WKM
Einführung und technische Referenz

— Version 1.0 —

Wolfgang Kopmann
imnf
TU Braunschweig

Ausgabe vom 1. Februar 2001

Inhaltsverzeichnis

Einleitung

WKM-Fit ist ein interaktives Programm zum Auswerten von Datensätzen, die mit Methoden zeitdifferentieller μSR gewonnen wurden. Dem Programm zugrunde liegt MINUIT [1].

Das Programm wurde auf einer IBM RS6000 unter AIX4-Unix in C++ entwickelt. Der Entwicklung lag ein auf einem IBM 3090 System existierende Variation des Vancouver-Fitprogramms (verändert von Heiko Pinkvoss, Hans-Hennig Klauß und Mauricio de Melo). Es wurde darauf geachtet den Code möglichst plattformunabhängig zu halten um nicht wie es bei dem alten Progamm der Fall war an eine Maschine gebunden zu sein. Für die graphische Ausgabe wurde die X11-Oberfläche gewählt, so daß eine Portierung des Pakets auf alle Plattformen, die eine X11-Fenster verarbeiten können, möglich erscheint. Es existieren bereits Portierungen auf VMS, Linux und OS/2.

1. Inhalt des Programmpakets

Das Programmpaket besteht aus folgenden Dateien:

`wkmfit` Das eigentliche Fitprogramm. Es bestimmt die für MINUIT notwendige FCN-Routine und führt den χ^2-Fit durch.

`wkmview` Interaktives graphisches Interface zum Darstellen der Fitergebnisse. Ebenfalls implementiert ist eine einfache FFT-Routine.

`wkmt0` Interaktives graphisches Programm zum Einstellen der Zeitfenster für Untergrund und Daten, sowie der Bestimmung des t_0-Kanals.

`wkmpara` Programm zum bequemen Erzeugen von Inputfiles für Plotprogramme (GNU-Plot, PHYSICA, ...).

`wkmxmgr` Programm zum Erzeugen von Datendateien für das Plotprogramm xmgr.

`start2msr` Konvertierungsprogramm für Fitparameter-Dateien vom alten `.start`-Format in das neue `.msr`-Format.

`msr2start` Rückkonvertierungsprogramm. Funktioniert nicht vollständig (bei der Rückkonvertierung gehen die Information über die Fittheorien verloren).

`glf.tbl` Binäre Tabellen für statische Gauß Kubo-Toyabe Relaxationsfunktion.

`llf.tbl` Binäre Tabellen für statische Lorenz Kubo-Toyabe Relaxationsfunktion.

`kdglf.tbl` Binäre Tabellen für dynamische Gauß Kubo-Toyabe Relaxationsfunktion.

`kdllf.tbl` Binäre Tabellen für dynamische Lorenz Kubo-Toyabe Relaxationsfunktion.

`wkm-mode.el` E-Lisp-Macro für den Emacs zu Steuerung des WKM-Pakets.

Die ausführbaren Programme sollten im Suchpfad stehen (unter UNIX z.B. /usr/local/bin).

Die Tabellen und Daten werden zunächst im aktuellen Verzeichnis gesucht, zumeist ist es jedoch sinnvoll zumindest die Tabellen an einem zentralen Platz zu lagern, besonders wenn das Programm von mehreren Nutzern eines Systems verwendet werden soll. Dann sollten sie an einen speziellen Pfad gebracht werden wo sie das Programm findet. Dieser ist jedoch von System zu System unterschiedlich (vgl. Tabelle 1.1 oder Anhang A).

System	Datenpfad	Tabellenpfad
exp3	/usr/local/exp3/MUSR/DATA/*format/type/year*	/usr/local/exp3/MUSR/TAB
Linux	/home/MUSR/DATA/*format/type/year*	/home/MUSR/TAB
VMS	wkmdata$dir	wkmtab$dir

Tabelle 1.1: Pfade für Daten und Tabellen auf verschiedenen Systemen

Das E-Lisp-Macro `wkm-mode.el` sollte mit dem Emacs compiliert und anschiessend in den `load-path` des Emacs gebracht werden. Die folgenden Zeilen müssen der persönlichen `.emacs` Datei oder der globalen `default.el` angefügt werden.

```
(setq auto-mode-alist (cons '("\\.msr\\'" . wkm-mode) auto-mode-alist))
(autoload 'wkm-mode "wkm-mode")
```

2. Bedienung

Das WKM-Paket besteht aus mehreren Programmen wie schon in Abschitt ?? aufgelistet. Gesteuert wird es durch eine Steuerdatei die alle notwendigen Informationen zum Fitten enthält. Die genaue Struktur dieser Datei wird in Abschnitt 3.1 beschrieben. Wird diese Datei als Parameter auf der Kommandozeile übergeben, so muß die Endung `.msr` weggelassen werden.

2.1 `wkmfit`

`wkmfit` ist das eigentliche Fitprogramm. Es verlangt auf der Kommandozeile die Steuerdatei des Fits als Parameter. Die Steuerdatei wird eingelesen, der darin beschriebene Fit ausgeführt, und die Ergebnisse in eine Datei mit der Endung `.mlog` geschrieben. Diese `.mlog` Datei hat das gleiche Format wie die Steuerdatei, nur das die Fitparameter die Fitergebnisse enthalten.

Eine Ausführliche Beschreibung der Steuerdatei befindet sich in Abschnitt 3.1.

2.2 `wkmview`

Dieses Programm stellt die Fitergebnisse graphisch dar. Als Kommandozeilen Parameter verlangt es die Steuerdatei und optional einen Ausgabedrucker (postscript).

Das Programm erzeugt ein X-Fenster in dem die Fitergebnisse dargestellt werden. Das Fenster teilt sich auf in einen Datenbereich (eingerahmt von einem Koordinatensystem) und einen Parameterbereich außenherum. In dem Datenbereich kann mit der Maus gezoomt werden. Dazu klickt man die Maustaste 1 (i.A. die linke Maustaste), hält die Taste gedrückt, wählt den gewünschten Zoombereich aus und läßt die Maustaste wieder los. Mit einem Klick auf Maustaste 2 gelangt man zurück zum urspünglichen Zoombereich.

Zudem sind folgende Tasten mit Funktionen belegt:

`q` **Quit** bendet `wkmview`,

`d` **Difference** zeichnet die Abweichung der Datenpunkte von der angepas-

sten Theoriefunktion. Mit erneutem betätigen von [d] kehrt man zur ursprünglichen Darstellung zurück.

[f] **Fourier** berechnet eine FFT[1] der Datensätze im Fenster, sowie der zugehörigen Theorien. Die Daten werden mit einer Baseline-Korrektur um die Abzisse zentriert und gaußförmig apodisiert. Die Apodisierungdämpfung ergibt sich aus dem rechten Rand der derzeitigen Darstellung $\sigma_A = \frac{2}{t_{max}}$. Transformiert wird jeweils der Teil der Datensätze, die im Fensterausschnitt dargestellt sind. Für die Transformation der Theorie wird ein Datensatz gebildet mit den gleichen Stützstellen, wie die Meßdaten.

Im Fouriermodus kann zwischen verschiedenen Darstellungen gewählt werden:

[b] **Betrag** Betrag der Fourieramplitude,

[i] **Imaginär** Imaginärteil der Fourieramplitude,

[r] **Real** Realteil der Fourieramplitude,

[w] **Phase** Phase der Fourieramplitude.

Mit erneuter Betätigung der Taste [f] beendet man den Fouriermodus.

[g] **Gauß** Zeigt die Gauß-Apodisierungskurve für die Fourier-Funktion.

