Factor Analysis

Principal components and factor analysis

- Principal components:
  - Purely mathematical.
  - Find eigenvalues, eigenvectors of correlation matrix.
  - No testing whether observed components reproducible, or even probability model behind it.
- Factor analysis:
  - some way towards fixing this (get test of appropriateness)
  - In factor analysis, each variable modelled as: "common factor" (eg. verbal ability) and "specific factor" (left over).
  - Choose the common factors to "best" reproduce pattern seen in correlation matrix.
  - Iterative procedure, different answer from principal components.

Packages

```
library(ggbiplot)
library(tidyverse)
```

Example

- 145 children given 5 tests, called PARA, SENT, WORD, ADD and DOTS. 3 linguistic tasks (paragraph comprehension, sentence completion and word meaning), 2 mathematical ones (addition and counting dots).
- Correlation matrix of scores on the tests:

```
para 1     0.722 0.714 0.203 0.095
sent 0.722 1     0.685 0.246 0.181
word 0.714 0.685 1     0.170 0.113
add  0.203 0.246 0.170 1     0.585
dots 0.095 0.181 0.113 0.585 1
```

- Is there small number of underlying "constructs" (unobservable) that explains this pattern of correlations?

To start: principal components

Using correlation matrix. Read that first:

```
my_url <- "http://ritsokiguess.site/datafiles/rex2.txt"
kids <- read_delim(my_url, " ")
kids
```

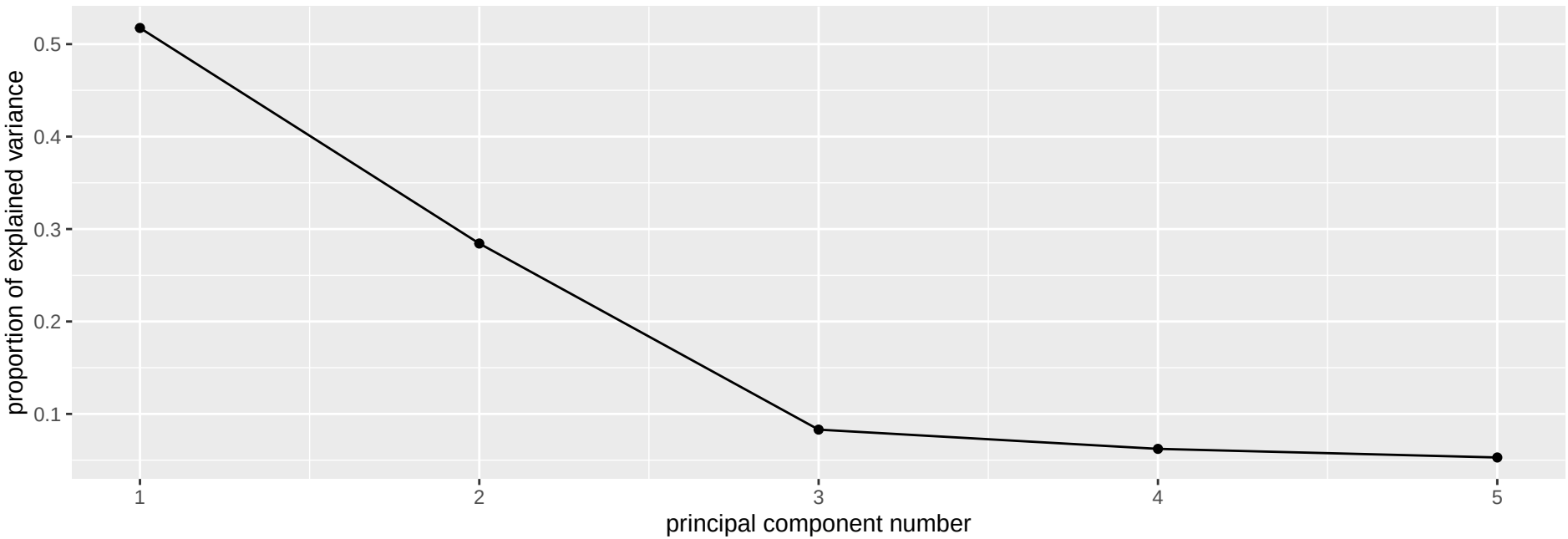
test	para	sent	word	add	dots
para	1.000	0.722	0.714	0.203	0.095
sent	0.722	1.000	0.685	0.246	0.181
word	0.714	0.685	1.000	0.170	0.113
add	0.203	0.246	0.170	1.000	0.585
dots	0.095	0.181	0.113	0.585	1.000

Principal components on correlation matrix

```
kids %>%
  select(where(is.numeric)) %>%
  as.matrix() %>%
  princomp(covmat = .) -> kids.pc
```

Scree plot

```
ggscreeplot(kids.pc)
```



Principal component results

- Need 2 components. Loadings:

```
kids.pc$loadings
```

```
##
## Loadings:
##       Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## para   0.534  0.245  0.114          0.795
## sent   0.542  0.164         0.660 -0.489
## word   0.523  0.247 -0.144 -0.738 -0.316
## add    0.297 -0.627  0.707
## dots   0.241 -0.678 -0.680         0.143
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0
```

Comments

- First component has a bit of everything, though especially the first three tests.
- Second component rather more clearly `add` and `dots`.
- No scores, plots since no actual data.

Factor analysis

- Specify number of factors first, get solution with exactly that many factors.
- Includes hypothesis test, need to specify how many children wrote the tests.
- Works from correlation matrix via `covmat` or actual data, like `princomp`.
- Introduces extra feature, *rotation*, to make interpretation of loadings (factor-variable relation) easier.

Factor analysis for the kids data

- Create "covariance list" to include number of children who wrote the tests.
- Feed this into `factanal`, specifying how many factors (2).

```
km <- kids %>%
  select(where(is.numeric)) %>%
  as.matrix()
km2 <- list(cov = km, n.obs = 145)
kids.f2 <- factanal(factors = 2, covmat = km2)
```

Uniquenesses

```
kids.f2$uniquenesses
```

```
##      para      sent      word       add      dots
## 0.2424457 0.2997349 0.3272312 0.5743568 0.1554076
```

- Uniquenesses say how "unique" a variable is (size of specific factor). Small uniqueness means that the variable is summarized by a factor (good).
- Very large uniquenesses are bad; add's uniqueness is largest but not large enough to be worried about.
- Also see "communality" for this idea, where *large* is good and *small* is bad.

Loadings

```
kids.f2$loadings
```

```
##
## Loadings:
##      Factor1 Factor2
## [1,] 0.867
## [2,] 0.820   0.166
## [3,] 0.816
## [4,] 0.167   0.631
## [5,]         0.918
##
##                Factor1 Factor2
## SS loadings      2.119   1.282
## Proportion Var   0.424   0.256
## Cumulative Var   0.424   0.680
```

- Loadings show how each factor depends on variables. Blanks indicate "small", less than 0.1.

Comments

- Factor 1 clearly the "linguistic" tasks, factor 2 clearly the "mathematical" ones.
- Two factors together explain 68% of variability (like regression R-squared).
- Which variables belong to which factor is *much* clearer than with principal components.

Are 2 factors enough?

```
kids.f2$STATISTIC
```

```
## objective
## 0.5810578
```

```
kids.f2$dof
```

```
## [1] 1
```

```
kids.f2$PVAL
```

```
## objective
##  0.445898
```

P-value not small, so 2 factors OK.

1 factor

```
kids.f1 <- factanal(factors = 1, covmat = km2)
kids.f1$STATISTIC
```

```
## objective
##  58.16534
```

```
kids.f1$dof
```

```
## [1] 5
```

```
kids.f1$PVAL
```

```
##    objective
## 2.907856e-11
```

1 factor rejected (P-value small). Definitely need more than 1.

