

線形最小2乗法

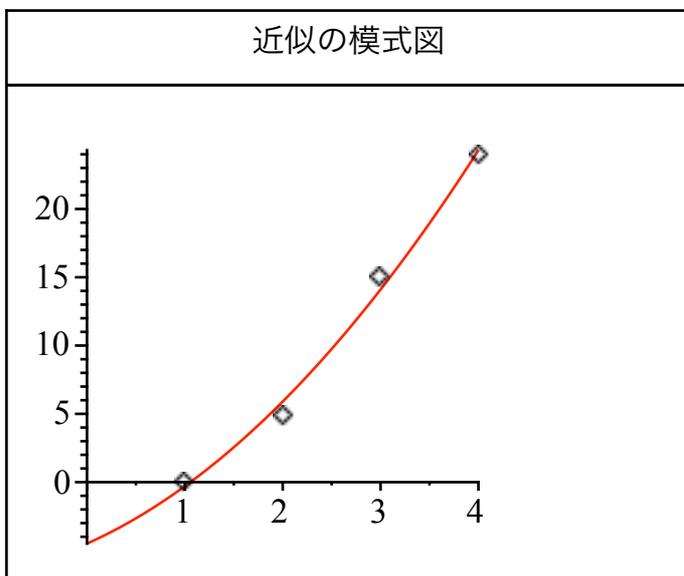
—数値計算(07/12/7)—

関西学院大学理工学部 西谷滋人

Copyright ©2007 by Shigeto R. Nishitani

最乗2乗法の原理

前回の授業では、データに多項式を完全にフィットする補間についてみた。今回は、近似的にフィットする最小二乗法について詳しくみていく。図のようなデータに直線をフィットする場合を考えよう。



コマンドleastsquareによるfitting(3変数の例)

```
> restart:  
X:=[1,2,3,4]:  
Y:=[0,5,15,24]:  
> with(plots):with(linalg):with(stats):  
> l1:=pointplot(transpose([X,Y]),symbolsize=30):  
> eq_fit:= fit[leastsquare([x,y], y=a*x^2+b*x+c, {a,b,c})](  
[X, Y]);  

$$eq\_fit := y = x^2 + \frac{16}{5}x - \frac{9}{2}$$
  
> f1:=unapply(rhs(eq_fit),x);  

$$f1 := x \rightarrow x^2 + \frac{16}{5}x - \frac{9}{2}$$
  
> p1:=plot(f1(x),x=0..4):  
> display(p1,l1);
```

最乗2乗法の原理

もっとも簡単な例で原理を解説する。近似関数として、

$$F(x) = a_0 + a_1 x$$

という直線近似を考える。もっともらしい関数はN点の測定データとの差 $d_i = F(x_i) - y_i$ を最小にすればよさそうであるが、これはプラスマイナスですぐに消えて不定になる。そこで、

$$\chi^2 = \sum_i^N d_i^2 = \sum_i^N (a_0 + a_1 x_i - y_i)^2$$

という関数を考える。この χ^2 (カイ二乗)関数が、 a_0, a_1 をパラメータとして変えた時に最小となる a_0, a_1 を求める。これは、それらの微分がそれぞれ0となる場合である。これは χ^2 の和 Σ (sum)の中身を展開し、

