[8] Big Data: Factor Models

Matt Taddy, University of Chicago Booth School of Business

faculty.chicagobooth.edu/matt.taddy/teaching

Today is all about Dimension Reduction (more than normal)

The setting: we have a high-dimensional matrix of data **X**.
We'd like to reduce this to a function few 'important' factors.

We'll do this by building a simple linear model for **X** and
use this model to represent **X** in a lower dimensional space.

Factor modeling is a super useful framework, whether you get a deep understanding or just learn how they work in practice.
We'll cover a variety of ways to understand.

Factor Models are parsimonious models for X

A factor model is regression for multivariate $\mathbf{x} = [x_1 \ldots x_p]$.

$$\mathbb{E}[x_{ij}] = \varphi_{j1} v_{i1} + \ldots \varphi_{jK} v_{iK}, \quad i = 1..n, \; j = 1..p$$

The v_{ik}'s are shared by all p dimensions of $\mathbf{x}_i$,
and the φ_{jk} are the same for all n observations.

For example

single factor: $\mathbb{E}[\mathbf{x}_i] = \varphi v_i$

two factors: $\mathbb{E}[\mathbf{x}_i] = \varphi_1 v_{i1} + \varphi_2 v_{i2}$

The φ_{jk} coefficients are called 'loadings' or 'rotations'.
They are just coefficients for regression of $\mathbf{x}_i$ onto $\mathbf{v}_i$.

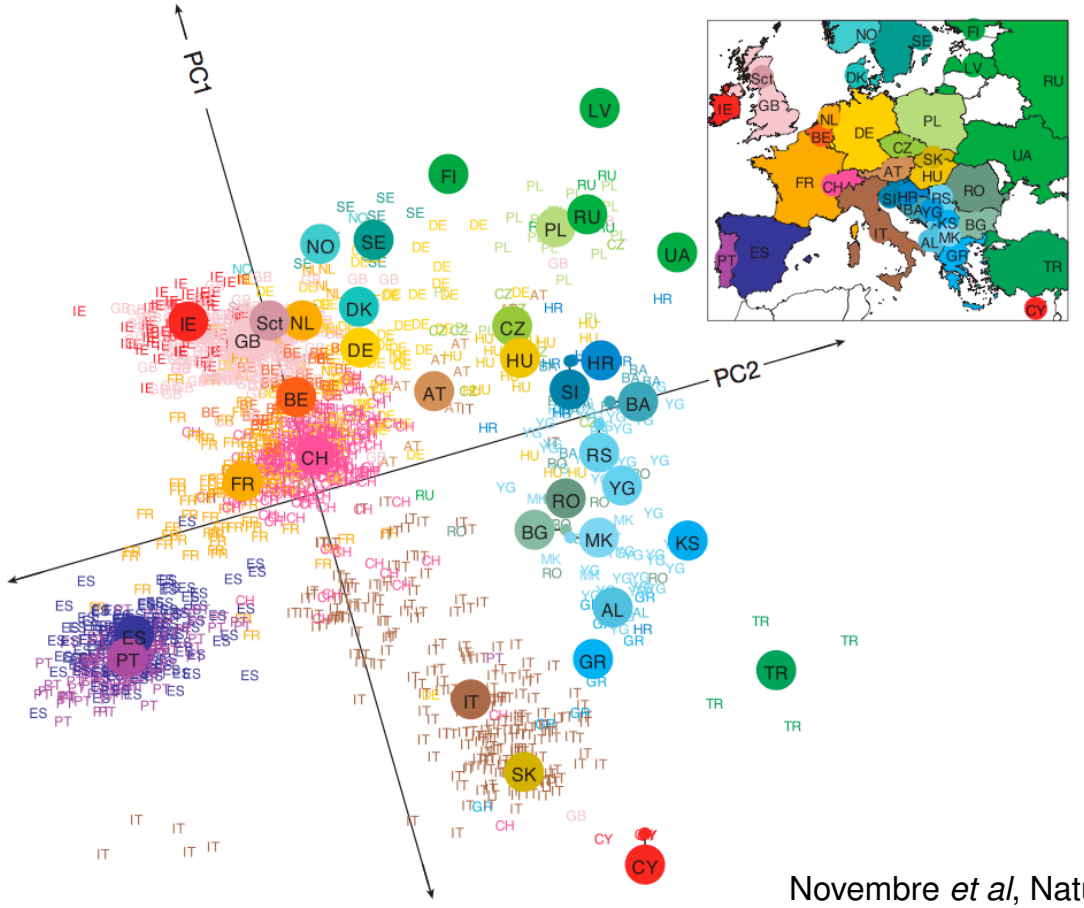
Factor Models: $\mathbb{E}[\mathbf{x}_i] = \varphi_1 v_{i1} + \ldots \varphi_K v_{iK}$

Each observation has K factors $v_1 \ldots v_K$. Since usually $K < p$, these factors are a lower dimension simplification of $\mathbf{x}$.

Each factor v_{ik} is a univariate variable, and $\mathbf{v}_i = [v_{i1} \ldots v_{iK}]$. The loadings are p-dimensional, and they translate from the simple (length-K) factor $\mathbf{v}_i$ to the complex (length-p) $\mathbf{x}_i$.

You can either treat the factors as unknown (PCA) or use y to build them (PLS).

Geo-Genes Example: two interpretable factors.



Novembre *et al*, Nature 456 (2008)

The **x** for each individual is a giant vector of SNPs. They've reduced it into two factors that explain most of the variation.

Turns out that location in this 2-D space looks geographical.

Mixture vs Factor models

Factor models imply *mixed membership*.

You don't have to be *in* one component, but can be a mix of shared latent factors.

For example, for protein consumption
Greece could be similar to Italy in some dimensions, closer to Turkey in others.

So topic models were actually factor models!

Topic Models: factors for text

Recall our PCA factor model: $\mathbb{E}[\mathbf{x}_i] = \varphi_1 v_{i1} + \ldots \varphi_K v_{iK}$.

The bag-of-words version is a topic model

$$\mathbf{x}_i \sim \text{MN}(\omega_{i1}\boldsymbol{\theta}_1 + \ldots + \omega_{iK}\boldsymbol{\theta}_K, m)$$

$\Rightarrow \mathbb{E}[\mathbf{x}_i / m_i] = \omega_{i1}\boldsymbol{\theta}_1 + \ldots + \omega_{iK}\boldsymbol{\theta}_K$,
so that ω_{ik} is like v_{ik} and $\boldsymbol{\theta}_k$ is like φ_k.

The basic interpretation is exactly the same as in PCA:

ω is a low-dimension version of $\mathbf{x}$

A greedy algorithm

Suppose you want to find the first set of factors, $\mathbf{v}^1 = v_{11} \dots v_{n1}$.
We can write our system of equations (for $i = 1 \dots n$)

$$\mathbb{E}[x_{i1}|v_{i1}] = \varphi_{11} v_{i1}$$
$$\vdots$$
$$\mathbb{E}[x_{ip}|v_{i1}] = \varphi_{1p} v_{i1}$$

Where $\sum_{j=1}^{p} \varphi_{kj}^2 = 1$ to nail down scale.

The factors $\mathbf{v}^1$ are fit to maximize average R^2 in this equation.
Note that we're optimizing over both φ *and* $\mathbf{v}$.

Next, calculate residuals $e_{ij} = x_{ij} - \mathbb{E}[x_{ij}|v_i]$.
Then find $\mathbf{v}^2$ to maximize R^2 for these residuals.

This repeats until residuals are all zero ($\min(p, n)$ steps).
This is not actually done in practice, but the intuition is solid.

A greedy algorithm for PCA

We're repeatedly finding *latent* v_{ik} to *minimize deviance*, across dimensions $j = 1 \dots p$ and observations $i = 1 \dots n$, for the model

$$e_{kij} \sim \mathrm{N}\left(v_{ik}\varphi_{kj},\ \sigma_k^2\right)$$

where $e_{kij} = x_{ij} - \mathbb{E}[x_{ij}|v_{i1} \dots v_{ik-1}]$ is the residual after $k-1$ factors, and we're optimizing over both φ_k *and* $\mathbf{v}^k$.