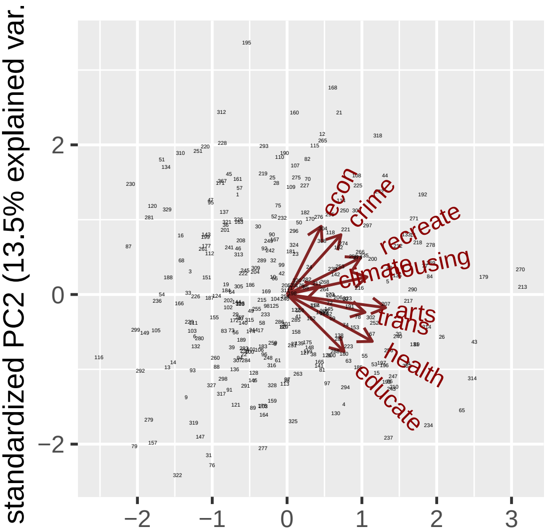
Places rated, again

Read data, transform, rerun principal components, get biplot:

```
my_url <- "http://ritsokiguess.site/datafiles/places.txt"
places0 <- read_table2(my_url)
places0 %>%
  mutate(across(-id, ~log(.))) -> places
places %>% select(-id) -> places_numeric
places.1 <- princomp(places_numeric, cor = TRUE)
g <- ggbiplot(places.1, labels = places$id,
         labels.size = 0.8)
```

The biplot

```
g
```


standardized PC2 (13.5% explained var.
econ
crime
recreate
housing
climate
arts
trans
health
educate
2
0
−2
−2
−1
0
1
2
3

Comments

- Most of the criteria are part of components 1 *and* 2.
- If we can rotate the arrows counterclockwise:
  - economy and crime would point straight up
    - part of component 2 only
  - health and education would point to the right
    - part of component 1 only
- would be easier to see which variables belong to which component.
- Factor analysis includes a rotation to help with interpretation.

Factor analysis

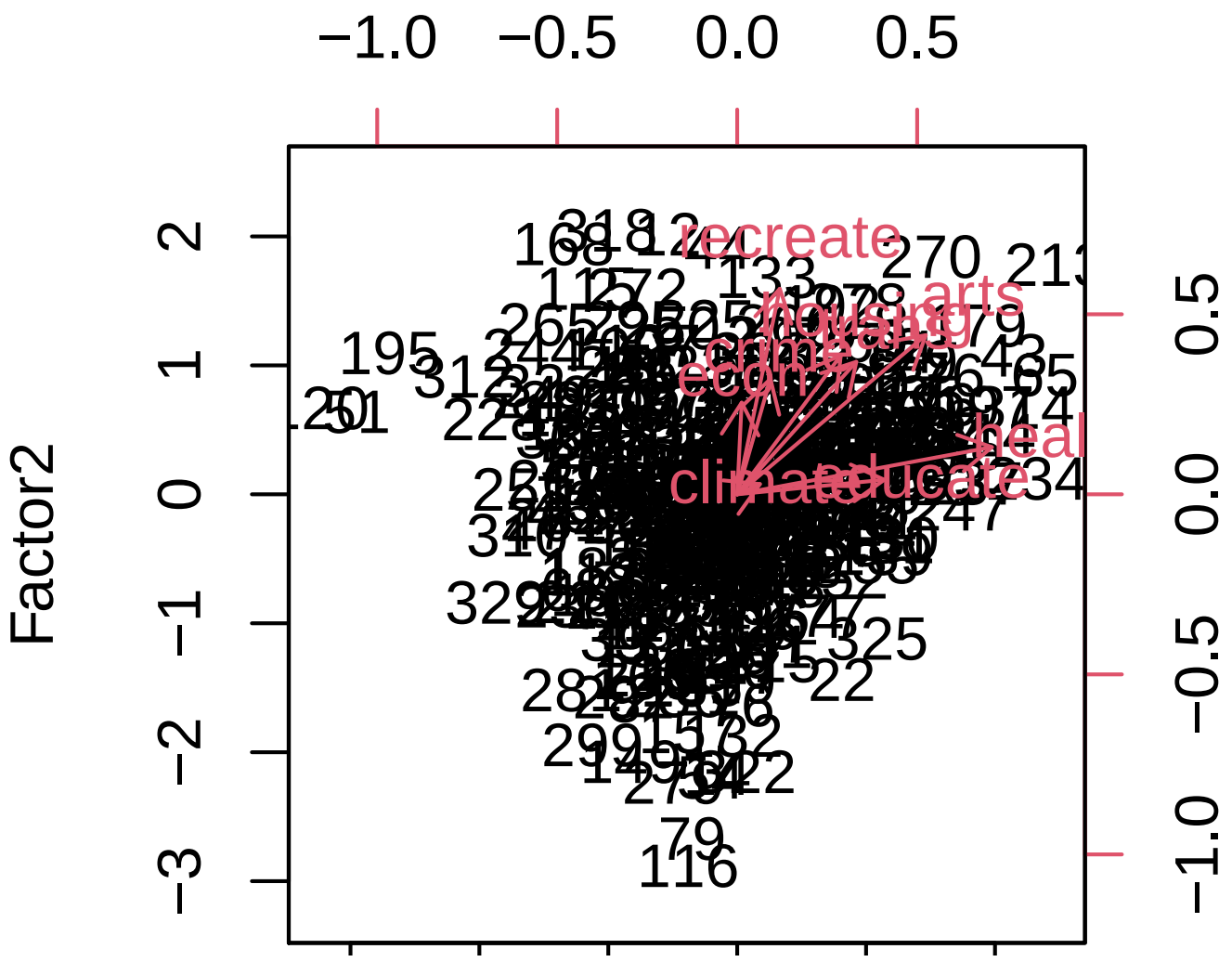
- Have to pick a number of factors *first*.
- Do this by running principal components and looking at scree plot.
- In this case, 3 factors seemed good (revisit later):

```
places.3 <- factanal(places_numeric, 3, scores = "r")
```

- There are different ways to get factor scores. These called "regression" scores.

A bad biplot

```
biplot(places.3$scores, places.3$loadings,
  xlabs = places$id)
```



Comments

- I have to find a way to make a better biplot!
- Some of the variables now point straight up and some straight across (if you look carefully for the red arrows among the black points).
- This should make the factors more interpretable than the components were.

Factor loadings

```
places.3$loadings
```

```
##
## Loadings:
##          Factor1 Factor2 Factor3
## climate                   0.994
## housing   0.360   0.482   0.229
## health    0.884   0.164
## crime     0.115   0.400   0.205
## trans     0.414   0.460
## educate   0.511
## arts      0.655   0.552   0.102
## recreate  0.148   0.714
## econ              0.318  -0.114
##
##                Factor1 Factor2 Factor3
## SS loadings      1.814   1.551   1.120
## Proportion Var   0.202   0.172   0.124
## Cumulative Var   0.202   0.374   0.498
```

Comments on loadings

- These are at least somewhat clearer than for the principal components:
  - Factor 1: health, education, arts: "well-being"
  - Factor 2: housing, transportation, arts (again), recreation: "places to be"
  - Factor 3: climate (only): "climate"
- In this analysis, economic factors don't seem to be important.

Factor scores

- Make a dataframe with the city IDs and factor scores:

```
cbind(id = places$id, places.3$scores) %>%
  as_tibble() -> places_scores
```

- Make percentile ranks again (for checking):

```
places %>%
  mutate(across(-id, ~percent_rank(.))) -> places_pr
```

Highest scores on factor 1, “well-being”:

```
places_scores %>%
  arrange(desc(Factor1))
```

id	Factor1	Factor2	Factor3
213	2.4695536	1.7788333	0.5060550
65	2.3857287	0.9247505	-0.2870941
234	2.3243460	0.1216503	0.5239580
314	2.2214264	0.6714295	0.5208330
43	2.1462172	1.0762072	0.4626511
214	1.9215857	0.4619138	0.3683036
197	1.9096321	0.6731837	-2.3923387
179	1.8550948	1.3074395	1.8249435
237	1.8219533	0.0833054	0.2870881
86	1.8008173	0.3510770	-0.0579466
247	1.7211705	-0.1131002	0.6808140
26	1.6604379	0.9236214	0.1460173

Check percentile ranks for factor 1

```
places_pr %>%
  select(id, health, educate, arts) %>%
  filter(id %in% c(213, 65, 234, 314))
```

id	health	educate	arts
65	0.9969512	0.9634146	0.9969512
213	1.0000000	0.7225610	1.0000000
234	0.9908537	1.0000000	0.9847561
314	0.9847561	0.9939024	0.9908537

- These are definitely high on the well-being variables.
- City #213 is not so high on education, but is highest of all on the others.

