**Factor Analysis (FA)**

**What is FA?**

FA has the same objectives as PCA:

1. Discover the “true dimension” of the data
2. Try to interpret “new” variables

However, the way FA goes about achieving these objectives is different than PCA. This can lead to more easily to interpret “new” variables, but this interpretation can come with some side effects (multiple ways to interpret, violation of underlying assumptions). Also, another difference between the two is that PCA was focused on explaining the variance structure of the data. FA is concerned with explaining the variance and covariance structure of data.

The classical way that FA is first explained is as follows.

Suppose test scores for a student can be modeled by the following equations:

Classics = λ1f + η1

French = λ2f + η2

Music = λ6f + η6

where

Classics, French, … , Music are original

(measurable) variables

f = random component **common** to all original

variables (not measurable)

ηj = random component **specific** for the jth

original variable (not measurable)

λj = **loading** on f specific to each original variable

(not eigenvalues!)

In summary,

* One may think that a student’s performance on a particular test was the sum of an overall intelligence component and a course specific component
* Note that f and ηj are not directly measurable
* f is a random variable measuring overall intelligence
* λj is a weight measuring the role overall intelligence has on the individual test score
* ηj is a random variable specific for a course which helps to account for items such as students may want to study for the classics more than French
* Given λjf + ηj, we can find the test score through the model
* Notice there is only one f – thus we are using only one random variable across all p test scores. This f is called a *factor*.
* The model is called a “factor analysis model”.

**FA model**

In regression analysis, we have models of the form

Yi = β0 + β1Xi1 + β2Xi2 + … + βpXip + εi

where εi ~ independent N(0,σ2). A similar type of model can be formed for FA using the factors (new variables) as the independent variables and the original variables as the dependent variable.

Let **x** ~ (**μ**, **Σ**) where **x** is p×1. Notice that we do not use a multivariate normal distribution assumption. The FA model assumes there are m “underlying factors,” denoted by f1,…,fm with m < p, for each of the p original variables. The model is:

x1 = μ1 + λ11f1 + λ12f2 + … + λ1mfm + η1

x2 = μ2 + λ21f1 + λ22f2 + … + λ2mfm + η2

xp = μp + λp1f1 + λp2f2 + … + λpmfm + ηp

where

* + - 1. xj is the jth random variable
      2. fk’s i.i.d. (0,1) for k = 1, …, m and are called *common factors* because they are part of each original variable. These are unobservable random variables (the “equivalent” in regression are observable). These factors are uncorrelated with each other.
      3. ηj’s ~ independent (0,ψj) for j = 1, …, p and are called *specific factors* because they are potentially different for each original variable.These are unobservable random variables.
      4. ψj is the *specific variance* of ηj
      5. fk and ηj are independent for all k = 1, …, m and j = 1, …, p
      6. λjk measures the contribution of the kth common factor to the jth original variable. These are called *factor loadings*. They will help us interpret the common factors! The λ’s are NOT eigenvalues. I chose to use λ’s because they are coefficients in a linear combination like we had in PCA.

In summary,

* We assume that the x1, …, xp come about through the FA model structure (p. FA.3).
* Each xj variable is made up of factors common to all – fk’s for k = 1, …, m different factors.
* Each xj variable has a factor that is specific for all – ηj for j = 1, …, p.
* Examine the factor loadings to determine the importance of a common factor to a particular independent variable.

Notes:

1. MAKE SURE that you understand ALL of the statements above!
2. i.i.d. stands for “independently and identically distributed”.
3. Ideally, we would like the common factors to account for as much information about the original variables as possible. Thus, ideally the specific factors would have a very small variance.
4. The fk and ηj zero mean assumption can be made without loss of generality (WLOG). Why? The answer will be given shortly.
5. The fk variance 1 assumption can be made WLOG (λj’s could be changed to create a new set of fk’s that have a variance 1).
6. The main assumption that we need to be concerned about is the fk and ηj are independent.
7. Most often, μj is assumed to be 0. This can be accounted for by simply subtracting the means from the xj’s. Therefore, we could use the following FA model:

 = λj1f1 + λj2f2 + … + λjmfm + ηj for j = 1, …, p

where the ’s are “mean adjusted.” The above model can be written in a more compact form using matrices:

****

where

* + ****
  + **f** = [f1, f2,…, fm]′ ~ (**0**, **I**) where **I** is the identity matrix
  + **η** = [η1, η2, …, ηp]′ ~ (**0**, **ψ**) where  = Diag(ψ1,…,ψp)
  +  is called the *factor loading matrix*
  + **f** and **η** are independent

1. Similar to PCA, we more often work with standardized data so that we also have a variance of 1 for the random variable on the left hand side of the equation. Thus, we could write a model as

zj = λj1f1 + λj2f2 + … + λjmfm + ηj for j = 1, …, p

instead. In matrix form, this becomes

****

where **z** = [z1, z2,…, zp]′ and Cov(**z**) = **P**. Note that the λjk’s will not be the same between using  or zj. I simply use the same notation for the factor loadings because otherwise the notation will get messier later ☹.

