21:06 Thursday 17th March, 2016


# Chapter 17

# Factor Models

## 17.1 From PCA to Factor Analysis

Let's sum up PCA. We start with $n$ different $p$-dimensional vectors as our data, i.e., each observation as $p$ numerical variables. We want to reduce the number of dimensions to something more manageable, say $q$. The principal components of the data are the $q$ orthogonal directions of greatest variance in the original $p$-dimensional space; they can be found by taking the top $q$ eigenvectors of the sample covariance matrix. Principal components analysis summarizes the data vectors by projecting them on to the principal components.

All of this is purely an algebraic undertaking; it involves no probabilistic assumptions whatsoever. It also supports no statistical inferences — saying nothing about the population or stochastic process which made the data, it just summarizes the data. How can we add some probability, and so some statistics? And what does that let us do?

Start with some notation. $\mathbf{X}$ is our data matrix, with $n$ rows for the different observations and $p$ columns for the different variables, so $X_{ij}$ is the value of variable $j$ in observation $i$. Each principal component is a vector of length $p$, and there are $p$ of them, so we can stack them together into a $p \times p$ matrix, say $\mathbf{w}$. Finally, each data vector has a projection on to each principal component, which we collect into an $n \times p$ matrix $\mathbf{F}$. Then

$$\begin{aligned} \mathbf{X} &= \mathbf{F}\mathbf{w} \\ [n \times p] &= [n \times p][p \times p] \end{aligned} \tag{17.1}$$

where I've checked the dimensions of the matrices underneath. This is an exact equation involving no noise, approximation or error, but it's kind of useless; we've replaced $p$-dimensional vectors in $\mathbf{X}$ with $p$-dimensional vectors in $\mathbf{F}$. If we keep only to $q < p$ largest principal components, that corresponds to dropping columns from

$\mathbf{F}$ and rows from $\mathbf{w}$. Let's say that the truncated matrices are $\mathbf{F}_q$ and $\mathbf{w}_q$. Then

$$\begin{aligned} \mathbf{X} &\approx \mathbf{F}_q \mathbf{w}_q \\ [n \times p] &= [n \times q][q \times p] \end{aligned} \tag{17.2}$$

The error of approximation — the difference between the left- and right- hand-sides of Eq. 17.2 — will get smaller as we increase $q$. (The line below the equation is a sanity-check that the matrices are the right size, which they are. Also, at this point the subscript $q$s get too annoying, so I'll drop them.) We can of course make the two sides match exactly by adding an error or residual term on the right:

$$\mathbf{X} = \mathbf{F}\mathbf{w} + \epsilon \tag{17.3}$$

where $\epsilon$ has to be an $n \times p$ matrix.

Now, Eq. 17.3 should look more or less familiar to you from regression. On the left-hand side we have a measured outcome variable ($\mathbf{X}$), and on the right-hand side we have a systematic prediction term ($\mathbf{F}\mathbf{w}$) plus a residual ($\epsilon$). Let's run with this analogy, and start treating $\epsilon$ as *noise*, as a random variable which has got some distribution, rather than whatever arithmetic says is needed to balance the two sides. (This move is the difference between just drawing a straight line through a scatter plot, and inferring a linear regression.) Then $\mathbf{X}$ will also be a random variable. When we want to talk about the random variable which goes in the $i^{\text{th}}$ column of $\mathbf{X}$, we'll call it $X_i$.

What about $\mathbf{F}$? Well, in the analogy it corresponds to the independent variables in the regression, which ordinarily we treat as fixed rather than random, but that's because we actually get to observe them; here we don't, so it will make sense to treat $\mathbf{F}$, too, as random. Now that they are random variables, we say that we have $q$ **factors**, rather than components, that $\mathbf{F}$ is the matrix of **factor scores** and $\mathbf{w}$ is the matrix of **factor loadings**. The variables in $\mathbf{X}$ are called **observable** or **manifest** variables, those in $\mathbf{F}$ are **hidden** or **latent**. (Technically $\epsilon$ is also latent.)

Before we can actually do much with this model, we need to say more about the distributions of these random variables. The traditional choices are as follows.

1. All of the observable random variables $X_i$ have mean zero and variance 1.
2. All of the latent factors have mean zero and variance 1.
3. The noise terms $\epsilon$ all have mean zero.
4. The factors are uncorrelated across individuals (rows of $\mathbf{F}$) and across variables (columns).
5. The noise terms are uncorrelated across individuals, and across observable variables.
6. The noise terms are uncorrelated with the factor variables.

Item (1) isn't restrictive, because we can always center and standardize our data. Item (2) isn't restrictive either — we could always center and standardize the factor variables without really changing anything. Item (3) actually follows from (1) and (2). The substantive assumptions — the ones which will give us predictive power but could also go wrong, and so really define the factor model — are the others, about lack of correlation. Where do they come from?

Remember what the model looks like:

$$\mathbf{X} = \mathbf{F}\mathbf{w} + \epsilon \tag{17.4}$$

*All* of the systematic patterns in the observations $\mathbf{X}$ should come from the first term on the right-hand side. The residual term $\epsilon$ should, if the model is working, be unpredictable noise. Items (3) through (5) express a very strong form of this idea. In particular it's vital that the noise be uncorrelated with the factor scores.

### 17.1.1 Preserving correlations

There is another route from PCA to the factor model, which many people like but which I find less compelling; it starts by changing the objectives.

PCA aims to minimize the mean-squared distance from the data to their projects, or what comes to the same thing, to preserve variance. But it doesn't preserve correlations. That is, the correlations of the features of the image vectors are not the same as the correlations among the features of the original vectors (unless $q = p$, and we're not really doing any data reduction). We might value those correlations, however, and want to preserve them, rather than trying to approximate the actual data.[1] That is, we might ask for a set of vectors whose image in the feature space will have the same correlation matrix as the original vectors, or as close to the same correlation matrix as possible while still reducing the number of dimensions. This leads to the factor model we've already reached, as we'll see.

## 17.2 The Graphical Model

It's common to represent factor models visually, as in Figure 17.1. This is an example of a **graphical model**, in which the nodes or vertices of the graph represent random variables, and the edges of the graph represent direct statistical dependencies between the variables. The figure shows the observables or features in square boxes, to indicate that they are manifest variables we can actual measure; above them are the factors, drawn in round bubbles to show that we don't get to see them. The fact that there are no direct linkages between the factors shows that they are independent of one another. From below we have the noise terms, one to an observable.

[1] Why? Well, originally the answer was that the correlation coefficient had just been invented, and was about the only way people had of measuring relationships between variables. Since then it's been propagated by statistics courses where it is the only way people are *taught* to measure relationships. The great statistician John Tukey once wrote "Does anyone know when the correlation coefficient is useful, as opposed to when it is used? If so, why not tell us?" (Tukey, 1954, p. 721).

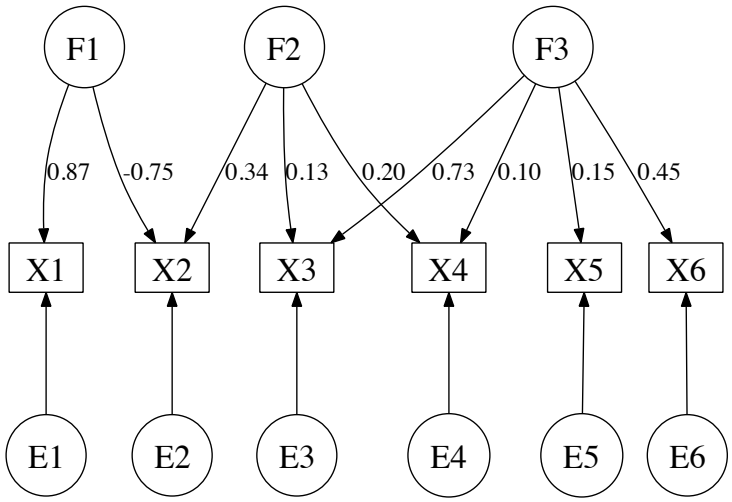



FIGURE 17.1: *Graphical model form of a factor model. Circles stand for the unobserved variables (factors above, noises below), boxes for the observed features. Edges indicate non-zero coefficients — entries in the factor loading matrix* **w**, *or specific variances* $\psi_i$. *Arrows representing entries in* **w** *are decorated with those entries. Note that it is common to omit the noise variables in such diagrams, with the implicit understanding that every variable with an incoming arrow also has an incoming noise term.*

Notice that not every observable is connected to every factor: this depicts the fact that some entries in **w** are zero. In the figure, for instance, $X_1$ has an arrow only from $F_1$ and not the other factors; this means that while $w_{11} = 0.87$, $w_{21} = w_{31} = 0$.

Drawn this way, one sees how the factor model is **generative** — how it gives us a recipe for producing new data. In this case, it's: draw new, independent values for the factor scores $F_1, F_2, \ldots F_q$; add these up with weights from **w**; and then add on the final noises $\epsilon_1, \epsilon_2, \ldots \epsilon_p$. If the model is right, this is a procedure for generating new, synthetic data with the same characteristics as the real data. In fact, it's a story about how the real data came to be — that there really are some latent variables (the factor scores) which linearly cause the observables to have the values they do.

