Fitting and Analysis of BNMR data at TRIUMF
A manual and documentation for BNMRFIT

by

Mark Xu

Honours Physics
in
Simon Fraser University
(Coop Education)

TRIUMF
December 22, 2006
© Mark Xu 2006
Abstract

This co-op work term report summarizes my work during my 8-month work term at TRIUMF, and hopefully serve as a useful technical article for the BNMR group. The focus of this report will be on BNMRFIT development and documentation. A brief manual for BNMRFIT is included in this manual, as well as a description of the files used in BNMRFIT.
Contents

Abstract ................................................................. ii

Contents ............................................................... iii

List of Tables ............................................................ v

List of Figures ........................................................... vi

Acknowledgements ......................................................... vii

I Report 1

1 BNMRFIT Development ................................................... 2
  1.1 BNMRFIT User Manual ............................................. 2
  1.1.1 Time-differential mode ....................................... 2
  1.1.2 Time-integral mode ........................................... 8
  1.2 BNMRFIT files ..................................................... 14
  1.2.1 List of all files used ......................................... 14
  1.2.2 Function of each file ......................................... 14
  1.2.3 Structure of BNMRFIT (dependency of files) ............... 14
  1.3 Previous attempts of re-writing BNMRFIT ...................... 16
  1.3.1 Borland C++ version. (Andre’s) ............................ 16
  1.3.2 Andrew Morrison’s gcc version (with his co-op report) ... 16
  1.3.3 My own attempt of breaking up current PHYSICA script
       files and wrapping it with PERL ............................ 16
  1.3.4 BNMRMINFIT .................................................. 17
  1.3.5 Hassan’s work on rewriting BNMRFIT ....................... 17
  1.3.6 Using commercial software, such as Origin ................. 17
  1.4 My opinion on future development .............................. 17

2 Demagnetization report ................................................ 19
  2.1 Summary ......................................................... 19
  2.2 What is demagnetization ........................................ 19
  2.3 Demagnetization and its effect on the local internal field ... 20
  2.4 Demagnetization and its effect on the Knight shift ......... 21
  2.5 Addressing finite thickness effects ............................ 22
Contents

2.6 Sample Calculations .............................................. 23
2.7 COMSOL simulation .............................................. 24

Bibliography .............................................................. 27
List of Tables

1.1 BNMRFIT script files ........................................ 14
1.2 BNMRFIT script descriptions .............................. 15
2.1 COMSOL simulation results .............................. 25
List of Figures

1.1 A screenshot of BNMRFIT graphics windows, after the run is loaded ................................................ 4
1.2 $\chi^2$ minimization fitting of deadtime ........................................... 5
1.3 Histogram after combining helicities .............................................. 6
1.4 Histogram after binning by a factor of 10 ................................. 7
1.5 Histogram after binning by a factor of 10, fit result is shown............... 8
1.6 Loading a 2e run, incorrect mode ............................................ 9
1.7 Loading a 2e run, correct mode selected. Note the label for Y-axis showing the mode BNMRFIT is in........... 10
1.8 2e run histogram, with helicities combined ................................... 11
1.9 2e run, after initial guess is made ........................................... 12
1.10 2e run histogram, successfully fitted ...................................... 13

2.1 Effect of Lorenzian sphere ....................................................... 22
2.2 COMSOL simulation results, magnetic field vs thickness, linear scale .................................................. 25
2.3 COMSOL simulation results, magnetic field vs thickness, log scale 26
Acknowledgements

I would like to thank Dr. Andrew MacFarlane for hiring me as a co-op student. I would also like to thank other members of BNMR group for giving me help during my work term.
Part I

Report
Chapter 1

BNMRFIT Development

1.1 BNMRFIT User Manual

So far, no written manual for BNMRFIT exists. In this section, a brief manual for BNMRFIT will be written, both for aiding new students getting familiar with BNMRFIT, and to help future BNMRFIT maintainers keep better track of changes done to BNMRFIT functionality.

1.1.1 Time-differential mode

Brief background

Time-differential mode measures asymmetry in the time domain, with X-axis being time. In most SLRs, beam is turned on for several seconds, (typically 4, but sometimes 1, 2, or 0.5) and then turned off for several seconds. The count rate is measured during both beam on and beam off period. After that, a few seconds are spent on other tasks such as helicity flipping. The beam is then turned back on, and the whole process is repeated. Such procedures are repeated until enough statistics is gathered. Most runs take about 30 minutes to 1 hour. Once the run is complete, the .msr data file contains count rate for four counters. Currently, forward and backward count rates are typically combined to obtain the asymmetry. Before the asymmetry is calculated, a deadtime correction is often performed. Once the deadtime correction is done and a single asymmetry line is obtained, the fitting begins. Currently, the user has the option of using either built-in physica fitting or minuit fitting. If fitting is successful, the user has the option of writing the result into a text file, or just leave the result on the display screen. Another option is to print the graphics screen.

starting and loading runs

To start BNMRFIT, go into the directory of bnmrfit, which is currently located at folder /bnmrfit in the bnmr@isdaq01. At the time of writing this manual, the version is 1.4.0beta. Type **physica** at the command prompt to start the PHYSICA script environment. Then type **@bnmrfit** to begin using BNMRFIT. The PHYSICA screen should clear up, and show the following:

- setting display colors back to default
- turn off Normalization by neutral beam flag
- turn off dead time correction

====================================================================
Welcome to Physica -- BNMRFit =
Version 1.4.0 beta =
type "minuit" to turn on/off minuit fitting =
====================================================================

This screen confirms that bnmrfit.pcm is successfully loaded. BNMRFIT is command-based, and you can type help to display a list of commands available. Here I will go through the process of fitting a normal run. First, we select the year, as run numbers reset at the beginning of the year. Type y, and enter the year, or just press ENTER to agree to the default year.

