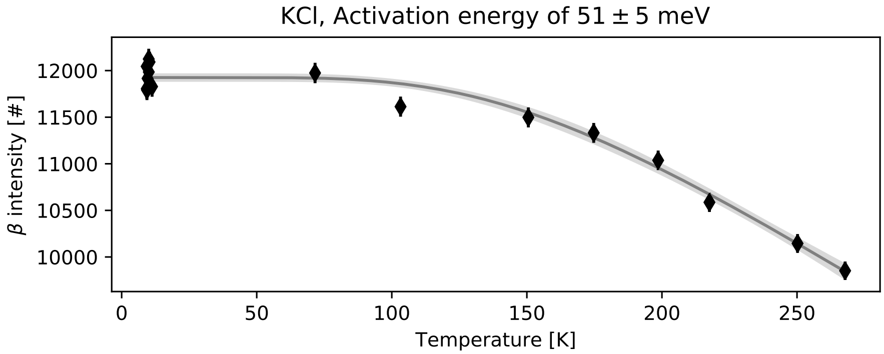


Figure 5.21: The activity in KCl, as in Figure 5.20, behaves according to an Arrhenius curve.

Chapter 6

Interpretation of the data

The deepest solace lies in understanding.
—Nightwish, *Shudder Before The Beautiful*

In this chapter, we discuss the β-asymmetry that has been extracted from the different measurements. The conditions that lead to the highest possible asymmetry are summarized. Furthermore, simulations are presented to estimate which processes could lead to a potential loss of polarization and thus a reduction of the observed β-asymmetry.

6.1 Extraction of β-asymmetry

Although the exact temperature dependence of the spin-relaxation time T_1 depends strongly on the physical processes in the crystal, they share the same global behaviour. Refs. [76, 77] report the spin-relaxation time T_1 is inversely proportional to the temperature. In alkali halides and insulators like NaF, NaCl, KBr and KCl crystals, the relationship is calculated to be of the form [67, 78]

$$T_1 \sim 1/T_L^2, \tag{6.1}$$

where T_L is the lattice temperature. In some cases, intermediate powers between -1 and -2 have also been found. Note that these references did not consider

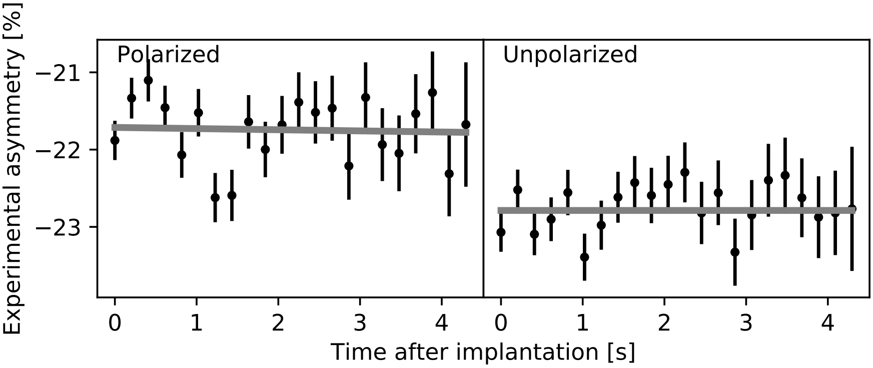


Figure 6.1: Relaxation curve in NaCl at 30 K, showing a near-constant offset between the polarized (left) and non-polarized (right) measurement.

the case of an implanted beam, where the intruder species is not part of the crystal compound and does not necessarily end up in a substitutional location.

As mentioned in Section 5.5.1, the parameter distributions of the fits to the relaxation curves of NaCl and KCl (at a magnetic field of 0.6 T) were explored via the Monte Carlo method. Although for some experimental curves (example shown in Figure 5.17) the relaxation time can be accurately determined, the majority of relaxation curves at low temperatures show a near-constant signal (Figure 6.1). From these curves, the spin-relaxation time cannot be extracted with any quantifiable certainty (Figure 6.2), and only the laser-induced asymmetry can be extracted. This unfortunately means that the temperature dependence of the spin-relaxation time can not be tested to see if it is of the form of Eq. (6.1). However, by observing a near-constant signal in the time range of multiple half-lives of ^{35}Ar (1.78 s) at low temperatures, the spin-relaxation time is guaranteed to be longer than the half-life. Figure 6.3 illustrates that the asymmetries deduced from the hyperfine spectra are consistent with the asymmetries extracted from the relaxation curves.

While the April beamtime shows very promising asymmetries extracted from hyperfine spectra in NaF even at room temperature, the subsequent measurements in August did not confirm these results. This change in observed asymmetry is not fully understood; one possibility is degradation of the crystal surface over the five months between the beamtimes as the same NaF crystal was used and it was not stored in a protective atmosphere. Due to the unreliable

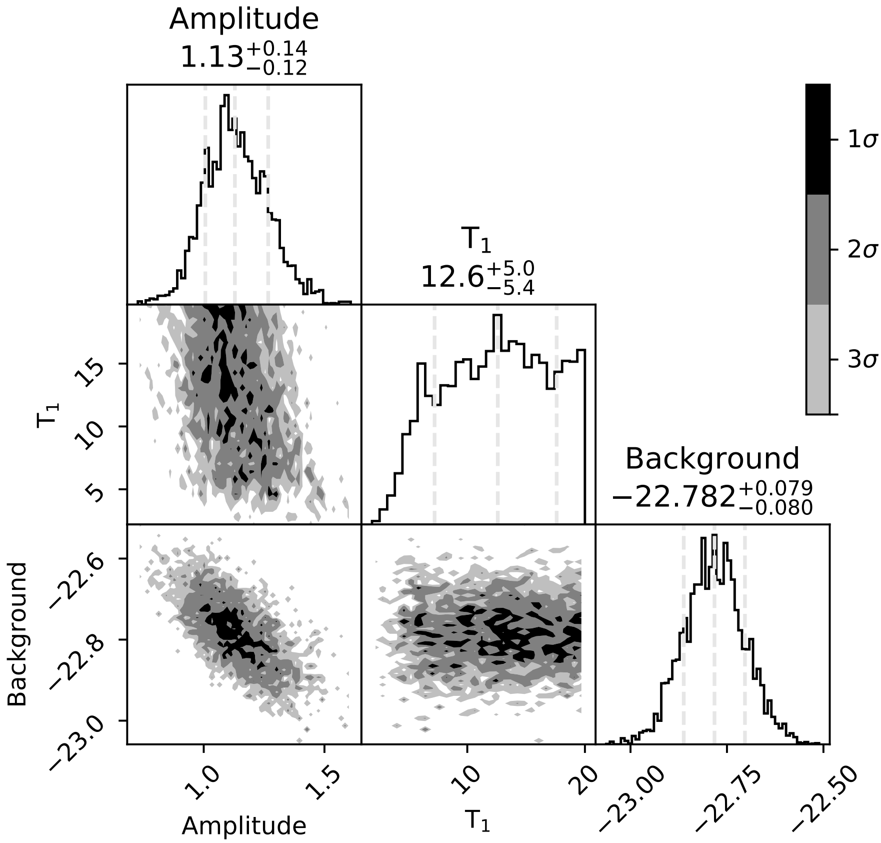


Figure 6.2: Posterior parameter distribution of the fit in Figure 6.1. The relaxation time T_1 is not well defined, while the background and amplitude are.

baseline estimation performed for the relaxation curves in April, these results are not included.

As the first measurements of August did not show any promising asymmetry in NaF, only the properties of KCl and NaCl were investigated down to very low temperatures. These crystals yield asymmetries of 1.16(7) and 1.21(14)% respectively when taking the average asymmetry for the data collected at temperatures < 15K. The observed asymmetry in all crystals is given in Table 6.1, where the small signal in KBr and Pt is very clear.

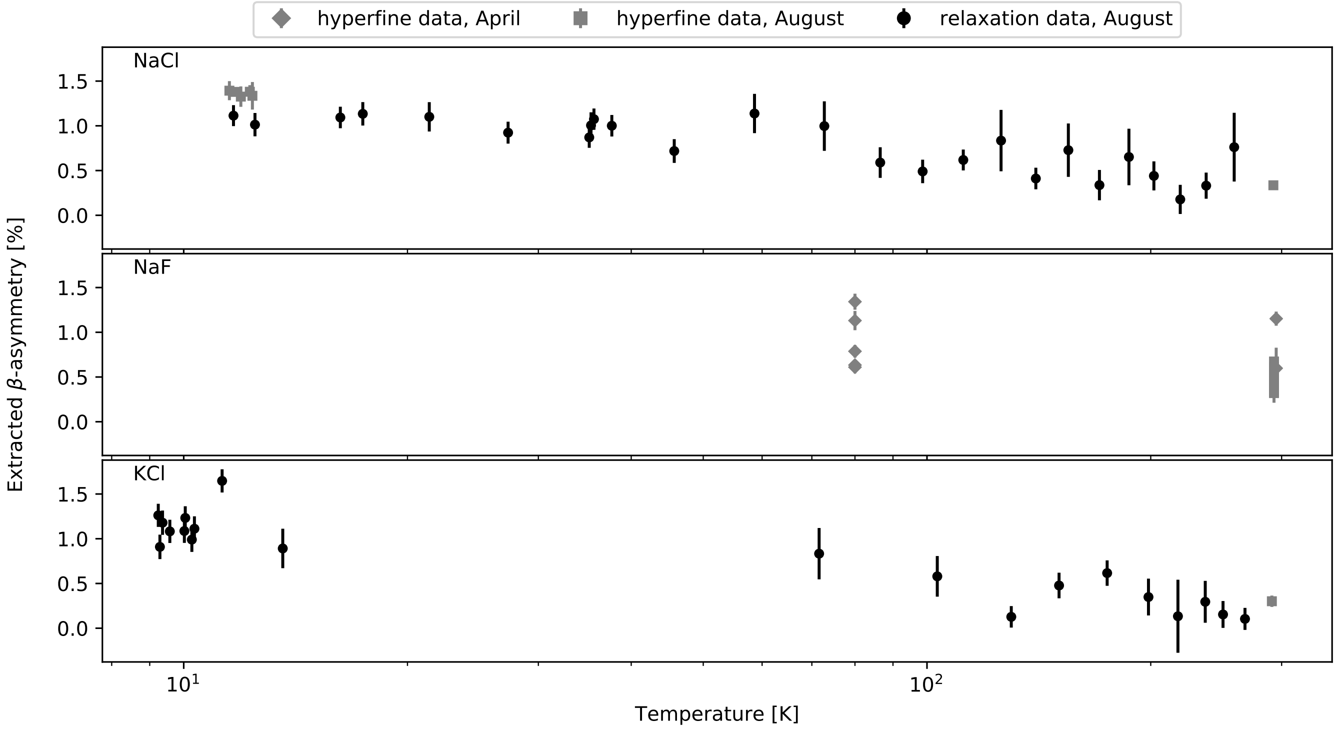


Figure 6.3: For the different crystals, the extracted asymmetry as a function of temperature is plotted. Black and gray are used to differentiate results from the hyperfine and relaxation scans, and different markers to distinguish the two beamtimes. NaCl and KCl are the most studied crystals, and also give the highest signal.

Summary

From the set of crystals, Pt and KBr can be eliminated as a good candidate hosts for the next phase of the research project. NaF, while providing a good asymmetry signal in April, did not do so in August, most likely due to degradation of the crystal surface. The two main candidates are NaCl and KCl, which both give an average of about 1.15% asymmetry.

The spin-relaxation time of these results could not be determined, but at low temperatures is greater than the half-life of ^{35}Ar.

6.2 Simulation of the possible losses of nuclear spin polarization

6.2.1 Losses due to spin rotation

When extracting the nuclear polarization from the rate equations, the approximation is made that the population of all n eigenstates stays in the nth eigenstate throughout the entire flight through the various magnetic fields. This is the definition of the adiabatic theorem [79], and using it gives the direct mapping between the set of quantum numbers when the external magnetic field is weak (F, m_F) and when the field is strong (m_I, m_J), introduced in Chapter 2. However, the adiabatic theorem relies on two assumptions [80]:

1. There are no degenerate levels in the spectrum of the Hamiltonian at each point in time.

Table 6.1: Weighted average of the observed asymmetry in hyperfine scans and relaxation curves at temperatures < 15 K and > 273 K for the different crystals. The result for NaF is deduced from the hyperfine spectra of April only.

	NaCl	KCl	KBr	NaF	Pt
Asymmetry <15 K [%]	1.21(14)	1.16(7)	0.27(05)	-	0.41(5)
Asymmetry >273 K	0.33(4)	0.30(6)	-	1.09(17)	-

2. The timescale of the changes in the Hamiltonian must obey

$$\frac{\langle\psi_m, t|\dot{H}|\psi_n, t\rangle}{E_n - E_m} \equiv \frac{1}{\tau} \ll \frac{E_m}{\hbar} \tag{6.2}$$

   where ψ_i is the ith eigenstate. This puts a restriction on the time derivative of each element on the mth row of the Hamiltonian matrix.

Previous β-asymmetry campaigns have always used a ground state $J = {}^1\!/_2$, where the levels are not degenerate except for an exactly zero magnetic field. In ^{35}Ar, this assumption is already violated due to the crossing of the levels in a weak magnetic field ($\approx 25\,\mathrm{mT}$, see the Breit-Rabi diagram in Chapter 2).

The appropriate timescale for the second assumption depends on both the speed of the particle as it traverses through the magnetic fields (since spatial variations are transformed into temporal variations), and the hyperfine interaction strength. Slow changes in the magnetic field will lower the time derivative on the left side of the equation, while a strong hyperfine interaction will increase the energy term on the right side. The speed also determines the time needed to reach the crystal location, starting from the beginning of the transitional field region. Note that the simulation is extended by 50% beyond the time needed to reach the crystal location, where the magnetic field is kept fixed to determine the static evolution. This is also the time period over which the time-averaged spin projection is calculated.

Therefore, the dynamics of the nuclear spin, i.e. the precession of it in the magnetic field, can lead to polarization losses if the spin rotation is not adiabatic. Explicit calculation of the rotation process quantifies the possible losses. The QuTiP library [81] was used for the calculation, enabling the automatic creation and calculation of quantum objects such as spin operators, Hamiltonians and kets. Thus the focus is shifted from the implementation of the equations to the formulation. The main part of the simulation code is given in Appendix A.2.

The magnetic field has been measured in three different situations: with only the large electromagnet on, with only the transitional field on, and with both the electromagnet and the transitional field coils on. The main electromagnet was set to a value of 0.35 T as measured at the center during the measurements of the magnetic field. As all experiments on ^{35}Ar have run with the magnet producing a field of 0.6 T, the main magnet field has to be multiplied with a factor of $1.71 (= 0.6/0.35)$ to get the correct magnetic field profile.

Hamiltonian

In the interaction between the two spins and the environment, three terms arise:

1. the hyperfine interaction between the two spins.
2. the interaction between the electronic spin and the external magnetic field ($\boldsymbol{B}$).
3. the interaction between the nuclear spin and the external magnetic field.

The Hamiltonian is given by the following three terms:

$$\begin{aligned}\mathcal{H} &= \mathcal{H}_{hyp} + \mathcal{H}_{B,J} + \mathcal{H}_{B,I} \qquad (6.3)\\ &= A\boldsymbol{I}\cdot\boldsymbol{J} + g_L\mu_B\boldsymbol{B}\cdot\boldsymbol{J} - g\mu_N\boldsymbol{B}\cdot\boldsymbol{I}, \qquad (6.4)\end{aligned}$$

where the electric quadrupole hyperfine interaction is neglected due to the small contribution compared to the magnetic dipole interaction. The hyperfine interaction strength is given by the magnetic hyperfine parameter A. The magnetic interaction with the electron spin is moderated by the Landé g-factor g_L and Bohr magneton μ_B. For the magnetic interaction with the nuclear spin, these factors are replaced by the nuclear g-factor g and nuclear magneton μ_N. The numerical values for the constants used in the calculations are [82]:

$$g_L = \frac{J(J+1) - S(S+1) + L(L+1)}{2J(J+1)} + \frac{2(J(J+1) + S(S+1) - L(L+1))}{2J(J+1)}, \tag{6.5}$$

$$\mu_B = 13.996\,245\,042 \times 10^9\,\mathrm{Hz\,T^{-1}}, \tag{6.6}$$

$$\mu_N = 7.622\,593\,285 \times 10^6\,\mathrm{Hz\,T^{-1}}. \tag{6.7}$$

From these values, the magnetic interaction with the nucleus is seen to be a factor 1000 weaker than the magnetic interaction with the electron. Although this term could be safely neglected, it is included in the calculations for completeness.

Spaces, states and operators

The system of interest consists of both the nuclear spin $\boldsymbol{I}$ and the electron spin $\boldsymbol{J}$. Every ket $|\psi\rangle$ describing the combined system is therefore part of the tensor product of both Hilbert spaces:

$$|\psi\rangle \in \mathbb{C}^{\dim I} \otimes \mathbb{C}^{\dim J} = \mathbb{C}^{\dim I \times \dim J}, \tag{6.8}$$

where the dimensions refer to the dimension of the spin subspaces generated by the given spins.

The spin operators themselves are trivially extended towards the combined space by taking the tensor product with the identity operator acting on the other subspace:

$$\forall i \in \{x, y, z\} : \sigma_{i,I} = \sigma_i \otimes \mathbb{1}, \sigma_{i,J} = \mathbb{1} \otimes \sigma_i, \tag{6.9}$$

$$\boldsymbol{I} = \sum_{i \in x,y,z} \sigma_{i,I}, \quad \boldsymbol{J} = \sum_{i \in x,y,z} \sigma_{i,J}, \quad \boldsymbol{I} \cdot \boldsymbol{J} = \sum_{i \in x,y,z} \sigma_{i,I} \cdot \sigma_{i,J}. \tag{6.10}$$

With the definition of these operators, the Hamiltonian of Eq. (6.4) can be formulated and used in the code.

In order to generate the initial ket, the populations of the F, m_F sub-states are extracted from the optical pumping results on the transition giving maximal polarization. These states correspond to the eigenstates of the hyperfine interaction in Eq. (6.4). Since the population in a ket $\alpha\,|\psi\rangle$ is given by $|\alpha|^2$, each eigenstate is weighed by the root of the calculated population. The sum of these weighted eigenstates is then used as the initial ket. For multi-frequency pumped ^{35}Ar, the initial ket is given by

$$|\psi_0\rangle = 0.74\,|3/2, 2\rangle - 0.23\,|3/2, -2\rangle + 0.25\,|-1/2, 1\rangle - 0.30\,|-3/2, 2\rangle\,, \tag{6.11}$$

where eigenstates with $|\alpha|^2 < 0.05$ have been omitted. Note that, in general, the ket is not a direct product of an electron and a nuclear spin ket:

$$\nexists\,|\phi\rangle \in \mathbb{C}^{\dim I}, \nexists\,|\eta\rangle \in \mathbb{C}^{\dim J} : |\psi_0\rangle = |\phi\rangle \otimes |\eta\rangle\,. \tag{6.12}$$

This shows directly that we cannot consider the nuclear and electron spin separately, but must always consider the system as a whole.

At each timestep t of the simulation, the nuclear polarization along the z axis can be determined for the ket $|\psi_t\rangle$ by calculating

$$I_z = \frac{\langle\psi_t|\sigma_{z,I}|\psi_t\rangle}{I}, \tag{6.13}$$

and similar definitions for the other axes. In contrast to the method of determining the nuclear spin-polarization given in Chapter 2, this method takes the applied magnetic field and subsequent decoupling of the spins into account. As the $|F, m_F\rangle$ eigenstates of the Hamiltonian at a magnetic field of $0\,\mathrm{T}$ are superpositions of the decoupled $|m_I, m_J\rangle$ basiskets and the eigenstates of the Hamiltonian approach the decoupled basis for large magnetic field strengths, the nuclear polarization can increase with magnetic field strength by reducing the mixing of the different $|m_I, m_J\rangle$ states.