### 17.2.1 Observables Are Correlated Through the Factors

One of the most important consequences of the factor model is that observable variables are correlated with each other *solely* because they are correlated with the hidden factors. To see how this works, take $X_1$ and $X_2$ from the diagram, and let's calculate their covariance. (Since they both have variance 1, this is the same as their correlation.)

$$\begin{aligned}
\mathrm{Cov}[X_1, X_2] &= \mathbb{E}[X_1 X_2] - \mathbb{E}[X_1]\mathbb{E}[X_2] && (17.5)\\
&= \mathbb{E}[X_1 X_2] && (17.6)\\
&= \mathbb{E}[(F_1 w_{11} + F_2 w_{21} + \epsilon_1)(F_1 w_{12} + F_2 w_{22} + \epsilon_2)] && (17.7)\\
&= \mathbb{E}\left[F_1^2 w_{11} w_{12} + F_1 F_2 (w_{11} w_{22} + w_{21} w_{12}) + F_2^2 w_{21} w_{22}\right] \\
&\quad + \mathbb{E}[\epsilon_1 \epsilon_2] + \mathbb{E}[\epsilon_1 (F_1 w_{12} + F_2 w_{22})] \\
&\quad + \mathbb{E}[\epsilon_2 (F_1 w_{11} + F_2 w_{21})] && (17.8)
\end{aligned}$$

Since the noise terms are uncorrelated with the factor scores, and the noise terms for different variables are uncorrelated with each other, *all* the terms containing $\epsilon$s have expectation zero. Also, $F_1$ and $F_2$ are uncorrelated, so

$$\begin{aligned}
\mathrm{Cov}[X_1, X_2] &= \mathbb{E}\left[F_1^2\right] w_{11} w_{12} + \mathbb{E}\left[F_2^2\right] w_{21} w_{22} && (17.9)\\
&= w_{11} w_{12} + w_{21} w_{22} && (17.10)
\end{aligned}$$

using the fact that the factors are scaled to have variance 1. This says that the covariance between $X_1$ and $X_2$ is what they have from both correlating with $F_1$, plus what they have from both correlating with $F_2$; if we had more factors we would add on $w_{31} w_{32} + w_{41} w_{42} + \ldots$ out to $w_{q1} w_{q2}$. And of course this would apply as well to any other pair of observable variables. So the general form is

$$\mathrm{Cov}\left[X_i, X_j\right] = \sum_{k=1}^{q} w_{ki} w_{kj} \tag{17.11}$$

so long as $i \neq j$.

The jargon says that observable $i$ **loads on** factor $k$ when $w_{ki} \neq 0$. If two observables do not load on to any of the same factors, if they do not share any common factors, then they will be independent. If we could condition on ("control for") the factors, all of the observables would be conditionally independent.

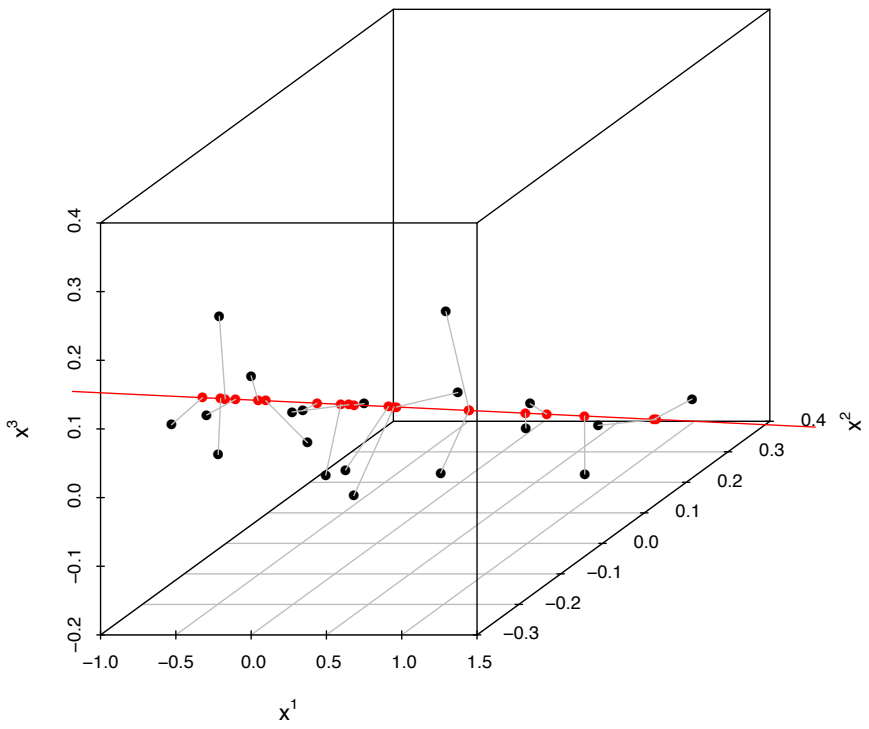
Graphically, we draw an arrow from a factor node to an observable node if and only if the observable loads on the factor. So then we can just *see* that two observables are correlated if they both have in-coming arrows from the same factors. (To find the actual correlation, we multiply the weights on all the edges connecting the two observable nodes to the common factors; that's Eq. 17.11.) Conversely, even though the factors are marginally independent of each other, if two factors both send arrows to the same observable, then they are dependent conditional on that observable.[2]

[2] To see that this makes sense, suppose that $X_1 = F_1 w_{11} + F_2 w_{21} + \epsilon_1$. If we know the value of $X_1$, we know what $F_1$, $F_2$ and $\epsilon_1$ have to add up to, so they are conditionally dependent.

### 17.2.2 Geometry: Approximation by Linear Subspaces

Each observation we take is a vector in a $p$-dimensional space; the factor model says that these vectors have certain geometric relations to each other — that the data has a certain shape. To see what that is, pretend for right now that we can turn off the noise terms $\epsilon$. The loading matrix $\mathbf{w}$ is a $q \times p$ matrix, so each row of $\mathbf{w}$ is a vector in $p$-dimensional space; call these vectors $\vec{w}_1, \vec{w}_2, \ldots \vec{w}_q$. Without the noise, our observable vectors would be linear combinations of these vectors (with the factor scores saying how much each vector contributes to the combination). Since the factors are orthogonal to each other, we know that they span a $q$-dimensional sub-space of the $p$-dimensional space — a line if $q = 1$, a plane if $q = 2$, in general a linear subspace. If the factor model is true and we turn off noise, we would find all the data lying *exactly* on this subspace. Of course, *with* noise we expect that the data vectors will be scattered around the subspace; how close depends on the variance of the noise. (Figure 17.2.) But this is still a rather specific prediction about the shape of the data.

A weaker prediction than "the data lie on a low-dimensional linear subspace in the high-dimensional space" is "the data lie on some low-dimensional surface, possibly curved, in the high-dimensional space"; there are techniques for trying to recover such surfaces. Chapter 18 introduces two such techniques, but this is a broad and still-growing area.




```
n <- 20; library(scatterplot3d)
f <- matrix(sort(rnorm(n)),ncol=1); w <- matrix(c(0.5,0.2,-0.1),nrow=1)
fw <- f %*% w; x <- fw + matrix(rnorm(n*3,sd=c(.15,.05,.09)),ncol=3,byrow=TRUE)
s3d <- scatterplot3d(x,xlab=expression(x^1),ylab=expression(x^2),
                     zlab=expression(x^3),pch=16)
s3d$points3d(matrix(seq(from=min(f)-1,to=max(f)+1,length.out=2),ncol=1)%*%w,
             col="red",type="l")
s3d$points3d(fw,col="red",pch=16)
for (i in 1:nrow(x)) {
  s3d$points3d(x=c(x[i,1],fw[i,1]),y=c(x[i,2],fw[i,2]),z=c(x[i,3],fw[i,3]),
               col="grey",type="l") }
```

FIGURE 17.2: *Geometry of factor models: Black dots are observed vectors $\vec{X}$ in $p = 3$ dimensions. These were generated from the $q = 1$ dimensional factor scores $\vec{F}$ by taking $\vec{F}\mathbf{w}$ (red dots) and adding independent noise (grey lines). The $q$-dimensional subspace along which all values of $\vec{F}\mathbf{w}$ must fall is also shown in red. (See also exercise 3.)*

## 17.3 Roots of Factor Analysis in Causal Discovery

The roots of factor analysis go back to work by Charles Spearman just over a century ago (Spearman, 1904); he was trying to discover the hidden structure of human intelligence. His observation was that schoolchildren's grades in different subjects were all correlated with each other. He went beyond this to observe a particular *pattern* of correlations, which he thought he could explain as follows: the reason grades in math, English, history, etc., are all correlated is performance in these subjects is all correlated with *something else*, a general or **common** factor, which he named "general intelligence", for which the natural symbol was of course $g$ or $G$.

Put in a form like Eq. 17.4, Spearman's model becomes

$$\mathbf{X} = \epsilon + \mathbf{G}\mathbf{w} \tag{17.12}$$

where $\mathbf{G}$ is an $n \times 1$ matrix (i.e., a row vector) and $\mathbf{w}$ is a $1 \times p$ matrix (i.e., a column vector). The correlation between feature $i$ and $G$ is just $w_i \equiv w_{1i}$, and, if $i \neq j$,

$$\mathbf{v}_{ij} \equiv \mathrm{Cov}\left[X_i, X_j\right] = w_i w_j \tag{17.13}$$

where I have introduced $\mathbf{v}_{ij}$ as a short-hand for the covariance.

Up to this point, this is all so much positing and assertion and hypothesis. What Spearman did next, though, was to observe that this hypothesis carried a very strong implication about the *ratios* of correlation coefficients. Pick any four distinct features, $i, j, k, l$. Then, if the model (17.12) is true,

$$\frac{\mathbf{v}_{ij}/\mathbf{v}_{kj}}{\mathbf{v}_{il}/\mathbf{v}_{kl}} = \frac{w_i w_j / w_k w_j}{w_i w_l / w_k w_l} \tag{17.14}$$

$$= \frac{w_i / w_k}{w_i / w_k} \tag{17.15}$$

$$= 1 \tag{17.16}$$

The relationship

$$\mathbf{v}_{ij}\mathbf{v}_{kl} = \mathbf{v}_{il}\mathbf{v}_{kj} \tag{17.17}$$

is called the "tetrad equation", and we will meet it again later when we consider methods for causal discovery in Part IV. In Spearman's model, this is one tetrad equation for every set of four distinct variables.

Spearman found that the tetrad equations held in his data on school grades (to a good approximation), and concluded that a single general factor of intelligence must exist[3]. This was, of course, logically fallacious.