[a] **All** Normalerweise ist die Anzahl der Punkte die im Datenfenster dargestellt werden begrenzt. Mit dieser Taste kann hin und hergeschaltet werden zu einer Dastellungsweise wo alle Datenpunkte dargestellt werden.

[S] **Save** Speichert die gerade dargestellten Daten in eine Datei. Im Zeitspektrum werden für jeden Datenpunkt *Zeit*, *Datenwert*, *Fehler* und *Theoriewert* gespeichert, im Fourierspektrum *Frequenz*, *Absolutwert*, *Phase*, *Realteil* und *Imaginärteil der Fourieramplitude*. Das erste Histogramm wird unter dem Namen `wkm1.dat` gespeichert, etwaige weitere Histogramme unter den Namen `wkm2.dat` ... usw.

[p] **Print** Schwarz-Weiß-Ausgabe der Darstellung auf den Standarddrucker bzw. den als zweiten Parameter übergebenen Drucker.

[e] **Eps** Schwarz-Weiß-Ausgabe des Datenbereichs des X-Fensters in die Datei `wkm.eps` im Encapsulated-Postscript Format.

[1] Transformiert wird mit einem einfachen Cooley-Tukey Algorithmus [3]

[m] **Metafile** Schwarz-Weiß-Ausgabe der Darstellung wie unter [p] nur in die Datei `wkm.ps`.

[P] **Print Color** Ausgabe wie [p] nur in Farbe.

[E] **Eps Color** Ausgabe wie [e] nur in Farbe.

[M] **Metafile Color** Ausgabe wie [m] nur in Farbe.

2.3 `wkmt0`

Mit diesem Programm können auf einfache Weise t_0-Kanäle, Daten- und Backgroundbereiche festgelegt werden. Als Kommandozeilen Parameter wird die Steuerdatei verlangt. Sie wird entsprechend der Modifikationen die vorgenommen werden verändert und überschrieben.

Das Zoomen funktioniert wie beim `wkmview`. Beendet wird das Programm mit der Taste [q].

Modifikationen werden wie folgt vorgenommen: Erst wäht man mit der Tatastur eine Marke aus und klickt anschließend mit der Maustaste 1 auf den Datenpunkt auf den diese Marke gesetzt werden soll.

Folgende Marken können gesetzt werden.

[t] t_0-Kanal,

[b] erster Background-Kanal,

[B] letzter Background-Kanal,

[d] erster Daten-Kanal,

[D] letzter Daten-Kanal.

Der erste Daten-Kanal sollte nicht vor den t_0-Kanal gelegt werden, da es sonst Probleme mit der Fourieranalyse im `wkmview` geben kann.

2.4 `wkmpara`

Sollen die Ergebnisse mehrer Fits dargestellt werden, so benötigen gängige Plotprogramme wie z.B. Gnuplot Eingabedatendatein in denen die Daten

Spaltenweise angeordnet sind. Um solche Dateien zu erzeugen kann `wkmpara` benutzt werden. Als Komandozeilenparameter wird eine Datei verlangt in der die Steuerdateien (ohne die Extension `.msr`) aufgelistet sind, aus denen die Daten extrahiert werden sollen. Die Liste muß so aussehen, daß am Anfang einer Zeile die Steuerdatei steht und anschließend durch Leerzeichen getrennt Parameter die nicht aus den Fitparametern hervorgehen aber dennoch zur Darstellung benötigt werden wie z.B. die jeweilige Temperatur. Die nächste Steuerdatei sollte dann in der nächsten Zeile stehen. `wkmpara` hängt an diese Zeilen die Fitparameter und ihre Fehler durch Leerzeichen getrennt an, und schreibt sie in eine Datei deren Name der der Liste ist mit einem zusätzlichen `.dat` angehängt.

Aus der Datei `liste`:

```
messung-bei-50K 50
messung-bei-100K 100
```

würde somit die Datei `liste.dat`

`messung-bei-50K 50` *par1 err1 par2 err2 ...*
`messung-bei-100K 100` *par1 err1 par2 err2 ...*

2.5 wkmxmgr

Mit `wkmxmgr` können spezielle Datensätze für das Plotprogramm xmgr [2] erstellt werden. Der Syntax der Eingabedatei, die auf der Komandozeile übergeben wird ist:

`@` *grafNr setNr setType*
steuerdatei parameter
steuerdatei parameter
.
.
`@` *grafNr setNr setType*
.
.

grafNr, setNr Geben jeweils die Graphennummer und die Datensatznummer an, die später von `xmgr` verwendet wird.

setType gibt an wie *parameter* interpretiert werden.

`xy` Als *parameter* werden die Nummern der Fitparameter für x und für y erwartet. Der erzeugte Datensatz hat das xmgr-Format `xydxdy`.

`Ty` Als *parameter* wird die Nummern des Fitparameter für y erwartet. Als x-Parameter wird die Temperatur des ersten Runs der jeweiligen Steuerdatei verwendet. Der erzeugte Datensatz hat das xmgr-Format `xydy`.

`xT` Wie `Ty` nur das diesmal x erwartet wird und y die Temperatur ist. Der erzeugte Datensatz hat das xmgr-Format `xydx`.

`Pxy` Als x und y Parameter werden zwei Ausdrücke verwendet die vom Parser erkannt werden können (vgl. Abschnitt 3.1.4). Die Ausdrücke werden durch Leerzeichen getrennt. Der erzeugte Datensatz hat das xmgr-Format `xy`.

steuerdatei Name der Steuerdatei ohne Extension `.msr`.

parameter Parameter wie unter *setType* beschrieben.

Beispiel einer Eingabedatei für `wkmxmgr`:

```
@ 0 0 Pxy
steuerfile1 1 (1-par6)
steuerfile2 2 par4*(1-par5)*(1-par6)
@ 1 0 Ty
steuerfile1 8
steuerfile2 8
steuerfile3 8
```

Die Erzeugte Datei erhält die Endung `.set` und kann von `xmgr` mit File↦Read↦Sets... eingelesen werden.

2.6 Emacs mit `wkm-mode`

Wird in Emacs eine Datei mit der Endung `.msr` geöffnet, erscheint in der Menue-Zeile ein neues Pull-Down-Menue (`MSR-Fit`), über das die Fitfunktionen erreicht werden können. Nach dem Fitten wird die so erzeugte `.mlog` Datei automatisch mit der `.msr` Datei getauscht, so daß die neuen Ergebnisse gleich im aktuellen Fenster zu sehen sind. Die beiden Kommandos `Fit-File` und `Fit-Buffer` unterscheiden sich darin, daß bei `Fit-Buffer` mit temporären Dateien gearbeitet wird und so die ursrüngliche `.msr` Datei erst überschrieben wird wenn das Emacs-Kommando zum Speichern gegeben wird, währernd bei `Fit-File` die Datei gleich vorm Fitten gespeichet wird.

3. Dateiformate

3.1 Die Steuerdatei (`.msr`)

Die Steuerfiles des WKM-Pakets tragen die Extension `.msr`. Sie sind aus verschiedenen Blöcken aufgebaut. Jeder Block (außer der ersten Zeile wird durch ein Kennwort einleitet, sowie durch eine Leerzeile abgeschlossen.[1] Kommentarzeilen werden durch ein vorangegestelltes # gekennzeichnet.