$$\chi^2 =$$

a_0, a_1 でそれぞれ微分すれば

$$\frac{\partial \chi^2}{\partial a_0} =$$

$$=$$

$$\frac{\partial}{\partial a_1} \chi^2 =$$

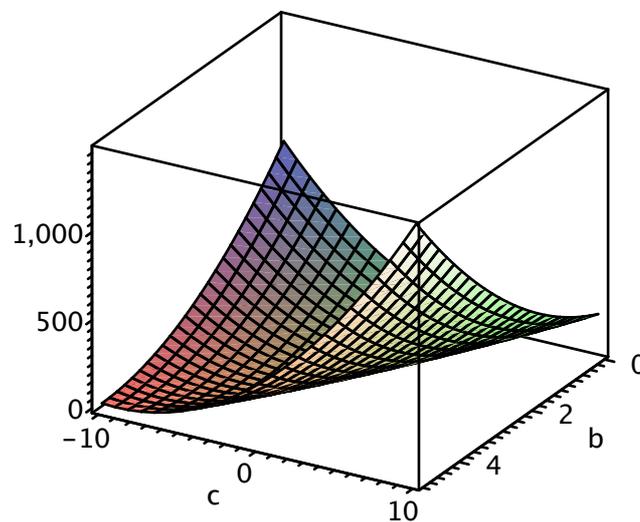
という a_0, a_1 を未知変数とする2元の連立方程式が得られる。これは前に説明した通り
逆行列で解くことができる。

▼ χ^2 乗の極小値から(3変数の例)

```
> restart;
X:=[1,2,3,4]:
Y:=[0,5,15,24]:
f1:=x->a*x^2+b*x+c:
> S:=0;
for i from 1 to 4 do
  S:=S+(f1(X[i])-Y[i])^2;
end do:
                                     S:=0

> fS:=unapply(S, (a,b,c));
fS:=(a,b,c)→(a+b+c)^2+(4a+2b+c-5)^2+(9a+3b+c-15)^2
+ (16a+4b+c-24)^2

> plot3d(subs(aa=1, fS(aa,b,c)), b=0..5, c=-10..10);
```



```
> eqs:={diff(expand(S), a)=0,
diff(expand(S), b)=0,
diff(expand(S), c)=0};
eqs:={-1078+60c+708a+200b=0, -302+200a+20c+60b=0, -88
+60a+20b+8c=0}

> solve(eqs, {a,b,c});
{a=1, b=16/5, c=-9/2}
```

▼ 正規方程式による解

より一般的な場合の最小二乗法の解法を説明する。先程の例では1次の多項式を近似関数とした。これをより一般的な関数、例えば、 \sin , \cos , \tan , \exp , \sinh などとする。これを線形につないだ関数を

$$F(X) = a_0 \sin(x) + a_1 \cos(x) + a_2 \exp(-x) + a_3 \sinh(x) + \dots = \sum_{k=1}^M a_k X_k(x)$$

ととる。実際には、 $X_k(x)$ はモデルや、多項式の高次項など論拠のある関数列をとる。これらを基底関数(base functions)と呼ぶ。ここで線形といているのは、パラメータ a_k について線形という意味である。この応用を次節で取り上げる。

このより一般的な基底関数を使っても、 χ^2 関数は

$$\chi^2 = \sum_{i=1}^N (F(x_i) - y_i)^2 = \sum_{i=1}^N \left(\sum_{k=1}^M a_k X_k(x_i) - y_i \right)^2$$

と求めることができる。この関数を、 a_k を変数とする関数とみなす。この関数が最小値を取るのは、 χ^2 を M 個の a_k で偏微分した式がすべて0となる場合である。これは

$$\sum_{i=1}^N \left(\sum_{j=1}^M a_j X_j(x_i) - y_i \right) X_k(x_i) = 0$$

ここで、 $k=1, \dots, M$ の M 個の連立方程式である。この連立方程式を最小二乗法の正規方程式(normal equations)と呼ぶ。

上記の記法のままでは、ややこしいので、行列形式で書き直す。 $N \times M$ で、各要素を

$$A_{ij} = X_j(x_i)$$

とする行列 A を導入する。この行列は、

$$A = \begin{bmatrix} X_1(x_1) & X_2(x_1) & \dots & X_M(x_1) \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}$$

となる。これをデザイン行列と呼ぶ。すると先程の正規方程式は、

$$A^T \cdot A \cdot a = A^T \cdot y$$

で与えられる。 A^T は行列 A の転置(transpose)を意味し、得られた行列は、 $M \times N$ である。 a, y はそれぞれ、

$$a = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix}$$

である。 $M=25, N=3$ として行列の次元だけで表現すると、

$$\begin{bmatrix} \dots & \dots & \dots \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ \dots & \dots & \dots \end{bmatrix} \begin{bmatrix} \dots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} \dots & \dots & \dots \\ \dots & \dots & \dots \\ \dots & \dots & \dots \end{bmatrix} \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}$$

