Moessfit
a Mössbauer fitting program

Sirko Kamusella

March 30, 2017

Contents

1 Introduction

Moessfit is a fitting programm written for complex fitting tasks including

- global fit parameters
- maximum entropy distributions
- user defined parameter distributions

- usage run specific parameter function as temperature dependent magnetic order parameter or Debye models for Isomer shift and spectral area (Debye-Waller factor)

To run the program you have to run the MoessFit.exe. The program needs access to the following dll located in the same folder like the executable:

- Windows XP: libgcc_s_dw2-1.dll, mingwm10.dll , QtCored4.dll and QtGuid4.dll, libgsl-0.dll, libgslcblas-0.dll.
- Windows7/8: Qt5Widgets.dll, icudt51.dll, icuin51.dll, icuuc51.dll, libEGL.dll, libGLESv2.dll, Qt5Core.dll, Qt5Gui.dll

The aim of Moessfit is the simultaneous fitting of multiple Mössbauer spectra (runs) with a common model (theory). For this purpose a data input from a mbs-textfile similar to musrfit[SW12] is used. By default the program is set up for ^{57}Fe Mössbauer spectroscopy, but may be used for any $I = 1/2 \leftrightarrow I = 3/2$ transition using *Nucleus* COMMAND (sec. 6). In sec. 8 it is explained how to use Moessfit to fit any analytical function to an arbitrary set of data.

1.1 Structure of a fit project and mbs-file

A typical Moessfit fitting project consists of a project folder consisting of subfolders which contain folded Mössbauer spectra (*.fld). Please note that only the newest fld-files of each subfolder may be considered by Moessfit (sec. 4). The project folder contains the Moessfit input file (*.mbs), which is a textfile with the following blocks: FITPARAMETER, THEORY, RUN, FUNCTIONS, COMMANDS, PLOT, FITDATA, MEMDATA and FITSTATISTICS. The first three are necessary for a valid Moessfit input. The blocks are explained in the sections below. A block starts by its identifier e.g. "THEORY" and ends at "##". Although the user can edit parameters via the main panel it is strongly recommended to get used to an advanced text editor (Notepad++, Medit etc.) to modify the mbs-file directly and reload it subsequently. Are working example is shown in sec. 3.3.

1.2 Operation

To perform a fit set up the *.mbs, select it under "File→open file" or use command line arguments passing. Moessfit supports all file types which can be opened by "File→open" as command line arguments, e.g.

- *.mbs: Moessfit fit model
- *.fld: loads processed data (velocity-count) and generates a mbs-file in the same folder to treat the data with an static Hamiltonian approach. Multiple fld-files can be opened to be handed to an single mbs-file.
- *.dat/*.ws5: raw data can be folded with Moessfit using a cos-velocity function, a fld-file will be generated in the same folder. Moessfit recommends a "probable fold channel" based on a auto correlation. If the user uses a float fold channel, only the velocities will be calculated, the number of data channels stays constant.

Keep in mind the following keys to operate Moessfit:

- "f" ... fit data
- "v" ... view data
- "e" ... error calculation. Has to be done manually as it can be more time consuming than the fit itself. Moessfit estimates 68.3 % (1σ) confidence intervals. Have a look on Errors-command for non-covariant error calculation.
- "u"/"r"/F5 ... reload/update model from mbs-file

1.3 Keep in mind the following hints

- Report Debugs and personal wishes concerning MoessFit directly to sirko.kammusella@tu-dresden.de
- Moessfit automatically generates a default mbs-file, if a fld-file is loaded. To tell the truth: Moessfit will actually load this mbs-file (including a default model), generated from the selected fld-file. For future work just rename that file.

- Use the I0-COMMANDS to auto-initialize the baseline and calculate the spectral face by an absorption coefficient. Alternatively auto-initialize the spectral area using the A-COMMANDS.
- Delete all the appended FITDATA section in your mbs-file and reload, if you want to start new.
- If a fit model successfully loads but does show no or wrong plots, verify the fld specified in the RUN block. At least one of them might contain weird data.
- to temporary disable lines within a block, type a single "#" at the beginning of the concerning line. "##" will end the block at this position.
- Moessfit supports multithreading, so take advantage of a multi core CPU if available to improve performance by the count of cores.
- Restrict the range of a fit parameter p to $[a, b]$ by introducing the function $a + (b - a) \cdot e^{-p^2}$
- Use the links in the plots and the mouse wheel to navigate through the data.
- "-1" error values indicate, that either the fitting minimum is yet not reached or that parameters are highly correlated.
- Extract spectra by plotting them and then using Menu Plot$\longrightarrow$ ...to clipboard
- A variable global parameter instantly leads to a global fit, which means: (a) the Nelder-Mead-simplex algorithm is replaced by a primitive steepest gradient algorithm (to keep simplex use ForceSimplex-COMMANDS) (b) multithreading can not be applied as efficient as in a serial fit.
- Respect the feasibility of error calculations: Consider a global fit of a 7 Parameter model (3 global) for 20 runs. This leads to a Hessian matrix with 389 different non-zero matrix entries. Every entry referring a global parameter effects the recalculation of 20 Mössbauer spectra. Even if there is only one such calculation necessary to determine a second derivative, altogether at least 5240 Mössbauer spectra have to be calculated. If these Mössbauer spectra contain a MEM, the error calculation can easily exceed 1 hour of calculation time. Consider the alternative Error calculation modes which be used by Errors-COMMANDS
- MoessFit takes fld-, mbs-, dat- and ws5-files as command line arguments, so auto-link them to the exe.
- Have a look on the example folder representing typical subjects of MoessFit.
- Change single fitting parameter values in the appended fitting section and reload to influence the data directly. More easily take advantage of the main panel.
- Mössbauer deals rather with physical numbers than "mm/s", this concerns especially the field gradient and magnetic hyperfine field. For conversation you should use the following values being applied in Moessfit:
  - $\frac{ceQ}{2E_\gamma} = 0.0167 \frac{mm/s}{V/\mathring{A}^2}$ in the quadrupolar interaction $\Delta v_{QS} = \frac{ceQ}{2E_\gamma} \cdot V_{zz} \sqrt{1 + \frac{\eta^2}{3}}$. This corresponds to an quadrupolar moment of 160 mb [Pyy08, DBS95].
  - $g_{1/2} = 0.18088$, $g_{3/2} = -0.10327$ [Bha98] and $\frac{c\mu_n}{E_\gamma} = 0.65572 \frac{mm/s}{T}$ for the Zeeman splitting $\Delta v_{zeeman} = g_I \frac{c\mu_n}{E_\gamma} B$
- If the number of MEM-Iteration meats the specified maximum, you have to increase the maximum or increase the smoothness of the distribution by reducing λ' (see COMMANDS-MEM)

2 FITPARAMETER

The FITPARAMETER block lists all parameters used in THEORY block and describes and classifies them into run specific fitting, global fitting and constant. Only fitting parameters will be effected by the fitting routine. Each line stand for a fit parameter and consists of name, initial value and step size. A step size of zero marks a parameter to be a constant. If a name begins with "global_", then this fit parameter will be treated as global, else as run specific. A typical FITPARAMETER block is shown in the next section.

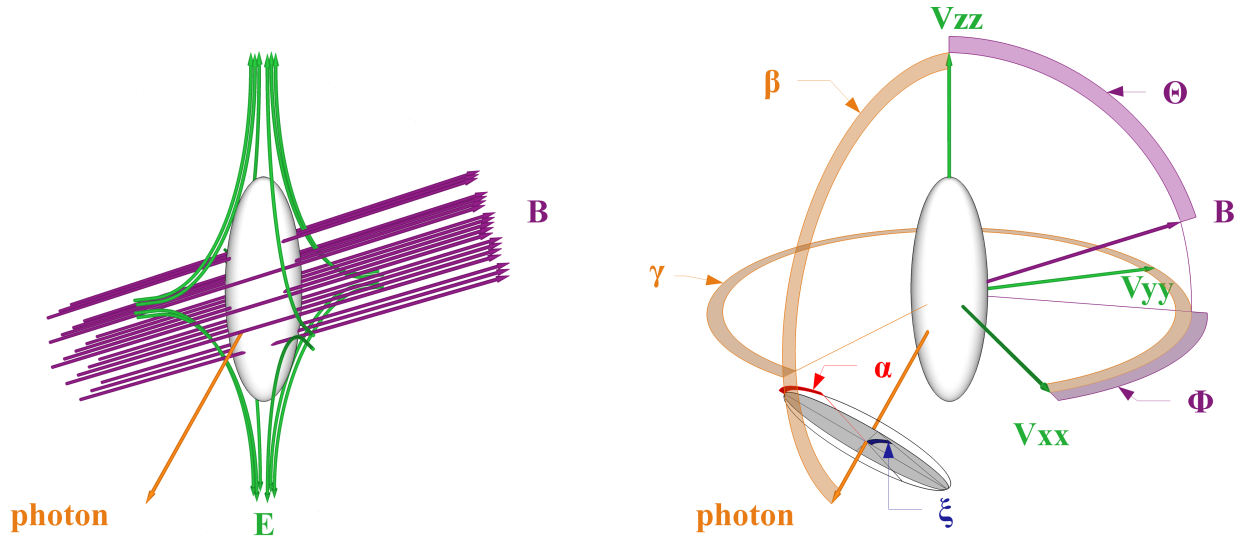
Keep in mind that the decimal place of step size is related to the decimal place of the fitdata and its calculated errors in the FITDATA output.