**Covariance and correlation matrices**

To understand this section, you need to know the following basic results of working with random vectors:

1. Let **A** be a matrix of constants, and let **y** be a vector of random variables. Then Cov(**Ay**) = **A**Cov(**y**)**A**′. This can be shown using standard properties of covariances and variances:

Var(ay1+by2) = a2Var(y1) + b2Var(y2) + 2abCov(y1,y2)

for constants a and b and random variables y1 and y2, see my Introduction to Mathematical Statistics notes.

1. Suppose **x** and **y** are independent random vectors. Then Cov(**x** + **y**) = Cov(**x**) + Cov(**y**).

From the matrix form of the FA model,

**Σ** = Cov(**x**)

= Cov(**Λf** + **η**)

= Cov(**Λf**) + Cov(**η**) because **f** and **η** are

independent

= **Λ**Cov(**f**)**Λ**′ + Cov(**η**)

= **ΛIΛ**′ + **ψ** because **f** ~ (**0**, **I**) and **η** ~ (**0**, **ψ**)

= **ΛΛ**′ + **ψ**

Instead of trying to find if **Λ**, **f**, and **η** exist such that   
**** = **Λf** + **η**, we try to find a **Λ** and **ψ** suchthat **Σ** = **ΛΛ**′ + **ψ**. These are often called the *factor analysis equations*.

Notes:

1. You may be wondering if there is a typo by specifying **Σ** = Cov(**x**) rather than **Σ** = Cov(****). There is not a typo because Cov(**x**) = Cov(****). Why? For the same reason the variance of a variable is the same as when a constant is added to it. This is often discussed in an introductory statistics course.
2. With **Σ** = **ΛΛ**′ + **ψ**, the common factors (fk’s) explain the covariances between the original variables exactly because **ψ** is a diagonal matrix. This can be seen from the matrices below.

**Σ** = **ΛΛ**′ + **ψ**

⇔

⇔

⇔

1. The above leads to seeing that Var(xj) = σjj can be written as  and Cov(xj, xj′) = .
2. The proportion of variance for xj explained by common factors is /σjj. The numerator in the proportion is called the *communality* of the jth original variable because it is contributed by the common factors.
3. Therefore, Var(xj) =  = communality + specific variance.
4. The specific variance is sometimes called the *uniqueness*.
5. Cov(xj, fk) = λjk

Cov(xj, fk)

= Cov(λj1f1 + … + λjmfm + ηj, fk)

= Cov(λj1f1, fk) + … + Cov(λjkfk , fk) + … +

Cov(λjmfm , fk) + Cov(ηj, fk)

= 0 +… + 0 + λjkVar(fk) + 0 +… + 0 + 0

= λjk.

Please remember that **P** is also the covariance matrix of the standardized data. This implies that **Λ** is a matrix of correlations between the zj’s (standardized data) and the fk’s. Then

1. Corr(zj, fk) = λjk (notice this also means that -1 ≤ λjk ≤ 1 due to the numerical range of correlations); using our previous notation, we could represent Corr(zj, fk) as  as well.
2. =1 because the diagonal elements of a correlation matrix are 1
3. The communality of the jth standardized variable is 

Make sure you can show these statements above by writing out **P** = **ΛΛ**′ + **ψ**!

**Solving the FA equations**

To determine if an actual set of m common factors exist, we need to determine if there exists a **Λ** and **ψ** such that   
**P** = **ΛΛ**′ + **ψ**.

Notes:

1. Suppose **P** is known. Then there are p(p+1)/2 known quantities in the correlation matrix (try p = 2, 3, … to see this).
2. The number of unknown quantities in **ΛΛ**′ is pm because this is the number of λij’s that exist (see below).

zj = λj1f1 + λj2f2 + … + λjmfm + ηj for j=1,…,p

1. The number of unknown quantities in **ψ** is p because ψ1, …, ψp on the diagonal of the matrix.

We can form p(p+1)/2 equations with mp + p = p(m+1) unknowns. Thus, more than one solution may exist. For example, suppose p = 4 and m = 2. Then we have 10 equations with 12 unknowns.

