THE UNIVERSITY OF MINNESOTA

Statistics 5401 November 14, 2005

Example of Principal Component and ULS Factor Analysis Estimation

This handout contains MacAnova output illustrating the use of the *principal component* and the *unweighted least squares* (ULS) methods of factor extraction. Two ways of doing ULS extraction are illustrated, the iterated principal factor method and direct minimization of the unweighted least squares criterion.

The data are the chicken bone data (matrix bonedata in file cbbones.txt).

```
Cmd> setoptions(format:"10.6f") # all output with 6 decimals

Cmd> bones <- read("","bonedata") # read from file cbbones.txt
bonedata      276      6 format labels
) Bone measurements on n = 276 outbred female chickens, all in mm.
) Col. 1:  skull length
) Col. 2:  skull breadth
) Col. 3:  femur length (leg bone)
) Col. 4:  tibia length (leg bone)
) Col. 5:  humerus length (wing bone)
) Col. 6:  ulna length (wing bone)
Read from file "TP1:Stat5401:Data:cbbones.txt"

Cmd> r <- cor(bones);r # Correlation matrix
         SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
SklLngth 1.000000   0.583009   0.569111   0.602259   0.621119   0.602334
SklBrdth 0.583009   1.000000   0.515310   0.547599   0.583552   0.524505
FemLngth 0.569111   0.515310   1.000000   0.926105   0.877221   0.877453
TibLngth 0.602259   0.547599   0.926105   1.000000   0.873628   0.893610
HumLngth 0.621119   0.583552   0.877221   0.873628   1.000000   0.936879
UlnLngth 0.602334   0.524505   0.877453   0.893610   0.936879   1.000000
```

