More about basic matrix operations for L X and L X D, etc.
& Illustrations of eigen & svd operations for small real data set (Exercise below, p. 3)

We can begin w/ an arbitrary data matrix X of order n x p. We shall illustrate in class for a small matrix; but also see the later pages of this handout. Recall, **L = I – (1/n) 11'**
If we choose the entries in a diagonal matrix D (Dn below) to be **reciprocals of diagonal entries** of the product X'LX, we first get **Dn** (as a row vector):

```
>Dn=diag(t(X) %*% L %*% X)
 Then redefine Dn to get:
>Dn=diag(1/sqrt(Dn))  #So in effect we have  diag(diag(X'L X))

 Dn becomes a diagonal (above) that contains reciprocals of the
diagonals of the product X' L X.
Now compute Z as:
>Z = L %*% Xa %*% Dn
```

#Try function scale( ) in R to get Z (almost) Try it...(you will need to multiply by a constant)
We shall examine these column VECTORS of Z geometrically in due course; but now we simply note that Z here has entries such that column sums of squares equal UNITY or one; further, the inner products of columns of Z are cosines of angles between unit length vectors, and these same cosines are also correlation coefficients for the various pairs of variables. **The product of Z' times Z will have ones in its diagonal, viz., should be correlation matrix. Write as** `R = t(Z) %*% Z`

```
# where we now use two different arguments, X and Z:
>cor(X); cor(Z)  #separating with semicolon gives both results
   You should find these results are the same.
```

>#Finally, we generate the **inverse of this R** (could do same for `var(X)` above, but will skip that here) using function `solve`.

```
>R.inv = solve(R)
```

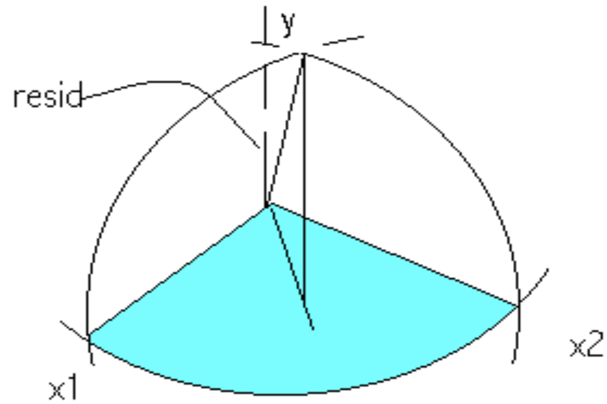
`R.inv` is a matrix with diagonals exceeding unity (the universal lower bound of the inverse of the correlation matrix diagonals is 1). To get a useful summary statistic, compute $R^2_{1.23} = 1 - 1/1.126$ (first entry). The statistic $R^2_{1.23}$ is the ***squared multiple correlation***, predicting variable 1 from ***optimal linear combination*** of 2 & 3. See the figure on the next page.

We can easily generalize this idea, getting each ***squared multiple correlation when predicting any column from all other columns:***
D.smc = diag(1 - diag(1/diag(R.inv))) [where R 'knows' that 1 means 'as many 1's as needed' for this R.inv operation]

For details about the geometry of regression, and then inverses, consider the vectors y, x1 and x2 on the next page just as if they were columns of the matrix I've computed as L X Dn above. I've concentrated on predicting y from x1 and x2, all vectors of unit (or equal) length. But then note that x1 could be predicted (using LS) from y and x2; also, x2 could be predicted from y and x1. Consideration of these last points leads to a sound geometric interpretation of INVERSES. I shall get to that in class.

Geometry for least squares regression: Two predictors

In principle, any least squares (LS) regression entails the type of projection seen in this figure, notably so when there are just two predictors.



Projecting vector y onto space of x1, x2; the perpendicular projection corresponds to y-hat, a linear combination of the two vectors x1 and x2. 'resid' is orthog. to o,x1,x2 space

When the LS criterion is used to find regression coefficients this assures that the linear combination of the ***x***'s, on ***x1-O-x2*** plane, and called $\hat{y}$ ***,*** is the *perpendicular projection* of ***y*** onto the space of the ***x***'s. ***Such a*** $\hat{y}$ ***is always in the space of the x's, and the corresponding residual vector is always orthogonal to the space of the x's.*** When ***x***'s are mutually orthogonal then interpretations (and computations) become especially straightforward because the regression coefficients for the ***y***-variable on any ***x***-variable are the same as when all coefficients are computed in so-called 'multiple regression.' When ***x***'s, i.e., the predictors, correspond to orthogonal (mutually uncorrelated) variables then interpretations are simplest of all, no matter how many ***x***'s there may be (remember the use of orthogonal contrasts in ANOVA).

