MPFIT(IDL의 대표 Curve Fitting 패키지)

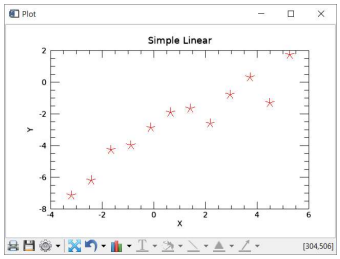
Intro

물론이죠. IDL에는 다양한 Fitting 함수들이 이미 존재합니다. CURVEFIT, LINFIT, LMFIT, COMFIT, GAUSSFIT, LADFIT, POLY_FIT, REGRESS, SVDFIT 등등 모두 Fitting 함수들입니다. 그런데, 대부분의 IDL 사용자들은 MPFIT을 추천합니다. 추가 설치를 해야하는 것이 유일한 약점이지만, 강력하고, 더 편리합니다.
아래, Craig Markwardt 박사님의 공개 페이지에서 모든 것을 제공하고 있습니다.
http://cow.physics.wisc.edu/~craigm/idl/fitting.html
200kb도 되지 않는 mpfit.zip 또는 mpfit.tar.gz 파일을 다운받아서, IDL 경로에 풀어 놓으면, 설치 끝!

MPFITFUN과 MPFITEXPR

두 함수가 하는 일은 완전히 같습니다. Fitting의 모델을 IDL의 함수로 만들어 쓸 것인지(MPFITFUN), 문자열의 수식으로 만들어 쓸 것인지(MPFITEXPR), 상황에 따라 편한 것을 쓰면 됩니다. 내부적으로는 MPFIT이라는 함수로 똑같이 실행됩니다.
쉬운 예제로 두 방식을 비교해 보겠습니다.

```
X = [-3.20,  4.49, -1.66,   0.64, -2.43, -0.89,$
     -0.12,  1.41,  2.95,   2.18, 3.72, 5.26]
Y = [-7.14, -1.30, -4.26, -1.90, -6.19, -3.98, $
     -2.87, -1.66, -0.78, -2.61, 0.31, 1.74]
gr1 = plot(X, Y, '*', color='red', xtitle='X', $
           ytitle='Y', title='Simple Linear')
```



MPFITEXPR

```
expr = 'P[0]*X + P[1]'
measure_err=sqrt(abs(Y))
start=[1.0, -4.0]            ;초기값

resulta = MPFITEXPR(expr, X, Y, $
                measure_error,start)
```

실행 결과는 아래와 같습니다. 3회의 Iteration을 거쳐서 P[0]=0.867345, P[1]=-3.44598을 계산했습니다.

```
Iter  1   CHI-SQUARE =   11.887898      DOF = 10
                P(0) =            1.00000
                P(1) =           -4.00000
Iter  2   CHI-SQUARE =   8.3086243      DOF = 10
                P(0) =           0.867325
                P(1) =           -3.44601
Iter  3   CHI-SQUARE =   8.3086233      DOF = 10
                P(0) =           0.867345
                P(1) =           -3.44598
```

계산 결과는 좌변 resulta에 저장됩니다.

```
IDL> print, resulta[0], resulta[1]
    0.867345    -3.44598
```

식을 y=aX+b로 세웠습니다. 이 식을 MPFITEXPR에 넘겨주기 위한 식으로 쓰는 것은 다음 규칙만 지키면 됩니다.

- 독립변수는 X로 씁니다.
- 구하고자 하는 계수는 모두 P의 배열로 씁니다.

P[0], P[1]은 IDL의 배열 표기에서 허용하는 P(0), P(1)로 써도 무방합니다.

MPFITFUN

식을 IDL의 Function으로 구성합니다. 번거로울 수 있으나 좀 더 복잡한 모델을 정의하기에 좋습니다.