Principal component method

```
Cmd> eigs <- eigen(r); eigs # shows one really dominant eigenvalues
component: values
(1)   4.567387   0.719078   0.411912   0.169279   0.079369   0.052976
component: vectors
              (1)        (2)        (3)        (4)        (5)        (6)
SklLngth 0.348285  -0.525610  -0.774171   0.046177   0.028249  -0.012691
SklBrdth 0.324583  -0.704358   0.626427   0.027304  -0.016657  -0.071353
FemLngth 0.434213   0.288068   0.057264   0.504631   0.609436  -0.314852
TibLngth 0.440343   0.231195   0.040361   0.469623  -0.610364   0.397401
HumLngth 0.443959   0.166382   0.057636  -0.542845   0.355212   0.592485
UlnLngth 0.440228   0.251983  -0.004365  -0.476700  -0.358877  -0.621811

Cmd> # Find the estimated loading matrix

Cmd> l6 <- sqrt(eigs$values)' * eigs$vectors # multiply columns

Cmd> # could also use l6 <- eigs$vectors %*% dmat(sqrt(eigs$values))
```

```
Cmd> l6 # loading matrix for 6 factors
              (1)        (2)        (3)        (4)        (5)        (6)
SklLngth 0.744335  -0.445709  -0.496866   0.018999   0.007958  -0.002921
SklBrdth 0.693681  -0.597285   0.402043   0.011234  -0.004693  -0.016423
FemLngth 0.927977   0.244277   0.036752   0.207623   0.171693  -0.072468
TibLngth 0.941077   0.196050   0.025904   0.193219  -0.171954   0.091468
HumLngth 0.948805   0.141089   0.036991  -0.223346   0.100072   0.136369
UlnLngth 0.940830   0.213678  -0.002801  -0.196131  -0.101104  -0.143119

Cmd> l6 %*% l6' # using all cols, l6 l6' reproduces r
         SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
SklLngth 1.000000   0.583009   0.569111   0.602259   0.621119   0.602334
SklBrdth 0.583009   1.000000   0.515310   0.547599   0.583552   0.524505
FemLngth 0.569111   0.515310   1.000000   0.926105   0.877221   0.877453
TibLngth 0.602259   0.547599   0.926105   1.000000   0.873628   0.893610
HumLngth 0.621119   0.583552   0.877221   0.873628   1.000000   0.936879
UlnLngth 0.602334   0.524505   0.877453   0.893610   0.936879   1.000000

Cmd> l1 <- eigs$vectors[,1]*sqrt(eigs$values[1])

Cmd> l1 # single factor loading matrix, same as column 1 of l6 above
               (1)
SklLngth   0.744335     Principal component single factor loadings
SklBrdth   0.693681
FemLngth   0.927977
TibLngth   0.941077
HumLngth   0.948805
UlnLngth   0.940830

Cmd> psihat1 <- diag(r - l1 %*% l1'); psihat1
(1)   0.445965   0.518806   0.138859   0.114375   0.099769   0.114839

Cmd> sigmahat1 <- l1 %*% l1' + dmat(psihat1)

Cmd> sigmahat1 # fitted matrix for m = 1
         SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
SklLngth 1.000000   0.516331   0.690726   0.700476   0.706229   0.700293
SklBrdth 0.516331   1.000000   0.643720   0.652807   0.658169   0.652636
FemLngth 0.690726   0.643720   1.000000   0.873297   0.880469   0.873068
TibLngth 0.700476   0.652807   0.873297   1.000000   0.892898   0.885393
HumLngth 0.706229   0.658169   0.880469   0.892898   1.000000   0.892665
UlnLngth 0.700293   0.652636   0.873068   0.885393   0.892665   1.000000

Cmd> dev_1 <- r - sigmahat1 # residuals from 1 factor fit

Cmd> vector(sum(vector(dev_1)^2),max(abs(vector(dev_1))))
(1)   0.200956   0.128410   SS residuals and max(abs(resids)

Cmd> l2 <- eigs$vectors[,run(2)]*sqrt(eigs$values[run(2)])'

Cmd> l2 # estimated loadings with m = 2 (1st 2 columns of l above)
               (1)        (2)
SklLngth   0.744335  -0.445709 2 factor loadings
SklBrdth   0.693681  -0.597285 Note first column is same
FemLngth   0.927977   0.244277
TibLngth   0.941077   0.196050
HumLngth   0.948805   0.141089
UlnLngth   0.940830   0.213678

Cmd> psihat2 <- diag(r - (l2 %*% l2'))
```

```
Cmd> sigmahat2 <- l2 %*% l2' + dmat(psihat2)

Cmd> dev_2 <- r - sigmahat2 # residuals from 2 factor fit

Cmd> vector(sum(vector(dev_2)^2),max(abs(vector(dev_2))))
(1)   0.096808   0.199537   SS residuals and max(abs(resids)

Cmd> print(psihat2,dev_1, dev_2)
psihat2:
(1)   0.247308   0.162057   0.079188   0.075939   0.079862   0.069180
dev_1: Deviations from single factor fit
          SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
SklLngth  0.000000   0.066678  -0.121615  -0.098217  -0.085110  -0.097959
SklBrdth  0.066678   0.000000  -0.128410  -0.105208  -0.074617  -0.128131
FemLngth -0.121615  -0.128410   0.000000   0.052808  -0.003248   0.004385
TibLngth -0.098217  -0.105208   0.052808   0.000000  -0.019270   0.008217
HumLngth -0.085110  -0.074617  -0.003248  -0.019270   0.000000   0.044214
UlnLngth -0.097959  -0.128131   0.004385   0.008217   0.044214   0.000000
dev_2: Deviations from 2 factor fit
          SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
SklLngth  0.000000  -0.199537  -0.012738  -0.010835  -0.022225  -0.002721
SklBrdth -0.199537   0.000000   0.017493   0.011890   0.009654  -0.000505
FemLngth -0.012738   0.017493   0.000000   0.004917  -0.037713  -0.047812
TibLngth -0.010835   0.011890   0.004917   0.000000  -0.046931  -0.033674
HumLngth -0.022225   0.009654  -0.037713  -0.046931   0.000000   0.014067
UlnLngth -0.002721  -0.000505  -0.047812  -0.033674   0.014067   0.000000
```

The underlined value is the worst fit element.