3.1.1 Die Titelzeile

In der ersten Zeile eines Inputfiles können Informationen in beliebigem Format geschrieben werden. Sie erscheinen bei der graphischen Darstellung des Fits als Überschrift.

```
DUMMY-MESSUNG (Ag at 300K B = 50G) ...
```

3.1.2 Der `FITPARAMETER`-Block

Hier werden die gewünschten Fitparameter in der für MINUIT üblichen Form eingeführt. Verlangt werden die Nummer des Parameters, sein Name[2], der Startwert, die Schrittweite und falls erforderlich die untere und obere Grenze des Parameters.

```
FITPARAMETER
#       Nr. Name        Value     Step      Bounderies
          1 ALPHA       1.087     0.01012   0         1.8
          2 PHASE       18.49     2.804     -360      360
          3 FREQ        0.6983    0.003484  0         99.9
          4 RATE        0.02582   0.02216   0         10
          5 ASY         0.2037    0.01032   0         1
```

[1] Zur Zeit reagiert das Programm noch sehr empfindlich auf Formatierungsfehler im Input-File.

[2] Im Parameternamen sind keine Leerzeichen erlaubt, seine Länge ist auf 10 Zeichen begrenzt.

3.1.3 Der `THEORY`-Block

In diesem Block wird die gewünschte Theoriefunktion spezifiziert mit der die Daten gefittet werden sollen. Folgende Theoriefunktionen stehen zur Zeit zur Verfügung (eine explizite Darstellung und Spezifizierung der nötigen Parameter findet sich in Tabelle 3.1)

`asymmetry` Asymmetry,

`simplExpo` Simple Exponential,

`generExpo` General Exponential,

`statGssKT` Static Gauß Kubo Toyabe,

`makusWebe` Makus Weber Function (noch nicht implementiert),

`spinGlas` Spinglas,

`statKTTab` Static Kubo Toyabe Table (`glf`,`llf`),

`dynmKTTab` Dynamic Kubo Toyabe Table (`kdglf`,`kdllf`),

`rdAnisoHf` Random Anisotropic Hyperfinefield,

`combiLGKT` Lorenz Gauß Kubo Toyabe,

`simpleGss` Simple Gauß Funktion[3].,

`abragam` Abragam Function[4]),

`internFld` Internal Field,

`internBsl` Internal Incomensurable Field.

Jede der Theoriefunktionen muß in einer eigenen Zeile stehen. Am Anfang der Zeile muß jeweils der Bezeichner oder das Kürzel der Theoriefunktion stehen, gefolgt von den Parametern. Stehen die verschiedenen Theorien direkt aufeinander folgenden Zeilen, so werden die Theorien miteinander multipliziert. Eine Zeile die nur ein `+` als erstes Zeichen enthält bewirkt eine Addition der vorherigen und folgenden Theoriefunktionen.

[3] In den Gaußfunktionen wird der Parameter σ verwendet. Manchmal wird in der Literatur auch Δ benutzt wobei gilt: $\Delta^2 = \frac{\sigma^2}{2}$.

[4] Es gilt $\gamma = \frac{1}{\tau_c}$ wobei τ_c die Korelationszeit ist. Daher entspricht γ nur ungefähr der mittleren Hüpfrate (hopprate) (vgl. [5].

Bezeichner	Kürzel	Parameter		Formel
`asymmetry`	`a`	$A[1]$		A
`simplExpo`	`se`	$\lambda[\mu s^{-1}]$		$e^{-\lambda t}$
`generExpo`	`ge`	$\lambda[\mu s^{-1}]$,	$\beta\,[1]$	$e^{-(\lambda t)^{\beta}}$
`statGssKT`	`stg`	$\sigma\,[\mu s^{-1}]$		$\frac{1}{3}\left[1+2(1-\sigma^2t^2)e^{-\frac{\sigma^2t^2}{2}}\right]$
`spinGlas`	`spg`	$\lambda[\mu s^{-1}]$,	$\gamma\,[\mu s^{-1}]$,	$\frac{1}{3}\left[e^{-\sqrt{\frac{4\lambda^2(1-q)t}{\gamma}}}\right.$
		$q\,[1]$		$\left.+2\left(1-\frac{(\lambda t)^2}{\sqrt{\frac{4\lambda^2(1-q)t}{\gamma}-(\lambda t)^2}}\right)e^{-\sqrt{\frac{4\lambda^2(1-q)t}{\gamma}-(\lambda t)^2}}\right]$
`statKTTab`	`sktt`	$\nu\,[MHz]$, *tabelle*	$\sigma\,[\mu s^{-1}]$,	statischer Kubo-Toyabe (tabelliert)
`dynmKTTab`	`dktt`	$\nu\,[MHz]$, $\gamma\,[\mu s^{-1}]$,	$\sigma\,[\mu s^{-1}]$, *tabelle*	dynamischer Kubo-Toyabe (tabelliert)
`rdAnisoHf`	`rahf`	$\nu\,[MHz]$,	$\lambda[\mu s^{-1}]$	$\frac{1}{6}\left(1-\frac{\nu t}{2}\right)e^{-\frac{\nu t}{2}}+\frac{1}{3}\left(1-\frac{\nu t}{4}\right)e^{-\frac{\nu t+2.44949\lambda t}{4}}$
`combiLGKT`	`lgkt`	$\lambda_L\,[\mu s^{-1}]$,	$\lambda_G\,[\mu s^{-1}]$	$\frac{1}{3}\left(1+2\left(1-(\lambda_G t)^2-\lambda_L t\right)e^{-\frac{(\lambda_G t)^2}{2}-\lambda_L t}\right)$
`TFieldCos`	`tf`	$\varphi\,[^\circ]$,	$\nu\,[MHz]$	$cos\left(2\pi\nu t+\frac{\pi\varphi}{180}\right)$
`simpleGss`	`sg`	$\sigma\,[\mu s^{-1}]$		$e^{-\frac{1}{2}(\sigma t)^2}$
`abragam`	`ab`	$\sigma\,[\mu s^{-1}]$,	$\gamma\,[\mu s^{-1}]$	$e^{-\left(\frac{\sigma}{\gamma}\right)^2\left(e^{-\gamma t}-1+\gamma t\right)}$
`internFld`	`if`	$\varphi\,[^\circ]$, $\lambda_T\,[\mu s^{-1}]$,	$\nu\,[MHz]$, $\lambda_L\,[\mu s^{-1}]$	$\frac{2}{3}cos\left(2\pi\nu t+\frac{\pi\varphi}{180}\right)e^{-\lambda_T t}+\frac{1}{3}e^{-\lambda_L t}$
`internBsl`	`ib`	$\varphi\,[^\circ]$, $\lambda_T\,[\mu s^{-1}]$,	$\nu\,[MHz]$, $\lambda_L\,[\mu s^{-1}]$	$\frac{2}{3}j_0\left(2\pi\nu t+\frac{\pi\varphi}{180}\right)e^{-\lambda_T t}+\frac{1}{3}e^{-\lambda_L t}$

Tabelle 3.1: Theoriefunktionen des WKM-Pakets und ihre explizite Darstellung.

Als Parameter wird normalerweise direkt die Nummer des Fitparameters angegeben, die im Block `FITPARAMETER` definiert worden ist. Die Reihenfolge der Parameter geht aus Tabelle 3.1 hervor. Z.B. wird mit

```
simplExpo     4
```

eine Exponentialfunktion definiert, bei der als Dämpfung λ der Fitparameter Nummer 4 benutzt wird.

Weitere Zeichen in den einzelnen Zeilen werden von dem Programm als Kommentar interpretiert. In der Ausgabedatei (`.mlog`) fügt das Programm hier einen Komentar zur Erläuterung der Parameter ein.

Durch folgenden `THEORY`-Block:

```
THEORY
asymmetry     5
simplExpo     4
TFieldCos     2     3
+
asymmetry     6
simplExpo     7
TFieldCos     8     9
```

wird die Theoriefunktion

$$A = P_5 \cdot e^{-t \cdot P_4} \cdot \cos(2\pi P_3 t + \frac{\pi}{180} P_2) + P_6 \cdot e^{-t \cdot P_7} \cdot \cos(2\pi P_9 t + \frac{\pi}{180} P_8)$$

definiert.

