

TILASTOLLISET MENETELMÄT TÄHTITIEESSÄ

IDL-harjoitus 6, Heikki Salo 24.4.2008

- Tehdään yksinkertainen 1-ulotteinen *bulge-disk* sovitus kiekkoglaksin radiaaliselle kirkkausprofiilille $f_{obs}(r)$. Kirkkauden lisäksi tunnetaan sen mittausvirhe $\sigma_f(r)$.

- Sovitettava teoreettinen malli on muotoa

$$f(r) = f_{bulge}(r) + f_{disk}(r),$$

jossa bulge- ja kiekko-komponentit ovat

Sersic-bulge:

$$f_{bulge}(r) = f_B e^{-(r/r_B)^\beta}$$

Eksponentiaalinen kiekko:

$$f_{disk}(r) = f_D e^{-(r/r_D)}$$

Huom:

jos $\beta = 1 \Rightarrow$ *Sersic-bulge* vastaa eksponentiaalista

jos $\beta = 0.25 \Rightarrow$ *Sersic-bulge* vastaa *de Vaucouleurs'* $r^{1/4}$ lakia

Tavallisemmin parametrin β sijasta käytetään ns. *Sersic-indeksiä* $n_b = 1/\beta$

($n_b = 1$ eksponentiaalinen, $n_b = 4$ de Vaucouleurs-laki)

- Sovitus tehdään etsimällä parametrien f_b, r_b, n_b, f_d, r_d arvot jotka mimoivat suureen

$$\Delta^2 = \sum_i^N w_i [f_{obs}(r_i) - f(r_i)]^2$$

jossa N on kirkkausprofiilin pisteiden lukumäärä ja w_i on kullekin mittauspisteelle annettu paino.

- Huom: jos

i) malli on oikea (eli erotus $f_{obs} - f$ johtuu satunnaisista virheistä) ja

ii) painokertoimena käytetään $w_i = 1/\sigma_f^2$ ja virheet ovat oikein arvioitu

$\Rightarrow \Delta^2$ noudattaa χ^2 jakaumaa vapausasteella $N - M$, jossa M on sovitettavien parametrien lukumäärä (nyt $M = 5$).

Tällöin hyvälle sovitukselle $\frac{\Delta^2}{N-M}$ on ykkösen luokkaa oleva luku.

- Tehdään sovitus käytännössä IDL:n valmiilla `curvefit`-ohjelmalla, jolle annetaan alkuarvoksi havaittu profiili ja kunkin mittauspisteen paino, sekä alkuarvot sovitettavan mallin parametreille.

Lisäksi tarvitaan aliohjelma, joka palauttaa teoreettisen mallin arvot mielivaltaisilla parametrien arvoilla. `curvefit`-ohjelma etsii iteroimalla eo. erotusfunktion minimin.

Käytännön toteutus:

1) Synteettisen data luonti

Sovitusohjelman tekeminen ja tarkistaminen on helpompaa, mikäli käytetään aluksi keinotekoisesti luotua ns. synteettistä dataa.

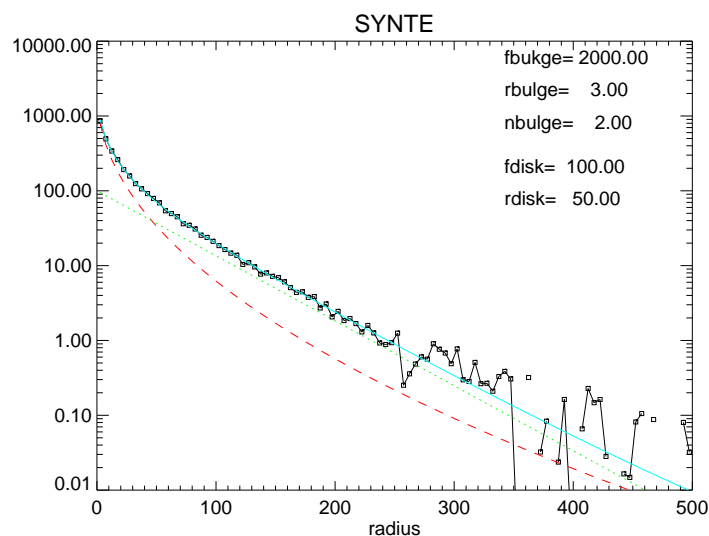
Eli tee ensimmäiseksi ohjelma, jossa annetaan arvot eo. 5 parametrille, ja joka rakentaa niistä kirkkausprofiilin.

Tarkista profiili plottaamalla, kokeile erilaisten parametrien vaikutusta.

Tallenna kirkkausprofiili (robs,fobs,sigmaobs) sekä käytetyt parametrit save-tiedostoon.

- Esimerkiksi arvoilla

```
fbulge=2000.d0
rbulge=3.d0
nbulge=2.d0
fdisk=100.d0
rdisk=50.d0
```



Symboleilla piirretyt 'havaintopisteet' on saatu lisäämällä mallifunktioiden antamaan kirkkauteen Gaussisesti jakaantunutta hälyä.

