Big Data BUS 41201

Week 8: Factor Models

Veronika Ročková

University of Chicago Booth School of Business

http://faculty.chicagobooth.edu/veronika.rockova/

The menu for today

- ✓ Dimension Reduction (DR) Revisited
- ✓ Unsupervised Data Analysis
- ✓ Factor Analysis (FA) and Latent Variables
- ✓ Principal Components Analysis (PCA)
- ✓ Principal Component Regression (PCR)
- ✓ Partial Least Squares (PLS)

Today is all about Dimension Reduction (more than normal)

The setting: we have a high-dimensional matrix of data $\mathbf{X}$.
We'd like to reduce this to a few 'important' factors.

We'll do this by building a **simple linear model for X** and use this model to represent $\mathbf{X}$ 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.

Dimension Reduction Revisited

So far, we've thought about clustering *data points* (rows of $\mathbf{X}$).

However, we can also cluster the *features* (columns of $\mathbf{X}$), or both.

Dimension reduction (DR): *the task of transforming our data set to one with less features/rows.*

A new feature/factor can can be some linear or nonlinear combination.

With factor models, we want to *capture the main structure* in the data with fewer and more informative features.

DR is often the first step in the analysis, followed by, e.g., visualization, clustering, regression, classification.

Factor Models are parsimonious models for $\mathbf{X}$

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

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

$\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_K]$ are *unobserved* lower-dimensional regressors that capture the essence of $\mathbf{X}$.

For example

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

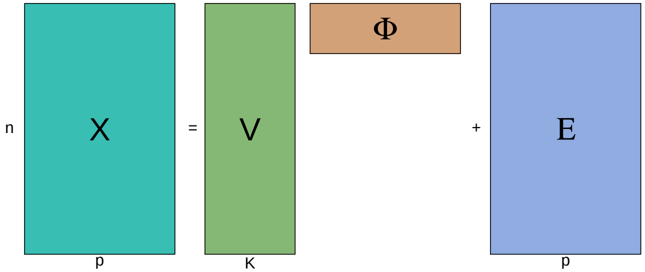
two factors: $\mathbb{E}[\mathbf{x}_i] = \boldsymbol{\varphi}_1 v_{i1} + \boldsymbol{\varphi}_2 v_{i2}$

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

The basic underlying model is *multivariate regression*

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



- $\mathbf{V} = \{v_{ij}\}_{i,j=1}^{n,K} = [\mathbf{v}_1, \ldots, \mathbf{v}_K]$: latent factors (unobserved regressors)
- $\mathbf{\Phi} = \{\varphi_{jk}\}_{j,k=1}^{p,K} = [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_K]$: factor loadings (regression coefficients)
- **E** Gaussian errors

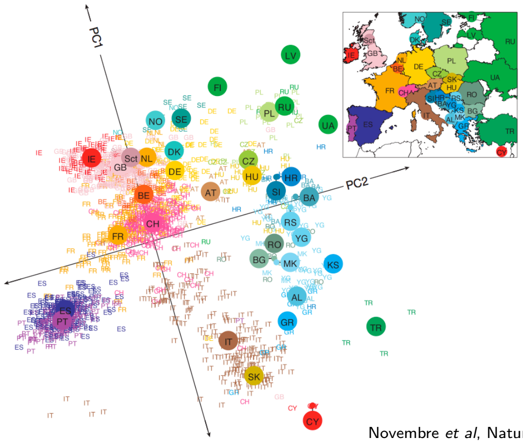
Factor Models: $\mathbb{E}[\mathbf{x}_i] = \varphi_1 v_{i1} + \ldots + \varphi_K v_{iK}$

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) (supervised learning) or *use y to build them* (PLS) (semi-supervised learning).

Geo-Genes Example: **two interpretable PC's.**



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 topic model: $\mathbb{E}[\mathbf{x}_i] = \omega_{i1}\boldsymbol{\theta}_1 + \ldots \omega_{iK}\boldsymbol{\theta}_K$.

The topic model is a factor model

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

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

The basic interpretation is exactly the same as in PCA:
$\boldsymbol{\omega}$ is a low-dimension version of $\mathbf{x}$

How can we compute FA?

For given K, the goal is to find *loadings* $\mathbf{\Phi}$ so that the *deviance* is small.

☹ We do not know *factors* $\mathbf{V}$ and we do not know *loadings* $\mathbf{\Phi}$

☹ *Chicken-and-Egg problem*

But!

(1) **If we knew $\mathbf{\Phi}$**:

we can estimate $\mathbf{V}$

(2) **If we knew $\mathbf{V}$**:

we can estimate $\mathbf{\Phi}$

Solution: iterate between (1) and (2)

This strategy relates to the EM algorithm, one of the workhorses of statistical computing.