Later work, using large batteries of different kinds of intelligence tests, showed that the tetrad equations do not hold in general, or more exactly that departures from them are too big to explain away as sampling noise. (Recall that the equations are about the true correlations between the variables, but we only get to see sample

[3] Actually, the equations didn't hold when music was one of the grades, so Spearman argued musical ability did not load on general intelligence.

correlations, which are always a little off.) The response, done in an *ad hoc* way by Spearman and his followers, and then more systematically by Thurstone, was to introduce *multiple* factors. This breaks the tetrad equation, but still accounts for the correlations among features by saying that features are really directly correlated with factors, and uncorrelated conditional on the factor scores. Thurstone's form of factor analysis is basically the one people still use — there have been refinements, of course, but it's mostly still his method.

## 17.4 Estimation

The factor model introduces a whole bunch of new variables to explain the observables: the factor scores $\mathbf{F}$, the factor loadings or weights $\mathbf{w}$, and the observable-specific variances $\psi_i$. The factor scores are specific to each individual, and individuals by assumption are independent, so we can't expect them to really generalize. But the loadings $\mathbf{w}$ are, supposedly, characteristic of the population. So it would be nice if we could separate estimating the population parameters from estimating the attributes of individuals; here's how.

Since the variables are centered, we can write the covariance matrix in terms of the data frames:

$$\mathbf{v} = \mathbb{E}\left[\frac{1}{n}\mathbf{X}^T\mathbf{X}\right] \tag{17.18}$$

(This is the true, population covariance matrix on the left.) But the factor model tells us that

$$\mathbf{X} = \mathbf{F}\mathbf{w} + \epsilon \tag{17.19}$$

This involves the factor scores $\mathbf{F}$, but reember that when we looked at the correlations between individual variables, those went away, so let's substitute Eq. 17.19 into Eq. 17.18 and see what happens:

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{n}\mathbf{X}^T\mathbf{X}\right] && (17.20)\\
&= \frac{1}{n}\mathbb{E}\left[(\epsilon^T + \mathbf{w}^T\mathbf{F}^T)(\mathbf{F}\mathbf{w} + \epsilon)\right] && (17.21)\\
&= \frac{1}{n}\left(\mathbb{E}\left[\epsilon^T\epsilon\right] + \mathbf{w}^T\mathbb{E}\left[\mathbf{F}^T\epsilon\right] + \mathbb{E}\left[\epsilon^T\mathbf{F}\right]\mathbf{w} + \mathbf{w}^T\mathbb{E}\left[\mathbf{F}^T\mathbf{F}\right]\mathbf{w}\right) && (17.22)\\
&= \psi + 0 + 0 + \frac{1}{n}\mathbf{w}^T n\mathbf{I}\mathbf{w} && (17.23)\\
&= \psi + \mathbf{w}^T\mathbf{w} && (17.24)
\end{aligned}$$

Behold:

$$\mathbf{v} = \psi + \mathbf{w}^T\mathbf{w} \tag{17.25}$$

The individual-specific variables $\mathbf{F}$ have gone away, leaving only population parameters on both sides of the equation.

### 17.4.1 Degrees of Freedom

It only takes a bit of playing with Eq. 17.25 to realize that we are in trouble. Like any matrix equation, it represents a system of equations. How many equations in how many unknowns? Naively, we'd say that we have $p^2$ equations (one for each element of the matrix $\mathbf{v}$), and $p + pq$ unknowns (one for each diagonal element of $\psi$, plus one for each element of $\mathbf{w}$). If there are more equations than unknowns, then there is generally no solution; if there are fewer equations than unknowns, then there are generally infinitely many solutions. Either way, solving for $\mathbf{w}$ seems hopeless (unless $q = p - 1$, in which case it's not very helpful). What to do?

Well, first let's do the book-keeping for degrees of freedom more carefully. The observables variables are scaled to have standard deviation one, so the diagonal entries of $\mathbf{v}$ are all 1. Moreover, any covariance matrix is symmetric, so we are left with only $p(p-1)/2$ degrees of freedom in $\mathbf{v}$ — only that many equations. On the other side, scaling to standard deviation 1 means we don't *really* need to solve separately for $\psi$ — it's fixed as soon as we know what $\mathbf{w}^T\mathbf{w}$ is — which saves us $p$ unknowns. Also, the entries in $\mathbf{w}$ are not completely free to vary independently of each other, because each row has to be orthogonal to every other row. (Look back at the notes on PCA.) Since there are $q$ rows, this gives is $q(q-1)/2$ constraints on $\mathbf{w}$ — we can think of these as either extra equations, or as reductions in the number of free parameters (unknowns).[4]

Summarizing, we really have $p(p-1)/2$ degrees of freedom in $\mathbf{v}$, and $pq - q(q-1)/2$ degrees of freedom in $\mathbf{w}$. If these two match, then there is (in general) a unique solution which will give us $\mathbf{w}$. But in general they will *not* be equal; then what? Let us consider the two cases.

**More unknowns (free parameters) than equations (constraints)** This is fairly straightforward: there is no unique solution to Eq. 17.25; instead there are *infinitely many* solutions. It's true that the loading matrix $\mathbf{w}$ does have to satisfy some constraints, that not just any $\mathbf{w}$ will work, so the data does give us some information, but there is a continuum of different parameter settings which are all match the covariance matrix perfectly. (Notice that we are working with the population parameters here, so this isn't an issue of having only a limited sample.) There is just no way to use data to decide between these different parameters, to identify which one is right, so we say the model is **unidentifiable**. Most software for factor analysis, include R's `factanal` function, will check for this and just refuse to fit a model with too many factors relative to the number of observables.

**More equations (constraints) than unknowns (free parameters)** This is more interesting. In general, systems of equations like this are **overdetermined**, meaning that there is no way to satisfy all the constraints at once, and there isn't even a single solution. It's just not possible to write an arbitrary covariance matrix $\mathbf{v}$ among, say, seven variables in terms of, say, a one-factor model (as $p(p-1)/2 = 7(7-1)/2 = 21 >$

[4] Notice that $\psi + \mathbf{w}^T\mathbf{w}$ is automatically symmetric, since $\psi$ is diagonal, so we don't need to impose any extra constraints to get symmetry.

$7(1)-1(1-1)/2 = 7 = pq - q(q-1)/2$). But it *is* possible for special covariance matrices. In these situations, the factor model actually has testable implications for the data — it says that only certain covariance matrices are possible and not others. For example, we saw above that the one-fator model implies the tetrad equations must hold among the observable covariances; the constraints on $\mathbf{v}$ for multiple-factor models are similar in kind but more complicated algebraically. By testing these implications, we can check whether or not the our favorite factor model is right.[5]

Now we don't know the true, population covariance matrix $\mathbf{v}$, but we can estimate it from data, getting an estimate $\widehat{\mathbf{v}}$. The natural thing to do then is to equate this with the parameters and try to solve for the latter:

$$\widehat{\mathbf{v}} = \widehat{\psi} + \widehat{\mathbf{w}}^T \widehat{\mathbf{w}} \tag{17.26}$$

The book-keeping for degrees of freedom here is the same as for Eq. 17.25. If $q$ is too large relative to $p$, the model is unidentifiable; if it is too small, the matrix equation can only be solved if $\widehat{\mathbf{v}}$ is of the right, restricted form, i.e., if the model is right. Of course even if the model is right, the sample covariances are the true covariances plus noise, so we shouldn't expect to get an *exact* match, but we can try in various way to minimize the discrepancy between the two sides of the equation.

### 17.4.2 A Clue from Spearman's One-Factor Model

Remember that in Spearman's model with a single general factor, the covariance between observables $i$ and $j$ in that model is the product of their factor weightings:

$$\mathbf{v}_{ij} = w_i w_j \tag{17.27}$$

The exception is that $\mathbf{v}_{ii} = w_i^2 + \psi_i$, rather than $w_i^2$. However, if we look at $\mathbf{u} = \mathbf{v} - \psi$, that's the same as $\mathbf{v}$ off the diagonal, and a little algebra shows that its diagonal entries are, in fact, just $w_i^2$. So if we look at any two rows of $\mathbf{U}$, they're proportional to each other:

$$\mathbf{u}_{ij} = \frac{w_i}{w_k} \mathbf{u}_{kj} \tag{17.28}$$

This means that, when Spearman's model holds true, there is actually only *one* linearly-independent row in in $\mathbf{u}$.

Recall from linear algebra that the **rank** of a matrix is how many linearly independent rows it has.[6] Ordinarily, the matrix is of **full rank**, meaning all the rows are linearly independent. What we have just seen is that when Spearman's model holds, the matrix $\mathbf{u}$ is *not* of full rank, but rather of rank 1. More generally, when the factor model holds with $q$ factors, the matrix $\mathbf{u} = \mathbf{w}^T \mathbf{w}$ has rank $q$. The diagonal entries of $\mathbf{u}$, called the **common variances** or **commonalities**, are no longer automatically 1,

[5] Actually, we need to be a little careful here. If we find that the tetrad equations don't hold, we know a one-factor model must be wrong. We could only conclude that the one-factor model must be right if we found that the tetrad equations held, *and* that there were no other models which implied those equations; but, as we'll see, there are.

[6] We could also talk about the columns; it wouldn't make any difference.

but rather show how much of the variance in each observable is associated with the variances of the latent factors. Like $\mathbf{v}$, $\mathbf{u}$ is a positive symmetric matrix.

Because $\mathbf{u}$ is a positive symmetric matrix, we know from linear algebra that it can be written as

$$\mathbf{u} = \mathbf{c}\mathbf{d}\mathbf{c}^T \tag{17.29}$$

where $\mathbf{c}$ is the matrix whose columns are the eigenvectors of $\mathbf{u}$, and $\mathbf{d}$ is the diagonal matrix whose entries are the eigenvalues. That is, if we use all $p$ eigenvectors, we can reproduce the covariance matrix exactly. Suppose we instead use $\mathbf{c}_q$, the $p \times q$ matrix whose columns are the eigenvectors going with the $q$ largest eigenvalues, and likewise make $\mathbf{d}_q$ the diagonal matrix of those eigenvalues. Then $\mathbf{c}_q \mathbf{d}_q {\mathbf{c}_q}^T$ will be a symmetric positive $p \times p$ matrix. This is a matrix of rank $q$, and so can only equal $\mathbf{u}$ if the latter also has rank $q$. Otherwise, it's an approximation which grows more accurate as we let $q$ grow towards $p$, and, at any given $q$, it's a better approximation to $\mathbf{u}$ than any other rank-$q$ matrix. This, finally, is the precise sense in which factor analysis tries preserve correlations, as opposed to principal components trying to preserve variance.