From the rate equations, only the population in each state can be calculated and the relative phase is unknown. This phase results in a different orientation angle of the spins about the magnetic field vector. Simulations with different orientation angles are used to average out the missing phase information. In order to rotate about the magnetic field vector, the operator

$$D\left(\theta\right) = \exp\left(-\frac{i\theta \boldsymbol{n}\cdot\boldsymbol{S}}{\hbar}\right). \tag{6.14}$$

is used to rotate over an angle of θ rad about the axis defined by the unit vector $\boldsymbol{n}$. In order to rotate both spins at the same time, $\boldsymbol{I}+\boldsymbol{J}$ is substituted for $\boldsymbol{S}$. The final form of the operator is

$$D\left(\theta\right) = \exp\left(-\frac{i\theta \boldsymbol{n}\cdot\left(\boldsymbol{I}+\boldsymbol{J}\right)}{\hbar}\right), \tag{6.15}$$

where $\boldsymbol{n}$ is the unit vector along the initial magnetic field. For every angle considered in the interval $[0, 2\pi)$ rad[1], the ket is left multiplied by the matrix representation of this operator. In the simulations, 10 different equidistant angles were used for this rotation. Figure 6.4 shows the dependency of the final calculated nuclear polarization, fitted with a sine function. Aside from the expected 2π period, it is clear that the behaviour is sampled well enough to extract a reliable average. The value extracted by Eq. (6.13) at each point in time is averaged over all 10 spin trajectories.

[1] Although the kets are spinors and thus would require a rotation of 4π rad to restore the original wavefunction, a rotation of 2π rad gives the same wavefunction up to a minus sign in the components. As our interest lies in the expectation value and not an off-diagonal element, this minus sign does not contribute and we only require a rotation of 2π rad.

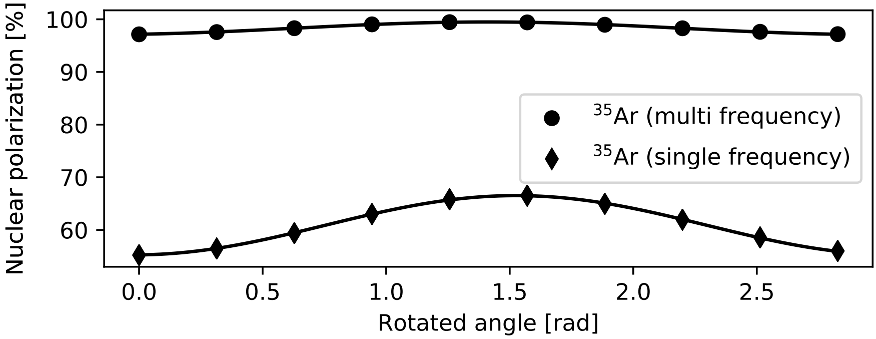


Figure 6.4: The achieved nuclear polarization using different starting orientations about the magnetic axis. The frequency shows the expected 2π rad rotation symmetry. The population in a multi-frequency pumped ensemble shows little impact, due to the high degree of polarization

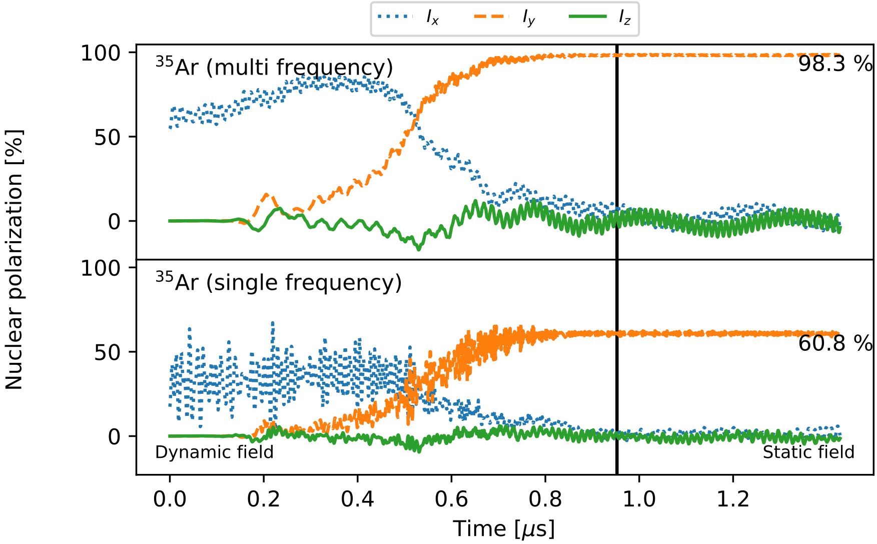


Figure 6.5: The trajectory of ^{35}Ar in spin space, averaged over the 10 different starting angles, results in an eventual spin-projection on the y-axis of approximately 100% for the multi-frequency pumped starting population (top), and 61% for the single-frequency pumped ensemble (bottom).

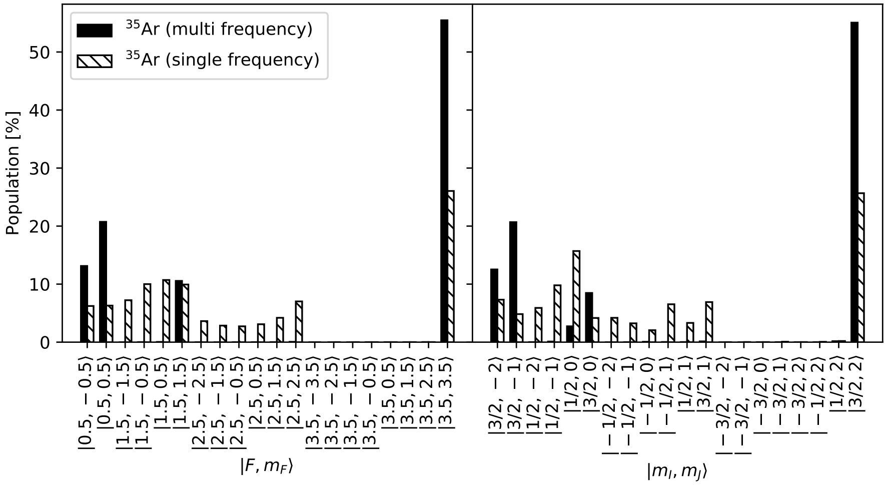


Figure 6.6: Population for the initial ket in the $|F, m_F\rangle$ basis (left), and for the final ket in the $|m_I, m_J\rangle$ basis (right) for the multi- and single-frequency pumped Ar. An adiabatic transfer would translate the population directly from the kets on the left plot to the corresponding ket on the right.

Results

The simulations for polarized ^{35}Ar, performed using the experimental conditions (50 keV beam energy, 0.6 T field strength) and using the results of the rate equations for both multi- and single-frequency pumping as initial kets, result in the spin-trajectory shown in Figure 6.5. The nuclear spin polarization is shown as a function of time for the x, y and z axis. The components have been labeled such that the positive x direction is along the beam path, the positive y axis in the direction of the main magnetic field, and the z axis points upward. This is the same axis system as used in the article in Chapter 4. We are therefore interested in converting a polarization along the x axis into a polarization along the y axis.

The calculations show a very effective rotation: applying Eq. (2.35) to the results of the rate equations, the final polarizations are estimated to be 99.9% for the multi-frequency pumped case and 68.8% for the single-frequency pumped case. With the spin-rotation calculations resulting in 98.3% and 60.8% respectively, there is practically no difference between the two calculations for the multi-frequency pumped situation. The single-frequency pumped case does deviate,

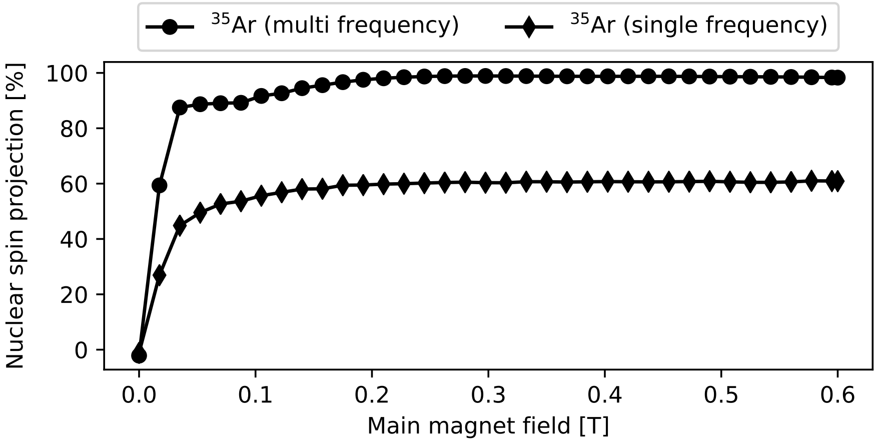


Figure 6.7: Nuclear spin projection as a function of main magnetic field strength. In both cases, the achieved spin polarization starts to saturate after the second data point, corresponding to a magnetic field of 35 mT. Full spin-rotation is achieved from about 0.2 T onward.

also showing a much more erratic trajectory in spin space. Comparing the populations in the starting kets (left panel of Figure 6.6), multi-frequency pumping results in a much more pure eigenket than the single-frequency pumping situation, which is most likely the cause of the wider distribution found in the final ket of the simulation (right panel of Figure 6.6) for the single-frequency pumping.

Variation of the strength of the main magnetic field (Figure 6.7) shows that the adiabatic rotation in the region probed in the decoupling curves (above 0.2 T) does not change in efficiency. No additional corrections are therefore needed, and the observed results depend purely on the crystal properties.

From these calculations, it is observed that the adiabatic approximation is fully satisfied for the case of multi-frequency pumped and there is no additional loss of polarization from this step. Single-frequency pumping prepares the ensemble in a more mixed initial ket, leading to less nuclear polarization along the y-axis than predicted by the adiabatic approximation.

One of the planned projects at the beamline is the installation of a fluorescence polarization-checker [83], where a second laser probes the population of the different substates directly. This additional measurement type can provide experimental verification of the quantum-mechanical spin-rotation calculations, and will also allow to perform off-line developments to identify suitable laser schemes for spin-polarizing other elements.

6.2.2 Losses due to beta-scattering

In order to determine any systematic effects due to the scattering of the β-particles in the setup, COMSOL simulations were carried out. The full setup including the copper heat shielding was imported from the Autodesk Inventor drawings.

In the simulation, the electronic stopping power of copper taken from Ref. [84] was applied to the *Ionization loss* simulation node to model the slowdown of the positrons in the copper shielding. The copper also interacted with the β-particles through a *Nuclear stopping* node that implements Coulomb scattering, which will dominate for low-energy positrons. A constant magnetic field of 0.6 T (direction given in Figure 6.8) was set, resulting in a Lorentz force on the positrons.

Annihilation of the positrons and the resulting gamma rays was not simulated, due to the much smaller cross section compared to the ionization loss cross section.

4×10^4 positrons were released from the surface of a 1×1 cm crystal surface with a uniform spread. The polar angle of the initial momentum was drawn from the angular distribution expected for a (25.18% of the population after charge exchange being 98% polarized=) 24.7% polarized ensemble with an asymmetry parameter of 0.43:

$$W\left(\theta\right) = \frac{1 + 0.106\cos\left(\theta\right)}{2\pi}, \tag{6.16}$$

where the factor of v/c was neglected. Compensating for the differential element of the surface of a sphere, the azimuth angle is given by

$$\phi = \arccos\left(u\right), \tag{6.17}$$

where u is a uniformly chosen random number in the interval $[0,1)$. The initial energy of the positrons is drawn from the β-spectrum shape for ^{35}Ar. For the shape of this spectrum, calculations were performed by L. Hayen [85]. The calculations were repeated for an ensemble with 0% asymmetry to establish the baseline.

Although a thin aluminium window (approximately 100 µm thick) is also present between the β-detector and the vacuum chamber, this was not simulated due to the proximity to the detectors. While this window would influence the measured β-spectrum, it would not change the asymmetry. As no spectrum is recorded in the experiment, only raw counts, this approximation saves calculation time by eliminating the need to propagate the particles through a thin layer.

The β-particles that arrive at the β-detectors were frozen in place and their track recorded. The top plot of Figure 6.8 shows a 3D representation of simulation in COMSOL, with the tracks colored according to the kinetic energy.

Unfortunately, an issue with generating random numbers in COMSOL appeared during the simulations. In order to generate random numbers from a user-defined distribution, a table of values is given and interpolated. This method has been used to generate the 10.6% and 0% asymmetry simulations. An asymmetry of respectively 4.37(93)% and −2.61(93)% is observed, where the uncertainty is an approximate statistical uncertainty by using $\sigma = \sqrt{N}$. As the setup is symmetric (mirrored over the yz-plane), the simulation with 0% asymmetry should result in 0% detected asymmetry. Repeating the simulation with 0% asymmetry using the built-in uniform distribution generator gives an asymmetry of 0.3(8)%, where the expected 0% is well within the statistical uncertainty. This points towards a problem in the generation of random numbers from user-defined functions.

By using the 0% simulations as a baseline, the effect of the faulty random numbers should be minimized. Subtracting both simulations from each other results in a signal height of 7.0(13), which is 4.4σ away from the 1.21(14)% average asymmetry observed in NaCl. Introducing a systematic uncertainty of 2%, which is not unreasonable given the unknown fault in the random number generator, the simulation result is evaluated as 7.0(33)% by linearly adding the systematic uncertainty to the statistical one. This reduces the deviation already to 1.75σ, but also illustrates the need for high precision simulations of the setup in order to get scattering and solid angle corrections to sub percent precision.

Although precise results cannot be extracted from the COMSOL simulations, several interesting features can still be identified and taken into account for future phases in the project. The middle panel of Figure 6.8 shows the location distributions for the detected β^+-particles. As can be seen, the crystal is offset by 5.5 mm in the z direction to capture less of the positrons scattered on the copper crystal mount. The location distribution clearly shows that the

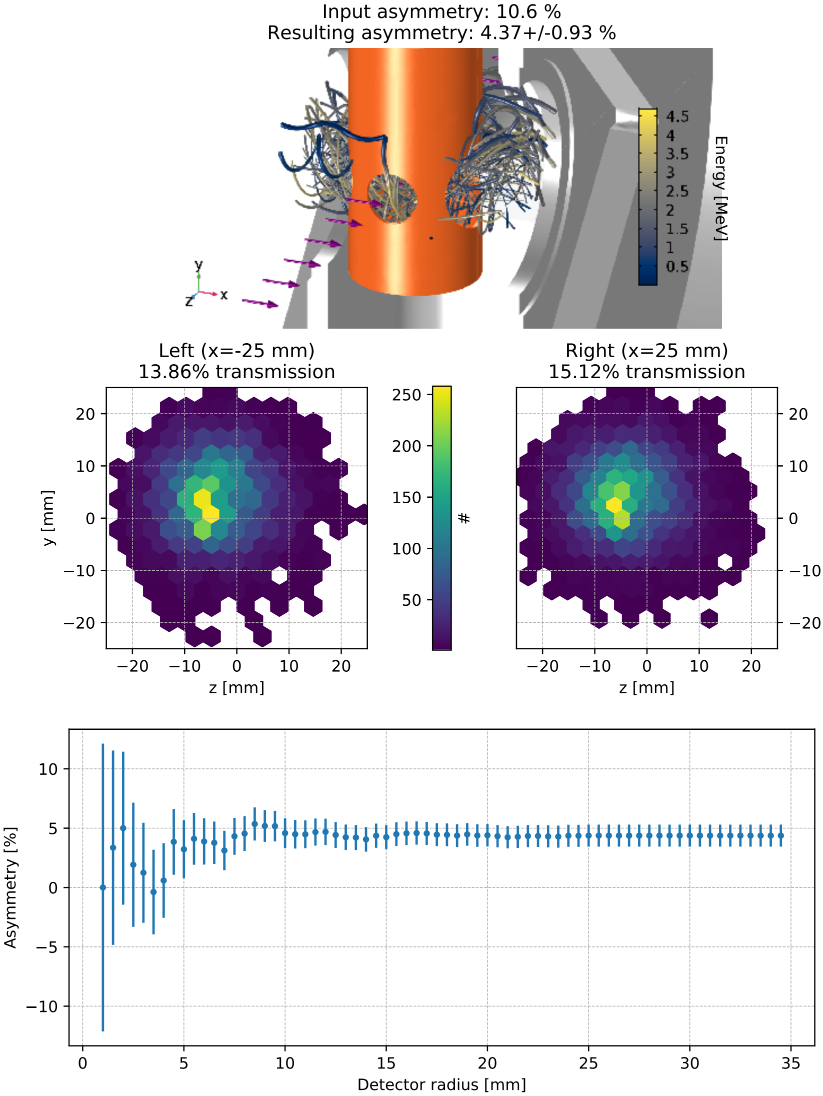


Figure 6.8: Top plot: setup as implemented in COMSOL (orange = copper). The magnetic field is oriented along the x-axis. Middle plots: location distribution of the detected particles in both detectors, along with the total transmission to each detector. Bottom plot: measured asymmetry as a function of detector size. The detectors used in the setup covered the entire hole (25 mm radius).

Table 6.2: The first section details experimental asymmetry modifiers and their effect. The optical pumping is assumed to be on the transition frequency giving maximal polarization. The second section details the input asymmetry for the β-scattering simulations and the result thereof.

Mechanism	Asymmetry impact
Optical pumping with σ^+	100%
β-asymmetry parameter	43%
CEC population distribution	25%
Adiabatic rotation	98%
Asymmetry for simulations	10.6%
Solid angle and scattering simulation	7.0(13)%

distribution peaks sharply in the middle of the opening and quickly drops off. Any deviation in detector location will therefore have a large impact on the measured asymmetry. The bottom plot of Figure 6.8, where the asymmetry of particles within a certain radius around the origin is calculated, shows the large impact that the detector size has on the measured asymmetry. The statistics improve with detector size, as more β-particles are detected and the statistical uncertainty is reduced. However, by integrating over a larger solid angle, the measured asymmetry is reduced as compared to the optimal detector size of 10 mm radius.

6.3 Possible improvements

Table 6.2 lists the different factors that influence the observed β-asymmetry, where the main loss channel is clearly the population distribution across the different fine structure levels after charge exchange. For future phases of the project, methods that would select only the metastable population of the beam would significantly improve the measurements. One of the possibilities for such a method is extending the beamline to make a 90° turn[2]. This would require selective re-ionization of the polarized ensemble, leaving the unpolarized population in its atomic configuration. This could be carried out in two different ways:

[2] Although it does not need to be 90°, this would be required to fit the beamline in the ISOLDE hall.

1. State-selective collisional re-ionization with Cl gas.
2. Selective re-ionization from the polarized metastable level using pulsed lasers.

The collisional re-ionization has been measured to have an approximately 3 times higher cross section with the metastable state than with the ground state [52]. This would give a population of the metastable state given by

$$\frac{w\,(\text{meta}) \cdot \sigma_{rel}\,(\text{meta})}{\sum_{\text{level}} w\,(\text{level}) \cdot \sigma_{rel}\,(\text{level})}, \tag{6.18}$$

where w is the population in the indicated level. Assuming a two-level system (ground and metastable states), a population of the metastable state of 25% would result in a re-ionized beam consisting of 50% Ar in the metastable state and 50% in the ground state, an enhancement factor of 2.

By using the calculated charge exchange populations (performed by A. Vernon, see Chapter 3), the relative ionization cross section for the other metastable levels can be estimated. The population of the metastable levels after a flight of 1.6 m can then be used to estimate the impact of including these additional levels in the calculation. This procedure estimates a re-ionized beam consisting of 55% Ar in the metastable level we are interested in, 23% in the ground state and the rest distributed over the other levels. Note that these estimates use the calculations where an Ar beam hits a gas target of K, not Cl. Nevertheless, these additional estimates show that collisional re-ionization should give a beam which is consisting of 50% Ar in the correct metastable state.

The re-ionization efficiency has been reported to be 9% [52], and collisional re-ionization of polarized Li on a He gas target has been shown to be possible [86]. However, the re-ionization efficiency of a Li beam on a He gas target has been shown to depend on the density of the gas target [87], where the ionization cross section for the metastable state is higher at a smaller density of gas and the total cross section grows with gas density. A careful tuning of the re-ionization cell gas density would therefore be required.

Using pulsed lasers to selectively re-ionize the meta-stable polarized ensemble should result in an almost pure beam of ^{35}Ar in the meta-stable state. In this case, the signal would increase by a factor 4. The laser ionization efficiency itself is typically on the order of 1-10%, comparable to collisional re-ionization. An appropriate laser scheme should maintain the polarization of the ensemble.

