Vorlesung “Astronomische Datenanalyse”

Praktikum 2: Datenmodellierung

Version vom 3. Juni 2002

Das Praktikum findet im Institut für Astronomie und Astrophysik, Abt. Astronomie im Gebäude Sand 1 statt. Unter der URL `http://astro.uni-tuebingen.de/~wilms/teach/data/` finden Sie PS- und PDF-Dateien dieser Anleitung, die Sie sich ausdrucken können, sowie alle hier angegebenen Programme.

1 Datenmodellierung: Lineare Regression

Einer der am häufigsten bei der Modellierung von Daten vorkommenden Fälle ist der der sogenannten *Linearen Regression*, das heißt, der Fall der Anpassung einer Geraden der Form

$$y(x) = a + bx \tag{1}$$

an gemessene Meßwerte. Im folgenden sollen diese Meßwerte durch das Variablenpaar (x_i, y_i) bezeichnet werden, d.h. y_i ist der an der Stelle x_i gemessene Datenwert. Im folgenden machen wir die Annahme, daß die *unabhängige Variable* x_i mit hoher Genauigkeit bekannt ist, während die *abhängige Variable* y_i mit einer Unsicherheit behaftet ist, die wir durch die Standardabweichung σ_i jedes Meßwerts beschreiben wollen. Wie genau σ_i ermittelt wird, hängt vom jeweiligen Experiment ab. Wir werden später dafür noch ein Beispiel sehen. Im folgenden leiten wir die wichtigsten Formeln für die lineare Regression kurz ab. Details zur Herleitung finden sich bei Bevington & Robinson (1992), Brandt (1992) und Roe (1992).

Ziel der folgenden Überlegungen ist es, aus den N Messungen (x_i, y_i) und den σ_i die Koeffizienten a und b zu bestimmen, wobei wir in der Notation Bevington & Robinson (1992) folgen. Aufgrund der Fehler σ_i wird dies nicht mit beliebig hoher Genauigkeit gelingen, d.h. wir können bestenfalls den Abstand zwischen den Daten und der Fitfunktion aus Gl. (1) möglichst kleinhalten. Dieser Abstand ist definiert als

$$\Delta y_i = y_i - y(x_i) = y_i - a - bx_i \tag{2}$$

Je kleiner dieser Abstand ist, desto besser beschreiben a und b die Daten.

Δy_i beschreibt den Abstand eines Datenpunkts (x_i, y_i) von der Geraden. Um *alle* Daten möglichst gut beschreiben zu können, muß aus allen Δy_i ein Maß für die Güte der Anpassung definiert werden. Ein solches Maß ist nicht für alle Anwendungen das gleiche. Beispielsweise könnte Ziel einer Messung sein, den Maximalabstand, also $\max_i |\Delta y_i|$, als Maß für die Güte zu benutzen. In einem solchen Fall würden mit Hilfe eines Computerprogramms a und b so lange minimiert werden, bis $\max_i |\Delta y_i|$ möglichst klein sind. Es gibt außer diesem Fall noch viele andere Verfahren, die für bestimmte Problemstellungen sinnvoll sind.

Im am häufigsten auftretenden Fall normalverteilter Meßwerte hat sich hier die *Methode der kleinsten Quadrate* durchgesetzt. Diese geht davon aus, daß die tatsächliche Beziehung zwischen y und x durch

$$y_0(x) = a_0 + b_0 x \tag{3}$$

gegeben ist. Ziel ist es, aus den Meßdaten Schätzwerte für a_0 und b_0 zu bestimmen.

