# Working with matrices in SAS

Søren Højsgaard
(slightly modified by Russ Lenth)

September 25, 2007

## Working with matrices

Submit the following statements. First, start PROC IML by submitting

```
proc iml;
```

Define some simple arrays and print them

```
a={1 2,
   3 4};
b={2 2,
   1 3};
print a; print b;
```

```
A
1          2
3          4

B
2          2
1          3
```

## Matrix manipulations

Multiplication and addition

```
ab  = a*b;
ba  = b*a;
apb = a+b;
print ab ba apb;
```

```
AB                BA                APB
 4        8        8        12        3        4
10       18       10        14        4        7
```

Transposing

```
ata = t(a)*a;
aat = a*t(a);
print ata aat;
```

```
      ATA                 AAT
       10        14         5        11
       14        20        11        25
```

Inverse

```
ai  = inv(a);
aai = a*ai;
print ai aai;
```

```
       AI                 AAI
       -2         1         1  1.11E-16
      1.5      -0.5 -2.22E-16         1
```

Special matrices

```
i = I(4);   * Identity matrix;
j = J(4,1);   * Column vector of 1's;
print i, j;
```

Additional operations

```
c={1 1 2,
   2 1 3,
   4 3 1};

c1   = c[3,1];    * Element in 3rd row 1st column;
c2   = c[1,];   * The first row;
c3   = c[,2];   * The second column;
dubc = c||c;    * Put two copies of c together;
cc3  = c||c3;   * Put c3 after c;
c32  = hdir(c3,c3); * Square each element in c3;
print c1, c2, c3, dubc, cc3 c32;
```

```
    C1
        4
             C2
        1         1         2
    C3
        1
        1
        3
                            DUBC
        1         1         2         1         1         2
        2         1         3         2         1         3
        4         3         1         4         3         1
```

```
     CC3                                    C32
       1         1         2         1         1
       2         1         3         1         1
       4         3         1         3         9
```

Stop PROC IML by submitting

```
quit;
```

## Linear Normal Models

Fit a simple model using matrix algebra

```
proc iml;

/* ***********************************************
Input the design matrix X and the data vector y
as matrix of numbers. Which model is this?
********************************************** */

x={1 1 1,
   1 2 4,
   1 3 9,
   1 4 16,
   1 5 25};

y={1,5,9,23,36};
```

See the results in the output window

```
print x; print y;
```

```
             X
       1         1         1
       1         2         4
       1         3         9
       1         4        16
       1         5        25

   Y
       1
       5
       9
      23
      36
```

Compute the least-squares estimate of $b$ using the traditional formula. Note that `t()` means "transpose"

```
beta=inv(t(x) * x) * t(x) * y;
print beta;
```

```
   BETA
      2.4
     -3.2
        2
```

The predicted values are simply the X matrix multiplied by the parameter estimates, and the residuals are the difference between actual and predicted y.

```
yhat  = x*beta;
resid = y-yhat;
print yhat; print resid;
```

```
   YHAT
      1.2
        4
     10.8
     21.6
     36.4

  RESID
     -0.2
        1
     -1.8
      1.4
     -0.4
```

To calculate the estimate of the variance of the responses, calculate the sum of squared errors (SSE) using the ssq-function, its degrees of freedom (DFE), and the mean squared error (MSE). Note that in computing the degrees, you use the function NCOL to return the number of columns of X.

```
sse = ssq(resid);
dfe = nrow(x)-ncol(x);
mse = sse/dfe;
print sse dfe mse;
```

```
      SSE       DFE       MSE
      6.4         2       3.2
```

We also want the standard errors of the parameter estimates

```
xtxi = inv(t(x)*x);
stdb = sqrt(vecdiag(xtxi)*mse);
print xtxi; print stdb;
```

```
             XTXI
      4.6      -3.3       0.5
     -3.3 2.6714286 -0.428571
      0.5 -0.428571 0.0714286
   STDB
3.8366652
 2.923794
0.4780914
```