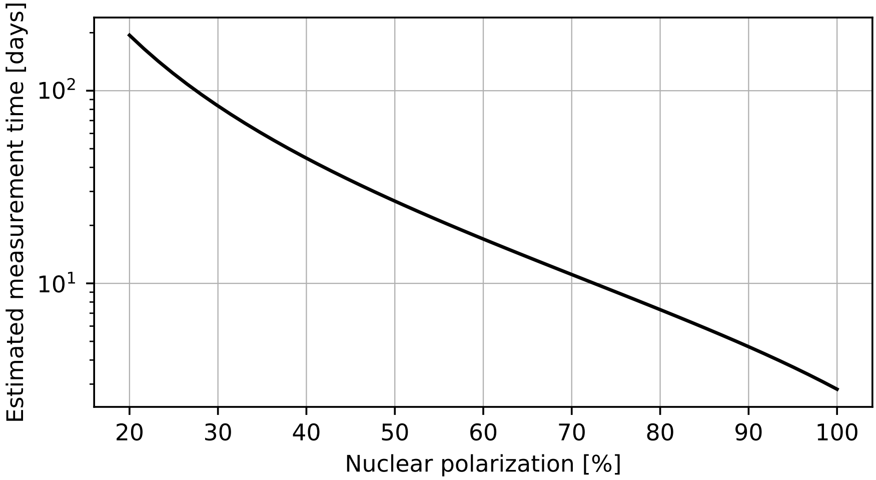


Figure 6.9: Estimated measurement time with updated yield information as a function of nuclear polarization.

For both the collisional and laser re-ionization approach, an efficiency of the order of 10% can be assumed. Assuming a charge exchange efficiency of the same order of 10%, approximately 1% of the measured yield could be delivered to the setup after state-selective re-ionization. Using Eq. (1.10) with 1% of the yield measured in the April campaign, the estimated measurement time as a function of achieved nuclear polarization can be calculated (Figure. 6.9). As the estimated time is approximately 30 days for a polarization of 40%, collisional re-ionization does not offer enough of an increase in signal to achieve a reasonable measurement time. Laser re-ionization should give a nuclear polarization high enough for a beamtime of less than 10 days, which is a more reasonable timeframe. Note that this estimation does not take into account any duty cycle losses due to the reheating and cooling down of the crystal when the asymmetry signal decreases.

Another factor to be considered is the covered solid angle and scattering in the setup. Based on the results of the COMSOL simulations, the placement of the detectors and their size will need to be simulated quite accurately in order to avoid large systematic uncertainties. As a cold shield will be needed to reach the low temperatures, the scattering on the shield itself will also need to be calculated. Geant4 is more suited for this exact task than COMSOL, although the definition of the geometry in the Geant4 framework will be more difficult.

As a decrease in signal has been observed when the setup is kept at the coldest temperature, a better vacuum is required to extend the measurement time. The current vacuum reached a pressure of 3×10^{-6} mbar, which can be improved by using CF connections with copper gaskets instead of an aluminium chamber and Viton O-rings. However, care must be taken that the vacuum chamber is made out of a magnetic field compatible material. Note furthermore that laser ionization is more easily compatible with high vacuum than a re-ionization gas cell.

6.4 Summary

The relaxation times and observed β-asymmetries in the different crystals show that either KCl or NaCl at 10 K maximize the β-asymmetry signal. The degradation of the crystal surface quality is most likely the cause of the drop in asymmetry observed between April and August for the NaF host, as the crystal was stored in air between the beamtimes. If the surface quality is better maintained, the best crystal host might still be NaF at low temperatures. Such measurements have unfortunately not been performed in the available beamtimes. While no numerical value can be given for the spin-relaxation time in the different crystals, it is long enough at low temperatures to generate a near-constant signal over a 4.2 s time period.

The spin-rotation simulations in the magnetic fields have been carried out in their entirety to probe the validity of the decoupling and rotation assumed in the rate equations. For the specific case of ^{35}Ar, the multi-frequency pumped ensemble does indeed rotate adiabatically. Taking this adiabatic rotation into account, simulations have been performed to estimate the asymmetry that can be expected. Simulations predict an observed asymmetry of 7.0(13)%. As the highest average asymmetry has been observed in NaCl and was 1.21(14)%, the deviation from the simulations is 4.4σ. While this is a large deviation from the expected value, the β-scattering simulations performed in COMSOL need to be reevaluated in order to asses the accuracy, possibly in Geant4. An additional uncertainty of 2% on the simulations would seem reasonable, and would already reduce the statistical deviation to 1.75σ when added linearly to the statistical uncertainty. This shows the need for accurate simulation work as well as accurate building of the setup, which must be taken into account in the design phase.

Among the possible improvements for a future setup, laser re-ionization offers the largest increase in polarization of the ensemble and decrease in final measurement time. The need for a better vacuum to increase the time available for measurements will also be crucial.

Chapter 7

Conclusions and outlook

It's not over.
—Blind Guardian, *Sacred Worlds*

In the course of this thesis, a new laser-polarization beamline has been designed and constructed at ISOLDE, CERN. Technical developments focused on the design of the magnetic fields and simulations of the expected transport efficiency [56].

As part of the ongoing research project on spin-polarized ^{35}Ar for the purpose of weak interaction studies [21], the first phase of this research project was carried out at the new beamline. In this phase, the feasibility of producing a spin-polarized beam of ^{35}Ar has been studied, as well as the crystal host that maintains the polarization the longest. In preparation, the rate equation formalism already in use for laser-polarization setups [43] was extended to incorporate interaction with multiple lasers and schemes with electron spins larger than $^1/_2$ in the lower state. Based on the results of these rate equation calculations, a laser scheme using 2 AOM-generated sidefrequencies was suggested in order to maximize the polarization of the ensemble.

The results of two beamtimes, carried out in April and August 2017, show that the extension of the rate equations reproduces the spectrum measured online. The measured signal gain (1.85) was higher than expected from the simulations (1.33), most likely due to not saturating the transition with a single laser beam. The relaxation time measurements, carried out at different temperatures, show that both NaCl and KCl at low temperatures give the highest asymmetry signal, of approximately 1.2%. The different signals observed in NaF, combined with the estimated implantation depth, indicate that the surface quality should be carefully maintained. In addition, NaF with a good surface quality might be better than either NaCl or KCl. Simulations of the charge exchange process, optical pumping, adiabatic rotation and scattering simulations result in an expected 7.0(33)% asymmetry signal that could be observed in the most ideal conditions. Systematic uncertainties in the solid angle and scattering simulations contribute to about half of the uncertainty.

In the second phase of the project, the signal would need to be improved significantly [30] in order to approach the requirements to measure the β-asymmetry parameter of ^{35}Ar with a relative precision of 0.5%. As the current major loss of signal is due to population distribution across multiple levels following charge exchange, state selective re-ionization offers a large possible increase in signal. As collisional re-ionization only offers a possible enhancement factor of 2.3, laser re-ionization with a gain factor of 6.25 is more favorable as the estimated final measuring time goes from approximately 80 days to only a few days. As the crystal quickly develops condensation at low temperatures, improvements to the vacuum are needed to extend the useful measuring time and reduce duty-cycle losses. Of critical importance are high-precision simulations of the setup to determine the required solid angle and the effect of scattering. These results will determine the location and size of the detectors, and their allowed deviation thereof.

Appendix A

Included source code

In data we trust; Let us focus on the biggest sums of life; It's now a matter of life and math
—Epica, *The Cosmic Algorithm*

A.1 Rate equation implementation

The source code presented in this section is the extension of the BaseModel class of the SATLAS package used in the optical pumping simulations and fits. By extending the BaseModel class and providing a link between a dictionary of parameters and the numerical values used in the optical pumping matrices, the full suite of analysis options available in SATLAS can be used with this set of differential equations.

```
import numpy as np
from satlas.profiles import Voigt, Lorentzian
from satlas.models.basemodel import BaseModel, SATLASParameters
from scipy import integrate
from sympy.physics.wigner import wigner_6j, wigner_3j
import scipy.constants as csts
import lmfit
import itertools

W6J = wigner_6j
W3J = wigner_3j

__all__ = ['RateModelDecay', 'RateModelPolar']
```

```

# Define constants
C = csts.physical_constants['speed of light in vacuum'][0]  # Speed of light, m/s
H = csts.physical_constants['Planck constant'][0]  # Planck's constant, Js
PI = np.pi  # pi...
GL = 1.0  # Orbital g-factor
GS = 2.00232  # Spin g-factor
MUB = csts.physical_constants['Bohr magneton'][0]  # Bohr magneton
EV_TO_MHZ = csts.physical_constants['electron volt-hertz relationship'][0] * 1e-6  # eV to MHz conversion factor

#######################
# CALCULATION OBJECTS #
#######################

class BxRho_Voigt(Voigt):
    def __init__(self, A=None, fwhmG=None, mu=None, laser=None, fwhmL=None):
        self._fwhmG = fwhmG
        self._fwhmL = fwhmL
        self._laser = laser
        self._A = A
        self._lorentzian = 0
        super(BxRho_Voigt, self).__init__(mu=mu, fwhm=[fwhmG, fwhmL], ampIsArea=True, amp=1.0)

    @property
    def A(self):
        return self._A

    @A.setter
    def A(self, value):
        self._A = value
        self.set_factor()

    @property
    def gaussian(self):
        return self._gaussian

    @gaussian.setter
    def gaussian(self, value):
        self._gaussian = value
        self.fwhm = [value, self.fwhmL]

    @property
    def lorentzian(self):
        return self._lorentzian

    @lorentzian.setter
    def lorentzian(self, value):
        self._lorentzian = value
        self.fwhm = [self.fwhmG, value]

    @property
    def mu(self):
        return self._mu

    @mu.setter
    def mu(self, value):
        self._mu = value * 1e6
        self.set_factor()

    @property
    def laser(self):
        return self._laser

    @laser.setter
    def laser(self, value):
        self._laser = value
        self.set_factor()

    def set_factor(self):
        self._factor = self.A * self.laser * C * C / (8 * PI * H * self.mu * self.mu)

    def __call__(self, x):
        return super(BxRho_Voigt, self).__call__(x*1e6) * self._factor / (x*1e6)

class BxRho_Lorentzian(Lorentzian):
    def __init__(self, A=None, mu=None, laser=None, fwhm=None):
        self._laser = laser
        self._A = A
        super(BxRho_Lorentzian, self).__init__(mu=mu, fwhm=fwhm, ampIsArea=True, amp=1.0)

    @property
    def A(self):
        return self._A

    @A.setter
    def A(self, value):
        self._A = value
        self.set_factor()

    @property
    def mu(self):
        return self._mu

    @mu.setter
    def mu(self, value):
        self._mu = value * 1e6
        self.set_factor()

    @property
    def laser(self):
        return self._laser

    @laser.setter
    def laser(self, value):
        self._laser = value
        self.set_factor()

    def set_factor(self):
        self._factor = self.A * self.laser * C * C / (8 * PI * H * self.mu * self.mu)

    def __call__(self, x):
        return super(BxRho_Lorentzian, self).__call__(x * 1e6) * self.
    _factor / (x * 1e6)

##############
# MAIN CLASS #
##############
class RateModel(BaseModel):
    def __init__(self, I, J, L, ABC, centroids, energies, A_array, scale
    =1.0, shape='Voigt', laser_intensity=80, laser_mode=None,
    interaction_time=1e-6, fwhmG=0.1, fwhmL=None, backgroundparams=[0],
    field=0, fixed_frequencies=None, frequency_mode='fixed', purity=1.0,
    pi_detect=1.0, sigma_detect=1.0):
        super(RateModel, self).__init__()
        self.I = I
        self.J = J
        self.L = L
        self.A_array = A_array
        self.shape = shape

        try:
            lasers = len(laser_intensity)
        except:
            laser_intensity = [laser_intensity]
            laser_mode = [laser_mode]
        if fixed_frequencies is not None:
            self.fixed_frequencies = fixed_frequencies
        else:
            self.fixed_frequencies = []
        self.vary_freqs = len(laser_intensity) - len(self.fixed_frequencies)
        self.frequency_mode = frequency_mode

        self.laser_intensity = laser_intensity
        self.mode = laser_mode

        self._calculate_F_levels()
        self._set_energies(energies)
        self._calculate_energy_coefficients()

        self._set_population()
        self._populateparams(laser_intensity, ABC, centroids, shape, scale,
    fwhmG, fwhmL, interaction_time, backgroundparams, field, purity,
    pi_detect, sigma_detect)
        self._calculate_A_partial()
        self._calculate_energy_changes()
        self._create_D_matrix()

        self.params = self.params

    @property
    def params(self):
        return self._parameters

    @params.setter
    def params(self, params):
        p = params.copy()
        p._prefix = self._prefix
        self._parameters = self._check_variation(p)
        self._calculate_energy_changes()
        A = np.zeros((self.level_counts_cs[-1], self.level_counts_cs[-1]))
        for key in self.transition_indices:
            for x, y in self.transition_indices[key]:
                A[x, y] = params[key].value
        A = A * self.partial_A
        A = np.transpose(A) - np.eye(A.shape[0]) * A.sum(axis=1)
        self.A_array_used = A
        self._edit_D_matrix()

    def _set_energies(self, energies):
        N = self.level_counts.sum()
        # Pre-allocate the energy and population vectors.
        E = np.zeros(N)
        Nlevcs = self.level_counts.cumsum()
        for i, (n, ncs) in enumerate(zip(self.level_counts, Nlevcs)):
            E[ncs - n:ncs] = energies[i]
        self.energies = E * EV_TO_MHZ

    def _populateparams(self, laser_intensity, ABC, centroids, shape, scale, fwhmG, FWHML, interaction_time, backgroundparams, field, purity, pi_detect, sigma_detect):
        p = SATLASParameters()
        for i, val in enumerate(laser_intensity):
            p.add('Laser_intensity_' + str(i), value=val, min=0, max=None)
        for i, j in enumerate(self.Jlist):
            p.add('A_level_' + str(i), value=ABC[i][0])
            p.add('B_level_' + str(i), value=ABC[i][1])
            p.add('C_level_' + str(i), value=ABC[i][2])
            if not i == len(self.Jlist)-1:
                p.add('Centroid_level_' + str(i), value=centroids[i])
            else:
                p.add('Centroid_level_' + str(i), value=0, vary=False)
        for i, _ in enumerate(self.level_counts):
            for j, _ in enumerate(self.level_counts):
                if i < j and np.isfinite(self.A_array[i, j]):
                    p.add('Transition_strength_' + str(i) + '_to_' + str(j), value=self.A_array[i, j], min=0, vary=False)
                    fwhmL = self.A_array[i, j]/(2*PI)*1e-6 if FWHML is None else FWHML
                    p.add('FWHML_' + str(i) + '_to_' + str(j), value=fwhmL, min=0)
                    if shape.lower() == 'voigt':
                        par_lor_name = 'FWHML_' + str(i) + '_to_' + str(j)
                        par_gauss_name = 'FWHMG_' + str(i) + '_to_' + str(j)
                        expr = '0.5346*{0}+(0.2166*{0}**2+{1}**2)**0.5'
                        p.add('FWHMG_' + str(i) + '_to_' + str(j), value=fwhmG, min=0.0001)
                        p.add('TotalFWHM_' + str(i) + '_to_' + str(j), value=0, vary=False, expr=expr.format(par_lor_name, par_gauss_name))
                    else:
                        p.add('FWHMG', value=fwhmG, vary=fwhmG > 0, min=0)
        p.add('Scale', value=scale)
        p.add('Interaction_time', value=interaction_time, min=0)
        for i, val in enumerate(reversed(backgroundparams)):
            p.add('Background' + str(i), value=backgroundparams[i])
        self.background_degree = i
        p.add('Field', value=field)
        p.add('Purity', value=purity, min=0, max=1)
        p.add('pi_detect', value=pi_detect, min=0, max=1, vary=False)
        p.add('sigma_detect', value=sigma_detect, min=0, max=1, vary=False)
        self._parameters = self._check_variation(p)

    def _check_variation(self, p):
        for key in self._vary:
            if key in p:
                p[key].vary = self._vary[key]

        for i, j in enumerate(self.Jlist):
            if j[0] < 1.5 or self.I < 1.5:
                p['C_level_' + str(i)].value = 0
                p['C_level_' + str(i)].vary = False
                if j[0] < 1.0 or self.I < 1.0:
                    p['B_level_' + str(i)].value = 0
                    p['B_level_' + str(i)].vary = False
                    if j[0] < 0.5 or self.I < 0.5:
                        p['A_level_' + str(i)].value = 0
                        p['A_level_' + str(i)].vary = False
        return p

    def _calculate_F_levels(self):
        I = self.I
        J = self.J
        L = self.L
        self.Flist = []
        self.MFlist = []
        self.Jlist = []
        self.Llist = []
        dummyJ = np.array([])
        dummyF = np.array([])
        dummyFz = np.array([])
        dummy = np.array([])
        dummyL = np.array([])
        for i, (j, l) in enumerate(zip(J, L)):
            F = np.arange(np.abs(j - I), j + I + 1)  # Values of F

            Flen = (2 * F + 1).astype('int')  # Lengths of F_z
            starts = np.cumsum(np.append([0], Flen[:-1]))  # Index for different F states

            # Pre-allocate
            f = np.zeros(int((2 * F + 1).sum()))  # F-states
            mz = np.zeros(int((2 * F + 1).sum()))  # F_z-states

            # Fill the pre-allocated arrays
            for i, (entry, start) in enumerate(zip(Flen, starts)):
                mz[start:start + entry] = np.arange(-F[i], F[i] + 1)
                f[start:start + entry] = F[i]
            self.Flist.append(f)
            self.MFlist.append(mz)
            self.Jlist.append([j]*len(f))
            self.Llist.append([l]*len(f))
            dummyF = np.append(dummyF, f)
            dummyFz = np.append(dummyFz, mz)
            dummyJ = np.append(dummyJ, np.ones(len(f))*j)
            dummyL = np.append(dummyL, np.ones(len(f))*l)
            dummy = np.append(dummy, np.array([len(f)]))
        self.F = dummyF
        self.Mf = dummyFz
        self.J = dummyJ
        self.L = dummyL
        self.level_counts = dummy.astype('int')
        self.level_counts_cs = self.level_counts.cumsum()

    def _calculate_energy_coefficients(self):
        S = 0.5
        L = self.L
        # Since I, J and F do not change, these factors can be calculated once
        # and then stored.
        I, J, F = self.I, self.J, self.F
        C = (F*(F+1) - I*(I+1) - J*(J + 1)) * (J/J) if I > 0 else 0 * J  #*(J/J) is a dirty trick to avoid checking for J=0
        D = (3*C*(C+1) - 4*I*(I+1)*J*(J+1)) / (2*I*(2*I-1)*J*(2*J-1))
        E = (10*(0.5*C)**3 + 20*(0.5*C)**2 + C*(-3*I*(I+1)*J*(J+1) + I*(I+1) + J*(J+1) + 3) - 5*I*(I+1)*J*(J+1)) / (I*(I-1)*(2*I-1)*J*(J-1)*(2*J-1))
        C = np.where(np.isfinite(C), 0.5 * C, 0)
        D = np.where(np.isfinite(D), 0.25 * D, 0)
        E = np.where(np.isfinite(E), E, 0)

        gJ = GL * (J * (J + 1) + L * (L + 1) - S * (S + 1)) / \
                (2 * J * (J + 1)) + GS * (J * (J + 1) - L * (L + 1) + S *
                                          (S + 1)) / (2 * J * (J + 1))
        gJ = np.where(np.isfinite(gJ), gJ, 0)
        gF = gJ * (F * (F + 1) + J * (J + 1) - I * (I + 1)) / \
                    (2 * F * (F + 1))
        gF = np.where(np.isfinite(gF), -gF, 0)
        self.A_coeff, self.B_coeff, self.C_coeff, self.field_coeff = C, D, E, gF * MUB * self.Mf * ((10 ** (-6)) / H)

    def _calculate_energy_changes(self):
        field = self.params['Field'].value
        A = np.zeros(self.level_counts_cs[-1])
        B = np.zeros(self.level_counts_cs[-1])
        C = np.zeros(self.level_counts_cs[-1])
        centr = np.zeros(self.level_counts_cs[-1])
        for i, (ncs, n) in enumerate(zip(self.level_counts_cs, self.level_counts)):
            A[ncs-n:ncs] = self.params['A_level_' + str(i)].value
            B[ncs-n:ncs] = self.params['B_level_' + str(i)].value
            C[ncs-n:ncs] = self.params['C_level_' + str(i)].value
            centr[ncs-n:ncs] = self.params['Centroid_level_' + str(i)].value
        self.energy_change = centr + self.A_coeff * A + self.B_coeff * B + self.C_coeff * C + self.field_coeff * field

    def _set_population(self, level=-1):
        try:
            levels = len(level)
        except:
            levels = 1
            level = [level]
        total_number = sum(self.level_counts[level])
        P = np.zeros(self.level_counts_cs[-1])
        for lev in level:
            N = self.level_counts_cs[lev]
            P[N - self.level_counts[lev]:N] = 1.0 / total_number
        self.P = P

    def _calculate_A_partial(self):
        I = self.I
        J = self.Jlist
        F = self.Flist
        Mf = self.MFlist
        N = self.level_counts_cs[-1]
        self.partial_A = np.zeros((N, N))
        self.transition_indices = {}
        self.characters = {0: {'x': [], 'y': []}, 1: {'x': [], 'y': []}}
        for i, _ in enumerate(self.level_counts):
            for j, _ in enumerate(self.level_counts):
                if i < j and not np.isclose(self.A_array[i, j], 0):
                    indices_ex = []
                    indices_gr = []
                    for k, (Jex, Fe, Mze) in enumerate(zip(J[i], F[i], Mf[i
    ])):
                        for l, (Jgr, Fg, Mzg) in enumerate(zip(J[j], F[j],
    Mf[j])):
                            A = float((2 * Jex + 1) * (2 * Fe + 1) * (2 * Fg
     + 1))
                            W3 = W3J(Fg, 1.0, Fe, -Mzg, Mzg - Mze, Mze)
                            W6 = W6J(Jgr, Fg, I, Fe, Jex, 1.0)
                            A = A * (W3 ** 2)
                            A = A * (W6 ** 2)
                            x = self.level_counts_cs[i] - self.level_counts[
    i] + k
                            y = self.level_counts_cs[j] - self.level_counts[
    j] + l
                            self.partial_A[x, y] = A
                            indices_ex.append(x)
                            indices_gr.append(y)
                            if A > 0:
                                self.characters[int(np.abs(Mzg-Mze))]['x'].
    append(x)
                                self.characters[int(np.abs(Mzg-Mze))]['y'].
    append(y)
                    self.transition_indices['Transition_strength_' + str(i)
    + '_to_' + str(j)] = list(zip(indices_ex, indices_gr))

    def _create_D_matrix(self):
        N = self.level_counts_cs[-1]
        D = np.zeros((N, N, len(self.laser_intensity)), dtype='object')
        bxrho = BxRho_Voigt if self.shape.lower() == 'voigt' else
    BxRho_Lorentzian

        self.indices = []
        for laser_index, laser in enumerate(self.laser_intensity):
            for i, j in itertools.combinations(range(len(self.level_counts))
    , 2):
                for k, (fe, mze) in enumerate(zip(self.Flist[i], self.MFlist
    [i])):
                    for l, (fg, mzg) in enumerate(zip(self.Flist[j], self.
    MFlist[j])):
```

```
                        x = self.level_counts_cs[i] - self.level_counts[i] + k
                        y = self.level_counts_cs[j] - self.level_counts[j] + l
                        if np.isclose(self.A_array[i, j], 0) or np.isclose(self.partial_A[x, y], 0):
                            continue
                        purity = self.params['Purity'].value
                        frac = purity if self.mode[laser_index] == (mze - mzg) else (1.0 - purity) if 0 == -(mze - mzg) else 0
                        if frac == 0:
                            pass
                        else:
                            intensity = frac * self.params['Laser_intensity_' + str(laser_index)].value
                            A = self.params['Transition_strength_' + str(i) + '_to_' + str(j)].value
                            mu = (self.energies[k] + self.energy_change[k]) - (self.energies[l] + self.energy_change[l])
                            kwargs = {'A': A, 'mu': mu, 'laser': intensity}
                            if self.shape.lower() == 'voigt':
                                kwargs['fwhmG'] = self.params['FWHMG_' + str(i) + '_to_' + str(j)].value * 1e6
                                kwargs['fwhmL'] = self.params['FWHML_' + str(i) + '_to_' + str(j)].value * 1e6
                            else:
                                kwargs['fwhm'] = self.params['FWHML_' + str(i) + '_to_' + str(j)].value * 1e6
                            D[x, y, laser_index] = bxrho(**kwargs)
                            self.indices.append((x, y, laser_index, i, j, mze, mzg))

        self.D = D

    def _edit_D_matrix(self):
        self.locations = []
        self.transitions = []
        for x, y, laser_index, i, j, mze, mzg in self.indices:
            purity = self.params['Purity'].value
            frac = purity if self.mode[laser_index] == (mze - mzg) else (1.0 - purity) if 0 == -(mze - mzg) else 0
            intensity = frac * self.params['Laser_intensity_' + str(laser_index)].value
            A = self.A_array_used[y, x]
            mu = (self.energies[x] + self.energy_change[x]) - (self.energies[y] + self.energy_change[y])
            self.D[x, y, laser_index].mu = mu
            self.D[x, y, laser_index].A = A
            if self.shape.lower() == 'voigt':
                self.D[x, y, laser_index].gaussian = self.params['FWHMG_' + str(i) + '_to_' + str(j)].value * 1e6
                self.D[x, y, laser_index].lorentzian = self.params['FWHML_' + str(i) + '_to_' + str(j)].value * 1e6
            else:
                self.D[x, y, laser_index].fwhm = self.params['FWHML_' + str(i) + '_to_' + str(j)].value * 1e6
            self.D[x, y, laser_index].laser = intensity
            self.locations.append(mu)
            self.transitions.append((self.F[x], self.F[y]))
        self.locations, indices = np.unique(np.array(self.locations), return_index=True)
        self.transitions = np.array(self.transitions)[indices]

    def _evaluate_matrices(self, f):
        D = np.zeros(self.D.shape)
        for i, j, laser_index, _, _, _, _ in self.indices:
            if laser_index < self.vary_freqs:
                freq = f
            else:
                freq = self.fixed_frequencies[laser_index - self.vary_freqs]
                if self.frequency_mode.lower() == 'offset':
                    freq += f
            D[i, j, laser_index] = self.D[i, j, laser_index](freq)

        D = D.sum(axis=2)
        D = np.transpose(D) + D
        D = D - np.eye(D.shape[0]) * D.sum(axis=0)
        self.M = self.A_array_used + D
        self.decay_rates = np.copy(self.M)
        self.decay_rates[self.characters[0]['y'], self.characters[0]['x']] *= self.params['pi_detect'].value
        self.decay_rates[self.characters[1]['y'], self.characters[1]['x']] *= self.params['sigma_detect'].value
        self.decay_rates = np.tril(self.decay_rates, -1).sum(axis=0)

    def _rhsint(self, y, t):
        """Define the system of ODE's for use in the odeint method from SciPy.
        Note that the input is (y, t)."""
        return np.dot(self.M, y)

    def _rhs(self, t, y):
        """Define the system of ODE's for use in the odeint method from SciPy.
        Note that the input is (y, t)."""
        return np.dot(self.M, y)

    def _jacobian(self, *args):
        return self.M

    def _process_population(self, y):
        raise NotImplementedError('Function should be implemented in child classes!')

    def get_level_population(self):
        return [self.P[ncs-n:ncs].sum() for ncs, n in zip(self.level_counts_cs, self.level_counts)]

    def set_relative_population(self, relative_population):
        for p, (ncs, n) in zip(relative_population, zip(self.level_counts_cs, self.level_counts)):
            self.P[ncs-n:ncs] = p/n
        self.P /= self.P.sum()
        return None

    def __call__(self, x):

        try:
            response = np.zeros(x.size)
            for i, f in enumerate(x.flatten()):
                self._evaluate_matrices(f)
                dt = self.params['Interaction_time'].value / 400
                y = integrate.odeint(self._rhsint, self.P, np.arange(0, self
    .params['Interaction_time'].value, dt), Dfun=self._jacobian)
                response[i] = self._process_population(y)[-1]
            response = response.reshape(x.shape)
        except:
            self._evaluate_matrices(x)
            dt = self.params['Interaction_time'].value / 400
            y = integrate.odeint(self._rhsint, self.P, np.arange(0, self.
    params['Interaction_time'].value, dt), Dfun=self._jacobian)
            response = self._process_population(y)[-1]
        response = self.params['Scale'].value * response
        backgroundparams = [self.params['Background' + str(int(deg))].value
    for deg in reversed(list(range(self.background_degree + 1)))]
        return response + np.polyval(backgroundparams, x)

    def integrate_with_time(self, x, beginning, duration, steps=401, mode='
    integral'):
        backup = self.params['Interaction_time'].value
        self.params['Interaction_time'].value = beginning
        time_vector = np.linspace(beginning, beginning + duration, steps)
        try:
            response = np.zeros(x.size)
            for i, f in enumerate(x.flatten()):
                self._evaluate_matrices(f)
                dt = self.params['Interaction_time'].value / 400
                if not np.isclose(dt, 0):
                    y = integrate.odeint(self._rhsint, self.P, np.arange(0,
    self.params['Interaction_time'].value, dt), Dfun=self._jacobian)
                    y = integrate.odeint(self._rhsint, y[-1, :], time_vector
    , Dfun=self._jacobian)
                else:
                    y = integrate.odeint(self._rhsint, self.P, time_vector,
    Dfun=self._jacobian)
                y = self._process_population(y)
                response[i] = integrate.simps(y, time_vector)
        except:
            self._evaluate_matrices(x)
            dt = self.params['Interaction_time'].value / 400
            if not np.isclose(dt, 0):
                y = integrate.odeint(self._rhsint, self.P, np.arange(0, self
    .params['Interaction_time'].value, dt), Dfun=self._jacobian)
                y = integrate.odeint(self._rhsint, y[-1, :], time_vector,
    Dfun=self._jacobian)
            else:
                y = integrate.odeint(self._rhsint, self.P, time_vector, Dfun
    =self._jacobian)
            y = self._process_population(y)
            if mode == 'integral':
                response = integrate.simps(y, time_vector)
            elif mode == 'mean' or mode == 'average':
                response = np.mean(y)
```

```python
        response = self.params['Scale'].value * response + self.params['Background0'].value
        self.params['Interaction_time'].value = backup
        return response

    def time_response(self, x, beginning, duration, steps=401, populations=[-1]):
        time_vector = np.linspace(beginning, beginning+duration, steps)
        backgroundparams = [self.params['Background' + str(int(deg))].value for deg in reversed(list(range(self.background_degree + 1)))]
        try:
            response = np.zeros((x.shape[0], time_vector.shape[0]))
            pops = np.zeros((x.shape[0], time_vector.shape[0], len(populations)))
            for i, f in enumerate(x.flatten()):
                self._evaluate_matrices(f)
                if not np.isclose(beginning, 0):
                    dt = self.params['Interaction_time'].value / 400
                    y = integrate.odeint(self._rhsint, self.P, np.arange(0, self.params['Interaction_time'].value, dt), Dfun=self._jacobian)
                    begin_pop = y[-1]
                else:
                    begin_pop = self.P
                y = integrate.odeint(self._rhsint, begin_pop, time_vector, Dfun=self._jacobian)
                response[i] = self.params['Scale'].value * self._process_population(y) + np.polyval(backgroundparams, f)
                pops[i] = self._get_fine_populations(y, populations=populations)
        except:
            self._evaluate_matrices(x)
            if not np.isclose(beginning, 0):
                dt = self.params['Interaction_time'].value / 400
                y = integrate.odeint(self._rhsint, self.P, np.arange(0, self.params['Interaction_time'].value, dt), Dfun=self._jacobian)
                begin_pop = y[-1]
            else:
                begin_pop = self.P
            y = integrate.odeint(self._rhsint, begin_pop, time_vector, Dfun=self._jacobian)
            response = self.params['Scale'].value * self._process_population(y) + np.polyval(backgroundparams, x)
            pops = self._get_fine_populations(y, populations=populations)
        return time_vector, response, pops

    def _get_fine_populations(self, y, populations=[-1]):
        pops = []
        for v in y:
            pops.append(
                [v[ncs-n:ncs].sum() for ncs, n in zip(self.level_counts_cs[populations], self.level_counts[populations])]
                )
        return np.array(pops)

class RateModelDecay(RateModel):
    def _process_population(self, y):
        return np.einsum('j,kj->k', self.decay_rates, y)