Highest scores on factor 2, “places to be”:

```
places_scores %>%
  arrange(desc(Factor2))
```

id	Factor1	Factor2	Factor3
318	-1.0124822	2.0460992	-0.0957484
12	-0.5397677	2.0192782	-3.8038802
168	-1.3485804	1.9401265	0.2734326
44	-0.1492306	1.9166960	-0.5558097
270	1.5031372	1.8418356	1.9389055
213	2.4695536	1.7788333	0.5060550
133	0.2269507	1.6857094	1.1329518
115	-1.1651392	1.5944350	0.2917337
272	-0.7763836	1.5892587	1.8887300
278	0.9295552	1.4684748	1.5394929
192	0.7204418	1.4568717	0.6308712
225	-0.2570416	1.3404700	-0.5442111

Check percentile ranks for factor 2

```
places_pr %>%
  select(id, housing, trans, arts, recreate) %>%
  filter(id %in% c(318, 12, 168, 44))
```

id	housing	trans	arts	recreate
12	0.9329268	0.7286585	0.6036585	0.8963415
44	0.9268293	0.9634146	0.7347561	0.9878049
168	0.8323171	0.8719512	0.4420732	0.9786585
318	0.8810976	0.7439024	0.6676829	0.9634146

- These are definitely high on housing and recreation.
- Some are (very) high on transportation, but not so much on arts.
- Could look at more cities to see if #168 being low on arts is a fluke.

Highest scores on factor 3, “climate”:

```
places_scores %>%
  arrange(desc(Factor3))
```

id	Factor1	Factor2	Factor3
227	-0.1838076	0.3850312	2.0395358
218	0.8809663	0.8966425	2.0211774
269	0.9315705	1.1891810	1.9772204
270	1.5031372	1.8418356	1.9389055
11	0.6884086	1.0489874	1.9248548
265	-1.4612289	1.3154920	1.8988849
272	-0.7763836	1.5892587	1.8887300
179	1.8550948	1.3074395	1.8249435
273	0.1196689	0.5806560	1.7998210
47	-1.5660795	-0.2601967	1.7993543
271	0.8425911	0.8608706	1.7638375
308	0.0754164	0.1883592	1.7144406

Check percentile ranks for factor 3

```
places_pr %>%
  select(id, climate) %>%
  filter(id %in% c(227, 218, 269, 270))
```

id	climate
218	0.9969512
227	0.9908537
269	0.9939024
270	0.9969512

This is very clear.

Uniquenesses

- We said earlier that the economy was not part of any of our factors:

```
places.3$uniquenesses
```

```
##   climate   housing    health     crime     trans
## 0.0050000 0.5859175 0.1854084 0.7842407 0.6165449
##   educate      arts  recreate      econ
## 0.7351921 0.2554663 0.4618143 0.8856382
```

- The higher the uniqueness, the less the variable concerned is part of any of our factors (and that maybe another factor is needed to accommodate it).
- This includes economy and maybe crime.

Test of significance

We can test whether the three factors that we have is enough, or whether we need more to describe our data:

```
places.3$PVAL
```

```
##    objective
## 1.453217e-14
```

- 3 factors are not enough.
- What would 5 factors look like?

Five factors

```
places.5 <- factanal(places_numeric, 5, scores = "r")
places.5$loadings
```

```
##
## Loadings:
##          Factor1 Factor2 Factor3 Factor4 Factor5
## climate                           0.131   0.559
## housing   0.286   0.505   0.289  -0.113   0.475
## health    0.847   0.214                    0.187
## crime             0.196   0.143   0.948   0.181
## trans     0.389   0.515           0.175
## educate   0.534
## arts      0.611   0.564           0.172   0.145
## recreate          0.705           0.115   0.136
## econ                      0.978   0.135
##
##                Factor1 Factor2 Factor3 Factor4
```

Comments

- On (new) 5 factors:
  - Factor 1 is health, education, arts: same as factor 1 before.
  - Factor 2 is housing, transportation, arts, recreation: as factor 2 before.
  - Factor 3 is economy.
  - Factor 4 is crime.
  - Factor 5 is climate and housing: like factor 3 before.
- The two added factors include the two "missing" variables.
- Is this now enough?

```
places.5$PVAL
```

```
##    objective
## 0.0009741394
```

- No. My guess is that the authors of Places Rated chose their 9 criteria to capture different aspects of what makes a city good or bad to live in, and so it was too much to hope that a small number of factors would come out of these.

A bigger example: BEM sex role inventory

- 369 women asked to rate themselves on 60 traits, like "self-reliant" or "shy".
- Rating 1 "never or almost never true of me" to 7 "always or almost always true of me' '.
- 60 personality traits is a lot. Can we find a smaller number of factors that capture aspects of personality?
- The whole BEM sex role inventory on next page.

The whole inventory

1. self reliant
2. yielding
3. helpful
4. defends own beliefs
5. cheerful
6. moody
7. independent
8. shy
9. conscientious
10. athletic
11. affectionate
12. theatrical
13. assertive
14. flatterable
15. happy
16. strong personality
17. loyal
18. unpredictable
19. forceful
20. feminine
21. reliable
22. analytical
23. sympathetic
24. jealous
25. leadership ability
26. sensitive to other's needs
27. truthful
28. willing to take risks
29. understanding
30. secretive
31. makes decisions easily
32. compassionate
33. sincere
34. self-sufficient
35. eager to soothe hurt feelings
36. conceited
37. dominant
38. soft spoken
39. likable
40. masculine
41. warm
42. solemn
43. willing to take a stand
44. tender
45. friendly
46. aggressive
47. gullible
48. inefficient
49. acts as a leader
50. childlike
51. adaptable
52. individualistic
53. does not use harsh language
54. unsystematic
55. competitive
56. loves children
57. tactful
58. ambitious
59. gentle
60. conventional

Some of the data

```
my_url <- "http://ritsokiguess.site/datafiles/factor.txt"
bem <- read_tsv(my_url)
bem
```

subno	help-ful	re-liant	def-bel	yield-ing	cheer-ful	in-dpt	ath-let	as-sert	str-pers	force-ful	af-fect	flat-ter	loyal	an-a-lyt	fem-i-nine	sym-pa-thy	mood	sen-si-tive	und-stand	com-pass	lead-er	ab-soot
1	7	7	5	5	7	7	7	1	7	7	2	7	4	7	1	2	6	3	7	7	7	6	7
2	5	6	6	6	2	3	3	3	4	1	3	5	4	5	6	5	5	4	5	5	4	4	4
3	7	6	4	4	5	5	2	3	4	4	3	5	2	7	5	6	5	2	4	6	6	4	7
4	6	6	7	4	6	6	3	4	4	3	3	5	3	7	6	6	5	2	6	6	6	2	6
5	6	6	7	4	7	7	7	2	7	7	5	7	4	7	7	4	7	3	7	6	7	7	7
7	5	6	7	4	6	6	2	4	4	2	2	5	3	7	7	7	6	4	5	5	5	3	5
8	6	4	6	6	6	3	1	3	3	4	1	7	7	6	4	4	7	3	6	6	6	1	6
9	7	6	7	5	6	7	5	2	5	6	6	7	3	7	7	6	6	4	6	6	6	6	5
10	7	6	6	4	4	5	2	2	5	7	6	6	6	6	7	6	4	4	6	6	5	5	5
11	7	4	7	4	7	5	2	1	5	6	4	7	7	7	7	5	5	1	5	6	6	4	6
13	7	7	7	4	6	7	1	3	6	6	4	6	5	7	7	5	7	6	6	7	7	5	6
14	7	7	5	5	7	1	3	5	6	7	1	7	7	7	3	5	7	5	7	7	7	2	7
15	7	7	7	7	7	7	7	1	7	7	7	7	5	7	7	7	7	1	7	7	7	7	7
23	5	6	7	5	6	6	1	1	6	6	7	7	7	7	7	6	6	4	6	6	5	4	4
25	6	6	7	4	4	7	7	1	5	5	7	6	1	7	4	4	7	5	4	7	5	4	5
26	6	6	5	5	7	6	2	2	5	5	4	4	5	7	7	4	6	2	6	6	5	6	6

