

Displays for Statistics 5401/8401

Lecture 25

November 4, 2005

Christopher Bingham, Instructor

612-625-1024, kb@umn.edu  
372 Ford Hall

Class Web Page

<http://www.stat.umn.edu/~kb/classes/5401>

© 2005 by Christopher Bingham

### Scale Dependence

The SVD  $\mathbf{X} = \mathbf{L}\mathbf{T}\mathbf{R}'$ , PC's  $\mathbf{z}_j = t_j \mathbf{L}_j = \mathbf{X}\mathbf{r}_j$  and the rank  $m$  approximation  $\mathbf{X}^{(m)} = \sum_{1 \leq j \leq m} \mathbf{z}_j \mathbf{r}_j'$ , are very dependent the scales of the columns of  $\mathbf{X}$ .

Suppose  $\mathbf{D} = \text{diag}[d_1, \dots, d_p]$  is diagonal. Then

$$\mathbf{X}^* = [\mathbf{X}_1/d_1, \mathbf{X}_2/d_2, \dots, \mathbf{X}_p/d_p] = \mathbf{X}\mathbf{D}^{-1}$$

is a *scaled version* of  $\mathbf{X}$ , for example by changing measurement units.

- Except when  $d_1 = d_2 = \dots = d_p$ , there is *no simple* relationship between the SVD  $\mathbf{L}^* \mathbf{T}^* \mathbf{R}^{*'} for  $\mathbf{X}^*$  and the SVD for  $\mathbf{X}$ . It's true that  $\mathbf{X}^* = \mathbf{L}\mathbf{T}\tilde{\mathbf{R}}' = \sum_{1 \leq j \leq p} t_j \mathbf{L}_j \tilde{\mathbf{r}}_j'$ ,  $\tilde{\mathbf{R}} = [\tilde{\mathbf{r}}_1, \dots, \tilde{\mathbf{r}}_p] \equiv [\mathbf{D}^{-1}\mathbf{r}_1, \dots, \mathbf{D}^{-1}\mathbf{r}_p]$  but  $\mathbf{L}\mathbf{T}\tilde{\mathbf{R}}'$  is not the SVD of  $\mathbf{X}^*$  ( $\tilde{\mathbf{R}}'\tilde{\mathbf{R}} \neq \mathbf{I}_p$ ).$
- And,  $\mathbf{X}^{(m)}\mathbf{D}^{-1} = \sum_{1 \leq j \leq m} t_j \mathbf{L}_j \tilde{\mathbf{r}}_j'$ ,  $\tilde{\mathbf{r}}_j = \mathbf{D}^{-1}\mathbf{r}_j$  is *not* the best rank  $m$  approximation to  $\mathbf{X}^*$  in the least squares sense.

## Example from Monday continued

Compute and print approximations to  $x$  of ranks 1, 2, 3, and 4 using original scaling.

```

Cmd> x <- run(10)^run(0,3)' # powers of run(10)
Cmd> setlabels(x,\
      structure("@",vector("i^0","i^1","i^2","i^3")))
Cmd> x # 10 ny 4 matrix
      i^0      i^1      i^2      i^3
(1)      1        1        1        1
(2)      1        2        4        8
(3)      1        3        9       27
(4)      1        4       16       64
(5)      1        5       25      125
(6)      1        6       36      216
(7)      1        7       49      343
(8)      1        8       64      512
(9)      1        9       81      729
(10)     1       10      100     1000

Cmd> svdresults <- svd(x,all:T)
Cmd> svdresults$values #just Sing values
(1)      1415.4      27.14      2.2961      0.41587

```

Because the two smallest singular values are so much smaller than the largest, one might say  $x$  is "almost of rank 2".

The `for` loop in the following `MacAnova` command computes and prints successively rank  $m$  approximations using  $m = 1, \dots, 4$  singular values and vectors.

```

Cmd> for(i,run(4)){
      left  <- svdresults$leftvectors[,run(i)]
      right <- svdresults$rightvectors[,run(i)]
      tmatrix <- dmat(svdresults$values[run(i)])
      approx <- left %*% tmatrix %*% right'
      print(approx, name=paste("Rank",i,"approximation"))
    }
Rank 1 approximation:
      i^0      i^1      i^2      i^3
(1)      0.0017032  0.014251  0.12416  1.112
(2)      0.012818  0.10725  0.93436  8.3681
(3)      0.042422  0.35494  3.0923  27.695
(4)      0.099592  0.83328  7.2597  65.018
(5)      0.1934    1.6182  14.098  126.26
(6)      0.33293  2.7856  24.269  217.35
(7)      0.52725  4.4115  38.434  344.21
(8)      0.78544  6.5718  57.255  512.77
(9)      1.1166   9.3424  81.393  728.95
(10)     1.5297  12.799  111.51  998.68