Trifft es zu, daß die Meßdaten normalverteilt sind, dann ist die Wahrscheinlichkeit, am Punkt x_i den Wert y_i zu messen, gegeben durch

$$P_i = \frac{1}{(2\pi\sigma_i^2)^{1/2}} \exp\left\{-\frac{1}{2}\left(\frac{y_i - y_0(x_i)}{\sigma_i}\right)^2\right\} \tag{4}$$

Die Gesamtwahrscheinlichkeit, alle N Meßwerte zu messen, ist dann durch das Produkt der einzelnen P_i gegeben:

$$\begin{aligned}
P(a_0, b_0) &= \prod_i P_i && (5)\\
&= \prod_i \frac{1}{(2\pi\sigma_i^2)^{1/2}} \exp\left\{-\frac{1}{2}\left(\frac{y_i - y_0(x_i)}{\sigma_i}\right)^2\right\} && (6)\\
&= \left(\prod_i \frac{1}{(2\pi\sigma_i^2)^{1/2}}\right) \cdot \exp\left\{-\frac{1}{2}\sum_i \left(\frac{y_i - y_0(x_i)}{\sigma_i}\right)^2\right\} && (7)
\end{aligned}$$

Für die Wahrscheinlichkeit, die gemessenen Datenwerte mit den *geschätzten* Werten a und b zu erhalten, gilt dementsprechend

$$P(a,b) = \left(\prod_i \frac{1}{(2\pi\sigma_i^2)^{1/2}} \right) \cdot \exp\left\{ -\frac{1}{2} \sum_i \left(\frac{y_i - y(x_i)}{\sigma_i} \right)^2 \right\} \tag{8}$$

Für die Kombination von a und b, bei der $P(a,b)$ ein Maximum hat, ist es daher am wahrscheinlichsten, daß sie der tatsächlichen Verteilung von $P(a_0,b_0)$ gehorcht. Dieses Prinzip wird auch als *maximum likelihood principle* bezeichnet.

Da der Vorfaktor in Gl. (8) eine Konstante ist, reduziert sich die Bestimmung des Maximums auf die Bestimmung des Minimums der Summe in der Exponentialfunktion, d.h. auf die Bestimmung des Minimums von

$$\chi^2 = \sum \left(\frac{y_i - y(x_i)}{\sigma_i} \right)^2 \tag{9}$$

Diese Summe wird die χ^2-Summe genannt, ihre statistische Verteilung ist gut bekannt.

Zur Berechnung des Minimums von χ^2 setzen wir $y(x)$ aus Gl. (1) ein und differenzieren nach a und b:

$$\frac{\partial}{\partial a}\chi^2 = -2 \sum \frac{y_i - a - bx_i}{\sigma_i^2} = 0 \tag{10}$$

$$\frac{\partial}{\partial b}\chi^2 = -2 \sum \frac{x_i(y_i - a - bx_i)}{\sigma_i^2} = 0 \tag{11}$$

Diese Gleichungen lassen sich leicht in ein lineares Gleichungssystem für a und b auflösen. Nach einiger Algebra ergibt sich

$$a = \frac{1}{\Delta} \left(\sum x_i^2 \sum y_i - \sum x_i \sum x_i y_i \right) \tag{12}$$

$$b = \frac{1}{\Delta} \left(N \sum x_i y_i - \sum x_i \sum y_i \right) \tag{13}$$

wo

$$\Delta = N \sum x_i^2 - \left(\sum x_i \right)^2 \tag{14}$$

Eine Verallgemeinerung dieser Formeln auf Polynome höheren Grades ist mit etwas mehr Aufwand verbunden, aber prinzipiell möglich. Solche Formeln sind in der IDL-Routine `poly_fit` implementiert, deren Beschreibung Sie sich jetzt in der IDL-Hilfe durchlesen sollten.

Ein in der Astronomie typisches Beispiel für eine lineare Regression ist die Bestimmung der Hubble-Konstanten. Wie Sie aus der Einführungsvorlesung sicherlich noch wissen, besteht zwischen der aus der (kosmologischen) Rotverschiebung ermittelten Geschwindigkeit v und der Entfernung d einer Galaxie die Beziehung

$$v = H_0 d \tag{15}$$

wo H_0 die Hubble-Konstante bezeichnet. Diese wird aus Messungen weit entfernter Objekte bestimmt. Ein Hauptproblem der modernen Kosmologie ist die Bestimmung der Entfernung, während die Geschwindigkeit meist mit hoher Genauigkeit ermittelt werden kann. Für die praktische Bestimmung von H_0 wird daher eine lineare Regression an

$$d = H_0^{-1} v + \text{const.} \tag{16}$$

durchgeführt[1]. Das folgende IDL-Programm ermittelt H_0 aus der Entfernung von Galaxienhaufen, die mit der Tully-Fisher-Methode gemessen wurden (Sakai et al., 2000)[2].