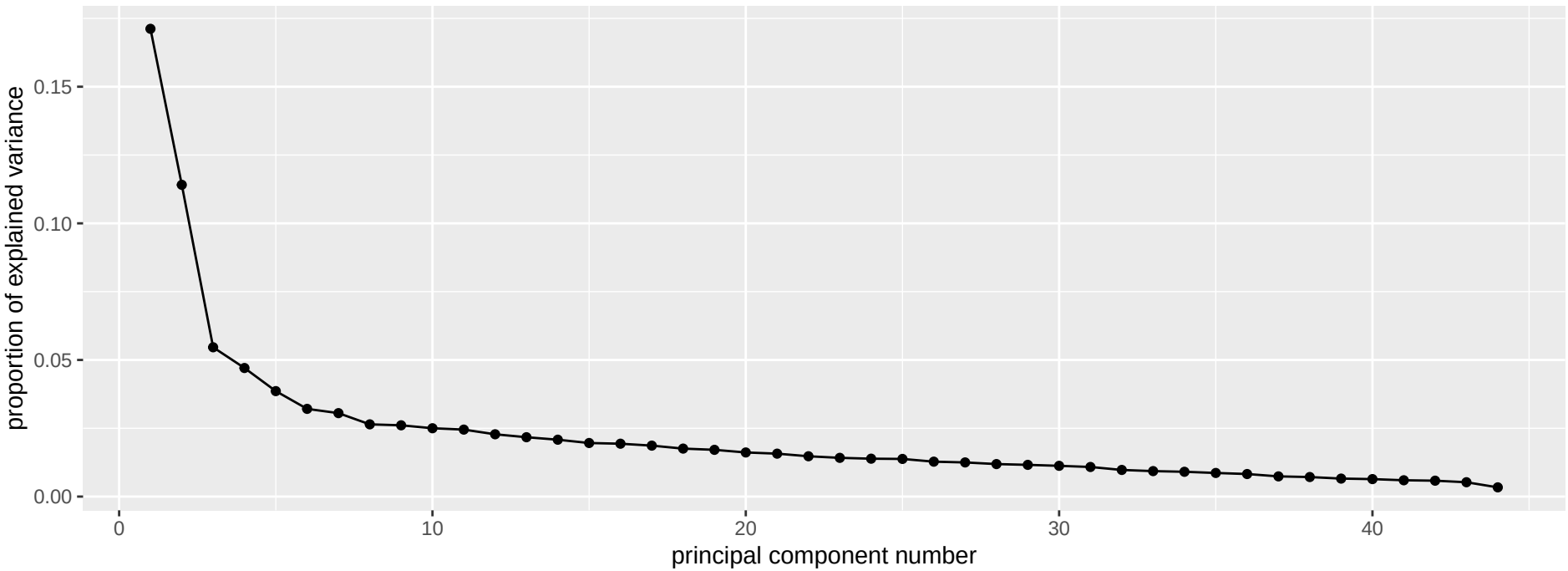
Principal components first

...to decide on number of factors:

```
bem.pc <- bem %>%
  select(-subno) %>%
  princomp(cor = T)
```

The scree plot

```
(g <- ggscreeplot(bem.pc))
```

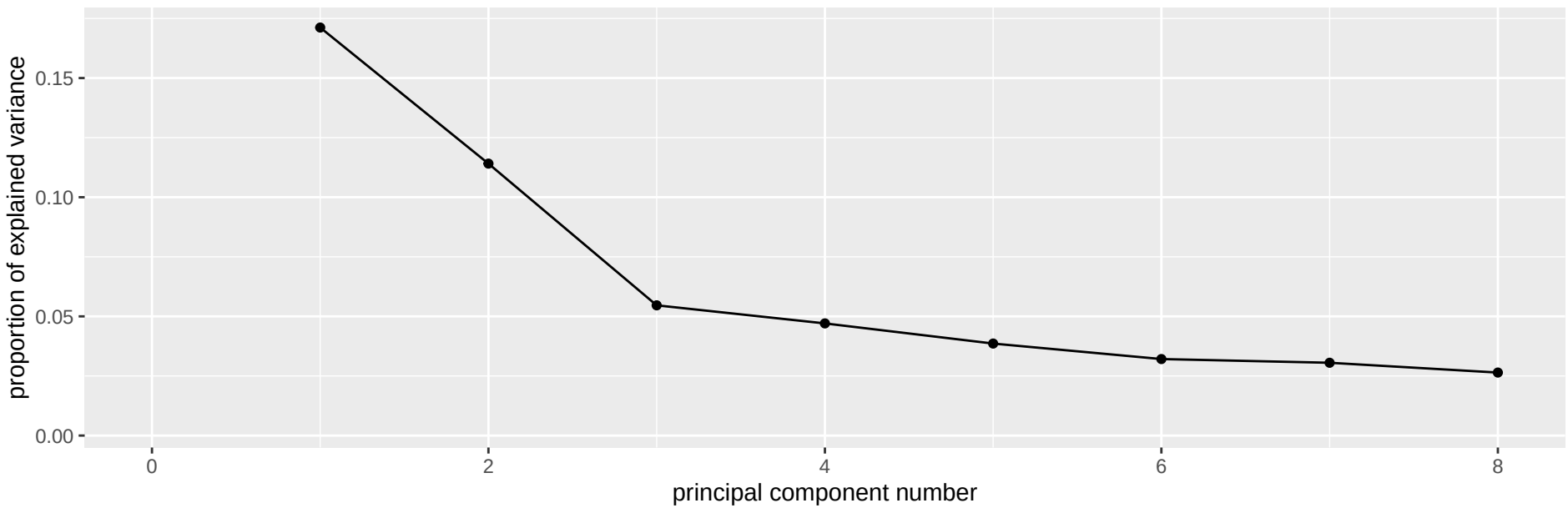


- No obvious elbow.

Zoom in to search for elbow

Possible elbows at 3 (2 factors) and 6 (5):

```
g + scale_x_continuous(limits = c(0, 8))
```