>> y

Take data from the year [2006]>> 2006

After that, enter the run number. We will use run 40132 as an example, and show the typical process of an SLR run. First, type 40132 at the command prompt, and BNMRFIT will attempt to find and load the run. If the data is available and not corrupt, BNMRFIT displays asymmetry on the graphics window, prints out run information on the command terminal, and return to its menu prompt: >> 40132

/isdaq/data11/bnmr/dlog/2006/

40132: Pt/MgO, SLR , Bias=20 kV, 4.1 T, T=20 K -- 1650 Rows, 14 Columns

This run took 22.8833333 minutes to acquire

x minimum is 90
x maximum is 15990

The following is a screen shot of the graphics window:

Deadtime correction and combining helicities

Due to the limits of detectors, a deadtime correction should be performed on the data before further fitting can be done. For more information on deadtime correction, consult the technical article by Jahangir Valiani on the BNMR website, under “Technical Information”. To find deadtime for the current run, type fd at the command prompt. BNMRFIT will then ask several questions, including number of iterations, step size in seconds, minimum deadtime in seconds, whether you want to a graphical display of the deadtime calculation, and length of each sweep. Turning on the deadtime calculation greatly slows down the deadtime finding process, therefore answer to that question is usually no. The input/output usually goes like this:

>> fd

entering dead time correction routine....press return to continue

Number of iterations [50]>> 50
Step size in seconds [1.E-09]>>
minimum deadtime in seconds [0]>> 4.e-8

/isdaq/data11/bnmr/dlog/2006/

40132: Pt/MgO, SLR , Bias=20 kV, 4.1 T, T=20 K -- 1650 Rows, 14 Columns

Display MidPoint-MidPointAverage? (y/(n)) n
enter the length of each sweep,(s) >> (20.5)
Chapter 1. BNMRFIT Development

Figure 1.1: A screenshot of BNMRFIT graphics windows, after the run is loaded.

**total time of each bin is 6.69756098E-01 seconds**
The deadtime is 8.E-08 seconds
Chi2 for this dead time was calculated to be:2.67850002
and the graphics windows looks like Figure 1.2. Note that if you don’t see a
minimum in the graph, your range of deadtime is too small, and you should
run deadtime correction again. The “length of each sweep” refers to the time
it takes for the BNMR DAQ (data acquisition system) to complete one sweep,
and is obtained by adding the various segments of a sweep. Calculating the
sweep length requires knowledge about data acquisition set up. For example, if
we go to BNMR Elog, and look up the run information for 40132. We add up
prebeam dwell times, beam on dwell times, and beam off dwell times. These
two variables are recorded as time bins in Elog, with the length of each time
bin usually being 10ms. Therefore, multiply the number shown by 10ms to ob-
tain the time spent on it. And then add data service time and helicity flip sleep
time. Since the DAQ setup varies from experiment to experiment, consult more
experienced group members if you are not familiar with the DAQ setup.
Once you find the deadtime, it’s time to apply the deadtime correction, by typ-
ing `dt` at the command prompt. The updated deadtime value is already the
default, therefore you can just press ENTER.

```plaintext
>> dt
Dead time correction turned ON
Dead time is [8.E-08]
```

```
/isdaq/data11/bnmr/dlog/2006/
```
After the run is corrected for detector deadtime, we usually select the range of data we want to fit over, and combine the helicities. Type s to beginning selection.

```
>> s
```

Data combination:
- ind = all scans treated independently
- com = average all scans of like helicity, yields 2 scans
- all = combine all data (baseline difference lost)

ind/com/(all) >> all

Select a subset for fitting [(yes)/key/no] >> key

Type the range boundaries (2 values of x) [-500 15990] >> 200 12000

```
x minimum is 190
x maximum is 11990
```

The graphics window now displays the combined histogram over the specified range, as shown in Figure 1.3. We use range 200 to 12000 due to the fact that the very beginning and very end of the sweep tend to have low count rates, and may not have valid data.
After we obtain the histogram to be fitted, we can bin the data to get a clearer picture of the run, and reduce the amount of time required for fitting. Binning reduces data numbers by averaging a set number of data points. Type `bin` at the command prompt to bin the data.

```plaintext
>> bin
```

Currently there are 1650 bins with a bin size of 10000 Hz
Enter integer binning factor (0 no change) [10]>> 10
x minimum is 240
x maximum is 12040

After binning, the histogram looks a lot clearer, and the 1650 bins are reduced to 165 bins. Note that information is lost in the binning process. If there is fast time dependence in the data, binning may prevent you from seeing that dependence. In other words, anything happening faster than the bin size will be very difficult to see. For example, if the original bin size is 10 ms, which is the default dwell time, and binning factor is 10, then the new bin size is $10\text{ms} \times 10 = 100\text{ms}$. Therefore anything happening in less than 100 ms will be “binned over” in the binned data. If you expect to see things that happen quickly, using unbinned data will be a better idea. Figure 1.4 shows the data binned with a factor of 10, which makes it easier to see.