Werden verschiedene Runs gleichzeitig gefittet, kann es sinnvoll sein bei gleicher Theorie für die verschieden Runs einzelne Parameter unterschiedlich zu belegen, z.B. beim gemeinsamen Fit von Entkopplungen bei verschiedenen Magnetfeldern. Hierzu besteht die sogenannte Möglichkeit Parameter zu *mappen*. Das bedeutet statt einer Parameternummer wird die Nummer des map-Feldes des Runs angegeben (vgl. Abschnitt 3.1.5). Mit `map1` anstelle einer Parameternummer wird für jeden Run der Parameter verwendet, der im map-Feld Nummer 1 steht. Da jeder Run sein eigenes map-Feld hat kann so für jeden Run auf einen anderen Fitparameter zugegriffen werden.

Im `FUNCTIONS`-Block können Verknüpfungen von Fitparametern definiert werden (vgl. Abschnitt 3.1.4). Auf diese kann zugegriffen werden indem als Parameter der Funktionsbezeichner der Verknüpfung angegeben wird (z.B. `fun1` für die erste Verknüpfung).

Sollen Funktionen gefittet werden die nicht direkt von WKM unterstützt werden, so besteht die Möglichkeit diese mit der Theorie `userfunc` zu bewerkstelligen. Als einziger Parameter wird die explizite Angabe der Funktion erwartet.

```
TFieldCos     2     4              (phase frequency)
```

ist somit identisch mit

```
userfunc   cos((2*pi*par4*T)+(pi*(par2/180))).
```

Der Fit einer `userfunc` ist allerdings wesentlich aufwendiger und benötigt somit wesentlich mehr Rechenzeit. Wie aus dem Beispiel deutlich wird erreicht man die Zeit mit der Variablen `T`. Genauere Angaben zum Syntax in dem die Funktion angegeben werden muß findet sich im Abschnitt 3.1.4.

3.1.4 Der `FUNCTIONS`-Block

WKM verfügt über einen Parser, der es ermöglicht, Funktionen zu definieren. Verwendet werden dürfen die Grundrechenarten.

`+`	Addition
`-`	Subtraktion
`*`	Multiplikation
`/`	Division
`^`	Potenzieren
`[()]`	Klammerung

Weiterhin stehen eine Reihe wissenschaftlicher Funktionen und Konstanten zur Verfügung (`cos, sin, tan, cosh, sinh, tanh, acos, asin, atan, acosh, asinh, atanh, log, ln, exp, pi`).

Der Benutzer kann dabei auf die Fitparameter (`par5` bezeichnet den Fitparameter Nummer 5) zurückgreifen. Gemappte Parameter erreicht man z.B. über `map1`.

Die definierten Funktionen werden mit `fun1`, `fun2` und so weiter bezeichnet, und müssen jeweils in einer eigenen Zeile stehen. Sie können dann anstelle der Parameternummern im `THEORY`-Block eingesetzt werden (vgl. Abschnitt 3.1.3) [5].

```
FITPARAMETER
#       Nr. Name        Value   Step    Bounderies
          1 ALPHA       0.9987  .0001   0.5     1.5
          2 PHASE       -5.01   .002    -360    360
```

[5] Bei der Definition von Funktionen ist es immer vorzuziehen, Klammern zu verwenden, da der Parser 'Punkt vor Strich' nicht richtig beherrscht.

```
          3 A_GES       0.28    .001    0       1
          4 LAMBDA_1    0.425   0.002   0       10
          5 ASY_1       0.33    0.001   0       1
          6 LAMBDA_2    0.132   0.001   0       10

##############################################################
THEORY
asymmetry       fun1
simplExpo       4               (rate)
+
asymmetry       fun2
simpleExpo      6                (rate)

##############################################################
FUNCTIONS
fun1 = par3 * par5
fun2 = par3 * (1 - par5)
```

In diesem Beispiel wird die Gesamtasymmetrie angepaßt oder vorgegeben, und die Funktionen `fun1` und `fun2` stellen die Bruchteile der Gesamtasymmetrie dar, die den beiden Unterspektren zugeordnet wird.

3.1.5 Der `RUN`-Block

In diesem Block stehen Informationen über die zu verwendenen Datensätze, die Art des Fits, Zeitmarken für Untergrund, Daten und Nullkanal, das gewählte Binning, sowie die gemappten Parameter.

Dieser Block unterscheidet sich insoweit von den vorherigen Blöcken, das hier auf Blockbezeichner noch Informationen in der gleichen Zeile folgen. Dieses sind der Name des Datensatzes sein Typ und sein Format. Der Name des Datensatzes entspricht dem Dateinamen des Datensatzes (ohne Extension), der Typ dem Spektrometer an dem der Datensatz aufgenommen wurde (z.B. `mue1`, `pim3`, `emu`, `m15`, `nemu`, ...), welcher auch gleichzeitig die Extension der Datendatei darstellt und das Format normalerweise dem Beschleuniger an dem das Spektrometer lokalisiert ist (z.B. `psi`, `ral`, `triumf`). Eine Ausnahme stellt hier das `nemu`-Format der Niederenergie-μSR-Beamline am PSI dar. [6] Sind die Daten in einem speziellen Datenpfad untergebracht, so sollte die ersten beiden Zeichen des Datensatznamen das Meßjahr bezeichnen. Diesen beiden Zeichen wird '19' vorangestellt und dann als *year* im Datensuchpfad

[6] vergleiche hierzu Abschnitt 3.2.

verwandt (vgl. Tabelle 1.1). Der 1997 am μE4 gemessene Run Nr. 953 könnte dann z.B.

```
9700953.mue4
```

heißen.

In diesem Block werden nach folgenden Schlüsselwörtern Parameter erwartet:

`fittype` an dieser Stelle wird gekennzeichnet auf welche Weise die Daten gefittet werden sollen.

- 0 Einzelspektrenfit (Rohdatenfit)
- 2 Asymmetriefit
- 4 Asymmetriefit nach dem Rotating Reference Frame Verfahren.

Von den folgenden Parameter werden nicht alle für jeden Fittyp benötigt. Bei den Parameter die nur für einige Fittypen relevant sind ist dies in Klammern angegeben.

`rrffrequency, rrfpacking` Frequenz und Binning des Rotating Reference Frame Verfahrens (*Fittype 4*).

`alpha, beta` hier werden die Parameternummer der Detektoreffizienz $\alpha = N^0_{backward}/N^0_{forward}$ und Detektorasymmetrie $\beta = |A^0_{backward}|/|A^0_{forward}|$ eingetragen.[7] Wird alpha und beta kein Parameter zugewiesen wird jeweils 1 angenommen (*Fittype 2,4*).

$$A_{reduziert} = \frac{A_{roh}(\alpha+1)(\alpha-1)}{(\alpha\beta+1)+A^{roh}(\alpha\beta-1)}$$

`alpha2, beta2` Parameternummern der Detektoreffizienz und Detektorasymmetie für das Asymmetiespektrum aus `right` und `left` (*Fittype 4*).

`norm` Nummer des Fitparameters der die Detektorzählrate zur Zeit 0 angibt $N_{forward}(t=0) = N^0_{forward}$ (*Fittype 0*).

`backgr.fit` Nummer des Fitparameters für den Background. Mit diesem Parameter kann bei Einzelspektrenfits ein konstanter Background gefittet werden. In diesem Fall werden die Angaben unter `background` ignoriert (*Fittype 0*).