Principal Component Analysis (PCA)

Factor models are related to principal components analysis

- *Factor Analysis* (FA): is a real model for data.
- *Principal Components Analysis* (PCA): is a model-free dimension reduction method.

PCA: finding a low-dimensional representation of data that captures *as much information as possible.*

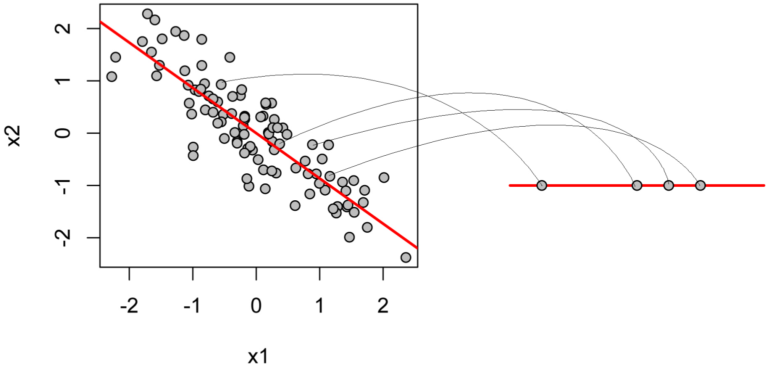
Linear dimension reduction: looking for straight lines in the feature space along which the data exhibit an interesting trend.

We interpret "interesting" as having high variance (information).

PCA: projections into latent space

Another way to think about principal components (factors) is through projections, in our 2D example:

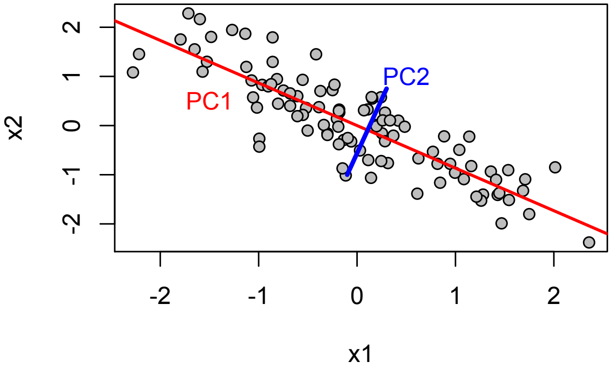
PCA equivalent to finding the line that fits through x_1 and x_2, and seeing where each observation lands (projects) on the line.



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 tries to find $\boldsymbol{\varphi}$'s and $\mathbf{v}$'s with a different argument.

Unlike with FA, there is no model. Principal component directions obtained by rotating coordinates.

To find the *first principal component*, we look for a linear combination of the original features $x_1, \ldots, x_p$ that maximizes variance (information).

The 1st principal component direction for observation i is

$$v_{i1} = \mathbf{x}_i' \boldsymbol{\varphi}_1 = \varphi_{11} x_{i1} + \cdots + \varphi_{1p} x_{ip}$$

where $\varphi_{11}, \ldots, \varphi_{1p}$ are obtained as a solution to

$$\text{maximize} \left\{ \frac{1}{n} \sum_{i=1}^{n} v_{i1}^2 \right\} \quad \text{where} \quad \sum_{j=1}^{p} \varphi_{1j}^2 = 1$$

PCA: Principal Components Analysis

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

$$v_{ik} = \mathbf{x}_i' \boldsymbol{\varphi}_k = \varphi_{k1} x_{i1} + \cdots + \varphi_{kp} x_{ip} \tag{2}$$

where $\varphi_{k1}, \ldots, \varphi_{kp}$ are obtained as a solution to

$$\text{maximize} \left\{ \frac{1}{n} \sum_{i=1}^{n} v_{ik}^2 \right\} \quad \text{where} \quad \sum_{j=1}^{p} \varphi_{kj}^2 = 1$$

and where $\mathbf{v}^k$ *is orthogonal to* $\mathbf{v}^1, \ldots, \mathbf{v}^{k-1}$.

Another way to write equations (2) altogether for $k = 1, \ldots, K$ is

$$\mathbf{v}_i = \mathbf{x}_i' \boldsymbol{\Phi}.$$

This is a **projection** from $\mathbf{x}$ into a low-dimensional feature space.
$\boldsymbol{\Phi} = [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_K]$ are called rotations.

