Least Squares Fitting and Equation Solving with MPFIT

Craig Markwardt
University of Maryland and NASA’s Goddard Spaceflight Center

http://purl.com/net/mpfit
2009-04-15
Data and Modeling

• Least squares fitting
• Act of hypothesis testing
• Simple Hypothesis:
  – data \( \{x_i, y_i\} \) are consistent with model \( f(x_i, p) \) to within the measurement uncertainties \( \sigma_i \)
  – where \( \{x_i\} \) is the independent variable and \( \{y_i\} \) is the dependent variable
  – \( f(x_i, p) \) is the model function, which is parameterized by parameters \( p \)
Fitting Example

![Graph showing data values and a model fit.](image)
We Often “Know” When a Fit is Bad ...
... because the Residuals are Large

Residuals $r_i = \text{Data} - \text{Model}$
Summary

• At its basic level a “good fit” should minimize the residuals, $r_i$, between the data and model.

• To balance the measurements with large and small uncertainty appropriately, our “scaled” residual looks like this,

$$r_i = \frac{y_i - f(x_i, p)}{\sigma_i}$$

“residual”

“data” $y_i$

“model” $f(x_i, p)$

“measurement uncertainty” $\sigma_i$
• In an ideal world, we would want a perfect match between data and model, i.e. solve the following equations simultaneously for $M$ data points:

\[
\begin{align*}
    r_0 & = 0 \\
    r_1 & = 0 \\
    \vdots & \\
    r_M & = 0
\end{align*}
\]
In Practice this is not possible because of measurement noise…
• Instead we solve the system of equations in a “least squares” sense
• Define the chi-square statistic as the sum of the squared residuals,

\[ \chi^2 = \sum_{i=1}^{M} r_i^2 \]

and minimize this statistic.
Statistical Background

• The chi-square statistic is important and well known
• For gaussian statistics, chi-square minimization should be equivalent to the maximum likelihood parameter estimate
• In the real world, an optimal chi-square value

\[ \chi^2 \sim M \]

each data point contributes one degree of freedom.
Chi-Square Test

- Good fit:
  \[ \chi^2 \sim 19.9 \]
  for 20 data points

- Bad fit:
  \[ \chi^2 \sim 756 \]
  for 20 data points
Existing Fitting Tools in IDL

• General non-linear:
  – CURVEFIT – Bevington algorithm (vectorized)
  – LMFIT – Numerical recipes (not vectorized)
• Specialized:
  – LINFIT – linear (y = ax + b)
  – POLY_FIT – polynomial
  – SVDFIT – linear combinations
  – GAUSSFIT – peak fitting
Introducing MPFIT

- Powerful fitting engine based on MINPACK-1 (Moré and collaborators; http://netlib.org/minpack/)
- Robust factorization and stepping algorithms
- Innovations:
  - Private data to user functions (FUNCTARGS)
  - Parameter upper and lower bounds (PARINFO)
  - User chooses who computes derivatives
  - Control over iteration, console output, stopping tolerances
• MPFIT is the main fitting engine, can solve any chi-square minimization problem expressable as

$$\min_{p} \sum_{i=1}^{M} r_i(p)^2$$

• Classic least squares fitting looks like this,

$$r_i(p) = \frac{y_i - f(x_i, p)}{\sigma_i}$$

but we will warp this to other uses later. Dimensionality of x or y are not important!
The MPFIT Family of Routines

• Core fitting engine: MPFIT
• 1D Convenience routines:
  – MPFITFUN
  – MPFITEXPR – command line fitting
  – MPFITPEAK – peak fitting
  – MPCURVEFIT – drop-in replacement for CURVEFIT
• 2D Convenience routines:
  – MPFIT2DFUN
  – MPFIT2DPEAK – peak fitting
Simple Example: MPFITEXPR

• Basic command line fitting, great for diagnosing data on the fly
• You supply a model function expression
  \( \text{expr} = 'P[0] + X*P[1] + X^2*P[2]' \)
• And then call MPFITEXPR
  "x" "y" "errors" "start values"
  \( p = \text{mpfitexpr(expr, xx, ys, ye, \ [1,1,1d], ...)} \)
• Demo (mpfit_expr.pro)
MPFITFUN, MPFIT2DFUN

• For general fitting of a known model function, you will probably graduate to MPFITFUN or MPFIT2DFUN quickly

  FUNCTION PARABOLA, X, P
  RETURN, F
  END
For More on the Basics

• Setting parameter boundaries (PARINFO)
• Passing private information (FUNCTARGS)
• Retrieving best-fit model function and chi-square value (YFIT and BESTNORM)

➤ See my website
More Advanced Topics

- Multi-dimensional data
- Complicated constraints
- Equation solving
Example: Line Fit, Errors in X and Y

- Numerical Recipes technique of modified chi-square value:
  \[ \chi^2 = \sum \frac{(y_i - a - bx_i)^2}{\sigma_{yi}^2 + b^2\sigma_{xi}^2} \]

- This looks exactly like an equation MPFIT will solve:
  \[ \chi^2 = \sum_{i=1}^{M} r_i^2 \]

- If we set
  \[ r_i = \frac{y_i - a - bx_i}{\sqrt{\sigma_{yi}^2 + b^2\sigma_{xi}^2}} \]
Fitting with X and Y Errors