Das folgende IDL-Programm `linhubble.pro` liest die Datenwerte aus Tabelle 4 von Sakai et al. (2000) ein (Datei `hubble.dat`), stellt sie dar, und bestimmt mit `poly_fit` die Hubble-Konstante. Dieses Programm verwendet Strukturen, d.h. Container, in denen mehrere logisch zusammengehörige Datenwerte in einer Variablen praktisch gespeichert werden können. Sollte Ihnen die im Programm verwendete Syntax zu unvertraut sein, dann lesen Sie bitte in der IDL-Hilfe die Informationen über Strukturen.

```
;;
;; (Zu komplizierte) Bestimmung der Hubble-Konstante, Tully-Fisher
;; Werte nach Sakai et al. (2000)
;;
;; Joern Wilms, 2002 Juni 03
;; Version 1.0
;;
PRO readhubble,data
  ;;
  ;; Struktur fuer die Beschreibung der Tully-Fisher Werte
  ;;
  ;; name: Name des Galaxienhaufens
  ;; mm  : Entfernungsmodul, ungenauigkeit 0.4 mag
  ;; d   : Entfernung in Mpc
  ;; vcmb: Geschwindigkeit bzgl. der 3K Hintergrundstrahlung
  ;;
  dat={name:'', n:0, mm:0.0,mmerr:0.0,d:0.,derr:0.,vcmb:0.}
  data=replicate(dat,23)

  ;; Datei
  openr,unit,'hubble.dat',/get_lun
  a=''
  FOR i=0,22 DO BEGIN
      readf,unit,a
      data[i].name=strmid(a,0,10)
      data[i].mm  =float(strmid(a,19,5))
      data[i].mmerr=0.4
      d=float(strmid(a,27,4))
      data[i].vcmb=float(strmid(a,50,5))*d
  ENDFOR

  ;; Berechnung der Entfernung aus dem Entfernungsmodul
  data.d = 10.^((data.mm+5.)/5.)

  ;; Grobe Abschaetzung fuer den Fehler...
  d1 = 10.^((data.mm+data.mmerr+5.)/5.)
```

[1] Diese Behauptung ist nicht wirklich richtig, da wir beispielsweise wissen, daß const. = 0 ist, daher ist eine Bestimmung von H_0 eigentlich einfacher möglich aus dem Mittelwert vieler gemessener v/d – aber wir wollen hier schließlich lernen, wie eine lineare Regression durchzuführen ist, daher ist diese Aufgabe etwas konstruiert …

[2] Sie können diesen Artikel einfach aus dem wichtigsten Handwerkszeug der Astronomie, dem *Astrophysics Data System*, ADS, unter `http://adsabs.harvard.edu/default_service.html` herunterladen. Suchen Sie dazu nach dem Erstautor und dem Jahr. Es erscheint eine Liste von Artikeln dieses Autors, durch Klicken auf das Tag "E" kommen Sie zur online-Version dieses Artikels. Das ADS ist für die astrophysikalische Forschung unverzichtbar – es ist keinesfalls herausgeworfene Zeit, wenn Sie sich im Rahmen dieses Praktikums mit dem Umgang mit dem ADS vertraut machen!

```
  d2 = 10.^((data.mm-data.mmerr+5.)/5.)
  data.derr=(d1-d2)/2.

  ;; pc -> Mpc
  data.d = data.d*1E-6
  data.derr = data.derr*1E-6

  free_lun,unit
END

;; Daten einlesen...
readhubble,data

;; ...und darstellen

;; Plot aufbauen, keine Daten anzeigen
plot,data.d,data.vcmb,/nodata, $
  xtitle='d / Mpc',ytitle=textoidl('v_{CMB} / km s^{-1}')

;; Daten mit Fehlerbalken drueberplotten
jwoploterr,data.d,data.vcmb,replicate(0.,n_elements(data)),dx=data.derr,psym=4

;; Lineare Regression, Modell ist d= (1/H0) * v + const
;; und wir hoffen natuerlich, dass const=0 ist...
res=poly_fit(data.vcmb,data.d,1,measure_errors=data.derr)

print,'H0 ist ',1./res[1],'km/s/Mpc'

END
```

Das Programm verwendet die Funktion `textoidl`, die die Syntax von TEX in die IDL-eigene, sehr unverständliche, Syntax konvertiert (siehe `http://astro.uni-tuebingen.de/software/idl/textoidl/index.html`).