[7] N bezeichnet die Detektorzählrate, A^0 die Anfangsasymmetrie und die Rohasymmetrie ist gegeben durch $A_{roh} = \frac{N_{forward}(t)-N_{backward}(t)}{N_{forward}(t)+N_{backward}(t)}$

`rphase` Nummer des Fitparameters für die Detektorphase.

`lifetime` Nummer des Fitparameters für die Myonenlebensdauer. Fehlt diese Zeile wird mit 2.197147 μs Lebensdauer gerechnet.

`lifetimecorrection` Benötigt keinen Parameter. Korrigiert den exponentiellen Zerfall der Myonen aus der graphischen Darstellung der Fitergebnisse (*Fittype 0*).

`map` Mapping-Feld. Hier können bis zu 5 Fitparameter angegeben werden die im `THEORY` und `FUNCTIONS`-Block mit `map1` bis `map5` erreicht werden können (vgl. Abschnitt 3.1.3)

`forward, backward` Zur Bildung der reduzierte Asymmetrie $A_{reduziert}$ muß zugeordnet werden welche Histogramme aus dem Datenfile als Forward- ($N_{forward}$) bzw. Backwardsignal ($N_{backward}$) interpretiert werden sollen. Dazu werden die Nummern der Histogramme jeweils hinter dem entsprechenden Schlüsselwort aufgelistet.

Sollen Einzelspektren gefittet werden (`fittype 0`) wird nur forward benötigt[8].

`right, left` Histogramme zur Bildung der reduzierte Asymmetrie aus den um 90° verschobenen Detektoren um das Signal Rauschverhätnis im Rotating Reference Frame zu verbessern (*Fittype 4*).

`backgr.fix` Hier kann explizit der Background angegeben werden, erst für das Forward- dann für das Backwardhistogramm. Die Angaben bei `background` werden dann nicht benutzt.

`background` An dieser Stelle stehen die Kanäle der Datenhistogramme, die zur Berechnung des Untergrundsignals benutzt werden, in der Reihenfolge $k^{first}_{forward}\, k^{last}_{forward}\, k^{first}_{backward}\, k^{last}_{backward}$.

`data` Analog zum `background` ist an dieser Stelle der maximal gültige Datenbereich angegeben.

`t0` Hier sind die die Kanalnummern der t_0-Kanäle für das Forward- und das Backwardspektrum angegeben, und zwar in der Reihenfolge $t_0^{forward}\, t_0^{backward}$.

`fit` Die Angaben bezeichnen den Bereich der gefittet werden soll. Es wird Anfangszeit und Endzeit in μs angegeben.

`packing` Die Angabe bezeichnet das anzuwendende Binning.

[8] Um `wkmt0` auszuführen ist es nötig sowohl forward als auch backward zu belegen.

Werden für einen Fit mehr als ein Datensatz benötigt, so werden entsprechend viele derartige Blöcke getrennt durch eine Leerzeile aneinandergehängt.

3.1.6 Der COMMANDS-Block

An dieser Stelle stehen, die Kommandos, mit denen MINUIT innerhalb von WKM aufgerufen wird. Eine genauere Erläuterung der Minuit-Kommandos findet man in [1]. Normalerweise kann mit folgenden Kommandos gearbeitet werden.

```
COMMANDS
SET BATCH
MIGRAD
SAVE
END RETURN
```

Es ist auch möglich MINUIT interaktiv zu betreiben dazu mehr in Anhang B.

3.1.7 Der PLOT-Block

In diesem Block werden Parameter für die graphische Ausgabe mit wkmview eingestellt. Dem Blockbezeichner PLOT folgt gleich eine Zahl die den Plottyp festlegt. Dieser sollte mit dem Fittyp übereinstimmen. Desweiteren kann angegeben werden:

runs Hier müssen die Nummern der Runs angegeben werden die graphisch dargestellt werden sollen. Der erste Run im RUN-Block wird mit 1 bezeichnet etc..

range Hier kann der Plotbereich explizit angegeben werden in der Reihenfolge t_{min}, t_{max}, A_{min}, A_{max}. Wird auf eine range Einstellung verzichtet, so wird als Zeitfenster der Fitbereich des ersten Runs gewählt. Wird auf eine Begrenzung der Ordinate verzichtet so wird ein Bereich gewählt in dem alle Daten dargestellt werden können.

3.1.8 Der STATISTIC-Block

Am Ende des Steuerfiles stehen noch einige Informationen zum durchgeführten Fit, wie Datum und Uhrzeit, sowie der Absolutwert und die reduzierte χ^2-Summe.

```
STATISTICS --- 08.08.97 13:35h
   chi2:      abs = 45.411      norm = 1.02234
```

Dieser Block enthält erst dann sinnvolle Angaben, wenn das Steuerfile bereits einmal im Rahmen eines Fits durchlaufen wurde.

3.2 Datendateiformate

In der Regel liegen die Rohdaten in einem dem jeweiligen Spektrometer entsprechenden - meist binären - Format vor. Diese werden von Konvertierungsprogrammen (vgl. Annhang C in eine ASCII Textdatei im `WKM`-Datenformat umgewandelt.

In älteren Versionen von `WKM` (<0.971) wurden auch andere Datenformate unterstützt deren Formate der Vollständigkeit halber im Anhang D erläutert werden.

Name	Bezeichnung
Run	Runnummer
Title	Probenname
Temp	Temperatur
Field	Magnetfeld
Setup	Probenorientierung
Date	Datum (optional)
Groups	Histogrammanzahl
Groupnames	Histogrammnamen (optional)
Channels	Kanalanzahl pro Histogramm
Resolution	Zeitauflösung in μs
Events(pG)	Mittlere Ereignisszahl pro Histogramm (optional)

Tabelle 3.2: Schlüsselworte des `WKM`-Datenformats

3.2.1 `WKM`-Datenformat

Die Datei beginnt mit einer Komentarzeile die meißt Informationen über das Konvertierungsprogramm enthält. Anschließend beginnt ein Kontrollblock in dem jede Zeile mit einem Schlüsselwort (Tabelle 3.2, gefolgt von einen ':', eingeleitet wird die den darauffolgenden Parameter spezifiziert. Ist

das Schlüsselwort unbekannt wird die Zeile übersprungen. Duch solche unbekannten Schlüßelwörter lassen sich zusätzlich Informationen bzw. Komentare in dem Datenfile unterbringen. Der Block wird durch eine Leerzeile beendet.

Anschließend wird eine Zeile ignoriert und es folgt das erste Histogramm. Zwischen den Histogrammen werden jeweils drei Zeilen ignoriert.

4. Die Fittypen

4.1 Einzelhistogrammfit

4.2 Asymmetriefit

4.3 Rotating Reference Frame Fit

Der Rotating Reference Frame Fit findet hauptsächlich in der Auswertung von Knightshiftmessungen Anwendung. In diesen Messungen wird oft mit einem hohen transversalen Magnetfeld gearbeitet, wodurch die Messungen eine hochfrequente Oszillation aufweisen. Um diese einfacher auswerten zu können wird die Messung in ein rotierendes Koordinatensystem transformiert.