but is 2 really good?

```
summary(bem.pc)
```

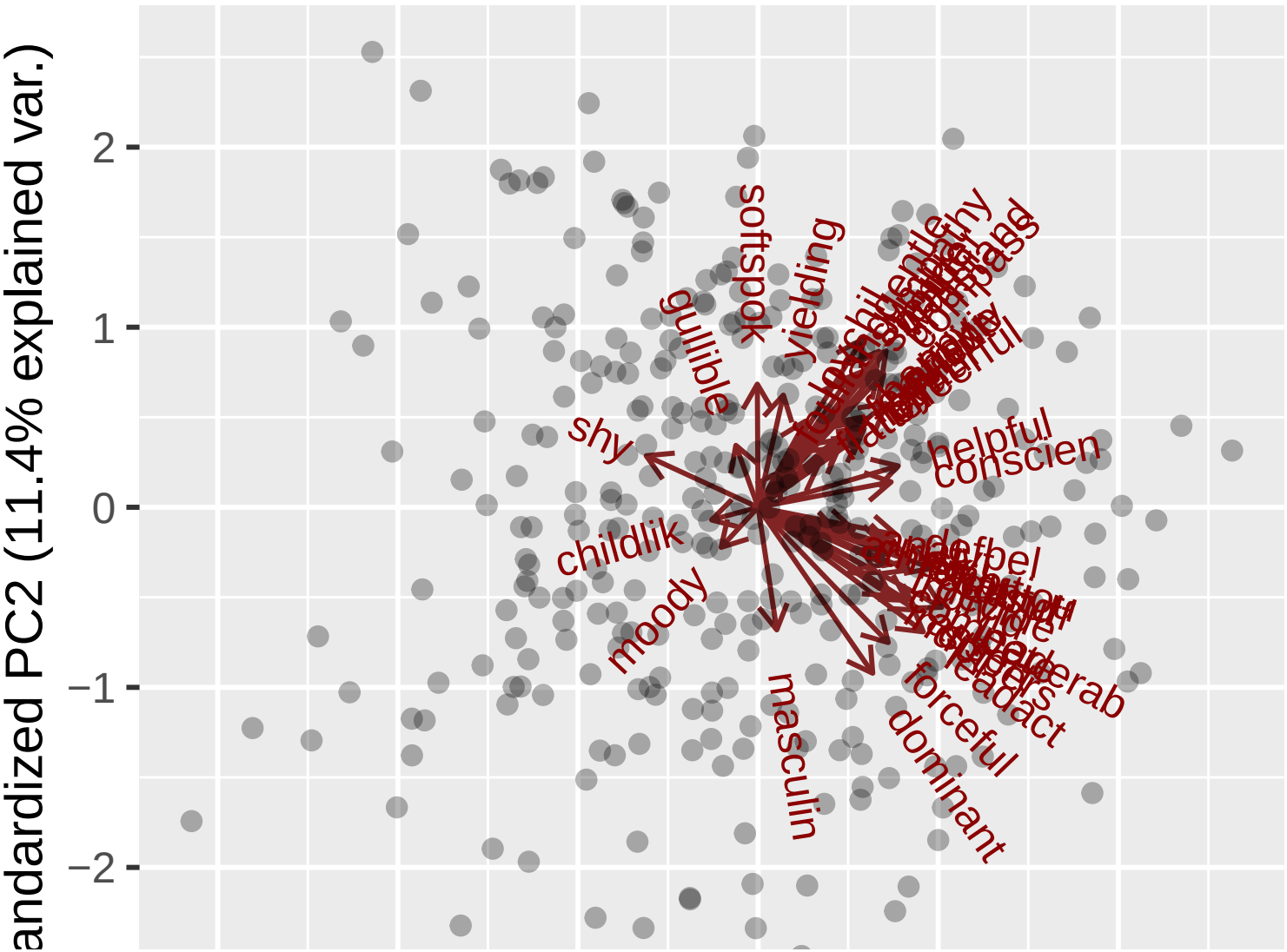
```
## Importance of components:
##                            Comp.1    Comp.2     Comp.3     Comp.4     Comp.5
## Standard deviation     2.7444993 2.2405789 1.55049106 1.43886350 1.30318840
## Proportion of Variance 0.1711881 0.1140953 0.05463688 0.04705291 0.03859773
## Cumulative Proportion  0.1711881 0.2852834 0.33992029 0.38697320 0.42557093
##                             Comp.6     Comp.7     Comp.8     Comp.9    Comp.10
## Standard deviation     1.18837867 1.15919129 1.07838912 1.07120568 1.04901318
## Proportion of Variance 0.03209645 0.03053919 0.02643007 0.02607913 0.02500974
## Cumulative Proportion  0.45766738 0.48820657 0.51463664 0.54071577 0.56572551
##                            Comp.11    Comp.12    Comp.13    Comp.14   Comp.15
## Standard deviation     1.03848656 1.00152287 0.97753974 0.95697572 0.9287543
## Proportion of Variance 0.02451033 0.02279655 0.02171782 0.02081369 0.0196042
## Cumulative Proportion  0.59023584 0.61303238 0.63475020 0.65556390 0.6751681
##                            Comp.16    Comp.17   Comp.18    Comp.19    Comp.20
## Standard deviation     0.92262649 0.90585705 0.8788668 0.86757525 0.84269120
## Proportion of Variance 0.01934636 0.01864948 0.0175547 0.01710652 0.01613928
## Cumulative Proportion  0.69451445 0.71316392 0.7307186 0.74782514 0.76396443
##                            Comp.21    Comp.22    Comp.23    Comp.24    Comp.25
## Standard deviation     0.83124925 0.80564654 0.78975423 0.78100835 0.77852606
## Proportion of Variance 0.01570398 0.01475151 0.01417527 0.01386305 0.01377506
## Cumulative Proportion  0.77966841 0.79441992 0.80859519 0.82245823 0.83623330
##                            Comp.26    Comp.27    Comp.28    Comp.29    Comp.30
```

Comments

- Want overall fraction of variance explained ("cumulative proportion' ') to be reasonably high.
- 2 factors, 28.5%. Terrible!
- Even 56% (10 factors) not that good!
- Have to live with that.

Biplot

```
ggbiplot(bem.pc, alpha = 0.3)
```



Comments

- Ignore individuals for now.
- Most variables point to 1 o'clock or 4 o'clock.
- Suggests factor analysis with rotation will get interpretable factors (rotate to 12 o'clock and 3 o'clock, for example).
- Try for 2-factor solution (rough interpretation, will be bad):

```
bem.2 <- bem %>%
  select(-subno) %>%
  factanal(factors = 2)
```

- Show output in pieces (just print `bem.2` to see all of it).

Uniquenesses, sorted

```
sort(bem.2$uniquenesses)
```

```
##  leaderab   leadact      warm    tender  dominant    gentle
## 0.4091894 0.4166153 0.4764762 0.4928919 0.4942909 0.5064551
##  forceful   strpers   compass     stand  undstand    assert
## 0.5631857 0.5679398 0.5937073 0.6024001 0.6194392 0.6329347
##    soothe    affect    decide  selfsuff  sympathy     indpt
## 0.6596103 0.6616625 0.6938578 0.7210246 0.7231450 0.7282742
##   helpful    defbel      risk   reliant   individ   compete
## 0.7598223 0.7748448 0.7789761 0.7808058 0.7941998 0.7942910
##  conscien     happy  sensitiv     loyal  ambitiou       shy
## 0.7974820 0.8008966 0.8018851 0.8035264 0.8101599 0.8239496
##  softspok  cheerful  masculin  yielding  feminine  truthful
## 0.8339058 0.8394916 0.8453368 0.8688473 0.8829927 0.8889983
##   lovchil    analyt    athlet   flatter  gullible     moody
## 0.8924392 0.8968744 0.9229702 0.9409500 0.9583435 0.9730607
##  childlik  foullang
## 0.9800360 0.9821662
```

Comments

- Mostly high or very high (bad).
- Some smaller, eg.: Leadership ability (0.409), Acts like leader (0.417), Warm (0.476), Tender (0.493).
- Smaller uniquenesses captured by one of our two factors.
- Larger uniquenesses are not: need more factors to capture them.

