Principal components

Principal Components

- Have measurements on (possibly large) number of variables on some individuals.
- Question: can we describe data using fewer variables (because original variables correlated in some way)?
- Look for direction (linear combination of original variables) in which values *most spread out*. This is *first principal component*.
- Second principal component then direction uncorrelated with this in which values then most spread out. And so on.

Principal components

- See whether small number of principal components captures most of variation in data.
- Might try to interpret principal components.
- If 2 components good, can make plot of data.
- (Like discriminant analysis, but for individuals rather than groups.)
- "What are important ways that these data vary?"

Packages

You might not have installed the first of these. See over for instructions.

```r
library(ggbiplot)
library(tidyverse)
library(ggrepel)
library(conflicted)
conflicts_prefer(dplyr::mutate)
```

ggbiplot has a special installation: see over.

Installing ggbiplot

- ggbiplot not on CRAN, so usual `install.packages` will not work. This is same procedure you used for `smmr` in C32:
- Install package `devtools` first (once):

```
install.packages("devtools")
```

- Then install ggbiplot (once):

```
library(devtools)
install_github("vqv/ggbiplot")
```

Small example: 2 test scores for 8 people

```
my_url <- "http://ritsokiguess.site/datafiles/test12.txt"
test12 <- read_table(my_url)
test12
```
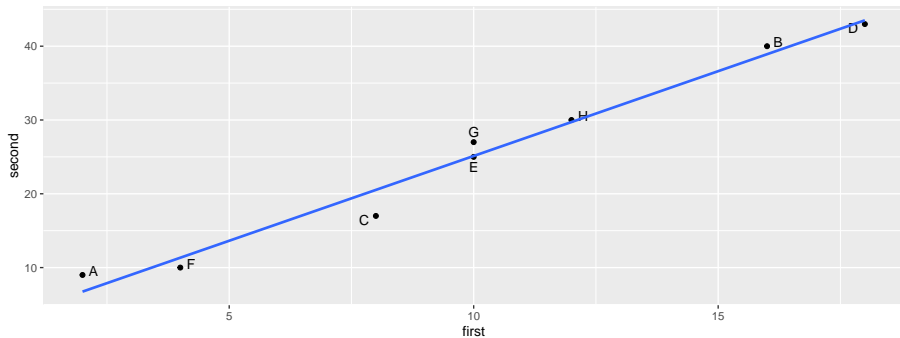
```
# A tibble: 8 x 3
  first second id
  <dbl>  <dbl> <chr>
1     2      9 A
2    16     40 B
3     8     17 C
4    18     43 D
5    10     25 E
6     4     10 F
7    10     27 G
8    12     30 H
```

```
g <- ggplot(test12, aes(x = first, y = second, label = id)) +
  geom_point() + geom_text_repel()
```

The plot

```
g + geom_smooth(method = "lm", se = F)
```

Principal component analysis

- Grab just the numeric columns:

```
test12 %>% select(where(is.numeric)) -> test12_numbers
test12_numbers
```

```
# A tibble: 8 x 2
  first second
  <dbl>  <dbl>
1     2      9
2    16     40
3     8     17
4    18     43
5    10     25
6     4     10
7    10     27
8    12     30
```

- Strongly correlated, so data nearly 1-dimensional:

Finding principal components

- Make a score summarizing this one dimension. Like this:

```
test12.pc <- princomp(test12_numbers, cor = TRUE)
summary(test12.pc)
```

```
Importance of components:
                         Comp.1      Comp.2
Standard deviation      1.410347 0.104508582
Proportion of Variance 0.994539 0.005461022
Cumulative Proportion  0.994539 1.000000000
```

Comments

- "Standard deviation" shows relative importance of components (as for LDs in discriminant analysis)
- Here, first one explains almost all (99.4%) of variability.
- That is, look only at first component and ignore second.
- `cor=TRUE` standardizes all variables first. Usually wanted, because variables measured on different scales. (Only omit if variables measured on same scale and expect similar variability.)