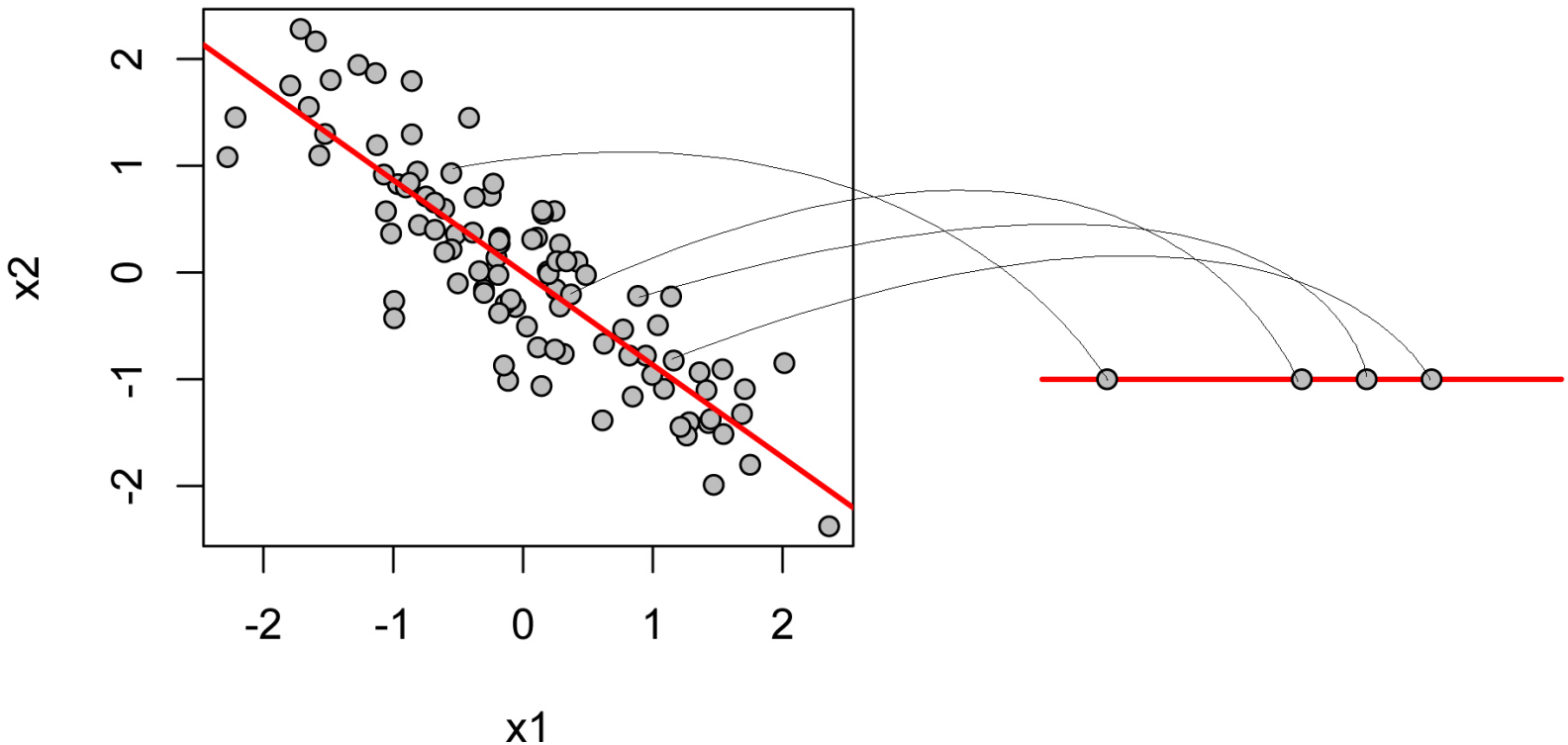
This yields principal component (PC) rotations $\mathbf{\Phi} = [\varphi_1 \cdots \varphi_K]$, with K the minimum of n and p (i.e., where all $e_{ijK} = 0$).

Think about the algorithm as repeatedly fitting regression onto the best possible factors to explain current residuals.

PCA: projections into latent space

Another way to think about factors, in our 2D example:

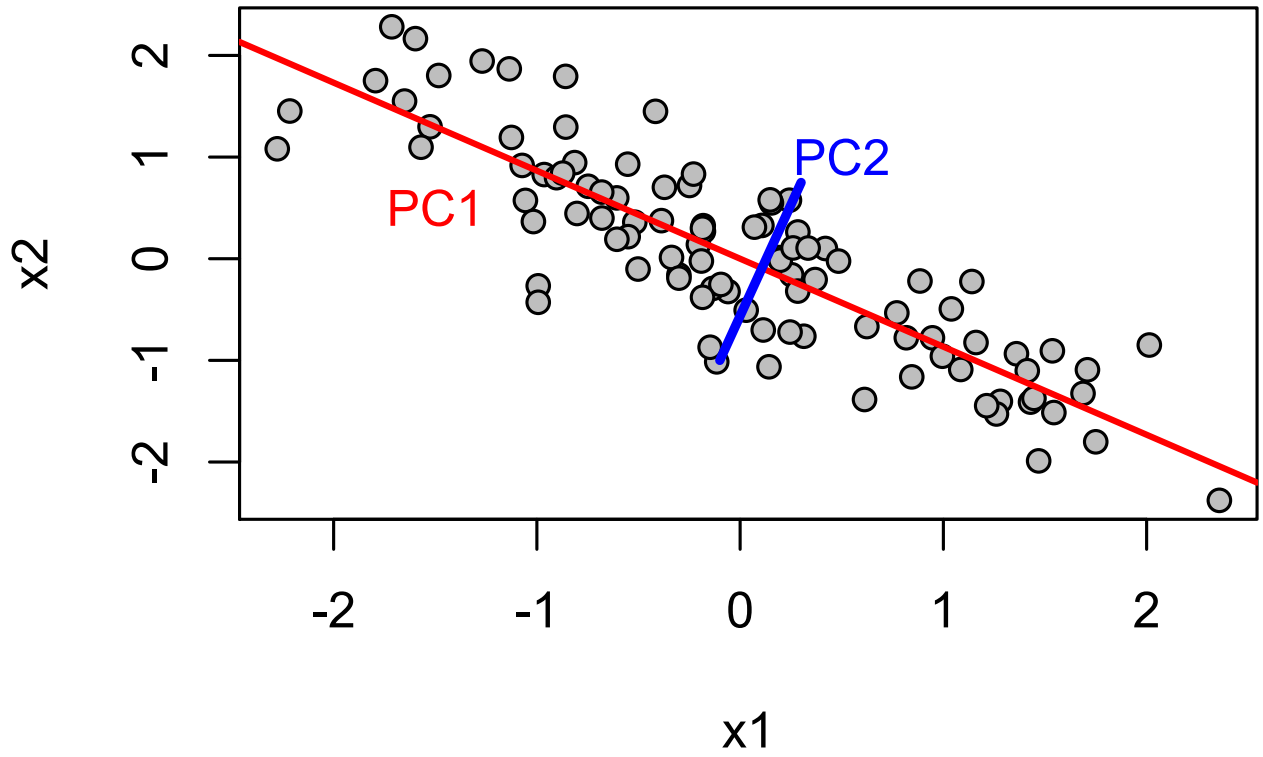
PCA equivalent to finding the line that fits through x_1 and x_2, and seeing where each observation lands (projects) on the line.



We've projected from 2D onto a 1D axis.

Fitting Principal Components via Least Squares

PCA looks for high-variance projections from multivariate **x** (i.e., the long direction) and finds the least squares fit.



Components are ordered by variance of the fitted projection.

PCA: Principal Components Analysis

PCA fits $\mathbb{E}[x_{ij}] = \varphi_{j1} v_{i1} + \ldots \varphi_{jK} v_{iK},\ j = 1...p$
by minimizing deviance over *both* φ's and $\mathbf{v}$'s.

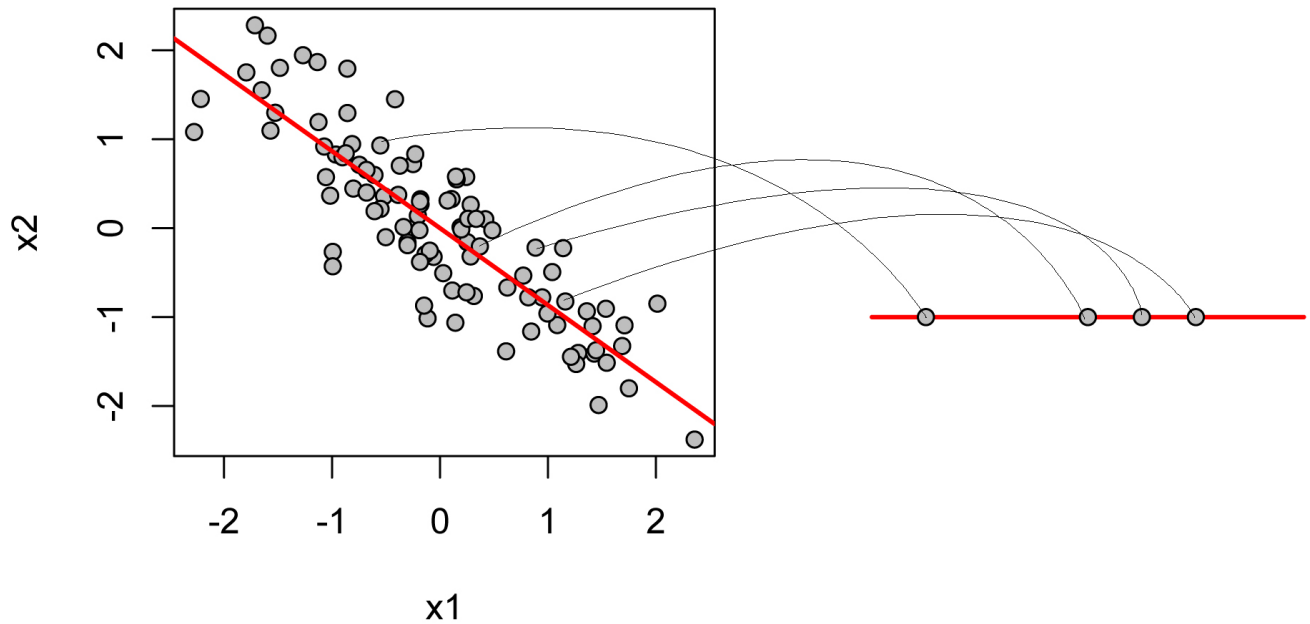
The k^{th} principal component direction for observation i is