Die Rotationsfrequenz

```
Beispiel für ein RRF Fit
#################################################################
FITPARAMETER
#       Nr. Name         Value     Step      Bounderies
          1 ALPHA_tb     0.9435    0
          2 ALPHA_rl     0.9002    0
          3 ASYM1        0.09097   0.001498  0         0.4
          4 ASYM2        0.04402   0.001327  0         0.4
          5 LAMBDA       0.6032    0.01544   0         50
          6 LAMBDA       0.4315    0.02181   0         50
          7 PHASE1       81.69     1.138     -270      270
          8 PHASE2       59.09     2.195     -270      270
          9 FREQ1        8.422     0.002834
         10 FREQ2        8.827     0.004476

#################################################################
THEORY
asymmetry     3
simplExpo     5                  (rate)
TFieldCos     7     9            (phase frequency)
+
asymmetry     4
simplExpo     6                  (rate)
TFieldCos     8    10            (phase frequency)
```

```
##################################################################
RUN   1234567  M15 TRIUMF       (name type format)
fittype         4               (rotating reference frame fit)
rrffrequency    805.00
rrfpacking      101
alpha           1
alpha2          2
map             0    0   0    0    0    0    0    0    0    0
forward         1
backward        3
right           4
left            2
background    247   2615    247   2615    247   2615    247   2615
data         3000 102247   3000 102247   3000 102247   3000 102247
t0           2910   2909   2911   2911
fit          0.01   8.00          (fw bw)
packing         2

##################################################################
COMMANDS
SET BATCH
MIGRAD
SAVE
END RETURN

##################################################################
PLOT 4   (rotating reference frame plot)
runs     1,1 1,2
range 0 4

##################################################################
STATISTIC --- 30.08.00 16:32h
  chi:      abs = 927.797           norm = 1.15685
```

Achtung: Die Detektoreffizienz α und Detektorasymmetrie β werden im `fittype 4` nicht gefittet, sondern als Konstanten für die Datenumwandlung verwendet. Sie sollten vorher im Asymmertiefit (`fittype 2`) bestimmt werden.

5. Datenvorbereitung für den Fit- und Plotvorgang

In diesem Abschnitt wird beschrieben, wie die Daten der gemessenen Histogramme eingelesen werden und wie sie verarbeitet werden bevor sie gefittet bzw. geplottet werden.

Da sich die Verarbeitung für die einzelnen Fit- und Plottypen deutlich voneinander unterscheidet werden diese im folgenden seperat behandelt.

Zunächst werden die Daten der Histogramme eingelesen und entsprechend der `RUN`-Block Parameter `forward` und `backward` zum forward- und backward-Histogramm zusammenaddiert.

5.1 Einzelhistogrammfit/plot (`fit/plottype 0`)

Im `fittype 0` wird ein einzelnes Histogramm gefittet bzw. im `plottype 0` dargestellt. Dieses Histogramm muss als forward angegeben werden. Weiterhin werden im `RUN`-Block folgende Parameter verwendet: `norm`, `lifetime`, `lifetimecorrection`, `rphase` und `backgr.fit`.

Wenn kein Background-Parameter (`backgr.fit`) angegeben ist wird der Background wie beim Asymmetriefit bestimmt und abgezogen (vgl. Abschnitt 5.2.1).

Dann werden die Daten gepackt. Dazu werden jeweils soviele aufeinanderfolgende Datenpunkte zusammen addiert wie im `RUN`-Block Parameter `packing` angegeben. Die Histogrammkanäle auf die diese Operation angewendet wird ergibt sich aus:

$$[max\left(t_0 + a/b, d_1\right)), min\left(t_0 + e/b, d_2\right))] \tag{5.1}$$

- t_0 : Kanal der dem Zeitnullpunkt entspricht (`t0` im `RUN`-Block)
- a : Fitanfang in µs (erster Wert von `fit` im `RUN`-Block)
- e : Fitende in µs (zweiter Wert von `fit` im `RUN`-Block)
- b : Zeitauflösung der Daten (aus dem Datenfile entnommen)
- d_1 : Erster guter Datenkanal (erster Wert von `data`im `RUN`-Block)
- d_2 : Letzter guter Datenkanal (zweiter Wert von `data`im `RUN`-Block)

Nach dem Packen werden die Fehler der gepackten Histogrammkanäle $\Delta N(k)$ berechnet:

$$\Delta N(k) = \sqrt{N(k) + b \cdot p} \tag{5.2}$$

$N(k)$: Zählrate des Histogrammkanals k
b : Background falls er vorher bestimmt wurde (vgl.oben)
p : Packing (`packing` im `RUN`-Block)

5.1.1 `Fittype 0`

Zum Fitten wird das so erhaltene Datenhistogramm mit folgender Theoriefunktion angepasst:

$$N(t) = N_0 \cdot e^{-t/\tau_\mu} \left(1 - cos\left(\varphi A\right)\right) + B \tag{5.3}$$

$N(t)$: Messdaten
N_0 : Anfangszählrate (`norm` im `Run`-Block)
τ_μ : Myonenlebensdauer (`lifetime` im `Run`-Block)
φ : Phase (`rphase` im `Run`-Block)
B : Background (`backgr.fit` im `Run`-Block)
A : Depolarisationsfunktion wie im `theory`-Block angegeben.

5.1.2 `Plottype 0`

Für die grafische Darstellung werden die Daten werden nach folgender Gleichung Background und Myonenzerfall herausgerechnet und die Datenpunkte $N_P(t)$ dargestellt:

$$N_P(t) = \frac{N(t) - B}{e^{-t/\tau_m u}}. \tag{5.4}$$

$N(t)$: Messdaten
τ_μ : Myonenlebensdauer (bestimmt über RBS `lifetime`)
B : Background (bestimmt über RBS `backgr.fit`)

Als Theoriefunktion T(t) wird folgende Funktion in der Grafik mit dargestellt:

$$T(t) = N_0 \left(1 + cos\left(\varphi A\right)\right) \tag{5.5}$$

N_0 : Anfangszählrate (bestimmt über RBS `norm`)
φ : Phase (bestimmt über RBS `rphase`)
A : Depolarisationsfunktion wie im `theory`-Block angegeben.

5.2 Asymmetriefit/plot (`fit/plottype 2`)

Für einen Asymmetriefit `fittype 2` bzw. Asymmetrieplot `plottype 2` werden sowohl `forward` als auch `backward`-Histogramm im `RUN`-Block benötigt. Weiterhin werden die Parameter `alpha`, `beta` und `backgr.fix` verwendet.

Ist im `RUN`-Block der Parameter `backgr.fix` angegeben, wird dieser feste Wert von den Histogrammen abgezogen. (Der erste vom Forwardhistogramm der zweite vom Backwardhistogramm.) Ansonsten wird der Background als Mittelwert aus dem Histogrammkanalintervall welches mit dem Parameter `background` angegeben wird berechnet und vom Histogramm abgezogen. Ist das Backgroundintervall länger als die Periode des Beschleunigers, wird das Intervall auf ein ganzzahliges Vielfaches der Beschleunigerperiode gekürzt. (Die in WKM verwendeten Perioden sind in Tabelle 5.1 aufgeführt.)

Beschleuniger	Periode (µs)
PSI	0.01975
TRIUMF	0.04337
RAL	0.0

Tabelle 5.1: Beschleunigerperioden in WKM

Anschließend werden die Histogramme synkronisiert, d.h. die Schittmenge der Datenbereiche für Forwardhistogramm und Backwardhistogramm (mit Parameter `data` angegeben) wird für beide Histogramme als Datenbereich eingestellt. Dann werden beide Histogramme einzelnd gepackt und deren Fehler (Gleichung 5.2) berechnet, wie schon in Abschitt 5.1 für den `fittype 0` beschrieben.