Below is **P** = **ΛΛ**′ + **ψ** written out:



Example: Possible problems with p = 3 and m = 1

There are 6 equations and 6 unknowns. The **P** = **ΛΛ**′ + **ψ** equation written out is



Then 1 = , 1 = , 1 = ,

ρ12 = λ11λ21, ρ13 = λ11λ31, and ρ23 = λ21λ31. One can show then that



Suppose that



Because ρ13 < 0, a solution cannot be found because



Suppose instead that



This leads to . Because 1 = , this means that ψ1 < 0. However, ψ1 denotes a variance and a variance can not be negative!

**FA implementation**

There are a number of ways for finding solutions to the FA equations. The most often used procedure is maximum likelihood estimation. Please see my separate set of notes on what maximum likelihood estimation is in the context of a simpler problem. Using maximum likelihood estimation allows one to use many tools available for maximum likelihood in general. Some of these will be discussed when we examine how to choose an appropriate number of factors.

Let **x**1,…,**x**N be a random sample from a Np(**μ**,**Σ**). Notice this is the first time that I have made a specific distributional assumption! The likelihood function is



The FA equations structure of **Σ** = **ΛΛ**′ + **ψ** for the covariance matrix leads to p(m+1) parameter estimates that need to be found. The MLEs are found through iterative numerical methods. When the estimates change very, very little at successive iterations, the estimates are said to “converge” to the MLEs. The corresponding estimates of **Λ** and **ψ** are denoted symbolically as . Of course, individual values inside of the matrices will have ^’s on them as well. Details regarding the estimation methods are excluded here, but are available in multivariate textbooks.

Although the above is presented for **Σ**, **P** can also be used with standardized random variables and then noting that **P** = **ΛΛ**′ + **ψ**.

The main function used in R to estimate these models is factanal(). This function automatically uses standardized data and maximum likelihood estimation.

Example: Goblet data (GobletFA.R, goblet.csv)

We will discover later that two common factors are o.k. with this data. I will use this result here to illustrate how to estimate the FA model.

> goblet <- read.csv("C:\\chris\\goblet.csv")

> head(goblet)

goblet x1 x2 x3 x4 x5 x6

1 1 13 21 23 14 7 8

2 2 14 14 24 19 5 9

3 3 19 23 24 20 6 12

4 4 17 18 16 16 11 8

5 5 19 20 16 16 10 7

6 6 12 20 24 17 6 9

> goblet2 <- data.frame(ID = goblet$goblet,

w1 = goblet$x1/goblet$x3,

w2 = goblet$x2/goblet$x3,

w4 = goblet$x4/goblet$x3,

w5 = goblet$x5/goblet$x3,

w6 = goblet$x6/goblet$x3)

> mod.fit2 <- factanal(x = ~ w1 + w2 + w4 + w5 + w6, factors

= 2, data = goblet2, rotation = "none")

> names(mod.fit2)

[1] "converged" "loadings" "uniquenesses"

[4] "correlation" "criteria" "factors"

[7] "dof" "method" "STATISTIC"

[10] "PVAL" "n.obs" "call"

> class(mod.fit2)

[1] "factanal"

> methods(class = factanal)

[1] print

see '?methods' for accessing help and source code

Notice that there is only one method function for the corresponding class produced by factanal()! For example, below is what happens when we use the usual summary() function with the results from the function:

> summary(mod.fit2) #Not useful - no summary.factanal()

Length Class Mode

converged 1 -none- logical

loadings 10 loadings numeric

uniquenesses 5 -none- numeric

correlation 25 -none- numeric

criteria 3 -none- numeric

factors 1 -none- numeric

dof 1 -none- numeric

method 1 -none- character

scores 50 -none- numeric

STATISTIC 1 -none- numeric

PVAL 1 -none- numeric

n.obs 1 -none- numeric

call 6 -none- call

Whenever an object is simply given, like mod.fit2, at a command prompt, a print() function is always used to print its contents (even if you do not actually see print()). Sometimes, there are specific method functions designed to control what is printed. In this case, the method function print.factanal() is for this purpose. Thus, to obtain a summary of what is inside of mod.fit2, we can simply execute mod.fit2 at a command prompt or use print(mod.fit2) at a command prompt. The one advantage of using the print() function is a cutoff = 0.0 argument can be included so that all of the loadings are printed.

> print(x = mod.fit2, cutoff = 0.0)

Call:

factanal(x = ~w1 + w2 + w4 + w5 + w6, factors = 2, data = goblet2, rotation = "none")

Uniquenesses:

w1 w2 w4 w5 w6

0.160 0.106 0.005 0.308 0.506

Loadings:

Factor1 Factor2

w1 0.555 0.730

w2 0.494 0.806

w4 0.997 -0.034

w5 0.733 0.394

w6 0.593 -0.378

Factor1 Factor2

SS loadings 2.434 1.481

Proportion Var 0.487 0.296

Cumulative Var 0.487 0.783

Test of the hypothesis that 2 factors are sufficient.