3 THEORY

The THEORY block sums up several spectra construction commands which are separated by line breaks. These predefined constructors consist of an identifier and a parameter list. So far, the following predefined constructors exist.

3.1 Constructors

- **BL 1**
  Baseline, sets a constant channel content for all channels
  1: I_0 [], channel content, counts



(a) Orientation of the nucleus in a predominant electric field gradient and a small magnetic field B.

(b) Angles of absorption and magnetic field in the electric field gradient coordinate system.

Figure 1: Electric/Magnetic environment and photon polarisation and its conventional description using the angles $\Theta, \Phi, \beta, \gamma, \alpha, \xi$

- **SHp 1 2 3 4 5 6 7 8**
  Static Hamiltonian, powder. Directions are described in the electric field gradient coordinate system (fig. 1): $V_{zz} = (0, 0, 1)$, $V_{yy} = (0, 1, 0)$, $V_{xx} = (1, 0, 0)$, $\vec{B} = (\sin\Theta\cos\Phi, \sin\Theta\sin\Phi, \cos\Theta)$
  1: B [T], magnetic hyperfine field
  2: V_{zz} [V/Å^2], principal component of field gradient
  3: η, asymmetry parameter
  4: Θ [°], polar angle (B, V_{zz})
  5: Φ [°], azimuthal angle (B, V_{zz})
  6: CS [mm/s], center shift
  7: ω [mm/s], line width
  8: A [mm/s], spectral Area

- **SHpLev 1 2 3 4 5 6 7 8 9**
  similar to SHp, but estimating the saturation effect for thick absorber with an effective thickness t_a. The line intensity of the i$^{\text{th}}$ line is estimated by[CY07, 71,377]

$$I(t_{a,i} = t_a * f_i) = t_{a,i} \cdot e^{-t_{a,i}/2} \cdot (I_0(t_{a,i}/2) + I_1(t_{a,i}/2)) \quad (1)$$

  with the thin absorber line intensity f_i ($\sum_i f_i = 1$). This approximation is valid for well separated lines and intended to treat thick absorber spectra with maximum entropy method. For a proper thick absorber

analysis use the *TransmissionIntegral* COMMAND instead. In this spectrum constructor t_a only describes the levelling, and not the line broadening and not the total absorption area, with is given by A instead. 1: B [T], magnetic hyperfine field
2: V_{zz} [V/Å^2], principal component of field gradient
3: η, asymmetry parameter
4: Θ [°], polar angle (B, V_{zz})
5: Φ [°], azimuthal angle (B, V_{zz})
6: CS [mm/s], center shift
7: ω [mm/s], line width
8: A [mm/s], spectral Area
9: t_a, effective thickness of the absorber

- **SHc 1 2 3 4 5 6 7 8 9 10**
  Static Hamiltonian, crystal. Directions are described in the electric field gradient coordinate system (fig. 1): $V_{zz} = (0, 0, 1)$, $V_{yy} = (0, 1, 0)$, $V_{xx} = (1, 0, 0)$, $\vec{B} = (\sin\Theta\cos\Phi, \sin\Theta\sin\Phi, \cos\Theta)$, $\vec{G} = (\sin\beta\cos\gamma, \sin\beta\sin\gamma, \cos\beta)$
  1: B [T], magnetic hyperfine field
  2: V_{zz} [V/Å^2], principal component of field gradient
  3: η, asymmetry parameter
  4: Θ [°], polar angle (B, V_{zz})
  5: Φ [°], azimuthal angle (B, V_{zz})
  6: CS [mm/s], center shift
  7: ω [mm/s], line width
  8: A [mm/s], spectral Area
  9: β [°], polar texture angle
  10: γ [°], azimuthal texture angle

- **SHcG 1 2 3 4 5 6 7 8 9 10**

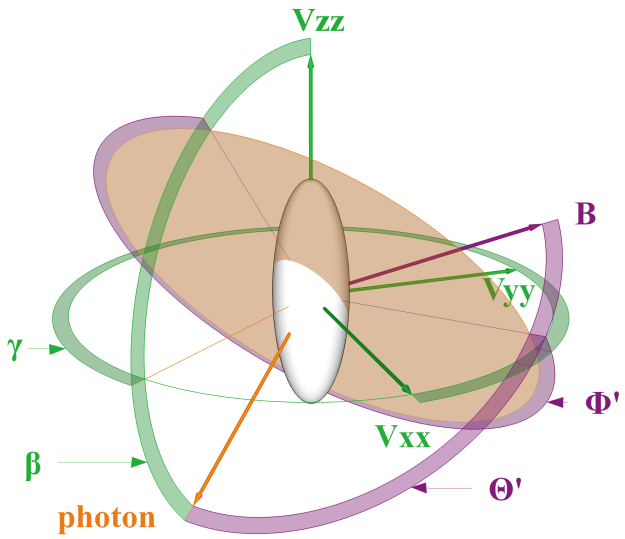


Figure 2: Photon coordinate system assumed in the **SHcG** construction command.

Directions are described in the photon's coordinate system (fig. 2): $\vec{G} = (0, 0, 1)$, $V_{zz} = (\sin\beta, 0, \cos\beta)$, $\vec{B} = (\sin\Theta'\cos\Phi', \sin\Theta'\sin\Phi', \cos\Theta')$ The V_{xx}-axis is obtained by rotating the axis $(\sin(\beta-\pi), 0, \cos(\beta-\pi))$ by an angle of $-\gamma$ around the V_{zz}-axis.

- **SHcB 1 2 3 4 5 6 7 8 9 10**

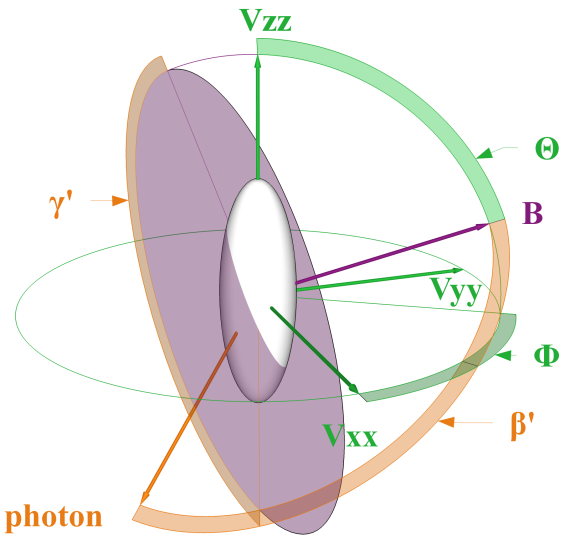


Figure 3: Magnetic field coordinate system assumed in the **SHcB** construction command.

Directions are described in the magnetic field coordinate system (fig. 3): $\vec{B} = (0, 0, 1)$, $V_{zz} = (\sin\Theta, 0, \cos\theta)$, $\vec{G} = (\sin\beta'\cos\gamma', \sin\beta'\sin\gamma', \cos\beta')$ The V_{xx}-axis is obtained by rotating the axis $(\sin(\Theta - \pi), 0, \cos(\Theta - \pi))$ by an angle of $-\Phi$ around the V_{zz}-axis.

- **Dyn 1 2 3**
  11 12 13 14 15 16 1W2 ... 1WJ
  21 22 23 24 25 26 2W1 ... 2WJ
  ...
  Blume dynamic line shape model [Blu68], defines J states each with $(J - 1)$ fluctuation rates from one state to the others. We stress that Moessfit at the moment only uses the naive implementation for powder spectra inverting a 8Jx8J matrix for every velocity. According to Clausser [Cla71] this can be avoided by solving a general complex diagonalization, saving two orders of magnitude computation time [SD76]. At the moment, this kind of calculation is not available in Moessfit. However, analytical inversions are possible for two or three diagonal states, they are applicable as DynDiagDiag and DynDiagDiagDiag.
  1: J , number of states
  2: ω [mm/s], global line width for all states
  3: A [mm/s], spectral area
  i1: B [T], state i, magnetic hyperfine field
  i2: V_{zz} [V/Å^2], state i, principal component of field gradient
  i3: η, state i, asymmetry parameter
  i4: Θ [°], state i, polar angle (B, V_{zz})
  i5: Φ [°], state i, azimuthal angle (B, V_{zz})
  i6: CS [mm/s], state i, center shift
  iWj: W_{ij} [MHz], fluctuation rate from state i to state j, provide $(J - 1)$ Fluctuation rates for every state
  The following example demonstrates the syntax for a fluctuation between two field gradients and an magnetic hyperfine field:
  ##########
  THEORY
  BL I0
  Dyn J omega A
  Bhyp 0 0 0 0 CS1 W12 W13
  0 Vzz1 0 0 0 CS1 W21 W23
  0 Vzz2 0 0 0 CS1 W31 W32
  ##########

- **DynDiagDiag 1 2 3 4 5 6 7 8 9 10 11**
  Blume dynamic line shape model [Blu68] for two states, in each case with diagonal Hamiltonian. Compared to the "Dyn" command DynDiagDiag is mush faster.
  1: B_1 [T], state 1, magnetic hyperfine field
  2: B_2 [T], state 2, magnetic hyperfine field
  3: V_{zz1} [V/Å^2], state 1, principal component of field gradient
  4: V_{zz2} [V/Å^2], state 2, principal component of field gradient
  5: CS_1 [mm/s], state 1, center shift
  6: CS_2 [mm/s], state 2, center shift
  7: ω [mm/s], line width
  8: A [mm/s], spectral Area
  9: β [°], angle between quantization axis and gamma beam, type magic angle (54.7°) for powder
  10: W_{12} [MHz], fluctuation rate from state 1 to state 2
  11: W_{21} [MHz], fluctuation rate from state 2 to state 1
  The following example demonstrates a flipping magnetic dipole with the principal axis of the field gradient parallel to the magnetic moment:
  ##########
  THEORY
  BL I0
  DynDiagDiag Bplus Bminus Vzz Vzz CS CS omega A0 54.7 W W
  ##########