```

```
class RateModelPolar(RateModel):
    def __init__(self, *args, **kwargs):
        super(RateModelPolar, self).__init__(*args, **kwargs)
        self._convertFMftoMIMJ()

    def _convertFMftoMIMJ(self):
        self.MIlist = []
        self.MJlist = []
        self.MI = np.array([])
        self.MJ = np.array([])
        for i, (J, F, Mf) in enumerate(zip(self.Jlist, self.Flist, self.
    MFlist)):
            I = self.I
            A = self.params['A_level_' + str(i)].value
            J = J[0]

            # Create the array of possible F-values.
            f = np.arange(np.abs(I - J), I + J + 1)

            # Create grids of MI and MJ
            I = np.arange(-I, I + 1)
            J = np.arange(-J, J + 1)
            I, J = np.meshgrid(I, J)

            # Calculate the total projection
            mf = I + J

            # Create an equal-size matrix with the correct
            # F-numbers in each place, depending on the sign of A
            M = np.zeros(I.shape)
            for i, val in enumerate(reversed(f)):
                if np.sign(A) == 1:
                    if i != 0:
                        M[0:-i, i] = val
                        M[-i - 1, i:] = val
                    else:
                        M[:, 0] = val
                        M[-1, :] = val
                else:
                    M[i, 0:- 1 - i] = val
                    M[i:, - 1 - i] = val

            f_select = []
            m_select = []
            for f, m in zip(F, Mf):
                f_select.append(np.isclose(M, f))
                m_select.append(np.isclose(mf, m))
            MI = []
            MJ = []
            for f, mf in zip(f_select, m_select):
                MI.append(I[np.bitwise_and(f, mf)][0])
                MJ.append(J[np.bitwise_and(f, mf)][0])
            self.MIlist.append(MI)
            self.MJlist.append(MJ)
            self.MI = np.append(self.MI, np.array(MI))
            self.MJ = np.append(self.MJ, np.array(MJ))

    def _process_population(self, y):
        return np.einsum(’j,kj->k’, self.MI, y) / self.I
```

A.2 Quantum simulations code

In order to simulate the rotation of the nuclear and electron spin in the applied external field, the following code using the QuTiP package was implemented and used.

```
import qutip as qt
import numpy as np
import pathlib
from scipy.interpolate import interp1d
from scipy.constants import physical_constants
import scipy.integrate as integrate

new_dict_35ar = {(0.5, -0.5): 0.13128594381070291, (0.5, 0.5):
     0.20732073182232927, (1.5, -1.5): 2.0213777697611927e-16, (1.5, -0.5):
     7.1920169974662088e-05, (1.5, 0.5): 0.00032833742111606118, (1.5, 1.5):
     0.10556403066631834, (2.5, -2.5): 1.4664467690758376e-15, (2.5, -1.5):
     1.6986636356929244e-15, (2.5, -0.5): 4.9445999860413073e-07, (2.5, 0.5):
      2.3843510123591184e-06, (2.5, 1.5): 6.2515158772916642e-05, (2.5, 2.5):
      0.00051917050261505765, (3.5, -3.5): 2.4175297242932896e-12, (3.5,
     -2.5): 1.3978814335234604e-12, (3.5, -1.5): 3.0008385031968567e-12,
     (3.5, -0.5): 4.902807170453987e-08, (3.5, 0.5): 3.2620946348784102e-07,
     (3.5, 1.5): 9.1056444057181363e-06, (3.5, 2.5): 0.00012725278722455733,
     (3.5, 3.5): 0.55470773796117456}
new_dict_35ar_noaom = {(0.5, -0.5): 0.062316793834386236, (0.5, 0.5):
     0.063034635024421068, (1.5, -1.5): 0.072290814707542159, (1.5, -0.5):
     0.10002975760771218, (1.5, 0.5): 0.10712964492288424, (1.5, 1.5):
     0.099343084438554566, (2.5, -2.5): 0.036289433728370457, (2.5, -1.5):
     0.028554668255823226, (2.5, -0.5): 0.027288188894721791, (2.5, 0.5):
     0.030899968816904884, (2.5, 1.5): 0.041958684126567808, (2.5, 2.5):
     0.070217031657739737, (3.5, -3.5): 5.1488546116229653e-13, (3.5, -2.5):
     2.1079703129444153e-06, (3.5, -1.5): 3.6477986956642128e-06, (3.5, -0.5)
     : 6.602099465026341e-06, (3.5, 0.5): 1.1929906708342068e-05, (3.5, 1.5):
      2.4085448794856476e-05, (3.5, 2.5): 6.8555536908427504e-05, (3.5, 3.5):
      0.26053036522297141}
new_dict_28na = {(0.5, -0.5): 0.19190123808040271, (0.5, 0.5):
     0.19530372400561233, (1.5, -1.5): 2.2480552435369259e-11, (1.5, -0.5):
     6.7376396348705003e-07, (1.5, 0.5): 6.4054092557849843e-06, (1.5, 1.5):
     0.61278795871828506}
new_dict_26na = {(2.5, -2.5): 0.077576233809443812, (2.5, -1.5):
     0.079430874035601354, (2.5, -0.5): 0.077433252441602402, (2.5, 0.5):
     0.079374503702014418, (2.5, 1.5): 0.084613321942975786, (2.5, 2.5):
     0.091105323777625188, (3.5, -3.5): 0.0069568692200341177, (3.5, -2.5):
     0.00059710468993008864, (3.5, -1.5): 0.00043804576815889736, (3.5, -0.5)
     : 0.00055963127832789057, (3.5, 0.5): 0.0013022663512108647, (3.5, 1.5):
      0.0040675363422603408, (3.5, 2.5): 0.042760074605092433, (3.5, 3.5):
     0.45378496203572238}
new_dicts = {’35Ar’: new_dict_35ar, ’26Na’: new_dict_26na, ’28Na’:
     new_dict_28na, ’35ArNoAOM’: new_dict_35ar_noaom}