$$z_{ki} = \mathbf{x}_i' \varphi_k = \sum_{j=1}^{p} \varphi_{kj} x_{ij}$$

This is a projection from $\mathbf{x}$ into the space of the unknown v_{ik}.
$\mathbf{\Phi} = [\varphi_1 ... \varphi_K]$ are called rotations, and we write $\mathbf{z}_i = \mathbf{x}_i' \mathbf{\Phi}$.

We say $x_{ij}\varphi_{jk}$ is a projection of x_{ij} in the direction of v_{ik}.
How does this help us simplify multivariate $\mathbf{x} = [x_1 \dots x_p]$?

If x_1 and x_2 are both linear functions of v, then view the projection from both of them into v as a single summary.



Instead of bothering with both x_1 and x_2, we have a single factor that underlies both of them: $z_i = x_{i1}\varphi_1 + x_{i2}\varphi_2$.
(z is the point on the line closest to $[x_{i1}, x_{i2}]$).

Beyond geometry, the easiest interpretation is to think of each principal component z_{ki} as our best guess at v_{ik} for new $\mathbf{x}_i$.

This occurs because of the $\sum_{j=1}^{p} \varphi_{kj}^2 = 1$, so $z_i = \sum_{j=1}^{p} \varphi_{kj} x_{ij}$ is like a sum of the portions of each x_{ij} related to v_{ik}.

Just think of $z_{ik} = \hat{v}_{ik}$, as an estimate for v_{ik}.

We work with 'directions' $\mathbf{z}$ rather than 'factors' $\mathbf{v}$ because this makes it easier to get PCs for new $\mathbf{x}$: just set $\mathbf{z} = \mathbf{x}'\mathbf{\Phi}$.

Principal Components in R

The best command to do PC is `prcomp(x, scale=TRUE)`.

- There are lots of other options; I've found prcomp solid.
- Since we're combining least squares, it is once again good to `scale` all the x_j's to have unit variance.
- `plot(mypca)` will produce a simple screeplot.

This finds the rotations $\varphi_1 \ldots \varphi_p$, also called 'loadings'.

Use `predict` to access the principal components:

`predict(mypca)[,1:2]` gives the first two.

`predict(mypca, newdata=newX)` gives **z** for new data.

Both of these function just multiply $\mathbf{x}'\mathbf{\Phi}$ for input vector **x**.

NB: **x** is first *scaled* by the SDs used in `mypca` if `scale=TRUE`.

Understanding Principal Components

Given (say) 4 factors, country 'i' has expected protein consumption $v_{i1}\varphi_1 + v_{i2}\varphi_2 + v_{i3}\varphi_3 + v_{i4}\varphi_4$.

We can interpret each v_{ik} as a 'diet factor': a way of eating.

Rotations $\varphi_k = [\varphi_{k,rm} \cdots \varphi_{k,fv}]$ (*rm*: red meat, *fv*: fr/veg) thus encode the *foods* associated with diet k.

The k^{th} principal component for each country

$$z_{ik} = \sum_{j=1}^{p} x_{ij}\varphi_{kj} = x_{i,rm}\varphi_{k,rm} + \ldots + x_{i,fv}\varphi_{k,fv}$$

is then a score for how much i's consumption matches diet k.

loadings as diet: use rotations on interpretable x_j to build a story for the latent factor space. This can be tough!

Note that if you fit `prcomp` with `scale=TRUE`, the `rotation` matrix is on the scale of standard deviations:

φ_{jk} is units of direction z_k gained for a 1 sd increase in x_j.

These 'units' are not easily interpretable, but we try.

Rotations (φ_k) for the first two food factors:

```
> t(round(pcfood$rotation[,1:2],2))
   R.Meat W.Meat  Eggs  Milk  Fish Cereal Starch Nuts Fr.Veg
PC1  -0.30  -0.31 -0.43 -0.38 -0.14   0.44  -0.30 0.42   0.11
PC2  -0.06  -0.24 -0.04 -0.18  0.65  -0.23   0.35 0.14   0.54
```

PC1 is high nut/grain, low meat/dairy.

PC2 is Iberian: move 0.65 in that direction with 1sd extra fish.

Understanding the principal components

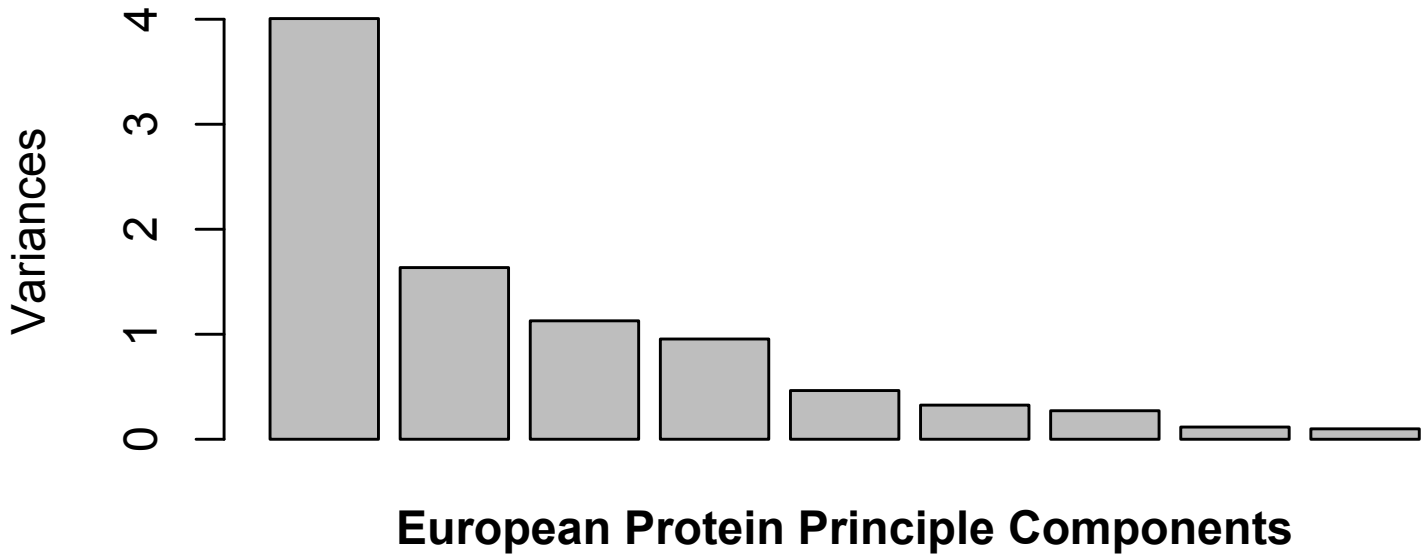
How many do we need? What do the factors contribute?

```
> summary(pcfood)
Importance of components:
                          PC1    PC2    PC3    PC4     PC5
Standard deviation     2.0016 1.2787 1.0620 0.9771 0.68106
Proportion of Variance 0.4452 0.1817 0.1253 0.1061 0.05154
Cumulative Proportion  0.4452 0.6268 0.7521 0.8582 0.90976
```

The summary tells us what `cumulative proportion` of variation is explained by the factors 1...K.

Each PC's contribution to this is decreasing with its variance.