When ***x***'s are mutually correlated, the *projection picture* does not change, nor do the facts in italics above; but computations (and interpretations) of regression coefficients do change, often radically. When ***x***'s are correlated, then each regression coefficient is explicitly a ***multiple (partial)*** regression coefficient. These regression coefficients are closely related to the cosines of the dihedral angles between planes in the figure; details in class. Each such coefficient may even have a different sign than does the correlation of the corresponding ***x*** with ***y***; and its magnitude, even for a 'standardized' regression coefficient, can range from zero to +/- infinity. (***Standardized regression coefficients*** *result when all variables,* ***y*** *and all* ***x****'s are scaled to have zero means and unit variances (or s.d.s)*; they can be helpful in some situations, and be sure in applications that you always consider the *metric* in use, either 'raw' score, or its standardized counterpart.)

Note that ***partial correlations*** can also be interpreted in the context of this figure. For example, the partial correlation between *y* and *x1* is the *cosine* of the ***dihedral angle*** *between the* ***planes*** *y-O-x1* and *x1-O-x2*. This is the partial correlation between *y* and *x2*; etc.

*(Remember, these matrices (above) can have any order (that can be stored in memory) for **X** of order n x p, except we generally will want n > p; that is the number of rows should exceed the number of columns. Recall that this is a necessary but not sufficient condition for the regular inverse of either the variance-covariance matrix or the correlation matrix to exist.)*

***Exercise**: Do all operations I do below on your machine using your own small matrix X (with real data)[see below], ensuring that you understand what you've done. Ask questions. Then calculate the **inverse** of your matrix R, and continue, using a relatively small data set (possibly a sample [subset] w/ only few rows [but with n > p]), to **conduct an eigen analysis and singular value decomposition**. You may use* **`eigen`** *directly on the matrix R, and* **`svd`** *for the matrix Z for your X. **Be sure to include graphics (say using pairs and more) and interpretations of all results.** Use my interpretations above to guide you, but be sure to take account of details in your explanations! Two possible data sets to consider are from are in the MASS library: painters and UScereal. (painters has fewer columns, but it would be fine to select a few columns from the cereal data.) The 'School' variable in the painters data is categorical, however, so it would have to be eliminated, or modified. For the UScereal data, the first and last variables are categorical. Note that you might also consider a thorough principal component analysis following your reading (web) on this.*

See my summary function at end; also, some algebra. (**`trees`** is in MASS library) Consider the trees data (see ?trees); but for our purposes choose only five rows:

```
> trees[c(27,5,29,12,15),]   #selected rows; e.g. 5th row of trees is second row here
   Girth Height Volume
27  17.5     82   55.7    #it would be good for you to plot using pairs
5   10.7     81   18.8     Let us call this matrix treess below.
29  18.0     80   51.5
12  11.4     76   21.0
15  12.0     75   19.1
>my.summary(trees[c(27,5,29,12,15),])
       Girth Height Volume
means  13.92  78.80  33.22
s.d.s   3.53   3.11  18.68
skewns  0.26  -0.21   0.30
krtsis -2.22  -2.15  -2.23  low and high values omitted...

The Z for this X is:
Z.trees=scale(trees[c(27,5,29,12,15),])/2 # Be sure you see why
> Z.trees                                   I divided by 2 here
    Girth Height Volume
27  0.507  0.514  0.602    #sums of squares of columns are one
5  -0.456  0.353 -0.386    which means lengths of col. vectors = 1
29  0.578  0.193  0.489
12 -0.357 -0.450 -0.327
15 -0.272 -0.610 -0.378
>t(Z.trees) %*% Z.trees  #which as you see gives the correlations R
       Girth Height Volume
Girth  1.000  0.537  0.983
Height 0.537  1.000  0.645
Volume 0.983  0.645  1.000

Next, we generate eigenvalues and vectors for this R (and svd of Z)
>sapply(eigen(cor(treess)),rnd3)   I might write R = V Dλ² V′
$values              # Note: rnd3 = function(x)round(x,3)
[1] 2.462  .530  .008  #← eigenvalues of matrix R (Dλ²-vector)
```

```
  The sum of these 3 values = 3 (see my notes on matrix algebra)
$vectors   #eigenvectors of same matrix R
       [,1]   [,2]   [,3]  #col SSqs all equal 1.00
[1,]  -.603   .440   .666   also columns are mutually orthogonal
[2,]  -.497  -.860   .119   so V'V = I (also, VV'=I)
[3,]  -.624   .259  -.737
    NB: rnd3 = function(x)round(x,3)
sapply(svd(Z.trees),rnd3)
$d
[1] 1.569  .728  .089   #square these to get D_λ^2 values!
$u
       [,1]   [,2]   [,3]
[1,]  -.597   .086   .506
[2,]   .217   .830  -.253
[3,]  -.478  -.296  -.533
[4,]   .410  -.199   .561
[5,]   .448  -.421  -.285  This matrix U is like V: U'U=I
  (These three derived variables are called principal components)
 First column contains MOST of the information … we shall discuss
$v
       [,1]   [,2]   [,3]  #COMPARE cols w/ those for R above!
[1,]  -.603  -.440  -.666   (signs are reversed for one column)
[2,]  -.497   .860  -.118   such reversals always a possibility)
[3,]  -.624  -.259   .737
```