Iterated principal factor method

stepuls() carries out one step of the iterated principal factor method:

```
Cmd> result <- stepuls(s,psi,m)
```

or

```
Cmd> result <- stepuls(s,psi,m,print:T)
```

where **s** is a p by p variance or correlation matrix, **psi** is a vector whose value is the current value of the diagonal elements of $\mathbf{\Psi}$, **m** is the number of factors. With **print:T**, the new value of **psi** and a goodness of fit quantity will be printed. Argument **psi** can also be a structure whose first component is a vector containing the diagonal elements of $\mathbf{\Psi}$.

The returned value, **result**, is a structure with three components, **psi** (the updated value of $\mathbf{\Psi}$), **loadings** (the updated value of **L**), and **crit**, the goodness of fit criterion $\mathrm{tr}(\hat{\mathbf{\Sigma}} - \mathbf{S})^2 = \sum_{m+1 \le i \le p} \hat{\delta}_i^2$, where $\hat{\delta}_1 \ge \ldots \ge \hat{\delta}_p$ are the eigenvalues of $\mathbf{S} - \hat{\mathbf{\Psi}}^{(i)}$.

crit actually measures the goodness of the fit provided by the argument **psi** *before* updating **L**.

Because **stepuls()** accepts a structure as second argument, you can use **result** as the argument **psi** in the next iteration. In fact, a generic step of the iteration is

```
Cmd> psi <- stepuls(s,psi,m [,print:T])
```

Besides returning a structure as values, **stepuls()** creates variables **PSI**, **LOADINGS** and **CRITERION** as "side effects". These are identical to components **psi**, **loadings** and **crit** of the returned value.

Note: The iterated principal factor method illustrated below is *not* one that can be recommended for actually doing ULS extraction. If it converges at all, it may do so slowly, each iteration being relatively expensive to compute. It is presented because it is an algorithm that can be readily implemented in MacAnova. In actual practice, one would prefer to use an algorithm directly to minimize $\sum_{m+1 \le i \le p} [\hat{\delta}_j(\hat{\boldsymbol{\Psi}})]^2$ such as is used by macro **facanal()** illustrated below.

First, I used $\hat{\boldsymbol{\Psi}}$ from the principal component method to start things off.

```
Cmd> psihat <- stepuls(r,psihat2,2,print:T) # do one step
WARNING: searching for unrecognized macro stepuls near psihat <-
stepuls(
psi:          Printed because of print:T
(1)   0.379274   0.223135   0.101921   0.094658   0.098008   0.085145
criterion:   Printed because of print:T
(1)   0.047746   Measure of lack of fit

Cmd> psihat
component: psi       Uniquenesses
(1)   0.379274   0.223135   0.101921   0.094658   0.098008   0.085145
component: loadings
                (1)        (2)
SklLngth   0.710270  -0.340945      factor loadings
SklBrdth   0.674893  -0.566908
FemLngth   0.923629   0.212106
TibLngth   0.937092   0.164924
HumLngth   0.943758   0.106361
UlnLngth   0.938349   0.185357
component: crit
(1)   0.047746   Value of criterion for psihat2
```

```
Cmd> print(PSI,LOADINGS,CRITERION) # print the "side effect" variables
PSI:
(1)   0.379274   0.223135   0.101921   0.094658   0.098008   0.085145
LOADINGS:
                (1)        (2)
SklLngth   0.710270  -0.340945
SklBrdth   0.674893  -0.566908
FemLngth   0.923629   0.212106
TibLngth   0.937092   0.164924
HumLngth   0.943758   0.106361
UlnLngth   0.938349   0.185357
CRITERION:
(1)   0.047746
Cmd> dev <- r - (LOADINGS %*% LOADINGS' + dmat(PSI)) #residuals
Cmd> vector(sum(vector(dev)^2),max(abs(vector(dev))))
(1)   0.025150   0.089631   SS residuals and max(abs(resids)
```