Scree plot

```
ggscreeplot(test12.pc)
```
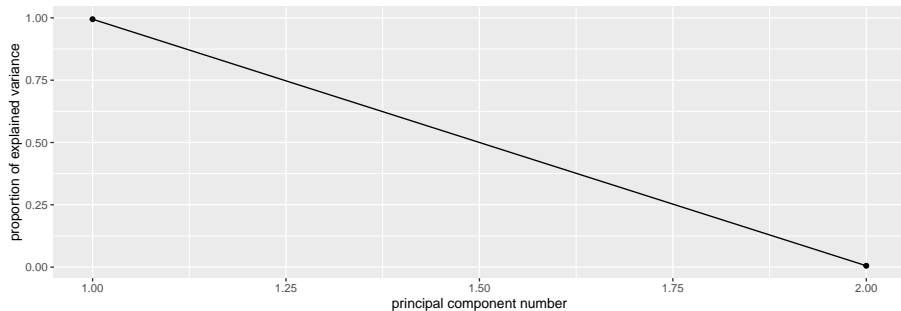

Imagine scree plot continues at zero, so 2 components is a *big* elbow (take one component).

Component loadings

explain how each principal component depends on (standardized) original variables (test scores):

```
test12.pc$loadings
```

```
Loadings:
      Comp.1 Comp.2
first  0.707  0.707
second 0.707 -0.707

              Comp.1 Comp.2
SS loadings      1.0    1.0
Proportion Var   0.5    0.5
Cumulative Var   0.5    1.0
```

First component basically sum of (standardized) test scores. That is, person tends to score similarly on two tests, and a composite score would summarize performance.

Component scores

```
d <- data.frame(test12, test12.pc$scores)
d
```

```
  first second id       Comp.1        Comp.2
1     2      9  A -2.071819003 -0.146981782
2    16     40  B  1.719862811 -0.055762223
3     8     17  C -0.762289708  0.207589512
4    18     43  D  2.176267535  0.042533250
5    10     25  E -0.007460609  0.007460609
6     4     10  F -1.734784030  0.070683441
7    10     27  G  0.111909141 -0.111909141
8    12     30  H  0.568313864 -0.013613668
```

- Person A is a low scorer, very negative comp.1 score.
- Person D is high scorer, high positive comp.1 score.
- Person E average scorer, near-zero comp.1 score.
- comp.2 says basically nothing.

Plot of scores

```
ggplot(d, aes(x = Comp.1, y = Comp.2, label = id)) +
  geom_point() + geom_text_repel()
```
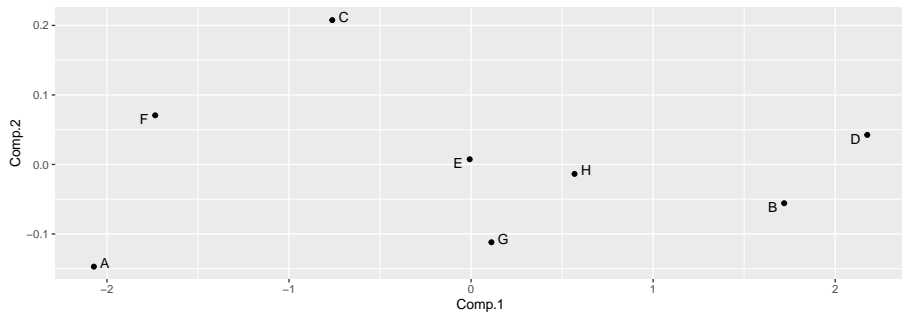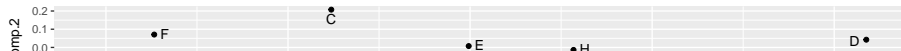
Comments

- Vertical scale exaggerates importance of comp.2.
- Fix up to get axes on same scale:

```
ggplot(d, aes(x = Comp.1, y = Comp.2, label = id)) +
  geom_point() + geom_text_repel() +
  coord_fixed() -> g
```