Also, you can find good discussions of principal component analysis (PCA) on the web or in books on multivariate analysis. If you find what you think is a good one, put the reference on our wiki!

>?painters ***Description:***

The subjective assessment, on a 0 to 20 integer scale, of 54 classical painters. The painters were assessed on four characteristics: composition, drawing, colour and expression. The data is due to the Eighteenth century art critic, de Piles. [More information at ?painters] More on these data next week.

```
painters[1:7,]   [[rank 2 works reasonably well...]]
                Composition Drawing Colour Expression School
Da Udine                 10       8     16          3      A
Da Vinci                 15      16      4         14      A
Del Piombo                8      13     16          7      A
Del Sarto                12      16      9          8      A
Fr. Penni                 0      15      8          0      A
Guilio Romano            15      16      4         14      A
Michelangelo              8      17      4          8      A
```

Note that a sample of say six rows could be obtained as:

```
Paint.subs = painters[sample(1:54,6,repl=F),-5] #-5 says skip School
    You should be able to give the dimensions (dim( )) of Paint.subs, etc.

  ----------- Copy the functions below for next week:


my.summary = function(xxx,dig=2)  #can change decimal accuracy w/ dig
{#generate means/s.d.s/skewness's & kurtosis for each column of xxx
xxx <- as.matrix(xxx)
xm <- apply(xxx, 2, mean)
```

```
s.d <- sqrt(apply(xxx, 2, var))
xs <- scale(xxx)
sk <- apply(xs^3, 2, mean)
kr <- apply(xs^4, 2, mean) - 3
rg <- apply(xxx, 2, range)
sumry <- round(rbind(xm, s.d, sk, kr, rg), 3)
dimnames(sumry)[1] <- list(c("means", "s.d.s", "skewns", "krtsis", "low",
"high"))
sumry <- round(sumry, dig)
sumry  }

#- -----------and looking ahead, I give you a function ifa for common
  #                factor analysis; for next week, if only briefly!
ifa = function(rr,mm,scrp=T)  {
# routine is based on image factor analysis; for exploratory factor analysis;
#generates an unrotated common factor coefficients matrix & scree plot.
# In R (v.2.0.1), follow w/, say, promax(  ) or varimax( ) where
# parentheses contains result$fac if 'ifa' produced object 'result'
# In Splus (v.6.2) follow w/ rotate; e.g. rotate( ), same as above, but
# second argument in rotate could be 'varimax' or 'promax' or, ‘oblimin'
# rr is taken to be symmetric matrix of correlations or covariances;
# mm is no. of factors.
# NB: this is routine that appears in my recent (2005, spring) article
# Factor analysis: Exploratory; Wiley Encyclopedia of Behavioral
 # Statistics For additional functions or assistance, contact:
#  rpruzek@uamail.albany.edu
#rr<-matrix(rr)
rinv <- solve(rr) #takes inverse of rr; so rr must be non-singular
sm2i <- diag(rinv)
smrt <- sqrt(sm2i)# smrt a vector here
dsmrt <- diag(smrt)
rsr <- dsmrt %*% rr %*% dsmrt
reig <- eigen(rsr, sym = T)
vlamd <- reig$va
vlamdm <- vlamd[1:mm]
qqm <- as.matrix(reig$ve[, 1:mm])
theta <- mean(vlamd[(mm + 1):nrow(qqm)])
dg <- sqrt(vlamdm - theta)
if(mm == 1)fac <- dg[1] * diag(1/smrt) %*% qqm
else fac <- diag(1/smrt) %*% qqm %*% diag(dg)
if(scrp){plot(1:nrow(rr), vlamd, type = "o", ylab='Eigenvalues of DRD')
abline(h = theta, lty = 3)
title("Scree plot for IFA")}
fac<-round(fac,2) #sets two decimal digits in output
rownames(fac)<-list(rownames(rr)[1:nrow(rr)])[[1]]
list(vlamd = vlamd, theta = theta, fac = fac)  }
```