```

Last column is fit pretty well, others quite poorly.

```

Rank 2 approximation:
      Better fit
      i^0      i^1      i^2      i^3
(1)      0.082063  0.34448  1.2546  0.98138
(2)      0.25314  1.0948  4.3149  7.9776
(3)      0.48224  2.1623  9.2793  26.98
(4)      0.7338   3.4394  16.181  63.987
(5)      0.97227  4.8188  25.054  125
(6)      1.1621   6.1929  35.933  216
(7)      1.2677   7.4541  48.85   343.01
(8)      1.2535   8.4951  63.839  512.01
(9)      1.084    9.2083  80.934  729
(10)     0.72351  9.4862  100.17  999.99

```

```

Rank 3 approximation:      Still better fit
      i^0      i^1      i^2      i^3
(1)  0.74505    1.1878    0.96351    1.0021
(2)  0.97771    2.0164    3.9968     8.0002
(3)  1.0892     2.9343    9.0128    26.999
(4)  1.1105     3.9186    16.016    63.999
(5)  1.0727     4.9465    25.01     125
(6)  1.0066     5.9951    36.001    216
(7)  0.94342    7.0417    48.992    343
(8)  0.91403    8.0633    63.988    512
(9)  0.94946    9.0372    80.993    729
(10) 1.0807     9.9406    100.01    1000

Rank 4 approximation:      = LTR' = x
      i^0      i^1      i^2      i^3
(1)  1          1          1          1
(2)  1          2          4          8
(3)  1          3          9          27
(4)  1          4          16         64
(5)  1          5          25         125
(6)  1          6          36         216
(7)  1          7          49         343
(8)  1          8          64         512
(9)  1          9          81         729
(10) 1          10         100        1000
    
```

The last is an exact reconstruction of  $x$ .

Rescale columns so that their sums of squares = 10, dividing column  $l$  by  $d_l = \sqrt{\{\sum x_{il}^2\}}/\sqrt{10}$ , and do the same.

```

Cmd> x # original matrix
      i^0      i^1      i^2      i^3
(1)  1          1          1          1
(2)  1          2          4          8
(3)  1          3          9          27
(4)  1          4          16         64
(5)  1          5          25         125
(6)  1          6          36         216
(7)  1          7          49         343
(8)  1          8          64         512
(9)  1          9          81         729
(10) 1          10         100        1000
    
```

```
Cmd> d <- sqrt(sum(x^2))/sqrt(10)
```

```
Cmd> scaledx <- x / d #divide each row by d
```

```

Cmd> scaledx # scaled matrix
      (1)      (2)      (3)      (4)
(1)  1      0.16116  0.019868  0.0022482
(2)  1      0.32233  0.079472  0.017986
(3)  1      0.48349  0.17881  0.060702
(4)  1      0.64466  0.31789  0.14389
(5)  1      0.80582  0.4967  0.28103
(6)  1      0.96699  0.71525  0.48562
(7)  1      1.1282  0.97354  0.77115
(8)  1      1.2893  1.2716  1.1511
(9)  1      1.4505  1.6093  1.639
(10) 1      1.6116  1.9868  2.2482
    
```

```

Cmd> sum(scaledx^2) # Column SS are all 10 as promised
      i^0      i^1      i^2      i^3
(1)  10         10         10         10
    
```

```
Cmd> svdresults1 <- svd(scaledx,all:T) # SVD of rescaled x
```

```

Cmd> svdresults1$values # singular values of scaled x
(1)  6.0089    1.9225    0.44084    0.051852
    
```

```

Cmd> svdresults1$values # singular values of unscaled x
(1)  1415.4    27.14     2.2961    0.41587
    
```

This computes approximations to scaled  $x$  and then "unscales" by multiplying by  $d$ , so they should be close to  $x$ :

```
Cmd> for(i,run(4)){
  left  <- svdresults1$leftvectors[,run(i)]
  right <- svdresults1$rightvectors[,run(i)]
  tmatrix <- dmat(svdresults1$values[run(i)])
  approx <- (left %**% tmatrix %**% right') * d
  print(approx,labels:F,\
        name:paste("Rank",i,"approximation"))}
```