- **DynDiagDiagDiag 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19**
  similiar to **DynDiagDiag**, but with three states
  1-3: B [T], state 1-3, magnetic hyperfine field
  4-6: V_{zz} [V/Å^2], state 1-3, principal component of field gradient
  7-9: CS [mm/s], state 1-3, center shift
  10: ω [mm/s], line width
  11: A [mm/s], spectral Area
  12: β [°], angle between quantization axis and gamma beam, type magic angle (54.7°) for powder
  13-18: $W_{12}, W_{13}, W_{21}, W_{23}, W_{31}, W_{32}$ [MHz], fluctuation rates

- **MLR 1 2 3 4 5 6 7 8 9**
  Multi Level Relaxation model as described in [Chu11] without presence of electric field gradient, describes $2S+1$ spin states in a parabolic potential, identified by the spin orientation m_i and energy $E_i = -KVm_i^2/S^2$
  1: B [T], magnetic hyperfine field, canted field is calculated by $B_i = Bm_i/S$
  2: V_{zz} [V/Å^2], principal component of field gradient parallel to the field
  3: CS [mm/s], center shift
  4: ω [mm/s], line width
  5: A [mm/s], spectral Area
  6: β [°], angle between quantization axis and gamma beam, type magic angle (54.7°) for powder, phenomenological only in this model
  7: KV/k_BT, ratio between energy barrier KV and thermal energy k_BT
  8: D [MHz], Diffusion constant, fluctuation rates are proportional calculated by $W_{ij} = (\delta_{i+1,j} + \delta_{i-1,j}) \cdot D(S(S+1) - m_i m_j) \cdot \min(1, e^{-\frac{E_j - E_i}{k_B T}})$
  9: S, total moment quantum number, choose $S > 30$ for a realistic spectrum

- **mSSH 1 2 3 4 5 6 7 8 9 10 11 12 13 14**
  magnetized source and Static Hamiltonian sample
  1: B_{spl} [T], magnetic hyperfine field in sample
  2: V_{zz} [V/Å^2], principal component of field gradient
  3: η, asymmetry parameter
  4: Θ [°], polar angle (B, V_{zz})
  5: Φ [°], azimuthal angle (B, V_{zz})
  6: CS [mm/s], center shift

7: ω [mm/s], line width
8: A [mm/s], spectral Area
9: β [°], polar texture angle
10: γ [°], azimuthal texture angle
11: B_{src} [T], magnetic hyperfine field of source
12: β_{src} [°], polar orientation of source magnetization with respect to the gamma beam, 0°...longitudinal polarization, 90° transverse polarization
13: α [°], azimuthal tilting (around gamma beam) of the magnetization with respect to principal axis of the sample system
14: $\epsilon \in [0, 1]$, polarisation, for an source in field typically =1

- **afmBex 1 2 3 4 5 6 7 8 9 10 11**
  antiferromagnet (uniaxial) in external fields, transverse geometry, powder
  1: B_{afm} [T], internal field of the moments
  2: B_A [T], anisotropy field
  3: B_J [T], exchange field
  4: B_{ex} [T], exchange field
  5: V_{zz} [V/Å^2], principal component of field gradient
  6: Θ [°], polar (B_{afm}, V_{zz})
  7: CS [mm/s], center shift
  8: ω [mm/s], line width
  9: A [mm/s], spectral Area
  10: N, direction number to simulate powder
  11: $N_{V_{zz}}$, V_{zz} sample number, rotated around easy axis

- **afmBexLong 1 2 3 4 5 6 7 8 9 10 11**
  same as **afmBex**, but in longitudinal field mode

- **fmBex 1 2 3 4 5 6 7 8 9 10**
  ferromagnet (uniaxial) in external fields, transverse geometry, powder
  1: B_{afm} [T], internal field of the moments
  2: B_A [T], anisotropy field
  3: B_{ex} [T], exchange field
  4: V_{zz} [V/Å^2], principal component of field gradient
  5: Θ [°], polar (B_{afm}, V_{zz})
  6: CS [mm/s], center shift
  7: ω [mm/s], line width
  8: A [mm/s], spectral Area
  9: N, direction number to simulate powder
  10: $N_{V_{zz}}$, V_{zz} sample number, rotated around easy axis

- **FeCal 1 2 3 4 5 6 7 8**
  iron foil calibration, assuming harmonic drive input, subfolder names should represent monitor voltage in mV
  1: ω_1 [mm/s], outer line width
  2: ω_2 [mm/s], mid line width
  3: ω_3 [mm/s], inner line width
  4: A [mm/s], outer peak Area
  5: A [mm/s], mid peak Area
  6: A [mm/s], inner peak Area
  7: α [(mm/s)/mV], calibration factor of monitor signal
  8: CS [mm/s], center shift

- **FeCaltriang 1 2 3 4 5 6 7 8**
  iron foil calibration, assuming triangular drive input, *rsp* should represent monitor voltage in mV

1: ω_1 [mm/s], outer line width
2: ω_2 [mm/s], mid line width
3: ω_3 [mm/s], inner line width
4: A [mm/s], outer peak Area
5: A [mm/s], mid peak Area
6: A [mm/s], inner peak Area
7: α [(mm/s)/mV], calibration factor of monitor signal
8: CS [mm/s], center shift
5: Δv_{12} [mm/s], line position output

- **PseudoVoigt 1 2 3 4**
  Pseudo-Voigt profile, $I(v) = \eta \cdot \frac{1}{\pi\omega} \frac{1}{1+\left(\frac{v-CS}{w}\right)^2} + (1-\eta) \cdot \frac{\sqrt{\ln 2}}{\sqrt{\pi}\omega} e^{-\ln 2\left(\frac{v-CS}{\omega}\right)^2}$
  1: CS [mm/s], center shift
  2: ω [mm/s], line width
  3: η, Lorentzian weight
  4: A [mm/s], spectral Area

- **FeAs 1 2 3 4 5**
  FeAs powder spectra, simulated from five SHp-constructors with the parameters ($B_i(T)$, $V_{zz}(T)$, 1, θ_i, 53, $CS(T)$, 0.13, A_i), where $B_i(T) = B_i \cdot (1-(T/69\text{ K})^{2.25})^{0.21}$, $V_{zz}(T) = -(28.96 + (300-T) * 0.0183685614)$, $CS(T) = CS_0 + QuadDop(T, \Theta_D, 57)$, $A_i = A_0 \cdot f_i \cdot AbsDeb(T, \Theta_D, 57)$ and (B_i, θ_i, f_i): (4.16,9,0.21), (0.88,27,0.27), (3.61,45,0.19), (2.52,63,0.17), (1.75,81,0.16). Such model was introduced by Häggström *et al.* [HGSF89].
  1: T [K], temperature, typically this parameter is "rsp"
  2: CS_0 [K], 0 K center shift, isomer shift with respect to room temperature $\alpha - Fe$ should be close to 0.6 mm/s at 0 K
  3: A_0 [mm/s], 0 K spectral area
  4: Θ_D [K], Debye Temperature, used to calculated quadratic Doppler effect end Debye-Waller factor, should be close to 390 K
  5: $T_{N,FeAs}$ [K], Neél temperature of FeAs, should be close to 69 K

3.2 Arguments

Every parameter is of one of the following types:

- # ... inserts the pure object value
- DISTR[#,#,#,#,#] ... DISTR[storage,min,max,steps,weightfunction] stores all values between min and max one after the other into storage and sets them for the parameter simultaneously, weights the resulting spectra with the weight function. This can be folded with DISTR of other arguments.
- DISTR[#,#,#,#,#,#] ... DISTR[storage,min,max,steps,weightfunction,parameterfunction] works the same as DISTR, but additionally calculates the parameter value by parameter function instead of inheriting directly from storage. The storage variable in that case acts as a parametrisation. The following two lines describe equivalent models:

```
SHp DISTR[theta,0,90,31,rho,B] Vzz 0 theta 0 CS omega A0
SHp B Vzz 0 DISTR[theta,0,90,31,rho] 0 CS omega A0
```

- MEM[#,#,#,#] ... MEM[storage,min,max,steps] arranges Maximum Entropy Method for this parameter [SB84]. For construction of MEM-subspectra the MEM-parameter is temporary stored in storage. This MEM can be combined MEM of other arguments. Pay Attention to MEM-command.
- MEM[#,#,#,#,#] ... MEM[storage,min,max,steps,parameterfunction] works the same as MEM[#,#,#,#], but MEM is executed on the storage parameter, whereas the actual argument is calculated with the parameter function. Thus storage may serve as a parametrization of the argument. The following two lines describe equivalent models:

```
SHp MEM[theta,0,90,31,B] Vzz 0 theta 0 CS omega A0
SHp B Vzz 0 MEM[theta,0,90,31] 0 CS omega A0
```

- HR[#,#,#,#] ... HR[storage,min,max,steps] applies a Hesse-Rübartsch distribution [HR74] on this parameter. The numeric approximation of the second derivative is calculated from five points as proposed by Le Cear and Dubois [LCD79]. For construction of HR-subspectra the HR-parameter is temporary stored in storage. Pay Attention to HR-command. The distribution can be constrained to positive values using with the HR-command.
- HR[#,#,#,#,#] ... MEM[storage,min,max,steps,parameterfunction] works the same as HR[#,#,#,#], but HR is executed on the storage parameter, whereas the actual argument is calculated with the parameter function. Thus storage may serve as a parametrization of the argument.