- Shows how exam scores really spread out along one dimension:
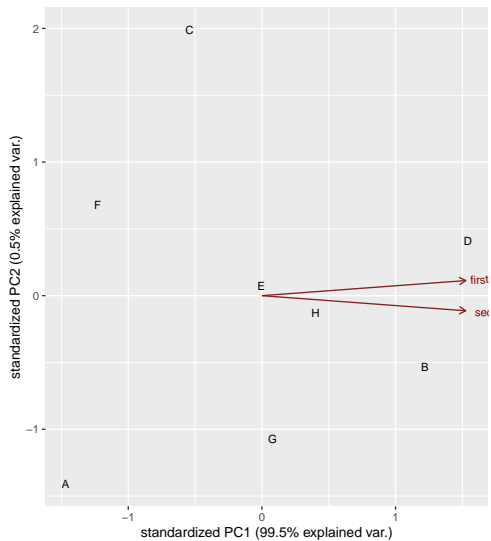
```
g
```

The biplot

- Plotting variables and individuals on one plot.
- Shows how components and original variables related.
- Shows how individuals score on each component, and therefore suggests how they score on each variable.
- Add `labels` option to identify individuals:

```
g <- ggbiplot(test12.pc, labels = test12$id)
```

The biplot

Comments

- Variables point almost same direction (right). Thus very positive value on comp.1 goes with high scores on both tests, and test scores highly correlated.
- Position of individuals on plot according to scores on principal components, implies values on original variables. Eg.:
- D very positive on comp.1, high scorer on both tests.
- A and F very negative on comp.1, poor scorers on both tests.
- C positive on comp.2, high score on first test relative to second.
- A negative on comp.2, high score on second test relative to first.

Places rated

Every year, a new edition of the Places Rated Almanac is produced. This rates a large number (in our data 329) of American cities on a number of different criteria, to help people find the ideal place for them to live (based on what are important criteria for them).

The data for one year are in http://ritsokiguess.site/datafiles/places.txt. The data columns are aligned but the column headings are not.

The criteria

There are nine of them:

- `climate`: a higher value means that the weather is better
- `housing`: a higher value means that there is more good housing or a greater choice of different types of housing
- `health`: higher means better healthcare facilities
- `crime`: higher means more crime (bad)
- `trans`: higher means better transportation (this being the US, probably more roads)
- `educate`: higher means better educational facilities, schools, colleges etc.
- `arts`: higher means better access to the arts (theatre, music etc)
- `recreate`: higher means better access to recreational facilities
- `econ`: higher means a better economy (more jobs, spending power etc)

Each city also has a numbered `id`.

Read in the data

```
my_url <- "http://ritsokiguess.site/datafiles/places.txt"
places0 <- read_table(my_url)
```
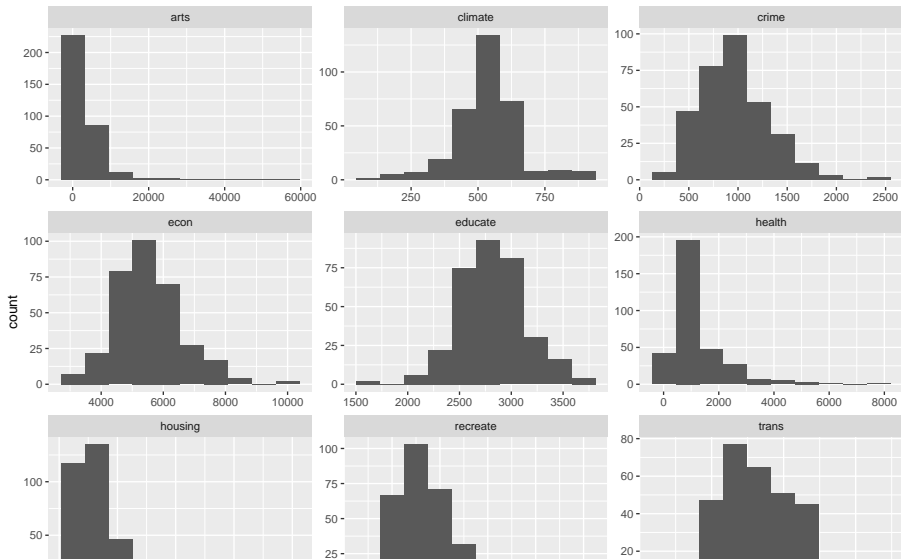
Look at distributions of everything

```
places0 %>%
  pivot_longer(-id, names_to = "criterion",
               values_to = "rating") %>%
  ggplot(aes(x = rating)) + geom_histogram(bins = 10) +
  facet_wrap(~criterion, scales = "free") -> g
```