After binning, we can start fitting the run. MINUIT fitting was recently added to BNMRFIT, and provides a more robust fitting algorithm. Type `minuit` at
command prompt to activate minuit fitting. The default is not using MINUIT.
Once activated, BNMRFIT will always use MINUIT for fitting until turned off.
The following is an example of fitting input/output:

```
>> fit
Number of lines/components (1-8) [1]? >> 1
 1- Exponential
 2- Stretch Exponential
 3- Exponential with Beam On/Off
 4- Constant
Fit to [1] >> 3
Number of parameters to fix ([0])? 0
Method of initial guess [mouse/keyboard] >> (keyboard) k
Input fit parameters:
now acquiring the initial values of component1
Amplitude #1 [1.42999839E-01] >>
Relaxation rate #1 [3.12264309E-05] >>
function fitted is 0+A1*(1/1209.3)/(Lam1+(1/1209.3))*((1+exp((ix-bondt*dt)))*(1-exp(-Lam1+(1/1209.3)*ix)))/(1-exp(-lamtau*ix))+(1-1/(1+exp((ix-bondt*dt))))*(1-exp(-((1/1209.3)*bondt*dt))/(1-exp(-((1/1209.3)*bondt*dt)))) *exp(-Lam1*(ix-bondt*dt))
Is it a good guess (yes no)? >> yes
Iter& A1 LAM1 Residual Step Size
 1 0.143 3.123E-05 1.756E+05 1.00
```
number of degrees of freedom = 117
total CHISQ = 636.525375
sqrt( CHISQ/point ) = 2.3127803
CHISQ/(degrees of freedom) = 5.44038782
confidence level = 0%

PARAMETER VALUE E1 E2
A1 0.17091 1.06594E-04 2.48627E-04
LAM1 3.77732E-05 4.40587E-07 1.02765E-06
average asymmetry is 1.63431004E-01 +- 5.26628659E-04
Write fit parameters to file y/(n)? >> n
If a fit is successful, theoretical result will be printed on the graphic window, with the red line representing the theoretical values, as shown in Figure 1.5.

Figure 1.5: Histogram after binning by a factor of 10, fit result is shown.

1.1.2 Time-integral mode
Time integral mode scans through a particular frequency range, usually producing one or more resonance peaks. In this section I will focus on 2e mode, which is what I am most familiar with.
loading the run

Same as SLR mode mentioned above, type y to set the year, and enter the run number to load the run.

```plaintext
>> y
Take data from the year [2006]\[\textgreater\] 2006
>> 40232
/isdaq/data11/bnmr/dlog/2006/
40232: 500A Pd/STO, 2e; Bias=27 kV, 4.1 T, T = 100 K -- 809 Rows, 14 Columns
This run took 34.4333333 minutes to acquire
x minimum is 0
x maximum is 25856
>>
```

Since the default mode of BNMRFIT is for SLR runs, the graphics windows is not displaying the data properly. Note that the caption for the Y-axis shows the current mode BNMRFIT is in. To correct the mode, type m, and select 8.

```plaintext
>> m
Choose mode for asymmetry calculation
1- (F-B)/(F+B)
2- F/(NBF+NBB)
3- B/(NBF+NBB)
4- Single Counter
```

Figure 1.6: Loading a 2e run, incorrect mode
5- \((\text{NBF-NBB})/(\text{NBF+NBB})\)
6- General Asymmetry from 4 counters
for the following modes, forward counter normalization
selection is based on previous mode selection
fmode is 0
8- 2e mode Differences
9- 2e mode Slopes
10- 2e mode \((\text{A-C})/(\text{A+C})\)
11- 2e mode \((\text{A-C})/(2\text{B})\)
12- 2e mode \((\text{A-B})/(\text{A+B})+(\text{B-C})/(\text{B+C})\)
Select mode [1] >> 8
/isdaq/data11/bnmr/dlog/2006/
40232: 500A Pd/STO, 2e; Bias=27 kV, 4.1 T, T = 100 K -- 809 Rows, 14 Columns
This run took 34.4333333 minutes to acquire
x minimum is 25800
x maximum is 25880
>>
After selecting the mode, BNMRFIT re-draws the graph, with frequency as the x-axis. As shown in Figure 1.7, the label for Y-axis shows the current mode of BNMRFIT, and its method of calculating asymmetry. Once the mode is set cor-

Figure 1.7: Loading a 2e run, correct mode selected. Note the label for Y-axis showing the mode BNMRFIT is in.
rectly, we can combine the two helicities and choose the range of frequency we want to fit with. Unlike SLR runs, the count rate through the entire frequency range tends to be fairly consistent, allowing the user to perform fitting using the entire spectrum. On the other hand, if there are certain artifacts present on the spectrum that you want to exclude, you can choose to select a subset for fitting using either keyboard or mouse. Similar to time-differential mode, you choose from the options provided by the command prompt:

`>> sel`

Data combination:

- `ind` = all scans treated independently
- `com` = average all scans of like helicity, yields 2 scans
- `all` = combine all data (baseline difference lost)

Select from `ind/com/(all) >> all`

Select a subset for fitting (yes/key/no) >> no

x minimum is 25800
x maximum is 25880

After this step, the graphics window returns the combined asymmetry histogram, as shown in Figure 1.8.

After the helicities are combined, we can start the fitting. The process is similar to SLR mode fitting, with different functions. For 2e modes, the method of initial guessing is mouse clicking.

`>> fit`

Number of lines/components (1-8) [1]? >>

![Figure 1.8: 2e run histogram, with helicities combined](image-url)
Chapter 1. BNMRFIT Development

1- Lorentzian
2- Gaussian
3- Erf
4- Lorentzian/Gaussian Combo

Fit to [1] >> 1
Number of parameters to fix ([0])? 0
Point at the baseline or background
Point on the peak 1
FWHM points (2 points)
Is it a good guess [(yes)/no]? >>

After the user clicks on the points asked by BNMRFIT, the graphics window show an initial guess. If the guess looks reasonable, such as that in Figure 1.9, type yes to proceed. BNMRFIT will then start the fitting process.