The chi square statistic is 4.74 on 1 degree of freedom.

The p-value is 0.0294

Notes:

1. The rotation = "none" argument needs to be included in factanal(). We will discuss why later when we examine “rotations”.
2. The “Uniquenesses” table provide the estimates of the specific variances. For example,  = 0.160.
3.  is given in the table labeled “Loadings”. Therefore, **z** = **f** + **η** is



What is your interpretation of the factors? Note this will not be the FINAL solution that we will use. Later, we will discuss how to find a more interpretable solution.

1. Remember that Corr(zj, fk) = λjk so these factor loadings give information about the estimated correlation between the standardized variables and the common factors.
2. The last table in the output gives information about the amount of variability accounted for by the common factors. Unlike PCA, we are not trying to maximize the amount of variability that our “new” variables account for with respect to the old variables. Thus, this information is not as important as it was with PCA. The table does show that 78.3% of the total variation is accounted for by two factors. With respect to our model, this is found by first summing the squared estimated common factor loadings:



Second, note that the total variation in the original variables is 5  because we are working with standardized variables. Thus, the total proportion of variation is 3.91/5 = 0.781. I recommend against looking at the cumulative variation prior to the last factor. The reason is because if less than m factors were actually used, these cumulative variations will not necessarily be the same as with m factors (e.g., estimating a one factor model here gives 0.524 for this first factor).

1. Convergence is not guaranteed. Warning messages will be printed when there are problems. Also, the converged and criteria component of the model fit object provides some information too.

The correlation component of mod.fit2 provides the estimated correlation matrix for the standardized values:

> mod.fit2$correlation

w1 w2 w4 w5 w6

w1 1.00000000 0.86583150 0.5289460 0.6583664 0.02200922

w2 0.86583150 1.00000000 0.4650182 0.6897947 0.03009547

w4 0.52894596 0.46501816 1.0000000 0.7182324 0.60473700

w5 0.65836637 0.68979469 0.7182324 1.0000000 0.15303061

w6 0.02200922 0.03009547 0.6047370 0.1530306 1.00000000

> Z <- goblet2[,-1]

> cor(Z)

w1 w2 w4 w5 w6

w1 1.00000000 0.86583150 0.5289460 0.6583664 0.02200922

w2 0.86583150 1.00000000 0.4650182 0.6897947 0.03009547

w4 0.52894596 0.46501816 1.0000000 0.7182324 0.60473700

w5 0.65836637 0.68979469 0.7182324 1.0000000 0.15303061

w6 0.02200922 0.03009547 0.6047370 0.1530306 1.00000000

The estimate of **ΛΛ**′ + **ψ** from the model:

> mod.fit2$loadings[,]%\*%t(mod.fit2$loadings[,]) +

diag(mod.fit2$uniqueness)

w1 w2 w4 w5 w6

w1 1.00000012 0.86232638 0.5284160 0.6943593 0.05295469

w2 0.86232638 1.00000010 0.4654412 0.6800965 -0.01180489

w4 0.52841603 0.46544125 1.0000201 0.7170565 0.60359994

w5 0.69435929 0.68009647 0.7170565 1.0000003 0.28504424

w6 0.05295469 -0.01180489 0.6035999 0.2850442 0.99999902

The [,] part of mod.fit2$loadings[,] was necessary to obtain only the factor loadings matrix because mod.fit2$loadings also provides some additional information within it (run it to see!). As an example of the computation above, the (1,1) component of the above matrix is



Also, for all diagonal elements,

> rowSums(mod.fit2$loadings[,]^2) + mod.fit2$uniqueness

w1 w2 w4 w5 w6

1.000000 1.000000 1.000020 1.000000 0.999999

A way to assess how good the common factors are in accounting for the information in the data is to examine the difference between the standard estimate of the correlation matrix and the estimate obtained from the model structure: . These are like residuals. Below are the results form R,

> round(mod.fit2$correlation – (mod.fit2$loadings[,] %\*%

t(mod.fit2$loadings[,]) + diag(mod.fit2$uniqueness)),

4)

w1 w2 w4 w5 w6

w1 0.0000 0.0035 0.0005 -0.0360 -0.0309

w2 0.0035 0.0000 -0.0004 0.0097 0.0419

w4 0.0005 -0.0004 0.0000 0.0012 0.0011

w5 -0.0360 0.0097 0.0012 0.0000 -0.1320

w6 -0.0309 0.0419 0.0011 -0.1320 0.0000

Overall, we see only one correlation (between w5 and w6) that is off by more than a little bit. This is a good sign that our model is generally accounting for the correlation between variables.