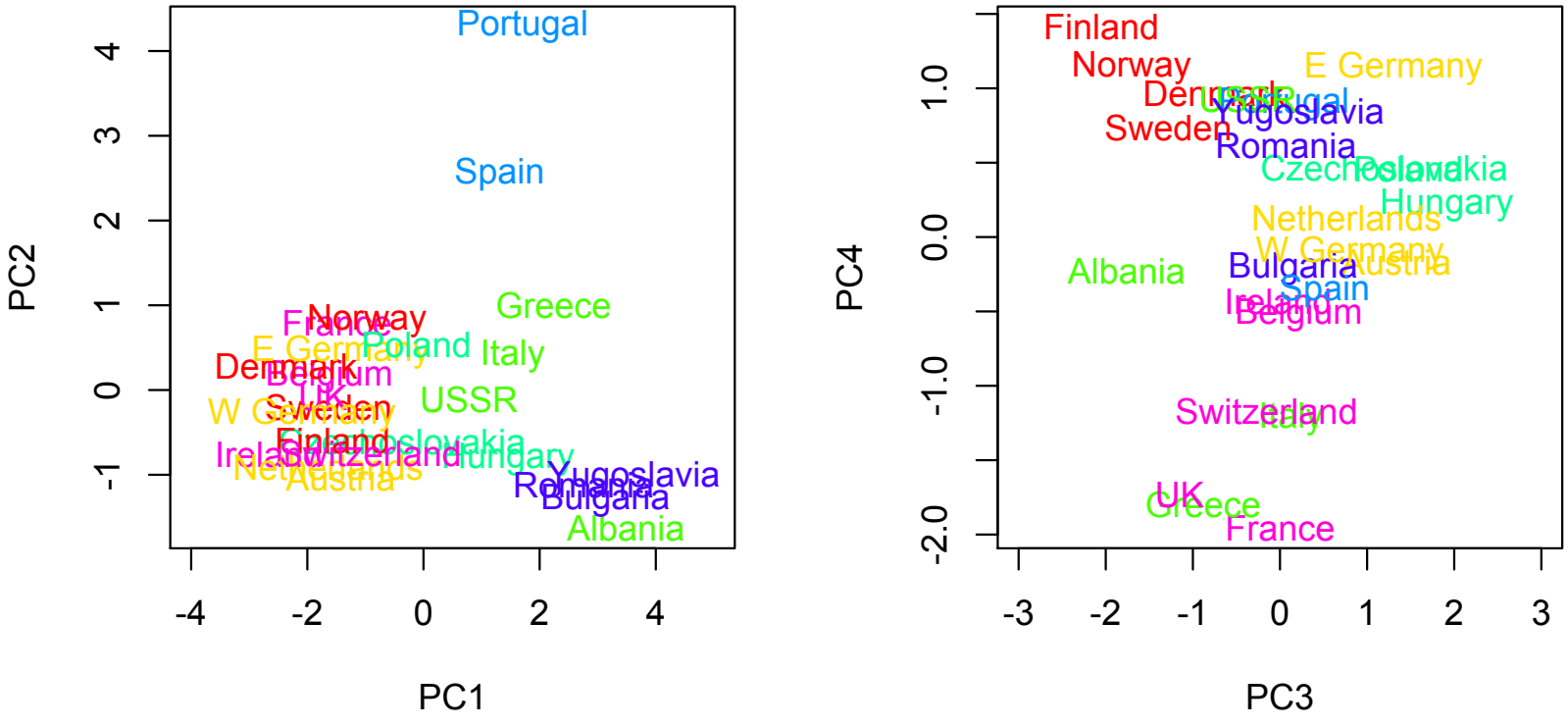
Screeplot: show variance for each principal component.



PC with high var(z_k) are useful to differentiate observations.
The directions are uninteresting once variance levels out, and
After a subjective threshold, we consider the PC "just noise".

Screeplots are heavily used, but barely useful. Here, it actually looks to me like only the first really matters. However...

Principal components in European protein consumption



Overlaying k-means clustering from week 5, we see that the nation-groups are far apart in the first 4 PC directions.

Like in any other purely unsupervised model, the goal is exploration and intuition. So use a *K* that makes sense to you.

Congress and Roll Call Voting

Votes in which names and positions are recorded are called ‘roll calls’.

The site `voteview.com` archives vote records and the R package `pscl` has tools for this data.

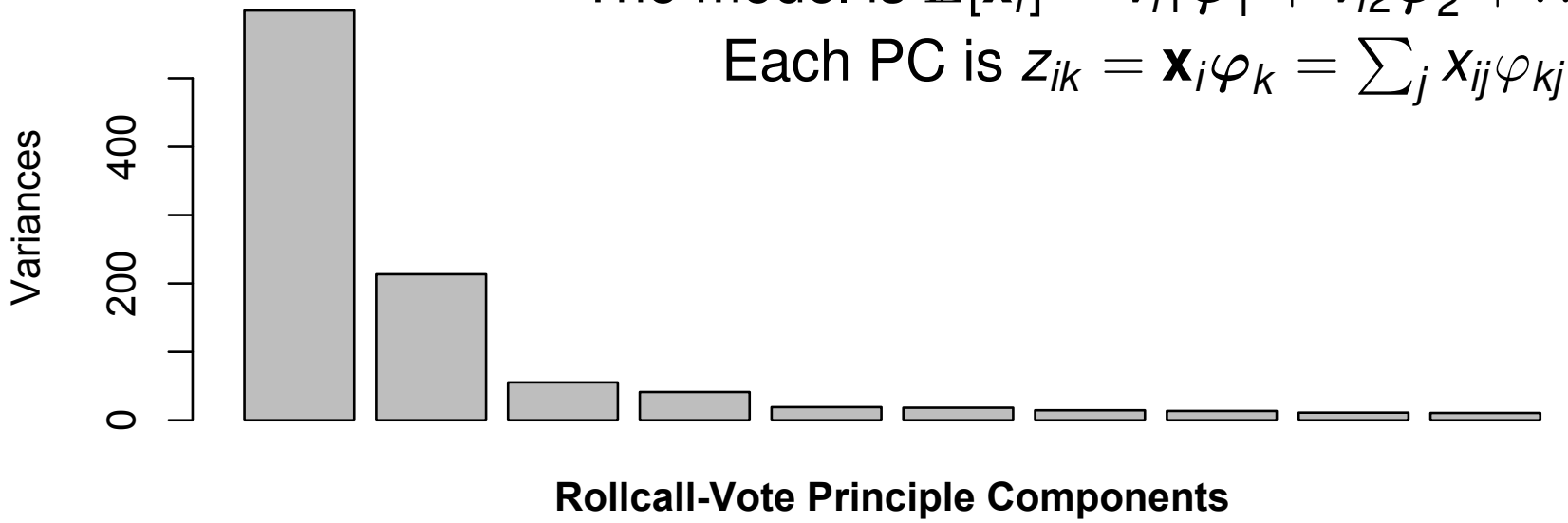
445 members in the last US House (the 111^{th})
1647 votes: nea = -1, yea=+1, missing = 0.

This leads to a large matrix of observations that can probably be reduced to simple factors (party).

Vote components in the 111^{th} house

The model is $\mathbb{E}[\mathbf{x}_i] = v_{i1}\varphi_1 + v_{i2}\varphi_2 + \ldots$