Rank 1 approximation: **Not good at all**

0.25452	1.7995	14.429	123.02
0.31094	2.1985	17.628	150.29
0.38296	2.7077	21.711	185.11
0.47367	3.349	26.853	228.95
0.58615	4.1444	33.23	283.32
0.7235	5.1155	41.016	349.71
0.8888	6.2842	50.387	429.61
1.0851	7.6724	61.518	524.51
1.3156	9.3019	74.583	635.91
1.5833	11.194	89.758	765.29

Rank 2 approximation: **A little better**

0.90039	2.2239	2.0124	-59.496
0.94675	2.6162	5.4043	-29.382
0.988	3.1052	10.079	14.125
1.0217	3.7091	16.316	74.068
1.0456	4.4462	24.398	153.49
1.0571	5.3346	34.603	255.44
1.0539	6.3927	47.214	382.96
1.0336	7.6385	62.51	539.09
0.99369	9.0904	80.772	726.88
0.9319	10.767	102.28	949.36

Rank 3 approximation: **Close except for column 4**

0.99753	1.0804	-0.109	6.1239
1.0005	1.9833	4.2302	6.9364
1.0017	2.9435	9.7789	23.401
1.0017	3.9453	16.755	60.514
1.0008	4.9728	25.375	123.27
0.99968	6.0104	35.857	216.66
0.99871	7.0421	48.419	345.69
0.99839	8.0524	63.278	515.34
0.99922	9.0253	80.651	730.61
1.0017	9.9451	100.76	996.5

Rank 4 approximation:	Perfect	fit	with all 4
1	1	1	1
1	2	4	8
1	3	9	27
1	4	16	64
1	5	25	125
1	6	36	216
1	7	49	343
1	8	64	512
1	9	81	729
1	10	100	1000

### MacAnova note:

In the loop,  $* d$  multiplies each row of the approximation by the elements of row vector  $d$ . This restores the original scaling of  $x$ .

With  $m = 3$ ,  $x[,1]$  and  $x[,2]$  are fit reasonably well, but not  $x[,3]$  and  $x[,4]$ .

In any event, the approximations from the SVD of  $scaledx$  are different from the approximations from the SVD of  $x$ .

SVD of  $\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{1}_N \bar{\mathbf{x}}'$  (resids from mean) is related to eigenvalues  $\hat{\lambda}_1, \hat{\lambda}_2, \dots, \hat{\lambda}_p$  and eigenvectors  $[\hat{\mathbf{v}}_1, \dots, \hat{\mathbf{v}}_p]$  of the sample variance matrix  $\mathbf{S} = (1/f_e)\tilde{\mathbf{X}}'\tilde{\mathbf{X}}$ ,  $f_e = N-1$ .

- Singular values of  $\tilde{\mathbf{X}}$  are  $t_j = \sqrt{\{f_e \hat{\lambda}_j\}}$   
The eigenvalues of  $\mathbf{S}$  are  $\hat{\lambda}_j = t_j^2/f_e$
- The right singular vector  $\mathbf{r}_j = \hat{\mathbf{v}}_j$ , where  $\hat{\mathbf{v}}_j$  is an eigenvector of  $\mathbf{S}$
- The left singular vectors  $\mathbf{L}_j$  are eigenvectors  $\hat{\mathbf{l}}_j$  of  $\tilde{\mathbf{X}}\tilde{\mathbf{X}}'$  with eigenvalues  $f_e \hat{\lambda}_1 = t_1^2, \dots, f_e \hat{\lambda}_p = t_p^2, 0, \dots, 0$  (N-p 0's)

$-\mathbf{L}_j$  and  $-\mathbf{r}_j$  (changing signs of both) are equally valid singular vectors. The choice of signs may differ between programs.

Similarly,  $\mathbf{L}_j = \pm \hat{\mathbf{l}}_j, \mathbf{r}_j = \pm \hat{\mathbf{v}}_j$ .

Continuing with PC's from the point of view of finding a low rank approximation to  $\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{1}_N \bar{\mathbf{x}}' = \sum_{1 \leq j \leq p} (t_j \mathbf{L}_j) \mathbf{r}_j' = \sum_{1 \leq j \leq p} t_j \hat{\mathbf{l}}_j \hat{\mathbf{v}}_j'$   
 $\hat{\mathbf{l}}_j$  and  $\hat{\mathbf{v}}_j$  are eigenvectors of  $\tilde{\mathbf{X}}'\tilde{\mathbf{X}}$  and  $\tilde{\mathbf{X}}\tilde{\mathbf{X}}'$

Because  $t_j = \sqrt{\{f_e \hat{\lambda}_j\}}$ ,  $\hat{\lambda}_j$  eigenvalue of  $\mathbf{S}$