The histograms

g

Transformations

- Several of these variables have long right tails
- Take logs of everything but id:

```
places0 %>%
  mutate(across(-id, \(x) log(x))) -> places
places
```

```
# A tibble: 329 x 10
   climate housing health crime trans educate  arts recreate
     <dbl>   <dbl>  <dbl> <dbl> <dbl>   <dbl> <dbl>    <dbl> <
 1    6.26    8.73   5.47  6.83  8.30    7.92  6.90     7.25
 2    6.35    9.00   7.41  6.79  8.49    7.80  8.62     7.88
 3    6.15    8.90   6.43  6.88  7.84    7.85  5.47     6.76
 4    6.17    8.98   7.27  6.41  8.84    8.13  8.45     7.39
 5    6.49    9.04   7.52  7.30  8.79    8.01  8.41     7.87
 6    6.25    8.67   6.46  6.59  7.80    8.00  5.81     6.93
 7    6.33    9.02   6.43  6.24  7.97    8.05  7.75     7.02
 8    6.29    8.78   6.87  6.56  8.51    7.99  7.30     7.15
```

Just the numerical columns

- get rid of the id column

```
places %>% select(-id) -> places_numeric
```

Principal components

```
places.1 <- princomp(places_numeric, cor = TRUE)
summary(places.1)
```

```
Importance of components:
                        Comp.1    Comp.2    Comp.3    Comp.4     Comp.5
Standard deviation     1.8159827 1.1016178 1.0514418 0.9525124 0.92770076
Proportion of Variance 0.3664214 0.1348402 0.1228367 0.1008089 0.09562541
Cumulative Proportion  0.3664214 0.5012617 0.6240983 0.7249072 0.82053259
                         Comp.6     Comp.7     Comp.8     Comp.9
Standard deviation     0.74979050 0.69557215 0.56397886 0.50112689
Proportion of Variance 0.06246509 0.05375785 0.03534135 0.02790313
Cumulative Proportion  0.88299767 0.93675552 0.97209687 1.00000000
```

scree plot

```
ggscreeplot(places.1)
```
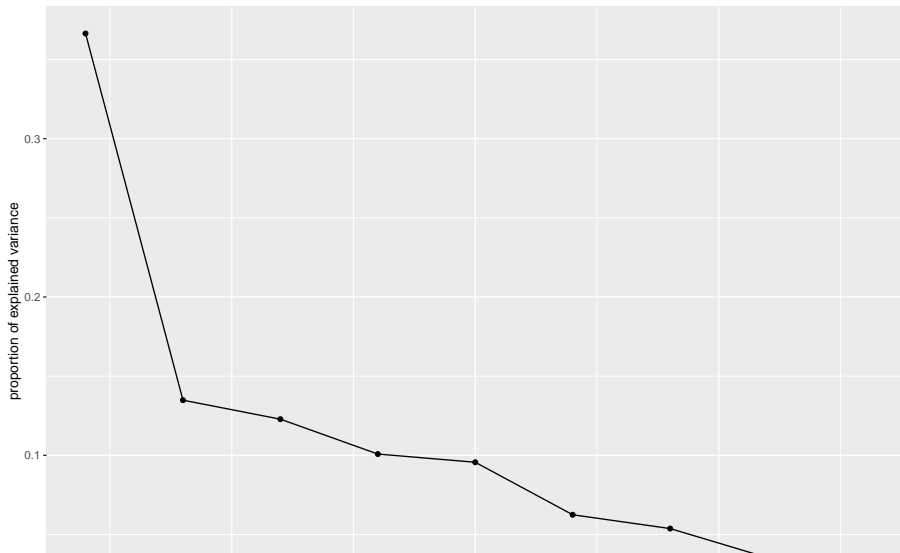
What is in each component?

```
places.1$loadings
```

```
Loadings:
         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
climate   0.158          0.800  0.377          0.217  0.151
housing   0.384  0.139          0.197 -0.580          0.275
health    0.410 -0.372          0.113         -0.535 -0.135
crime     0.259  0.474  0.128          0.692 -0.140 -0.110
trans     0.375 -0.141 -0.141 -0.430  0.191  0.324  0.679
educate   0.274 -0.452 -0.241  0.457  0.225  0.527 -0.262
arts      0.474 -0.104         -0.147         -0.321 -0.120
recreate  0.353  0.292         -0.404 -0.306  0.394 -0.553
econ      0.164  0.540 -0.507  0.476                 0.147
         Comp.8 Comp.9
climate   0.341
housing  -0.606
health    0.150  0.594
crime    -0.420
```