```

```
BOHR = 13.996245042 * 10**9
NUCL = 7.622593285 * 10**6

moments = {'35Ar': 0.633, '35ArNoAOM': 0.633, '28Na': 2.426, '26Na': 2.851, }

masses = {'35Ar': 5.80777807 * 10**-26, '35ArNoAOM': 5.80777807 * 10**-26, '28Na': 27.998938 * physical_constants['atomic mass unit-kilogram relationship'][0], '26Na': 25.992633 * physical_constants['atomic mass unit-kilogram relationship'][0], }

spins = {'35Ar': 3/2, '35ArNoAOM': 3/2, '28Na': 1, '26Na': 3, }
atomics = {'35Ar': (2, 1.5, 0.5), '35ArNoAOM': (2, 1.5, 0.5), '28Na': (0.5, 1.0, 0.5), '26Na': (0.5, 1.0, 0.5), }

hyperfines = {'35Ar': 266.0e0, '35ArNoAOM': 266.0e0, '28Na': 154.7e0, '26Na': 60.6e0, }

def generate_matrix(selection='35Ar'):
    """Generate Mathematica-compliant version of the interaction matrix, to calculate the eigenvalues as a function of magnetic field.

    Parameters
    ----------
    selection: str
        Key for the dictionaries defined above
    """
    SELECTION = selection

    MAGNETIC_MOMENT = moments[SELECTION]

    I = spins[SELECTION]
    J, L, S = atomics[SELECTION]

    if np.isclose(J, 0):
        LANDE = 0
    else:
        LANDE = (J*(J+1)-S*(S+1)+L*(L+1))/(2*J*(J+1)) + 2*(J*(J+1)+S*(S+1)-L*(L+1))/(2*J*(J+1))

    F = [temp for temp in np.arange(np.abs(I-J), I+J+1,1) for _ in range(int(np.abs(2*temp+1)))]
    mF = [a for temp in np.arange(np.abs(I-J), I+J+1,1) for a in np.arange(-temp, temp+1, 1)]

    A = hyperfines[SELECTION]
    if A < 0:
        F = F[::-1]
        mF = mF[::-1]
    F = np.array(F)
    mF = np.array(mF)

    Ix, Iy, Iz = qt.jmat(I)

    Iz = qt.jmat(I, 'z')
    Id = Iz.dims[0][0]
    Jz = qt.jmat(J, 'z')
    Jd = Jz.dims[0][0]

    Iz = qt.tensor(qt.jmat(I, 'z'), qt.qeye(Jd))
    Ix = qt.tensor(qt.jmat(I, 'x'), qt.qeye(Jd))
    Iy = qt.tensor(qt.jmat(I, 'y'), qt.qeye(Jd))

    Jz = qt.tensor(qt.qeye(Id), qt.jmat(J, 'z'))
    Jx = qt.tensor(qt.qeye(Id), qt.jmat(J, 'x'))
    Jy = qt.tensor(qt.qeye(Id), qt.jmat(J, 'y'))

    IZcoeff = -MAGNETIC_MOMENT*NUCL

    JZcoeff = LANDE*BOHR

    HYP = A * (Iz * Jz + Ix * Jx + Iy * Jy)*1e6
    import sympy
    M = sympy.Matrix(HYP.full())
    B = sympy.symbols('B')
    magn = B*IZcoeff*Iz.full() + B*JZcoeff*Jz.full()
    total = M+magn
    entries = total.tolist()
    print('{')
    for entry in entries:
        print('{' + ','.join([str(e) for e in entry]) + '},')
    print('}')

def simulate(selection='35Ar', energy=50, scaling=1.0, force_recalc=False, angle_numbers=10, pop_dict=None, freq=None):
    """Simulate the spin-dynamics in the VITO beamline.

    Parameters
    ----------
    selection: str
        Key for the dictionaries defined above.
    energy: float
        Kinetic energy (in keV) of the beam, determines the TOF.
    scaling: float
        Scales the total strength of the magnetic field.
    force_recalc: boolean
        Forces the recalculation instead of loading the saved file.
    angle_numbers: int
        Number of angles between 0 and 2pi to use for the rotation of the initial ket.
    pop_dict: dictionary
        Dictionary (format given above) for custom used populations.
    freq: float
        Identifier added to the filename.
    """
    SELECTION = selection
    if pop_dict is None:
        pop_dict = new_dicts[SELECTION]
        features = ''
    else:
        features = str(freq)

    MASS = masses[SELECTION]

    MAGNETIC_MOMENT = moments[SELECTION]

    I = spins[SELECTION]
    J, L, S = atomics[SELECTION]

    if np.isclose(J, 0):
        LANDE = 0
    else:
        LANDE = (J*(J+1)-S*(S+1)+L*(L+1))/(2*J*(J+1)) + 2*(J*(J+1)+S*(S+1)-L
     *(L+1))/(2*J*(J+1))

    A = hyperfines[SELECTION]

    ENERGY = energy*10**3
    EV2JOULES = 1.602176565 * 10**-19

    SPEED = (2*ENERGY*EV2JOULES/MASS)**0.5

    # Magnetic field definition

    y_full = np.array([120, 119, 118, 117, 116, 115, 114, 113, 112, 111,
     110, 109, 108, 107, 106, 105, 104, 103, 102, 101, 100, 99, 98, 97, 96,
     95, 94, 93, 92, 91, 90, 89, 88, 87, 86, 85, 84, 83, 82, 81, 80, 78, 76,
     74, 72])
    y_full = y_full / 100
    Bx_full = -10**-4*np.array([0.8, 0.7, 1.6, 5.5, 8.33, 5.1, 1.1, -1.9, 2,
      8.3, 4.8, 1.7, 0.01, -0.9, -3.6, -4.3, -4.2, -3.8, -3.6, -4.4, -7.6,
     -9.5, -11.6, -17.4, -17, -8.2, -1.5, -2.6, 17.3, 24.7, 34.8, 31, 47.3,
     55.4, 41.6, 42.8, 70.5, 83.8, 89, 110, 91.2, 21.3, -70.8, 36.1, -42.2])
    By_full = -10**-4*np.array([-3.4, -3.9, -3.6, -0.7, 1.6, -1.1, -4.9, -8,
      -5.4, -0.1, -3.9, -8.4, -12.2, -15.6, -18.7, -22.8, -27.2, -32.2,
     -38.2, -44.2, -55.9, -68.2, -81.8, -99.1, -118.5, -133.7, -159.4, -192,
     -237.2, -283.9, -346, -421.2, -524.2, -621.9, -765.7, -926.3, -1096.5,
     -1358.3, -1698.2, -2072.6, -2554, -3209.7, -3519.3, -3586.1, -3607.2])
    Bz_full = -10**-4*np.array([-30.6, -32.9, -35.7, -42.2, -64.4, -81.2,
     -93.3, -95.6, -95.5, -119.2, -144.4, -157.7, -165.7, -170.2, -173.3,
     -175.5, -177.3, -179.3, -181.7, -183, -183.1, -179.5, -171.7, -155.3,
     -126.7, -110.6, -112.6, -116.4, -116.8, -118.5, -118.7, -114.3, -96.3,
     -98, -112.4, -135.3, -157.6, -203.3, -242.15, -282.5, -307.3, -234.8,
     -126.8, -100, -87.5])
    y_full -= y_full.min()
    y_full = y_full[::-1]

    y_magnet = np.array([120, 119, 118, 117, 116, 115, 114, 113, 112, 111,
     110, 109, 108, 107, 106, 105, 104, 103, 102, 101, 100, 99, 98, 98, 97,
     96, 96, 95, 94, 94, 93, 92, 92, 91, 90, 90, 89, 88, 88, 87, 86, 85, 84,
     82, 80, 78, 76, 74, 72, 70])
    y_magnet = y_magnet / 100
    Bx_magnet = -10**-4*np.array([0.15, 0.7, 0.6, 1.1, 1.5, -0.09, 1, 0.7,
     0.7, 0.6, -0.1, 0.8, 0.8, 0.9, -0.2, 0, -0.9, -1, -2.4, -2.7, -3.9,
     -4.1, -6.9, 1.16, -9.1, -11.6, 1.7, -12.1, -6.5, 4.3, -12.3, -20.2, 0.9,
      -21.3, -19, 2, 2.9, 0.4, 8, -34.5, -16.7, -23.168, -63, 0.3, 7.9, 24.1,
      85.9, -165.2, -60.5, -42.1])
    By_magnet = -10**-4*np.array([-5.2, -5.6, -6, -6.5, -7.2, -7.6, -8.9,
     -9.8, -11.1, -12.7, -14.3, -16.1, -18.2, -21.1, -24.3, -28, -32.2,
     -37.1, -43.3, -49.6, -59, -68.9, -84.5, -77.9, -101.2, -121.8, -109.5,
     -144.2, -175, -161.6, -213.6, -256.3, -244.8, -309.6, -379.2, -359.4,
     -454.7, -549.2, -530.8, -648, -761.5, -910.8, -1051, -1731, -2549.5,
     -3237.4, -3526.8, -3591.6, -3609.7, -3613.1])
    Bz_magnet = -10**-4*np.array([-0.8, -0.7, -1, -0.8, -0.8, -1, -1.2,
     -1.3, -1.3, -1.5, -1.6, -1.8, -2.2, -2.6, -3, -3.4, -4.1, -5, -6.1,
     -7.3, -8, -9.6, -11.9, -10.2, -14.8, -18.3, -15.1, -22, -26.2, -23.4,
     -32.9, -39.4, -36.1, -48.4, -59.8, -55.1, -65.5, -79.2, -74.5, -97,
     -113.5, -136.9, -166.4, -260.5, -317.1, -223.4, -125.1, -96, -92.7,
     -80.4])
    y_magnet, ind = np.unique(y_magnet, return_index=True)
    Bx_magnet, By_magnet, Bz_magnet = Bx_magnet[ind][::-1], By_magnet[ind
     ][::-1], Bz_magnet[ind][::-1]
    y_magnet = y_magnet[::-1]

    y_magnet -= y_magnet.min()
    y_magnet = y_magnet[::-1]

    speed = SPEED
    y = y_magnet

    t = (y-y[0]) / speed

    t_full = (y_full-y_full[0]) / speed
    Bx_full[:5] = 0*Bx_full[:5]
    By_full[:5] = 0*By_full[:5]

    Bx_magnet[:5] = 0*Bx_magnet[:5]
    By_magnet[:5] = 0*By_magnet[:5]

    Bz_full_inter = interp1d(t_full, Bz_full, bounds_error=False, fill_value
     =(Bz_full[0], Bz_full[-1]), kind=’cubic’)
    Bx_full_inter = interp1d(t_full, Bx_full, bounds_error=False, fill_value
     =(Bx_full[0], Bx_full[-1]), kind=’cubic’)
    By_full_inter = interp1d(t_full, By_full, bounds_error=False, fill_value
     =(By_full[0], By_full[-1]), kind=’cubic’)

    Bz_magnet_inter = interp1d(t, Bz_magnet, bounds_error=False, fill_value
     =(Bz_magnet[0], Bz_magnet[-1]), kind=’cubic’)
    Bx_magnet_inter = interp1d(t, Bx_magnet, bounds_error=False, fill_value
     =(Bx_magnet[0], Bx_magnet[-1]), kind=’cubic’)
    By_magnet_inter = interp1d(t, By_magnet, bounds_error=False, fill_value
     =(By_magnet[0], By_magnet[-1]), kind=’cubic’)

    Bz_transitional = Bz_full_inter(t) - Bz_magnet_inter(t)
    Bx_transitional = Bx_full_inter(t) - Bx_magnet_inter(t)
    By_transitional = By_full_inter(t) - By_magnet_inter(t)

    Bz_transitional_inter = interp1d(t, Bz_transitional, bounds_error=False,
      fill_value=(Bz_transitional[0], Bz_transitional[-1]), kind=’cubic’)
    Bx_transitional_inter = interp1d(t, Bx_transitional, bounds_error=False,
      fill_value=(Bx_transitional[0], Bx_transitional[-1]), kind=’cubic’)
    By_transitional_inter = interp1d(t, By_transitional, bounds_error=False,
      fill_value=(By_transitional[0], By_transitional[-1]), kind=’cubic’)

    interZ = lambda t: (Bz_transitional_inter(t)+scaling*Bz_magnet_inter(t))
    interX = lambda t: (Bx_transitional_inter(t)+scaling*Bx_magnet_inter(t))
    interY = lambda t: (By_transitional_inter(t)+scaling*By_magnet_inter(t))

    # Operator definitions

    Ix, Iy, Iz = qt.jmat(I)

    Iz = qt.jmat(I, 'z')
    Id = Iz.dims[0][0]
    Jz = qt.jmat(J, 'z')
    Jd = Jz.dims[0][0]

    Iz = qt.tensor(qt.jmat(I, 'z'), qt.qeye(Jd))
    Ix = qt.tensor(qt.jmat(I, 'x'), qt.qeye(Jd))
    Iy = qt.tensor(qt.jmat(I, 'y'), qt.qeye(Jd))
    Im = qt.tensor(qt.jmat(I, '-'), qt.qeye(Jd))
    Ip = qt.tensor(qt.jmat(I, '+'), qt.qeye(Jd))

    I_op = Ix + Iy + Iz

    Jz = qt.tensor(qt.qeye(Id), qt.jmat(J, 'z'))
    Jx = qt.tensor(qt.qeye(Id), qt.jmat(J, 'x'))
    Jy = qt.tensor(qt.qeye(Id), qt.jmat(J, 'y'))
    Jm = qt.tensor(qt.qeye(Id), qt.jmat(J, '-'))
    Jp = qt.tensor(qt.qeye(Id), qt.jmat(J, '+'))

    J_op = Jx + Jy + Jz

    def IXcoeff(t, args):
        BX = interX(t+args[0])
        return -MAGNETIC_MOMENT*NUCL*BX/I

    def IYcoeff(t, args):
        BY = interY(t+args[0])
        return -MAGNETIC_MOMENT*NUCL*BY/I

    def IZcoeff(t, args):
        BZ = interZ(t+args[0])
        return -MAGNETIC_MOMENT*NUCL*BZ/I

    def JXcoeff(t, args):
        BX = interX(t+args[0])
        return LANDE*BOHR*BX

    def JYcoeff(t, args):
        BY = interY(t+args[0])
        return LANDE*BOHR*BY

    def JZcoeff(t, args):
        BZ = interZ(t+args[0])
        return LANDE*BOHR*BZ

    HYP = A * (Iz * Jz + Ix * Jx + Iy * Jy)*1e6

    H = [HYP, [Ix, IXcoeff], [Iy, IYcoeff], [Iz, IZcoeff], [Jx, JXcoeff], [Jy, JYcoeff], [Jz, JZcoeff]]

    # Preparing the initial ket

    eigenvals, eigenstates = HYP.eigenstates()
    F = []
    MF = []
    states = []
    for val in np.unique(np.around(eigenvals, -4)):
        indices = np.isclose(eigenvals, val)
        f = np.around((qt.expect(Jz, eigenstates[indices])+qt.expect(Iz, eigenstates[indices])).max(), 1)
        f = [f]*(indices.sum())
        F.extend(f)
        MF.extend(np.around((qt.expect(Jz, eigenstates[indices])+qt.expect(Iz, eigenstates[indices])), 1))
        states.extend(eigenstates[indices])
    weights = np.array([pop_dict[f, mf] for f, mf in zip(F, MF)])
    initial = sum([(w**0.5)*s for w, s in zip(weights, states)]).unit()

    # Define calculation objects

    time = np.linspace(0, 1.5*t[-1], 1500)

    options = qt.Options(nsteps=5000)
    axis = np.array([interX(0), interY(0), interZ(0)])
    axis = axis/((axis**2).sum()**0.5)
    spins_rot = np.array([Ix+Jx, Iy+Jy, Iz+Jz])*2

    # Prepare naming convention

    name = str(SELECTION+str(energy)+init+interaction+str(scaling))
    name = name+’_’+str(angle_numbers)+’angles’
    if features != ’’:
        name = name+’_’+features

    try: # Load the file. If it can’t be loaded, or a recalculation is demanded, throws an error to be caught
        if force_recalc:
            raise ArithmeticError
        load_folder = pathlib.Path(’Data/’+SELECTION+’/’)
        output = qt.qload((load_folder / name).as_posix())
    except: # Do the calculation for every angle requested. Concatenate the results and save to a file
        def calculate(angle, axis, spins_rot, H, initial, time, options):
            operator = (-1j*angle*np.dot(axis,spins_rot)).expm()
            return qt.mesolve(H, operator*initial, time, c_ops=[], e_ops=[], args=[0], options=options, progress_bar=True)
        angles = np.linspace(0, np.pi, angle_numbers+1)
        angles = angles[:-1]
        output = qt.serial_map(calculate, angles, task_args=(axis, spins_rot, H, initial, time, options), num_cpus=1)

        output.append(angles.tolist())
        save_folder = pathlib.Path(’Data/’ + SELECTION + ’/’)
        save_folder.mkdir(parents=True, exist_ok=True)
        qt.qsave(output, (save_folder / name).as_posix())

    operators = Iz, Ix, Iy, Jz, Jx, Jy
    return output, time, operators, I, J, (interX, interY, interZ), HAMIL
```

Appendix B

Optical detection simulations

You will never feel my touch upon your skin, your face; For I–; I'm lost without a trace
—Stratovarius, *Lost Without A Trace*

In the optical setup at both COLLAPS and VITO, fluorescent decay from the excited ionic or atomic beam is detected with PMT's. In order to maximize the detection efficiency, a set of lenses is used to focus the emitted light onto the surface of the PMT's. As the index of refraction of the material of the lenses is wavelength dependent, the optimal point to place the PMT needs to be recalculated each time. The propagation of light through the optical setup is calculated via ray tracing techniques and the steps taken in the calculation are described here.

B.1 Lens system

As the ion/atomic beam emits light from every location overlapping with the laser beam, the use of aspherical lenses minimizes the loss of detection efficiency from *spherical aberration*, where the focal point of the beam changes as the light enters the lens off-axis. The aspherical lenses in the VITO setup have been produced by Asphericon GmbH (product number A100-100LPX-S-U), are plano-convex lenses and have a diameter of 100 mm. The curved surface has been designed to have the form [88]

$$z\left(r\right)=C\frac{r^{2}}{1+\sqrt{1-\left(1+K\right)C^{2}}}+A_{2}r^{2}+A_{4}r^{4}+A_{6}r^{6}+\ldots \qquad \text{(B.1)}$$

where the z-axis is assumed to be the optical axis of the lens. $z\left(r\right)$ is the displacement along the z-axis from the surface point on the central axis as a function of distance r. Table B.1 gives the parameters for the purchased lenses.

Two such aspherical lenses form the imaging system for each PMT. Figure B.1 shows the CAD design of the imaging system. Indicated on this schematic drawing is the distance from the surface of the lens to the center of the beamline (76.2 mm), the aperture of the lenses (92 mm), the diameter of the PMT surface (47.5 mm) and the distance of the PMT to the flat edge of the second lens. This distance can be increased by using a mechanical spacer to maximize the optical detection efficiency. When the wavelength indicates a closer distance is needed, a different mounting system is required.

B.2 Index of refraction

The index of refraction for the lenses can be calculated from the Sellmeier equation or the Cauchy equation, empirical equations relating the wavelength and index of refraction for different glasses. The Sellmeier equation used is

$$n^{2}\left(\lambda\right)=A_{0}+A_{1}\lambda^{2}+A_{2}\lambda^{-2}+A_{3}\lambda^{-4}+A_{4}\lambda^{-6}+A_{5}\lambda^{-8}, \qquad \text{(B.2)}$$

Table B.1: Aspherical surface parameters

Parameter	Value
C	$0.016\,767\,270\,288\,4$
K	$-1.035\,958\,114\,8$
A_2	$1.287\,485\,311\,9 \times 10^{-3}\,\mathrm{mm}^{-2}$
A_4	$4.285\,329\,204\,317 \times 10^{-7}\,\mathrm{mm}^{-4}$
A_6	$2.660\,598\,011\,977 \times 10^{-11}\,\mathrm{mm}^{-6}$
A_8	$1.889\,503\,380\,77 \times 10^{-15}\,\mathrm{mm}^{-8}$
A_{10}	$9.063\,010\,624\,48 \times 10^{-21}\,\mathrm{mm}^{-10}$

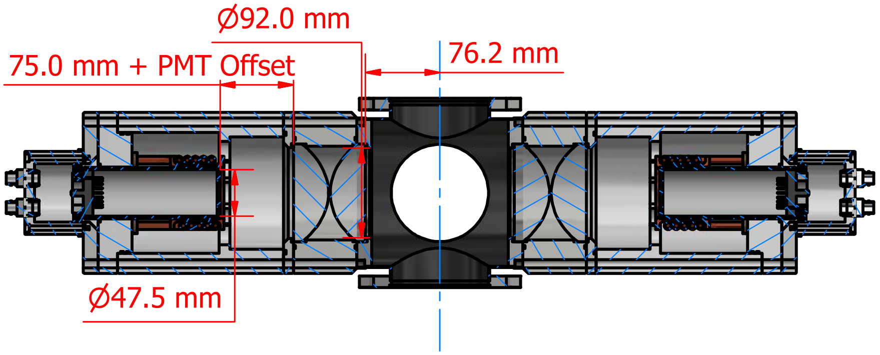


Figure B.1: Optical detection setup with cooled housing as modelled in Inventor. Relevant dimensions have been indicated. Note that the PMT Offset is the spacer thickness and 75 mm is the starting distance (defined further in the text).

while the Cauchy equation is

$$n^2(\lambda) = 1 + \frac{B_1\lambda^2}{\lambda^2 - C_1} + \frac{B_2\lambda^2}{\lambda^2 - C_2} + \frac{B_3\lambda^2}{\lambda^2 - C_3} \tag{B.3}$$

For BK7 crown glass (also known under the name Schott glass or B270 glass), which is the material of the ordered lenses, the Sellmeier equation has been used (see Table B.2). For some other materials that can be used to construct lenses, the Cauchy coefficients are also tabulated.

Table B.2: Sellmeier and Cauchy equation parameters for different glasses.