5.2.1 `fittype 2`

Diese Histogramme werden in das Asymmetriespektrum $A(k)$ umgerechnet:

$$A(k) = \frac{N_f(k) - N_b(k)}{N_f(k) + N_b(k)} \tag{5.6}$$

$N_f(t)$: Forwardhistogrammzählrate des Kanals k
$N_b(t)$: Backwardhistogrammzählrate des Kanals k

Im Fit wird dieses Asymmetriespektrum an folgende Theoriefunktion angepasst:

$$A(t) = \frac{A\,(\alpha\beta + 1) - (\alpha - 1)}{(\alpha + 1) - A\,(\alpha\beta - 1)} \tag{5.7}$$

A : Depolarisationsfunktion wie im `theory`-Block angegeben.
α : Detektoreffizienz (`alpha` im `RUN`-block)
β : Detektorasymmetrie (`beta` im `RUN`-block)

5.2.2 `plottype 2`

Für die graphische Darstellung werden Forward- und Backwardhistogramme in ein reduziertes Asymmetriespektrum $A_{reduziert}(k)$ umgerechnet:

$$A_{reduziert}(k) = \frac{\alpha N_f(k) - N_b(k)}{\alpha\beta N_f(k) + N_b(k)} \tag{5.8}$$

$N_f(t)$: Forwardhistogrammzählrate des Kanals k
$N_b(t)$: Backwardhistogrammzählrate des Kanals k
α : Detektoreffizienz (`alpha` im `RUN`-block)
β : Detektorasymmetrie (`beta` im `RUN`-block)

Als Theoriefunktion wird wie Depolarisationsfunktion wie im `theory`-Block angegeben in der Grafik mit dargestellt.

5.3 Rotating Reference Frame Fit/Plot (`fit/plottype 4`)

Für einen Rotating Reference Frame Fit `fittype 4` bzw. Plot `plottype 4` werden wie beim Asymmetriefit/plot sowohl `forward` als auch `backward`-Histogramme im `RUN`-Block benötigt, es konnen auch zusätzlich `right` und `left`-Histogramme mit verwendet werden. Wie die Histogramme bezeichnet werden geht aus Tabelle 5.2 hervor. Weiterhin werden die Parameter `rrffrequeny`, `rrfpacking`, `alpha`, `beta`, `alpha2`, `beta2` und `backgr.fix` verwendet.

Histogramm	Winkel
`forward`	0°
`right`	90°
`backward`	180°
`left`	270°

Tabelle 5.2: Winkel der Fithistogramme (im Drehsinn der Myonenpräzession).

Zunächst wir aus `forward` und `backward` die reduzierte Asymmetrie A_{fb} wie in Gleichung 5.8 und Abschnitt 5.2 gebildet und soweit vorhanden aus `right` und `left` die reduzierte Asymmetrie A_{rl}. Für die Berechnung von A_{rl} wird `alpha2` und `beta2` als α und β verwendet.

Hieraus werden dann die Spektren des Realteils R_{RRF} und Imaginärteils I_{RRF} im rotierenden Koordinatensystem gebildet:

$$R_{RRF} = \quad A_{fb}\, cos(2\pi\, \nu_{RRF}\, t) + Arl\; sin(2\pi\, \nu_{RRF}\, t) \tag{5.9}$$

$$I_{RRF} = -A_{fb}\, sin(2\pi\, \nu_{RRF}\, t) + Arl\; cos(2\pi\, \nu_{RRF}\, t) \tag{5.10}$$

ν_{RRF} : Frequenz des rotierenden Koordinatensystems (`rrffrequency` im `RUN`-Block)

Diese Spektren werden dann nochmals gepackt. Die neuen Datenpunkte ergeben sich jeweils als arithmetisches Mittel über soviele Datenpunkte wie im `RUN`-Block-Parameter `rrfpacking` angegeben.

5.3.1 `fittype 4`

Das Realteilspektrum wird im Fit direkt an die Depolarisationsfunktion, wie im `THEORY`-Block angegeben, angepasst. Für das Imaginärteilspektrum wird die Phase der `TFieldCos`-Funktion jeweils um $\pi/2$ verringert.

5.3.2 `plottype 4`

Im Plot werden Realteil und Imaginärteil sowie die Theoriefunktionen wie sie im Fit verwendet werden dargestellt.

A. Besonderheiten der Installation auf verschiedenen Systemen

A.1 Installation unter VMS

Für VMS gibt es bislang nur eine WKM-Version für die **ALPHA-Architektur**. Da für diese Plattform die erforderliche `Emacs`-Version noch nicht existiert, wird hier die Programmsteuerung und das Dateihandling mit DCL-Files durchgeführt (Tabelle A.1). Zu diesem Zweck werden folgende Logicals definiert: [1]. Wobei die Pfade entsprechend dem eigenen Account angepaßt werden müssen.

```
"WKM$PRINTER" :== "@WKM$EXE:WKMPRINT"
"WKM$DIR" = "DISK_142_SRC0:[NEMU.WKM]"
"WKM$EXE" = "DISK_142_SRC0:[NEMU.WKM.EXE]"
"WKM$WORK" = "DISK_142_SRC0:[NEMU.WKM.WORK]"
"WKMDATA$DIR" = "DISK_142_DAT0:[NEMU.WKM]"
"WKMTAB$DIR" = "DISK_142_SRC0:[NEMU.WKM.TAB]"
```

Dies geschieht in der Regel im `LOGIN.COM` . durch Aufrufen der DCL-Routine `SETUPWKM.COM` .

Name	Aufruf	Beschreibung
TIME.COM	TIME DUMMY	Aufruf von `wkmt0`
FIT.COM	FIT DUMMY	Aufruf von `wkmfit`
SWAP.COM	SWAP DUMMY	Tauscht DUMMY.MSR und DUMMY.MLOG
PLOT.COM	PLOT DUMMY	Aufruf von `wkmview`
PARA.COM	PARA DUMMY	Aufruf von `wkmpara`

Tabelle A.1: DCL-Routinen zur Steuerung von WKM

Folgender Verzeichnisbaum wurde eingerichtet, wobei im folgenden `DUMMY` durch eine beliebige Bezeichnung für einen entsprechenden μSR-Run ersetzt werden muß. `DUMMY.MSR` kennzeichnet das Input-File, `DUMMY.NEMU` das Datenfile (vgl. 3.2, 3.1) [2]

[1] Die beiden für den fehlerlosen Ablauf des Programms wesentlichen Logicals sind `WKMDATA$DIR`und `WKMTAB$DIR`. Sie zeigen auf die Daten- , bzw. die Tabellverzeichnisse.

[2] Je nach verwendetem Spektrometer haben diese Datenfiles eine andere Extension, damit `WKMFIT` ein entsprechendes Datenformat zuordnen kann.

```
WKM$DIR --- [.EXE]  -WKMFIT.EXE
          |            WKMVIEW.EXE
          |            WKMTO.EXE
          |            WKMPARA.EXE
          |            SETUPWKM.COM
          |            FIT.COM
          |            PLOT.COM
          |            SWAP.COM
          |            TIME.COM
          |            WHAT.COM
          |            WKMPRINT.COM
          |
          |- [.TAB]  -GLF.TBL
          |            LLF.TBL
          |            KDLLF.TBL
          |            KDGLF.TBL
          |
          |- [.WORK] -DUMMY.MSR
          |
          |- [.DOC]

WKMDATA$DIR ---     -DUMMY.NEMU
```

B. Minuit von WKM interaktiv betreiben

Es ist möglich `MINUIT` von WKM interaktiv zu betreiben. Dazu muß der `COMMANDS`-Block wie folgt abgeändert werden:

```
COMMANDS
SET OUT 6
SET INP 5
SET INT
SAVE
END RETURN
```

Für eine genauere Beschreibung des interaktiven Modus von `MINUIT` sei auf das Manual verwiesen [1].