となる。これは少しの計算で 3×3 の逆行列を解く問題に変形できる。

▼ 正規方程式(normal equations)

```
> restart;
x:=[1,2,3,4]:
y:=[0,5,15,24]:
ff:=x->a[1]+a[2]*x+a[3]*x^2:
> with(LinearAlgebra):
Av:=Matrix(1..4,1..3):
> ff:=(x,i)->x^(i-1);
for i from 1 to 3 do
for j from 1 to 4 do
Av[j,i]:=ff(X[j],i);
end do;
end do;
Av;
```

$$ff := (x, i) \rightarrow x^{i-1}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 4 & 16 \end{bmatrix}$$

```
> Ai:=MatrixInverse(Transpose(Av).Av);
```

$$A_i := \begin{bmatrix} \frac{31}{4} & -\frac{27}{4} & \frac{5}{4} \\ -\frac{27}{4} & \frac{129}{20} & -\frac{5}{4} \\ \frac{5}{4} & -\frac{5}{4} & \frac{1}{4} \end{bmatrix}$$

```
> b:=Transpose(Av).Vector(Y);
```

$$b := \begin{bmatrix} 44 \\ 151 \\ 539 \end{bmatrix}$$

```
> Ai.b;
```

$$\begin{bmatrix} -\frac{9}{2} \\ \frac{16}{5} \\ 1 \end{bmatrix}$$

▼ 特異値分解(Singular Value Decomposition)

正規方程式での共分散行列,特異値分解の導出や標準偏差との関係はNumRecipeを参照せよ.

```
> restart;
X:=[1,2,3,4]:
Y:=[0,5,15,24]:
ff:=x->a[1]+a[2]*x+a[3]*x^2:
```

```
> with(LinearAlgebra):
Av:=Matrix(1..4,1..3):
> ff:=(x,i)->x^(i-1):
for i from 1 to 3 do
for j from 1 to 4 do
Av[j,i]:=ff(X[j],i);
end do;
end do;
Av;
```

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 4 & 16 \end{bmatrix}$$

```
> U,S,Vt:=evalf(SingularValues(Av,output=['U','S','Vt'))):
```

```
> DiagonalMatrix(S[1..3],4,3);
U.DiagonalMatrix(S[1..3],4,3).Vt:
```

```

[ 19.6213640200000015      0      0
  0      1.7120698739999999      0
  0      0      0.266252879300000022
  0      0      0 ]

```

```

> iS:=Vector(3):
  for i from 1 to 3 do
    iS[i]:=1/S[i];
  end do:
  DiS:=DiagonalMatrix(iS[1..3],3,4);

```

```

DiS := [ 0.05096485642      0      0      0
         0      0.5840883104      0      0
         0      0      3.755827928      0 ]

```

```

> Transpose(Vt).DiS.(Transpose(U).Vector(Y));

```

```

[ -4.50000000198176498
  3.20000000035008324
  1.00000000040565196 ]

```

▼ 2次元曲面へのフィット

先程の一般化をより発展させると、3次元(x_i, y_i, z_i)で提供されるデータへの、2次元平面でのフィットも可能となる。2次元の単純な曲面は、方程式を使って、

$$F(x, y) = a_1 + a_2x + a_3y + a_4xy + a_5x^2 + a_6y^2$$

となる。デザイン行列の行目の要素は、

$$[1, x_i, y_i, x_i y_i, x_i^2, y_i^2]$$

として、それぞれ求める。このデータの変換の様子をMapleスクリプトで詳しく示した。後は、通常の正規方程式を解くようにすれば、このデータを近似する曲面を定めるパラメータ a_1, a_2, \dots, a_6 が求まる。最小二乗法はパラメータ a_k について線形であればよい。