FA versus PCA

In FA, we are fitting

$$\mathbb{E}[\mathbf{x}_i] = \varphi_1 v_{i1} + \ldots + \varphi_K v_{iK} = \mathbf{\Phi} \mathbf{v}_i.$$

In PCA, we are fitting

$$\mathbf{x}_i = \varphi_1 v_{i1} + \ldots + \varphi_K v_{iK} = \mathbf{\Phi} \mathbf{v}_i.$$

Because the principal component directions are orthogonal, we have $\mathbf{\Phi}'\mathbf{\Phi} = \mathrm{I}$ (identity matrix). This implies

$$\mathbf{v}_i' = \mathbf{x}_i' \mathbf{\Phi}$$

Thus, factors and principal components are related.

How many principal components do we need?

Choose smallest K needed to explain a *sizeable* amount of variation.

Total variance present in the data

$$TV = \sum_{j=1}^{p} \mathrm{var}(x_j) = \sum_{j=1}^{p} \frac{1}{n} \left(\sum_{i=1}^{n} x_{ij}^2 \right)$$

Variance **explained by k^{th} principal component** is

$$d_k^2 = \mathrm{var}(z_k) = \frac{1}{n} \sum_{i=1}^{n} z_{ik}^2$$

Proportion of explained variance (PEV) with K principal components

$$PEV(K) = \frac{\sum_{k=1}^{K} d_k^2}{TV}$$

$PEV(K)$ should be large

Principal Components in R

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

- There are other options.
- 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 $\mathbf{z}$ for new data.

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

NB: $\mathbf{x}$ is first *scaled* by the SDs used in `mypca` if `scale=TRUE`.

Understanding Principal Components

Suppose that each principal component/ factor represents a *diet*

Each diet is a weighted combination of proteins.

(1) Principal component score v_{ik}:
for i^{th} country and k^{th} component (diet) represents

how much protein is consumed in i^{th} country under k^{th} diet

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

(2) Rotation ϕ_{kj}:
for k^{th} component (diet) and j^{th} protein represents

gives the weight of j^{th} protein in k^{th} diet

and thus encodes the *foods* associated with diet k

Interpreting the rotations/loadings

Sometimes we can find a meaningful structure from the PCA output.

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.

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

Understanding the principal components

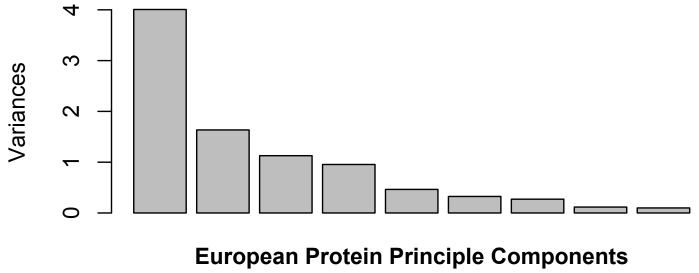
How many do we need? What do the factors contribute?

```
> 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 $1 \ldots K$.

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

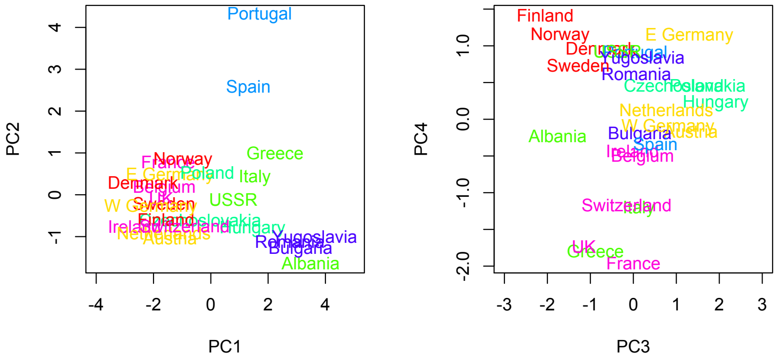
Screeplot: show variance for each principal component.



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. Here, it actually looks like only the first really matters.

Principal components in European protein consumption



Overlaying k-means clustering from Week 7, 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

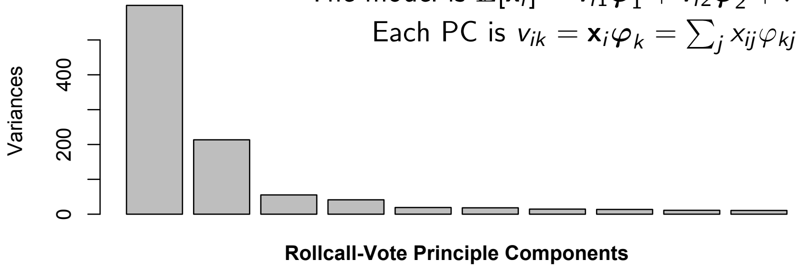
Votes in which names and positions
are recorded are called ‘roll calls’.

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

445 members in the US House (the 111th)
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 111th house



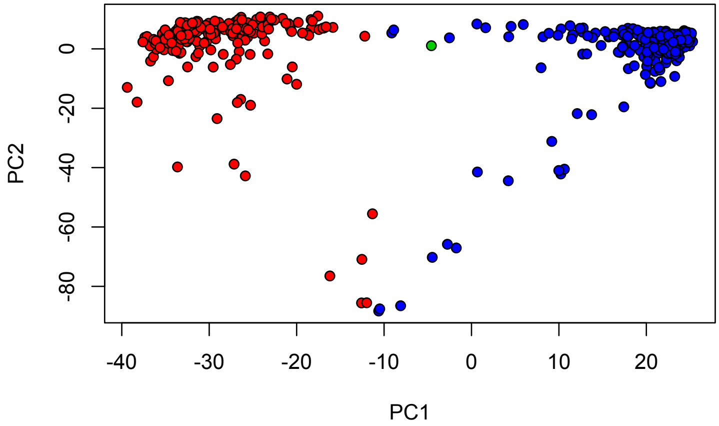
The model is $\mathbb{E}[\mathbf{x}_i] = v_{i1}\boldsymbol{\varphi}_1 + v_{i2}\boldsymbol{\varphi}_2 + \ldots$

Each PC is $v_{ik} = \mathbf{x}_i\boldsymbol{\varphi}_k = \sum_j x_{ij}\varphi_{kj}$

Huge drop in variance from 1st to 2nd and 2nd to 3rd 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 111th house



Republicans in red and Democrats in blue:

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

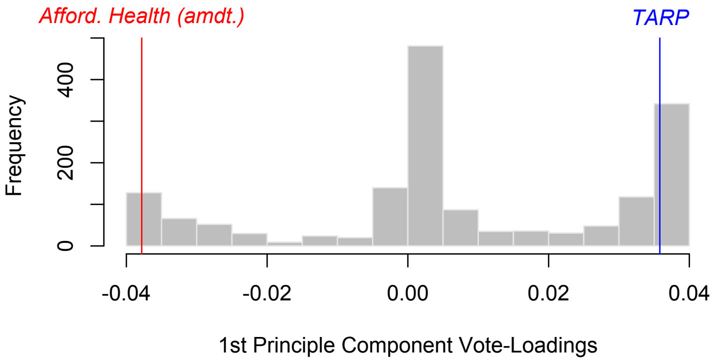
Interpreting the principal components