C. Das Datenkonvertierungsprogram

Am PSI und am TRIUMF können die binären Meßdaten mittels des Programms `convert` in das ASCII `WKM`-Datenformat umgewandelt werden.
Der Syntax lautet:

`convert` *experiment jahr start stop*

Dabei steht *experiment* für das Spektrometer an dem gemessen wurde, als z.B. `mue1`, `pim3` oder `m15`. Für *jahr* gibt man das Meßjahr im Format jj ein, und *start stop* geben die erste bzw. letzte Runnummer an die konvertiert werden soll.
Mit

```
convert m20 99 17
```

wird z.B. der Run 17 aus dem Jahr 1999 am M20 (TRIUMF) in die ASCII-Datei `9900017.m20` konvertiert.

Die konvertierten Dateien liegen dann jeweils auf den Scratch-Disks.

D. Alte Datendateiformate

D.1 PSI-Datenformat

Das ASCII-PSI-Datenformat besteht aus Kolonnen von Zahlen die in Tabelle D.2 erläutert werden. WKM erkannte dies als PSI-Format (vgl. Abschnitt 3.1.5)[1].

Name	Bezeichnung
Run	Runnummer
Title	Probenname
Field	Magnetfeld
Setup	Probenorientierung
Temp	Temperatur
Groups	Histogrammanzahl
Channels	Kanalanzahl pro Histogramm
Resolution	Zeitauflösung in μs

Tabelle D.1: Schlüsselworte des RAL-ASCII Datenformats

D.2 RAL-Datenformat

Aus dem binären RAL-Datenformat wird mit einem am RAL verfügbaren ASCII-Konvertierungsprogramm (???) folgendes ASCII-Format erstellt:

Beginnt die Datei mit dem Schlüsselwort `UDALASER` so werden die ersten vier Zeilen als Kommentar ignoriert, ansonsten die ersten drei. Anschließend beginnt ein Kontrollblock in dem jede Zeile mit einem Schlüsselwort (Tabelle D.1, gefolgt von einen ':', eingeleitet wird die den darauffolgenden Parameter spezifiziert. Ist das Schlüsselwort unbekannt wird die Zeile übersprungen. Der Block wird durch eine Leerzeile beendet.

Anschließend wird eine Zeile ignoriert und es folgt das erste Histogramm. Zwischen den Histogrammen werden jeweils drei Zeilen ignoriert.

Das RAL-Datenformat ist zum `WKM`-Datenformat kompatiebel.

[1] Versionen vor 0.971

Name	Beschreibung
KDTRES	TDC Zeitauflösung (0-15)
KDOFTI	TDC Zeitüberlauf. Überlauf bei (*KDOFTI*+0.5)*160 nsec
NRUN	Runnummer
LENHIS	Länge eines Histogramms in Kanälen
NUMHIS	Anzahl der benutzten Histogramme
IBR	CAMAC branch
ICR	CAMAC crate
NTD	Linke CAMAC Station des TDC
NHM	CAMAC Station des Histogrammspeichers
HMTYPE	Typ des Histogrammspeichers (`CES` oder `LRS`)
MONDEV	Typ des Temperaturmonitor
NUMDAV	Anzahl der Histogrammdatensätze (*NUMHIS * KDAFHI*)
LENDAV	Länge (in Kanälen) der Histogrammdatensätze
KDAFHI	Anzahl der Histogrammdatensätze pro Histogramm
KHIDAF	Anzahl der Histogramme pro Histogrammdatensatz
TITLE	4 x 10 Zeichen für Probe, Temperatur, Magnetfeld, Orientierung
SETUP	gewählter Datenerfassungsmodus (10 Zeichen)
DATE1	Datum des Meßbeginns
DATE2	Datum des Schreibens der Datei (Meßende)
TIME1	Uhrzeit des Meßbeginns
TIME2	Uhrzeit des Schreibens der Datei (Meßende)
CNTOLD	Anzahl der Events im Histogramm (16 Zahlen)
TOTOLD	Gesammtzahl der Events
NTO	t_0-Kanal im Histogramm (16 Zahlen)
NTINI	Erster guter Datenkanal im Histogramm (16 Zahlen)
NTFIN	Letzter guter Datenkanal im Histogramm (16 Zahlen)
I2ADC	Letzter Temperaturwert (4 Zahlen) (alt)
ILT	unteres Temperaturlimit(4 Zahlen) (alt)
IUT	oberes Temperaturlimit(4 Zahlen) (alt)
SCTYPE	singles scaler Typ (`S500` oder `S500A`)
IFTYPE	Typ der CAMAC Interfaces (6 = SCI-2280, 9 = CCP)
NIVG	Stationsnummer des CAMAC Interfaces
DKSPER	Periode zwischen Speichern und Datenaufnahme
MONPER	Periode zwischen Temperaturmessungen
I4SCAL	Inhalt der scaler 1 bis 6 (6 Zahlen)
NCS	CAMAC Station der singles scaler
	(8 Zahlen) jeweils 1
NIO	CAMAC Station der IO506
C62TXT	Run Untertitel (Beschreibung)
SCALA	Label der Scaler 1 bis 5 (jeweils 4 Zeichen)
HISLA	Label der Histogramme 1 bis 16 (jeweils 4 Zeichen)
NUMHIS **Histogramme mit jeweils** *LENHIS* **Kanälen**	

Tabelle D.2: Aufschlüsslung des PSI ASCII Datenformats. Die fett gedruckten Werte werden von WKM benötigt.

D.3 TRIUMF-Datenformat

Das ASCII-TRIUMF-Datenformat entspricht dem PSI-Datenformat wie in Anhang D.1 beschrieben, da die Daten zunächst mit dem Programm TRI2PSI in das binäre PSI-Format gewandelt wurden.

D.4 NEMU-Datenformat

Aufgrund der eventorientierten Datenaufnahme bei der LEμSR liegen die Rohdaten hier in Form sogenannter "n-Tuple"vor. Die für WKM notwendigen Zerfallsspektren müssen zunächst erstellt werden. Dazu wird das Programmpaket `PAW` (Physics-Analysis-Workstation) benutzt [4]. [2].

Das resultierende Datenfile hat die Extension `.nemu`, und ist wie folgt aufgebaut, wobei die für `WKMFIT` wesentlichen Teile die letzten 3 Zeilen des Kopfes sind, in denen die Anzahl der Gruppen, der Kanäle pro Gruppe und die im Experiment verwendete TDC-Auflösung stehen.

```
- data file created by wkm.kumac (07.02.97/MB) -
NEMU_Run:           R1290$1291
WKM_Run:            A1_1
Date:               18-NOV-96 2:08:00
Title:              Ag 125um
Field:              50 Gauss TF
Setup:              15 kV
Temp:               RT
TOF(M3S1):          63.4      77.4
Groups:             4
Channels:           10000
Resolution:         0.001
```

Nach einer Leerzeile folgen nun in diesem Fall 4 Gruppen von 10000 Integer-Zahlen, die den vier Rohspektren in Reihenfolge *Left, Top, Right, Bottom* entspricht. Die Gruppen sind jeweils durch eine Leerzeile voneinander getrennt.

Das NEMU-Datenformat ist zum `WKM`-Datenformat kompatiebel.

[2] Hinweise zum Gebrauch findet man mit `PAW> WKM ?`

Literaturverzeichnis

[1] MINUIT, Function Minimization and Error Analysis. Reference Manual; Application Software Group, CERN.

[2] Grace (Xmgr) Homepage: http://plasma-gate.weizmann.ac.il/grace/

[3] Brigham, E.O.: FFT Schnelle Fourier-Transformation, R. Oldenburger Verlag 1985.

[4] Physics Analysis Workstation Tutorial, Oliver Couef, Aplication Software Group, Computing and Networks Devision, CERN. PAW Homepage: http://wwwcn.cern.ch/pl/paw/

[5] Schatz, G.; Weidinger, A.: Nukleare Festkörperphysik, Teubner Studienbücher Physik