Sellmeier coefficients	B270	Cauchy coefficients	Lithosil	Corning 7960
A_0	2.287 782 8	B_1	$6.694\,226 \times 10^{-1}$	$6.852\,562\,45 \times 10^{-1}$
A_1	$-9.314\,872\,3 \times 10^{-3}\,\mu\text{m}^{-2}$	B_2	$4.345\,839 \times 10^{-1}$	$4.188\,670\,77 \times 10^{-1}$
A_2	$1.098\,644\,3 \times 10^{-2}\,\mu\text{m}^{2}$	B_3	$8.716\,947 \times 10^{-1}$	$5.111\,042\,39 \times 10^{-1}$
A_3	$4.846\,520\,3 \times 10^{-4}\,\mu\text{m}^{4}$	C_1	$4.480\,112 \times 10^{-3}\,\mu\text{m}^{2}$	$4.619\,907\,90 \times 10^{-3}\,\mu\text{m}^{2}$
A_4	$-3.394\,473\,8 \times 10^{-5}\,\mu\text{m}^{6}$	C_2	$1.328\,470 \times 10^{-2}\,\mu\text{m}^{2}$	$1.340\,584\,57 \times 10^{-2}\,\mu\text{m}^{2}$
A_5	$1.695\,855\,4 \times 10^{-6}\,\mu\text{m}^{8}$	C_3	$9.534\,148 \times 10^{1}\,\mu\text{m}^{2}$	$5.647\,696\,99 \times 10^{1}\,\mu\text{m}^{2}$

B.3 Refraction of light

At the interface between two media with different refraction indices, the path of the light has to break according to Snell's law. Snell's law in 3D can be stated as [89]

$$c = -\hat{n} \cdot \hat{\imath}, \tag{B.4}$$

$$\hat{e} = \frac{n_1}{n_2}\hat{\imath} + \left(\frac{n_1}{n_2} c - \sqrt{1 - \left(\frac{n_1}{n_2}\right)^2 (1 - c^2)} \right) \hat{n}, \tag{B.5}$$

where $\hat{\imath}$ is the direction of the incident ray, $\hat{n}$ is the normal vector of the curved surface and n_1 and n_2 are the refraction indices of the first and second medium. As the incident ray direction is known and the Sellmeier equation can be used to calculate the refraction index, only the normal vector of the surface still needs to be calculated.

Instead of deriving the formulation of the normal vector analytically, it can also be approximated numerically. The normal vector is calculated by taking the lens surface at the incident point, and calculating the lens surface position when x and y (assuming z is the optical axis) are slightly varied. The vector product of the two vectors connecting the incident point with the two additional points results in the normal vector on the surface in the incident point.

B.4 Emission pattern

The decay from an excited state to a lower lying level, and subsequent generation of a photon, is described by the dipole operator. The emission pattern can be calculated from quantum mechanics [90, 91], yielding two different angular distributions:

$$I(\theta) \propto \sin^2(\theta) \tag{B.6}$$

$$I(\theta) \propto 1 + \cos^2(\theta) \tag{B.7}$$

for decays with $\Delta m_F = 0$ and $\Delta m_F = \pm 1$ respectively. These waves are respectively the π and σ waves and are separately propagated through the optical setup.

As the distribution is $\sin^2(\theta)$ distributed and θ is the angle between the oscillation direction of the electrical field, it follows that no radiation is emitted along the oscillation path. Assuming that the oscillation path is vertical along the beamline, since the laser light is vertically polarized in the COLLAPS setup, the dipole emission pattern for π-waves indicates most light from the signal will be emitted in the horizontal plane. As the lenses are located perpendicular to the beam direction, the $\sin^2(\theta)$ distribution will have the largest impact on the signal.

The *hit-and-miss* algorithm is used to generate the initial directions for the rays. For each ray, a random direction vector $\boldsymbol{d} = (x, y, z)$ is generated within the cube $x, y, z \in [-3, 3]$. If the endpoint of this vector is within the boundary defined by the function $r(\theta) = \sin^2(\theta)$, respectively $r(\theta) = 1 + \cos^2(\theta)$, the direction vector is accepted and normalized. Otherwise, another direction vector is randomly generated.

In the simulation code, only one set of lenses is simulated. The axis sytem assumed in building the setup is that the optical axis is the x-axis and the beamline axis is the y-axis. Therefore, x is restricted to $[0, 1.5]$ to select only the part of the cube that would point in the direction of x.

The emission pattern is combined with the position from which these rays are released. Several options are available in the ray tracer:

1. Specific positions are supplied to the ray tracer along with a number of rays. This will release this amount of rays from each of those specific y-positions, with $y = 0$ corresponding to the symmetry axis of the lens system.

2. When the option to use a distribution is used, the y-position is uniformly random distributed between the minimal and maximal position provided. The number of rays now indicates the total number of rays.

3. The size of the laser spot can also be used, and will distribute the starting x and z position uniformly over a circle with the given diameter. Setting this to 0 or lower will restrict the x and z coordinates to 0.

B.5 Simulation scripts and output

The python script *COLLAPS_raytracer.py* contains the simulation code for the optical setup as described here. A sequential ray tracer tracks the emitted rays to the next optical element and removes the rays that exit the yz-area corresponding to the aperture of the lenses. It generates a compressed .npz file containing the ray positions and directions of the signal and background rays when the rays have arrived at the flat edge of the second lens, along with the number of rays originally emitted. This script is used from the command line and understands the following arguments:

-wavelength Supply the wavelength of the light in nm (default: 811)

-material Supply the material of the lenses (b270, corning7960 or lithosil, default: b270)

-positions Supply the position offsets from which to release the rays, in mm (default: 0.0 5.0 10.0 15.0 20.0 25.0)

-rays Supply the number of rays to be released from each position (default: 10000)

-laserbeamsize Supply the diameter of the laser beam in mm (default: 5)

–**distributed** If True, N rays are released from a random released position between the minimum and maximum supplied instead of N rays from each position (default: False)

The script *plot_COLLAPS_raytracer_data.py* will process the data and propagate the rays onto the surface defined by the PMT as in Figure B.1. Additionally, the PMT is assumed to be covered except for a horizontal slit. The detection efficiency is calculated by counting the number of rays that are incident on the PMT inside this slit. The script can be called with the following arguments:

filename Filename of the datafile

-startingdistance Distance between the second lens and the PMT without spacer in mm (default: 47)

-pmtsize PMT diameter in mm (default: 47.5)

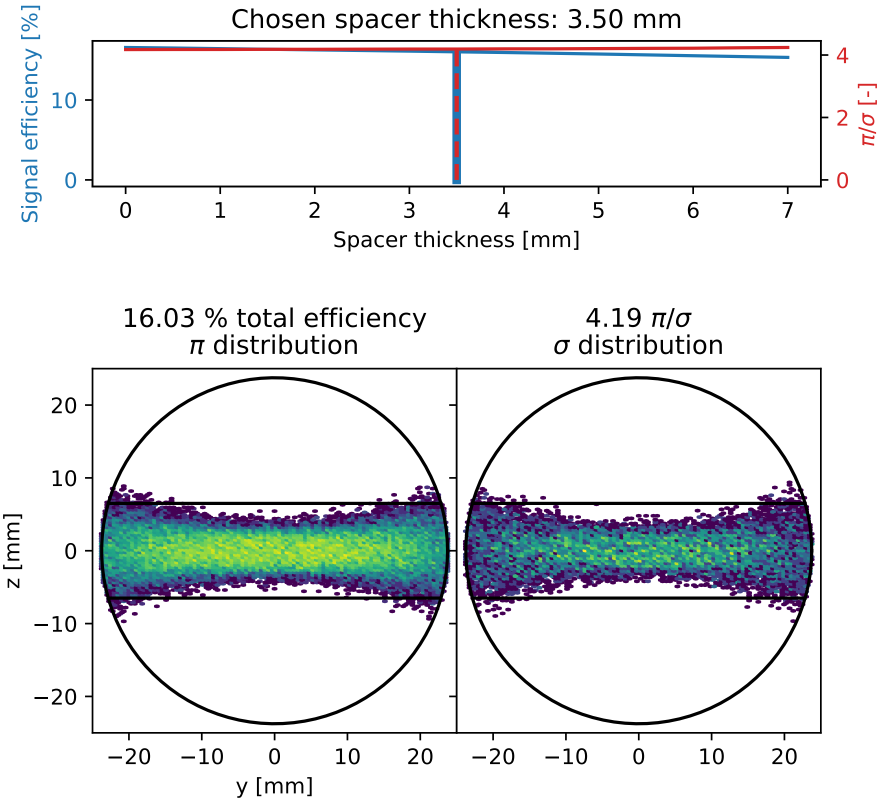


Figure B.2: Result of calculating the light propagation for 811 nm light through the setup.

-masksize Diameter inside which the ray distribution is drawn (default: 47.5)

-spacerthickness Maximal spacer thickness in mm (default: 100)

-slitsize Size of the slit in mm (default: 13)

–**interactive** If True, the interactive plots are shown before the PDF is saved (default: False)

Of these, only the filename is a required argument.

The resulting pattern of light on the plane will be drawn, along with the calculated light detection efficiency and the σ/π ratio, representing the difference in detection efficiency for the different emission patterns. The optimal spacer thickness for maximal signal efficiency is automatically calculated and set as initial thickness. When the interactive flag is given, a GUI is opened with all plots along with a slider for the spacer thickness. The initial value of the slider is the result of the optimizer. When this window is closed, the final value of the slider is used to generate the figures saved to the PDF. The calculated light distribution for a wavelength of 811 nm with a spacer thickness of 3.5 mm is shown in Figure B.2.

Note that the signal efficiency is calculated as the amount of π-rays incident on the PMT divided by the number of π-rays originally released. As only rays propagating along the positive x-axis are generated, this efficiency is for the total system of 2 PMT's across from each other. The detection efficiency of a single PMT is half of this number. The ratio of detected π-rays to σ-rays is also calculated and displayed.

B.6 Source code

B.6.1 Ray tracing code

```
import numpy as np
import time
import scipy.optimize as optimize
import argparse

def hit_and_miss_3d_dipole_donut(n):
    pos = 6*np.random.rand(n, 3)-3
    pos[:, 0] = np.abs(pos[:, 0])

    r = (pos**2).sum(axis=1)**0.5
    pol = np.arccos(pos[:, 2]/r)
    azi = np.arctan2(pos[:, 1], pos[:, 0])

    accepted = r<=np.sin(pol)**2
    while not np.all(accepted):
        naccepted = ~accepted
        naccepted_num = naccepted.sum()
        pos[naccepted] = 6*np.random.rand(naccepted_num, 3)-3
        pos[naccepted, 0] = np.abs(pos[naccepted, 0])
        r[naccepted] = (pos[naccepted]**2).sum(axis=1)**0.5
        pol[naccepted] = np.arccos(pos[naccepted, 2]/r[naccepted])
        azi[naccepted] = np.arctan2(pos[naccepted, 1], pos[naccepted, 0])
        accepted[naccepted] = r[naccepted] <= np.sin(pol[naccepted])**2
    return pos

def hit_and_miss_3d_dipole_dumbbell(n):
    pos = 6*np.random.rand(n, 3)-3
    pos[:, 0] = np.abs(pos[:, 0])

    r = (pos**2).sum(axis=1)**0.5
    pol = np.arccos(pos[:, 2]/r)
    azi = np.arctan2(pos[:, 1], pos[:, 0])

    accepted = r<=(1+np.cos(pol)**2)
    while not np.all(accepted):
        naccepted = ~accepted
        naccepted_num = naccepted.sum()
        pos[naccepted] = 6*np.random.rand(naccepted_num, 3)-3
        pos[naccepted, 0] = np.abs(pos[naccepted, 0])
        r[naccepted] = (pos[naccepted]**2).sum(axis=1)**0.5
        pol[naccepted] = np.arccos(pos[naccepted, 2]/r[naccepted])
        azi[naccepted] = np.arctan2(pos[naccepted, 1], pos[naccepted, 0])
        accepted[naccepted] = r[naccepted] <= (1+np.cos(pol[naccepted])**2)
    return pos

def sellmeier(lamda, material='b270'):
    if material == 'corning7960':
        B1 = 6.85256245e-1
        B2 = 4.18867077e-1
        B3 = 5.11104239e-1
        C1 = 4.61990790e-3
        C2 = 1.34058457e-2
        C3 = 5.64769699e1
    elif material == 'lithosil':
        B1 = 6.694226e-1
        B2 = 4.345839e-1
        B3 = 8.716947e-1
        C1 = 4.480112e-3
        C2 = 1.328470e-2
        C3 = 9.534148e1
    elif material == 'b270':
        A0 = 2.2877828
        A1 = -9.3148723e-3
        A2 = 1.0986443e-2
        A3 = 4.8465203e-4
        A4 = -3.3944738e-5
        A5 = 1.6958554e-6
        return (A0+A1*lamda**2+A2*(lamda**-2)+A3*(lamda**-4)+A4*(lamda**-6)+
     A5*(lamda**-8))**0.5
    ls = lamda*lamda
    return (1+ (B1/(ls-C1)+B2/(ls-C2)+B3/(ls-C3))*ls)**0.5

def sag(arr):
    y = arr[0]
    z = arr[1]
    rs = y*y+z*z
    r = rs**0.5
    Ap = 100
    C = 0.0167672702884
    K = -1.0359581148
    A2 = 1.2874853119e-3
    A4 = 4.285329204317e-7
    A6 = 2.660598011977e-11
    A8 = 1.88950338077e-15
    A10 = 9.06301062448e-21
    sag = C*rs/(1+(1-(1+K)*C*C*rs)**0.5)+A2*rs+A4*rs**2+A6*rs**3+A8*rs**4+A10*rs**5
    rs = (Ap/2)**2
    sag[r>Ap/2] = C*rs/(1+(1-(1+K)*C*C*rs)**0.5)+A2*rs+A4*rs**2+A6*rs**3+A8*rs**4+A10*rs**5
    return sag

def snell_3d(n1, n2, s1, normal_vector):
    index_ratio = n1/n2
    try:
        cost = np.atleast_2d(np.einsum('ai,ai->a', -normal_vector, s1)).T
    except ValueError:
        cost = np.atleast_2d(np.dot(-normal_vector, s1.T)).T
    return index_ratio*s1+(index_ratio*cost-np.sqrt(1-index_ratio*index_ratio*(1-cost*cost)))*normal_vector

def normal_vector(x, lens):
    if lens == 1:
        f = lens1
    else:
        f = lens2
    p1 = np.hstack([np.atleast_2d(f((x[:, 1], x[:, 2]))).T, x[:, 1:]])
    x[:, 1] = x[:, 1]+1e-12
    p2 = np.hstack([np.atleast_2d(f((x[:, 1], x[:, 2]))).T, x[:, 1:]])
    x[:, 1] = x[:, 1]-1e-12
    x[:, 2] = x[:, 2]+1e-12
    p3 = np.hstack([np.atleast_2d(f((x[:, 1], x[:, 2]))).T, x[:, 1:]])
    v1 = p2-p1
    v2 = p3-p1
    v1 = v1 / np.atleast_2d(((v1**2).sum(axis=1)**0.5)).T
    v2 = v2 / np.atleast_2d(((v2**2).sum(axis=1)**0.5)).T
    direc = np.cross(v2, v1)
    if lens == 3:
        direc = -direc
    return direc/np.atleast_2d((direc**2).sum(axis=1)).T**0.5

def lens1(arr):
    return lens_position+lens_thickness-sag(arr)

def lens2(arr):
    return lens_position+lens_thickness+gap+sag(arr)

def first_prop(ray_pos, ray_direc):
    def first_intersection(x, ray_pos, ray_direc):
        new_ray = ray_pos + np.abs(np.atleast_2d(x).T)*ray_direc
        return_val = new_ray[:, 0] - lens_position
        return return_val

    for i in range(0, ray_pos.shape[0], stepping):
        result = optimize.root(first_intersection, np.ones(ray_pos[i:i+stepping].shape[0])*76, args=(ray_pos[i:i+stepping], ray_direc[i:i+stepping]))
        try:
            distance = np.vstack([distance, np.abs(np.atleast_2d(result.x).T)])
        except (ValueError, UnboundLocalError):
            distance = np.abs(np.atleast_2d(result.x).T)
        rp = ray_pos[i:i+stepping]
```

```
            rd = ray_direc[i:i+stepping]

    ray_pos = ray_pos + distance*ray_direc
    ray_direc = snell_3d(r_gap, r_lens, ray_direc, flat_1)
    return ray_pos, ray_direc

def second_prop(ray_pos, ray_direc):
    def first_intersection(x, ray_pos, ray_direc):
        new_ray = ray_pos + np.abs(np.atleast_2d(x).T)*ray_direc
        return new_ray[:, 0] - lens1((new_ray[:, 1], new_ray[:, 2]))

    for i in range(0, ray_pos.shape[0], stepping):
        result = optimize.root(first_intersection, np.ones(ray_pos[i:i+stepping].shape[0])*35, args=(ray_pos[i:i+stepping], ray_direc[i:i+stepping]))
        try:
            distance = np.vstack([distance, np.abs(np.atleast_2d(result.x).T)])
        except (ValueError, UnboundLocalError):
            distance = np.abs(np.atleast_2d(result.x).T)

    ray_pos = ray_pos + distance*ray_direc
    norm_lens1 = normal_vector(ray_pos, lens=1)
    ray_direc = snell_3d(r_lens, r_gap, ray_direc, norm_lens1)
    return ray_pos, ray_direc

def third_prop(ray_pos, ray_direc):
    def second_intersection(x, ray_pos, ray_direc):
        new_ray = ray_pos + np.abs(np.atleast_2d(x).T)*ray_direc
        return new_ray[:, 0] - lens2((new_ray[:, 1], new_ray[:, 2]))

    for i in range(0, ray_pos.shape[0], stepping):
        result = optimize.root(second_intersection, np.ones(ray_pos[i:i+stepping].shape[0])*5, args=(ray_pos[i:i+stepping], ray_direc[i:i+stepping]))
        try:
            distance = np.vstack([distance, np.abs(np.atleast_2d(result.x).T)])
        except (ValueError, UnboundLocalError):
            distance = np.abs(np.atleast_2d(result.x).T)

    ray_pos = ray_pos + distance*ray_direc
    norm_lens2 = normal_vector(ray_pos, lens=2)
    ray_direc = snell_3d(r_gap, r_lens, ray_direc, norm_lens2)

    return ray_pos, ray_direc

def fourth_prop(ray_pos, ray_direc):
    location_last_surface = lens_position+2*lens_thickness+gap

    def third_intersection(x, ray_pos, ray_direc):
        new_ray = ray_pos + np.abs(np.atleast_2d(x).T)*ray_direc
        return new_ray[:, 0] - location_last_surface

    for i in range(0, ray_pos.shape[0], stepping):
        result = optimize.root(third_intersection, np.ones(ray_pos[i:i+stepping].shape[0])*5, args=(ray_pos[i:i+stepping], ray_direc[i:i+stepping]))
        try:
            distance = np.vstack([distance, np.abs(np.atleast_2d(result.x).T)])
        except (ValueError, UnboundLocalError):
            distance = np.abs(np.atleast_2d(result.x).T)
    ray_pos = ray_pos + distance*ray_direc
    ray_direc = snell_3d(r_lens, r_gap, ray_direc, flat_2)
    return ray_pos, ray_direc

parser = argparse.ArgumentParser(description="COLLAPS light detection region raytracer simulator")
parser.add_argument('-wavelength', metavar='nu', type=float, nargs=1, default=[811.0], help='Supply the wavelength of the light in nm (default: 811)')
parser.add_argument('-material', metavar='mat', type=str, nargs=1, default=['b270'], help='Supply the material of the lenses (b270, corning7960 or lithosil, default: b270)', choices=['b270', 'corning7960', 'lithosil'])
parser.add_argument('-positions', metavar='pos', type = float, nargs='+', help='Supply the position offsets from which to release the rays, in mm (default: 0.0 5.0 10.0 15.0 20.0 25.0)', default=[0.0, 5.0, 10.0, 15.0, 20.0, 25.0])
parser.add_argument('-rays', metavar='N', type=int, nargs=1, help='Supply the number of rays to be released from each position (default: 10000)', default=[10000])
parser.add_argument('-laserbeamsize', type=float, nargs=1, help='Supply the diameter of the laser beam in mm (default: 5)', default=[5])
parser.add_argument('--distributed', action='store_true', help='If True, N rays are released from a random released position between the minimum and maximum supplied instead of N rays from each position (default: False)')
result = parser.parse_args()

wavelength = result.wavelength[0]
material = result.material[0]
positions = result.positions
N = result.rays[0]
distributed = result.distributed
laserbeamsize = result.laserbeamsize[0]

lens_position = 76.268702359
lens_thickness = 36
gap = 0
wavelength = wavelength/1000
stepping = 20

aperture = 92

x_pos = np.array([])
y_pos = np.array([])
z_pos = np.array([])

x_pos_back = np.array([])
y_pos_back = np.array([])
z_pos_back = np.array([])

if not distributed:
    for p in positions:
        if laserbeamsize > 0:
            pos = laserbeamsize*np.random.rand(N, 2)-laserbeamsize/2
            while np.any(((pos**2).sum(axis=1))**0.5>laserbeamsize):
                not_ok = ((pos**2).sum(axis=1))**0.5>laserbeamsize
                pos[not_ok] = laserbeamsize*np.random.rand(not_ok.sum(), 2)-laserbeamsize/2
        else:
            pos = np.zeros((N, 2))
        x_pos = np.append(x_pos, pos[:, 0])
        y_pos = np.append(y_pos, np.zeros(N)+p)
        z_pos = np.append(z_pos, pos[:, 1])
        if laserbeamsize > 0:
            pos = laserbeamsize*np.random.rand(N, 2)-laserbeamsize/2
            while np.any(((pos**2).sum(axis=1))**0.5>laserbeamsize):
                not_ok = ((pos**2).sum(axis=1))**0.5>laserbeamsize
                pos[not_ok] = laserbeamsize*np.random.rand(not_ok.sum(), 2)-laserbeamsize/2
        else:
            pos = np.zeros((N, 2))
        x_pos_back = np.append(x_pos_back, pos[:, 0])
        y_pos_back = np.append(y_pos_back, np.zeros(N)+p)
        z_pos_back = np.append(z_pos_back, pos[:, 1])
else:
    low, high = np.min(positions), np.max(positions)
    y_pos = (high-low)*np.random.rand(N)+low
    if laserbeamsize > 0:
        pos = laserbeamsize*np.random.rand(N, 2)-laserbeamsize/2
        while np.any(((pos**2).sum(axis=1))**0.5>laserbeamsize):
            not_ok = ((pos**2).sum(axis=1))**0.5>laserbeamsize
            pos[not_ok] = laserbeamsize*np.random.rand(not_ok.sum(), 2)-laserbeamsize/2
    else:
        pos = np.zeros((N, 2))
    x_pos = pos[:, 0]
    z_pos = pos[:, 1]

    low, high = np.min(positions), np.max(positions)
    y_pos_back = (high-low)*np.random.rand(N)+low
    if laserbeamsize > 0:
        pos = laserbeamsize*np.random.rand(N, 2)-laserbeamsize/2
        while np.any(((pos**2).sum(axis=1))**0.5>laserbeamsize):
            not_ok = ((pos**2).sum(axis=1))**0.5>laserbeamsize
            pos[not_ok] = laserbeamsize*np.random.rand(not_ok.sum(), 2)-laserbeamsize/2
    else:
        pos = np.zeros((N, 2))
    x_pos_back = pos[:, 0]
    z_pos_back = pos[:, 1]

direc = hit_and_miss_3d_dipole_donut(len(x_pos))
direc = direc / np.atleast_2d((direc**2).sum(axis=1)**0.5).T
x_direc, y_direc, z_direc = direc[:, 0], direc[:, 1], direc[:, 2]
ray_pos = np.vstack([x_pos, y_pos, z_pos]).T
ray_direc = np.vstack([
    x_direc,
    y_direc,
    z_direc,
    ]).T