To resume our algebra, define ${\mathbf{d}_q}^{1/2}$ as the $q \times q$ diagonal matrix of the square roots of the eigenvalues. Clearly $\mathbf{d}_q = {\mathbf{d}_q}^{1/2}{\mathbf{d}_q}^{1/2}$. So

$$\mathbf{c}_q \mathbf{d}_q {\mathbf{c}_q}^T = \mathbf{c}_q {\mathbf{d}_q}^{1/2}{\mathbf{d}_q}^{1/2}{\mathbf{c}_q}^T = \left(\mathbf{c}_q {\mathbf{d}_q}^{1/2}\right)\left(\mathbf{c}_q {\mathbf{d}_q}^{1/2}\right)^T \tag{17.30}$$

So we have

$$\mathbf{u} \approx \left(\mathbf{c}_q {\mathbf{d}_q}^{1/2}\right)\left(\mathbf{c}_q {\mathbf{d}_q}^{1/2}\right)^T \tag{17.31}$$

but at the same time we know that $\mathbf{u} = \mathbf{w}^T\mathbf{w}$. So we just identify $\mathbf{w}$ with $\left(\mathbf{c}_q {\mathbf{d}_q}^{1/2}\right)^T$:

$$\mathbf{w} = \left(\mathbf{c}_q {\mathbf{d}_q}^{1/2}\right)^T \tag{17.32}$$

and we are done with our algebra.

Let's think a bit more about how well we're approximating $\mathbf{v}$. The approximation will always be exact when $q = p$, so that there is one factor for each feature (in which case $\psi = 0$ always). Then all factor analysis does for us is to rotate the coordinate axes in feature space, so that the new coordinates are uncorrelated. (This is the same was what PCA does with $p$ components.) The approximation can *also* be exact with fewer factors than features if the reduced covariance matrix is of less than full rank, and we use at least as many factors as the rank.

### 17.4.3 Estimating Factor Loadings and Specific Variances

The classical method for estimating the factor model is now simply to do this eigenvector approximation on the *sample* correlation matrix. Define the **reduced** or **adjusted** sample correlation matrix as

$$\widehat{\mathbf{u}} = \widehat{\mathbf{v}} - \widehat{\psi} \tag{17.33}$$

We can't actually calculate $\widehat{\mathbf{u}}$ until we know, or have a guess as to, $\widehat{\psi}$. A reasonable and common starting-point is to do a linear regression of each feature $j$ on all the other features, and then set $\widehat{\psi_j}$ to the mean squared error for that regression. (We'll come back to this guess later.)

Once we have the reduced correlation matrix, find its top $q$ eigenvalues and eigenvectors, getting matrices $\widehat{\mathbf{c}_\mathbf{q}}$ and $\widehat{\mathbf{d}_\mathbf{q}}$ as above. Set the factor loadings accordingly, and re-calculate the specific variances:

$$\widehat{\mathbf{w}} = \left(\mathbf{c}_\mathbf{q}\mathbf{d}_\mathbf{q}^{1/2}\right)^T \tag{17.34}$$

$$\widehat{\psi_j} = 1 - \sum_{r=1}^{k} w_{rj}^2 \tag{17.35}$$

$$\tilde{\mathbf{v}} \equiv \widehat{\psi} + \widehat{\mathbf{w}}^T\widehat{\mathbf{w}} \tag{17.36}$$

The "predicted" covariance matrix $\tilde{\mathbf{v}}$ in the last line is exactly right on the diagonal (by construction), and should be closer off-diagonal than anything else we could do with the same number of factors. However, our guess as to $\mathbf{u}$ depended on our initial guess about $\psi$, which has in general changed, so we can try iterating this (i.e., re-calculating $\mathbf{c}_\mathbf{q}$ and $\mathbf{d}_\mathbf{q}$), until we converge.

## 17.5 Maximum Likelihood Estimation

It has probably not escaped your notice that the estimation procedure above requires a starting guess as to $\psi$. This makes its consistency somewhat shaky. (If we continually put in ridiculous values for $\psi$, why should we expect that $\widehat{\mathbf{w}} \to \mathbf{w}$?) On the other hand, we know from our elementary statistics courses that maximum likelihood estimates are generally consistent, unless we choose a spectacularly bad model. Can we use that here?

We can, but at a cost. We have so far got away with just making assumptions about the means and covariances of the factor scores $\mathbf{F}$. To get an actual likelihood, we need to assume something about their distribution as well.

The usual assumption is that $F_{ik} \sim \mathcal{N}(0,1)$, and that the factor scores are independent across factors $k = 1,\ldots q$ and individuals $i = 1,\ldots n$. With this assumption, the features have a multivariate normal distribution $\vec{X}_i \sim \mathcal{N}(0, \psi + \mathbf{w}^T\mathbf{w})$. This means that the log-likelihood is

$$L = -\frac{np}{2}\log 2\pi - \frac{n}{2}\log|\psi + \mathbf{w}^T\mathbf{w}| - \frac{n}{2}\mathrm{tr}\left((\psi + \mathbf{w}^T\mathbf{w})^{-1}\widehat{\mathbf{v}}\right) \tag{17.37}$$

where tr**a** is the **trace** of the matrix **a**, the sum of its diagonal elements. Notice that the likelihood only involves the data through the sample covariance matrix $\widehat{\mathbf{v}}$ — the actual factor scores $\mathbf{F}$ are not needed for the likelihood.

One can either try direct numerical maximization, or use a two-stage procedure. Starting, once again, with a guess as to $\psi$, one finds that the optimal choice of $\psi^{1/2}\mathbf{w}^T$ is given by the matrix whose columns are the $q$ leading eigenvectors of $\psi^{1/2}\widehat{\mathbf{v}}\psi^{1/2}$.

Starting from a guess as to $\mathbf{w}$, the optimal choice of $\psi$ is given by the diagonal entries of $\widehat{\mathbf{v}} - \mathbf{w}^T\mathbf{w}$. So again one starts with a guess about the unique variances (e.g., the residuals of the regressions) and iterates to convergence.[7]

The differences between the maximum likelihood estimates and the "principal factors" approach can be substantial. If the data appear to be normally distributed (as shown by the usual tests), then the additional efficiency of maximum likelihood estimation is highly worthwhile. Also, as we'll see below, it is a lot easier to test the model assumptions if one uses the MLE.

### 17.5.1 Alternative Approaches

Factor analysis is an example of trying to approximate a full-rank matrix, here the covariance matrix, with a low-rank matrix, or a low-rank matrix plus some corrections, here $\psi + \mathbf{w}^T\mathbf{w}$. Such matrix-approximation problems are currently the subject of very intense interest in statistics and machine learning, with many new methods being proposed and refined, and it is very plausible that some of these will prove to work better than older approaches to factor analysis.

In particular, Kao and Van Roy (2013) have recently used these ideas to propose a new factor-analysis algorithm, which simultaneously estimates the number of factors and the factor loadings, and does so through a modification of PCA, distinct from the old "principal factors" method. In their examples, it works better than conventional approaches, but whether this will hold true generally is not clear. They do not, unfortunately, provide code.

### 17.5.2 Estimating Factor Scores

The probably the best method for estimating factor scores is the "regression" or "Thomson" method, which says

$$\widehat{F}_{ir} = \sum_j X_{ij} b_{ij} \tag{17.38}$$

and seeks the weights $b_{ij}$ which will minimize the mean squared error, $\mathbf{E}[(\widehat{F}_{ir} - F_{ir})^2]$. You can work out the $b_{ij}$ as an exercise, assuming you know $\mathbf{w}$.

## 17.6 The Rotation Problem

Recall from linear algebra that a matrix $\mathbf{o}$ is **orthogonal** if its inverse is the same as its transpose, $\mathbf{o}^T\mathbf{o} = \mathbf{I}$. The classic examples are rotation matrices. For instance, to rotate a two-dimensional vector through an angle $\alpha$, we multiply it by

$$\mathbf{r}_\alpha = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \tag{17.39}$$

[7] The algebra is tedious. See section 3.2 in Bartholomew (1987) if you really want it. (Note that Bartholomew has a sign error in his equation 3.16.)

The inverse to this matrix must be the one which rotates through the angle $-\alpha$, $\mathbf{r}_\alpha^{-1} = \mathbf{r}_{-\alpha}$, but trigonometry tells us that $\mathbf{r}_{-\alpha} = \mathbf{r}_\alpha^T$.

To see why this matters to us, go back to the matrix form of the factor model, and insert an orthogonal $q \times q$ matrix and its transpose:

$$\begin{aligned} \mathbf{X} &= \epsilon + \mathbf{F}\mathbf{w} && (17.40)\\ &= \epsilon + \mathbf{F}\mathbf{o}\mathbf{o}^T\mathbf{w} && (17.41)\\ &= \epsilon + \mathbf{H}\mathbf{y} && (17.42) \end{aligned}$$

We've changed the factor scores to $\mathbf{H} \equiv \mathbf{F}\mathbf{o}$, and we've changed the factor loadings to $\mathbf{y} \equiv \mathbf{o}^T\mathbf{w}$, but nothing about the features has changed *at all*. We can do as many orthogonal transformations of the factors as we like, with no observable consequences whatsoever.[8]

Statistically, the fact that different parameter settings give us the same observational consequences means that the parameters of the factor model are **unidentifiable**. The rotation problem is, as it were, the revenant of having an ill-posed problem: we thought we'd slain it through heroic feats of linear algebra, but it's still around and determined to have its revenge.[9]

Mathematically, this should not be surprising at all. The factors live in a $q$-dimensional vector space of their own. We should be free to set up any coordinate system we feel like on that space. Changing coordinates in factor space will just require a compensating change in how factor-space coordinates relate to feature space (the factor loadings matrix $\mathbf{w}$). That's all we've done here with our orthogonal transformation.