- $\tilde{\mathbf{X}} = \sum_{1 \leq j \leq p} (\sqrt{\{f_e \hat{\lambda}_j\}} \mathbf{L}_j) \mathbf{r}_j' = \sum_{1 \leq j \leq p} \tilde{\mathbf{z}}_j \mathbf{r}_j'$   
where  $\tilde{\mathbf{z}}_j \equiv t_j \mathbf{L}_j = \sqrt{\{f_e \hat{\lambda}_j\}} \times \mathbf{L}_j = \tilde{\mathbf{X}} \mathbf{r}_j = \tilde{\mathbf{X}} \hat{\mathbf{v}}_j$
- $\tilde{\mathbf{z}}_j = (1/N) \mathbf{1}_N' \tilde{\mathbf{z}}_j = 0$  (because  $\mathbf{1}_N' \tilde{\mathbf{X}} = \mathbf{0}$ )
- The best rank m approximation to  $\tilde{\mathbf{X}}$  is

$$\tilde{\mathbf{X}}^{(m)} = \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}_j' = \tilde{\mathbf{X}} \sum_{1 \leq j \leq m} \mathbf{r}_j \mathbf{r}_j'$$

**Reminder:**  $\pm \mathbf{r}_j = \hat{\mathbf{v}}_j$  is the  $j^{\text{th}}$  eigenvector of  $\mathbf{S}$  and satisfies  $\mathbf{S} \mathbf{r}_j = \hat{\lambda}_j \mathbf{r}_j$ , so you can compute PCs by either using the SVD to find  $\mathbf{r}_j$  or by computing eigenvectors of  $\mathbf{S}$ .

**MacAnova:** Compute all PCs by

```
Z <- (X - xbar') %*% eigen(s)$vectors
```

Here's how to think about this.

- Each  $N \times 1$  principal component vector

$$\tilde{\mathbf{z}}_j = \tilde{\mathbf{X}} \mathbf{r}_j = \sum_{1 \leq \ell \leq p} r_{\ell j} \tilde{\mathbf{X}}_\ell$$

is a new variable, a linear combination of the columns  $\tilde{\mathbf{X}}_\ell$  of  $\tilde{\mathbf{X}}$ . The coefficients in the linear combination come from  $\mathbf{r}_j$ , the  $j^{\text{th}}$  eigenvector of  $\mathbf{S}$ .

- Each column  $\tilde{\mathbf{X}}_\ell^{(m)}$  of the rank  $m$  approximation  $\tilde{\mathbf{X}}^{(m)}$  is a linear combination of the first  $m$  PC's

$$\begin{aligned} \tilde{\mathbf{X}}_\ell^{(m)} &= \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j r_{\ell j} \\ &= \text{column } \ell \text{ of } \tilde{\mathbf{X}}^{(m)} = \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}_j'. \end{aligned}$$

The coefficients of  $\tilde{\mathbf{z}}_j \{r_{\ell j}\}_{1 \leq j \leq m}$  are from row  $\ell$  of  $\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_m$

So the elements of  $[\mathbf{r}_1 \ \mathbf{r}_2 \ \dots \ \mathbf{r}_m]$  play a role in defining principal components and in approximating  $\tilde{\mathbf{X}}$  by  $\tilde{\mathbf{X}}^{(m)}$ .

- The "total sum of squares" is

$$\begin{aligned} \|\tilde{\mathbf{X}}\|^2 &= \sum_{1 \leq \ell \leq p} \left\{ \sum_{1 \leq i \leq N} (x_{i\ell} - \bar{x}_{\ell})^2 \right\} \\ &= f_e \sum_{1 \leq \ell \leq p} s_{\ell\ell} = f_e \text{trace}(\mathbf{S}) \\ &= f_e \sum_{1 \leq j \leq p} \hat{\lambda}_j = \sum_{1 \leq j \leq p} t_j^2. \end{aligned}$$

Here  $s_{\ell\ell} = s_{\ell}^2 = \sum_{1 \leq i \leq N} (x_{i\ell} - \bar{x}_{\ell})^2 / f_e$  are the sample variances for each column of  $\mathbf{X}$ .