![Figure 1.9: 2e run, after initial guess is made.](image)

<table>
<thead>
<tr>
<th>Iter</th>
<th>BCK</th>
<th>AMP1</th>
<th>RES1</th>
<th>WID1</th>
<th>Residual</th>
<th>Step Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.967E-03</td>
<td>-1.178E-02</td>
<td>2.583E+04</td>
<td>12.0</td>
<td>279. 1.00</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1.683E-03</td>
<td>-9.360E-03</td>
<td>2.583E+04</td>
<td>9.67</td>
<td>164. 1.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.773E-03</td>
<td>-9.870E-03</td>
<td>2.583E+04</td>
<td>9.19</td>
<td>162. 1.00</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1.773E-03</td>
<td>-9.870E-03</td>
<td>2.583E+04</td>
<td>9.19</td>
<td>162. 1.00</td>
<td></td>
</tr>
</tbody>
</table>

...Iterations omitted here

number of degrees of freedom = 157
total CHISQ = 162.494797
sqrt( CHISQ/point ) = 1.0046315
CHISQ/(degrees of freedom) = 1.03499871
confidence level = 36.5361976
PARAMETER VALUE E1 E2
BCK 1.77308E-03 1.51226E-04 1.53850E-04
AMP1 -9.87013E-03 4.92923E-04 5.01475E-04
RES1 25834. 0.22888 0.23285
WID1 9.1920 0.78471 0.79832
L: AMP1=-0.00987+- 0.00050 RES1= 25833.508+- 0.233 WID1= 9.192+- 0.798
Write fit parameters to file y/(n)? >>
Once the fit is complete, graphics window draws the fitted curve on top of experimental data, with fitted parameters at the top of the window, as shown in Figure 1.10.

Figure 1.10: 2e run histogram, successfully fitted

After a successful fit, you can choose to save the fit result to a text file. During a series of runs such as temperature scan, saving fit results to the same text file allows you to open it with a spread sheet program later and do further analysis. The terminal input/output is shown below:
Write fit parameters to file y/(n)? >> y
What is the value and error in the independent variable? (100 10) 100 1
What file name should I use? results.out PDST0.OUT
bline variable do not exist, going to add it
Write the header line? (y/(n)) Y
Parameters appended to file :./outdata/PDSTO.OUT1
>>
The output data file is saved under the outdata directory in BNMRFIT’s folder. The independent variable is the variable you want to record for this run. For run 40232, it’s the temperature of 100K. To find the error in the variable, you should usually look in the log book.

1.2 BNMRFIT files

This section will serve as documentation for future maintainers of BNMRFIT. It may also provide some hint for people who want to make minor modifications to BNMRFIT code to suit their needs.

1.2.1 List of all files used

The files are:

<table>
<thead>
<tr>
<th>bnmr.asycalc.pcm</th>
<th>bnmr_bin.pcm</th>
<th>bnmr_fit.pcm</th>
<th>bnmr_fitstat.pcm</th>
</tr>
</thead>
<tbody>
<tr>
<td>bnmr.glbinit.pcm</td>
<td>bnmr_par.pcm</td>
<td>bnmr_inputpar.pcm</td>
<td>bnmr_guess.pcm</td>
</tr>
<tr>
<td>bnmr.pores.pcm</td>
<td>bnmr_asy.pcm</td>
<td>bnmr_fdt.pcm</td>
<td>bnmr_graph.pcm</td>
</tr>
<tr>
<td>bnmr_savefp.pcm</td>
<td>bnmr_indvar.pcm</td>
<td>bnmr_fitpar.pcm</td>
<td>bnmr_selector.pcm</td>
</tr>
<tr>
<td>bnmr_fixvary.pcm</td>
<td>bnmr_init.pcm</td>
<td>bnmr_average.pcm</td>
<td>bnmr_fitpar.pcm</td>
</tr>
<tr>
<td>bnmr_genfun.pcm</td>
<td>bnmr_yab.pcm</td>
<td>bnmr_fit.pcm</td>
<td>bnmr_getrun.pcm</td>
</tr>
</tbody>
</table>

Table 1.1: BNMRFIT script files

1.2.2 Function of each file

Table 1.2.2 provides a brief description for the .pcm files used in the current version of BNMRFIT. Usually in the BNMRFIT folder, you will find a large number of other .pcm files. They are usually modified version of BNMRFIT scripts that group members make for special purposes.

1.2.3 Structure of BNMRFIT (dependency of files)

In this section, I’ll try to group the .pcm files into several categories, so that people trying to read the code have a better sense of what connects to what.