Substantively, this should be rather troubling. If we can rotate the factors as much as we like without consequences, how on Earth can we interpret them?

## 17.7 Factor Analysis as a Predictive Model

Unlike principal components analysis, factor analysis really does give us a predictive model. Its prediction is that if we draw a new member of the population and look at the vector of observables we get from them,

$$\vec{X} \sim \mathcal{N}(0, \mathbf{w}^T\mathbf{w} + \psi) \quad (17.43)$$

[8] Notice that the log-likelihood only involves $\mathbf{w}^T\mathbf{w}$, which is equal to $\mathbf{w}^T\mathbf{o}\mathbf{o}^T\mathbf{w} = \mathbf{y}^T\mathbf{y}$, so even assuming Gaussian distributions doesn't let us tell the difference between the original and the transformed variables. In fact, if $\vec{F} \sim \mathcal{N}(0, \mathbf{I})$, then $\vec{F}\mathbf{o} \sim \mathcal{N}(0\mathbf{o}, \mathbf{o}^T\mathbf{I}\mathbf{o}) = \mathcal{N}(0, \mathbf{I})$ — in other words, the rotated factor scores still satisfy our distributional assumptions.

[9] Remember that we obtained the loading matrix $\mathbf{w}$ as a solution to $\mathbf{w}^T\mathbf{w} = \mathbf{u}$, that is to we got $\mathbf{w}$ as a kind of matrix square root of the reduced correlation matrix. For a real number $u$ there are two square roots, i.e., two numbers $w$ such that $w \times w = u$, namely the usual $w = \sqrt{u}$ and $w = -\sqrt{u}$, because $(-1) \times (-1) = 1$. Similarly, whenever we find one solution to $\mathbf{w}^T\mathbf{w} = \mathbf{u}$, $\mathbf{o}^T\mathbf{w}$ is another solution, because $\mathbf{o}\mathbf{o}^T = \mathbf{I}$. So while the usual "square root" of $\mathbf{u}$ is $\mathbf{w} = \mathbf{d_q}^{1/2}\mathbf{c}$, for any orthogonal matrix $\mathbf{o}^T\mathbf{d_q}^{1/2}\mathbf{c}$ will always work just as well.

if we make the usual distributional assumptions. Of course it might seem like it makes a more refined, conditional prediction,

$$\vec{X}|\vec{F} \sim \mathcal{N}(F\mathbf{w}, \psi) \tag{17.44}$$

but the problem is that there is no way to guess at or estimate the factor scores $\vec{F}$ until after we've seen $\vec{X}$, at which point anyone can predict $X$ perfectly. So the actual *forecast* is given by Eq. 17.43.[10]

Now, without going through the trouble of factor analysis, one could always just postulate that

$$\vec{X} \sim \mathcal{N}(0, \mathbf{v}) \tag{17.45}$$

and estimate $\mathbf{v}$; the maximum likelihood estimate of it is the observed covariance matrix, but really we could use any consistent estimator of the covariance matrix. The closer our is to the true $\mathbf{v}$, the better our predictions. One way to think of factor analysis is that it looks for the maximum likelihood estimate, but constrained to matrices of the form $\mathbf{w}^T\mathbf{w} + \psi$.

On the plus side, the constrained estimate has a faster rate of convergence. That is, both the constrained and unconstrained estimates are consistent and will converge on their optimal, population values as we feed in more and more data, but for the same amount of data the constrained estimate is probably closer to its limiting value. In other words, the constrained estimate $\widehat{\mathbf{w}}^T\widehat{\mathbf{w}} + \widehat{\psi}$ has less variance than the unconstrained estimate $\widehat{\mathbf{v}}$.

On the minus side, maybe the true, population $\mathbf{v}$ just can't be written in the form $\mathbf{w}^T\mathbf{w} + \psi$. Then we're getting biased estimates of the covariance and the bias will *not* go away, even with infinitely many samples. Using factor analysis rather than just fitting a multivariate Gaussian means betting that either this bias is really zero, or that, with the amount of data on hand, the reduction in variance outweighs the bias.

(I haven't talked about estimation errors in the parameters of a factor model. With large samples and maximum-likelihood estimation, one could use the usual asymptotic theory. For small samples, one bootstraps as usual.)

### 17.7.1 How Many Factors?

How many factors should we use? All the tricks people use for the how-many-principal-components question can be tried here, too, with the obvious modifications. However, some other answers can also be given, using the fact that the factor model does make predictions, unlike PCA.

1. *Log-likelihood ratio tests* Sample covariances will almost never be exactly equal to population covariances. So even if the data comes from a model with $q$ factors, we can't expect the tetrad equations (or their multi-factor analogs) to

[10] A subtlety is that we might get to see some but not all of $\vec{X}$, and use that to predict the rest. Say $\vec{X} = (X_1, X_2)$, and we see $X_1$. Then we could, in principle, compute the conditional distribution of the factors, $p(F|X_1)$, and use that to predict $X_2$. Of course one could do the same thing using the correlation matrix, factor model or no factor model.

hold exactly. The question then becomes whether the observed covariances are compatible with sampling fluctuations in a $q$-factor model, or are too big for that.

We can tackle this question by using log likelihood ratio tests. The crucial observations are that a model with $q$ factors is a special case of a model with $q+1$ factors (just set a row of the weight matrix to zero), and that in the most general case, $q = p$, we can get *any* covariance matrix $\mathbf{v}$ into the form $\mathbf{w}^T\mathbf{w}$. (Set $\psi = 0$ and proceed as in the "principal factors" estimation method.)

As explained in Appendix I, if $\widehat{\theta}$ is the maximum likelihood estimate in a restricted model with $s$ parameters, and $\widehat{\Theta}$ is the MLE in a more general model with $r > s$ parameters, containing the former as a special case, and finally $\ell$ is the log-likelihood function

$$2[\ell(\widehat{\Theta}) - \ell(\widehat{\theta})] \rightsquigarrow \chi^2_{r-s} \tag{17.46}$$

when the data came from the small model. The general regularity conditions needed for this to hold apply to Gaussian factor models, so we can test whether one factor is enough, two, etc.

(Said another way, adding another factor never reduces the likelihood, but the equation tells us how much to *expect* the log-likelihood to go up when the new factor really adds nothing and is just over-fitting the noise.)

Determining $q$ by getting the smallest one without a significant result in a likelihood ratio test is fairly traditional, but statistically messy.[11] To raise a subject we'll return to, if the true $q > 1$ and all goes well, we'll be doing lots of hypothesis tests, and making sure this compound procedure works reliably is harder than controlling any one test. Perhaps more worrisomely, calculating the likelihood relies on distributional assumptions for the factor scores and the noises, which are hard to check for latent variables.

2. If you are comfortable with the distributional assumptions, use Eq. 17.43 to predict new data, and see which $q$ gives the best predictions — for comparability, the predictions should be compared in terms of the log-likelihood they assign to the testing data. If genuinely new data is not available, use cross-validation.

   Comparative prediction, and especially cross-validation, seems to be somewhat rare with factor analysis. There is no good reason why this should be so.

#### 17.7.1.1 $R^2$ and Goodness of Fit

For PCA, we saw that $R^2$ depends on the sum of the eigenvalues 16.1.3. For factor models, the natural notion of $R^2$ is the sum of squared factor loadings:

$$R^2 = \frac{\sum_{j=1}^{q}\sum_{k=1}^{p} w_{jk}^2}{p} \tag{17.47}$$

[11] Suppose $q$ is really 1, but by chance that gets rejected. Whether $q = 2$ gets rejected in turn is not an independent event!

(Remember that the factors are, by design, uncorrelated with each other, and that the entries of $\mathbf{w}$ are the correlations between factors and observables.) If we write $\mathbf{w}$ in terms of eigenvalues and eigenvectors as in §17.4.2, $\mathbf{w} = \left(\mathbf{c_q}\mathbf{d_q}^{1/2}\right)^T$, then you can show that the numerator in $R^2$ is, again, a sum of eigenvalues.

People sometimes select the number of factors by looking at how much variance they "explain" — really, how much variance is kept after smoothing on to the plane. As usual with model selection by $R^2$, there is little good to be said for this, except that it is fast and simple.

In particular, $R^2$ should not be used to assess the goodness-of-fit of a factor model. The bluntest way to see this is to simulate data which does *not* come from a factor model, fit a small number of factors, and see what $R^2$ one gets. This was done by Peterson (2000), who found that it was easy to get $R^2$ of 0.4 or 0.5, and sometimes even higher[12] The same paper surveyed values of $R^2$ from the published literature on factor models, and found that the typical value was also somewhere around 0.5; no doubt this was just a coincidence[13].

Instead of looking at $R^2$, it is much better to check goodness-of-fit by actually goodness-of-fit tests. We looked at some tests of multivariate goodness-of-fit in Chapter E. In the particular case of factor models with the Gaussian assumption, we can use a log-likelihood ratio test, checking the null hypothesis that the number of factors $= q$ against the alternative of an arbitrary multivariate Gaussian (which is the same as $p$ factors). This test is automatically performed by `factanal` in R.

If the Gaussian assumption is dubious but we want a factor model and goodness-of-fit anyway, we can look at the difference between the empirical covariance matrix $\mathbf{v}$ and the one estimated by the factor model, $\widehat{\psi} + \widehat{\mathbf{w}}^T\widehat{\mathbf{w}}$. There are several notions of distance between matrices (matrix norms) which could be used as test statistics; one could also use the sum of squared differences between the entries of $\mathbf{v}$ and those of $\widehat{\psi} + \widehat{\mathbf{w}}^T\widehat{\mathbf{w}}$. Sampling distributions would have to come from bootstrapping, where we would want to simulate from the factor model.

## 17.8 Factor Models versus PCA Once More

We began this chapter by seeking to add some noise, and some probabilistic assumptions, into PCA. The factor models we came up with are closely related to principal components, but are *not* the same. Many of the differences have been mentioned as we went, but it's worth collecting some of the most important ones here.

1. Factor models assume that the data comes from a certain distribution, IID across data points. PCA assumes nothing about distributions at all. Moreover, factor models can be used *generatively*, to say how the latent factors cause the observable variables. PCA has nothing to say about the data-generating process.