Schreiben Sie nun eine IDL-Unterroutine zur Berechnung der Koeffizienten a und b und vergleichen Sie Ihre Ergebnisse mit denen von `poly_fit`.

Tip: In IDL können Sie die Summe der Werte eines Array mit dem Befehl `total` berechnen.

2 Datenmodellierung: Bahnbestimmung

2.1 Astronomische Einführung

In diesem Abschnitt des Praktikums kümmern wir uns um ein aus dem Leben gegriffenes Beispiel der Datenanalyse: die Bahnbestimmung eines Sterns in einem Doppelsternsystems aus Radialgeschwindigkeitsmessungen.

Im konkreten Fall beschäftigen wir uns mit der Neubestimmung der Ephemeride des Schwarzlochkandidaten Cygnus X-1/HDE 226868. Cyg X-1 ist der am besten bekannte Schwarzlochkandidat stellarer Masse (ca. $10\,M_{\odot}$). In einem galaktischen Schwarzlochkandidaten können wir im Optischen nur den stellaren Begleiter beobachten, und im Röntgenbereich die Emission des vom Begleiter Material akkretierenden schwarzen Lochs. Im Fall von Cygnus X-1 ist die Inklination des Systems (die Neigung der Bahnebene) dergestalt, daß das kompakte Objekt ständig sichtbar ist. Dies macht die Bestimmung der Position des schwarzen Lochs sehr schwierig, da keine Röntgeneklipse vorhanden ist. Da das schwarze Loch und der Begleitstern sich aber um ihren gemeinsamen Schwerpunkt bewegen, ist es immer noch möglich, die Bahn zu bestimmen. Dazu wird im Optischen aus der Dopplerverschiebung charakteristischer Spektrallinien im Sternspektrum die Radialgeschwindigkeit des optischen Begleiters gemessen. Aus dem Verhalten der Radialgeschwindigkeit mit der Zeit ist es dann möglich, auf die Position des schwarzen Lochs zu schließen.

Im Röntgenbereich sehen wir zur Zeit der oberen Konjunktion des kompakten Objekts (das schwarze Loch steht also "hinter" dem Begleitstern) charakteristische kurzzeitige Einbrüche der Röntgenintensität, die sogenannten Röntgendips. Diese Dips entstehen durch Photoabsorption der Röntgenstrahlung in Klumpen in dem akkretierenden Material, das sich zwischen uns und dem schwarzen Loch befindet. Aus Beobachtungen dieses Materials kann man viel über die Eigenschaften des Sternwinds und die Akkretion von Material auf galaktische kompakte Objekte lernen.

Für eine Beobachtungskampagne 1998/1999 mit dem amerikanischen Röntgensatelliten RXTE wollten wir derartige Röntgendips beobachten. Dabei stellte sich heraus, daß die Zeit der oberen Konjunktion des schwarzen Loches nur sehr ungenau bekannt war, da die Ungenauigkeit der bisherigen Ephemeride fast einen halben Tag betrug weil die meisten Bahnbestimmungen auf Daten aus den 1970er Jahren basierten. Wir haben daher neue Radialgeschwindigkeitsmessungen durchgeführt, um die Ephemeride zu verbessern. In diesem Praktikum werden Sie anhand dieser Daten und anhand von historischen Messungen der Radialgeschwindigkeit diese neue Ephemeride selbst bestimmen.