- bnmrfit.pcm: The main file.
<table>
<thead>
<tr>
<th>Script File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bnmr_asycalc.pcm</td>
<td>Separate asymmetry and error for each scan from AsyFB.</td>
</tr>
<tr>
<td>bnmr_bin.pcm</td>
<td>Performs the binning function.</td>
</tr>
<tr>
<td>bnmrfit.pcm</td>
<td>The main script, which reads in input from prompt, and calls the other script files.</td>
</tr>
<tr>
<td>bnmr_fitstat.pcm</td>
<td>Moment statistics from selected part of the fitted data, graphs selected data and statistics writes statistics to a file (if desired). Written by Bren Campbell, last modified on July 5, 2004.</td>
</tr>
<tr>
<td>bnmr_globinit.pcm</td>
<td>Initializes most BNMRFIT variables.</td>
</tr>
<tr>
<td>bnmr_par.pcm</td>
<td>It’s actually not part of the source code, but rather a temporary file written by BNMRFIT to store fitting parameters. If BNMRFIT is having trouble remembering or “forgetting” previous run parameters, try deleting this file, or creating a blank one with the same name.</td>
</tr>
<tr>
<td>bnmr_plres.pcm</td>
<td>Show the saved fit parameters on graph.</td>
</tr>
<tr>
<td>bnmr_asy.pcm</td>
<td>This returns the frequency in vector BFRQ, the asymmetry in a matrix BASY(scan number, asymmetry value) and the error in a matrix BAERR(scan number, asymmetry value)</td>
</tr>
<tr>
<td>bnmr_fdt.pcm</td>
<td>Script file for finding deadtime.</td>
</tr>
<tr>
<td>bnmr_graph.pcm</td>
<td>Plot asymmetry for all scans, helicities or combinations.</td>
</tr>
<tr>
<td>bnmr_savefp.pcm</td>
<td>Save fit parameters to table format file.</td>
</tr>
<tr>
<td>bnmr_guess.pcm</td>
<td>Plot the guess of the user’s input, so that user can see if the guess is reasonable.</td>
</tr>
<tr>
<td>bnmr_indvar.pcm</td>
<td>Sort independent variables obtained from header of .msr file. Result of sort is that PPG are at beginning.</td>
</tr>
<tr>
<td>bnmr_selector.pcm</td>
<td>Select different combinations of the data or subsets of the data.</td>
</tr>
<tr>
<td>bnmr_fixvary.pcm</td>
<td>A temporary file written by BNMRFIT to store the current setting of fixing some parameters to certain values. Try deleting this file if BNMRFIT has trouble remembering or “forgetting” which variables to fix.</td>
</tr>
<tr>
<td>bnmr_init.pcm</td>
<td>Initialize variables related to each run. These variables are reset to default for each run. Update this file if DAQ system changes the location of data file.</td>
</tr>
<tr>
<td>bnmr_average.pcm</td>
<td>Find the average on select parts of the data.</td>
</tr>
<tr>
<td>bnmr_fitpar.pcm</td>
<td>A temporary file written by BNMRFIT to store the default initial guess of the fit.</td>
</tr>
<tr>
<td>bnmr_genfun.pcm</td>
<td>Generate fitting function. If you have questions about what equation is used in each fitting, this file is the best starting point.</td>
</tr>
<tr>
<td>bnmr_yab.pcm</td>
<td>Calculate asymmetry and error for all scans (AsyFB and Asyerr). Sum different helicities of the same counter.</td>
</tr>
<tr>
<td>bnmr_fit.pcm</td>
<td>A script for performing the actual fitting.</td>
</tr>
<tr>
<td>bnmr_getrun.pcm</td>
<td>A script that fetches the .msr file and calls the built-in PHYSICA utility to read it. If BNMRFIT seem to have trouble finding the data file, look into this file. The directory of data file is located in bnmr_init.pcm. This file uses the variable directory to know the location of data.</td>
</tr>
<tr>
<td>bnmr_inputpar.pcm</td>
<td>Get initial input, (guess value) for fitting.</td>
</tr>
</tbody>
</table>

Table 1.2: BNMRFIT script descriptions
Chapter 1. BNMRFIT Development

16

- `bnmr_asycalc.pcm; bnmr_asy.pcm; bnmr_yab.pcm; bnmr_getrun.pcm`: Usually the first files to be run, since they read the .msr files and calculates the asymmetry.

- `bnmr_fdt.pcm; bnmr_selector.pcm`: Scripts run after .msr file is read in, but before fitting. They are used to pre-process the data so that they can be fitted.

- `bnmr_inputpar.pcm; bnmr_fit.pcm; bnmr_fitpar.pcm; bnmr_genfun.pcm; bnmr_inputpar.pcm; bnmr_indvar.pcm`: Files related to fitting of data, after data has been pre-processed.

The rest of the files do not fit into one of the particular categories, but have their individual functions.

1.3 Previous attempts of re-writing BNMRFIT

1.3.1 Borland C++ version. (Andre’s)

This is the version that got the furthest, but it’s based on Borland C++ 5.0, and I could only get it to compile on German version of Borland C++ 5.0. I find the code to be difficult to read, and Borland doesn’t seem to provide any easy way of porting its proprietary GUI to Linux.

1.3.2 Andrew Morrison’s gcc version (with his co-op report)

Andrew Morrison’s attempt did not go as far as Andre’s, but he laid an object-oriented framework, and the development is done using open-source tools. At time of writing this report, the source code can be found on computer chin12 on directory `/home/andrewrt/dev/MUDlib`. This version uses `plplot` for its graphical interface, which is not as feature-rich as PHYSICA or Qt3.

1.3.3 My own attempt of breaking up current PHYSICA script files and wrapping it with PERL

Based on suggestion from Gerald, I tried breaking up the bnmrfit script into smaller files, and assigning an alias to each file. The advantage of this is that when a command crashes, it goes back to physica environment, and the user can try the command again, instead of restarting bnmrfit, reloading the run, and repeat the process of fitting. I made some progress in breaking up major tasks such as loading the run, selecting the range, and fitting. Going this route will likely require a substantial amount of investment in coding, and I abandoned the
Chapter 1. BNMRFIT Development

1.3.4 BNMRMINFIT

BNMRMINFIT is a FORTRAN program that performs BNMR fitting by accepting input files written by the user. It's more flexible, and faster than BNMRFIT, but it does not have a graphical interface or real-time user interaction. Currently it can be found in folder fit-bnmr-fortran under the username markx. Its documentation files are fit-bnmr-howto.txt and inputbnmr_explained.p. fit-bnmr-howto.txt provides brief instruction on how to use BNMRMINFIT, while inputbnmr_explained.p is a sample input file with extensive documentation on the function of each line.