```
## 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 **v** 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

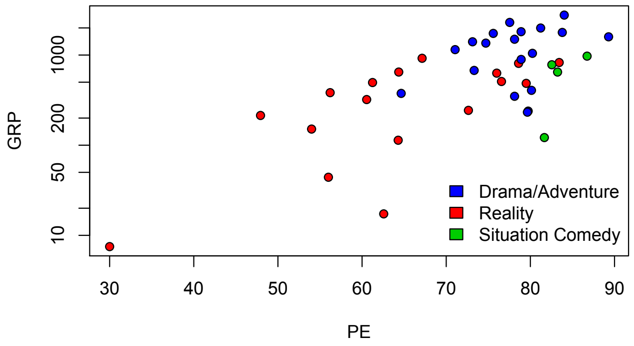
Primary goal is predicting engagement.

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!).

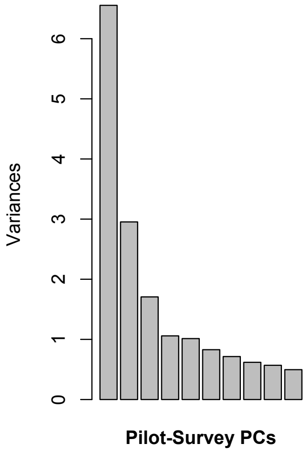
Predicting TV Engagement with PCR

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?