Each PC is $z_{ik} = \mathbf{x}_i\varphi_k = \sum_j x_{ij}\varphi_{kj}$

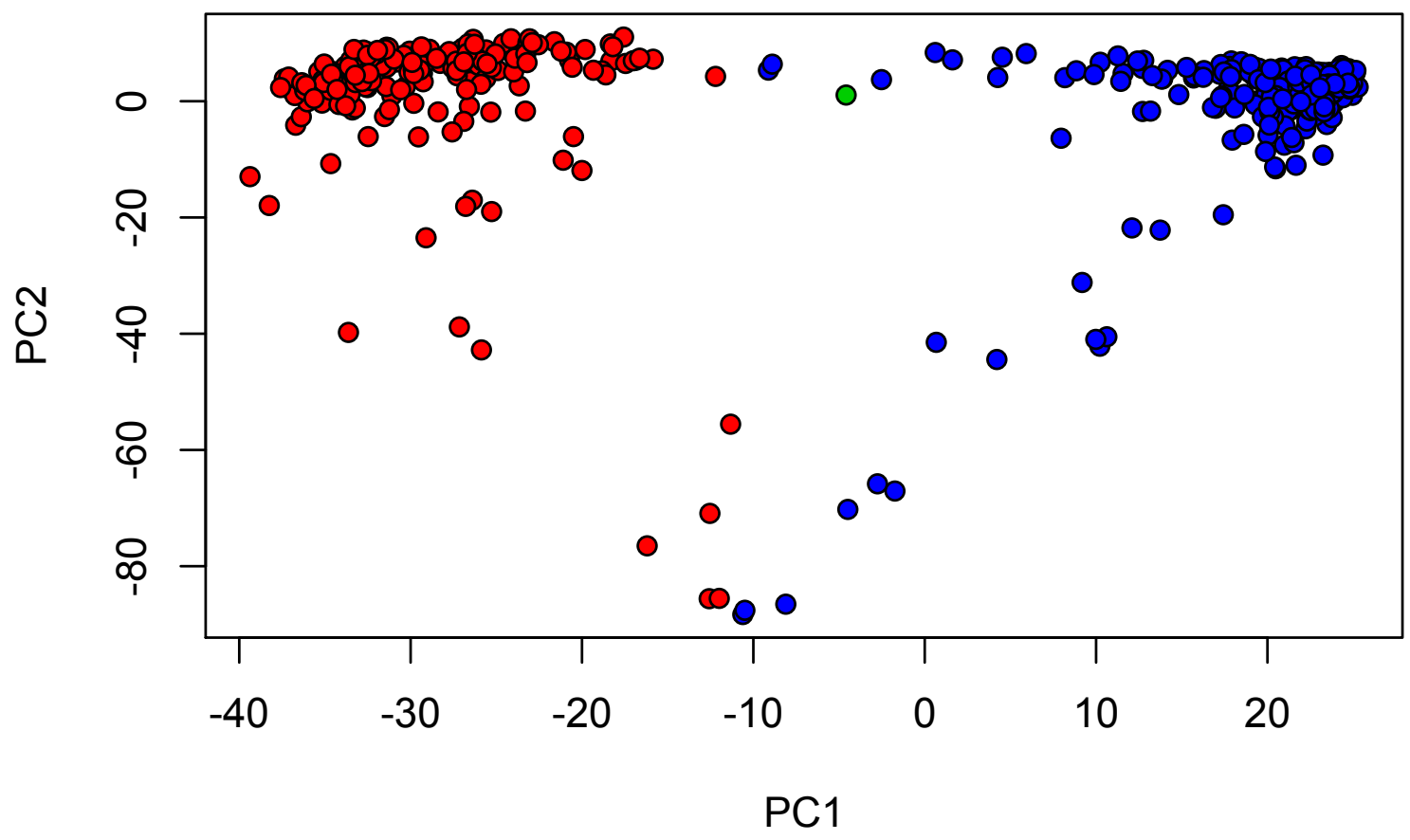


Huge drop in variance from 1^{st} to 2^{nd} and 2^{nd} to 3^{rd} PC.

Poli-Sci holds that PC1 is usually enough to explain congress.

2nd component has been important twice: 1860's and 1960's.

Top two PC directions in the 111^{th} house



Republicans in red and Democrats in blue:

- Clear separation on the first principal component.
- The second component looks orthogonal to party.

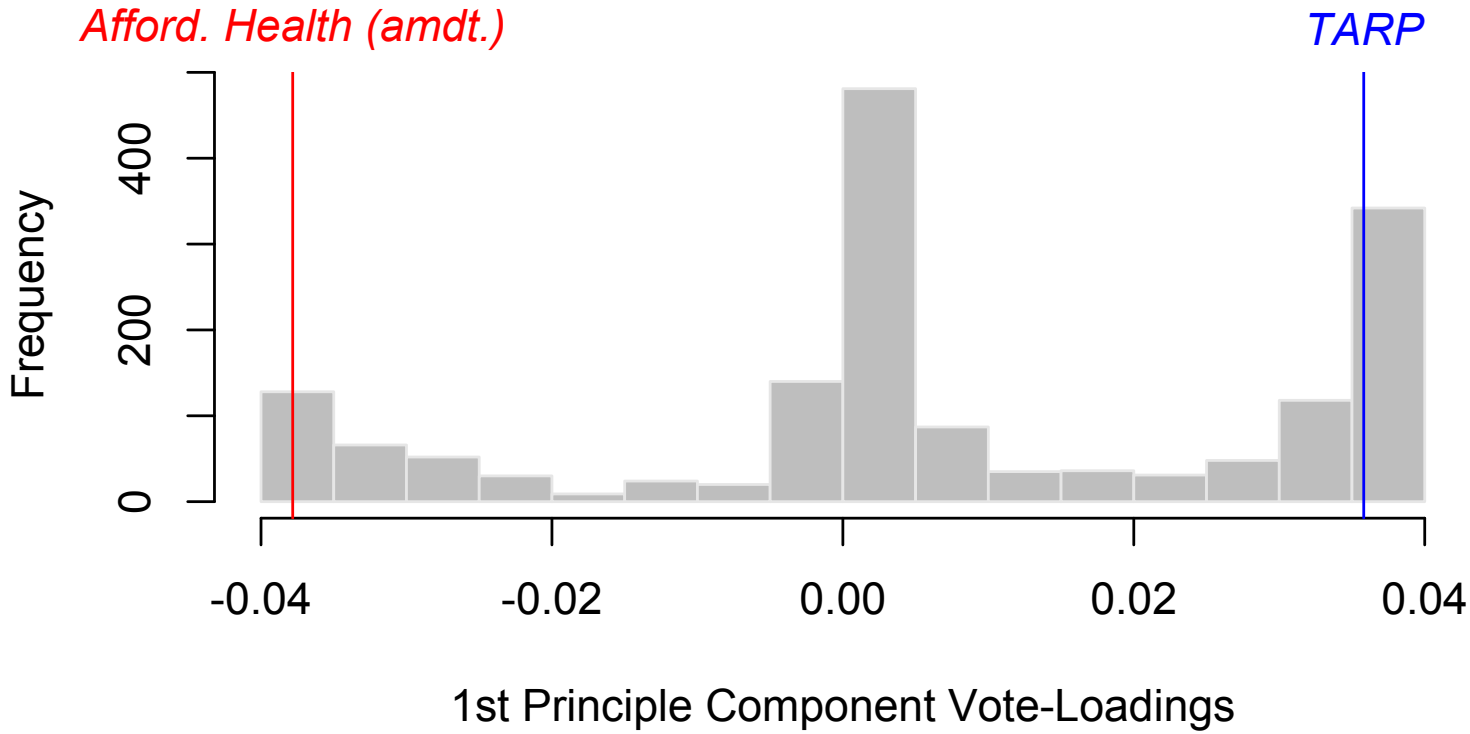
Interpreting the principal components

```
## Far right (very conservative)
> sort(votepc[,1])
     BROUN (R GA-10)       FLAKE (R AZ-6)   HENSARLIN (R TX-5)
         -39.3739409          -38.2506713          -37.5870597

## Far left (very liberal)
> sort(votepc[,1], decreasing=TRUE)
    EDWARDS (D MD-4)   PRICE (D NC-4)    MATSUI (D CA-5)
         25.2915083        25.1591151         25.1248117

## social issues?  immigration?  no clear pattern
> sort(votepc[,2])
     SOLIS (D CA-32) GILLIBRAND (D NY-20)      PELOSI (D CA-8)
        -88.31350926         -87.58871687         -86.53585568
   STUTZMAN (R IN-3)       REED (R NY-29)      GRAVES (R GA-9)
        -85.59217310         -85.53636319         -76.49658108
```

PC1 is easy to read, PC2 is ambiguous (is it even meaningful?)

High PC1-loading votes are ideological battles.

These tend to have informative voting across party lines.



A vote for Republican amendments to 'Affordable Health Care for America' strongly indicates a negative PC1 (more conservative), while a vote for TARP indicates a positive PC1 (more progressive).

Look at the largest loadings in φ_2 to discern an interpretation.

```
> loadings[order(abs(loadings[,2]), decreasing=TRUE)[1:5],2]
 Vote.1146   Vote.658  Vote.1090  Vote.1104  Vote.1149
0.05605862 0.05461947 0.05300806 0.05168382 0.05155729
```

These votes all correspond to near-unanimous symbolic action.

For example, 429 legislators voted for resolution 1146:
'Supporting the goals and ideals of a Cold War Veterans Day'

If you didn't vote for this, you weren't in the house.

Mystery Solved: the second PC is just attendance!

```
> sort(rowSums(votes==0), decreasing=TRUE)
     SOLIS (D CA-32) GILLIBRAND (D NY-20)        REED (R NY-29)
                1628                  1619                  1562
  STUTZMAN (R IN-3)       PELOSI (D CA-8)      GRAVES (R GA-9)
                1557                  1541                  1340
```

PCR: Principal Component Regression

The concept is very simple: instead of regressing onto **x**, use a lower dimension set of principal components **z** as covariates.

This works well for a few reasons:

- PCA reduces dimension, which is always good.
- Higher variance covariates are good in regression, and we choose the top PCs to have highest variance.
- The PCs are independent: no multicollinearity.

The 2-stage algorithm is straightforward. For example,

```
mypca = prcomp(X, scale=TRUE)
z = predict(mypca)[,1:K]
reg = glm(y~., data=as.data.frame(z))
```

For new data, `znew = predict(mypca,xnew)[,1:K]`

`pred = predict(reg,as.data.frame(znew))`.

Data from NBC on response to TV pilots

6241 views and 20 questions for 40 shows

Primary goal is predicting engagement.

Classic measures of broadcast marketability are Ratings.

GRP: gross ratings points; estimated total viewership.