Assessing the components

Look at component loadings and make a call about "large" (in absolute value) vs "small". Large loadings are a part of the component and small ones are not. Thus, if we use 0.4 as cutoff:

- component #1 depends on health and arts
- #2 depends on economy and crime, and negatively on education.
- #3 depends on climate, and negatively on economy.
- #4 depends on education and the economy, negatively on transportation and recreation opportunities.
- #5 depends on crime and negatively on housing.

Comments

- The use of 0.4 is arbitrary; you can use whatever you like. It can be difficult to decide whether a variable is "in" or "out".
- The large (far from zero) loadings indicate what distinguishes the cities as places to live, for example:
  - ▶ places that are rated high for health also tend to be rated high for arts
  - ▶ places that have a good economy tend to have a bad climate (and vice versa)
  - ▶ places that have a lot of crime tend to have bad housing.

Making a plot 1/3

How can we make a visual showing the cities? We need a "score" for each
city on each component, and we need to identify the cities (we have a
numerical id in the original dataset):

```
cbind(city_id = places$id, places.1$scores) %>%
  as_tibble() -> places_score
places_score
```

```
# A tibble: 329 x 10
   city_id Comp.1 Comp.2  Comp.3 Comp.4  Comp.5 Comp.6
     <dbl>  <dbl>  <dbl>   <dbl>  <dbl>   <dbl>  <dbl>
 1       1  -1.20   1.48  -0.946  0.557   0.676  0.953
 2       2  0.940 -0.249   1.11  -1.66   -0.412 -0.708
 3       3  -2.35   0.340  0.0254 0.650   0.399 -0.740
 4       4   1.38  -1.62   -1.26  0.179   0.0710 0.808
 5       5   2.44   0.192  0.416 -0.265   0.984  0.423
 6       6  -2.24  -0.617 -0.105  1.09    0.613  0.221
 7       7  -0.794 -1.27   0.0429 1.15   -0.684  0.299
```
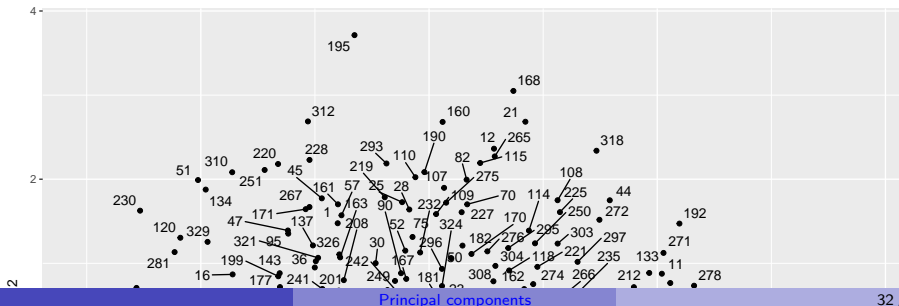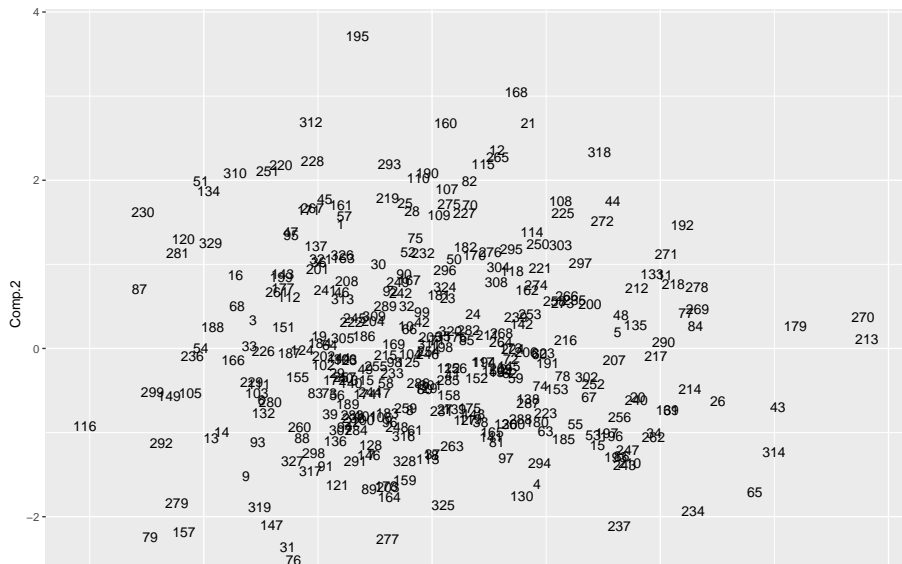
Making a plot 2/3