Perform a $t$-test for the parameters being zero

```
tstat = beta/stdb;
prob  = 1-probf(tstat#tstat,1,dfe);
print prob;
```

```
   PROB
0.5954801
 0.387969
0.0526691
```

Print it nicely on the screen

```
print,"Parameter Estimates",,
        beta stdb tstat prob;
print,y yhat resid;
quit;
```

```
Parameter Estimates
     BETA      STDB     TSTAT      PROB
      2.4 3.8366652 0.6255432 0.5954801
     -3.2  2.923794 -1.094468  0.387969
        2 0.4780914 4.1833001 0.0526691
        Y      YHAT     RESID
        1       1.2      -0.2
        5         4         1
        9      10.8      -1.8
       23      21.6       1.4
       36      36.4      -0.4
```

## Getting SAS datasets in and out of IML

Here's a sample dataset:

```
proc print data=sashelp.class; run;
```

| Obs | Name | Sex | Age | Height | Weight |
|---|---|---|---|---|---|
| 1 | Alfred | M | 14 | 69.0 | 112.5 |
| 2 | Alice | F | 13 | 56.5 | 84.0 |
| 3 | Barbara | F | 13 | 65.3 | 98.0 |
| 4 | Carol | F | 14 | 62.8 | 102.5 |
| 5 | Henry | M | 14 | 63.5 | 102.5 |
| 6 | James | M | 12 | 57.3 | 83.0 |
| 7 | Jane | F | 12 | 59.8 | 84.5 |
| 8 | Janet | F | 15 | 62.5 | 112.5 |
| 9 | Jeffrey | M | 13 | 62.5 | 84.0 |
| 10 | John | M | 12 | 59.0 | 99.5 |
| 11 | Joyce | F | 11 | 51.3 | 50.5 |
| 12 | Judy | F | 14 | 64.3 | 90.0 |
| 13 | Louise | F | 12 | 56.3 | 77.0 |
| 14 | Mary | F | 15 | 66.5 | 112.0 |
| 15 | Philip | M | 16 | 72.0 | 150.0 |
| 16 | Robert | M | 12 | 64.8 | 128.0 |
| 17 | Ronald | M | 15 | 67.0 | 133.0 |
| 18 | Thomas | M | 11 | 57.5 | 85.0 |
| 19 | William | M | 15 | 66.5 | 112.0 |

Let's read it in IML:

```
proc iml;
  use sashelp.class;
  read all var _NUM_ into x;
  print x;
```

```
              X
       14       69     112.5
       13     56.5        84
       13     65.3        98
       14     62.8     102.5
       14     63.5     102.5
       12     57.3        83
       12     59.8      84.5
       15     62.5     112.5
       13     62.5        84
       12       59      99.5
       11     51.3      50.5
       14     64.3        90
       12     56.3        77
       15     66.5       112
       16       72       150
       12     64.8       128
       15       67       133
       11     57.5        85
       15     66.5       112
```

. . . and convert to metric:

```
  x[,2] = x[,2] * 2.54;
  x[,3] = x[,3] / 2.2;
```

Now export it to a SAS dataset and quit:

```
  create newdata from x[colname={'Age_yrs', 'Ht_cm', 'Wt_kg'}];
    append from x;
quit;
```

Here's the resulting SAS dataset:

```
proc print data=newdata; run;
```

```
Obs    Age_yrs     Ht_cm      Wt_kg
  1       14      175.260    51.1364
  2       13      143.510    38.1818
  3       13      165.862    44.5455
  4       14      159.512    46.5909
  5       14      161.290    46.5909
  6       12      145.542    37.7273
  7       12      151.892    38.4091
  8       15      158.750    51.1364
  9       13      158.750    38.1818
 10       12      149.860    45.2273
 11       11      130.302    22.9545
 12       14      163.322    40.9091
 13       12      143.002    35.0000
```

```
14        15       168.910    50.9091
15        16       182.880    68.1818
16        12       164.592    58.1818
17        15       170.180    60.4545
18        11       146.050    38.6364
19        15       168.910    50.9091
```