Now we do iterated principal factor extraction in earnest, using starting values $\hat{\psi}_i = 1/r^{ii}$, where r^{ii} are the diagonal elements of $\mathbf{R}^{-1}$.

```
Cmd> psi0 <- 1/diag(solve(r)); psi0 # initial values
(1)   0.528691   0.565077   0.121657   0.109907   0.099535   0.096204
Cmd> psihat <- stepuls(r,psi0,2,print:T) # one step
psi:
(1)   0.459160   0.502397   0.113128   0.101689   0.096087   0.092890
crit:
(1)   0.025713   lower than when starting from PCA solution
Cmd> dev <- r - (LOADINGS %*% LOADINGS' + dmat(PSI))
Cmd> vector(sum(vector(dev)^2),max(abs(vector(dev))))
(1)   0.016787   0.065824
Cmd> psihat # complete structure returned by stepuls()
component: psi
(1)   0.459160   0.502397   0.113128   0.101689   0.096087   0.092890
component: loadings
                (1)        (2)
SklLngth   0.670382  -0.302372
SklBrdth   0.618399  -0.339391
FemLngth   0.926237   0.170165
TibLngth   0.940972   0.113502
HumLngth   0.950231   0.031209
UlnLngth   0.944115   0.125526
component: crit
(1)   0.025713
```

```
Cmd> iter <- 1 # let's keep count of the number of iterations

Cmd> n <- 5; for(i,1,n){
     iter <- iter+1; psihat<-stepuls(r,psihat,2,print:T);;}
psi:      After iteration 2
(1)   0.425537   0.470137   0.110469   0.099550   0.096978   0.092299
crit:
(1)   0.012390
psi:      After iteration 3
(1)   0.409382   0.453321   0.109581   0.098979   0.098062   0.092139
crit:
(1)   0.009116
psi:      After iteration 4
(1)   0.401896   0.444252   0.109276   0.098831   0.098770   0.092021
crit:
(1)   0.008295
psi:      After iteration 5
(1)   0.398733   0.439059   0.109183   0.098801   0.099171   0.091908
crit:
(1)   0.008085
psi:      After iteration 6
(1)   0.397725   0.435802   0.109169   0.098804   0.099386   0.091809
crit:
(1)   0.008028
```

The value of crit keeps decreasing.

```
Cmd> n <- 100;for(i,1,n){ # 100 more without printing
iter <- iter+1; psihat <- stepuls(r,psihat,2,print:F);;}

Cmd> dev <- r - (LOADINGS %*% LOADINGS' + dmat(PSI))

Cmd> vector(sum(vector(dev)^2),max(abs(vector(dev))))
(1)   0.008016   0.036081   Not much smaller

Cmd> n <- 100; for(i,1,n){ # another 100 without printing
iter <- iter+1; psihat <- stepuls(r,psihat,2,print:F);;} #100 more

Cmd> psihat
component: psi
(1)   0.456472   0.332882   0.112318   0.099348   0.099511   0.088939
component: loadings
                (1)        (2)
SklLngth   0.680221  -0.284302
SklBrdth   0.651070  -0.493179
FemLngth   0.923895   0.184661
TibLngth   0.939103   0.136884
HumLngth   0.946334   0.070292
UlnLngth   0.941313   0.158084
component: crit
(1)   0.007794                    Some reduction from .008028

Cmd> dev <- r - (LOADINGS %*% LOADINGS' + dmat(PSI))

Cmd> vector(sum(vector(dev)^2),max(abs(vector(dev))))
(1)   0.007794   0.034970
```

```
Cmd> n <- 500; for(i,run(n)){
iter <- iter+1; psihat <- stepuls(r,psihat,2,print:F);;} #500 more

Cmd> psihat
component: psi
(1)   0.463469   0.315316   0.112895   0.099432   0.099455   0.088573
component: loadings
                (1)        (2)
SklLngth   0.679132  -0.274429
SklBrdth   0.653753  -0.507240
FemLngth   0.923633   0.184411
TibLngth   0.938951   0.137615
HumLngth   0.946231   0.072056
UlnLngth   0.941245   0.159639
component: crit
(1)   0.007791                    Very little change from .007094

Cmd> sigmahat <- LOADINGS %*% LOADINGS' + dmat(PSI)

Cmd> dev <- r - sigmahat # residuals

Cmd> vector(sum(vector(dev)^2),max(abs(vector(dev))))
(1)   0.007791   0.034741

Cmd> iter# total number of iterations
(1) 706.000000      Fewer would have been adequate

Cmd> sigmahat # estimated rho from two factors
         SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
SklLngth 1.000000   0.583185   0.576661   0.599906   0.622841   0.595420
SklBrdth 0.583185   1.000000   0.510287   0.544038   0.582051   0.534367
FemLngth 0.576661   0.510287   1.000000   0.892624   0.887258   0.898804
TibLngth 0.599906   0.544038   0.892624   1.000000   0.898381   0.905752
HumLngth 0.622841   0.582051   0.887258   0.898381   1.000000   0.902138
UlnLngth 0.595420   0.534367   0.898804   0.905752   0.902138   1.000000

Cmd> dev # errors in fit
          SklLngth   SklBrdth   FemLngth  TibLngth   HumLngth   UlnLngth
SklLngth  0.000000  -0.000176  -0.007550  0.002353  -0.001722   0.006913
SklBrdth -0.000176   0.000000   0.005023  0.003561   0.001501  -0.009862
FemLngth -0.007550   0.005023   0.000000  0.033481  -0.010036  -0.021351
TibLngth  0.002353   0.003561   0.033481  0.000000  -0.024753  -0.012142
HumLngth -0.001722   0.001501  -0.010036 -0.024753   0.000000   0.034741
UlnLngth  0.006913  -0.009862  -0.021351 -0.012142   0.034741   0.000000

Cmd> LOADINGS' %*% LOADINGS #cols of loadings orthogonal
            (1)        (2)
(1)   4.404635   0.000000
(2)   0.000000   0.416225
```