In this notation # stands for one of the following objects:

- a fitparameter name defined in FITPARAMETER block
- a function name defined in FUNCTIONS block
- map, i.e. you state "map0" or "map1" or ... to access the first or second or ... run specific map defined in RUN Block
- a constant number

Function objects are updated just before calculation, after all parameters are deployed.

3.3 Minimum example

A simple iron sextet could be described the following way

```
####################################################################
FITPARAMETER
Bhyp 33.0 0
omega 0.12 0.01
CS 0.2 0.1
I0 10000 100
A 2000 100
####################################################################
THEORY
BL I0
SHp Bhyp 0 0 0 0 CS omega A
####################################################################
RUN
"C:/path/to/data.fld" 1
####################################################################
```

4 RUN

The lists data to be taken into account for the fitting and should contain at least all runs stated in the PLOT block. Every line of the RUN block consists at least of a float, representing the run specific parameter "rsp" (usually temperature, but also field/voltage might be used instead). If only the rsp is stated, than Moessfit will search for a subfolder named like rsp to load the newest *.fld filed contained in. Alternatively the full path or the path relative to the mbs-file can be given in quotation marks, before stating any rsp value. Optionally the rsp is followed by the space separated declaration of x-intervals the fit is restricted to. These intervals are stated in the form "[x0min,x0max] [x1min,x1max] ... [xnmin,xnmax]". Run specific maps – if there are some – follow space separated (type the name of the referring fit parameter/function/map or a number). Keep in mind that all runs have to state the same number of maps. The following example shows a RUN block which reads essentially the same data file but with different rsp value, fit range restrictions and/or maps.

```
#################################################################
RUN
"C:/absolute/path/to/mbs_file_folder/4/newest.fld" 1 fun7 fun2
"../mbs_file_folder/4/newest.fld" 2 fun7 fun2
"4/newest.fld" 3 fun7 fun2
4 fun7 fun2
"4/newest.fld" 5 [-5,-3] [2,3] fun7 fun2
"4/newest.fld" 6 [-5,-3] par4 0
"4/newest.fld" 7 par4 par4
"4/newest.fld" 8 par4 map0
#################################################################
```

The first four runs are identical except for the rsp. Run 5 and run 6 have restricted fit ranges. Run 6, 7, and 8 have different maps compared to the previous runs. fun2, fun7, and par4 have to be declared in the FITPARAMETER or the FUNCTIONS block. Run7 and 8 are identical again – except for the rsp – and demonstrate the use of a map. "map0" refers to the first stated map of this run, which is "par4".

5 FUNCTIONS

The FUNCTIONS block has to list all user defined functions being used in the THEORY block. The block is of the following form:

```
#################################################################
FUNCTIONS
functionname1 = ...
functionname2 = ...
#################################################################
```

Define a function on the "..."-position. The function parser is taken from http://warp.povusers.org/FunctionParser/, so please refer to the page concerning syntax questions. A function can contain objects of every kind (fit parameters, other function from FUNCTIONS block, map parameters ("map#")) and the run specific parameter (i.e. temperature/field/voltage) ("rsp"), which is stated within the RUN block.

The following functions are implemented additionally (simply type the left hand name and the arguments as stated above):

1. $AbsDeb(T,\Theta_D,M_{eff}) = e^{-\frac{\hbar^2 k_\gamma^2}{2M_{eff}}\frac{1}{k_B\Theta_D}\left(\frac{3}{2}+6\left(\frac{T}{\Theta_D}\right)^2\int_0^{\Theta_D/T}\frac{x\,dx}{e^x-1}\right)}$ ... Debye Waller factor of the phonon Debye model, $[T,\Theta_D,M]$=K,K,u; for ^{57}Fe nucleus by default, to change k_γ use "Nucleus" command

2. $FerroMag(B,M_s,J,T,T_c,g)$ ... magnetisation of a ferromagnet, described in mean field theory by Weiss, $[B,M_s,J,T,T_c,g] = T,A/m,-,K,K,-$, with the applied field B, the saturation Magnetisation M_s, the total spin J, the temperature T and transition temperature T_c and the gyromagnetic factor.

3. $QuadDop(T,\Theta_D,M_{eff}) = -\frac{9R}{2N_A M_{eff} c}\frac{T^4}{\Theta_D^3}\int_0^{\Theta_D/T}\frac{x^3 dx}{e^x-1}$ ... quadratic Doppler effect in Debye approximation of heat capacity; $[T,\Theta_D$,M]=K,K,u

4. $SwaveSFD(T,T_c,\lambda_0^{-2},\Delta_0) = \lambda_0^{-2}\left(1+\int_\Delta^\infty \underbrace{-\frac{e^{E/k_bT}}{k_bT(e^{E/k_bT}+1)}}_{\frac{\partial f}{\partial E}}\cdot\frac{E}{\sqrt{E^2-\Delta^2}}\right)$ ... super fluid density in the s-wave model for μSR fitting purpose with $\Delta = \Delta_0\cdot\tanh(1.82\cdot(1.018\cdot(\frac{T_c}{T}-1)^{0.51}))$; $[T,T_c,\lambda_0^{-2},\Delta_0]$=K,K,$\mu$m^{-2},eV

5. $erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ ... Gaussian error function

6. $Hc2WHH(T,T_C,dH_{c2}/dt\mid_{t=1},\alpha,\lambda_{SO})$ ... Temperature dependence of the 2nd critical field of a superconductor within the WHH model [Col86]. $[T,T_C,dH_{c2}/dT\mid_{t=1},\alpha,\lambda_{SO})]$=K,K,G,1. The model has the following restrictions

- transition is of 2nd order
- spin orbit scattering frequency λ_{SO} is much less than normal scattering frequency
- strong coupling is neglected
- Fermi surface anisotropy is neglected
- there is normal state spin (Pauli) paramagnetism
- there is mixed state spin orbit scattering

$$\alpha ... \ Maki \ paraeter \tag{2}$$

$$\lambda_{SO} = \frac{2\hbar}{3\pi k_B T_C \tau_{SO}} ... \ Spin \ orbit \ scattering \ frequency \tag{3}$$

$$t = \frac{T}{T_C} \tag{4}$$

$$\bar{h} = \frac{4}{\pi^2} \cdot h^* \tag{5}$$

$$h^* = \frac{H_{C2}}{-(H_{C2}/dt)_{t=1}} \tag{6}$$

$$\ln\left(\frac{1}{t}\right) = \sum_{n=-\infty}^{n=+\infty} \left(\frac{1}{|2n+1|} - \frac{1}{|2n+1| + \bar{h}/t + \frac{(\alpha\bar{h}/t)^2}{|2n+1|+(\bar{h}+\lambda_{SO})/t}} \right) ...implicit \ WHH - equation \tag{7}$$

7. $NGLsigmaquad(B_{ex}, B_{C2}, \lambda)$ ... Numeric Ginzburg-Landau theory [Bra03] to describe the superconducting damping rate σ_{SC} in an quadratic flux line lattice with critical field B_{C2} [T], applied field B_{ex} and penetration depth λ. $[B_{ex}, B_{C2}, \lambda]$=T,T,$\mu$m

8. $NGLsigmatri(B_{ex}, B_{C2}, \lambda)$ ... Numeric Ginzburg-Landau theory [Bra03] to describe the superconducting damping rate σ_{SC} in an triangular flux line lattice with critical field B_{C2} [T], applied field B_{ex} and penetration depth λ. $[B_{ex}, B_{C2}, \lambda]$=T,T,$\mu$m

9. $OrdParaDistr(T, T_N, \sigma, \alpha, \beta)$ ... $= \frac{1}{\sqrt{2\pi\sigma^2}} \int_T^\infty e^{-\frac{1}{2}\left(\frac{T_N - t}{\sigma}\right)^2} \left(1 - \left(\frac{T}{t}\right)^\alpha\right)^\beta dt$, describes an order parameter model with Gaussian distributed order temperature T_N with standard deviation σ.

6 COMMANDS

- A # ... parameter # is to be overwritten with the automatically determined run specific spectral area
- ConvergenceCriterium # ... sets minimum improvement per step for fitting algorithms. Fitting stops if improvement is insufficient. (by default set to 0.01)
- CosineSmearing # # # # # ... source radius, sample radius, detector radius, source-sample distance, source-detector distance; The finite size of source, sample and detector allows photons to pass the sample to the detector at an angle θ with respect to the axis of source motion. Depending on θ the initial Energy shift v_0 is reduced in the following manner: $v = v_0(1 - \cos\theta)$. Riesenmann *et al.* [RSK69] gave a introduction and a analytical formula for the case of a dominant sample or detector geometry. Moessfit numerically tests the occurrence probability density $\rho(\cos\theta)$ and then smears the raw Mössbauer spectrum according to:

$$I(v_i)dv_i = \sum_j \left[< \rho >_j I(v_i)dv_i) \right] dv_j \quad \text{with} \quad < \rho >_j = dv_j^{-1} \int_{\cos\theta(v_j - dv_j/2)}^{\cos\theta(v_j + dv_j/2)} \rho(\cos\theta')d\cos\theta' \tag{8}$$

- Errors # #... mode, sample count; for mode put one of the following words:
  - Uncorrelated: errors indicate the particular parameter change to increase χ^2 by 1.
  - Hesse: calculates the Hessian matrix and inverts it to gain covariances of every parameter. If the Hessian matrix is not positive definite, the off diagonal terms are scaled towards zero until positive definiteness is achieved. This mode is the default setting.