For a larger p, it may be difficult to look at all of the “residuals” as we do here. In this case, it may better to examine overall summaries of the residuals for each original variable:

> resid2 <- mod.fit2$correlation –

(mod.fit2$loadings[,]%\*%t(mod.fit2$loadings[,]) +

diag(mod.fit2$uniqueness))

> abs(resid2)>0.1

w1 w2 w4 w5 w6

w1 FALSE FALSE FALSE FALSE FALSE

w2 FALSE FALSE FALSE FALSE FALSE

w4 FALSE FALSE FALSE FALSE FALSE

w5 FALSE FALSE FALSE FALSE TRUE

w6 FALSE FALSE FALSE TRUE FALSE

> sum(abs(resid2)>0.1)

[1] 2

> colMeans(abs(resid2))

w1 w2 w4 w5

0.0141947124 0.0111053780 0.0006572041 0.0357761950

w6

0.0411994995

**Choosing an appropriate number of common factors**

We need to decide on an initial guess for m (number of common factors) before solving the factor equations. Performing a PCA and finding the number of principal components is often done to determine the initial m.

When making final decisions for a number of common factors, here are a few guidelines:

* Do not include trivial common factors; i.e., factors that are accounting for only one original variable.
* Use tools available through maximum likelihood methods

Likelihood ratio test (LRT)

Please see the additional notes for an introduction to likelihood ratio tests.

The hypotheses of interest are

H0:m common factors are sufficient (i.e., **P** = **ΛΛ**′ + **ψ** where **Λ** is p×m)

Ha:More common factors are needed

Detailed derivation of the statistic is available in Johnson and Wichern’s textbook. The transformed test statistic is

–2log() = 

For a large sample, a χ2 approximation can be used with this statistic to perform the test. Instead, a slightly modified version of the statistic is used to hopefully obtain a better approximation. Using what is known as a Bartlett correction, the modified statistic is



This statistic can be approximated by a  from a large sample. We can reject H0 if A is larger than the 1-α quantile from a  distribution.

Akaike’s information criterion (AIC)

The AIC statistic is

 + 2(degrees of freedom for model)

The best number of common factors to use correspond to the smallest AIC.

Unfortunat ely, there is not an implementation of finding the AIC in R for a FA model. One would need to program in the above equation into a function to find it.

The AIC and LRTs may suggest too many common factors are needed. In particular, one needs judge the difference between “statistical” significance and “practical” significance when applying the LRT.

Example: Goblet data (GobletFA.R, goblet.csv)

The LRT information is given at the bottom of the print(x = mod.fit2, cutoff = 0.0) output:

Test of the hypothesis that 2 factors are sufficient.

The chi square statistic is 4.74 on 1 degree of freedom.

The p-value is 0.0294

Also, the information can be extracted from the model fit object:

> mod.fit2$STATISTIC

objective

4.742428

> mod.fit2$dof

[1] 1

> mod.fit2$PVAL

objective

0.02942752

Thus, A = 4.74 and a  approximation is used with it. The p-value is 0.0294 indicating there is moderate evidence that more than two common factors are needed.

Below is what happens when I try one common factor:

> mod.fit1 <- factanal(x = ~ w1 + w2 + w4 + w5 + w6, factors

= 1, data = goblet2, rotation = "none")

> mod.fit1$PVAL

objective

6.623168e-05

> round(mod.fit1$correlation - (mod.fit1$loadings[,] %\*%

t(mod.fit1$loadings[,]) + diag(mod.fit1$uniqueness)),

4)

w1 w2 w4 w5 w6

w1 0.0000 0.0121 -0.0102 -0.0361 -0.0553

w2 0.0121 0.0000 -0.0738 -0.0042 -0.0471

w4 -0.0102 -0.0738 0.0000 0.2800 0.5560

w5 -0.0361 -0.0042 0.2800 0.0000 0.0902

w6 -0.0553 -0.0471 0.5560 0.0902 0.0000

As would be expected due to the m = 2 results, the p-value for the one common factor test is very small indicating more than one common factor is needed. For illustrative purposes, I also calculated  to examine the fit of this model. We see that the there are some much larger in absolute value “residuals” here than we saw when using two factors. Again, this is a sign the model with one factor is not sufficient.