```

```
direc_back = hit_and_miss_3d_dipole_dumbbell(len(x_pos))
direc_back = direc_back / np.atleast_2d((direc_back**2).sum(axis=1)**0.5).T
x_direc_back, y_direc_back, z_direc_back = direc_back[:, 0], direc_back[:, 1], direc_back[:, 2]
ray_pos_back = np.vstack([x_pos_back, y_pos_back, z_pos_back]).T
ray_direc_back = np.vstack([
    x_direc_back,
    y_direc_back,
    z_direc_back,
    ]).T

r_gap = 1.0
r_lens = sellmeier(wavelength, material=material)

flat_1 = np.array([-1, 0, 0])
flat_2 = np.array([-1, 0, 0])

prop_distance = lens_position

print(’Starting pi wave propagation’)
start = time.time()
ray_pos1, ray_direc1 = first_prop(ray_pos, ray_direc)
mask1 = ((ray_pos1[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos1 = ray_pos1[mask1]
ray_direc1 = ray_direc1[mask1]
print(’Lightrays at the first lens, starting second propagation’)
ray_pos2, ray_direc2 = second_prop(ray_pos1, ray_direc1)
mask2 = ((ray_pos2[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos2 = ray_pos2[mask2]
ray_direc2 = ray_direc2[mask2]
print(’Lightrays at the gap, starting third propagation’)
ray_pos3, ray_direc3 = third_prop(ray_pos2, ray_direc2)
mask3 = ((ray_pos2[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos3 = ray_pos3[mask3]
ray_direc3 = ray_direc3[mask3]
print(’Lightrays at the second lens, starting fourth propagation’)
ray_pos4, ray_direc4 = fourth_prop(ray_pos3, ray_direc3)
mask4 = ((ray_pos4[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos4 = ray_pos4[mask4]
ray_direc4 = ray_direc4[mask4]
stop = time.time()
print(’Done in {:.3g} s’.format(stop-start))

original = ray_pos.shape[0]
surviving = ray_pos4.shape[0]
print(’Light ray survival (inside aperture of lenses): {:.0f}/{:.0f}={:.2f}%’.format(surviving, original, surviving/original*100))

print(’Starting sigma wave propagation’)
start = time.time()
ray_pos1_back, ray_direc1_back = first_prop(ray_pos_back, ray_direc_back)
mask1_back = ((ray_pos1_back[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos1_back = ray_pos1_back[mask1_back]
ray_direc1_back = ray_direc1_back[mask1_back]
print(’Lightrays at the first lens, starting second propagation’)
ray_pos2_back, ray_direc2_back = second_prop(ray_pos1_back, ray_direc1_back)
mask2_back = ((ray_pos2_back[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos2_back = ray_pos2_back[mask2_back]
```

```
ray_direc2_back = ray_direc2_back[mask2_back]
print('Lightrays at the gap, starting third propagation')
ray_pos3_back, ray_direc3_back = third_prop(ray_pos2_back, ray_direc2_back)
mask3_back = ((ray_pos2_back[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos3_back = ray_pos3_back[mask3_back]
ray_direc3_back = ray_direc3_back[mask3_back]
print('Lightrays at the second lens, starting fourth propagation')
ray_pos4_back, ray_direc4_back = fourth_prop(ray_pos3_back, ray_direc3_back)
mask4_back = ((ray_pos4_back[:, 1:]**2).sum(axis=1)**0.5)<aperture/2
ray_pos4_back = ray_pos4_back[mask4_back]
ray_direc4_back = ray_direc4_back[mask4_back]
stop = time.time()
print('Done in {:.3g} s'.format(stop-start))

original = ray_pos_back.shape[0]
surviving = ray_pos4_back.shape[0]
print('Light ray survival (inside aperture of lenses): {:.0f}/{:.0f}={:.2f}%'.format(surviving, original, surviving/original*100))

np.savez_compressed(str(wavelength*1000)+'.npz', (ray_pos.shape, ray_pos4, ray_direc4, ray_pos4_back, ray_direc4_back))

print('Files saved as '+str(wavelength*1000)+'.npz')
```

B.6.2 Plotting code

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.widgets import Slider
from mpl_toolkits.mplot3d import Axes3D
from mpl_toolkits.mplot3d.art3d import Line3DCollection
import argparse
import scipy.optimize as optimize
import pathlib

parser = argparse.ArgumentParser(description="COLLAPS light detection region raytracer processor")
parser.add_argument('filename', type=str, nargs=1, help='Filename of the datafile')
parser.add_argument('-startingdistance', type=float, nargs=1, help='Distance between the second lens and the PMT without spacer  in mm (default: 47)', default=[47])
parser.add_argument('-pmtsize', type=float, nargs=1, help='PMT diameter in mm (default: 47.5)', default=[47.5])
parser.add_argument('-masksize', type=float, nargs=1, help='Diameter inside which the ray distribution is drawn (default: 47.5)', default=[47.5])
parser.add_argument('-spacerthickness', type=float, nargs=1, help='Maximal spacer thickness in mm (default: 100)', default=[100])
parser.add_argument('-slitsize', type=float, nargs=1, help='Size of the slit in mm (default: 13)', default=[13])
parser.add_argument('--interactive', action='store_true', help='If True, the interactive plots are shown before the PDF is saved (default: False)')
result = parser.parse_args()

filename = result.filename[0]
startingdistance = result.startingdistance[0]
```

```
spacerthickness = result.spacerthickness[0]
pmt_radius = result.pmtsize[0]/2
mask_radius = result.masksize[0]/2
slitsize = result.slitsize[0]
interactive = result.interactive

r = np.load(filename)
wavelength = float('.'.join(filename.split('.')[:-1]))
original_shape, pos, direc, pos_back, direc_back = r['arr_0']
original = original_shape[0]

x_start = pos[0, 0]

def propagate(ray_pos, ray_direc, distance):
    location = x_start+startingdistance+distance
    distance = np.atleast_2d(location - ray_pos[:, 0]).T
    return ray_pos + distance*ray_direc/np.atleast_2d(ray_direc[:, 0]).T

def SB(val):
    new_pos = propagate(pos, direc, val)
    new_pos_back = propagate(pos_back, direc_back, val)
    pmt_mask = np.logical_and.reduce([((new_pos[:, 1:]**2).sum(axis=1)**0.5)<pmt_radius, np.abs(new_pos[:, 2])<slitsize/2])
    pmt_mask_back = np.logical_and.reduce([((new_pos_back[:, 1:]**2).sum(axis=1)**0.5)<pmt_radius, np.abs(new_pos_back[:, 2])<slitsize/2])
    return pmt_mask.sum()/pmt_mask_back.sum()

def efficiency(val):
    new_pos = propagate(pos, direc, val)
    pmt_mask = np.logical_and.reduce([((new_pos[:, 1:]**2).sum(axis=1)**0.5)<pmt_radius, np.abs(new_pos[:, 2])<slitsize/2])
    return pmt_mask.sum()/original*100

resultx = 76.268702359-startingdistance
result = optimize.minimize_scalar(lambda x: -efficiency(x), bounds=(0, spacerthickness), method='bounded')
resultx = result.x
new_pos = propagate(pos, direc, resultx)
new_pos_back = propagate(pos_back, direc_back, resultx)

mask = ((new_pos[:, 1:]**2).sum(axis=1)**0.5)<mask_radius
mask_back = ((new_pos_back[:, 1:]**2).sum(axis=1)**0.5)<mask_radius

pmt_mask = ((new_pos[:, 1:]**2).sum(axis=1)**0.5)<pmt_radius
x_range = new_pos[mask, 1].ptp()
x_range_back = new_pos_back[mask_back, 1].ptp()

var = np.arange(0, spacerthickness+0.5, 0.5)
eff = np.array([efficiency(val) for val in var])
S = np.array([SB(val) for val in var])

distance = resultx

if interactive:
    def update(val):
        val = sdist.val
        new_pos = propagate(pos, direc, val)
        mask = ((new_pos[:, 1:]**2).sum(axis=1)**0.5)<mask_radius
        pmt_mask = ((new_pos[:, 1:]**2).sum(axis=1)**0.5)<pmt_radius

        xlims = ax.get_xlim()
        ylims = ax.get_ylim()

        ax.clear()
        x_range = new_pos[mask, 1].ptp()
        hexbin = ax.hexbin(new_pos[mask, 1], new_pos[mask, 2], gridsize=int(2*x_range), mincnt=1, bins='log')
        ax.plot(x, y,zorder=10)
        ax.plot(x_bars, y_bars,zorder=10)
        ax.plot(x_bars, -y_bars,zorder=10)
        ax.set_xlim(xlims)
        ax.set_ylim(ylims)

        new_pos_back = propagate(pos_back, direc_back, val)
        mask_back = ((new_pos_back[:, 1:]**2).sum(axis=1)**0.5)<mask_radius
        pmt_mask_back = ((new_pos_back[:, 1:]**2).sum(axis=1)**0.5)<pmt_radius

        xlims = ax2.get_xlim()
        ylims = ax2.get_ylim()

        ax2.clear()
        x_range = new_pos_back[mask_back, 1].ptp()
        hexbin = ax2.hexbin(new_pos_back[mask_back, 1], new_pos_back[mask_back, 2], gridsize=int(2*x_range), mincnt=1, bins='log')
        ax2.plot(x, y,zorder=10)
        ax2.plot(x_bars, y_bars,zorder=10)
        ax2.plot(x_bars, -y_bars,zorder=10)
        ax2.set_xlim(xlims)
        ax2.set_ylim(ylims)

        e = efficiency(val)
        s = SB(val)

        ax.set_title('{:.2f} % total efficiency'.format(e))
        ax2.set_title('{:.2f} $\pi$ wave/$\sigma$ wave'.format(s))

        eff_vert.set_data([val, val], [0, e])
        sb_vert.set_data([val, val], [0, s])

        fig.canvas.draw_idle()
        global distance
        distance = val

    fig = plt.figure()
    ax = fig.add_axes([0, 0.15, 0.5, 0.5])
    ax2 = fig.add_axes([0.5, 0.15, 0.5, 0.5], sharex=ax, sharey=ax)
    ax.set_aspect(1)
    ax2.set_aspect(1)
    sax = fig.add_axes([0.2, 0.05, 0.6, 0.05])
    ax3 = fig.add_axes([0.065, 0.75, 1-2*0.065, 0.2])
    tax3 = ax3.twinx()
    line, = ax3.plot(var, eff)
    ax3.set_ylabel('Signal efficiency [%]', color=line.get_color())
    ax3.tick_params(axis='y', labelcolor=line.get_color())
    color = 'tab:red'
    tax3.plot(var, S, color=color)
    tax3.set_ylabel('$\pi$/$\sigma$ [-]', color=color)
    tax3.tick_params(axis='y', labelcolor=color)
    ax3.set_xlabel('Spacer thickness [mm]')

    eff_vert, = ax3.plot([resultx, resultx], [0, efficiency(resultx)], linestyle='solid', color=line.get_color(), lw=4.0)
    sb_vert, = tax3.plot([resultx, resultx], [0, SB(resultx)], linestyle='dashed', color=color, lw=2.0)

    hexbin = ax.hexbin(new_pos[mask, 1], new_pos[mask, 2], gridsize=int(2*x_range), mincnt=1, bins='log')
    hexbin2 = ax2.hexbin(new_pos_back[mask_back, 1], new_pos_back[mask_back, 2], gridsize=int(2*x_range_back), mincnt=1, bins='log')

    ax.set_title('{:.2f} % total efficiency'.format(efficiency(resultx)))
    ax2.set_title('{:.2f} $\pi$/$\sigma$'.format(SB(resultx)))
    sdist = Slider(sax, 'Spacer thickness', 0, spacerthickness, valinit=resultx)
    sdist.on_changed(update)

    theta = np.linspace(0, 2*np.pi, 100)
    r = pmt_radius
    x = np.cos(theta)*r
    y = np.sin(theta)*r
    y_bars = np.array([slitsize/2, slitsize/2])
    x_bars = np.cos(np.arcsin(slitsize/2/r))*r
    x_bars = np.array([-x_bars, x_bars])
    ax.set_xlim(-25, 25)
    ax.set_ylim(-25, 25)
    ax.plot(x, y,zorder=10)
    ax.plot(x_bars, y_bars,zorder=10)
    ax.plot(x_bars, -y_bars,zorder=10)

    ax2.set_xlim(-25, 25)
    ax2.set_ylim(-25, 25)
    ax2.plot(x, y,zorder=10)
    ax2.plot(x_bars, y_bars,zorder=10)
    ax2.plot(x_bars, -y_bars,zorder=10)

    plt.show()

new_pos = propagate(pos, direc, distance)
new_pos_back = propagate(pos_back, direc_back, distance)

mask = ((new_pos[:, 1:]**2).sum(axis=1)**0.5)<mask_radius
mask_back = ((new_pos_back[:, 1:]**2).sum(axis=1)**0.5)<mask_radius

x_range = new_pos[mask, 1].ptp()
x_range_back = new_pos_back[mask_back, 1].ptp()

fig = plt.figure()
fig.set_dpi(200)
fig.set_size_inches(120*0.0393700787, 120*0.0393700787)
ax = fig.add_axes([0, 0, 0.5, 0.5])
ax2 = fig.add_axes([0.5, 0, 0.5, 0.5], sharex=ax, sharey=ax)
ax.set_ylabel('z [mm]')
ax.set_xlabel('y [mm]')
```

```
ax2.tick_params(labelleft=False)
ax.set_aspect(1)
ax2.set_aspect(1)
ax3 = fig.add_axes([0, 0.75, 1, 0.2])
tax3 = ax3.twinx()

line, = ax3.plot(var, eff)
ax3.set_ylabel('Signal efficiency [%]', color=line.get_color())
ax3.tick_params(axis='y', labelcolor=line.get_color())
color = 'tab:red'
tax3.plot(var, S, color=color)
tax3.set_ylabel('$\pi$/$\sigma$ [-]', color=color)
tax3.tick_params(axis='y', labelcolor=color)
ax3.set_xlabel('Spacer thickness [mm]')
ax3.set_title('Chosen spacer thickness: {:.2f} mm'.format(distance))

eff_vert, = ax3.plot([distance, distance], [0, efficiency(distance)], linestyle='solid', color=line.get_color(), lw=4.0)
sb_vert, = tax3.plot([distance, distance], [0, SB(distance)], linestyle='dashed', color=color, lw=2.0)

hexbin = ax.hexbin(new_pos[mask, 1], new_pos[mask, 2], gridsize=int(2*x_range), mincnt=1, bins='log')
hexbin2 = ax2.hexbin(new_pos_back[mask_back, 1], new_pos_back[mask_back, 2], gridsize=int(2*x_range_back), mincnt=1, bins='log')

ax.set_title('{:.2f} % total efficiency\n$\pi$ distribution'.format(efficiency(distance)))
ax2.set_title('{:.2f} $\pi$/$\sigma$\n$\sigma$ distribution'.format(SB(distance)))

theta = np.linspace(0, 2*np.pi, 100)
r = pmt_radius
x = np.cos(theta)*r
y = np.sin(theta)*r
y_bars = np.array([slitsize/2, slitsize/2])
x_bars = np.cos(np.arcsin(slitsize/2/r))*r
x_bars = np.array([-x_bars, x_bars])
ax.set_xlim(-25, 25)
ax.set_ylim(-25, 25)
ax.plot(x, y,zorder=10, color='k')
ax.plot(x_bars, y_bars,zorder=10, color='k')
ax.plot(x_bars, -y_bars,zorder=10, color='k')

ax2.set_xlim(-25, 25)
ax2.set_ylim(-25, 25)
ax2.plot(x, y,zorder=10, color='k')
ax2.plot(x_bars, y_bars,zorder=10, color='k')
ax2.plot(x_bars, -y_bars,zorder=10, color='k')