  - Minos: fixes a parameter an fits the model again to find the most distant parameter leading to a χ^2 increase of 1. The MaxIter-command should be used to set a sufficiently high iteration count, so that the χ^2-minimum indeed can be reached. Otherwise the displayed errors are underestimated.
  - MonteCarlo: samples the parameter space give by the 3σ intervals determined from uncorrelated errors to map the probability density function. The individual density distributions are fitted by a Gaussian distribution giving the standard deviation. Type a sample count as second command argument. It is 10000 by default. As the precision of the Monte Carlo error estimation increases with the sample count, it might be done over night or at any time when computing resources area available. The Monte Carlo error estimation can be stooped an any time (just limiting the quality of the estimation) giving meaningful errors in contrast to the other modes.

- ForceSimplex ... forces Moessfit to use the Nelder-Mead-simplex even in global fits instead of a primitive but faster gradient approach

- I0 # ... parameter # is to be overwritten with the automatically determined run specific baseline

- HR # #... lambda, constraint for Hesse-Rübartsch distributions. To achieve a positive weight constraint the numeric minimization of χ^2 as proposed by Le Cear and Dubois[LCD79] is applied.

- MaxIter # ... maximum Iterations # for one fitting procedure. (by default set to: 1000)

- MEM # # # # #... maximum steps, step size, tolerance, maxlambda' λ_2' and the baseline I_{02} concerning this lambda; by default set to 100, 0.012, 0.1 and 1. The run dependent λ' is calculated by

$$\lambda' = \frac{\sqrt{\frac{I_{02}}{I_0}}\lambda_2'}{1 + \left(\sqrt{\frac{I_{02}}{I_0}} - 1\right)\lambda_2'} \tag{9}$$

  with regard of the run dependent baseline I_0 to achieve run independent smoothness of the MEM-distributions. λ_2' and I_{02} respectively should be taken from the run with the highest statistics. That means first MEM-fit these runs with $\lambda_2' = 1$ to gain the maximum λ the algorithm may achieve. Spectra with lower channel content should achieve an equal smoothness more easily.

- Nucleus # # # # # ... transition energy [keV], gyromagnetic ratio ground state, gyromagnetic ratio excited state, quadrupole moment [barn], mass [u]; changes the properties of the nucleus, if Moessfit should be applied to a nucleus with a $I = 1/2 \leftrightarrow I = 3/2$ transition different from ^{57}Fe. For ^{57}Fe you could type "14.4129 0.09044 -0.1549 0.16 46.993" as argument.

- TransmissionIntegral # # # # ... effective thickness, source line width [mm/s], natural absorber line width [mm/s], resonant fraction; calculates the full transition integral for the specified theory, useful to fit thick absorbers. Keep care that the spectrum has exact symmetric velocity spacing (7 velocity digits at least), otherwise a slow naive convolution will be applied. In this mode you have to set the pure sample line widths as line width, i.e. subtract the source line width from the total line width. The total spectral area argument should be set constant, as the actual area is varied by the resonant fraction f_r and effective thickness[1] t_a only. The meaning of f_r is sketched in fig. 4, it typically takes values between 0.2 and 0.6. It is important to understand that f_r and t_a are highly correlated, because both determine the total absorption area. However, the typical thick absorber effects (line width increase, levelling, saturation effect) are exclusively described by t_a. Therefore it is wise to assume f_r to be temperature independent and associate any change in absorption area with t_a, which is natural because $t_a \propto f_a$ with the temperature dependent absorber Debye Waller factor f_a. The following ways of determining f_r may be considered:

  1. fit both t_a and f_r to a spectra with different but known line intensities, typically to the sextet at low temperatures. Because of the saturation effect the differences in line intensities will be levelled and f_r and t_a can be determined simultaneously.

[1] $t_a = \sigma_0 n_a f_a d$. The nuclear absorption cross section $\sigma_0 = \frac{\lambda^2}{2\pi}\frac{1+2I_e}{1+2I_g}\frac{1}{1+\alpha}$ only depends on the properties of the Mössbauer transition (quantum number I_g and I_e of the ground and excited state, photon wave length λ and conversion factor α). In contrast, the density of Mössbauer nuclei n_a, the Debye-Waller factor f_a of the absorber and the sample thickness d strongly depend on the sample composition, quality and preparation.

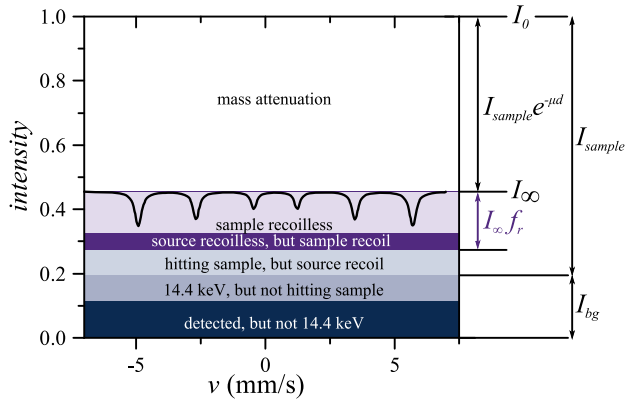


Figure 4: Specrum composition for a 25 µm iron foil measured with a PIN detector.

2. set f_r such that the resulting temperature dependence $t_a(T)$ is a smooth function irrespective of any splitting. For example if the f_r is set too high than t_a will relatively low. If the spectral weight is redistributed to multiple lines e.g. at a magnetic phase transition, than the saturation effect is underestimated and t_a will experience a jump, which is not physical.
3. estimate t_a from the known Mössbauer nucleus phase density. For ^{57}Fe Mössbauer spectroscopy the following formula can be applied $t_a \approx f_a \cdot 0.588 \frac{\mathrm{cm}^2}{\mathrm{mg}} \cdot \frac{m_{Fe}}{A}$ with the total[2] Fe mass m_{Fe} and the face A covered by the absorber.
4. calibrate the spectrometer with absorbers of known total absorption $e^{-\mu d}$ (Compton scattering, photo effect) as described in [Kam16, 55-56]

- UnfoldedData # # #... arguments: velocity, Intensity correction, any constant parameter; recalculates the velocities and adapts intensities. The user might take this command to handle unfolded data and thus assign arbitrary channel-velocity relations and float number fold points. The formula for the velocity calculation should be declared in the FUNCTIONS block. The parameter stated as 3rd argument can be referred in the velocity function as channel. The same accounts for a intensity correction of the raw data. The correction e.g. can account for the change of solid angle of source emission due source movement. If no such correction is desired, type "1". Otherwise one may consider the solid angle from a point source (moving function m) to the detector (distance z, radius r). The intensity correction c_I than is:

$$c_I = \frac{I(m)}{I(0)} = \frac{2\pi \int_0^{\theta(m)} \sin x dx}{2\pi \int_0^{\theta(0)} \sin x dx} = \frac{1-\cos\theta(m)}{1-\cos\theta(0)} \quad \text{with} \quad \cos\theta(m) = \frac{z+m}{\sqrt{(z+m)^2+r^2}}$$

$$\Longrightarrow c_I \approx 1 - \frac{r^2}{(z^2+r^2)^{3/2} - z^3 - r^2 z} m \tag{10}$$

For sinusoidal source movement $m(chn) = v_{max}/(2\pi f) \cdot \sin(2\pi(chn + chn_{fold})/N_{chn})$ with the drive frequency f. The following example shows how the unfold data can be treated.

```
##################################################################
FITPARAMETER
chn 0 0
alpha 0.035 0.001
chnfold 512 0.2
f 25 0
R 5 0
z 100 0
U 209.5 0
Nchn 1024 0
```

[2]i.e. NOT only ^{57}Fe, 2.2 % natural abundance of ^{57}Fe is assumed

```
...
#####################################################################
FUNCTIONS
velocity = vmax*cos((chn+chnfold)/Nchn*2*3.14159)
intensity = 1-(R^{}2)/((z^2+R^2)^(3/2)-z^3-R^2*z)*m
m = vmax / (2*3.15149*f)*sin((chn+chnfold)/Nchn*2*3.14159)
vmax = alpha*U
#####################################################################
COMMANDS
UnfoldedData velocity intensity chn
#####################################################################
RUN
"unfolded_data.txt" 1
#####################################################################
```

7 PLOT

The PLOT block defines the runs to be plotted and the range. For several runs, the channel contents will be normalized to baseline. Plotting is called by pressing "p". The typical PLOT block looks like the following:

```
#####################################################################
PLOT
runs 4,2 103,5 290,2
range -2.5 3.5 0.85 1.04
#####################################################################
```

8 Fitting anything

For general fitting purpose Mössbauer may serve as proper tool. Data to be fit is a *.fld file with three tab separated columns: xdata, ydata, yerror. To fit this data you have to type "fit # #" as the constructor in the THEORY block. First argument is used as the x buffer, i.e. xdata will be loaded into this parameter and a fit function might work upon this parameter. This fit function is stated as second argument. The x buffer should be declared as a constant. The fit function is as usual declared in FUNCTIONS block.