The underlined value is the largest residual from the two factor fit.

Direct minimization of ULS criterion

You can directly minimize of the ULS criterion using **facanal()**. The simplest usage is

```
Cmd> result <- facanal(r, m, method:"uls")
```

where **m** is the number of factors. You can suppress printed output by **silent:T**,

specify the minimum and/or maximum number of iterations by `minit:n1` (default is 1) and `maxit:n2` (default is 30), or provide starting values by `start:psi0` (default is `1/diag(solve(r))`).

```
Cmd> result <- facanal(r,2,method:"uls")
Convergence in 21 iterations by criterion 2
estimated uniquenesses:
   SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
   0.463630   0.314887   0.113034   0.099155   0.099355   0.088849
unrotated estimated loadings:
           Factor 1   Factor 2
SklLngth   0.679107  -0.274192
SklBrdth   0.653819  -0.507574
FemLngth   0.923601   0.184379
TibLngth   0.939007   0.137740
HumLngth   0.946249   0.072118
UlnLngth   0.941183   0.159581
minimized uls criterion:
(1)   0.003895
Cmd> compnames(result)
(1) "psihat"
(2) "loadings"
(3) "criterion"
(4) "eigenvals"
(5) "gradient"
(6) "method"
(7) "rotation"
(8) "iter"
(9) "status"
Cmd> result # full contents of result
component: psihat
   SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
   0.463630   0.314887   0.113034   0.099155   0.099355   0.088849
component: loadings
           Factor 1   Factor 2
SklLngth   0.679107  -0.274192
SklBrdth   0.653819  -0.507574
FemLngth   0.923601   0.184379
TibLngth   0.939007   0.137740
HumLngth   0.946249   0.072118
UlnLngth   0.941183   0.159581
component: criterion
(1)   0.003895
component: eigenvals
(1)   4.404653   0.416448   0.069697   0.004090  -0.027071  -0.046728
component: gradient      Should be close to 0
(1)  -0.000001  -0.000001   0.000008  -0.000014  -0.000006   0.000013
component: method
(1) "uls"
component: rotation
(1) "none"
component: iter
```

```
(1)  21.000000
component: status
(1)   2.000000
```

```
Cmd> loadings <- LOADINGS # or loadings <- result$loadings; save L
```

You can use `rotation()` to rotate the factor loadings. The general usage is `rotloadings <- rotation(loadings,method:meth)` or `rotloadings <- rotation(loadings,method:meth,kaiser:T)`. The latter form does Kaiser normalization and is preferred. Permissible values for `meth` are `"varimax"` (default), `"quartimax"` and `"equimax"`.