TRP: targeted ratings points; viewership in specific categories.

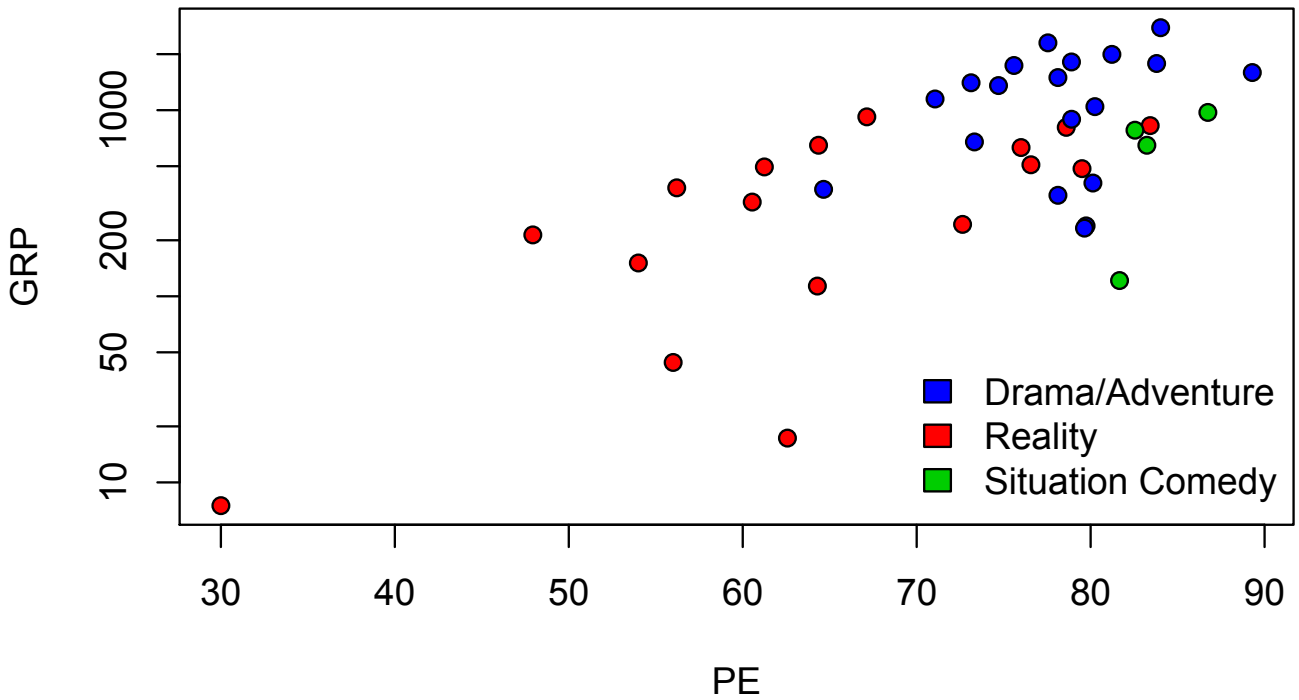
Projected Engagement: a more subtle measure of audience.

After watching a show, viewer is quized on order and detail.

This measures their engagement with the show (and ads!).

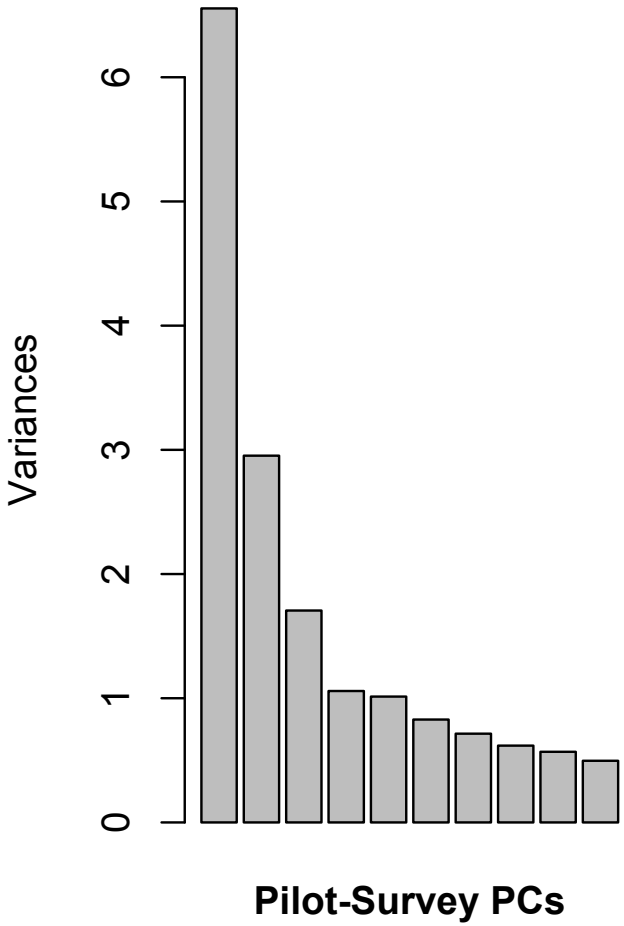
Predicting TV Engagement with PCR

Engagement matters for GRP, and also in adjusted GRP/PE.



Given the survey responses and eventual projected engagement (PE), can we find a low-D model for predicting engagement from survey response in pilot focus groups?

NBC Pilot-Survey PCA

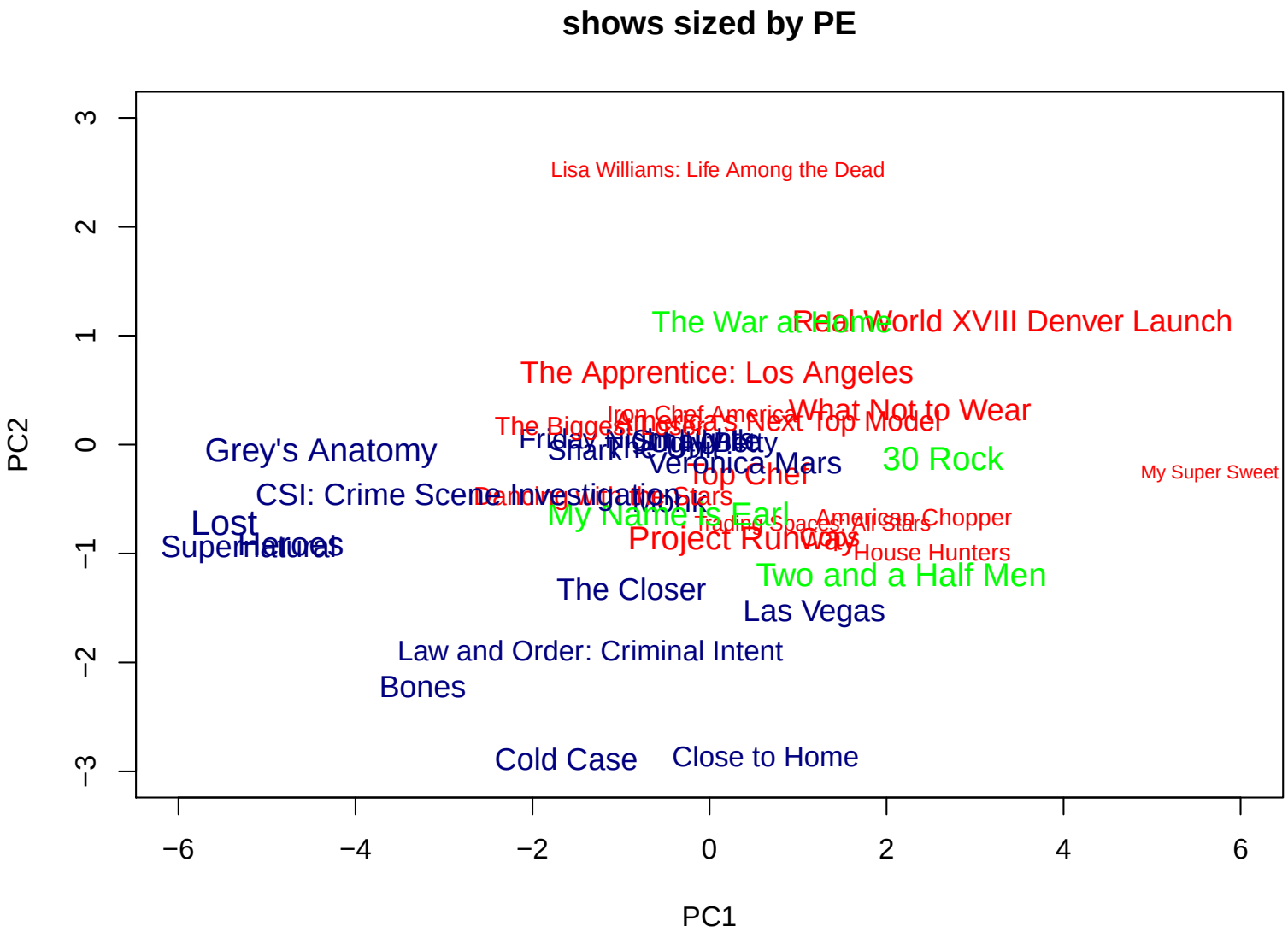



```
> round(PCApilot$rotation[,1:3],1)
                    PC1  PC2  PC3
Q1_Excited         -0.3  0.1 -0.1
Q1_Happy           -0.1  0.2 -0.5
Q1_Engaged         -0.3  0.0  0.0
Q1_Annoyed          0.2  0.3  0.1
Q1_Indifferent      0.2  0.4  0.1
Q2_Funny            0.1  0.2 -0.5
Q2_Confusing       -0.1  0.3  0.2
Q2_Predictable      0.2  0.3  0.0
Q2_Entertaining    -0.3 -0.1 -0.3
Q2_Original        -0.3  0.1 -0.2
Q2_Boring           0.2  0.4  0.1
Q2_Dramatic        -0.2  0.0  0.4
Q2_Suspenseful     -0.3  0.0  0.3
```

Huge drop after the first PC, but a few could be influential.