Häly on laskettu summaamalla kutakin havaintopistettä vastaavan radiaalisen vyöhykkeen yksittäisten pisteiden häly.

Tämän oletetaan koostuvan taustataivaan hälystä ('SKY') sekä kirkkausarvojen Poisson-fluktuaatiosta (häly on \sqrt{f}). (eli oletetaan että f vastaa suoraan CCD:n mittaamien elektronien lukumäärää). Kussakin renkaassa on NPIX pikseliä \Rightarrow renkaan keskimääräisen kirkkauden häly on yksittäisen pisteen häly / \sqrt{NPIX} .

```
nr=200
r=(dindgen(nr)+.5)/nr*500
.....
sky=1.
dr=r(1)-r(0)
npix=2*pi*r*dr
sigma_f=(r*0.+sky+sqrt(f))/sqrt(npix)
seed=1
fobs=f+randomn(seed,nr)*sigma_f
```

2 Sovitusohjelman teko

Tutustu seuraavalla sivulla listattuihin curvefit-ohjelman ohjeisiin (?curvefit)

Kirjoita kaksi ohjelmaa:

- pääohjelma:(esim. bd_fit.pro)
 - lukee synteettisen datan (restore-komennolla)
 - antaa alkuarvot mallin parametreille
 - asettaa curvefit-proseduuria sääteleviä muuttujia (kts ?curvefit)
 - kutsuu curvefit-proseduuria
 - plottaa sovitustulokset, tarkastelee mallin onnistumista saadun χ^2 arvon perusteella.
- aliohjelma (esim. bd_model.pro)
 - palauttaa mallin arvon, sille syötetyillä parametrien arvoilla (käytetään aluksi /noderivative keywordia, jolloin ei tarvita analyttisiä derivaattojen lausekkeita)

Kokeile sovituksen toimivuutta esim. käyttämällä eo.parametrejä synteettisessä profiilissa. Aloita iteraatio väärillä arvoilla: kerro oikeat arvot tekijällä 1.2

Esimerkki iteraation suppenemisesta (tol=1d-4)

ITE	fb	rb	nb	fd	rd	chi2norm
0	2400.0	3.600	2.400	120.0	60.0	59525.820
1	2008.5	3.033	2.349	20.4	52.6	2877.650
2	2410.7	1.773	2.387	59.7	35.4	487.147
3	2666.9	2.022	2.260	56.0	58.6	110.632
4	2162.5	2.551	2.133	79.1	46.9	80.580
5	2023.3	2.957	2.025	91.1	51.4	3.807
6	1984.4	3.054	1.994	98.4	50.2	1.144
7	1974.2	3.078	1.987	99.4	50.2	1.115
8	1973.3	3.080	1.986	99.5	50.2	1.115

curvefit.pro

```
;
; PURPOSE:
;   Non-linear least squares fit to a function of an arbitrary
;   number of parameters. The function may be any non-linear
;   function. If available, partial derivatives can be calculated by
;   the user function, else this routine will estimate partial derivatives
;   with a forward difference approximation.
;
; CATEGORY:
;   E2 - Curve and Surface Fitting.
;
; CALLING SEQUENCE:
;   Result = CURVEFIT(X, Y, Weights, A, SIGMA, FUNCTION_NAME = name, $
;                   ITMAX=ITMAX, ITER=ITER, TOL=TOL, /NODERIVATIVE)
;
; INPUTS:
;   X: A row vector of independent variables. This routine does
;       not manipulate or use values in X, it simply passes X
;       to the user-written function.
;
;   Y: A row vector containing the dependent variable.
;
;   Weights: A row vector of weights, the same length as Y.
;             For no weighting,
;               Weights(i) = 1.0.
;             For instrumental (Gaussian) weighting,
;               Weights(i)=1.0/sigma(i)^2
;             For statistical (Poisson) weighting,
;               Weights(i) = 1.0/y(i), etc.
;
;             For no weighting, set Weights to an undefined variable.
;
;   A: A vector, with as many elements as the number of terms, that
;       contains the initial estimate for each parameter. IF A is double-
;       precision, calculations are performed in double precision,
;       otherwise they are performed in single precision. Fitted parameters
;       are returned in A.
```

```

;
; KEYWORDS:
; FITA:  A vector, with as many elements as A, which contains a zero for
;        each fixed parameter, and a non-zero value for elements of A to
;        fit. If not supplied, all parameters are taken to be non-fixed.
;
;        FUNCTION_NAME:  The name of the function (actually, a procedure) to
;        fit. IF omitted, "FUNCT" is used. The procedure must be written as
;        described under RESTRICTIONS, below.
;
;        ITMAX:  Maximum number of iterations. Default = 20.
;        ITER:   The actual number of iterations which were performed
;        TOL:    The convergence tolerance. The routine returns when the
;                relative decrease in chi-squared is less than TOL in an
;                interation. Default = 1.e-3.
;        CHI2:   The value of chi-squared on exit (obselete)
;
;        CHISQ:  The value of reduced chi-squared on exit
;        NODERIVATIVE:  IF this keyword is set THEN the user procedure will not
;                be requested to provide partial derivatives. The partial
;                derivatives will be estimated in CURVEFIT using forward
;                differences. IF analytical derivatives are available they
;                should always be used.
;
;        DOUBLE = Set this keyword to force the calculation to be done in
;                double-precision arithmetic.
;
;        STATUS: Set this keyword to a named variable in which to return
;                the status of the computation. Possible values are:
;                STATUS = 0: The computation was successful.
;                STATUS = 1: The computation failed. Chi-square was
;                increasing without bounds.
;                STATUS = 2: The computation failed to converge in ITMAX
;                iterations.
;
;        YERROR: The standard error between YFIT and Y.
;
; OUTPUTS:
;        Returns a vector of calculated values.
;        A:  A vector of parameters containing fit.
;