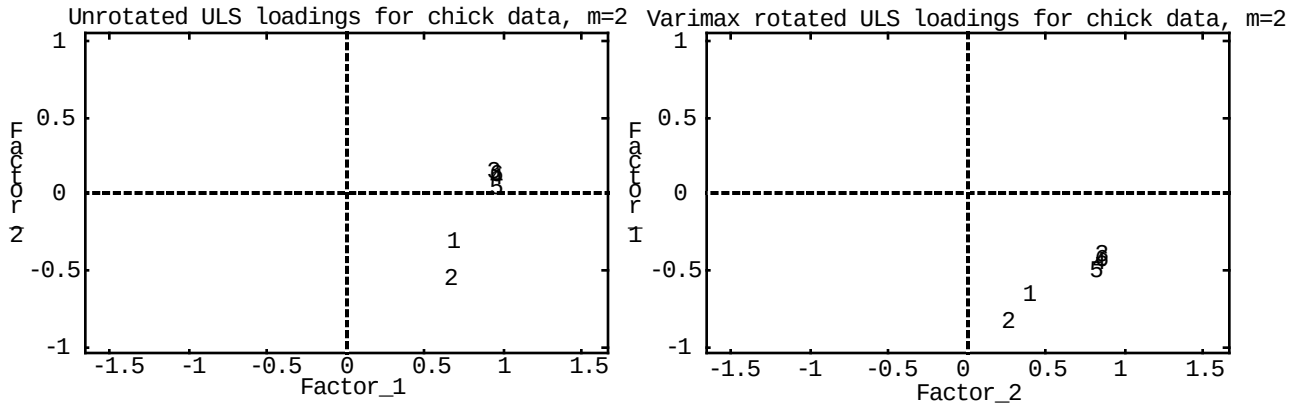
```
Cmd> rotation(loadings,kaiser:T) #same with method:"varimax"
                 (1)        (2)
SklLngth   0.422066  -0.598522
SklBrdth   0.274313  -0.780938
FemLngth   0.876091  -0.345686
TibLngth   0.863758  -0.393229
HumLngth   0.834274  -0.452301
UlnLngth   0.877425  -0.376054
```

You can do factor extraction and rotation with Kaiser normalization in one step:

```
Cmd> result_rot <- facanal(r,2,method:"uls",rotate:"varimax")
Convergence in 21 iterations by criterion 2
estimated uniquenesses:
   SklLngth   SklBrdth   FemLngth   TibLngth   HumLngth   UlnLngth
   0.463630   0.314887   0.113034   0.099155   0.099355   0.088849
varimax rotated estimated loadings:
           Factor 1   Factor 2  Note factor 2 has different sign
SklLngth   0.422066   0.598522
SklBrdth   0.274313   0.780938
FemLngth   0.876091   0.345686
TibLngth   0.863758   0.393229
HumLngth   0.834274   0.452301
UlnLngth   0.877425   0.376054
minimized uls criterion:
(1)   0.003895
```

```
Cmd> # Scatter plot of unrotated and varimax rotated factor loadings
Cmd> plot(loadings[,1],loadings[,2], xmin:-1.6,symbols:run(6),\
```

```
      xmax:1.6,ymin:-1,ymax:1,xlab:"Factor_1",ylab:"Factor_2",\
      title:"Unrotated ULS loadings for chick data, m=2")
Cmd> loadings_rot <- result_rot$loadings # rotated loadings
Cmd> plot(loadings_rot[,1],-loadings_rot[,2],xmin:-1.6,symbols:run(6),\
      xmax:1.6,ymin:-1,ymax:1,xlab:"Factor_2",ylab:"Factor_1",\
      title:"Varimax rotated ULS loadings for chick data, m=2")
```



I changed the sign of rotated factors 1 and 2 to undo a sign change that **rotation()** made. The values for xmin and xmax were chosen so that the horizontal and vertical scales would be about the same.