To fit parables with equal curvature a and run specific offset (2.3, 4.7, -4) to three data sets e.g., the concerning *.mbs file could look like the following:

```
#####################################################################
FITPARAMETER
global_a 0.5 0.1
b 3 0.1
x 0 0
#####################################################################
THEORY
fit x fitfunc
#####################################################################
FUNCTIONS
fitfunc = global_a*x*x+b*x + rsp
#####################################################################
RUN
"C:\path\to\file1\data1.fld" 2.3
"C:\path\to\file2\data2.fld" 4.7
"C:\path\to\file3\data3.fld" -4
#####################################################################
```

9 How to do a proper calibration

For the velocity calibration a thick iron foil (25 µm) is measured in sinusoidal mode of the drive. From the pickup coil the an effective induction voltage $U = 203.18\,\mathrm{mV}$ proportional to the maximum velocity v_{max} of the source is known. The calibrations aims for the proportional factor α in $v_{max} = \alpha U$ and eventually the fold channel and the resonant fraction f_r. The following model allows for this task, assuming an hyperfine field of 33 T in iron.

```
#################################################################
FITPARAMETER
B 33.0 0
CS -0.11 0.02
omega 0.08 0.01
I0 1 1
A0 10000 0
chn 0 0
alpha 0.035 0.001
chnfold 512 0.2
fr 0.5 0.05
T 295 0
Nchn 1024 0
#################################################################
THEORY
BL I0
SHp B 0 0 0 0 CS omega A0
#################################################################
FUNCTIONS
velocity = alpha*rsp*cos((chn+0.5+chnfold)/Nchn*2*3.1415)
ta = 11.57*AbsDeb(T,470,57)
#################################################################
COMMANDS
I0 I0
MaxIter 5000
Errors Hesse
TransmissionIntegral ta 0.0515 0.0485 fr
UnfoldedData velocity 1 chn
#################################################################
RUN
"unfolded_data.txt" 203.18
#################################################################
PLOT
runs 203.18
#################################################################
```

10 Maximum entropy method (MEM) and other parameter distribution

The most prominent example of the application of the maximum entropy method (MEM) is the determination of a magnetic hyperfine field distribution from a broadened spectrum. The initial situation of types of arguments listed in sec. 3.2 is basically the same: By the argument DISTR[B,Bmin,Bmax,N,rho], MEM[B,Bmin,Bmax,N], or HR[B,Bmin,Bmax,N] at the argument position of the hyperfine field[3] the program calculates N spectra which differ only by the hyperfine field B. N different but equidistant values for B in the interval $[B_{min},B_{max}]$ are assumed. Usually these N spectra do not contribute equally to the total spectrum but have to be weighted. If the weight function $\rho = \rho(B,...)$ is known the DISTR-argument can be applied by defining the function $\rho(B,...)$ in the FUNCTIONS block. A prominent example is the spin density wave which is supposed to generate the following hyperfine field distribution.

[3] of any constructors listed in sec. 3.1. In most constructors the magnetic hyperfine field is the first argument in the list of arguments

$$\rho(B) = \frac{\pi}{2} \frac{1}{\sqrt{1-(B/B_0)^2}} \quad \forall\ 0 \leq B < B_0 \tag{11}$$

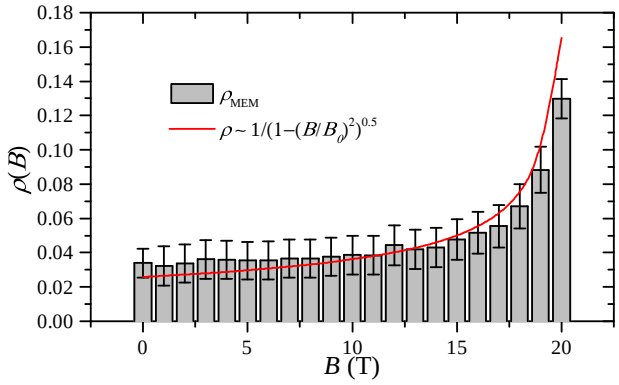
On the other hand, the resulting spectrum may be fitted using maximum entropy or Hesse-Rübartsch method to extract the distribution $\rho(B)$ without knowing its analytical form. In fig. 5 such reconstruction of a simulated spectra is shown. The spectra was simulated using the following mbs file, based on eq. 11 with $B_0 = 20.5\,\mathrm{T}$ with 21 subspectra.

```
######################################################################
FITPARAMETER
B 0 0
######################################################################
THEORY
BL I0
SHc DISTR[B,0,20,21,rho] 0 0 0 0 0 0.12 1000000 0 0
######################################################################
FUNCTIONS
rho = 1/sqrt(1-(B/20.5)^2)
######################################################################
```

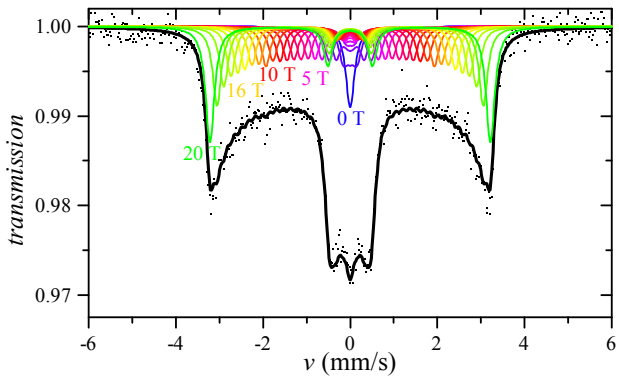
Noise was added to the data using *data processing* → *add noise* accessible from the user interface. Subsequently, this spectrum was fitted using the following model to extract the hyperfine field distribution as shown in fig. 5(a).

```
######################################################################
FITPARAMETER
B 0 0
######################################################################
THEORY
BL 1000000
SHc MEM[B,0,20,21] 0 0 0 0 0 0.12 100000 0 0
######################################################################
COMMANDS
MEM 500 0.013 0.1 0.388 70000
######################################################################
```

In fig. 5(b) it is shown that good fitting is achieved with the 21 subspectra. Their weight roughly reproduce the analytical function (11), with slightly reduced range of values due to the minimization of entropy criterion.



(a) Hyperfine field distribution as simulated and reconstructed using MEM. The simulated spectrum was subject to *data processing* → *add noise* accessible from the user interface.



(b) Constitution of the total MEM spectrum by 21 subspectra

Figure 5: Simulation and reconstruction of magnetic hyperfine distribution with the hyperfine field longitudinal to the gamma beam. For the distribution eq. (11) was assumed with $B_0 = 20.5\,\mathrm{T}$.

Noise was added to the data using *data processing* → *add noise* accessible from the user interface. Subsequently, this spectrum was fitted using the following model to extract the hyperfine field distribution as shown in fig. 5(a).

```
##################################################################
FITPARAMETER
B 0 0
##################################################################
THEORY
BL 1000000
SHc MEM[B,0,20,21] 0 0 0 0 0 0.12 100000 0 0
##################################################################
COMMANDS
MEM 500 0.013 0.1 0.388 70000
##################################################################
```

In sec. 3.2 extended version of argument types are shown: DISTR[x,xmin,xmax,rho,B], MEM[x,xmin,xmax,rho,B], and HR[x,xmin,xmax,rho,B]. The difference now is, that the control variable x and the parameter value B are not the same anymore. They not even have to be related! B usually will be a function of x, but it can be a number, rsp, fit parameter or map – each of them independent of x – as well. In the latter case there is no specific reason to put the DISTR, MEM, or HR command at the B-position of the spectrum constructor. The only important thing to remember is: the first argument after "[" is in any case the parameter, where the value of the control variable is stored. That's why it is called "storage" above. Other parameters may be calculated from the actual value stored in the control variable.

One possible application of this extended distribution argument type is the simulation of Gaussian distributed transition temperatures where the saturation field B_s and the critical exponent β are known. It could be modelled in the following way.

```
##################################################################
FITPARAMETER
Tn 0 0
sigma 0.5 0.1
Tn0 12 1
beta 0.125 0
Bs 20 0
##################################################################
THEORY
BL 1000000
SHp B 0 DISTR[Tn,Tnmin,Tnmax,61,rho,0] 0 0 0 0.12 100000
##################################################################
FUNCTIONS
Tnmin = Tn0 - 3*sigma
Tnmax = Tn0 + 3*sigma
rho = exp(-0.5*((Tn-Tn0)/sigma)^2)
B = if(rsp<Tn,Bs*(1-rsp/Tn)^beta,0)
##################################################################
```

In that example the DISTR argument is placed on the argument position of η, and zero value is assigned to η in any case. I.e., that in this example the position of the DISTR argument is indifferent, as long has the hyperfine field is calculated from the order parameter function B.

11 Example of a complex mbs-file

```
##################################################################
FITPARAMETER
B 0 0
B0 0 00
sigma 1.5 0.2
global_Vzz 12 1
theta 58 5
CS 0.5 0.02
```

```
omega 0.14 0.01
I0 1 1
Abs 0.058535 0.002
global_AbsFe0 0.0609 0.0001
global_B0Fe 33.5 0.1
################################################################
THEORY
BL I0
SHp DISTR[B,Bmin,Bmax,15,rhoB] global_Vzz 0 theta 0 CS omega A0
SHp BFe 0 0 0 0 quadDopllerFe omega AFe
################################################################
FUNCTIONS
rhoB = exp(-0.5*((B-B0)/sigman)^2)
sigman = max(0.01,abs(map0))
Bmin = B0-3*sigman
Bmax = B0+3*sigman
Bsteps = 15
A0 = I0 * Abs
AFe = I0 * AbsFe
AbsFe = global_AbsFe0*AbsDeb(rsp,453.4442,57.56)
BFe = global_B0Fe*(1-(rsp/1043)^2.0811)^0.3358
quadDopllerFe = 0.109 + QuadDop(rsp,453.4442,57.56)
################################################################
RUN
4.0 sigma
"../parentfolder/12.0/data.fld" 12.0 sigma
18.0 sigma
"C:/path/to/file/outside/parent/folder/filename.fld" 19.0 sigma
20.0 sigma
22.0 0
"120.0/data.fld" 120.0 0
320.0 0
################################################################
COMMANDS
I0 I0
################################################################
```

11.1 Explanation

The mbs file should be read starting from THEORY. It describes a baseline "BL" and two powder subspectra in the full static Hamiltonian model ("SHp"). All three components will be summed up to form the total spectrum. The first subspectrum represents a doublet that broadens below 22 K whereas the second aubspectrum describes α-Fe, which was attached to extract experimental line width. This is, why both subspectra share the same line width *omega*.