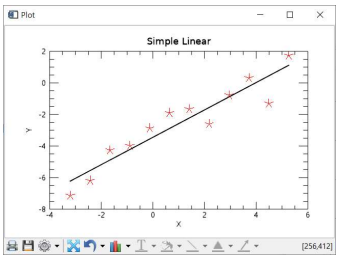
```
function my_simple_model, X, P
  return, P[0]*X + P[1]
end
```

이와 같이 Function을 만들고 my_simple_model.pro로 저장 후 compile을 하면 MPFITFUN에서 다음과 같이 호출하여 사용할 수 있습니다.

```
resultb = MPFITFUN( 'my_simple_model', X, Y, $
                    measure_error, start)
```

출력되는 내용 및 resultb에 저장되는 값이 resulta의 값과 똑같습니다. 그러므로 MPFITEXPR과 MPFITFUN은 본인 취향에 맞게 쓰면 됩니다. 측정오차(measure_error)는 X, Y와 같은 크기의 배열이어야 하고, 초기값(start)은 구하고자 하는 계수의 개수와 같아야 합니다. Fitting이 잘 되었는지 확인해 보겠습니다.

```
YFIT=resulta[0]*X+resulta[1]
gr2 = plot(X, YFIT, /OVERPLOT)
```



마지막에 설명할 YFIT 키워드를 이용하면 YFIT을 따로 계산할 필요는 없습니다.

파라미터 고정 또는 범위 지정

위 예제에서 Y 절편 P(1)의 값을 -3.5로 고정하고 Fitting해 볼 수 있을까요? 기울기 P(0)이 1 이상이 되도록 하한을 정하고 Fitting해 볼 수 있을까요?

MPFIT에는 PARINFO 라는 키워드가 있어 이러한 설정을 할 수 있습니다. 파라미터 설정은 구조체로 하는데 이렇게 생겼습니다.

{fixed:0, limited:[0,0], limits:[0.0, 0.0]}

사실 이 구조체에는 더 많은 필드가 있는데, 위 세 개만 주로 쓰입니다. 파라미터가 2개이니, 2개를 생성해야죠.

```
pi = replicate({fixed:0, limited:[0,0], $
                limits:[0., 0.]}, 2)
```

P(1)을 고정하고 초기값을 -3.5로 줄 수 있습니다.

```
pi[1].fixed=1          ;P(1)을 고정한다는 의미로 1
start=[1.0, -3.5]

resulta = MPFITEXPR(expr, X, Y, $
  measure_error, start, PARINFO=pi)
```

```
Iter  1   CHI-SQUARE =   9.8778963     DOF = 11
                P(0) =            1.00000
                P(1) =           -3.50000
Iter  2   CHI-SQUARE =   8.3390856     DOF = 11
                P(0) =           0.874192
                P(1) =           -3.50000
```

계산 과정에서 P(1)은 -3.5로 고정되어 있는 것을 확인할 수 있습니다. MPFITFUN도 똑같이 사용하면 됩니다.

```
resultb = MPFITFUN( 'my_simple_model', X, Y, $
          measure_error, start, PARINFO=pi)
```

이번에는 P(1)은 고정하지 않고, P(0)의 하한을 1.0으로 지정해 보겠습니다.

```
pi[1].fixed=0  ; 고정해제(디폴트값으로 되돌림)
pi[0].limited=[1, 0] ;하한지정(1), 상한지정 없음(0)
pi[0].limits = [1.0, 0]
    ; 하한 1.0, 상한은 지정하지 않으므로 의미 없는 값
start=[3., 0.]
resulta = MPFITEXPR(expr, X, Y, $
  measure_error, start, PARINFO=pi)
```

```
Iter  1   CHI-SQUARE =   774.53711     DOF = 10
                P(0) =            3.00000
                P(1) =           0.000000
Iter  2   CHI-SQUARE =   11.275211     DOF = 10
                P(0) =            1.00000
                P(1) =           -3.23143
Iter  3   CHI-SQUARE =   9.7962227     DOF = 10
                P(0) =            1.00000
                P(1) =           -3.58247
```