[12] See also `http://bactra.org/weblog/523.html` for a similar experiment, with (not very elegant) R code.

[13] Peterson (2000) also claims that reported values of $R^2$ for PCA are roughly equal to those of factor analysis, but by this point I hope that none of you take that as an argument in favor of PCA.

2. Factor models can be tested by their predictions on new data points; PCA cannot.
3. Factor models assume that the variance matrix of the data takes a special form, $\mathbf{w}^T\mathbf{w}+\psi$, where $\mathbf{w}$ is $q \times p$ and $\psi$ is diagonal. That is, the variance matrix must be "low rank plus noise". PCA works no matter what sample variance matrix the data might have.
4. If the factor model *is* true, then the principal components are (or approach with enough data) the eigenvectors of $\mathbf{w}^T\mathbf{w} + \psi$. They do not approach the eigenvectors of $\mathbf{w}^T\mathbf{w}$, which would be the principal factors. If the noise is small, the difference may also be small, but the factor model can be correct, if perhaps not so useful, while $\psi$ is as big as $\mathbf{w}^T\mathbf{w}$.
5. Factor models are subject to the rotation problem; PCA is not. Which one has the advantage here is unclear.
6. Similarly, a principal component is just a linear combination of the observable variables. A latent factor is another, distinct random variable. Differences in factor scores imply differences in the *expected* values of observables. Differences in projections on to principal components imply differences in realized values of observables. (It's a little like the distinction between the predicted value for the response in a linear regression, which is a combination of the covariates, and the actual value of the response.)

## 17.9 Examples in R

### 17.9.1 Example 1: Back to the US *circa* 1977

We resume looking at the properties of the US states around 1977. In §16.3, we did a principal components analysis, finding a first component that seemed to mark the distinction between the South and the rest of the country, and a second that seemed to separate big, rich states from smaller, poorer ones. Let's now subject the data to factor analysis. We begin with one factor, using the base R function `factanal`.

```
(state.fa1 <- factanal(state.x77,factors=1,scores="regression"))
##
## Call:
## factanal(x = state.x77, factors = 1, scores = "regression")
##
## Uniquenesses:
## Population     Income Illiteracy   Life Exp     Murder    HS Grad
##      0.957      0.791      0.235      0.437      0.308      0.496
##      Frost       Area
##      0.600      0.998
##
## Loadings:
##            Factor1
```

```
## Population -0.208
## Income      0.458
## Illiteracy -0.875
## Life Exp    0.750
## Murder     -0.832
## HS Grad     0.710
## Frost       0.632
## Area
##
##                Factor1
## SS loadings      3.178
## Proportion Var   0.397
##
## Test of the hypothesis that 1 factor is sufficient.
## The chi square statistic is 91.97 on 20 degrees of freedom.
## The p-value is 3.34e-11
```

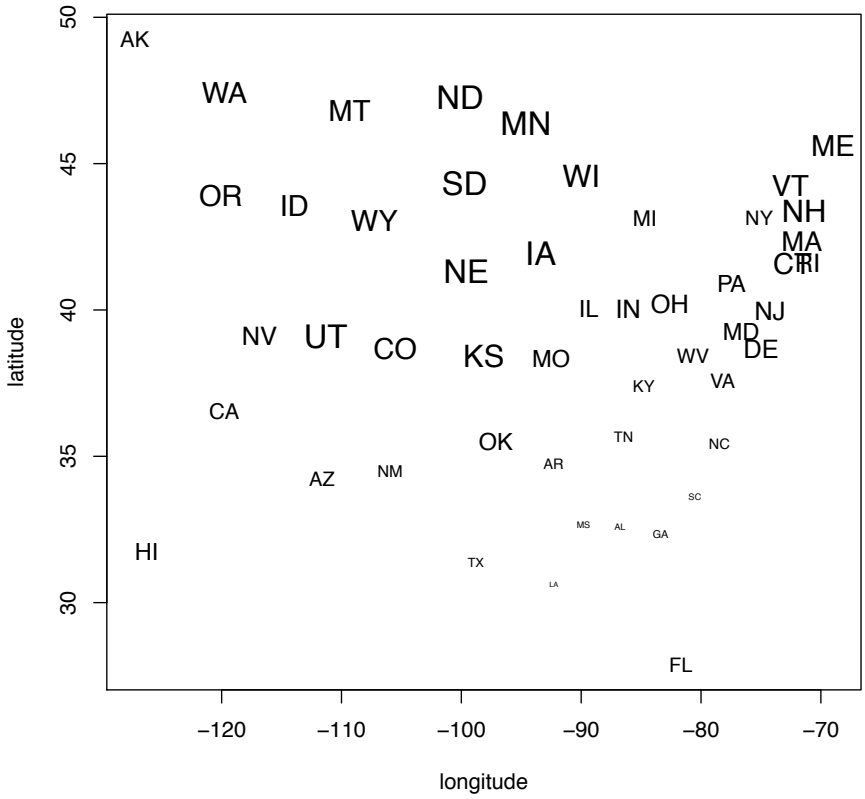
The output here tells us what fraction of the variance in each observable comes from its own noise (= the diagonal entries in $\widehat{\psi}$ = "uniquenesses"). It also gives us the factor loadings, i.e., the rows of $\widehat{\mathbf{w}}$. Here there's only one loading vector, since we set `factors` $= q = 1$. As a courtesy, the default printing method for the loadings leaves blanks where the loadings would be very small (here, for `Area`); this can be controlled through options (see `help(loadings)`). The last option picks between different methods of estimating the factor scores.

For comparison, here is the first principal component:

```
##
## Loadings:
##            PC1    PC2    PC3    PC4    PC5    PC6    PC7    PC8
## Population  0.126  0.411 -0.656 -0.409  0.406                -0.219
## Income     -0.299  0.519 -0.100        -0.638  0.462
## Illiteracy  0.468                0.353         0.387 -0.620 -0.339
## Life Exp   -0.412        -0.360  0.443  0.327  0.219 -0.256  0.527
## Murder      0.444  0.307  0.108 -0.166 -0.128 -0.325 -0.295  0.678
## HS Grad    -0.425  0.299         0.232        -0.645 -0.393 -0.307
## Frost      -0.357 -0.154  0.387 -0.619  0.217  0.213 -0.472
## Area               0.588  0.510  0.201  0.499  0.148  0.286
##
##                  PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8
## SS loadings    1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
## Proportion Var 0.125 0.125 0.125 0.125 0.125 0.125 0.125 0.125
## Cumulative Var 0.125 0.250 0.375 0.500 0.625 0.750 0.875 1.000
```

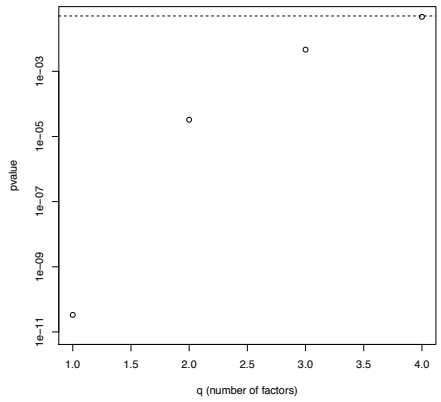
The first principal component is clearly not the same as the single common factor we extracted, even after a sign change, but it's not shockingly dissimilar either, as a map shows.

Of course, why use just one factor? Given the number of observables, we can fit up to four factors before the problem becomes totally unidentified and `factanal`




```
plot.states_scaled(state.fa1$score[,1],min.size=0.3,max.size=1.5,
                   xlab="longitude",ylab="latitude")
```

FIGURE 17.3: *The US states, plotted in position with symbols scaled by their factor scores in a one-factor model. Compare to Figure 16.5, which is where the* `plot.states_scaled` *function comes from. (Try plotting the negative of the factor scores to make the maps look more similar.)*




```
plot(1:4,pvalues,xlab="q (number of factors)", ylab="pvalue",
     log="y",ylim=c(1e-11,0.04))
abline(h=0.05,lty="dashed")
```

FIGURE 17.4: *Gaussian likelihood ratio test p-value for models with various numbers of latent factors, fit to the US-in-1977 data.*

refuses to work. That function automatically runs the likelihood ratio test every time it fits a model, assuming Gaussian distributions for the observables. As remarked, this can work reasonably well for non-Gaussian distributions if they're not too non-Gaussian, especially if $n$ is much larger than the number of parameters; of course $n = 50$ is pretty modest. Still, let's try it:

```
pvalues <- sapply(1:4,function(q){factanal(state.x77,factors=q)$PVAL})
signif(pvalues,2)
## objective objective objective objective
##   3.3e-11   3.3e-05   4.6e-03   4.7e-02
```

(Figure 17.4 plots the results.) None of the models has a $p$-value crossing the conventional 0.05 level, meaning all of them show systematic, detectable departures from what the data should look like if the factor model were true. Still, the four-factor model comes close.

Notice that the first factor's loadings do not stay the same when we add more factors, unlike the first principal component:

```
print(factanal(state.x77, factors=4)$loadings)
##
## Loadings:
##            Factor1 Factor2 Factor3 Factor4
## Population                          0.636
```

```
## Income      0.313   0.281   0.561   0.189
## Illiteracy -0.466  -0.878
## Life Exp    0.891   0.191
## Murder     -0.792  -0.384   0.109   0.405
## HS Grad     0.517   0.418   0.581
## Frost       0.128   0.679   0.105  -0.460
## Area       -0.174           0.796
##
##                Factor1 Factor2 Factor3 Factor4
## SS loadings      2.054   1.680   1.321   0.821
## Proportion Var   0.257   0.210   0.165   0.103
## Cumulative Var   0.257   0.467   0.632   0.734
```

### 17.9.2 Example 2: Stocks

Classical financial theory suggests that the log-returns of corporate stocks should be IID Gaussian random variables, but allows for the possibility that different stocks might be correlated with each other. In fact, theory suggests that the returns to any given stock should be the sum of two components: one which is specific to that firm, and one which is common to all firms. (More specifically, the common component is one which couldn't be eliminated even in a perfectly diversified portfolio.) This in turn implies that stock returns should match a one-factor model.