How do questions load? Maybe these are three genres...

NBC Pilot-Survey Principal Components



We first aggregated responses by show, then fit PC.

Choosing the number of factors

Like in K-means, this is tough without supervision.

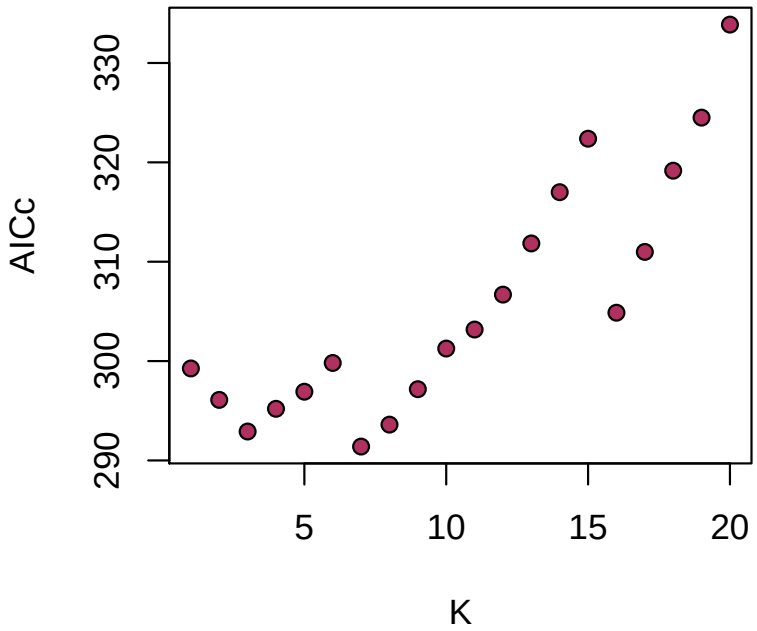
For PCR, though, we can just use the usual tools.

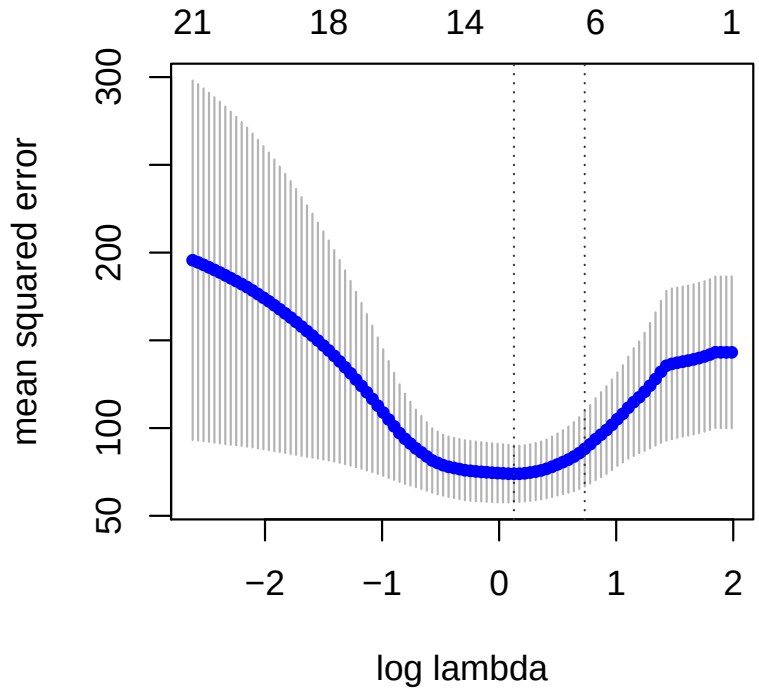
There are two ways to do this

1. Regress onto factors 1 through K for a few K, and choose the model with lowest IC or CV error.
2. Lasso all p factors with λ selected via IC or CV.

Both are fine.

Factor selection for NBC pilot survey





AICc building one-at-a-time chooses $K = 7$,
but the curve is all over the place.

CV lasso chooses the first three plus a couple others.

Factor Models vs Variable Selection

Both are good tools; you can mix and match as needed.
What to use often comes down to preference and experience.

PCA/PCR is nice in social science because you
get latent structure (e.g., the 'partisan factor').
But sometimes this is imaginary, so be careful.

Sparse vs Dense regression models

More conceptually, lasso finds a *sparse* model (many $\beta_j = 0$), whereas PCR assumes all the x's matter but only through the information they provide on a few simple factors.

Both do dimension reduction!
Which is best will depend upon the application.

EXTRA: supervised factors

An issue we discussed with clustering is relevant here too:

Factor model regression (e.g., PCR) will only work if the dominant directions of variation in $\mathbf{x}$ are related to y.

Is there a way to force factors $\mathbf{v}$ to be relevant to both $\mathbf{x}$ and y?

Yes, and its a nice Big Data technique.

Marginal Regression

An old idea

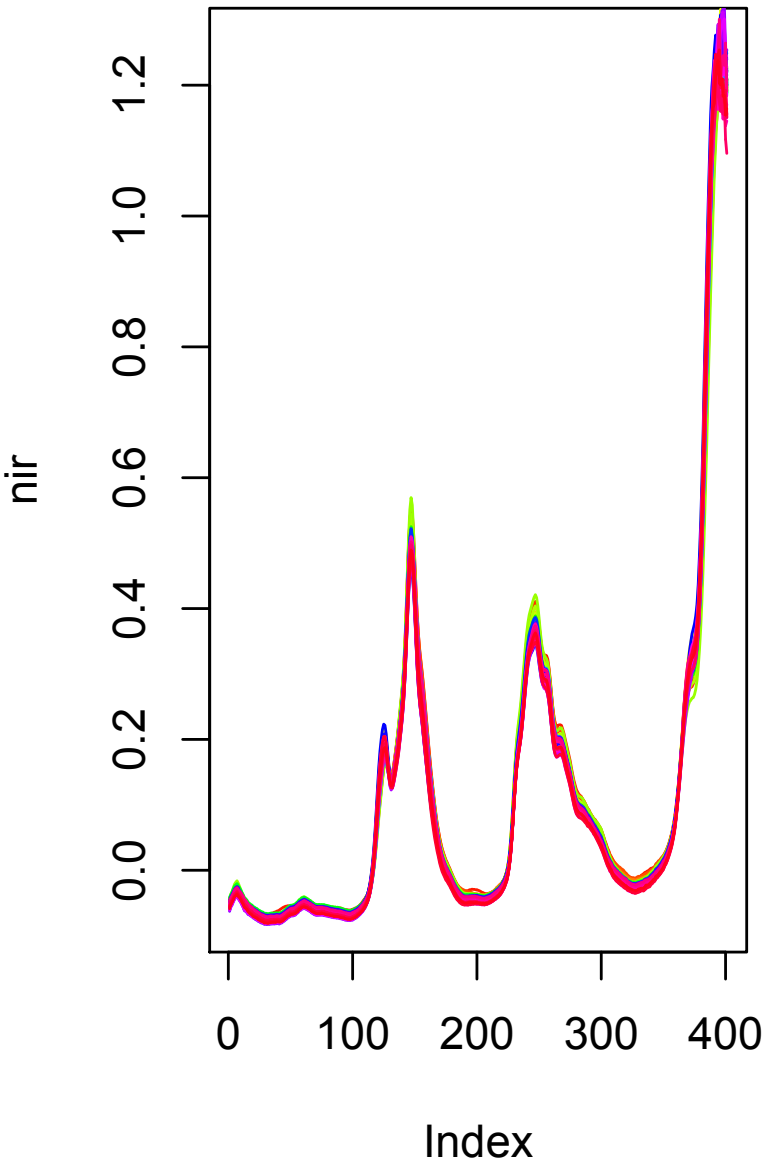
- Calculate $\varphi = [\varphi_1 \dots \varphi_p] = \mathrm{cor}(\mathbf{x}, y)/\mathrm{sd}(\mathbf{x})$.
- Set $z_i = \mathbf{x}_i'\varphi = \sum_j x_{ij}\varphi_j$.
- Fit the simple linear regression $y \sim z$.

