MATLAB(3)

MATLAB によるデータ処理

最小 2 乗法は、ノイズの含まれた実験データから直線、曲線のパラメータを推定する基本的かつ有力なデータ処理のひとつである。ここでは、最小 2 乗法の導出と MATLAB を用いた計算法について解説していく。

簡便のため直線近似の計算法を考えてみる．入力値 x_0，x_1，x_2，x_3 があった時の出力値が y_0，y_1，y_2，y_3 であると仮定しよう。これらのデータは直線の式 $y = ax + b$ で表現できるとすると、

$$y_0 = ax_0 + b$$
$$y_1 = ax_1 + b$$
$$y_2 = ax_2 + b$$
$$y_3 = ax_3 + b$$

が成り立つはずである。しかし一般に、データの点数 4 点に対し 2 点のパラメータ係数 a,b では、必ずしも解が見つかるとは限らない。そこで、評価関数として、

$J =$

を導入し、J が最小になるような a,b を求めるという問題として扱い解くのが最小二乗法である．

$J \to \min$ になる a,b を求めるには、

$$\frac{\partial J}{\partial a} = 2\sum_{i=0}^{3}\{y_i - (ax_i + b)\}x_i = 0, \frac{\partial J}{\partial b} = 2\sum_{i=0}^{3}\{y_i - (ax_i + b)\} = 0$$

となる必要がある．つまり上の式を展開していくと、

$$\sum x_i y_i - a\sum x_i^2 - b\sum x_i = 0, \sum y_i - a\sum x_i - b\sum 1 = 0$$

よって

$$\sum x_i y_i = a\sum x_i^2 + b\sum x_i, \sum y_i = a\sum x_i + b\sum 1$$

を行列表現に直すと

$$\begin{bmatrix}\sum x_i y_i \\ \sum y_i\end{bmatrix} = \begin{bmatrix}\sum x_i^2 & \sum x_i \\ \sum x_i & \sum 1\end{bmatrix}\begin{bmatrix}a \\ b\end{bmatrix}$$

この行列式を正規方程式とよぶ。これを変形しパラメータ a,b を求めると

$$\begin{bmatrix}a \\ b\end{bmatrix} = \begin{bmatrix}\sum x_i^2 & \sum x_i \\ \sum x_i & \sum 1\end{bmatrix}^{-1}\begin{bmatrix}\sum x_i y_i \\ \sum y_i\end{bmatrix}$$

となる．この式を計算すれば、自動的に $J \to \min$ を満たす a,b が求めることができる．

次に先ほどの式を行列ベースで考え直してみよう。行列ベースで書きなおすと

$$\begin{bmatrix}y_0 \\ y_1 \\ y_2 \\ y_3\end{bmatrix} = \begin{bmatrix}x_0 & 1 \\ x_1 & 1 \\ x_2 & 1 \\ x_3 & 1\end{bmatrix}\begin{bmatrix}a \\ b\end{bmatrix}$$

とあらわすことができる。次にベクトル、行列

$\mathbf{x} = \begin{bmatrix} a \\ b \end{bmatrix}$, **A =_, b =_**

を定義すると、連立方程式は $\mathbf{b} = \mathbf{Ax}$ **のようにあらわせる。両辺に** $\mathbf{A}^T$ **を乗算すると**

$$\mathbf{A^T b} = \mathbf{A^T A x}$$

$$\begin{bmatrix} x_0 & x_1 & x_2 & x_3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_0 & x_1 & x_2 & x_3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 & 1 \\ x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

となる。この式を良く見てみると実は、

$$\begin{bmatrix} \sum x_i y_i \\ \sum y_i \end{bmatrix} = \begin{bmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & \sum 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