- The "residual sum of squares" is

$$\|\tilde{\mathbf{X}} - \tilde{\mathbf{X}}^{(m)}\|^2 = f_e \sum_{m+1 \leq j \leq p} \hat{\lambda}_j = \sum_{m+1 \leq j \leq p} t_j^2$$

- Relative RSS (analogous to  $1 - R^2$ ) is

$$\begin{aligned} &\|\tilde{\mathbf{X}} - \tilde{\mathbf{X}}^{(m)}\|^2 / \|\tilde{\mathbf{X}}\|^2 \\ &= \sum_{m+1 \leq j \leq p} t_j^2 / \sum_{1 \leq j \leq p} t_j^2 = \sum_{m+1 \leq j \leq p} \hat{\lambda}_j / \sum_{1 \leq j \leq p} \hat{\lambda}_j \\ &= \sum_{m+1 \leq j \leq p} \hat{\lambda}_j / \sum_{1 \leq \ell \leq p} s_{\ell\ell} = 1 - \left( \sum_{1 \leq j \leq m} \hat{\lambda}_j / \sum_{1 \leq j \leq p} \hat{\lambda}_j \right) \\ &= 1 - \left( \sum_{1 \leq j \leq m} \hat{\lambda}_j / \sum_{1 \leq \ell \leq p} s_{\ell\ell} \right) \end{aligned}$$

Thus  $\sum_{1 \leq j \leq m} \hat{\lambda}_j / \sum_{1 \leq j \leq p} \hat{\lambda}_j = \sum_{1 \leq j \leq m} \hat{\lambda}_j / \sum_{1 \leq \ell \leq p} s_{\ell\ell}$  is the proportion of the total SS "explained" by  $\tilde{\mathbf{z}}_1, \dots, \tilde{\mathbf{z}}_m$ .

$\tilde{\mathbf{X}}^{(m)}$  should be a good rank- $m$  approximation to  $\tilde{\mathbf{X}}$  when the "trailing" eigenvalues  $\hat{\lambda}_{m+1}, \dots, \hat{\lambda}_p$  are small compared to  $\sum_{1 \leq j \leq p} \hat{\lambda}_j$ , so that  $\sum_{1 \leq j \leq m} \hat{\lambda}_j / \sum_{1 \leq j \leq p} \hat{\lambda}_j \approx 1$ .

You can add back the mean  $\bar{\mathbf{x}}$  to get a rank  $m+1$  approximation  $\mathbf{X}^{(m)*}$  for  $\mathbf{X}$ :

$$\mathbf{X}^{(m)*} = \mathbf{1}_N \bar{\mathbf{x}}' + \tilde{\mathbf{X}}^{(m)} = (\mathbf{1}_N \bar{\mathbf{x}}') + \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}_j'$$

Since  $\|\tilde{\mathbf{X}} - \tilde{\mathbf{X}}^{(m)}\|^2 = \|\mathbf{X} - \mathbf{X}^{(m)*}\|^2$ ,  $\mathbf{X}^{(m)*}$  is a good approximation to  $\mathbf{X}$  whenever  $\tilde{\mathbf{X}}^{(m)}$  is a good approximation to  $\tilde{\mathbf{X}}$ .

$\mathbf{X}^{(m)*}$  is a rank  $m+1$  approximation, but not the *best* rank  $m+1$  approximation  $\mathbf{X}^{(m+1)}$ . We prefer  $\mathbf{X}^{(m)*}$  to  $\mathbf{X}^{(m+1)}$  because one of the outer products  $(\mathbf{1}_N \bar{\mathbf{x}}')$  has an immediate interpretation as a matrix all of whose rows are  $\bar{\mathbf{x}}$ .

You can generalize this to a linear model situation when  $\mathbf{X} = \mathbf{UB} + \boldsymbol{\varepsilon}$ , where

- the  $k+1$  columns  $\mathbf{U}_0, \mathbf{U}_1, \dots, \mathbf{U}_k$  of  $\mathbf{U}$  are predictor (independent) variables, possibly dummy variables
- $\mathbf{B} = [\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_k]'$  is a matrix of coefficients (not the same as J&W use of  $\mathbf{B}$  as "between" groups matrix).

The estimate  $\hat{\mathbf{B}}$  has rows  $\hat{\boldsymbol{\beta}}_0', \hat{\boldsymbol{\beta}}_1', \dots, \hat{\boldsymbol{\beta}}_k'$  and the "fitted values" are

$$\hat{\mathbf{X}} = \mathbf{UB}\hat{\mathbf{B}} = \sum_{0 \leq j \leq k} \mathbf{U}_j \hat{\boldsymbol{\beta}}_j'$$

$\hat{\mathbf{X}}$  is a rank  $k+1$  approximation to  $\mathbf{X}$ , just as  $\mathbf{1}_N \bar{\mathbf{x}}'$  is a rank 1 approximation to  $\mathbf{X}$ .

$\hat{\mathbf{X}}$  is not the best rank  $k+1$  since it is not constructed from the SVD of  $\mathbf{X}$ .

Consider  $\tilde{\mathbf{X}} \equiv \mathbf{X} - \mathbf{U}\hat{\mathbf{B}}$ , the matrix of residuals from the linear model.