1.3.5 Hassan's work on rewriting BNMRFIT

Hassan is currently rewriting BNMRFIT using C++ and Qt3. At the time of writing this report, Hassan’s program uses BNMRFIT to perform deadtime correction and helicity combination. The user exports BNMRFIT results to ASCII files, and load them using the new fitting program. Currently it seem very promising, especially if it can borrow some existing code from previous C++ attempts.

1.3.6 Using commercial software, such as Origin

Another method of doing fitting is exporting data to ASCII file in BNMRFIT, and load them into a commercial software such as Origin or Matlab. TRIUMF has a site license for Matlab, which is a popular choice for data analysis. To load Matlab, simply type “matlab” in linux terminal of any linux computer with triumfcs installed.

1.4 My opinion on future development

After seeing the presentation by Hassan, I feel that his project has the best chance of completing. I’ve given him a hand-written summary of the deadtime correction algorithm and asymmetry calculation algorithm for SLR mode. I spent some time learning about Qt, and found it to be better than Borland C++. Since it’s linux based, and has an open-source version, we can reasonably expect it to be maintained in the near future. Given the prominence of Linux
machines at TRIUMF, a software based on Qt3 will be easier to use than a software based on the Windows platform.
Chapter 2

Demagnetization report

2.1 Summary

The effect of demagnetization due to magnetic thin films was calculated, using oblate ellipsoid as an approximation. Geometric correction to this approximation is applied using techniques of magnetic charges and analogy to parallel plate capacitors. Extrapolation of simulation data was used to confirm theoretical calculations. Calculation result was applied to silver and palladium samples.

2.2 What is demagnetization

When a magnetic material of arbitrary shape is placed in a uniform applied field $H_0$ and becomes magnetized, the magnetic field inside and outside the object is modified by the magnetization of the object. This modification is often non-uniform throughout the sample, due to the contributions from various parts of the object. The exact change in magnetic field due to the geometry of the magnetized object is called “demagnetization”, and has been studied extensively for various finite sized shapes. If the material is a homogeneous ellipsoid, the effect of demagnetization can be calculated analytically. In the case of very thin films, approximation by a very thin oblate ellipsoid can be used, by applying the extreme case condition. The finite thickness correction factor for the extreme case can be calculated using either ellipsoid approximation, or by analogy to a parallel plate capacitor. The demagnetization effect is present both inside and outside the object. For a thin film sample, the demagnetization effect is mostly confined to the interior of film, while the demagnetization effect outside the thin film is equal to the finite size correction. As we will show later, the finite size correction leads to a correction a few orders of magnitude smaller than the sensitivity of our experiment, while the demagnetization effect in the interior of the film is large enough to be observed for certain strongly paramagnetic materials.

One thing worth noting is that the $\beta - NMR$ experiment measures the value of $H$, which includes contribution from magnetic dipoles, while the value of $B$ does not. Using the equation for the Larmor frequency

$$\omega_0 = \gamma H$$

(2.1)
Although $B$ and $H$ are often used interchangeably, inside magnetic materials, there is a difference. As we will show later, for thin films, $B$ is the same inside and outside the film, while $H$ is different.\[1\] Demagnetization effects exist for all magnetic materials, but this article only discusses applications in paramagnetic samples, as that’s what we are interested in.

### 2.3 demagnetization and its effect on the local internal field

When a paramagnetic object is placed in a uniform magnetic field, magnetic dipole moments affect the internal $H$ field, but $B$ field stays the same. Inside the object, $H$ decreases due to contributions from magnetic dipole moments. In NMR, we are interested in the resonance frequency, which is determined by the local magnetic field $H_{loc} = H_0 + H_m + H_{other}$. In this equation, $H_0$ is the magnetic field caused by free current running through the coil, while $H_m$ is caused by the gradient of magnetic dipole moment at the boundary of object. Without demagnetization effect, we have $\omega = \gamma H_0 = \omega_0$. Therefore, magnetization contributes to a shifting of resonance frequency, which is included in resonance frequency measured in NMR. In NMR, one parameter we are interested in is the relative frequency shift, such as the Knight shift. The experimental quantity is

$$k_{\text{observed}} = \frac{\omega_{\text{observed}} - \omega_0}{\omega_0}$$

(2.2)

The measured parameter, $\omega_{\text{observed}}$, includes contribution from both demagnetization effects, and frequency shift due to properties of the material. Since the purpose of the experiment usually is to find out the contribution from material being studied, it is necessary to subtract the effect of demagnetization from the measured frequency shift. Therefore, the observed knight shift should be

$$k_{\text{observed}} = \frac{\gamma(H_0 + H_{dem} + H_{\text{knight}}) - \gamma H_0}{\gamma H_0}$$

(2.3)

which reduces to

$$k_{\text{observed}} = \frac{H_{dem} + H_{\text{knight}}}{H_0}$$

(2.4)

where $H_{\text{knight}}$ represents shifts in magnetic field due to intrinsic property of material. Expressed in measured frequency, we have

$$k_{\text{knight}} = \frac{\omega_{\text{observed}} - (\omega_0 - \gamma H_{\text{dem}})}{\omega_0}$$

(2.5)

where $\omega_0$ is the larmor resonance frequency of the free spin, and $\omega_{\text{measured}}$ is resonance frequency measured in experiment. In practice, $\omega_0$ is measured in a nonmagnetic cubic insulator such as MgO, which is the capping layer or substrate.
In the above equations, we have assumed that the change of magnetic field due to demagnetization is uniform. Even though this is only valid for ellipsoidal objects, powders and thin film samples are usually well-approximated by an ellipsoid. Using this approximation, $H_m$ can be approximated as

$$H_m = N \times M$$

(2.6)

where $N$ is the demagnetization factor, and $M$ is the bulk magnetization of sample. $N$ is a unitless variable that depends on the geometric shape of sample, and ranges from 0 to 1 in mks units, and 0 to $4\pi$ in cgs units. For solid samples, $N$ can be calculated by approximating an ellipsoid. For powder samples, the packing factor of the sample should also be taken into account. In the case of thin film samples, $N$ is usually approximated as 1.