となり、先ほどの式と同様になる．つまり、最小2乗法は、行列ベースで簡潔に表すと $\mathbf{A}^T\mathbf{b} = (\mathbf{A}^T\mathbf{A})\mathbf{x}$ **となり、両辺に** $(\mathbf{A}^T\mathbf{A})^{-1}$ **をかけることで、** $\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$ **とパラメータを 1 行で計算することができる．ここでは、パラメータが2つ推定する場合を扱ったが、パラメータが3つである2次式** $y = ax^2 + bx + c$ **でも同様に扱うことができる．つまり行列 x,A,b 行列を、**

$\mathbf{x} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$, **A =___, b =** $\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}$

と定義しなおすだけでよい．行列で表した場合、 $\mathbf{b} = \mathbf{Ax}$ **の表現に表せれば** $J = \sum \{\mathbf{b} - \mathbf{Ax}\}^2$ **で求められる J が最小となるパラメータ x を** $\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$ **で求められる．つまり A 行列 b 行列を変形すれば、直線、二次曲線のパラメータ近似以外にも例えば、分数** $y = a(1/x) + b$ **、ルート** $y = a\sqrt{x} + b$ **、べき乗** $y = ax^b$ → $\log y = b\log x + \log a$ **、自然対数** $y = a\log x + b$ **、指数** $y = ab^x$ → $\log y = \log b \cdot x + \log a$ **などなど様々な曲線にカーブフィットさせるパラメータを推定することも可能である．**

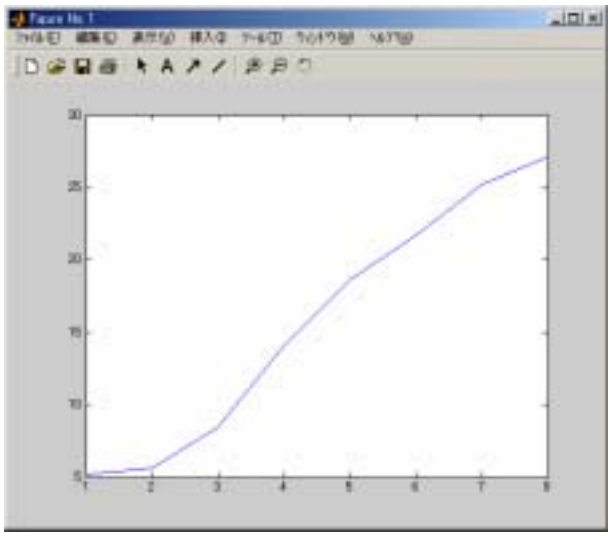
MATLAB による最小 2 乗法の計算例

MATLAB では，グラフ化，行列データ処理などが C 言語などと比較すると非常に容易にできる．ここでは，MATLAB を用い最小二乗法処理の実例を学ぶ．

時間	温度	時間	温度
1	5.2	5	18.6
2	5.6	6	21.7
3	8.5	7	25.2
4	14.1	8	27.1

のデータがあるとする．このデータを MATLAB でグラフ化するには、

```
》 x=1:8;y=[5.2 5.6 8.5 14.1 18.6 21.7 25.2 27.1];
》 plot(x,y)
```



とする。変数 y の中の数値の区切りは、スペースであるので、必ずスペースを入れるようにする。それでは、この x,y を元に y=ax+b の 1 次直線で近似するパラメータ a,b を求めてみよう．一応、確認のため公式通り計算してみよう．

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & \sum 1 \end{bmatrix}^{-1} \begin{bmatrix} \sum x_i y_i \\ \sum y_i \end{bmatrix}$$

であるのでこれを MATLAB で表現すると以下のようになる．

```
》 inv([x*x' sum(x);sum(x) 8])*[x*y'; sum(y)]
```

この場合も、スペースに注意すること、x*x'と sum(x)の間、それに sum(x)と 8 の間はスペースである。また、sum(x)と sum(x)の間はセミコロン、x*y'と sum(y)の間もセミコロンである。

$y = \sum_{i=1}^{n} x_i^2$ を計算するには？

まず、C 言語でのインプリメントを見てみよう．C 言語では、

```
y = 0.0;
for (i = 0;i< n;i++) {
        y += x[i] * x[i]; //  これは y = y+ x[i] * x[i];と同じ意味
        //C 言語の場合配列は通常 0 から始まり n-1 で終わる
}
```