Factor loadings, some

```
bem.2$loadings
```

```
##
## Loadings:
##          Factor1 Factor2
## helpful   0.314   0.376
## reliant   0.453   0.117
## defbel    0.434   0.193
## yielding -0.131   0.338
## cheerful  0.152   0.371
## indpt     0.521
## athlet    0.267
## shy      -0.414
## assert    0.605
## strpers   0.657
## forceful  0.649  -0.126
## affect    0.178   0.554
## flatter           0.223
## loyal     0.151   0.417
## analyt    0.295   0.127
## feminine  0.113   0.323
## sympathy          0.526
## moody            -0.162
## sensitiv  0.135   0.424
```

Making a data frame

There are too many to read easily, so make a data frame. A bit tricky:

```
loadings <- as.data.frame(unclass(bem.2$loadings)) %>%
  mutate(trait = rownames(bem.2$loadings))
loadings %>% slice(1:12)
```

	Factor1	Factor2	trait
helpful	0.3137466	0.3764849	helpful
reliant	0.4532904	0.1171406	reliant
defbel	0.4336574	0.1926030	defbel
yielding	-0.1309965	0.3376293	yielding
cheerful	0.1523718	0.3705305	cheerful
indpt	0.5212403	0.0058703	indpt
athlet	0.2670788	0.0755429	athlet
shy	-0.4144579	-0.0653728	shy
assert	0.6049588	0.0330048	assert
strpers	0.6569855	0.0207776	strpers
forceful	0.6487190	-0.1264058	forceful
affect	0.1778911	0.5537994	affect

Pick out the big ones on factor 1

Arbitrarily defining > 0.4 or < -0.4 as "big":

```
loadings %>% filter(abs(Factor1) > 0.4)
```

	Factor1	Factor2	trait
reliant	0.4532904	0.1171406	reliant
defbel	0.4336574	0.1926030	defbel
indpt	0.5212403	0.0058703	indpt
shy	-0.4144579	-0.0653728	shy
assert	0.6049588	0.0330048	assert
strpers	0.6569855	0.0207776	strpers
forceful	0.6487190	-0.1264058	forceful
leaderab	0.7654924	0.0695136	leaderab
risk	0.4416176	0.1612384	risk
decide	0.5416796	0.1128080	decide
selfsuff	0.5109964	0.1336268	selfsuff
dominant	0.6676490	-0.2448558	dominant
stand	0.6066864	0.1718489	stand
leadact	0.7627129	-0.0406672	leadact
individ	0.4448064	0.0891461	individ
compete	0.4504188	0.0532073	compete
ambitiou	0.4136498	0.1368696	ambitiou

Factor 2, the big ones

```
loadings %>% filter(abs(Factor2) > 0.4)
```

	Factor1	Factor2	trait
affect	0.1778911	0.5537994	affect
loyal	0.1512127	0.4166622	loyal
sympathy	0.0230146	0.5256654	sympathy
sensitiv	0.1347697	0.4242037	sensitiv
undstand	0.0911130	0.6101294	undstand
compass	0.1135064	0.6272223	compass
soothe	0.0606175	0.5802714	soothe
happy	0.1189301	0.4300698	happy
warm	0.0795698	0.7191610	warm
tender	0.0511381	0.7102763	tender
gentle	-0.0187322	0.7022768	gentle

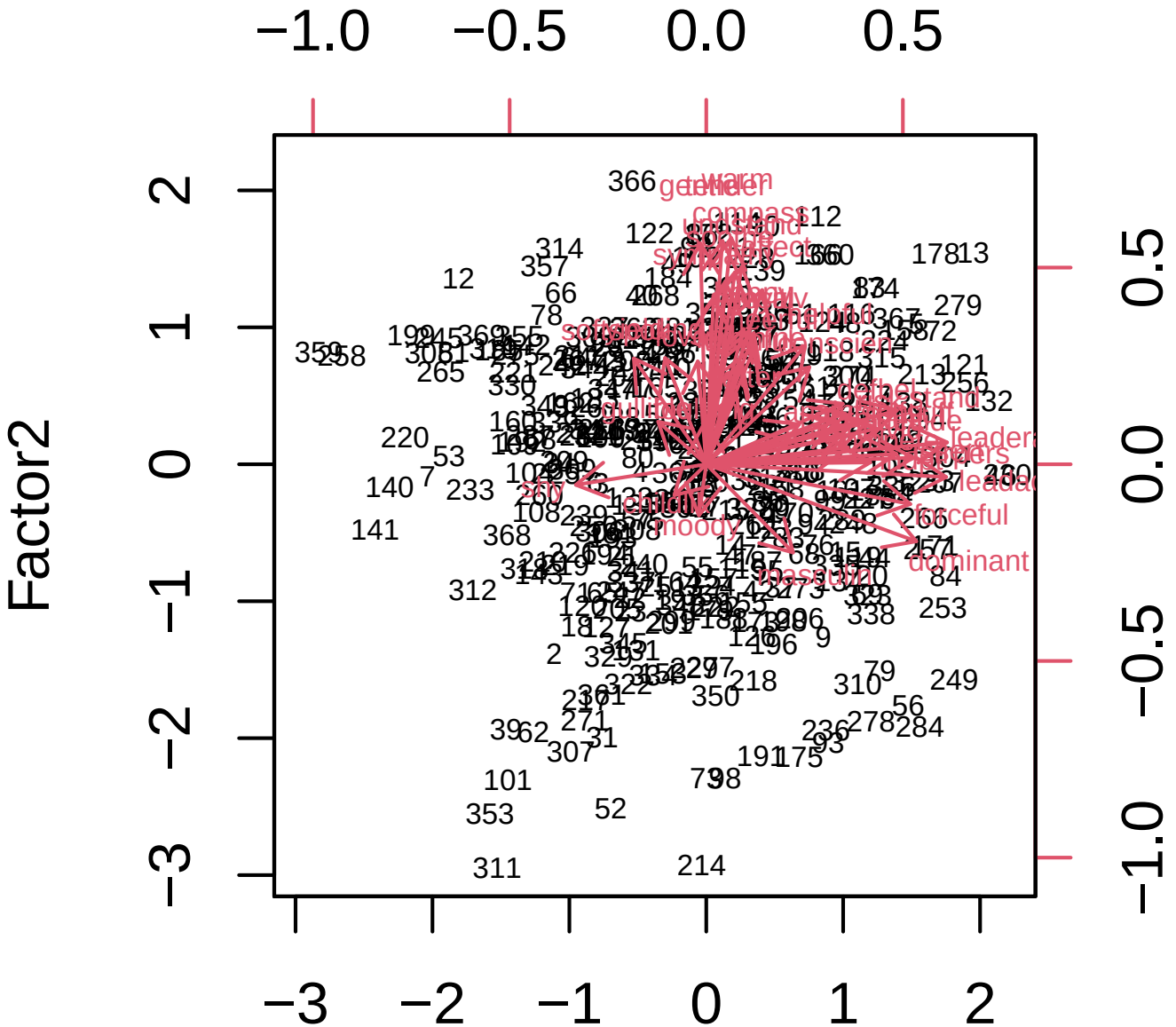
Plotting the two factors

- A bi-plot, this time with the variables reduced in size. Looking for unusual individuals.
- Have to run `factanal` again to get factor scores for plotting.

```
bem %>% select(-subno) %>%
  factanal(factors = 2, scores = "r") -> bem.2a
biplot(bem.2a$scores, bem.2a$loadings, cex = c(0.5, 0.5))
```

- Numbers on plot are row numbers of `bem` data frame.

The (awful) biplot



Comments

- Variables mostly up ("feminine") and right ("masculine"), accomplished by rotation.
- Some unusual individuals: 311, 214 (low on factor 2), 366 (high on factor 2), 359, 258 (low on factor 1), 230 (high on factor 1).

Individual 366

```
bem %>% slice(366) %>% glimpse()
```

```
## Rows: 1
## Columns: 45
## $ subno    <dbl> 755
## $ helpful  <dbl> 7
## $ reliant  <dbl> 7
## $ defbel   <dbl> 5
## $ yielding <dbl> 7
## $ cheerful <dbl> 7
## $ indpt    <dbl> 7
## $ athlet   <dbl> 7
## $ shy      <dbl> 2
## $ assert   <dbl> 1
## $ strpers  <dbl> 3
## $ forceful <dbl> 1
## $ affect   <dbl> 7
## $ flatter  <dbl> 9
## $ loyal    <dbl> 7
## $ analyt   <dbl> 7
## $ feminine <dbl> 7
## $ sympathy <dbl> 7
## $ moody    <dbl> 1
## $ sensitiv <dbl> 7
## $ undstand <dbl> 7
## $ compass  <dbl> 6
## $ leaderab <dbl> 3
## $ soothe   <dbl> 7
## $ risk     <dbl> 7
## $ decide   <dbl> 7
## $ selfsuff <dbl> 7
## $ conscien <dbl> 7
```

Comments

- Individual 366 high on factor 2, but hard to see which traits should have high scores (unless we remember).
- Idea: *tidy* original data frame to make easier to look things up.