• LINFITEX: a user function which implements this technique
  \[ \text{resid} = \frac{y - f}{\sqrt{\sigma_y^2 + (b\sigma_x)^2}} \]

• We call MPFIT with this user function
  \[
p = \text{mpfit('LINFITEX', [1d, 1d], FUNCTARGS={X: XS, Y: YS, SIGMA_X: XE, SIGMA_Y: YE}, ...)}
  \]

• Demo (MPFIT_LINFITEX)
Fitting In Two Dimensions

- **Example problem:**
  - A meteor appeared and exploded near a remote Alaskan town
  - The townspeople located several meteorite fragments and reports to us at local observing station
- **Goal:** determine point of meteor explosion from fragment locations
Centroid Point

• We approximate the explosion point as the centroid, \((x_c, y_c)\), the point which has the smallest joint distance from all fragments

\[
\min(\text{dist}^2) = \sum_{i=1}^{M} (x_i - x_c)^2 + (y_i - y_c)^2
\]

• Again, this looks exactly like a problem MPFIT can solve, except with \textbf{twice} as many residuals

\[
\chi^2 = \sum_{i=1}^{M} r_{xi}^2 + r_{yi}^2
\]
MPFIT Model

• MPFIT will understand this if the two sets of residuals are appended:

```plaintext
FUNCTION MPFIT_CENT, p, X=X, Y=Y
   resid_x = (x-p[0])
   resid_y = (y-p[1])
   resid = [resid_x, resid_y]
   return, resid
END
```

• Demonstration (MPFIT_METEORITE)
Metropolitan Fragment Locations

Best Fit
Centroid
Adding Complex Constraints

• New Meteorite Information!
  – Station sensors detected a light flash and sonic boom, allowing range to be determined to 9.56 km
• We now know the explosion point lies somewhere on the circle
• How can we add this new information?
MPFIT Model

• Express this new parameter constraint as,
\[(x_c - x_s)^2 + (y_c - y_s)^2 = R^2 = (9.56\text{km})^2\]

• But how can we force MPFIT to honor this constraint?

• Trick: re-write the constraint to be another “residual”, which should equal 0, and MPFIT will solve it along with all the other residuals
\[
(r_0 = 0, r_1 = 0, \ldots, r_M = 0)
\]

New Constraint!

\[
(x_c - x_s)^2 + (y_c - y_s)^2 - R^2 = 0
\]
MPFIT Model

• MPFIT will understand this we now append **three** sets of residuals:

  resid_x = (x-p[0])
  resid_y = (y-p[1])
  resid_r = (p[0]-xs)^2 + (p[1]-ys)^2 - radius^2
  resid = [resid_x, resid_y, resid_r/tolerance]

• Demonstration (MPFIT_Meteorite)
Meteorite Fragment Locations

New Best Fit Centroid
Two Constraints!

- Even Newer Meteorite Information!
  - A station staffer observed the flash visually and obtained a sight line of 48 deg north of east
- We now know a second line of constraint
- How can we add this new information?
\[ y_c = \tan(\theta)(x_c - x_s) + y_s \]

\[ (x_c - x_s)^2 + (y_c - y_s)^2 - R^2 = 0 \]

\[ \tan(\theta)(x_c - x_s) + y_s - y_c = 0 \]
MPFIT Model

• MPFIT will understand this if we now append a **fourth** residual function:

```
    resid_s = tan(sight_angle!*dtor)*(p[0]-xs) + ys - p[1]
    resid = [..., resid_s / sight_pos_tolerance]
```

• Demonstration (MPFIT_METEORITE)
New Best Fit Centroid
Equation Solving

• The two last constraints in this problem are alone enough completely specify the explosion point

\[
(x_c - x_s)^2 + (y_c - y_s)^2 - R^2 = 0
\]

\[
\tan(\theta)(x_c - x_s) + y_s - y_c = 0
\]

• We don’t even need data!
MPFIT Model

• MPFIT will understand this if we keep only the last two constraint equations:

\[
\text{resid}_r = (p[0]-xs)^2 + (p[1]-ys)^2 - \text{radius}^2
\]
\[
\text{resid}_s = \tan(\text{sight\_angle} \times \! \text{dtor}) \times (p[0]-xs) + \text{ys} - p[1]
\]
\[
\text{resid} = [\text{resid}_r/\text{tol}, \text{resid}_s / \text{sight\_pos\_tolerance}]
\]

• Demonstration (MPFIT_METEORITE)
Meteorite Fragment Locations

Equation Solution
Caveats

• This technique depends on using MPFIT, not the other more “convenient” routines
• If MPFIT finds a solution, that does not prove it is unique (could be multiple solutions!)
  – Depends on initial conditions
• If equations are poorly behaved, MPFIT may get stuck
  – Need to adjust derivatives parameters via PARINFO
Take Away Messages

• MPFIT is a power equation solver
• Fitting M points is just solving M equations
• Adding new constraints is straightforward, just add new “residual” equations
  – i.e. re-express constraint as \( \frac{\text{function}}{\text{tolerance}} = 0 \)
• Other estimators like “least absolute deviation” (LAD) and Poisson likelihood can be solved by warping their equations to look like a sum of squared residuals
Getting MPFIT

- IDL version: [http://purl.com/net/mpfit](http://purl.com/net/mpfit)
- C version at same site