Marginal Regression is a natural for MapReduce.

- Map data to $n \times 2$ { `x[i,j],y[i]` } matrices.
- Send each matrix to a reducer for calculating φ_j.
- In a 2nd MapReduce step, map each $\mathbf{x}_i$ and reduce $\mathbf{x}_i'\varphi$
- Finally, just fit OLS $\mathbb{E}[y_i|z_i] = \alpha + \beta z_i$.

For this reason, MR is having a comeback.

Example: Gas Data



When you buy gas,
it has an octane (quality) rating.
This is measured through failure testing on a model engine.

More frequent testing is possible through NIR sensors:

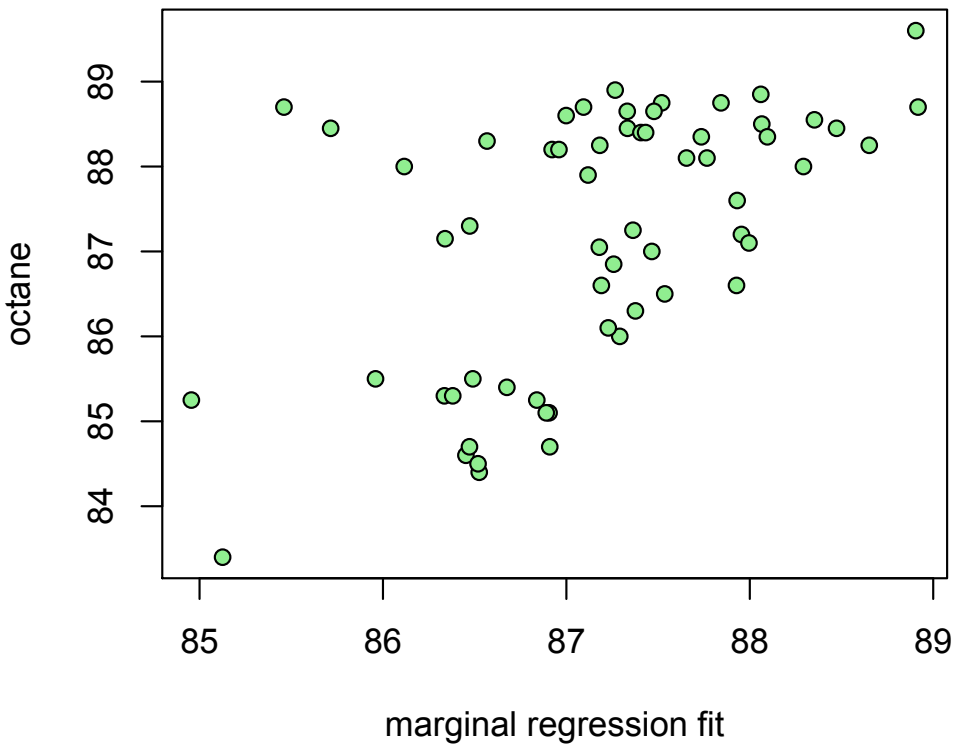
- Near infrared Spectroscopy measures reflectance at wave-lengths (1700 here) longer than visible light.
- It is useful for determining chemical composition

Marginal Regression

The model is $\mathbb{E}[\mathbf{x}_i/\mathrm{sd}(\mathbf{x})] = \varphi y_i$.

Marginal 'direction' is $z_i = \sum_j x_{ij}\mathrm{cor}(x_j, y)/\mathrm{sd}(x_j)$

And the 'forward regression model' is $\mathbb{E}[y_i \mid \mathbf{x}_i] = \alpha + \beta z_i$




```
### marginal regression
r <- cor(nir, octane)
s <- apply(nir,2,sd)
phi <- r/s
z <- nir%*%phi
fwd <- glm(octane ~ z)
```

$R^2 \approx 30\%$

PLS: Partial Least Squares

Repeated marginal regressions to overcome 'marginalness'

- PLS starts with $v_{i1} = y_i$.
- fits φ_1 for marginal regression.
- then iteratively sets v_{ik} to residuals from the regression $\mathbb{E}[y] = \alpha + \beta_1 z_1 + \ldots + \beta_k z_k$.

$z_k = \sum_j x_{ij}\varphi_{jk}$ is the kth PLS direction

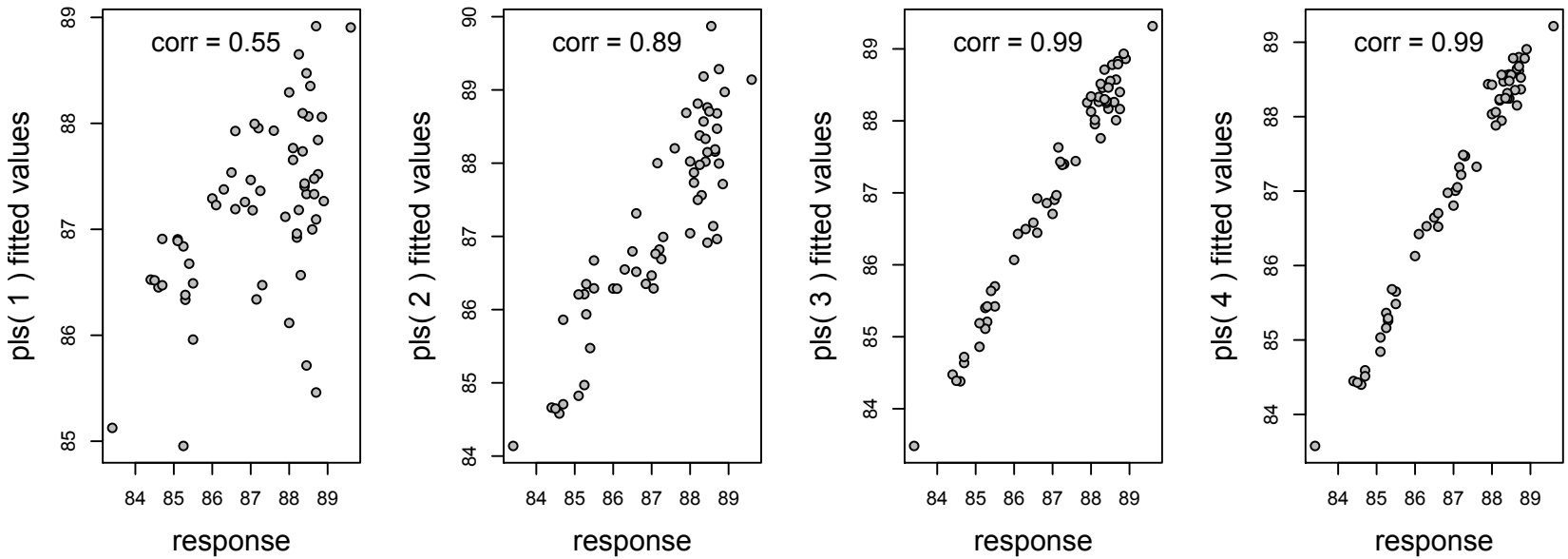
Recursively fit residuals onto covariates: stage-wise regression.

Since you just keep feeding back residuals, you can repeat this to get as good a fit as desired: it is easy to overfit!

Gas Data PLS(4) Fit

`textir` has a pls function, along with `summary`, `plot`, etc.
Get predictions with, e.g., `predict(gaspls, nir)`

```
gaspls <- pls(X=nir, y=octane,  K=4)
```



Note: `pls(1)` is just marginal regression

Foreign Exchange

What are the latent factors of international currency pricing? And how do these factor move against US equities?

We're going to investigate underlying factors in currency exchange rates and regress the S&P 500 onto this information.

FX data is in `FXmonthly.csv`.

SP500 returns are in `sp500csv`.

Currency codes are in `currency_codes.txt`.

Translate the prices to 'returns' via

```
fx <- read.csv("FXmonthly.csv")
fx <- (fx[2:120,]-fx[1:119,])/(fx[1:119,])
```

Homework Due Next Week

[1] Discuss correlation amongst dimensions of `fx`.
How does this relate to the applicability of factor modelling?

[2] Fit, plot, and interpret principal components.

[3] Regress SP500 returns onto currency movement factors,
using both '`glm` on first *K*' and lasso techniques.
Use the results to add to your factor interpretation.

[4] Fit lasso to the original covariates and
describe how it differs from PCR here.

[+] Fit marginal regression and PLS and compare to PCA.