Below is what happens when I try three common factors:

> mod.fit3 <- factanal(x = ~ w1 + w2 + w4 + w5 + w6, factors

= 3, data = goblet2, rotation = "none")

Error in factanal(x = ~w1 + w2 + w4 + w5 + w6, factors = 3, data = goblet2, :

3 factors are too many for 5 variables

> p <- 5

> m <- 3

> ((p-m)^2 - p - m)/2

[1] -2

The test involving three common factors can not be performed! There are too many factors given the number of original variables.

**Nonuniqueness of the common factors**

If m > 1, the factor loading matrix (**Λ**) is not unique!

To understand why, we will need to use a special type of matrix known as an *orthogonal matrix*. This matrix has its individual column vectors within the matrix orthogonal to each other and each with a length of 1. Thus, if **T** is an orthogonal matrix, **T**′**T** = **TT**′ = **I**.

Example: Let **x** ~ .

We have shown previously that the eigenvectors with length 1 from **Σ** are orthogonal to each other. If these eigenvectors are put as the columns of a matrix, the result is an orthogonal matrix:

**T** = 

Note that **T**′**T**=**I**.

Let **T** be an m×m orthogonal matrix. Notice that

**P** = **ΛΛ**′ + **ψ**

= **ΛTT**′**Λ**′ + **ψ** because **TT**′ = **I**

= (**ΛT**)(**ΛT)**′ + **ψ** because (**AB**)′ = **B**′**A**′ for two

matrices **A** and **B**

= **Λ**\*(**Λ**\*)′ + **ψ** where **Λ**\* = **ΛT**

Therefore, if **Λ** is a loading matrix, **ΛT** also is a loading matrix! There are an infinite number of orthogonal matrices.

Notice from the FA model that the same result can be seen:

**** = **Λf** + **η**

= **ΛTT**′**f** + **η**

=(**ΛT**)(**T**′**f**) + **η**

= **Λ**\***f**\* + **η**

where **Λ**\* = **ΛT** and **f**\* = **T**′**f**.

Multiplying **Λ** by an orthogonal matrix is called a *rotation*. A different **T** will lead to a different **Λ**\*. As with **Λ, Λ**\* allows one to interpret the common factors. Therefore, a researcher could perform rotations until an interpretable set of factor loadings are found! This is both a good and bad outcome!

When rotating, we try to find a **Λ**\* that allows us to more easily interpret the common factors. This usually means making the loadings close to 0 or 1 or -1. The reason is that if a factor loading is 0, then the common factor does not play a large part in forming an original variable. Similarly, if a loading is close to -1 or 1, the common factor plays a large part in forming an original variable.

Example: Goblet data (GobletFA.R, goblet.csv)

Plots of the common factor loadings are often used to help interpret the factors. Below is a plot of the initial common factor loadings:

> mod.fit2$loadings[,]

Factor1 Factor2

w1 0.5549549 0.72955841

w2 0.4943930 0.80591297

w4 0.9969260 -0.03403834

w5 0.7327331 0.39438303

w6 0.5925498 -0.37815166

> plot(x = mod.fit2$loadings[,1], y =

mod.fit2$loadings[,2], main = "Factor loadings before

rotation", xlim = c(-1,1), ylim = c(-1,1), xlab =

expression(hat(lambda)[i1]), ylab =

expression(hat(lambda)[i2]), type = "n", panel.first =

grid(lty = "dotted", col = "lightgray"))

> abline(h = 0)

> abline(v =0)

> text(x = mod.fit2$loadings[,1], y =

mod.fit2$loadings[,2], labels =

row.names(mod.fit2$loadings[,]))



To make the common factors easier to interpret, the loading axes can be rotated like

A graph with red lines and blue arrows

Description automatically generated

Notice that w1, w2, and w6 fall close to the axes. Values for w1 and w2 are close to 1 for the first factor and 0 for the second factor (i.e., , , , and ). The opposite is true for w6. This should help us then in our interpretation of the corresponding factors. Therefore, we would like to choose a **T** such that we obtain **Λ**\* = **ΛT** similar to what is shown in the above plot.



### Orthogonal rotation methods

There are many established ways to choose a **T**. We will discuss the one most often used called the *varimax* method. The GPArotation and psych packages provide other types of rotations.

Let  where **T** is an orthogonal matrix. Note that **B** plays the role of a new loading matrix of the form



Kaiser (1958) suggested to find a **T** that maximizes the following:



where bjq is the jth row and qth column element of **B**. For each column of **B**, the formula essentially finds the variance of the squared elements.

From a STAT 801-like course, remember that



is the “usual” biased sample variance.

These variances are summed over the m columns of **B** (m factors) to form V\*.

Remember that **Λ** is the matrix of the “initial” factor loadings. **B** is a matrix of the “rotated” factor loadings. Therefore, the varimax method tries to maximize the variance of these factor loadings. This forces the factor loadings to be as spread out as possible. Because the squared factor loadings are between 0 and 1, it forces the loadings to be as close to 0, 1, or -1 as possible. Why is this desirable?