となる．

MATLAB でも C 言語と同等に表現することが出来る．

```
y = 0;
for i=1:n
        y = y+x(i)* x(i);%MATLAB では，行列は 1 から始まり n で終わる
end
```

となる．しかし，MATLAB に備わっている sum 関数や演算子を活用すると以下のように簡潔に表現もできる．x が行アレイであるとすると

```
y=x*x';
```

と表現できる．また、x が行アレイ、列アレイ問わず'.*'演算子を活用すると

```
y = sum(x.*x);
```

表現できる．MATLAB ではこの様に書いた方が，処理速度の面からもプログラムの可読性の面からも優れている．全体を把握しやすくなるので開発効率が良い．

あるいは、

```
》 inv([sum(x.*x) sum(x);sum(x) 8])*[sum(x.*y); sum(y)]
ans =
      3.5167
     -0.0750
》
```

となるつまり、a=3.5167,b=-0.0750 として計算できた．では、行列ベースの計算を MATLAB でやらせてみよう．まず、A 行列を作成し $\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$ で計算する．

```
》  A=[x' ones(length(x),1)];inv(A'*A)*A'*y'
ans =
      3.5167
     -0.0750
》
```

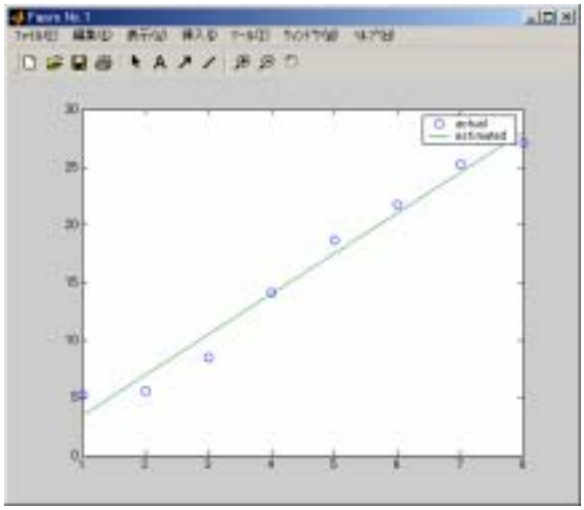
ここで length とは、ベクトル x の長さ（この場合データ点数）を求める関数であり、ones 関数で length(x)×1の行列を作っている．inv 関数は、逆行列を求める関数である．
このように MATLAB では、行列ベースで数式を書くと1行で計算できる．一般に正方行列でない行列の割り算は定義されていないが、実は，MATLAB では，\（バックスラッシュ）を使うと正方行列でない A に対しても計算することができる．つまり

```
》 A\y'
ans =
      3.5167
     -0.0750
》
```

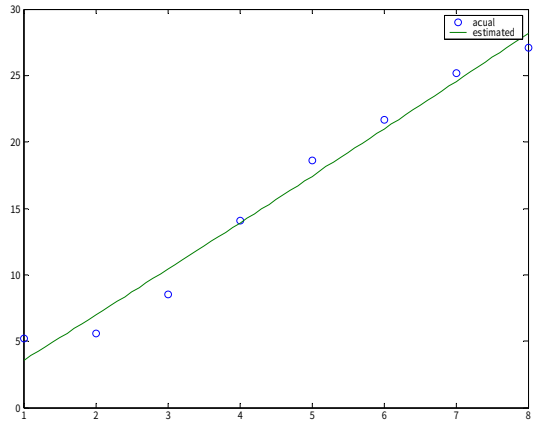
としてもよい．MATLAB では、この演算の方が逆行列を計算せず消去法により直接最小二乗解を求めるため精度が良いとされる．以上よりどの方法を用いても同じ様に求められるが、最小二乗法による1次直線は、y=3.51x-0.07
と計算できた．これをグラフ化してみよう．グラフ化するには、

```
》 plot(x,y,'o',x,3.51*x-0.07);legend('actual','estimated')
```