As the hyperfine parameters of α-Fe are well known and it is only intended to provide the experimental line width *omega*, the arguments of the subspectrum are strongly restricted. The temperature dependence of the magnetic hyperfine field *BFe* is modelled with a two exponent function in the FUNCTIONS block in dependence of the run specific parameter ("rsp"), which refers to the number stated in the RUN block[4], representing temperature in our case. The only free parameter is the 0-K-hyperfine-field *global_B0Fe* to compensate for calibration inaccuracy of the Mössbauer drive. In the same way the area *AFe* and the centre shift *quadDopllerFe* are modelled with Debye models as a function of temperature, with fixed Debye temperature and effective mass. Again the 0-K-absorption *global_AbsFe* and 0-K-centre-shift are the only variables and simultaneously have to account for all runs.

The actual sample is described in the first subspectrum. Only a small broadening was observed below 22 K, which is attributed to magnetism and thus described by an increasing magnetic hyperfine field. Actually the field

[4]Usually in the folder, where the mbs file is located, there should exist subfolders to which the run numbers refer. Moessfit will look into these subfolder to load the newest data file. Alternatively a file path can be given, see sec. 4.

is described as a Gaussian distribution, because instead a simple argument the "DISTR" identifier was put to announce the declaration of a parameter distribution. In the example it describes 15 equally spaced sample values B between $Bmin$ and $Bmax$, whose related subspectra are weighted by the $rhoB$ function. The latter is declared in the FUNCTIONS block as a Gaussian distribution. The normalization of is done automatically, currently the distribution is centred around $B0$=0. The standard deviation $sigman$ could be a simple FITPARAMETER, but here first it is protected against the zero value and second we used the mapping option ("map0" refers to the string stated in the RUN block after the run number), i.e. we forced zero for runs with temperatures $\geq$ 22 K, below the variable $sigma$ is used. Due to the small broadening the field gradient $global\_Vzz$ is assumed temperature independent to not interfere with the magnetic hyperfine field.

12 Programming Moessfit

This section is intended to provide an easy entry point in the Moessfit code, i.e. to sketch the principal program structure. Please consider also the following thesis [Kam16] and the Moessfit-article [KK16] for an introduction in Moessfit. Moessfit is a typical QT application, i.e. in main.cpp a *MainWindow* is generated. The constructor *MainWindow::MainWindow* will generate the two *plot2D* windows *MoessSpectrum* and *EnergySpectrum* and eventually generate a *fitmodel*, either using *MainWindow::start_empty* or in case a mbs-file was given in argv[] using *MainWindow::load_from_mbs*. Please note: The current fit model is stored in *MainWindow.cfm*. This structure can be relived in fig. 6. In the following section the *fitmodel* class is explained in detail.

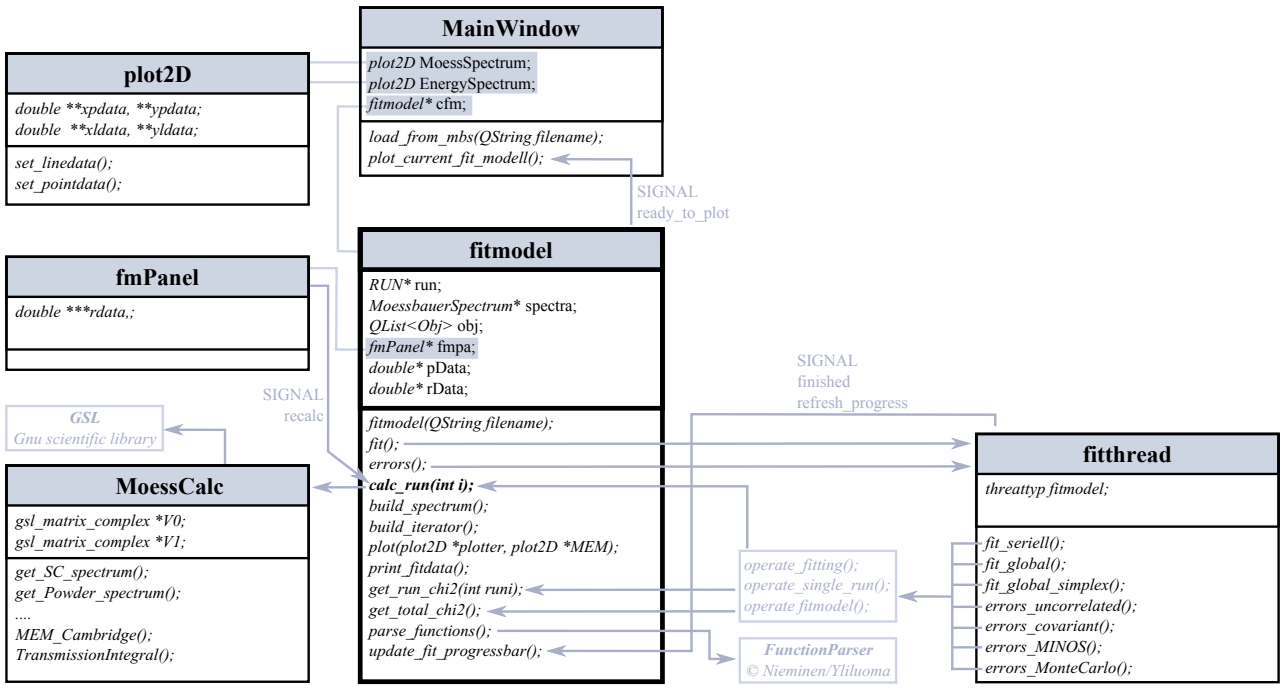


Figure 6: Important relations of Moessfit classes (rectangles), its important attributes (upper rectangle) and functions (lower rectangle), and external dependencies.

12.1 *fitmodel* class

The instantiation of a *fitmodel* using the constructor *fitmodel::fitmodel(QString filename)* has the following goals:

- conistency check of the mbs file (*read_commands, read_fitparameters, read_theory, read_functions, read_run, read_plot*)

- create model specific container (*link, process_theory*), as sketched in fig. 7
- create a model specific user interface ("fit model panel") *fmpanel* and connect it to the fitmodel

Only a valid mbs-file allows for any further user input. The principal user inputs are the fitting and the error calculation, which are both outsourced in the *fitthread* class. A *fitthread* recognizes its task (fitting or error calculation, global or serial) by the constructor *fitthread::fitthread* it is instantiated with. To perform the required calculations the *fitthread* must be provided with a χ^2-function, which are the global functions *operate_fitmodel*, *operate_single_run* and *operate_fitting*. These global function make the *fitmodel* recalculating the spectra by eventually call *fitmodel::calc_run* and inquire the updated χ^2 values and provide *fitthread* with these values. The *fitthread* itself directly modifies the parameter container $fitmodel.pData$, so that χ^2 can change at all.

Furthermore *fitmodel* will plot data with the public function *fitmodel::plot*. *fitmodel::plot* should be called by *MainWindow* after receiving the *fitmodel::ready_to_plot*-signal from *fitmodel*, and it should not be called by fitmodel itself. *fitmodel* is also responsible to update fit- and error-data in the mbs-file using *fitmodel::print_fitdata*.

12.2 *fitmodel::calc_run* function

The *fitmodel::calc_run* organizes the calculation of the spectra and therefore can be considered as the most important function of the *fitmodel* class. In general the theory of only a single run is calculated to avoid redundant calculation whilst fitting. The order of routines executed in *calc_run* is sophisticated and should not be changed cluelessly:

1. if necessary calculate velocity data
2. if necessary change the properties of the nucleus
3. calculate all non-MEM subspectra (actually MEM or Hesse Rübartsch ("HR"))
4. if necessary apply MEM/HR to the rest of the spectrum
5. if necessary apply cosine-smearing
6. if necessary apply the transmission integral
7. calculate the new χ^2

An complex application of types of arguments (sec. 3.2) within the argument list of the THEORY constructor (sec. 3.1) is enabled by the recursively called *fitmodel::build_spectrum / build_iterator* function. *fitmodel::build_iterator* will – as soon as the recursion reached the final step – acquire the specified spectrum. The calculation of the spectrum is outsourced to the *MoessCalc* class. The *MoessCalc* class offers a range of public functions *MoessCalc::get_#_spectrum(double *I, int cc , double* v, double* p)*, which take an array of parameters *p*, an array of velocities *v*, the channel count *cc* and the desired array *I* for the output of the array. The *p* is filled according to the order of arguments as given by the user in the mbs-file and should be interpreted like that in any *MoessCalc::get_#_spectrum* function.