▼ xy平面への応用

実際のデータ解析での例。データの座標を x, y, z で用意して、Mapleの埋め込み関数のleastsquareでfitしている。

```

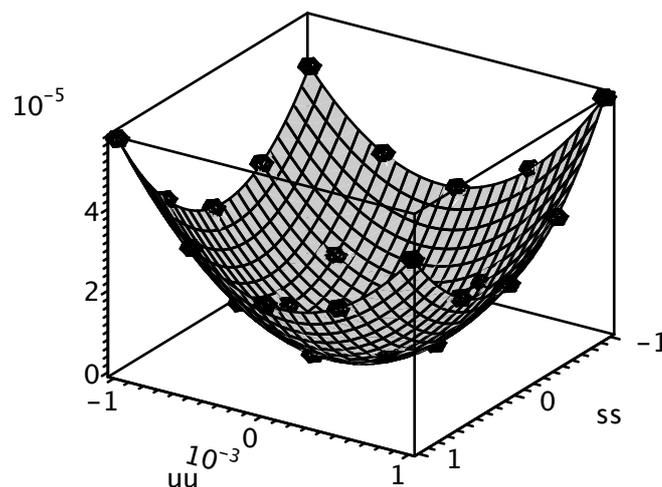
> with(plots):with(plottools):
  z:=[0.000046079702088, 0.000029479057275, 0.000025769637830, 0.000034951410953,
  0.000057024385455, 0.000029485453808, 0.000011519913869, 0.000006442404299, 0.000014252898382,
  0.000034951410953, 0.000025769637773, 0.000006442404242, 0.000000000000057, 0.000006442404242,
  0.000025769637773, 0.000034932221524, 0.000014246501905, 0.000006442404299, 0.000011519913926,
  0.000029479057332, 0.000056973214100, 0.000034932221467, 0.000025769637773, 0.000029485453808,
  0.000046079702031]:
> x:=[]:
  y:=[]:
  pl:=2:

```

```

for i from -p1 to p1 do
  for j from -p1 to p1 do
    x:=[op(x),i*0.0005];
    y:=[op(y),j*0.0005];
  end do;
end do;
> with(LinearAlgebra):
p2:=convert(Transpose(Matrix([x,y,z])),listlist):
> pp2:=pointplot3d(p2,symbol=circle,symbolsize=30,color=
black):
> with(stats):
data:=[x,y,z]:
fit1:=fit[leastsquare[[t,s,u],
u=a+b*t+c*s+d*t*s+e*t^2+f*s^2,
{a,b,c,d,e,f}]](data);
fit1 := u = -8.657142857 10-13 - 0.000006396456800 t + 0.000006396438400 s
- 5.459553587 t s + 25.76962838 t2 + 25.76962835 s2
> f1:=unapply(rhs(fit1),(s,t)):
> pf1:=plot3d(f1(ss,uu),ss=-0.001..0.001,uu=-0.001..0.001,
color=gray):
> display(pf1,pp2);

```



▼ 正規方程式による解法

デザイン行列へのデータ変換

```

> bb:=Vector(25):
A:=Matrix(25,6):
p1:=2:
for i from 1 to 25 do
  A[i,1]:=1;
  A[i,2]:=x[i];
  A[i,3]:=y[i];
  A[i,4]:=x[i]*y[i];
  A[i,5]:=x[i]^2;
  A[i,6]:=y[i]^2;
  bb[i]:=z[i];
end do:

```

正規方程式の解

```

> MatrixInverse(Transpose(A).A).(Transpose(A).bb);

```

```
-9.18525719622708541 10-13  
-0.00000639644675999994798  
0.00000639644220000032532  
-5.45955358336000173  
25.7696284050857187  
25.7696284050857543
```

▼ 課題

▼ 1 1次元の線形最小二乗法

次の4点のデータを $y = a_1 + a_2x + a_3x^2$ で近似せよ.

$X := [0, 1, 2, 3];$

$Y := [1, 3, 4, 10];$

▼ 2 2次元の最小二乗フィット

以下のデータを

$$f(x, y) = a_1 + a_2x + a_3y + a_4xy$$

で近似せよ

x, y, z

-1, -1, 2.00000

-1, 0, 0.50000

-1, 1, -1.00000

0, -1, 0.50000

0, 0, 1.00000

0, 1, 1.50000

1, -1, -1.00000

1, 0, 1.50000

1, 1, 4.00000