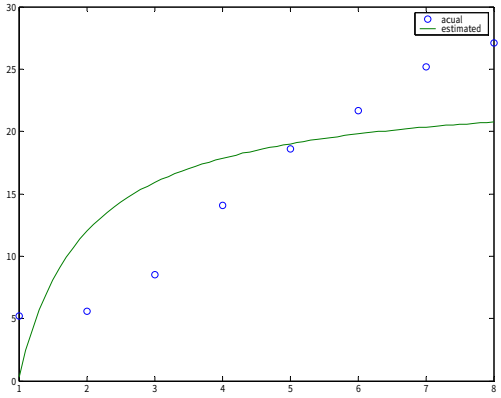
とする．plot 関数では、x 座標、y 座標、線種が指定できる．1つの直線、点線だけ出なく複数の直線、点を書くことが可能である．詳しくは、help plot で確認すると良い．
legend コマンドはその線、点の名前を書くのに用いる．legend に書かれた線種の名前は、マウスによりドラックすることで場所を好きな場所に移動することができる．



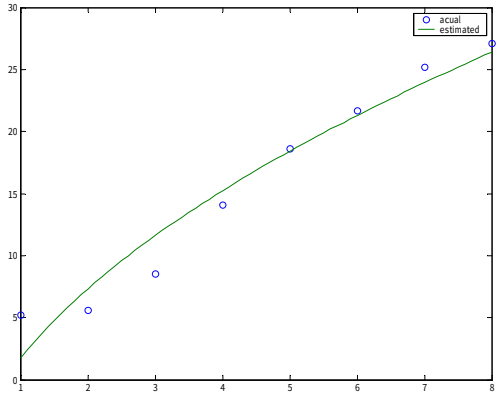
また，MATLAB は行列ベースなので 2 次曲線近似のパラメータ推定でも行列の一部とプロット関数の部分を少し手直しするだけで，容易に 2 次曲線 y=ax^2+bx+c 近似も可能である．
以下に様々なカーブフィット例をするときの MATLAB スクリプトの例を示す。

2 次曲線による近似	$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_0^2 & x_0 & 1 \\ x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}$
計算例	**A=[x(:).^2 x(:) ones(length(x),1)];b=y(:);xxx=A\b**
表示例	**xx=1:0.1:8;plot(x,y,'o',xx,xxx(1)*xx.^2+xxx(2)*xx+xxx(3));** **legend('acual','estimated');** 

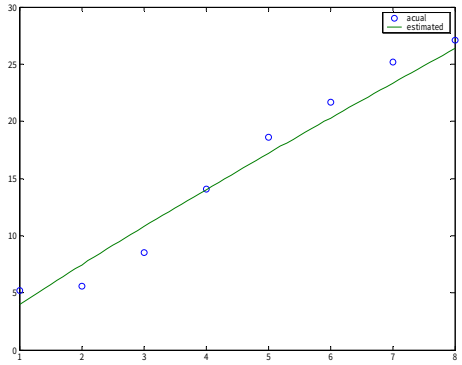
分数による近似	$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{x_0} & 1 \\ \frac{1}{x_1} & 1 \\ \frac{1}{x_2} & 1 \\ \frac{1}{x_3} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$
計算例	**A=[1./x(:) ones(length(x),1)];b=y(:);xxx=A\b**
表示例	**xx=1:0.1:8;plot(x,y,'o',xx,xxx(1)./xx+xxx(2));legend('acual','estimated');**