12.3 How to implement a new spectrum constructor

The implementation of a new spectrum constructor for the THEORY block has the goal to enable a correct parsing in *read_theory*, program the calculation in a *MoessCalc::get_#_spectrum(double *I, int cc , double* v, double* p)* function and to call that function in *fitmodel::build_spectrum*. Therefore consider the following steps to introduce a new spectrum constructor:

1. write down and test the new *MoessCalc::get_#_spectrum(double *I, int cc , double* v, double* p)* function
2. declare a new Mössbauer spectrum type (*MSTyp*) in fitmodel.h
3. specify the parameter count in the *fitmodel::pc(MSTyp m)* function
4. introduce the new constructor in *fitmodel::read_theory*, adopt the structure given for other constructors, e.g. "SHp", i.e. first check for the desired string, then check the correct number of arguments, and then specify the *MSTyp* of the *fitmodel.spectra[]*.
5. call the new MoessCalc-function in *fitmodel::build_iterator* below the comment "// now the spectrum has to be calculated". Specify, which parameter contains the spectral area!

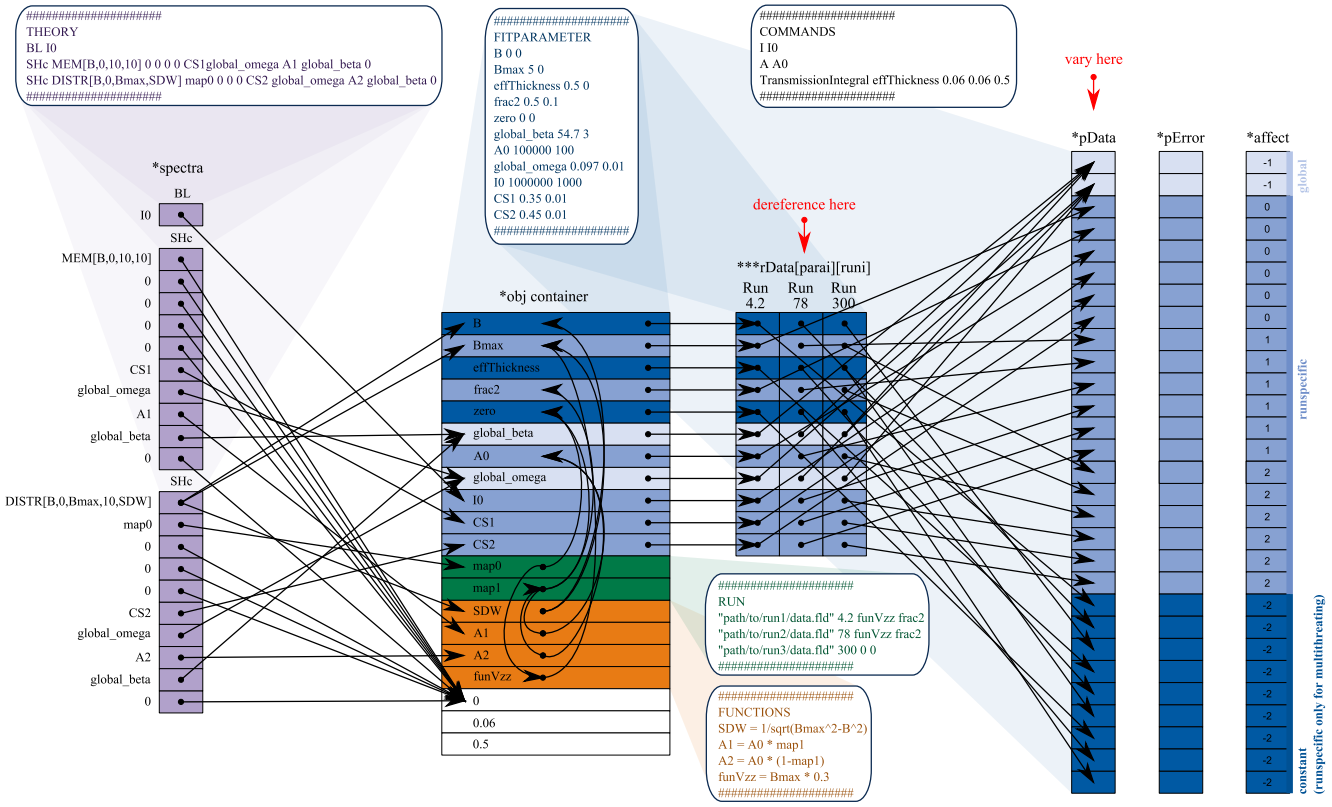


Figure 7: Parsing and internal mbs representation. The obj-container is fed by FITPARAMETER, FUNCTIONS and RUN block. The arguments of the spectra in the THEORY block are linked against this obj-container. The model describes a complex two component fit with global parameters, functions, maps, user defined parameter distributions ("DISTR") and Maximum Entropy method.

12.4 mbs Parsing

Moessfit offers more flexibility concerning the usage of parameters, functions and maps then musrfit. For that reason I like to draw attention on the parsing process of the mbs file, which carries all information. Wherever an argument is expected in the mbs script, the user may type any string identifier (in fact there are few exceptions). After the script once was read in and there was no syntax error in the mandatory FITPARAMETER, THEORY and RUN blocks, the interpretation will proceed by linking the string identifiers against the so called *fitmodel::obj*-container. The *obj*-container was fed before with objects by the reading of the FITPARAMETERS, FUNCTIONS and RUN block. I.e. the *obj*-container carries the semantic information of certain string identifiers: either fit parameter, functions, maps, rsp (run specific parameter, usually temperature or field) or a simple number. The latter are established, if the string identifier could not be found, but can be interpreted as a number.

After a successful linking the internal representation of the *fitmodel*, i.e. of the mbs file, is basically complete. If a spectrum has to be calculated, the argument objects are evaluated using the function *fitmodel::eval*. In the internal representation each argument is the index of an object in the *obj*-container. The evaluation of its value may depend on the run. Therefore the related run number *runi* must always be supplied. The use of the intermediate *fitmodel::eval* function allows for a nested calling. In doing so, a evaluation covering several levels is possible, because a map can lead to a function using other functions and so on.

In fig. 7 the parsing and interpretation process is sketched.

References

[Bha98] Bhat, MR: Nuclear data sheets for A= 57. In: *Nuclear Data Sheets* 85 (1998), Nr. 3, S. 415–536

[Blu68] Blume, M: Stochastic theory of line shape: generalization of the Kubo-Anderson model. In: *Physical Review* 174 (1968), Nr. 2, S. 351

[Bra03] Brandt, Ernst H.: Properties of the ideal Ginzburg-Landau vortex lattice. In: *Physical Review B* 68 (2003), Nr. 5, S. 054506

[Chu11] Chuev, MA: Multi-level relaxation model for describing the Mössbauer spectra of single-domain particles in the presence of quadrupolar hyperfine interaction. In: *Journal of Physics: Condensed Matter* 23 (2011), Nr. 42, S. 426003

[Cla71] Clauser, MJ: Relaxation Effects in Spectra: Eigenvalue Treatment of Superoperators. In: *Physical Review B* 3 (1971), Nr. 11, S. 3748

[Col86] Collings, EW: *Applied Superconductivity, Metallurgy, and Physics of Titanium Alloys: Volume 1: Fundamentals*. Bd. 1. Springer, 1986

[CY07] Chen, Yi-Long ; Yang, De-Ping: *Mössbauer effect in lattice dynamics: experimental techniques and applications*. John Wiley & Sons, 2007

[DBS95] Dufek, Philipp ; Blaha, Peter ; Schwarz, Karlheinz: Determination of the nuclear quadrupole moment of 57 Fe. In: *Physical review letters* 75 (1995), Nr. 19, S. 3545

[HGSF89] Häggström, Lennart ; Gustavsson-Seidel, A ; Fjellvåg, Helmer: A Mössbauer study of helimagnetic FeAs. In: *EPL (Europhysics Letters)* 9 (1989), Nr. 1, S. 87

[HR74] Hesse, J ; Rubartsch, A: Model independent evaluation of overlapped Mossbauer spectra. In: *Journal of Physics E: Scientific Instruments* 7 (1974), Nr. 7, S. 526

[Kam16] Kamusella, Sirko: *Electronic phase diagrams and competing ground states of complex iron pnictides and chalcogenides -A Mössbauer spectroscopy and muon spin rotation/relaxation study*, TU Dresden, Diss., 2016

[KK16] Kamusella, Sirko ; Klauss, Hans-Henning: Moessfit. In: *Hyperfine Interactions* 237 (2016), Nr. 1, S. 1–10

[LCD79] Le Caer, G ; Dubois, JM: Evaluation of hyperfine parameter distributions from overlapped Mossbauer spectra of amorphous alloys. In: *Journal of Physics E: Scientific Instruments* 12 (1979), Nr. 11, S. 1083

[Pyy08] Pyykkö, Pekka: Year-2008 nuclear quadrupole moments. In: *Molecular Physics* 106 (2008), Nr. 16-18, S. 1965–1974

[RSK69] Riesenman, R ; Steger, J ; Kostiner, E: Cosine effect in Mössbauer spectroscopy involving a source of non-zero radius. In: *Nuclear Instruments and Methods* 72 (1969), Nr. 1, S. 109–110

[SB84] Skilling, John ; Bryan, RK: Maximum entropy image reconstruction: general algorithm. In: *Monthly notices of the royal astronomical society* 211 (1984), Nr. 1, S. 111–124

[SD76] Shenoy, GK ; Dunlap, BD: Mössbauer relaxation line shapes in the presence of complex hyperfine interactions. In: *Physical Review B* 13 (1976), Nr. 3, S. 1353

[SW12] Suter, A. ; Wojek, B.M.: Musrfit: A Free Platform-Independent Framework for μSR Data Analysis. In: *Physics Procedia* 30 (2012), Nr. 0, S. 69 – 73. – 12th International Conference on Muon Spin Rotation, Relaxation and Resonance (μSR2011)