To understand the structure of  $\tilde{\mathbf{X}}$  it is natural to look at the rank  $m$  approximation  $\tilde{\mathbf{X}}^{(m)} = \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}_j'$  for  $\tilde{\mathbf{X}}$ . The PC's  $\tilde{\mathbf{z}}_j = \tilde{\mathbf{X}} \mathbf{r}_j = t_j \mathbf{L}_j$  computed from the matrix  $\tilde{\mathbf{X}}$  of residuals are multiples of  $\tilde{\mathbf{X}}$ 's left singular vectors  $\mathbf{L}_j$ .  $\mathbf{r}_j$  is an eigenvector of  $\mathbf{S} = \mathbf{f}_e^{-1} \mathbf{E}$ .

Then

$$\mathbf{U}\hat{\mathbf{B}} + \tilde{\mathbf{X}}^{(m)} = \sum_{0 \leq j \leq k} \mathbf{U} \hat{\boldsymbol{\beta}}_j' + \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}_j'$$

is a rank  $k+m+1$  approximation to  $\mathbf{X}$  with the first  $k+1$  components having an interpretation as predictors. It is not the "best" rank  $k+m+1$  approximation to  $\mathbf{X}$ . That would be  $\mathbf{X}^{(k+m+1)}$  computed from the SVD of  $\mathbf{X}$ .

```

Cmd> irisdata <- read("", "t11_05", silent:T)
Cmd> x <- irisdata[, -1]; xbar <- describe(x, mean:T) #col vector
Cmd> svdstuff <- svd(x - xbar', all:T) # resid from mean
Cmd> svals <- svdstuff$values; leftvecs <- svdstuff$leftvectors
Cmd> svals # Singular values
(1)      25.1      6.0131      3.4137      1.8845
Cmd> rightvecs <- svdstuff$rightvectors
Cmd> xhat_2 <- svals[1]*leftvecs[,1] %*% rightvecs[,1]' + \
            svals[2]*leftvecs[,2] %*% rightvecs[,2]' + xbar'
```

$\mathbf{xhat\_2} = \tilde{\mathbf{X}}^{(2)} + \mathbf{1}_N \bar{\mathbf{x}}' =$  approximation to  $\mathbf{X}$

```

Cmd> sum(vector(x - xhat_2)^2) # residual SS
(1)      15.205

Cmd> sum(svals[-run(2)]^2) # t_3^2 + t_4^2 (trailing sing vals)
(1)      15.205

Cmd> sum(svals[run(2)]^2)/sum(svals^2)
(1)      0.97769      97.8% of variability "explained"

Cmd> head(x, 8) # first 8 rows of x
      SepLen      SepWid      PetLen      PetWid
(1)      5.1      3.5      1.4      0.2 Original
(2)      4.9      3      1.4      0.2 values of
(3)      4.7      3.2      1.3      0.2 x_ij
(4)      4.6      3.1      1.5      0.2
(5)      5      3.6      1.4      0.2
(6)      5.4      3.9      1.7      0.4
(7)      4.6      3.4      1.4      0.3
(8)      5      3.4      1.5      0.2

Cmd> print(head(xhat_2, 8), format=".2f") #first 8 rows of xhat_2
MATRIX:      Pretty good match of rank 2 approximation
      SepLen      SepWid      PetLen      PetWid
(1)      5.08      3.52      1.40      0.21 These are the
(2)      4.75      3.16      1.46      0.24 approximate
(3)      4.70      3.20      1.31      0.18 values of x_ij
(4)      4.64      3.06      1.46      0.24 computed using
(5)      5.07      3.53      1.36      0.20 the first two
(6)      5.51      3.79      1.68      0.33 principal com-
(7)      4.77      3.23      1.36      0.20 ponents
(8)      5.00      3.40      1.48      0.25
```



## MacAnova Notes:

- `head(x)` gets the first 10 rows of `x`.
- `head(x,n)` gets the first `n` rows of `x`.
- `tail(x)` gets the last 10 rows of `x`.
- `tail(x,n)` gets the last `n` rows of `x`.

There is some ambiguity as to how the principal components associated with a  $N \times p$  data matrix  $\mathbf{X}$  are defined:

- Linear combinations  $\mathbf{Z}_j = \mathbf{X}\mathbf{r}_j$  of the data themselves

or, as I have defined them,

- Linear combinations  $\tilde{\mathbf{Z}}_j = \tilde{\mathbf{X}}\mathbf{r}_j$  of residuals in  $\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{1}_N \bar{\mathbf{x}}'$ .