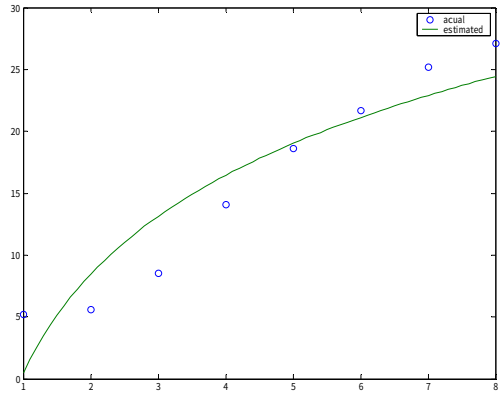
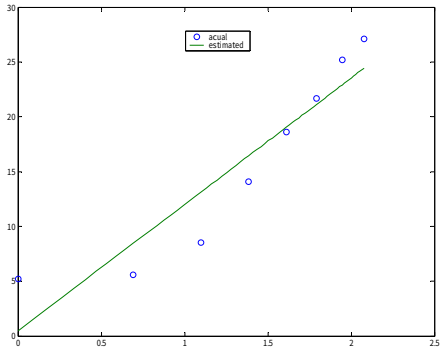


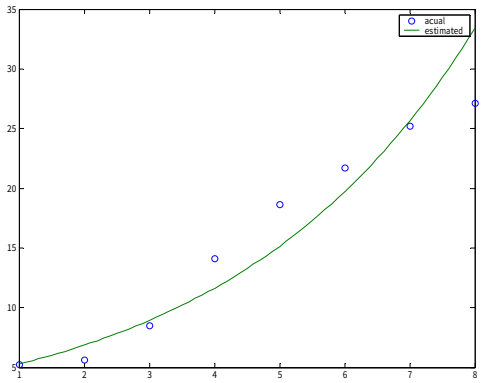
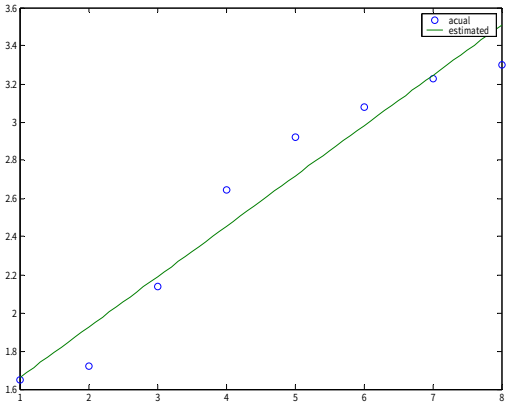
平方根による近似	$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \sqrt{x_0} & 1 \\ \sqrt{x_1} & 1 \\ \sqrt{x_2} & 1 \\ \sqrt{x_3} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$
計算例	**A=[sqrt(x(:)) ones(length(x),1)];b=y(:);xxx=A\b**
表示例	**xx=1:0.1:8;plot(x,y,'o',xx,xxx(1)*sqrt(xx)+xxx(2));legend('acual','estimated');**



べき乗による近似 $y = ax^b \rightarrow b\log x + a$	$\begin{bmatrix} \log y_0 \\ \log y_1 \\ \log y_2 \\ \log y_3 \end{bmatrix} = \begin{bmatrix} \log x_0 & 1 \\ \log x_1 & 1 \\ \log x_2 & 1 \\ \log x_3 & 1 \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix}$ **注：対数にすると a,b の位置が変わるので注意**
計算例	**A=[log(x(:)) ones(length(x),1)];b=log(y(:));xxx=A\b**
表示例	**xx=1:0.1:8;plot(x,y,'o',xx,exp(xxx(1)*log(xx)+xxx(2)));** **legend('acual','estimated');**



自然対数による近似 $y = a\log x + b$	$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \log x_0 & 1 \\ \log x_1 & 1 \\ \log x_2 & 1 \\ \log x_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$
計算例	**A=[log(x(:)) ones(length(x),1)];b=y(:);xxx=A\b**
表示例	**xx=1:0.1:8;plot(x,y,'o',xx,xxx(1)*log(xx)+xxx(2));** **legend('acual','estimated');**  **直線近似したい場合には、軸を変えて** **xx=1:0.1:8;plot(log(x),y,'o',log(xx),xxx(1)*log(xx)+xxx(2));** **legend('acual','estimated');** 