- Plot the first two scores against each other, labelling each point by the id of the city it belongs to:

```
ggplot(places_score, aes(x = Comp.1, y = Comp.2,
                          label = city_id)) +
  geom_text() -> g
ggplot(places_score, aes(x = Comp.1, y = Comp.2,
                          label = city_id)) +
  geom_point() + geom_text_repel()
```

g

Comments

- Cities 213 and 270 are high on component 1, and city 116 is low. City 195 is high on component 2, and city 322 is low.
- This suggests that cities 213 and 270 are high on health and arts, and city 116 is low. City 195 should be high on economy and crime and low on education, and city 322 should be the other way around.

Checking this 1/2

- The obvious way of checking this is in two steps: first, work out what high or low means for each variable:

```
summary(places)
```

```
    climate          housing           health           crime
Min.   :4.654    Min.   : 8.548    Min.   :3.761    Min.   :5.730
1st Qu.:6.174    1st Qu.: 8.819    1st Qu.:6.368    1st Qu.:6.561
Median :6.295    Median : 8.972    Median :6.725    Median :6.853
Mean   :6.260    Mean   : 8.997    Mean   :6.805    Mean   :6.796
3rd Qu.:6.384    3rd Qu.: 9.107    3rd Qu.:7.276    3rd Qu.:7.053
Max.   :6.813    Max.   :10.071    Max.   :8.968    Max.   :7.823
    trans            educate           arts            recreate
Min.   :7.043    Min.   :7.439    Min.   : 3.951    Min.   :5.704
1st Qu.:8.052    1st Qu.:7.871    1st Qu.: 6.657    1st Qu.:7.182
Median :8.314    Median :7.935    Median : 7.534    Median :7.421
Mean   :8.283    Mean   :7.936    Mean   : 7.383    Mean   :7.429
3rd Qu.:8.557    3rd Qu.:8.010    3rd Qu.: 8.254    3rd Qu.:7.685
Max.   :9.062    Max.   :8.238    Max.   :10.946    Max.   :8.476
    econ              id
Min.   :8.021    Min.   :  1
1st Qu.:8.485    1st Qu.: 83
Median :8.591    Median :165
Mean   :8.598    Mean   :165
3rd Qu.:8.718    3rd Qu.:247
Max.   :9.208    Max.   :329
```

