Matlab Programming Help

Online Help:

4. http://terpconnect.umd.edu/~nsw/ench250/for-mat.htm (Comparison between Matlab and Fortran F77)

Programming Basic

1. **Input and output statement: output is written first.**

   *Example 1:*

   \[ X = Y + 1 \]

   Define \( X \) as \( Y+1 \).

   \( X \) can be a scalar, vector, matrix. To do so, you have to define it first. For scalar case, you don't need to do so.

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y=1</td>
<td>Variable ( Y ) is defined as 1</td>
</tr>
<tr>
<td>Y=1+Y</td>
<td>Variable ( Y ) is defined as 1 + 1, so that ( Y ) becomes 2</td>
</tr>
<tr>
<td>X=Y+1</td>
<td>Variable ( X ) is defined as 2 + 1</td>
</tr>
<tr>
<td>Y = zeros(1,2)</td>
<td>( Y ) becomes a 1x2 null vector. ( Y = (0, 0) )</td>
</tr>
<tr>
<td>Y(1,1) =1</td>
<td>( Y = (1,2) )</td>
</tr>
<tr>
<td>Y(1,2) =2</td>
<td></td>
</tr>
</tbody>
</table>

2. **Do Loop**

   Suppose that you want to add the sequence of numbers from 1 to 10. Let's program this.

   *Ex2. \( X = 1 + 2 + 3 + 4 + 5 + \ldots + 10 \)*

   *Sol 1: Write as*

   \[ X =1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 \]

   *Sol 2: Write as*

   \[ X = 1  \\
   X = X + 2  \\
   X = X + 3  \\
   \ldots  \\
   X = X + 10 \]
Sol 3: Use "For" statement

```plaintext
X = 0;
For i = 1:10;
X = X + i;
End;
```

Assign 0 to X. Need ";" to continue program it.

Start Do loop. First assign i to be 1, and increase it by 1 up to 10.
X becomes X + i,
Repeat this until i = 10

Let X = [ 1 3 4 2 2 2 4 1 23 5]; That is, X is a 10x1 vector. Calculate its mean by using For statement

```plaintext
Z = 0;
For i = 1:10;
Z = Z + X(i);
End;
```

Assign 0 to Z.
Start Do loop. First assign i to be 1, and increase it by 1 up to 10.
Z becomes Z + X(i),
Repeat this until i = 10

Exercise: For Statement

1. Add 1 through 100
2. Multiply 1 through 20
3. $2 \times 4 \times 6 \times 8 \times \ldots \times 20$
4. $X(1)Y(1) + X(2)Y(2) + \ldots + X(n)Y(n)$

3. IF Statement

Format:

IF condition statement End

Example: $X = [1 -2 3 -4]$

We want to change X to index such that $Y = 0$ if $X >0$, $Y=1$ o.w.

$Y = X;
For i = 1:4;
if X(i) > 0; Y(i) = 0; end;
if X(i) < 0; Y(i) = 1; end;
end;

Exercise: IF Statement

1. $X = [1 3 4 8]$. Find the maximum of X.
2. Find the minimum of X
3. Sort X.
4. Data (Matrix & Vector) Modification

Ex: \( A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \) implies \( A = \begin{bmatrix} 1 \\ 3 \\ 2 \\ 4 \end{bmatrix} \)

Type the following commands.
1. \( A' \)
2. \( \text{sum}(A) \)
3. \( \text{sum}(A') \)
4. \( \text{sum}(A')' \)
5. \( \text{diag}(A) \)
6. \( \text{sum}(	ext{diag}(A)) \)
7. \( \text{inv}(A) \)
8. \( A(1,2) \)
9. \( A(1,1) \)
10. \( A(2,1) \)
11. \( A(:,1) \)
12. \( A(1,:) \)
13. \( A(:) \)
14. \( A(:,\text{end}) \)

Expression

\( \ast \) element by element product
\( \div \) element by element division
\( \backslash \) inverse
\( \wedge \) element by element power

Type \( A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}; \ B = \begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix}; \)

1. a
2. A
3. b
4. B
5. A.*B
6. A./B
7. A\B
8. Inv(A).*B
9. A'*A
10. B'*B
11. Inv(A).*A
12. Inv(A).*A

Ex \( b = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \). You want to calculate \( A - b = \begin{bmatrix} 1-2 \\ 2-3 \\ 3-2 \\ 4-3 \end{bmatrix} \)

Important Functions

Type \( A = \begin{bmatrix} -3 & 4; 1 & 3; 2 \end{bmatrix} \)

1. a
2. A
3. mean(A)
4. sum(A)
5. sort(A)
6. [B,id] = sort(A)
7. max(A)
8. min(A)
9. std(A)
10. var(A)
11. cov(A)
12. abs(A)

5. Function Statement

Format

Function output = functionname(inputs)

Example: Average

Function y = mymean(x)
  t = length(x);
  y = 0;
  for i = 1:t;
    y = y + x(i);
  end;

Then in the main program, you can recall `mymean'.

z = mymean(x);

In Fortran, this function statement is called as `subroutine' program.
In Gauss, it is called as `proc' program.
Matlab library contains many function statements.
**Assignment 1: Download X and Y variables from the class homepage.**

A. Sort X from smallest to largest
B. Sort Y from largest to smallest
C. Calculate mean and variance of X and Y.
D. Calculate correlation between X and Y
E. Make functions (mymean, myvar, mycorr) and use them to calculate C, D and E.
F. Program OLS function.