Tidying original data

```
bem %>%
  mutate(row = row_number()) %>%
  pivot_longer(c(-subno, -row), names_to="trait",
               values_to="score") -> bem_tidy
bem_tidy
```

subno	row	trait	score
1	1	helpful	7
1	1	reliant	7
1	1	defbel	5
1	1	yielding	5
1	1	cheerful	7
1	1	indpt	7
1	1	athlet	7
1	1	shy	1
1	1	assert	7
1	1	strpers	7
1	1	forceful	2
1	1	affect	7
1	1	flatter	4
1	1	loyal	7
1	1	analyt	1
1	1	feminine	2

Recall data frame of loadings

```
loadings %>% slice(1:10)
```

	Factor1	Factor2	trait
helpful	0.3137466	0.3764849	helpful
reliant	0.4532904	0.1171406	reliant
defbel	0.4336574	0.1926030	defbel
yielding	-0.1309965	0.3376293	yielding
cheerful	0.1523718	0.3705305	cheerful
indpt	0.5212403	0.0058703	indpt
athlet	0.2670788	0.0755429	athlet
shy	-0.4144579	-0.0653728	shy
assert	0.6049588	0.0330048	assert
strpers	0.6569855	0.0207776	strpers

Want to add the factor scores for each trait to our tidy data frame `bem_tidy`. This is a left-join (over), matching on the column `trait` that is in both data frames (thus, the default):

Looking up loadings

```
bem_tidy %>% left_join(loadings) -> bem_tidy
```

```
## Joining, by = "trait"
```

```
bem_tidy %>% sample_n(12)
```

subno	row	trait	score	Factor1	Factor2
385	221	warm	6	0.0795698	0.7191610
295	172	flatter	5	0.0964226	0.2230714
88	54	foullang	4	-0.0049285	0.1334296
349	206	gullible	2	-0.1528502	0.1352727
7	6	shy	4	-0.4144579	-0.0653728
3	3	softspok	4	-0.2303283	0.3362171
706	357	sympathy	7	0.0230146	0.5256654
501	288	foullang	2	-0.0049285	0.1334296
558	327	undstand	6	0.0911130	0.6101294
250	144	analyt	5	0.2949559	0.1270016
34	24	analyt	6	0.2949559	0.1270016
424	240	soothe	4	0.0606175	0.5802714

Individual 366, high on Factor 2

So now pick out the rows of the tidy data frame that belong to individual 366 (`row`=366) and for which the `Factor2` score exceeds 0.4 in absolute value (our "big" from before):

```
bem_tidy %>% filter(row == 366, abs(Factor2) > 0.4)
```

subno	row	trait	score	Factor1	Factor2
755	366	affect	7	0.1778911	0.5537994
755	366	loyal	7	0.1512127	0.4166622
755	366	sympathy	7	0.0230146	0.5256654
755	366	sensitiv	7	0.1347697	0.4242037
755	366	undstand	7	0.0911130	0.6101294
755	366	compass	6	0.1135064	0.6272223
755	366	soothe	7	0.0606175	0.5802714
755	366	happy	7	0.1189301	0.4300698
755	366	warm	7	0.0795698	0.7191610
755	366	tender	7	0.0511381	0.7102763
755	366	gentle	7	-0.0187322	0.7022768

As expected, high scorer on these.

Several individuals

Rows 311 and 214 were *low* on Factor 2, so their scores should be low. Can we do them all at once?

```
bem_tidy %>% filter(
  row %in% c(366, 311, 214),
  abs(Factor2) > 0.4
)
```

subno	row	trait	score	Factor1	Factor2
369	214	affect	1	0.1778911	0.5537994
369	214	loyal	7	0.1512127	0.4166622
369	214	sympathy	4	0.0230146	0.5256654
369	214	sensitiv	7	0.1347697	0.4242037
369	214	undstand	5	0.0911130	0.6101294
369	214	compass	5	0.1135064	0.6272223
369	214	soothe	3	0.0606175	0.5802714
369	214	happy	4	0.1189301	0.4300698
369	214	warm	1	0.0795698	0.7191610
369	214	tender	3	0.0511381	0.7102763
369	214	gentle	2	-0.0187322	0.7022768

Individual by column

Un-tidy, that is, `pivot_wider`:

```
bem_tidy %>%
  filter(
    row %in% c(366, 311, 214),
    abs(Factor2) > 0.4
  ) %>%
  select(-subno, -Factor1, -Factor2) %>%
  pivot_wider(names_from=row, values_from=score)
```

trait	214	311	366
affect	1	5	7
loyal	7	4	7
sympathy	4	4	7
sensitiv	7	4	7
undstand	5	3	7
compass	5	4	6
soothe	3	4	7
happy	4	3	7
warm	1	3	7
tender	3	4	7
gentle	2	3	7

366 high, 311 middling, 214 (sometimes) low.

Individuals 230, 258, 359

These were high, low, low on factor 1. Adapt code:

```
bem_tidy %>%
  filter(row %in% c(359, 258, 230), abs(Factor1) > 0.4) %>%
  select(-subno, -Factor1, -Factor2) %>%
  pivot_wider(names_from=row, values_from=score)
```

trait	230	258	359
reliant	7	4	1
defbel	7	1	1
indpt	7	7	1
shy	2	7	5
assert	7	3	1
strpers	7	1	3
forceful	7	1	1
leaderab	7	1	1
risk	7	5	7
decide	7	1	2
selfsuff	7	4	1
dominant	7	1	1
stand	7	1	6
leadact	7	1	1
individ	7	3	3
compete	6	2	1
ambitiou	7	2	4

Is 2 factors enough?

Suspect not:

```
bem.2$PVAL
```

```
##     objective
## 1.458183e-150
```

2 factors resoundingly rejected. Need more. Have to go all the way to 15 factors to not reject:

```
bem.15 <- bem %>%
  select(-subno) %>%
  factanal(factors = 15)
bem.15$PVAL
```

```
## objective
##  0.132617
```

Even then, only just over 50% of variability explained.

Get 15-factor loadings

into a data frame, as before:

```
loadings <- as.data.frame(unclass(bem.15$loadings)) %>%
  mutate(trait = rownames(bem.15$loadings))
```

then show the highest few loadings on each factor.

Factor 1 (of 15)

```
loadings %>%
  arrange(desc(abs(Factor1))) %>%
  select(Factor1, trait) %>%
  slice(1:10)
```

	Factor1	trait
compass	0.8127595	compass
undstand	0.6756043	undstand
sympathy	0.6611293	sympathy
sensitiv	0.6408327	sensitiv
soothe	0.5971006	soothe
warm	0.3481290	warm
gentle	0.2797159	gentle
tender	0.2788627	tender
helpful	0.2501505	helpful
conscien	0.2340594	conscien

Compassionate, understanding, sympathetic, soothing: thoughtful of others.

Factor 2

```
loadings %>%
  arrange(desc(abs(Factor2))) %>%
  select(Factor2, trait) %>%
  slice(1:10)
```

	Factor2	trait
strpers	0.7615492	strpers
forceful	0.7160312	forceful
assert	0.6981500	assert
dominant	0.5041921	dominant
leaderab	0.3929344	leaderab
stand	0.3669560	stand
leadact	0.3507080	leadact
softspok	-0.3131682	softspok
shy	-0.2866862	shy
analyt	0.2602525	analyt

Strong personality, forceful, assertive, dominant: getting ahead.