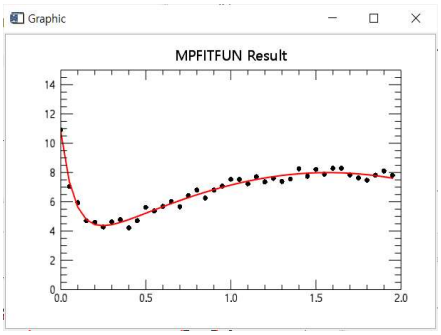
- fixed : 값을 고정할 것인가? (초기값 그대로)
- limited : [하한, 상한]을 지정할 것인가?
- limits : [하한, 상한] 값

계수가 두 개 밖에 없는 계산에서 하나의 계수를 속박하는 것은 Fitting이라고 하기엔 너무 쉬운 계산 상황이지만, 이 예제에서는 P(0)이 시작값 3.0에서 1.0으로 내려오고 그 이하로 내려가지 않는 것만 확인하면 됩니다.

비선형 함수의 Fitting

계산이니까, 얼마든지 복잡한 모델도 Fitting을 합니다. 이번에는 약간 더 긴 수식으로 Fitting을 해 보겠습니다. 예제는 이상우의 IDL 블로그에 있는 "비선형 함수의 근사"에 있는 내용입니다. 원문 링크는 다음과 같습니다.

http://blog.daum.net/swrush/474




```
FUNCTION my_nl_funct, X, P
  Y = P[0]*EXP(P[1]*X)+P[2]+P[3]*SIN(X)
  RETURN, Y
END

PRO test_nonlinear_fitting
  X = FINDGEN(40)/20.0
  Y = 8.7*EXP(-12.9*X)+2.6+5.3*SIN(X)+ $
        RANDOMU(-1, 40)-0.5
  errors = MAKE_ARRAY(40, VALUE=1.0)
  start = [10.0, -5.1, 6.0, 4.0]
  yrn = [0, 15]

  win = WINDOW(DIMENSIONS=[600, 500])
  pl = PLOT(X, Y, YRANGE=yrn, SYMBOL='circle', $
          /SYM_FILLED, LINESTYLE=6, /CURRENT)
  coefs = MPFITFUN('my_nl_funct', X, Y, errors, $
                   start, NITER=n_iter)
  chisq = TOTAL((Y-my_nl_funct(X, coefs))^2 * $
                          ABS(1D/errors^2) )

  HELP, coefs   &   PRINT, coefs   &   PRINT, start
  PRINT, chisq  &   PRINT, n_iter

  plo = PLOT(X, my_nl_funct(X, coefs), $
        COLOR='red', THICK=2, FONT_SIZE=11, $
        TITLE='MPFITFUN Result',  /OVERPLOT)
END
```

$y = p_0 e^{p_1 x} + p_2 + p_3 \sin x$ 를 모델로 MPFITFUN을 실행하는 일반적인 과정입니다. 결과는 다음과 같습니다.

```
Iter   1   CHI-SQUARE =   465.64874     DOF = 36
                P(0) =            10.0000
                P(1) =           -5.10000
                P(2) =            6.00000
                P(3) =            4.00000
      ---------------- 중략 -------------
Iter   7   CHI-SQUARE =   2.8507075     DOF = 36
                P(0) =            8.17166
                P(1) =           -11.9435
                P(2) =            2.62498
                P(3) =            5.37054

COEFS           FLOAT     = Array[4]  ;4개의 파라미터
   8.17166  -11.9435   2.62498   5.37054  ;coefs
   10.0000  -5.10000   6.00000   4.00000  ;start
     2.8507078                           ;chisq
       7                                 ;n_iter
```

자주 사용하는 키워드

MPFIT의 모든 키워드를 보려면, 소스코드의 주석문을 보세요. 자주 사용되는 키워드는 다음과 같습니다.

- /QUIET : 계산 수행 과정을 화면에 출력하지 않음
- NITER=a : a 변수로 iteration 회수를 리턴.
- YFIT=Yf : Yf 변수로 모델에 의한 적합값을 리턴.
- WEIGHTS= : 가중치를 지정.
- PERROR=pe : 각 파라미터의 1σ 에러를 리턴
- COVAR=c : c 변수로 공분산 행렬을 리턴.
- BESTNORM : TOTAL(DEVIATES ^ 2)