The rest of this example is deliberately omitted, since it will be a homework assignment for 2015.

[[ATTN: Get population coding data from neuro., and use that as an example?]]

## 17.10 Reification, and Alternatives to Factor Models

A natural impulse, when looking at something like Figure 17.1, is to **reify** the factors, and to treat the arrows **causally**: that is, to say that there really is *some* variable corresponding to each factor, and that changing the value of that variable will change the features. For instance, one might want to say that there is a real, physical variable corresponding to the factor $F_1$, and that increasing this by one standard deviation will, on average, increase $X_1$ by 0.87 standard deviations, decrease $X_2$ by 0.75 standard deviations, and do nothing to the other features. Moreover, changing any of the other factors has no effect on $X_1$.

Sometimes all this is even right. How can we tell when it's right?

### 17.10.1 The Rotation Problem Again

Consider the following matrix, call it $r$:

$$\begin{bmatrix} \cos 30 & -\sin 30 & 0 \\ \sin 30 & \cos 30 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{17.48}$$

Applied to a three-dimensional vector, this rotates it thirty degrees counter-clockwise around the vertical axis. If we apply $r$ to the factor loading matrix of the model in the figure, we get the model in Figure 17.5. Now instead of $X_1$ being correlated with

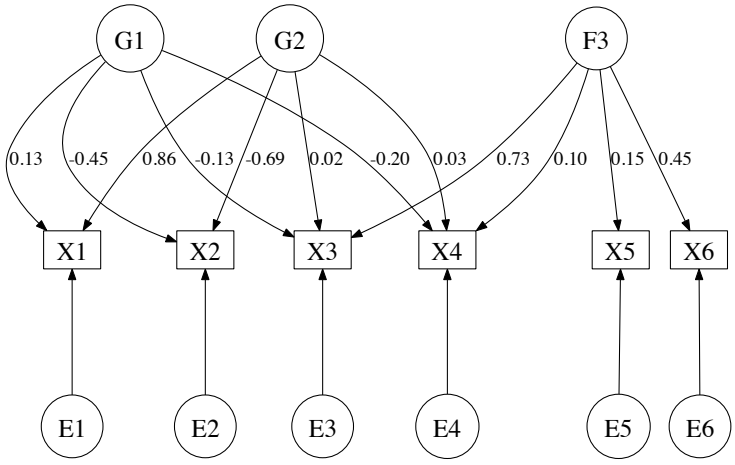



FIGURE 17.5: *The model from Figure 17.1, after rotating the first two factors by 30 degrees around the third factor's axis. The new factor loadings are rounded to two decimal places.*

the other variables only through one factor, it's correlated through two factors, and $X_4$ has incoming arrows from three factors.

Because the transformation is orthogonal, the distribution of the observations is unchanged. In particular, the fit of the new factor model to the data will be *exactly* as good as the fit of the old model. If we try to take this causally, however, we come up with a very different interpretation. The quality of the fit to the data does not, therefore, let us distinguish between these two models, and so these two stories about the causal structure of the data.

The rotation problem does not rule out the idea that checking the fit of a factor model would let us discover *how many* hidden causal variables there are.

### 17.10.2 Factors or Mixtures?

Suppose we have two distributions with probability densities $f_0(x)$ and $f_1(x)$. Then we can define a new distribution which is a **mixture** of them, with density $f_\alpha(x) = (1-\alpha)f_0(x) + \alpha f_1(x)$, $0 \leq \alpha \leq 1$. The same idea works if we combine more than two distributions, so long as the sum of the **mixing weights** sum to one (as do $\alpha$ and $1-\alpha$). Mixture models are a very flexible and useful way of representing complicated

probability distributions[14], and we will look at them in detail in Chapter 19.

I bring up mixture models here because there is a very remarkable result: any linear factor model with $q$ factors is equivalent to some mixture model with $q+1$ clusters, in the sense that the two models have the same means and covariances (Bartholomew, 1987, pp. 36–38). Recall from above that the likelihood of a factor model depends on the data only through the correlation matrix. If the data really were generated by drawing from $q+1$ clusters, then a model with $q$ factors can match the covariance matrix very well, and so get a very high likelihood. This means it will, by the usual test, seem like a very good fit. Needless to say, however, the causal interpretations of the mixture model and the factor model are very different. The two *may* be distinguishable if the clusters are well-separated (by looking to see whether the data are unimodal or not), but that's not exactly guaranteed.

All of which suggests that factor analysis can't alone really tell us whether we have $q$ continuous latent variables, or one discrete hidden variable taking $q+1$ values.

### 17.10.3 The Thomson Sampling Model

We have been working with fewer factors than we have features. Suppose that's not true. Suppose that each of our features is actually a linear combination of a *lot* of variables we don't measure:

$$X_{ij} = \eta_{ij} + \sum_{k=1}^{q} A_{ik} T_{kj} = \eta_{ij} + \vec{A}_i \cdot \vec{T}_j \tag{17.49}$$

where $q \gg p$. Suppose further that the latent variables $A_{ik}$ are totally independent of one another, but they all have mean 0 and variance 1; and that the noises $\eta_{ij}$ are independent of each other and of the $A_{ik}$, with variance $\phi_j$; and the $T_{kj}$ are independent of everything. What then is the covariance between $X_{ia}$ and $X_{ib}$? Well, because $\mathbb{E}\left[X_{ia}\right] = \mathbb{E}\left[X_{ib}\right] = 0$, it will just be the expectation of the product of the

[14] They are also a probabilistic, predictive alternative to the kind of clustering techniques you may have seen in data mining: each distribution in the mixture is basically a cluster, and the mixing weights are the probabilities of drawing a new sample from the different clusters.

features:

$$\mathbb{E}\left[X_{ia}X_{ib}\right] \tag{17.50}$$
$$= \mathbb{E}\left[(\eta_{ia} + \vec{A}_i \cdot \vec{T}_a)(\eta_{ib} + \vec{A}_i \cdot \vec{T}_b)\right] \tag{17.51}$$
$$= \mathbb{E}\left[\eta_{ia}\eta_{ib}\right] + \mathbb{E}\left[\eta_{ia}\vec{A}_i \cdot \vec{T}_b\right] + \mathbb{E}\left[\eta_{ib}\vec{A}_i \cdot \vec{T}_a\right] + \mathbb{E}\left[(\vec{A}_i \cdot \vec{T}_a)(\vec{A}_i \cdot \vec{T}_b)\right] \tag{17.52}$$
$$= 0 + 0 + 0 + \mathbb{E}\left[\left(\sum_{k=1}^{q} A_{ik}T_{ka}\right)\left(\sum_{l=1}^{q} A_{il}T_{lb}\right)\right] \tag{17.53}$$
$$= \mathbb{E}\left[\sum_{k,l} A_{ik}A_{il}T_{ka}T_{lb}\right] \tag{17.54}$$
$$= \sum_{k,l} \mathbb{E}\left[A_{ik}A_{il}\right] T_{ka}T_{lb} \tag{17.55}$$
$$= \sum_{k,l} \mathbb{E}\left[A_{ik}A_{il}\right] \mathbb{E}\left[T_{ka}T_{lb}\right] \tag{17.56}$$
$$= \sum_{k=1}^{q} \mathbb{E}\left[T_{ka}T_{kb}\right] \tag{17.57}$$

where to get the last line I use the fact that $\mathbb{E}\left[A_{ik}A_{il}\right] = 1$ if $k = l$ and $= 0$ otherwise. If the coefficients $T$ are fixed, then the last expectation goes away and we merely have the same kind of sum we've seen before, in the factor model.

Instead, however, let's say that the coefficients $T$ are themselves random (but independent of $A$ and $\eta$). For each feature $X_{ia}$, we fix a proportion $z_a$ between 0 and 1. We then set $T_{ka} \sim \text{Bernoulli}(z_a)$, with $T_{ka} \perp\!\!\!\perp T_{lb}$ unless $k = l$ and $a = b$. Then

$$\mathbb{E}\left[T_{ka}T_{kb}\right] = \mathbb{E}\left[T_{ka}\right]\mathbb{E}\left[T_{kb}\right] = z_a z_b \tag{17.58}$$

and

$$\mathbb{E}\left[X_{ia}X_{ib}\right] = q z_a z_b \tag{17.59}$$

Of course, in the one-factor model,

$$\mathbb{E}\left[X_{ia}X_{ib}\right] = w_a w_b \tag{17.60}$$

So this random-sampling model looks *exactly* like the one-factor model with factor loadings proportional to $z_a$. The tetrad equation, in particular, will hold.

Now, it doesn't make a lot of sense to imagine that every time we make an observation we change the coefficients $T$ randomly. Instead, let's suppose that they are first generated randomly, giving values $T_{kj}$, and then we generate feature values according to Eq. 17.49. The covariance between $X_{ia}$ and $X_{ib}$ will be $\sum_{k=1}^{q} T_{ka}T_{kb}$. But this is a sum of IID random values, so by the law of large numbers as $q$ gets large this will become very close to $q z_a z_b$. Thus, for nearly all choices of the coefficients, the feature covariance matrix should come very close to satisfying the tetrad equations and looking like there's a single general factor.

```
rthomson <- function(n, d, a, q, ability.mean = 0, ability.sd = 1) {
    require(MASS)
    general.per.test = sample(1:a, size = d, replace = TRUE)
    specifics.per.test = sample(1:q, size = d, replace = TRUE)
    general.to.tests = matrix(0, a, d)
    for (i in 1:d) {
        abilities = sample(1:a, size = general.per.test[i], replace = FALSE)
        general.to.tests[abilities, i] = 1
    }
    sigma = matrix(0, a, a)
    diag(sigma) = (ability.sd)^2
    mu = rep(ability.mean, a)
    x = mvrnorm(n, mu, sigma)
    general.tests = x %*% general.to.tests
    specific.tests = matrix(0, n, d)
    noisy.tests = matrix(0, n, d)
    for (i in 1:d) {
        j = specifics.per.test[i]
        specifics = rnorm(n, mean = ability.mean * j, sd = ability.sd * sqrt(j))
        specific.tests[, i] = general.tests[, i] + specifics
        noises = rnorm(n, mean = 0, sd = ability.sd)
        noisy.tests[, i] = specific.tests[, i] + noises
    }
    tm = list(data = noisy.tests, general.ability.pattern = general.to.tests,
        numbers.of.specifics = specifics.per.test, ability.matrix = x, specific.tests = specific
    return(tm)
}
```

CODE EXAMPLE 28: *Function for simulating the Thomson latent-sampling model.*

In this model, each feature is a linear combination of a *random sample* of a huge pool of completely independent features, plus some extra noise specific to the feature.[15] Precisely *because* of this, the features are correlated, and the pattern of correlations is that of a factor model with one factor. The appearance of a single common cause actually arises from the fact that the number of causes is immense, and there is no particular pattern to their influence on the features.