### 2.4 Demagnetization and its effect on the Knight shift

For a magnetic object in a uniform magnetic field, the field outside the object, far from the sample, is:

$$B = \mu_0(H_0 + H_m)$$

(2.7)

Outside the object, $H_m$ is just the dipole field of the magnetized object. At distance far enough away from the sample, $H_m$ can even be approximated by the field of a point dipole. On the other hand, the field inside the object is:

$$B = \mu_0(H_0 + H_m + M)$$

(2.8)

The term $H_m$ is the bulk demagnetization we are looking for, for both outside and inside the thin film. Outside the thin film, it accounts for the disturbance of local magnetic field due to the geometry of the sample. Inside the sample, it arises from discontinuity of magnetization at the surface of thin film. In the limit of an infinitely thin film, lines of $H_m$ emerge uniform and perpendicular to the surface in both directions, just as lines of $E$ from a uniform plane distribution of charge. Outside the plate, the contribution to $H$ from the surface discontinuity in $M$ at each surface will just cancel. Thus $H_m$ contributes nothing. Inside the plate, Gauss’s flux theorem shows that $H_m$ is equal to $-M$, as the expression for $B$ becomes

$$B = \mu_0[(H_0 - M) + M] = \mu_0 H_0$$

(2.9)

In the above equation, $H_0 - M$ is the quantity that $\beta - NMR$ measures. In this case, the demagnetization factor $N$ is equal to $-1$.

One way of confirming this is to use the formula for calculating the demagnetization factor of an oblate ellipsoid, and show that as ellipsoid gets very thin, the first and second order terms vanish, and $N$ becomes one. The expression of $N$ for an oblate ellipsoid\[4\] is

$$N = -4\pi(1 - \frac{\pi}{2m} + \frac{2}{m^2})$$

(2.10)
where \( m = a/c \) is the ratio of long and short axis of ellipsoid. For a very flat oblate spheroid, \( m \) approaches infinity, and \( N \) approaches \( 4\pi \), in cgs units. The first-order correction term is inversely proportional to \( m \), which we will discuss later in finite thickness corrections. Another equation, calculating demagnetizing field along its axis in a disc magnetized perpendicular to its plane can be derived\[5\] through classical electromagnetics as:

\[
N = 4\pi \left( -1 + \frac{\frac{c}{a}}{1 + (c/a)^2} \right) \tag{2.11}
\]

In the limit where \( c/a \) approaches zero, \( N \) approaches \( 4\pi \) in cgs units, or 1 in mks units. Here \( c \) and \( a \) are the long and short axis of ellipsoid.

The demagnetization effect can be related to Knight shift by

\[
k_{\text{correct}} = k_{\text{observed}} - k_{\text{dem}} - k_{\text{Lorenz}} \tag{2.12}
\]

In the above equation, \( k_{\text{dem}} \) is proportional to the value of demagnetization. When the demagnetization factor is \(-1\), value of \( k_{\text{dem}} \) is equal to the magnitude of magnetization. The Lorenzian effect arises from the sphere that Lithium ion occupies, as shown in Figure 2.1. \( k_{\text{Lorenz}} \) is equal of \( \frac{1}{3} \) of \( M \), and it’s opposite sign of \( k_{\text{dem}} \) for a paramagnetic material. Therefore, in the case of thin films, without considering finite thickness corrections, \( k_{\text{dem}} \) and \( k_{\text{Lorenz}} \) combine to have an effective \( N \) of \( \frac{2}{3} \) in mks units, or \( 4\pi \frac{2}{3} \) in cgs units. Therefore,

\[
H_m = 4\pi \frac{2}{3} M \tag{2.13}
\]

After Lorenzian effect is considered.

### 2.5 Addressing finite thickness effects

In the previous section, we assumed the film to be infinitely thin. Now we take a look at the effects of finite geometric dimensions. My own approach to
finite thickness correction is utilizing the similarity to thin parallel capacitors. The magnetic thin film is modeled as a very thin cylinder containing fictitious magnetic charges on both top and bottom surfaces, like a “magnetic parallel plate capacitor”. After that, the electrical field calculation of a parallel plate capacitor is applied, to obtain the magnitude of magnetic fields right outside the thin film. This approach is important for $\beta$-NMR samples, since our thin film can not be approximated as ellipsoid. We are interested in the magnetic field right outside the thin film, since some lithium ions stop right below or above the thin film, on the substrate or the capping layer. The signal obtained from those lithium ions can be used as a reference like $\omega_0$, therefore it’s important to know the local magnetic field disturbance due the thin film accurately.