Im Fall von Cyg X-1 können wir annehmen, daß das schwarze Loch sich auf einer Kreisbahn um seinen Begleiter bewegt. Die zum Zeitpunkt t gemessene Radialgeschwindigkeit ist dann gegeben durch

$$v(t) = \gamma + K \sin\left(2\pi \cdot \frac{t - T_0}{P}\right) \tag{17}$$

Hier ist K die Geschwindigkeitsamplitude[3], T_0 ist die *Epoche* der Ephemeride, in der obigen Definition ist T_0 der Zeitpunkt der oberen Konjunktion. Schlußendlich ist P die Bahnperiode des Systems, also die Umlaufzeit des schwarzen Lochs um den Begleiter. Da sich das System noch in der Galaxie bewegen kann und aufgrund systematischer Fehler (z.B. Unsicherheit der Wellenlängenkalibration) kann zusätzlich noch ein Geschwindigkeitsoffset γ auftreten.

[3]Die Geschwindigkeitsamplitude ist $v \sin i$, wo v die Kreisbahngeschwindigkeit und i die (unbekannte) Inklination des Systems ist

2.2 Aufgabenstellung

Zur Bahnbestimmung, also zur Bestimmung der Parameter T_0, P, K und γ, minimieren wir die χ^2-Summe

$$\chi^2 = \sum \frac{(v_{\text{gemessen}}(t) - v(t;K,T_0,P,\gamma))^2}{\sigma^2 v_{\text{gemessen}}(t)} \tag{18}$$

mit dem Levenberg-Marquardt Algorithmus.

Wir beginnen mit den von Katja Pottschmidt und Jörn Wilms 1998 am Observatoire d'Haute Provence gemessenen Radialgeschwindigkeiten. Diese sind in vorreduzierter Form in der Datei `ohp.dat` enthalten. Jede Zeile der Datei enthält der Reihe nach:

1. Der Zeitpunkt der Messung, im sogenannten trunkierten Julianischen Datum[4], TJD = JD – 2440000.
2. Die gemessene Radialgeschwindigkeit (in km/s)
3. Den zugehörigen Fehler
4. Eine Kennung der Quelle des Datensatzes (in diesem Fall die Zahl 16).

Schreiben Sie nun eine Routine, die diese Daten in IDL einliest. Sie können am einfachsten diese Daten "auf einen Rutsch" einlesen, indem Sie sich mit dem Unix-Kommando `wc` die Länge der Datei zählen lassen, diese dann in ein geeignetes IDL Array einlesen, und dann die Variablen zuweisen. Also zum Beispiel:[5]

```
length=82
dummy=dblarr(length,4)
openr,unit,'ohp.dat',/get_lun
readf,unit,dummy
free_lun,unit
time=reform(dummy[*,0])
vrad=reform(dummy[*,1])
sigma=reform(dummy[*,2])
author=fix(reform(dummy[*,3]))
```

Plotten Sie sich nun diese Daten und den zugehörigen Fehler auf, damit Sie einen Überblick über das vorhandene Material bekommen. Zum Zeichnen der Daten und Fehlerbalken können Sie die Tübinger IDL-Unterroutine `jwoploterr` verwenden, die Sie ja schon oben gesehen haben (siehe `http://astro.uni-tuebingen.de/software/idl/aitlib/misc/jwoploterr.html`):

```
plot,time,vrad,xtitle='Time [JD]',ytitle=textoidl('v_{rad} [km/s]')
jwoploterr,time,vrad,sigma,psym=4
```

Sie sehen also, daß die Radialgeschwindigkeit einen sinusoidalen Verlauf hat, wie wir ja auch erwartet hatten.

Ab jetzt sollten Sie selbst zur Tastatur greifen, denn die restlichen Programmschritte (es sind nicht viele, aber vielleicht nicht gerade die einfachsten!) werden Sie selbst programmieren müssen!

[4]Das JD ist eine seit ca. 4700 v. Chr. durchlaufende Tageszählung, die in der Astronomie verwendet wird.

[5]Das folgende Beispiel ist im IDL Programm `datascreen.pro` enthalten, Sie müssen es also nicht selbst eintippen.