指数関数による近似 $y = ab^x \rightarrow$ $\log y = x\log b + \log a$	$\begin{bmatrix} \log y_0 \\ \log y_1 \\ \log y_2 \\ \log y_3 \end{bmatrix} = \begin{bmatrix} x_0 & 1 \\ x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \end{bmatrix} \begin{bmatrix} \log b \\ \log a \end{bmatrix}$
計算例	**A=[x(:) ones(length(x),1)];b=log(y(:));xxx=A\b**
表示例	**xx=1:0.1:8;plot(x,y,'o',xx,exp(xxx(1)*xx+xxx(2))); legend('acual','estimated');**  **同様に直線近似したい場合には、軸を変更する。** **xx=1:0.1:8;plot(x,log(y),'o',xx,xxx(1)*xx+xxx(2));** **legend('acual','estimated');** 

最小 2 乗法で求めたパラメータの誤差の計算とビジュアライズによる確認

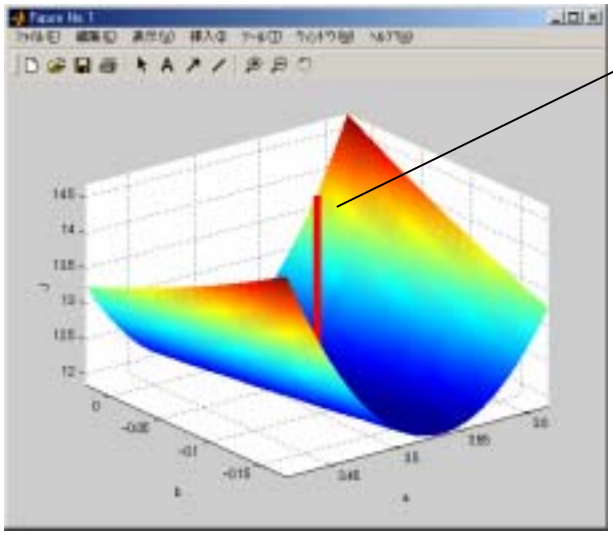
最小になる求められた a,b を基に評価関数 $J = \sum_{i=0}^{3} \{y_i - (ax_i + b)\}^2$ **を求めてみよう．MATLAB では，以下のようにすることで計算できる．**

```
》 tmp=A\y';a=tmp(1);b=tmp(2);J=sum((y-(a*x+b)).^2)
J =     11.8483
```

MATLAB では，スクリプトを組むことで容易にその確認もできる．ここでは，2 変数 a,b の値を for 文によりスイープし、それにより変化した評価関数 J のビジュアライゼーションした例を示す。

```
x=1:8;y=[5.2 5.6 8.5 14.1 18.6 21.7 25.2 27.1];
A=[x' ones(length(x),1)];
tmp=A\y';a=tmp(1);b=tmp(2);J=sum((y-(a*x+b)).^2);
% first determine sweep range of a and b.
sweepa=(-0.1:0.01:0.1)+a;sweepb=(-0.1:0.01:0.1)+b;
```

```
JJJ=[];
for bb=sweepb
   JJ=[];
   for aa=sweepa
     JJ=[JJ sum((y-(aa*x+bb)).^2)];
   end
   JJJ=[JJJ;JJ];
end
surf(sweepa,sweepb,JJJ);
hold on
plot3([tmp(1) tmp(1)],[tmp(2) tmp(2)],[min(JJJ(:)) max(JJJ(:))],'r','linewidth',5);
hold off
shading interp
xlabel('a');ylabel('b');zlabel('J');axis tight
```



変数 JJJ,JJ の使い方に注意すること。C 言語などでこのようなデータを計算する場合、まずデータを格納する配列を確保する必要があるが MATLAB では、動的に行列を確保できるためその必要はない。

MATLAB によるデータ処理課題

ある地域の 1 年の温度変化データを基に最小 2 乗法によりそのデータは，どの予測モデル（分数，ルート，べき乗，自然対数，指数，1 次近似，2 次近似など）が一番フィットするか比較，検討せよ．