Because V\* gives equal weight to original variables with small and large communalities, the rotated factor loadings are divided by the variable’s communality:

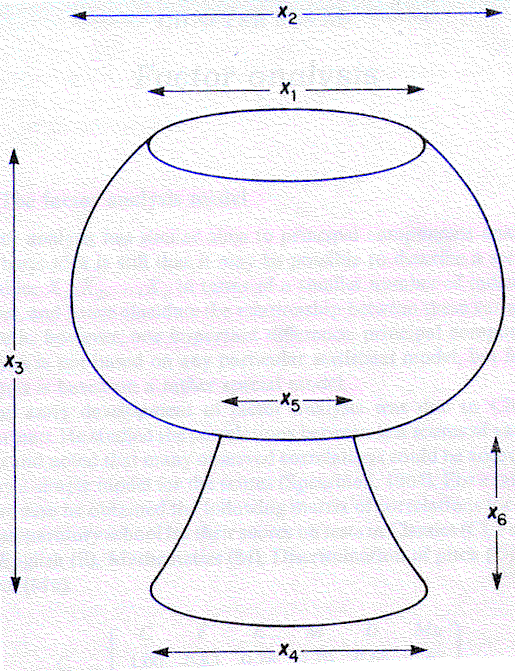


where  is the communality for the jth original variable; i.e., the variance that the common factors account for zj. The **T** which maximizes V produces the varimax rotation of **Λ**.

Fill in the blanks: Corr(\_\_\_, \_\_\_) = bij

Example: Goblet data (GobletFA.R, goblet.csv)

Below is the goblet diagram from earlier:



To find the varimax rotation, we change the rotation argument to "varimax" in factanal(). Note that this is the default value for the argument.

> mod.fit2v <- factanal(x = ~ w1 + w2 + w4 + w5 + w6, factors

= 2, data = goblet2, rotation = "varimax")

> print(x = mod.fit2v, cutoff = 0.0)

Call:

factanal(x = ~w1 + w2 + w4 + w5 + w6, factors = 2, data = goblet2, rotation = "varimax")

Uniquenesses:

w1 w2 w4 w5 w6

0.160 0.106 0.005 0.308 0.506

Loadings:

Factor1 Factor2

w1 0.909 0.118

w2 0.945 0.027

w4 0.467 0.881

w5 0.707 0.439

w6 -0.033 0.702

Factor1 Factor2

SS loadings 2.439 1.477

Proportion Var 0.488 0.295

Cumulative Var 0.488 0.783

Test of the hypothesis that 2 factors are sufficient.

The chi square statistic is 4.74 on 1 degree of freedom.

The p-value is 0.0294

R gives the “rotated” factor loadings to be



Remember that our FA model is **z** = **TT**′**f** + **η** =  + **η**. The estimate of this model is:



Below is a plot of the rotated common factor loadings (see program for code):



In addition to a rotation, a “reflection” of w6 occurs here. This is why the w6 part does not look like what was expected given the previous rotation of the axes. See Johnson and Wichern’s textbook if you would like to see some of the geometry involved.

The final interpretation of the common factors are:

* Factor #1 is a measure of the goblet width.
* Factor #2 is a measure of the goblet base.

Note that R provides the orthogonal matrix used in the rotation as

> mod.fit2v$rotmat # T

[,1] [,2]

[1,] 0.8669910 -0.4983238

[2,] 0.4983238 0.8669910

Through matrix multiplication for , we obtain

> mod.fit2$loadings[,]%\*%t(mod.fit2v$rotmat)

[,1] [,2]

w1 0.11758459 0.90906782

w2 0.02702861 0.94508706

w4 0.88128798 0.46728103

w5 0.43874254 0.70706488

w6 0.70217730 -0.03257242

Interestingly, the first column above is the second column given in the factanal() output, and the second column above is the first column given in the factanal() output. Within factanal(), there is a function named sortloadings() which is run that puts the common factor loadings in the correct order.

Oblique rotation methods

It is often not possible to rotate the orthogonal axes in a factor loading plot in a way such that most loadings are close to -1, 0, or 1. Oblique rotations allow one to rotate without keeping the factor axes orthogonal.

For these types of rotations, **ΛQ** is used to do the rotations where **Q** is not an orthogonal matrix. The new factors produced are NOT orthogonal, which is a contradiction to the FA model assumptions (Cov(**f**) = **I**)!