Checking this 2/2

- and then find the values on the variables of interest for our cities of interest, and see where they sit on here.
- Cities 270, 213, and 116 were extreme on component 1, which depended mainly on health and arts:

```
conflicts_prefer(dplyr::filter)
places %>% select(id, health, arts) %>%
  filter(id %in% c(270, 213, 116))
```

```
# A tibble: 3 x 3
     id health  arts
  <dbl>  <dbl> <dbl>
1   116   6.43  5.03
2   213   8.97 10.9
3   270   8.22  9.56
```

City 166 is near or below Q1 on both variables. City 213 is the highest of all on both `health` and `arts`, while city 270 is well above Q3 on both.

Checking component 2

- Component 2 depended positively on economy and crime and negatively on education. City 195 was high and 322 was low:

```
places %>% select(id, econ, crime, educate) %>%
  filter(id %in% c(195, 322))
```

```
# A tibble: 2 x 4
     id  econ crime educate
  <dbl> <dbl> <dbl>   <dbl>
1   195  9.21  7.06    7.79
2   322  8.10  6.14    7.97
```

- City 195 is the highest on economy, just above Q3 on crime, and below Q1 on education. City 322 should be the other way around: nearly the lowest on economy, below Q1 on crime, and between the median and Q3 on education. This is as we'd expect.

A better way: percentile ranks

- It is a lot of work to find the value of each city on each variable in the data summary.
- A better way is to work out the percentile ranks of each city on each variable and then look at those:

```
places %>%
  mutate(across(-id, \(x) percent_rank(x))) -> places_pr
places_pr
```

```
# A tibble: 329 x 10
   climate housing health  crime trans educate   arts recreate
     <dbl>   <dbl>  <dbl>  <dbl> <dbl>   <dbl>  <dbl>    <dbl>
 1   0.387  0.0976 0.0274  0.473 0.494   0.454  0.296    0.296
 2   0.671  0.576  0.780   0.427 0.683   0.0945 0.851    0.848
 3   0.220  0.390  0.314   0.530 0.119   0.207  0.0823   0.0762
 4   0.238  0.518  0.744   0.162 0.957   0.951  0.808    0.470
 5   0.912  0.662  0.820   0.909 0.939   0.762  0.796    0.845
 6   0.384  0.0518 0.332   0.262 0.104   0.704  0.122    0.119
```

Look up cities and variables again

```
places_pr %>% select(id, health, arts) %>%
  filter(id %in% c(270, 213, 166))
```

```
# A tibble: 3 x 3
     id health   arts
  <dbl>  <dbl>  <dbl>
1   166  0.152 0.0488
2   213  1      1
3   270  0.970  0.982
```

This shows that city 270 was also really high on these two variables: in the 97th percentile for `health` and the 98th for `arts`.

Component 2

- What about the extreme cities on component 2?

```
places_pr %>% select(id, econ, crime, educate) %>%
  filter(id %in% c(195, 322))
```

```
# A tibble: 2 x 4
     id    econ   crime educate
  <dbl>   <dbl>   <dbl>   <dbl>
1   195 1       0.762   0.0884
2   322 0.00610 0.0732  0.631
```

- City 322 was really low on economy and crime, but only just above average on education. City 195 was the highest on economy and really low on education, but only somewhat high on crime (76th percentile).
- This, as you see, is much easier once you have set it up.