2.3 Datenmodellierung

Unsere Aufgabe ist, diese Radialgeschwindigkeiten zu modellieren. Schreiben Sie sich dafür zunächst eine Funktion `radvel`, die Gl. (17) implementiert. Diese Funktion sollte die folgenden Argumente (in dieser Reihenfolge) haben:

1. ein Array, das die Zeiten t enthält, für die wir die Radialgeschwindigkeiten berechnen wollen,
2. ein Array, in dem die Parameter unserer Fitfunktion (Gl. 17) enthalten sind, in der Reihenfolge T_0, P, K und γ,
3. Die Funktion liefert die für die Zeiten t berechneten Radialgeschwindigkeiten zurück.

Erzeugen Sie nun Zeiten zwischen `min(time)` und `max(time)` mithilfe der folgenden IDL-Anweisung

```
npt=200
test=min(time)+findgen(npt)/(npt-1)*(max(time)-min(time))
```

Berechnen Sie mit Ihrer obigen Testfunktion die zugehörigen Radialgeschwindigkeiten und plotten Sie diese über die Meßwerte (IDL-Befehl `oplot,test,vtest`). Variieren Sie T_0, P, K und γ von Hand, bis Sie eine gute Übereinstimmung mit den Meßwerten bekommen.

2.4 Auf ins Fitland

Wir haben jetzt all die Dinge gesammelt, die wir für eine erfolgreiche Fitarbeit benötigen: Wir haben unsere Meßwerte und zugehörigen Fehler, wir haben eine Fitfunktion und wir haben eine grobe Anpassung an die Daten, also die "Startwerte" für unseren Fit.

In der Vorlesung haben Sie oder werden Sie prinzipiell erfahren, wie der Levenberg-Marquardt Algorithmus zur Minimierung der χ^2-Summe funktioniert. Dieser Algorithmus ist zu kompliziert, um ihn hier selbst zu implementieren. Wir müssen daher auf die "Konserve" zurückgreifen und den Algorithmus verwenden, wie er von Craig Marquardt vom NASA Goddard Space Flight Center implementiert wurde (dieser ist übrigens nicht verwandt oder verschwägert mit einem der Hauptautoren des Levenberg-Marquardt-Algorithmus). Zur Beschreibung der Routine lesen Sie bitte `http://cow.physics.wisc.edu/~craigm/idl/down/mpfitfun.txt` oder den Header der Datei `mpfitfun.pro`, die Sie im Unterverzeichnis `/usr/local/share/rsi/local/marquardt/` finden. Wenn Sie interessiert, wie der Levenberg-Marquardt-Algorithmus funktioniert, dann sollten Sie die gute Beschreibung von Bevington & Robinson (1992) oder von Press et al. (1992) lesen.

Weil die Argumente von `mpfitfun` gelinde gesagt recht kompliziert sind, hier der Aufruf, wie Sie ihn in Ihrem Programm verwenden sollten:

```
;; das array startparm enthaelt den startwert
bestfit=mpfitfun('radvel',time,vrad,sigma,startparm,bestnorm=chi2)
dof=n_elements(time)-n_elements(astart) ;; Degree of freedoms
```

Nach dem Aufruf enthält `chi2` den Wert der Fitparameter, bei dem χ^2 minimal ist, `dof` die Zahl der Freiheitsgrade, und `bestnorm` enthält den χ^2 Wert. Das *reduzierte* χ^2, d.h. χ^2 dividiert durch die Zahl der Freiheitsgrade, sollte bei einem guten Fit ungefähr gleich eins sein.

Berechnen Sie jetzt mit den erhaltenen best-fit Werten eine neue theoretische χ^2-Kurve und plotten Sie diese über die Meßdaten.