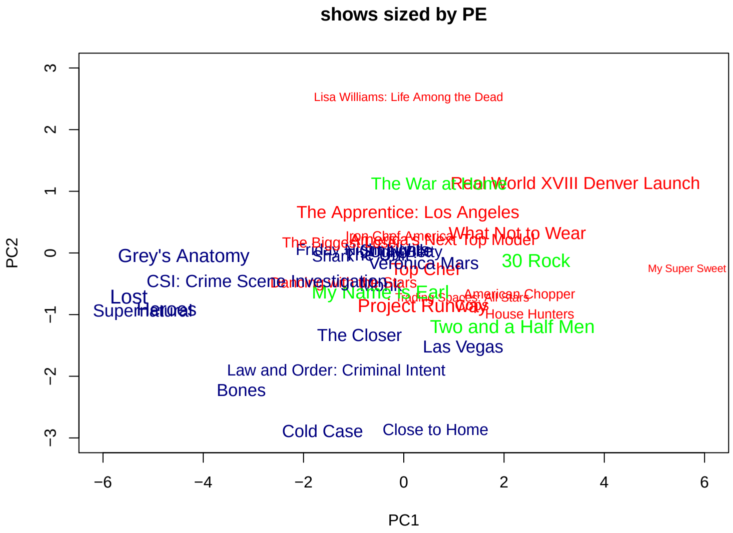
NBC Pilot-Survey PCA



```
> 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.

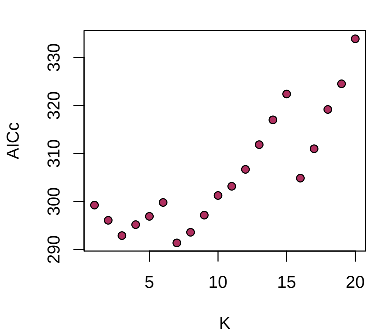
For PCR, though, we can just use the usual tools.

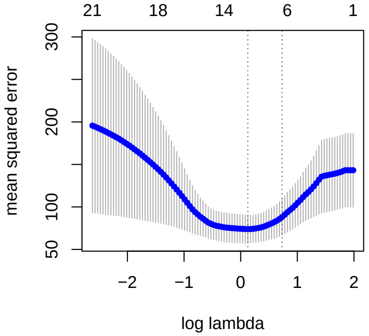
There are two ways to do this

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.

PLS (Partial Least Squares):
finds directions that help explain **both** $\mathbf{x}$ **and** y.

Partial Least Squares

How can we come up with a **set** *of linear combinations* $\mathbf{v}_1, \ldots, \mathbf{v}_K$ *of features* $\mathbf{x}_1, \ldots, \mathbf{x}_p$ *that are best for predicting* $\mathbf{y}$*?*

(1) *Regress* $\mathbf{y}$ onto each $\mathbf{x}_j$ and store the regression slope φ_j.

(2) Set the first component as a *weighted average* of $\mathbf{x}_j'$s with weights φ_j, i.e.

$$\mathbf{v}_1 = \mathbf{x}'\boldsymbol{\varphi}$$

(3) *Adjust* $\mathbf{x}_j$'s by regressing each $\mathbf{x}_j$'s on $\mathbf{v}_1$ to get *residuals*

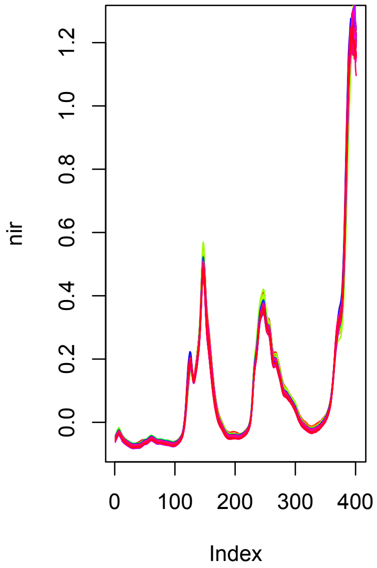
$$\mathbf{r}_1, \ldots, \mathbf{r}_p.$$

These residuals capture what has not yet been explained by $\mathbf{v}_1$.

For the *second direction* $\mathbf{v}_2$, we proceed with steps (1) and (2) using the residuals to obtain $\mathbf{v}_2$.

For the *third direction* $\mathbf{v}_3$, we regress each $\mathbf{x}_1, \ldots, \mathbf{x}_p$ on **both** $\mathbf{v}_1$ **and** $\mathbf{v}_2$ and get *residuals* etc.

Example: Gas Data



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

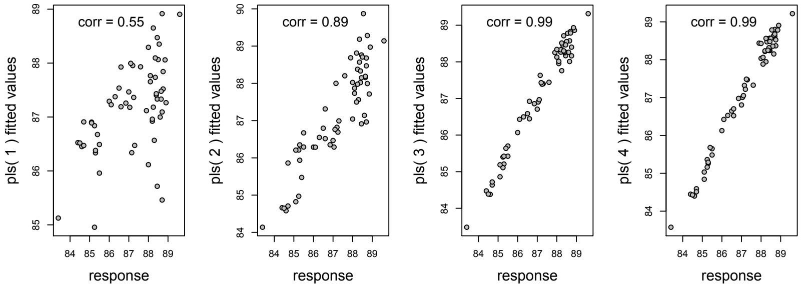
More frequent testing is possible through NIR sensors:

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

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