file_path = pathlib.Path(filename)
filename = '{:.0f}_{:.0f}.pdf'.format(wavelength, slitsize)
fig.savefig(filename, bbox_inches='tight')
```

Bibliography

[1] E. Kugler. “The ISOLDE facility”. In: *Hyperfine Interactions* 129.1 (Dec. 2000), pp. 23–42. DOI: 10.1023/A:1012603025802 (p. 3).

[2] P. A. Butler, J. Cederkall, and P. Reiter. “Nuclear-structure studies of exotic nuclei with MINIBALL”. In: *Journal of Physics G: Nuclear and Particle Physics* 44.4 (2017). DOI: 10.1088/1361-6471/aa5c4e (p. 3).

[3] R. Neugart et al. “Collinear laser spectroscopy at ISOLDE”. In: *Journal of Physics G: Nuclear Physics* 44.6 (2017). DOI: 10.1088/1361-6471/aa6642 (pp. 3, 12).

[4] A. J. St Murphy. “Physics with post accelerated beams: Nuclear astrophysics”. In: *Journal of Physics G: Nuclear and Particle Physics* 44.5 (2017). DOI: 10.1088/1361-6471/aa6411 (p. 3).

[5] N. Severijns and B. Blank. “Weak interaction physics at ISOLDE”. In: *Journal of Physics G: Nuclear and Particle Physics* 44.8 (2017), p. 24. DOI: 10.1088/1361-6471/aa71e3 (p. 3).

[6] K. Johnston et al. “The solid state physics programme at ISOLDE: Recent developments and perspectives”. In: *Journal of Physics G: Nuclear and Particle Physics* 44.10 (2017). DOI: 10.1088/1361-6471/aa81ac (p. 3).

[7] V. Fedosseev et al. “Ion beam production and study of radioactive isotopes with the laser ion source at ISOLDE”. In: *Journal of Physics G: Nuclear and Particle Physics* 44.8 (2017). DOI: 10.1088/1361-6471/aa78e0 (p. 3).

[8] J. D. Jackson, S. B. Treiman, and H. W. Wyld. “Possible tests of time reversal invariance in beta decay”. In: *Physical Review* 106.3 (1957), pp. 517–521. DOI: 10.1103/PhysRev.106.517 (p. 4).

[9] D. T. Yordanov et al. “Spin and Magnetic Moment of ^{33}Mg: Evidence for a Negative-Parity Intruder Ground State”. In: *Phys. Rev. Lett.* 99 (21 Nov. 2007), p. 212501. DOI: `10.1103/PhysRevLett.99.212501` (pp. 4–5, 12).

[10] T. Ohtsubo et al. “The on-line low temperature nuclear orientation facility NICOLE”. In: *Journal of Physics G: Nuclear and Particle Physics* 44.4 (2017). DOI: `10.1088/1361-6471/aa5f22` (pp. 4, 15).

[11] M. Lindroos et al. “Measurement of the magnetic moment of ^{23}Mg using the tilted foil polarization technique”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 126.1 (1997), pp. 423–426. DOI: `10.1016/S0168-583X(96)01062-2` (p. 5).

[12] M. Keim et al. “Measurement of the electric quadrupole moments of $^{26-29}$Na”. In: *The European Physical Journal A* 8.1 (July 2000), pp. 31–40. DOI: `10.1007/s100500070117` (p. 5).

[13] M. Kowalska et al. “Nuclear ground-state spins and magnetic moments of ^{27}Mg, ^{29}Mg, and ^{31}Mg”. In: *Phys. Rev. C* 77 (2008), p. 034307. DOI: `10.1103/PhysRevC.77.034307` (pp. 5, 12).

[14] H. Törnqvist et al. “Tilted foils polarization at REX-ISOLDE”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 317 (2013), pp. 685–688. DOI: `10.1016/j.nimb.2013.07.058` (p. 5).

[15] M. Madurga. *Commissioning of a beta-decay spectroscopy station at VITO*. Tech. rep. CERN-INTC-2017-089. Geneva: CERN, Oct. 2017 (p. 5).

[16] N. Severijns et al. “Fundamental weak interaction studies using polarised nuclei and ion traps”. In: *Hyperfine Interactions* 129.1 (Dec. 2000), pp. 223–236. DOI: `10.1023/A:1012665917625` (p. 5).

[17] L. Hemmingsen et al. *β-NMR as a novel technique for biological applications*. Tech. rep. CERN-INTC-2010-015. Geneva: CERN, Jan. 2010 (p. 5).

[18] M. Stachura et al. *β-NMR of copper isotopes in ionic liquids*. Tech. rep. CERN-INTC-2013-034. Geneva: CERN, Sept. 2013 (p. 5).

[19] M. Deicher et al. *VITO Versatile Ion-polarized Techniques On-line at ISOLDE (former ASPIC UHV beamline)*. Tech. rep. CERN-INTC-2013-013. Geneva: CERN, May 2013 (p. 5).

[20] R. F. Garcia Ruiz et al. “Perspectives for the VITO beam line at ISOLDE, CERN”. In: *EPJ Web Conf.* 93 (2015), p. 7004. DOI: `10.1051/epjconf/20159307004` (p. 5).

[21] Ph. Velten et al. *Measurement of the β-asymmetry parameter in ^{35}Ar decay with a laser polarized beam.* Tech. rep. CERN-INTC-2014-062. Geneva: CERN, Oct. 2014 (pp. 5–6, 11, 37, 125).

[22] N. Severijns, M. Beck, and O. Naviliat-Cuncic. “Tests of the standard electroweak model in nuclear beta decay”. In: *Reviews of Modern Physics* 78.September (2006), pp. 991–1040. DOI: `10.1103/RevModPhys.78.991` (pp. 5, 8).

[23] M. Tanabashi et al. “Review of Particle Physics”. In: *Phys. Rev. D* 98 (3 2018), p. 030001. DOI: `10.1103/PhysRevD.98.030001` (p. 5).

[24] J. D. Garnett et al. “β-decay asymmetry parameter for ^{35}Ar: An anomaly resolved”. In: *Phys. Rev. Lett.* 60 (1988), pp. 499–502. DOI: `10.1103/PhysRevLett.60.499` (pp. 6, 13).

[25] A. Converse et al. “Measurement of the asymmetry parameter for 35Ar β-decay as a test of the CVC hypothesis”. In: *Physics Letters B* 304.1 (1993), pp. 60–64. DOI: `https://doi.org/10.1016/0370-2693(93)91400-H` (pp. 6, 13).

[26] N. Severijns and O. Naviliat-Cuncic. “Structure and symmetries of the weak interaction in nuclear beta decay”. In: *Physica Scripta* T152 (2013). DOI: `10.1088/0031-8949/2013/T152/014018` (pp. 6, 8–10).

[27] K. S. Krane. *Introductory Nuclear Physics.* John Wiley & Sons, 1988, p. 844. ISBN: 9780471805533 (p. 6).

[28] M. R. Bhat. *Evaluated Nuclear Structure Data File (ENSDF).* 2011. URL: `https://www.nndc.bnl.gov/ensdf/` (visited on Sept. 18, 2018) (p. 7).

[29] O. Naviliat-Cuncic and N. Severijns. “Test of the conserved vector current hypothesis in T=1/2 mirror transitions and new determination of |Vud|”. In: *Physical Review Letters* 102.14 (2009), pp. 1–4. DOI: `10.1103/PhysRevLett.102.142302` (p. 9).

[30] Ph. Velten. *Preliminary study of the β-γ detection setup for the measurement of β-asymmetry parameter in ^{35}Ar decay.* Tech. rep. IKS, 2015, p. 19 (pp. 10, 126).

[31] G. Neyens et al. “Measurement of the Spin and Magnetic Moment of ^{31}Mg: Evidence for a Strongly Deformed Intruder Ground State”. In: *Phys. Rev. Lett.* 94 (Jan. 2005), p. 022501. DOI: 10.1103/PhysRevLett.94.022501 (p. 12).

[32] M. Keim. “Messung der Kernquadrupolmomente neutronenreicher Natriumisotope”. PhD thesis. Johannes Gutenberg-Universität, 1996, p. 117 (pp. 12, 59).

[33] G. Neyens. “Ontwikkeling van een doelgerichte LEMS opstelling en bepalen van het quadrupoolmoment van isomere toestanden in radium isotopen”. PhD thesis. Katholieke Universiteit Leuven, 1993 (p. 13).

[34] J. Korringa. “Nuclear magnetic relaxation and resonnance line shift in metals”. In: *Physica* 7 (1950), pp. 601–610 (p. 13).

[35] K. Matsuta et al. “Precise magnetic moment of short-lived beta emitter ^{35}Ar”. In: *Nuclear Physics A* 701 (2002), pp. 383–386 (pp. 13, 34, 59, 87, 102).

[36] K. Turzó et al. “Spin polarization of ^{34}Al fragments produced by nucleon pickup at intermediate energies”. In: *Phys. Rev. C* 73 (4 Apr. 2006), p. 044313. DOI: 10.1103/PhysRevC.73.044313 (p. 15).

[37] S. Cottenier and M. Rots. *Hyperfine Interactions and their Applications in Nuclear Condensed Matter Physics: a microscopic introduction.* 2005 (p. 16).

[38] V. Jaccarino et al. “Hyperfine Structure of I^{127}. Nuclear Magnetic Octupole Moment”. In: *Phys. Rev.* 94 (6 June 1954), pp. 1798–1799. DOI: 10.1103/PhysRev.94.1798 (p. 16).

[39] C. Schwartz. “Theory of Hyperfine Structure”. In: *Phys. Rev.* 97 (2 Jan. 1955), pp. 380–395. DOI: 10.1103/PhysRev.97.380 (p. 17).

[40] A. Khawam and D. R. Flanagan. “Solid-state kinetic models: Basics and mathematical fundamentals”. In: *Journal of Physical Chemistry B* 110.35 (2006), pp. 17315–17328. DOI: 10.1021/jp062746a (p. 19).

[41] A. Einstein. “Zur Quantentheorie der Strahlung”. In: *Mitteilungen der Physikalischen Gesellschaft Zürich* 18.17 (1916), pp. 47–62 (p. 19).

[42] C. J. Foot. *Atomic Physics.* Oxford Master Series in Physics. Oxford University Press, 2005, p. 346. ISBN: 0198506953 (pp. 19, 21).

[43] D. T. Yordanov. “From ^{27}Mg to ^{33}Mg: transition to the Island of inversion”. PhD thesis. Katholieke Universiteit Leuven, 2007, p. 154 (pp. 19, 25–26, 125).

[44] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions With Formulas, Graphs and Mathematical Tables.* Dover Publications, 1964, p. 470. ISBN: 0486612724 (p. 21).

[45] I. I. Sobelman. *Atomic Spectra and Radiative Transitions.* 2nd ed. Springer-Verlag Berlin Heidelberg, 1992, p. 356. ISBN: 3662059053 (p. 23).

[46] L. Ballentine. *Quantum Mechanics: a modern development.* World Scientific Publishing, 1998. ISBN: 981-02-4105-4 (p. 23).

[47] K. L. Heyde. *The Nuclear Shell Model.* 1st ed. Springer-Verlag Berlin Heidelberg, 1990, p. 377. ISBN: 978-3-642-97203-4. DOI: `10.1007/978-3-642-97203-4` (p. 23).

[48] A. C. Hindmarsh. *ODEPACK, A Systematized Collection of ODE Solvers.* 1983 (p. 25).

[49] E. Jones, T. Oliphant, P. Peterson, et al. *SciPy: Open Source Scientific Tools for Python.* URL: `http://www.scipy.org/` (p. 25).

[50] A. Kramida et al. *NIST Atomic Spectra Database.* 2018. URL: `https://physics.nist.gov/PhysRefData/ASD/lines_form.html` (visited on May 29, 2018) (pp. 33–34, 38).

[51] J. Weite et al. "Towards the realization of atom trap trace analysis for ^{39}Ar". In: *New Journal of Physics* 12.July (2010). DOI: `10.1088/1367-2630/12/6/065031` (pp. 34–35).

[52] A. Klein et al. "Moments and mean square charge radii of short-lived argon isotopes". In: *Nuclear Physics A* 607 (1996), pp. 1–22. DOI: `10.1016/0375-9474(96)00192-3` (pp. 34, 37, 80, 85, 121).

[53] T. Mertzimekis, K. Stamou, and A. Psaltis. "An online database of nuclear electromagnetic moments". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 807 (2016), pp. 56–60. DOI: `10.1016/j.nima.2015.10.096` (p. 35).

[54] A. Vernon. Private communication. 2018 (p. 37).

[55] D. Rapp and W. E. Francis. "Charge Exchange between Gaseous Ions and Atoms". In: *The Journal of Chemical Physics* 37.11 (1962), pp. 2631–2645. DOI: `10.1063/1.1733066` (p. 37).

[56] W. Gins et al. "A new beamline for laser spin-polarization at ISOLDE". In: (2018). arXiv: `1809.04385 [physics.ins-det]` (pp. 43, 125).

[57] ISOLDE Target and Ionsource Development Group (TISD). *ISOLDE Yield Database (Ar)*. 2018. URL: `https://isoyields2.web.cern.ch/YieldByElement_Basic.aspx?Z=18` (visited on June 28, 2018) (p. 63).

[58] Y. Martinez Palenzuela. Private communication. 2018 (pp. 63–64).

[59] "Off-line commissioning of the ISOLDE cooler". In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 266.19 (2008), pp. 4502–4504 (p. 65).

[60] E. Mané et al. "An ion cooler-buncher for high-sensitivity collinear laser spectroscopy at ISOLDE". In: *European Physical Journal A* 42.3 (2009), pp. 503–507. DOI: `10.1140/epja/i2009-10828-0` (pp. 65–66).

[61] Q. Composers. 2018. URL: `https://www.quantumcomposers.com/pulse-delay-generator-9520` (p. 65).

[62] W. Gins et al. "Analysis of counting data: Development of the SATLAS Python package". In: *Computer Physics Communications* 222 (2018), pp. 286–294. DOI: `https://doi.org/10.1016/j.cpc.2017.09.012` (p. 68).

[63] K. D. Kreim. "Collinear Laser Spectroscopy of Potassium". PhD thesis. Heidelberg, 2013 (p. 81).

[64] J. Papuga. "Structure of potassium isotopes studied with collinear laser spectroscopy". PhD thesis. Katholieke Universiteit Leuven, 2015, p. 184 (p. 81).

[65] M. Goldman. "Formal theory of spin–lattice relaxation". In: *Journal of Magnetic Resonance* 149.2 (Apr. 2001), pp. 160–87. DOI: `10.1006/jmre.2000.2239` (p. 87).

[66] B. H. Suits. "Nuclear Quadrupole Resonance Spectroscopy". In: *Handbook of Applied Solid State Spectroscopy*. Ed. by D. Vij. Springer, 2006. Chap. 2, pp. 65–96. ISBN: 978-0-387-32497-5 (p. 87).

[67] E. G. Wikner, W. E. Blumberg, and E. L. Hahn. "Nuclear quadrupole spin-lattice relaxation in alkali halides". In: *Physical Review* 360 (1960). DOI: `10.1103/PhysRev.118.631` (pp. 87, 105).

[68] T. Ohtsubo et al. "Polarization of ^{23}Ne, ^{24m,25}Al and ^{28}P produced through single nucleon pickup and charge-exchange reactions at 100 A MeV". In: *Hyperfine Interactions* 180.1-3 (2008), pp. 85–89. DOI: `10.1007/s10751-008-9690-0` (p. 87).

[69] C. T. J. Alkemade. “Anomalous saturation curves in laser-induced fluorescence”. In: *Spectrochimica Acta Part B: Atomic Spectroscopy* 40.10-12 (1985), pp. 1331–1368. DOI: `10.1016/0584-8547(85)80160-9` (p. 92).

[70] K. Vyvey et al. “Preservation of orientation of fusion-evaporation reaction residues recoiling into vacuum in a level mixing spectroscopy experiment”. In: *Phys. Rev. C* 62 (3 Aug. 2000), p. 034317. DOI: `10.1103/PhysRevC.62.034317` (p. 93).

[71] R. de Groote. “High resolution collinear resonance ionization spectroscopy of neutron-rich ^{76,77,78}Cu isotopes”. PhD thesis. KU Leuven, 2017, p. 206 (p. 97).

[72] N. Peterson. “Self-diffusion in pure metals”. In: *Journal of Nuclear Materials* 69-70 (1978), pp. 3–37. DOI: `10.1016/0022-3115(78)90234-9` (p. 102).

[73] A. P. Dickin. *Radiogenic Isotope Geology*. 2nd ed. Vol. 1. Cambridge University Press, 2018. ISBN: 9781316163009. DOI: `10.1017/9781316163009` (p. 102).

[74] G. Turner et al. “Short lived ^{36}Cl and its decay products ^{36}Ar and ^{36}S in the early solar system”. In: *Geochimica et Cosmochimica Acta* 123 (2013), pp. 358–367. DOI: `10.1016/j.gca.2013.06.022` (p. 102).

[75] J. Burton and G. Jura. “Self Diffusion in Solid Argon: The Activation Energy”. In: *Journal of Physics and Chemistry of Solids* 28.1 (1967), pp. 705–710 (p. 102).

[76] T. Shaw and N. Stone. “Nuclear spin-lattice relaxation attenuation coefficients for use in on-line nuclear orientation experiments”. In: *Atomic Data and Nuclear Data Tables* 42.2 (1989), pp. 339–363 (p. 105).

[77] Y. Yafet and V. Jaccarino. “Nuclear spin relaxation in transition metals; core polarization”. In: *Physical Review* 133.6A (1964). DOI: `10.1103/PhysRev.133.A1630` (p. 105).

[78] J. Lu et al. “Nuclear spin-lattice relaxation in n-type insulating and metallic GaAs single crystals”. In: *Physical Review B* 74.12 (Sept. 2006), p. 125208. DOI: `10.1103/PhysRevB.74.125208` (p. 105).

[79] D. Griffiths. *Introduction to Quantum Mechanics*. Pearson Prentice Hall, 2005. ISBN: 9780131118928 (p. 109).

[80] J. J. Sakurai and J. Napolitano. *Modern Quantum Mechanics*. Ed. by L. Kenney et al. 2e. Addison-Wesley, 2011, p. 550 (p. 109).

[81] J. Johansson, P. Nation, and F. Nori. “QuTiP 2: A Python framework for the dynamics of open quantum systems”. In: *Computer Physics Communications* 184.4 (2013), pp. 1234–1240. DOI: `https://doi.org/10.1016/j.cpc.2012.11.019` (p. 110).

[82] P. J. Mohr, D. B. Newell, and B. N. Taylor. “CODATA recommended values of the fundamental physical constants: 2014”. In: *Rev. Mod. Phys.* 88 (2016), p. 035009. DOI: `10.1103/RevModPhys.88.035009` (p. 111).

[83] M. Kowalska et al. *Development of new beams of spin-polarized radioactive nuclei using stable beams from ISOLDE.* Tech. rep. CERN-INTC-2017-076. Geneva: CERN, May 2017 (p. 116).

[84] M. Berger et al. *ESTAR, PSTAR, and ASTAR: Computer Programs for Calculating Stopping-Power and Range Tables for Electrons, Protons, and Helium Ions (version 1.2.3).* 2017. URL: `https://physics.nist.gov/PhysRefData/Star/Text/ESTAR.html` (visited on Sept. 18, 2018) (p. 117).

[85] L. Hayen et al. “High precision analytical description of the allowed β spectrum shape”. In: *Rev. Mod. Phys.* 90 (1 Mar. 2018), p. 015008. DOI: `10.1103/RevModPhys.90.015008` (p. 117).

[86] C. Levy et al. “Polarized radioactive beam at ISAC”. In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 204 (2003), pp. 689–693. DOI: `10.1016/S0168-583X(03)00485-3` (p. 121).

[87] C. Levy et al. “Feasibility study of in-beam polarization of fluorine”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 580.3 (2007), pp. 1571–1577. DOI: `10.1016/j.nima.2007.07.013` (p. 121).

[88] C. Pruss, E. Garbusi, and W. Osten. “Testing Aspheres”. In: *Opt. Photon. News* (2008), pp. 24–29. DOI: `10.1364/OPN.19.4.000024` (p. 148).

[89] B. de Greve. *Reflections and Refractions in Ray Tracing.* 2006. URL: `https://graphics.stanford.edu/courses/cs148-10-summer/docs/2006-degreve-reflection_refraction.pdf` (p. 150).

[90] I. Hertel. *Introduction to Quantum Physics.* 2002. URL: `http://staff.mbi-berlin.de/hertel/physik3/chapter8/8.7html/01.htm#top` (p. 150).

[91] L. J. Curtis. *Atomic Structure and Lifetimes.* 2002. URL: `http://astro1.panet.utoledo.edu/~ljc/chapa6.html` (p. 150).

FACULTY OF SCIENCE
DEPARTMENT OF PHYSICS AND ASTRONOMY
NUCLEAR MOMENTS
Celestijnenlaan 200D - 2418
B-3001 Leuven
wouter.gins@kuleuven.be
http://fys.kuleuven.be/iks/nm/