They are closely related:

$$\mathbf{Z}_j = \mathbf{X}\mathbf{r}_j = \mathbf{1}_N \bar{\mathbf{x}}' \mathbf{r}_j + \tilde{\mathbf{X}}\mathbf{r}_j = \bar{z}_j \mathbf{1}_N + \tilde{\mathbf{Z}}_j$$

This means that  $\mathbf{Z}_j - \tilde{\mathbf{Z}}_j = \bar{z}_j \mathbf{1}_N$  is a constant vector. Scatter plots of  $\mathbf{Z}_j$  vs  $\mathbf{Z}_k$  will look like plots of  $\tilde{\mathbf{Z}}_j$  vs  $\tilde{\mathbf{Z}}_k$ .

```
Cmd> z_a <- x %%% rightvecs # PC's computed from X
```

```
Cmd> z_b <- (x - xbar') %%% rightvecs # PC's computed from resid
```

Columns of `z_a` are  $\mathbf{Z}_j = \mathbf{X}\mathbf{r}_j$ .

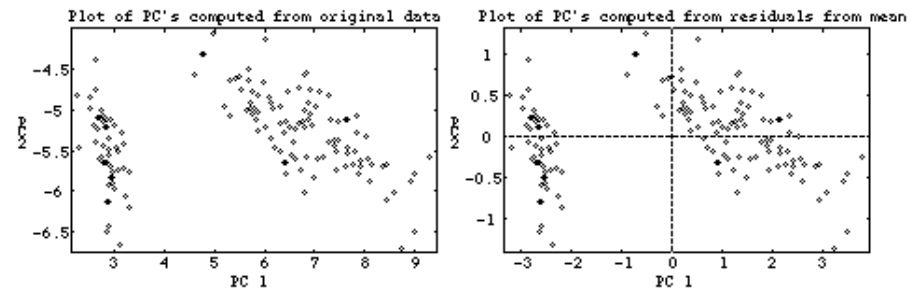
Columns of `z_b` are  $\tilde{\mathbf{Z}}_j = \tilde{\mathbf{X}}\mathbf{r}_j$ .

```
Cmd> head(z_a - z_b,3) #difference is constant
```

	(1)	(2)	(3)	(4)
(1)	5.5024	-5.327	0.63185	0.033352
(2)	5.5024	-5.327	0.63185	0.033352
(3)	5.5024	-5.327	0.63185	0.033352

```
Cmd> plot(z_a[,1], z_a[,2],xlab="PC 1",ylab="PC2", \
title:"Plot of PC's computed from original data",symbols:"\11")
```

```
Cmd> plot(z_b[,1], z_b[,2],xlab="PC 1",ylab="PC2", \
title:"Plot of PC's computed from residuals from mean", \
symbols:"\11")
```



Here is another way to view the best rank  $m$  approximation  $\tilde{\mathbf{X}}^{(m)} = \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}'_j$ :

$$\begin{aligned} \mathbf{X} &= \mathbf{1}_N \bar{\mathbf{x}}' + \tilde{\mathbf{X}} = (\text{mean} + \text{residuals}) \\ &= \mathbf{1}_N \bar{\mathbf{x}}' + \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}'_j + (\sum_{m+1 \leq j \leq p} \tilde{\mathbf{z}}_j \mathbf{r}'_j) \\ &= \mathbf{1}_N \bar{\mathbf{x}}' + \tilde{\mathbf{X}}^{(m)} + \mathbf{e}, \end{aligned}$$

where  $\mathbf{e} \equiv \tilde{\mathbf{X}} - \sum_{1 \leq j \leq m} \tilde{\mathbf{z}}_j \mathbf{r}'_j = \sum_{m+1 \leq j \leq p} \tilde{\mathbf{z}}_j \mathbf{r}'_j$  is the "error" made when you approximate  $\tilde{\mathbf{X}}$  by  $\tilde{\mathbf{X}}^{(m)}$ .

This decomposes the data into

- a part coming from the *means* in  $\bar{\mathbf{x}}$
- the *best rank  $m$  approximation*  $\tilde{\mathbf{X}}^{(m)}$  to the residuals from  $\bar{\mathbf{x}}$
- an "error"  $\mathbf{e} = \tilde{\mathbf{X}} - \tilde{\mathbf{X}}^{(m)}$ , the residuals from the approximation.