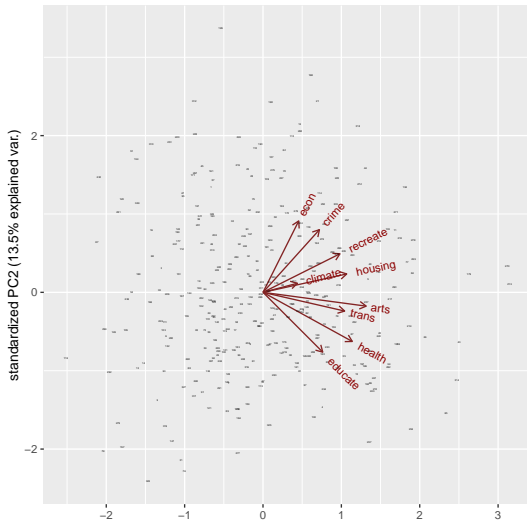# The biplot

```
ggbiplot(places.1, labels = places$id)
```
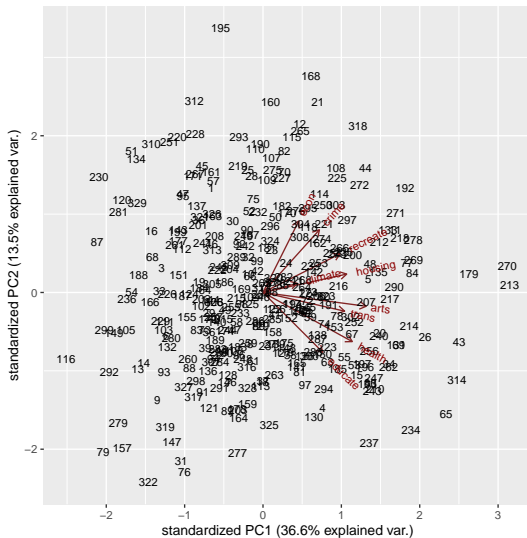
Comments

- This is hard to read!
- There are a lot of cities that overshadow the red arrows for the variables.
- reduce the size of the city labels

Biplot, attempt 2

```
ggbiplot(places.1, labels = places$id,
         labels.size = 0.8)
```

Comments on attempt #2

- Now at least can see the variables
- All of them point somewhat right (all belong partly to component 1)
- Some of them (economy, crime, education) point up/down, belong to component 2 as well.
- In this case, cannot really see both observations (cities) and variables (criteria) together, which defeats the purpose of the biplot.
- Have to try it and see.

Principal components from correlation matrix

Create data file like this:

```
 1         0.9705 -0.9600
 0.9705    1      -0.9980
-0.9600   -0.9980  1
```

and read in like this:

```
my_url <- "http://ritsokiguess.site/datafiles/cov.txt"
mat <- read_table(my_url, col_names = F)
mat
```

```
# A tibble: 3 x 3
      X1     X2     X3
   <dbl>  <dbl>  <dbl>
1  1      0.970 -0.96
2  0.970  1     -0.998
3 -0.96  -0.998  1
```

Pre-processing

A little pre-processing required:

- Turn into matrix (from data frame)
- Feed into `princomp` as `covmat=`

```
mat.pc <- mat %>%
  as.matrix() %>%
  princomp(covmat = .)
```

Scree plot: one component fine

```
mat.pc
```

```
Call:
princomp(covmat = .)

Standard deviations:
    Comp.1     Comp.2     Comp.3
1.71826118 0.21544865 0.03406486

 3  variables and  NA observations.
```

```
# ggscreeplot(mat.pc)
```

Component loadings

Compare correlation matrix:

```
mat
```

```
# A tibble: 3 x 3
      X1     X2     X3
   <dbl>  <dbl>  <dbl>
1  1      0.970 -0.96
2  0.970  1     -0.998
3 -0.96  -0.998  1
```

with component loadings

```
mat.pc$loadings
```

```
Loadings:
   Comp.1 Comp.2 Comp.3
X1  0.573  0.812  0.112
X2  0.581 -0.306 -0.755
X3 -0.578  0.498 -0.646

              Comp.1 Comp.2 Comp.3
SS loadings    1.000  1.000  1.000
```

Comments

- When X1 large, X2 also large, X3 small.
  - Then comp.1 *positive*.
- When X1 small, X2 small, X3 large.
  - Then comp.1 *negative*.

No scores

- With correlation matrix rather than data, no component scores
  - ▶ So no principal component plot
  - ▶ and no biplot.