Factor 3

```
loadings %>%
  arrange(desc(abs(Factor3))) %>%
  select(Factor3, trait) %>%
  slice(1:10)
```

	Factor3	trait
reliant	0.6697542	reliant
selfsuff	0.6475496	selfsuff
indpt	0.6204018	indpt
helpful	0.3899607	helpful
gullible	-0.3393605	gullible
individ	0.3333813	individ
decide	0.3319003	decide
conscien	0.3294806	conscien
leaderab	0.2877396	leaderab
defbel	0.2804170	defbel

Self-reliant, self-sufficient, independent: going it alone.

Factor 4

```
loadings %>%
  arrange(desc(abs(Factor4))) %>%
  select(Factor4, trait) %>%
  slice(1:10)
```

	Factor4	trait
gentle	0.6956206	gentle
tender	0.6920303	tender
warm	0.5992467	warm
affect	0.4465546	affect
softspok	0.3942568	softspok
lovchil	0.2779793	lovchil
undstand	0.2444249	undstand
happy	0.2442119	happy
loyal	0.2125905	loyal
soothe	0.2022861	soothe

Gentle, tender, warm (affectionate): caring for others.

Factor 5

```
loadings %>%
  arrange(desc(abs(Factor5))) %>%
  select(Factor5, trait) %>%
  slice(1:10)
```

	Factor5	trait
compete	0.6956846	compete
ambitiou	0.6743459	ambitiou
risk	0.3453425	risk
individ	0.3423456	individ
athlet	0.2808623	athlet
leaderab	0.2695570	leaderab
decide	0.2449656	decide
dominant	0.2064415	dominant
leadact	0.1928159	leadact
strpers	0.1854989	strpers

Ambitious, competitive (with a bit of risk-taking and individualism): Being the best

Factor 6

```
loadings %>%
  arrange(desc(abs(Factor6))) %>%
  select(Factor6, trait) %>%
  slice(1:10)
```

	Factor6	trait
leadact	0.8675651	leadact
leaderab	0.6078869	leaderab
dominant	0.3378645	dominant
forceful	0.2014835	forceful
shy	-0.1915632	shy
risk	0.1789256	risk
masculin	0.1703440	masculin
decide	0.1639190	decide
compete	0.1594585	compete
athlet	0.1466037	athlet

Acts like a leader, leadership ability (with a bit of Dominant): Taking charge

Factor 7

```
loadings %>%
  arrange(desc(abs(Factor7))) %>%
  select(Factor7, trait) %>%
  slice(1:10)
```

	Factor7	trait
happy	0.6698996	happy
cheerful	0.6667105	cheerful
moody	-0.5219125	moody
athlet	0.2191425	athlet
warm	0.2126626	warm
gentle	0.1719953	gentle
masculin	-0.1640302	masculin
reliant	0.1601472	reliant
yielding	0.1472926	yielding
lovchil	0.1410481	lovchil

Acts like a leader, leadership ability (with a bit of Dominant): Taking charge

Factor 8

```
loadings %>%
  arrange(desc(abs(Factor8))) %>%
  select(Factor8, trait) %>%
  slice(1:10)
```

	Factor8	trait
affect	0.6296764	affect
flatter	0.5158355	flatter
softspok	-0.2512066	softspok
warm	0.2214623	warm
tender	0.1878549	tender
strpers	0.1846225	strpers
shy	-0.1804838	shy
compete	0.1801992	compete
loyal	0.1658105	loyal
helpful	0.1548617	helpful

Affectionate, flattering: Making others feel good.

Factor 9

```
loadings %>%
  arrange(desc(abs(Factor9))) %>%
  select(Factor9, trait) %>%
  slice(1:10)
```

	Factor9	trait
stand	0.8633171	stand
defbel	0.3403294	defbel
individ	0.2446971	individ
risk	0.1941110	risk
shy	-0.1715481	shy
decide	0.1710978	decide
assert	0.1197126	assert
conscien	0.1157729	conscien
analyt	0.1120308	analyt
gullible	-0.1115140	gullible

Taking a stand.

Factor 10

```
loadings %>%
  arrange(desc(abs(Factor10))) %>%
  select(Factor10, trait) %>%
  slice(1:10)
```

	Factor10	trait
feminine	0.8075127	feminine
masculin	-0.2637851	masculin
softspok	0.2450718	softspok
conscien	0.2317560	conscien
selfsuff	0.2019203	selfsuff
yielding	0.1758423	yielding
gentle	0.1412707	gentle
flatter	0.1128203	flatter
decide	0.1093453	decide
lovchil	-0.0940798	lovchil

Feminine. (A little bit of not-masculine!)

Factor 11

```
loadings %>%
  arrange(desc(abs(Factor11))) %>%
  select(Factor11, trait) %>%
  slice(1:10)
```

	Factor11	trait
loyal	0.9162259	loyal
affect	0.1894908	affect
truthful	0.1588386	truthful
helpful	0.1246453	helpful
analyt	0.1044066	analyt
tender	0.1007679	tender
lovchil	0.0972046	lovchil
gullible	0.0963522	gullible
cheerful	0.0935062	cheerful
conscien	0.0820760	conscien

Loyal.

Factor 12

```
loadings %>%
  arrange(desc(abs(Factor12))) %>%
  select(Factor12, trait) %>%
  slice(1:10)
```

	Factor12	trait
childlik	0.6106933	childlik
selfsuff	-0.2845004	selfsuff
conscien	-0.2786751	conscien
moody	0.2588843	moody
shy	0.2013245	shy
decide	-0.1669301	decide
masculin	0.1542031	masculin
dominant	0.1455526	dominant
compass	0.1379163	compass
leaderab	-0.1297408	leaderab

Childlike. (With a bit of moody, shy, not-self-sufficient, not-conscientious.)

Factor 13

```
loadings %>%
  arrange(desc(abs(Factor13))) %>%
  select(Factor13, trait) %>%
  slice(1:10)
```

	Factor13	trait
truthful	0.5729242	truthful
gullible	-0.2776490	gullible
happy	0.2631046	happy
warm	0.1885152	warm
shy	-0.1671924	shy
loyal	0.1646031	loyal
yielding	-0.1438127	yielding
assert	-0.1302900	assert
defbel	0.1137074	defbel
lovchil	-0.1105583	lovchil

Truthful. (With a bit of happy and not-gullible.)

Factor 14

```
loadings %>%
  arrange(desc(abs(Factor14))) %>%
  select(Factor14, trait) %>%
  slice(1:10)
```

	Factor14	trait
decide	0.4429926	decide
selfsuff	0.2369714	selfsuff
forceful	0.1945034	forceful
softspok	-0.1862756	softspok
risk	0.1604175	risk
strpers	-0.1484606	strpers
dominant	0.1461972	dominant
happy	0.1279456	happy
compass	0.1154479	compass
masculin	0.1054078	masculin

Decisive. (With a bit of self-sufficient and not-soft-spoken.)

Factor 15

```
loadings %>%
  arrange(desc(abs(Factor15))) %>%
  select(Factor15, trait) %>%
  slice(1:10)
```

	Factor15	trait
compass	-0.3244092	compass
athlet	0.2471884	athlet
sensitiv	0.2292980	sensitiv
risk	0.1986878	risk
affect	-0.1638296	affect
moody	0.1632164	moody
individ	-0.1118135	individ
warm	0.1100678	warm
cheerful	0.1047347	cheerful
reliant	0.1012342	reliant

Not-compassionate, athletic, sensitive: A mixed bag. ("Cares about self"?)

Anything left out? Uniquenesses

```
enframe(bem.15$uniquenesses, name="quality", value="uniq") %>%
  arrange(desc(uniq)) %>%
  slice(1:10)
```

quality	uniq
foullang	0.9136126
lovchil	0.8242992
analyt	0.8120934
yielding	0.7911748
masculin	0.7228739
athlet	0.7217327
shy	0.7033071
gullible	0.7000779
flatter	0.6625008
helpful	0.6516863

Uses foul language especially, also loves children and analytical. So could use even more factors.