```

```

; OPTIONAL OUTPUT PARAMETERS:
;   Sigma: A vector of standard deviations for the parameters in A.
;
;   Note: if Weights is undefined, then you are assuming that
;   your model is correct. In this case, SIGMA is multiplied
;   by  $\text{SQRT}(\text{CHISQ}/(N-M))$ , where N is the number of points
;   in X and M is the number of terms in the fitting function.
;   See section 15.2 of Numerical Recipes in C (2nd ed) for details.
;
; COMMON BLOCKS:
;   NONE.
;
; SIDE EFFECTS:
;   None.
;
; RESTRICTIONS:
;   The function to be fit must be defined and called FUNCT,
;   unless the FUNCTION_NAME keyword is supplied. This function,
;   (actually written as a procedure) must accept values of
;   X (the independent variable), and A (the fitted function's
;   parameter values), and return F (the function's value at
;   X), and PDER (a 2D array of partial derivatives).
;   For an example, see FUNCT in the IDL User's Libaray.
;   A call to FUNCT is entered as:
;   FUNCT, X, A, F, PDER
; where:
;   X = Variable passed into CURVEFIT. It is the job of the user-written
;   function to interpret this variable.
;   A = Vector of NTERMS function parameters, input.
;   F = Vector of NPOINT values of function,  $y(i) = \text{func}(x)$ , output.
;   PDER = Array, (NPOINT, NTERMS), of partial derivatives of func.
;           PDER(I,J) = DERivative of function at ith point with
;           respect to jth parameter. Optional output parameter.
;           PDER should not be calculated IF the parameter is not
;           supplied in call. IF the /NODERIVATIVE keyword is set in the
;           call to CURVEFIT THEN the user routine will never need to
;           calculate PDER.
;
; PROCEDURE:
;   Copied from "CURFIT", least squares fit to a non-linear
;   function, pages 237-239, Bevington, Data Reduction and Error

```

```

; Analysis for the Physical Sciences. This is adapted from:
; Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear
; Parameters", J. Soc. Ind. Appl. Math., Vol 11, no. 2, pp. 431-441,
; June, 1963.
;
; "This method is the Gradient-expansion algorithm which
; combines the best features of the gradient search with
; the method of linearizing the fitting function."
;
; Iterations are performed until the chi square changes by
; only TOL or until ITMAX iterations have been performed.
;
; The initial guess of the parameter values should be
; as close to the actual values as possible or the solution
; may not converge.
;
; EXAMPLE: Fit a function of the form  $f(x) = a * \exp(b*x) + c$  to
; sample pairs contained in x and y.
; In this example,  $a=a(0)$ ,  $b=a(1)$  and  $c=a(2)$ .
; The partials are easily computed symbolically:
;  $df/da = \exp(b*x)$ ,  $df/db = a * x * \exp(b*x)$ , and  $df/dc = 1.0$ 
;
; Here is the user-written procedure to return F(x) and
; the partials, given x:
;
; pro gfunct, x, a, f, pder      ; Function + partials
;   bx = exp(a(1) * x)
;   f= a(0) * bx + a(2)          ;Evaluate the function
;   IF N_PARAMS() ge 4 THEN $    ;Return partials?
;   pder= [[bx], [a(0) * x * bx], [replicate(1.0, N_ELEMENTS(f))]]
; end
;
;   x=findgen(10)                ;Define indep & dep variables.
;   y=[12.0, 11.0,10.2,9.4,8.7,8.1,7.5,6.9,6.5,6.1]
;   Weights=1.0/y                ;Weights
;   a=[10.0,-0.1,2.0]            ;Initial guess
;   yfit=curvefit(x,y,Weights,a,sigma,function_name='gfunct')
;   print, 'Function parameters: ', a
;   print, yfit
; end
;

```