Input = x and y. both them are Tx1 vectors.

Regression: \( y = bx + u \)

output:

\[ b = \text{inv}(x'x) \times x'y \]

\( R^2 \),

ordinary t-value.

Function \([b, r2, tb] = \text{myols}(y, x)\)
Lecture 2: OLS & GLS

Cross section or Time series data

Model $y = a + X*b + u$

Where $X$ is a matrix (nxk, k is number of regressors), $a$ is scalar, $b$ is a vector.

Define a vector such that

$n = \text{length}(y)$;
$a = \text{ones}(n,1)$;

Next, define a matrix such that

$Z = [a \ X]$;
$[n,k] = \text{size}(Z)$;

**OLS estimator:**

$bhat = \text{inv}(Z'*Z)*Z'*y$;

1. Regression residuals: $uhat = y - Z*bhat$;
   a. t-ratio needs variance of $bhat$:
      i. IID case:
         sigma = $uhat.*uhat$;
         sigma = sum(sigma)/(n-k);
         sigma = diag(sigma);
      ii. IDIN case:
         uuhat = $Z.*\text{repmat}(uhat,1,2)$;
         sigma = uuhat'*uuhat;
         sigma = sum(sigma)/(n-k);
         sigma = inv(Z'*Z)*sigma*inv(Z'*Z)*n;
   b. R-squares $Rbar$-squares:

Example file: ex2.m

$n = 100$;
y = randn(n,1);
x = randn(n,1);

$z = [\text{ones}(n,1) \ x]$;
b = inv(z'*z)*z'*y;
u = y - z*b;

$\text{sig1} = u'*u/(n-2)$;
$\text{sig1} = \text{sig1}*\text{inv}(z'*z)$;

$h = z.*\text{repmat}(u,1,2)$;
$\text{sig2} = h'*h/(n-2)$;
$\text{sig2} = \text{inv}(z'*z)*\text{sig2}*\text{inv}(z'*z)*n$;

$\text{tra1} = b./\sqrt{\text{diag(sig1)}}$;
$\text{tra2} = b./\sqrt{\text{diag(sig2)}}$;

[\text{tra1} \ \text{tra2}]$
GLS Estimator: AR(1) coefficient case

```matlab
n = 100;
y = randn(n,1);
x = randn(n,1);

z = [ones(n,1) x];
b = inv(z'*z)*z'*y;
u = y - z*b;

% estimation of AR(1) coefficient
uy = u(2:n); ux = u(1:n-1);
rho = inv(ux'*ux)*ux'*uy; % or equivalently rho = sum(ux.*uy)/sum(ux.*ux)
e = uy-ux*rho;
ve = var(e);

% constructing co-variance and variance matrix
omega = eye(n).*ve./(1-rho^2);
for i = 1:n;
    for j = i+1:n;
        omega(i,j) = rho^(j-i);
        omega(j,i) = omega(i,j);
    end;
end;

% Cholesky Decomposition
P = chol(omega);

% Pre-multiplying P matrix
ys = P*y;
zs = P*z;
c = inv(zs'*zs)*zs'*ys;

% variance matrix for c
vc = inv(zs'*zs);

% t-values
tra = c./sqrt(diag(vc));
tra
```
**Pooled OLS and LSDV**

clear;
t=2;
n=10;

y = randn(t,n);
x = randn(t,n);

% LSDV
a = ones(t,1);
a = kron(eye(n),a);

vx = x(:);
z = [a vx];
b = inv(z'*z)*z'*y(:);
b

% POLS
a = ones(t,1);
a = repmat(a,1,n);
a = a(:);
z = [a vx];
b = inv(z'*z)*z'*y(:);
b

**Assignment 3:**
A. Suppose that you want to program the following regressions
\[ y(it) = \alpha_i + cX(it) + u(it) \]
1. Input must be y and x where X is a nxk matrix
2. Output must include point estimates, their standard errors (ordinary one, panel robust one), r-bar squares etc.
B. Suppose that you want to program the following regressions
\[ Y(it) = \alpha_i + bz(i) + cX(it) + u(it) \]
3. Make function for LSDV and POLS
4. Output must include point estimates, their standard errors (ordinary one, panel robust one), r-bar squares etc.