2.5 Fehlerbestimmung – Das χ^2-Tal

Auch wenn `mpfitfun` ihnen im Prinzip im Argument `sigmaa` den statistischen Fehler der best-fit Werte zurückgibt, ist dieser nicht sehr vertrauenswürdig. Wie Sie in der Vorlesung gelernt haben, liegt dies daran, daß die Figur des χ^2-Tals sehr kompliziert sein kann und es daher nur sehr schwer möglich ist, mit einfachen mathematischen Verfahren die Fehler der Fitparameter abzuschätzen. Besser ist es, diese aus der Form des χ^2 Tals selbst zu bestimmen. Wenn wir wissen, daß die Fitparameter statistisch unabhängig sind, dann genügt es, die Variation des χ^2 Tals in einer Dimension zu betrachten. Wir variieren dazu den Fitparameter um seinen best-fit Wert herum und bestimmen für jeden Wert des Parameters ein neuen besten Fit. Beim Fitten wird der uns interessierende Parameter dabei auf dem jeweiligen Wert festgehalten und nur die anderen Parameter werden variiert. Der 1σ Fehlerbereich ist in diesem Fall gegeben durch die Punkte, bei denen $\chi^2 = \chi^2_{\rm min} + 1$, wo $\chi^2_{\rm min}$ der χ^2-Wert beim best-fit ist. Eine mathematische Begründung dieser Behauptungen finden Sie in der oben zitierten statistischen Literatur. Für astronomische Zwecke ist zusätzlich noch der Artikel von Lampton, Margon & Bowyer (1976) zu erwähnen, den Sie ebenfalls über das ADS erhalten können.

Das gerade beschriebene Verfahren ist in der Tübinger IDL-Prozedur `mpsteppar` implementiert. Zur Bedienung von Steppar lesen Sie bitte `http://astro.uni-tuebingen.de/software/idl/aitlib/fitting/mpsteppar.html`. Um für *einen* Parameter, z.B. Parameter Nummer `1`, einen Plot des χ^2 Tals zu bekommen, führen Sie den folgenden IDL Aufruf durch:

```
mpsteppar,'radvel',time,vrad,sigma,bestfit, par1ind=1,par1min=5.5,
          par1max=5.6,nstep1=10,bestnorm=chi2,/plot
```

Berechnen Sie nun die Fehler aller Ihrer Fit-Parameter. Um einen möglichst glatten Plot zu bekommen müssen Sie dabei eventuell das Keyword `nstep1` verändern. Die Keywörter `par1min` und `par1max` geben an, in welchem Bereich der interessierte Parameter variiert werden soll.

Beim Modellieren der Daten werden Sie festgestellt haben, daß der Fit nicht immer konvergiert. Das liegt daran, daß die Fit-Parameter eben *nicht* voneinander unabhängig sind. Verwenden Sie nun `steppar` und lassen Sie sich Konturen für alle (x,y) Kombinationen der Fitparameter berechnen. Der Aufruf von `mpsteppar` ist nun etwas komplizierter, da Sie Bereiche für zwei Parameter angeben müssen.

Welche Parameter sind korreliert? Wie groß ist für diese Parameter der 1σ Fehler (verglichen mit dem 1σ Fehler für einen freien Parameter?).

2.6 Verbesserung der Ephemeride: Historische Daten

Sie haben jetzt eine grobe Ephemeride bestimmt, die sich nur auf einen relativ kurzen Zeitraum bezieht. Anhand von historischen Daten kann die Genauigkeit dieser Ephemeride stark verbessert werden, ebenso wird sich dadurch Ihre Programmierzeit stark erhöhen. Daher ist der folgende Aufgabenteil freiwillig!

In der Datei `cyg.dat`, die wie die bisher verwendete Datei `ohp.dat` aufgebaut ist, findet sich eine Zusammenstellung aller historischer Radialgeschwindigkeiten von Cygnus X-1, gemessen über eine Zeitspanne von fast 30 Jahren.

Modifizieren Sie jetzt Ihre Einleseroutine von oben, so daß Sie Zugriff auf diesen Datensatz bekommen. Zeichnen Sie erneut die Daten. Da die Daten sich über sehr viele Umläufe von Cyg X-1 beziehen, können Sie nicht mehr einfach die Punkte in ihrer zeitlichen Abfolge zeichnen – versuchen Sie es, aber Sie werden sehen, daß das nicht viel bringt. Besser ist es, die Daten auf der aus den OHP Daten berechneten Ephemeride zu “falten”, d.h. die einzelnen Radialgeschwindigkeitsdaten als Funktion der Bahnphase zu zeichnen. Die Bahnphase können Sie mit der IDL Anweisung

```
phase=((time-t0) MOD p)/p
ndx=where(phase LT 0.)
IF (ndx[0] NE -1) THEN phase[ndx]=phase[ndx]+1.
```

berechnen, wo `t0` und `p` der aus den OHP-Daten berechnete Zeitpunkt der oberen Konjunktion und die Bahnperiode sind. Der obige Ausdruck stellt sicher, daß die Bahnphase im Intervall von 0 bis 1 liegt.