**Common factor scores**

Similar to PCA, scores need to be assigned to each experimental unit for each “new” variable (common factor). This is not as easy to do in FA because **z** = **Λf** + **η** contains **η** which is unknown and **Λ** is estimated.

Bartlett’s method (a.k.a.,weighted least-squares method)

The FA model again is **z** = **Λf** + **η** where **z** denotes standardized data. For the rth observation, find the **f** that minimizes



where  is a column vector of the standardized values for the rth observation. Notice that  is a multivariate residual. It can be shown that the **f** that minimizes the above expression is



Please note that the **f** includes a subscript r here! Thus, this needs to be done for each observation. For students who have taken a regression analysis, this is direct application of weighted least squares estimation.

Suppose the linear model is **Y** = **Xβ**+**ε** where **ε** ~ N(**0**,**Σ**). The weighted least squares estimate of **β** is .

Thompson’s method (a.k.a., regression method)

For normally distributed data, the joint distribution of **z** and **f** is

****

Remember that Cov(**z**) = **P** because **z** is standardized and Cov(**z**, **f**) = **Λ**.

The conditional expectation of **f** given **z** = **z**\* is

E[**f** | **z** = **z**\*] = **Λ**′**P**-1**z**\*

This result comes from what would be taught in regression course. To take into account our model, we can substitute  for **P** to obtain:



The scores argument in factanal() is used to calculate the factor scores. The default value is “none”. Barlett’s method uses the value “Bartlett” and Thompson’s method uses the value “regression”. Factor scores are returned in the scores component of the model fit object.

Example: Goblet data (GobletFA.R, goblet.csv)

I will be using the rotated factor loadings from the varimax method for this example.

> mod.fit2reg <- factanal(x = ~ w1 + w2 + w4 + w5 + w6,

factors = 2, data = goblet2, rotation = "varimax",

scores = "regression")

> head(mod.fit2reg$scores)

Factor1 Factor2

1 -0.1392788 -0.8951589

2 -1.4868762 1.3525355

3 0.1691278 0.8430629

4 1.4717396 1.5829969

5 2.1175764 1.2319320

6 -0.6162346 0.1926406

> #Example calculation for first observation

> Z <- scale(goblet2[,-1])

> t(mod.fit2reg$loadings[,]) %\*% solve(

mod.fit2reg$loadings[,] %\*% t(mod.fit2reg$loadings[,])

+ diag(mod.fit2reg$uniqueness)) %\*% Z[1,]

[,1]

Factor1 -0.1392788

Factor2 -0.8951589

> mod.fit2Bart <- factanal(x = ~ w1 + w2 + w4 + w5 + w6,

factors = 2, data = goblet2, rotation = "varimax",

scores = "Bartlett")

> head(mod.fit2Bart$scores)

Factor1 Factor2

1 -0.1144012 -0.9142048

2 -1.6483235 1.4477549

3 0.1484676 0.8595147

4 1.5171701 1.5687074

5 2.2238781 1.1827534

6 -0.6687441 0.2220353

> #Example calculation for first observation

> solve(t(mod.fit2reg$loadings[,]) %\*%

diag(1/mod.fit2reg$uniqueness) %\*%

mod.fit2reg$loadings[,]) %\*% t(mod.fit2reg$loadings[,])

%\*% diag(1/mod.fit2reg$uniqueness) %\*% Z[1,]

[,1]

Factor1 -0.1144012

Factor2 -0.9142048

Plot of the factor scores from the regression method:

> par(pty = "s")

> common.limits <- c(min(mod.fit2reg$scores),

max(mod.fit2reg$scores))

> plot(x = mod.fit2reg$scores[,1], y =

mod.fit2reg$scores[,2], xlab = "Factor #1", ylab =

"Factor #2", main = "Factor score plot (regression

method)", xlim = common.limits, ylim = common.limits,

panel.first = grid(col = "lightgray", lty = "dotted"))

> abline(h = 0)

> abline(v = 0)

> text(x = mod.fit2reg$scores[,1], y =

mod.fit2reg$scores[,2]+0.2)



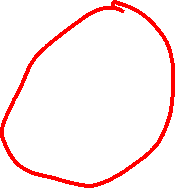
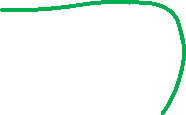
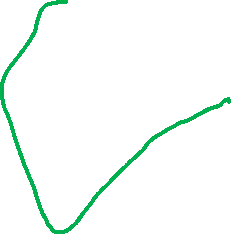
What are we looking for in this plot?

* 1. Trends
  2. Possible groupings
  3. Outliers

Below are the scatter plots from the PCA for comparison purposes:

Correlation matrix





Covariance matrix



Suppose a different rotation method was used. Could this plot change?