A uniformly magnetized flat cylinder can be considered to be equivalent to two circular parallel plates of magnetic charges. [1] According to Maxwell’s equations, the cumulative effect of magnetic dipole moments inside the cylinder can be modelled as uniform and opposite surface magnetic charge density $\sigma = M_0$ per $m^2$ on both surfaces. If the radius of cylinder, $R$, is much greater than the thickness, $L$, the cylinder becomes the magnetic equivalent of parallel plate capacitor problem. Using this model, the magnetic field outside is

$$B_{outside} = \frac{L}{2R}M$$  \hspace{1cm} (2.14)

According to this equation, the finite thickness correction to demagnetization factor is proportional to the ratio of thickness to radius. This result is identical to Equation 2.11, and close to the result of Equation 2.10. In the next section, we’ll see that due to the small thickness of thin film samples used in $\beta-NMR$, this correction due to finite thickness is negligible.

### 2.6 Sample Calculations

We now apply the equations to various materials, and have a quantitative look at the demagnetization effect. A first example is a palladium sample, which we approximate as a thin cylinder. The maximum susceptibility of Palladium is $700 \times 10^{-6}$emu/mole, at 175K. The molar mass of Pd is 106.43g/mol, and Palladium density 12023 $kg/m^3$. Calculations will be done in cgs units. Assuming demagnetization factor of 1, the knight shift can be calculated as:

$$\chi_{cgs} = \chi_{molar} \times \rho \\
= 700 \times 10^{-6} \frac{\text{emu}}{\text{mol}} \times 12.034 \frac{g}{cm^3} \frac{1\text{mol}}{106.4g} \\
= 79 \times 10^{-6} \frac{\text{emu}}{cm^3}$$  \hspace{1cm} (2.15)

Where $\chi_{cgs}$ is the volume susceptibility in $emu/cm^3$, or a unitless variable. The correction to knight shift, which includes both demagnetization effect and the lorentzian field effect, is slightly less

$$k_{lor} = \frac{4\pi}{3} \frac{M}{H_0}$$
Chapter 2. Demagnetization report

\[
\kappa_{\text{dem}} = \frac{N \times M}{H} = N\chi = 4\pi \times N\chi_{\text{cgs}} \tag{2.16}
\]

Combining the bulk demagnetization and the lorenzian effect, we have:

\[
k = 4\pi \left(\frac{1}{3} - 1\right)\chi_{\text{cgs}}
\]

The knight shift is about 660ppm, which is comparable to the shift measured experimentally.

Now we look at finite thickness effects. Since the finite thickness correction to value of \(N\) is proportional to ratio of thickness to width, and knight shift is proportional to value of \(N\), we have

\[
K_{\text{outside}} = \frac{T}{2R}K_{\text{in}} \tag{2.18}
\]

with thickness \(T = 10\text{nm}\) and radius \(R = 10\text{mm}\), and \(K_{\text{in}} = 4\pi \times 79 \times 10^{-6}\), we obtain \(K_{\text{outside}} = 496 \times 10^{-12}\). In external magnetic field of 4.1 tesla, the finite thickness effect of local magnetic field outside the magnetic field is

\[
B_{\text{adj}} = K_{\text{outside}} \times H_0 B_{\text{adj}} = 20.3 \mu G \tag{2.19}
\]

The result is in the micro-Gauss range, which is too small to detect with our current technique.

Another sample we can look at is silver, which has magnetic susceptibility \(\mu_{\text{Ag}} = 9.6 \times 10^{-6}\text{emu/mol}\), which is temperature independent. Similar calculation shows that the knight shift due to demagnetization and lorenz field is

\[
K_{\text{dem,L}} = 4\pi \left(\frac{1}{3} - 1\right)p_{\text{mol}}\chi_{\text{mol}} \tag{2.20}
\]

\[
= -7.8 \times 10^{-6} \tag{2.21}
\]

The shift is only 7.8ppm, which is fairly small. The effect of finite thickness correction will be even smaller, and impossible to detect.

2.7 COMSOL simulation

To confirm the validity of the equations used, and the unit conversions, COMSOL was used to simulate the magnetic field produced by a uniform flat cylinder. COMSOL uses finite-element analysis for simulation, therefore it is limited in the ratio of radius to thickness. I simulated cylinders of various radius-to-thickness ratios, and extrapolated the result to the dimensions of palladium film used in
Figure 2.2: COMSOL simulation results, magnetic field vs thickness, linear scale

our experiment. The result agrees with theoretical calculation. Table (1) shows the simulation results. The simulation results do not have uncertainties associated with them, due to the limit in simulation software. Figure (2.2) shows the data in linear scale, while figure (2.3) shows the same data, but in log scale.

<table>
<thead>
<tr>
<th>D(mm)</th>
<th>T(mm)</th>
<th>T/D</th>
<th>B (T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>\frac{1}{5}</td>
<td>2.2 \times 10^{-8}</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>\frac{10}{1}</td>
<td>1.05 \times 10^{-8}</td>
</tr>
<tr>
<td>10</td>
<td>0.5</td>
<td>\frac{10}{2}</td>
<td>4.3 \times 10^{-9}</td>
</tr>
<tr>
<td>10</td>
<td>0.25</td>
<td>\frac{10}{4}</td>
<td>2.7 \times 10^{-9}</td>
</tr>
<tr>
<td>10</td>
<td>0.125</td>
<td>\frac{10}{8}</td>
<td>1.35 \times 10^{-9}</td>
</tr>
<tr>
<td>10</td>
<td>0.0625</td>
<td>\frac{10}{16}</td>
<td>6.7 \times 10^{-10}</td>
</tr>
</tbody>
</table>

Table 2.1: COMSOL simulation results
Figure 2.3: COMSOL simulation results, magnetic field vs thickness, log scale
Bibliography