Verwenden Sie nun ihr bisheriges IDL-Programm, um diesen Datensatz zu modellieren und eine bessere Ephemeride zu bekommen. Aufgrund der von Ihnen oben gefundenen Korrelationen wird der Fit nicht unbedingt konvergieren. Eine bessere Stabilität des Fits erhalten Sie, wenn Sie den Nullpunkt der Ephemeride, T_0 auf die Mitte der Beobachtungsdaten legen. Überlegen Sie sich, wie Sie diesen neuen Startzeitpunkt geschickt berechnen können!

Die neue Ephemeride ist schon eine gute Verbesserung im Vergleich zu den OHP Daten alleine. Wenn Sie einzelne Datensätze mit ihrem best fit vergleichen, dann stellen Sie fest, daß diese Datensätze einen Offset haben im Vergleich zur best fit Ephemeride. Dieser Offset rührt daher, daß die Wellenlängenkalibration der Spektren, aus denen die Radialgeschwindigkeiten bestimmt wurden, nicht optimal war. Er stellt also einen systematischen Fehler der einzelnen Datensätze dar. Um den Einfluß dieses systematischen Offsets zu minimieren empfiehlt es sich, für jeden Datensatz eine eigene Offset-Geschwindigkeit γ zu fitten. Diese γs stellen also zusätzliche Fit-Parameter dar. Modifizieren Sie ihr Fitprogramm, so daß es verschiedene γs fitten kann. Dazu müssen Sie Ihre Fitfunktion `vrad` so modifizieren, daß je nach Datensatz ein anderes γ verwendet wird. Die Tübinger Modifikation von `curvefit` erlaubt es, bestimmte Keywörter an die Fitfunktion zu übergeben. Für diese Aufgabe sollten Sie das Array `author` der Fitfunktion übergeben. Die Fitfunktion bekommt dadurch das folgende Aussehen:

```
FUNCTION radvel,t,param,author=author
:
END
```

und Ihrem Aufruf von `mpcurvefit` fügen Sie das Keyword `author` hinzu:

```
:
bestfit=mpfitfun('radvel',time,vrad,sigma,startparm,
                 bestnorm=chi2,author=author)
:
```

Sie können nun das Array mit den Fit-Parametern `startparm` so erweitern, daß es einen γ-Wert für jeden Datensatz enthält und müssen natürlich `vrad` so umschreiben, daß die Geschwindigkeit für Daten aus dem Datensatz i auch mit dem Gamma-Wert für den Datensatz i versehen werden. Dafür können Sie sich die IDL-Syntax für Array-Adressierung zu Nutze machen!

Bestimmen Sie jetzt mit diesem modifizierten Programm Ihre endgültige beste Ephemeride und die zu den Fitparametern gehörigen Fehler. Beachten Sie auch dieses Mal, daß es Korrelationen zwischen den Parametern geben kann!

Literatur

Bevington, P. R., & Robinson, D. K., 1992, Data Reduction and Error Analysis for the Physical Sciences, (New York: McGraw-Hill), 2nd edition

Brandt, S., 1992, Datenanalyse, (Mannheim: BI Wissenschaftsverlag), 3. edition

Lampton, M., Margon, B., & Bowyer, S., 1976, ApJ, 208, 177

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P., 1992, Numerical Recipes in FORTRAN, (Cambridge: Cambridge Univ. Press), 2nd edition

Roe, B. P., 1992, Probability and Statistics in Experimental Physics, (New York: Springer)

Sakai, S., et al., 2000, ApJ, 529, 698