Code Example 28 simulates the Thomson model.

[15]When Godfrey Thomson introduced this model in 1914, he used a slightly different procedure to generate the coefficient $T_{kj}$. For each feature he drew a uniform integer between 1 and $q$, call it $q_j$, and then sampled the integers from 1 to $q$ *without replacement* until he had $q_j$ random numbers; these were the values of $k$ where $T_{kj} = 1$. This is basically similar to what I describe, setting $z_j = q_j/q$, but a bit harder to analyze in an elementary way. — Thomson (1916), the original paper, includes what we would now call a simulation study of the model, where Thomson stepped through the procedure to produce simulated data, calculate the empirical correlation matrix of the features, and check the fit to the tetrad equations. Not having a computer, Thomson generated the values of $T_{kj}$ with a deck of cards, and of the $A_{ik}$ and $\eta_{ij}$ by rolling *5220* dice.

```
tm <- rthomson(50,11,500,50)
factanal(tm$data,1)
##
## Call:
## factanal(x = tm$data, factors = 1)
##
## Uniquenesses:
##  [1] 0.169 0.816 0.368 0.458 0.537 0.332 0.378 0.758 0.757 0.999 0.844
##
## Loadings:
##       Factor1
##  [1,] 0.911
##  [2,] 0.428
##  [3,] 0.795
##  [4,] 0.737
##  [5,] 0.680
##  [6,] 0.817
##  [7,] 0.789
##  [8,] 0.492
##  [9,] 0.493
## [10,]
## [11,] 0.394
##
##                Factor1
## SS loadings      4.582
## Proportion Var   0.417
##
## Test of the hypothesis that 1 factor is sufficient.
## The chi square statistic is 50.17 on 44 degrees of freedom.
## The p-value is 0.242
```

The first command generates data from $n = 50$ items with $p = 11$ features and $q = 500$ latent variables. (The last argument controls the average size of the specific variances $\phi_j$.) The result of the factor analysis is of course variable, depending on the random draws; this attempt gave the proportion of variance associated with the factor as 0.42, and the $p$-value as 0.24. Repeating the simulation many times, one sees that the $p$-value is pretty close to uniformly distributed, which is what it should be if the null hypothesis is true (Figure 17.6). For fixed $n$, the distribution becomes closer to uniform the larger we make $q$. In other words, the goodness-of-fit test has little or no power against the alternative of the Thomson model.

Modifying the Thomson model to look like multiple factors grows notationally cumbersome; the basic idea however is to use multiple pools of independently-sampled latent variables, and sum them:

$$X_{ij} = \eta_{ij} + \sum_{k=1}^{q_1} A_{ik} T_{kj} + \sum_{k=1}^{q_2} B_{ik} R_{kj} + \ldots \tag{17.61}$$

where the $T_{kj}$ coefficients are uncorrelated with the $R_{kj}$, and so forth. In expectation,

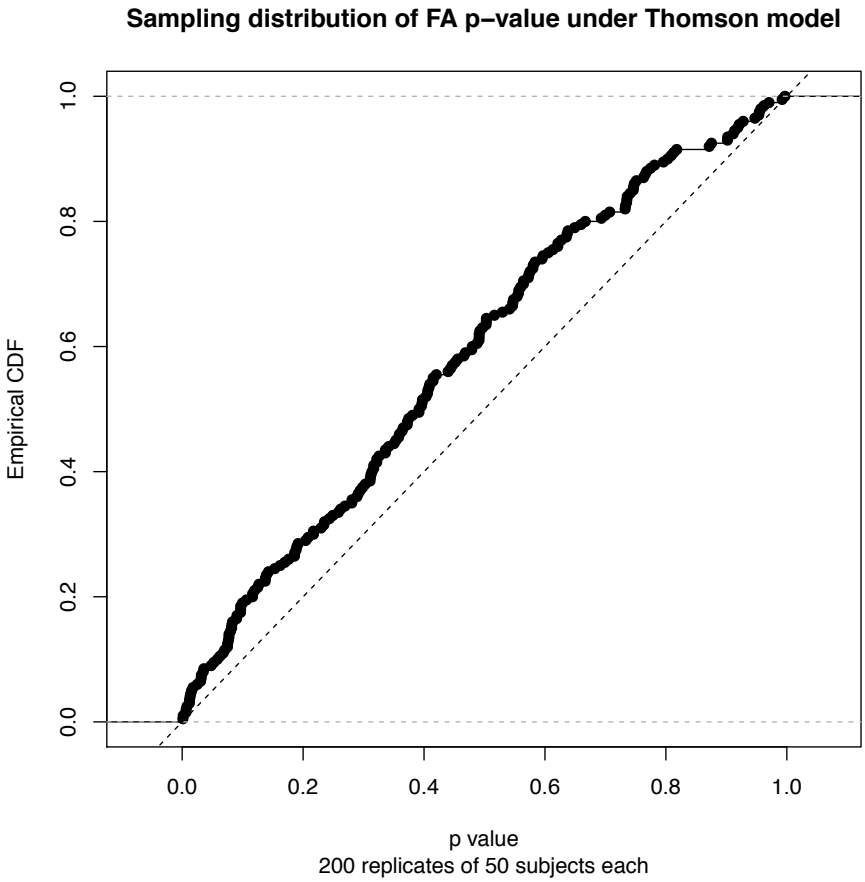



FIGURE 17.6: *Mimcry of the one-factor model by the Thomson model. The Thomson model was simulated 200 times with the parameters given above; each time, the simulated data was then fit to a factor model with one factor, and the $p$-value of the goodness-of-fit test extracted. The plot shows the empirical cumulative distribution function of the $p$-values. If the null hypothesis were exactly true, then $p \sim \text{Unif}(0,1)$, and the theoretical CDF would be the diagonal line (dashed).*

if there are $r$ such pools, this *exactly* matches the factor model with $r$ factors, and any particular realization is overwhelmingly likely to match if the $q_1, q_2, \ldots q_r$ are large enough.[16]

It's not feasible to estimate the $T$ of the Thomson model in the same way that we estimate factor loadings, because $q > p$. This is not the point of considering the model, which is rather to make it clear that we actually learn very little about where the data come from when we learn that a factor model fits well. It could mean that the features arise from combining a small number of factors, or on the contrary from combining a huge number of factors in a random fashion. A lot of the time the latter is a more plausible-sounding story.

For example, a common application of factor analysis is in marketing: you survey consumers and ask them to rate a bunch of products on a range of features, and then do factor analysis to find attributes which summarize the features. That's fine, but it may well be that each of the features is influenced by lots of aspects of the product you don't include in your survey, and the correlations are really explained by different features being affected by many of the same small aspects of the product. Similarly for psychological testing: answering any question is really a pretty complicated process involving lots of small processes and skills (of perception, several kinds of memory, problem-solving, attention, motivation, etc.), which overlap partially from question to question.

## 17.11 Further Reading

The classical papers by Spearman (1904) and Thurstone (1934) are readily available online, and very much worth reading for getting a sense of the problems which motivated the introduction of factor analysis, and the skill with which the founders grappled with them. Loehlin (1992) is a decent textbook intended for psychologists; the presumably mathematical and statistical level is decidedly lower than that of this book, but it's still useful. Thomson (1939) remains one of the most *insightful* books on factor analysis, though obviously there have been a lot of technical refinements since he wrote. It's strongly recommended for anyone who plans to make much use of the method. While out of print, used copies are reasonably plentiful and cheap, and at least one edition is free online.

On purely statistical issues related to factor analysis, Bartholomew (1987) is by far the best reference I have found; it quite properly sets it in the broader context of latent variable models, including the sort of latent class models we will explore in Chapter 19. The computational advice of that edition is, necessarily, now quite obsolete; there is an updated edition from 2011, which I have not been able to consult by the time of writing.

The use of factor analysis in psychological testing has given rise to a large controversial literature, full of claims, counter-claims, counter-counter-claims, and so on *ad nauseam*. Without, here, going into that, I will just note that to the extent the

[16] A recent paper on the Thomson model (Bartholomew *et al.*, 2009) proposes just this modification to multiple factors and to Bernoulli sampling. However, I proposed this independently, in the fall 2008 version of these notes, about a year before their paper.

strongest arguments against (say) reifying the general factor extracted from intelligence tests as "general intelligence" are good arguments, they do not just apply to intelligence, but also to personality tests, and indeed to many procedures outside psychology. In other words, if there's a problem, it's not *just* a problem for intelligence testing alone, or even for psychology alone. On this, see Glymour (1998) and Borsboom (2005, 2006).

## Exercises

1. Prove Eq. 17.13.
2. Why is it fallacious to go from "the data have the kind of correlations predicted by a one-factor model" to "the data were generated by a one-factor model"?
3. Consider Figure 17.2 and its code. What is $\mathbf{w}$ here? What is $\psi$? What is the implied covariance matrix of $\vec{X}$? .
4. Show that the correlation between the $j^{\text{th}}$ feature and $G$, in the one-factor model, is $w_j$.
5. Check that Eq. 17.11 and Eq. 17.25 are compatible.
6. Find the weights $b_{ij}$ for the Thomson estimator of factor scores (Eq. 17.38), assuming you know $\mathbf{w}$. Do you need to assume a Gaussian distribution?