In the more complex linear model case

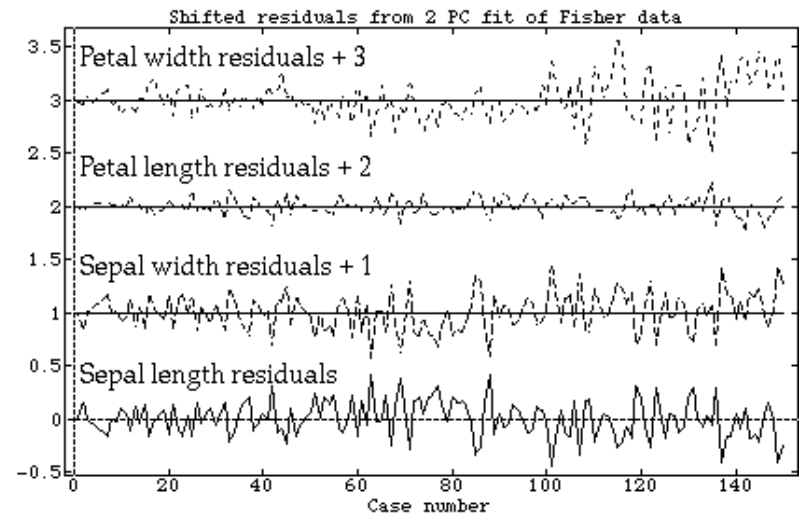
$$\mathbf{X} = \mathbf{U}\hat{\mathbf{B}} + \tilde{\mathbf{X}}^{(m)} + \mathbf{e}$$

where  $\mathbf{e} = \tilde{\mathbf{X}} - \tilde{\mathbf{X}}^{(m)}$ ,  $\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{U}\hat{\mathbf{B}}$ .

What might the elements of  $\mathbf{e}$  look like?

Computation of  $\mathbf{e}$  for Fisher data.

```
Cmd> e <- x - xhat_2 # Fisher residuals, 150 by 4 matrix
Cmd> sum(e) # column sums and hence means are 0
      SepLen      SepWid      PetLen      PetWid
(1)  5.7732e-14 -6.2617e-14 -3.1308e-14  4.8628e-14
Cmd> sum(describe(e,var:T))/sum(describe(x,var:T))
(1)  0.022315 Like 1 - R^2
Cmd> sum(svals[-run(2)]^2)/sum(svals^2) # trailing sing vals
(1)  0.022315
Cmd> lineplot(1,e + run(0,3)',xlab:"Case number",\
             title:"Shifted residuals from 2 PC fit of Fisher data",show:F)
Cmd> addlines(vector(0,150,?,0,150,?,0,150),\
             vector(1,1,?,2,2,?,3,3)) # add 0 lines
```

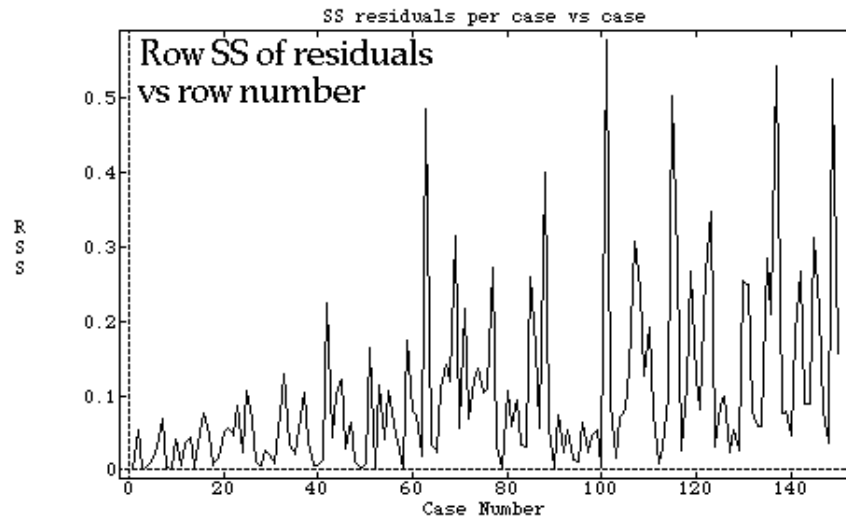


It looks cases 1-50 may have slightly smaller residuals than cases 51-150

The row sums of squares  $\sum_{1 \leq l \leq p} e_{il}^2$ ,  $i = 1, \dots, N$  measure the lack of fit to each case:

```
Cmd> rowss <- vector(sum(e'^2)) # Note the transpose
```

```
Cmd> lineplot(1,rowss,xlab:"Case Number",ylab:"RSS",\
             title:"SS residuals per case vs case")
```



For reasons that might be explored further, the rank 2 fit seems to do better for variety 1 (cases 1-50) than for the other two varieties as appeared might be the case from the earlier plot.