

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(lavaan) # for confirmatory, later  
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://www.utsc.utoronto.ca/~butler/d29/rex2.txt"
kids <- read_delim(my_url, " ")
kids
```

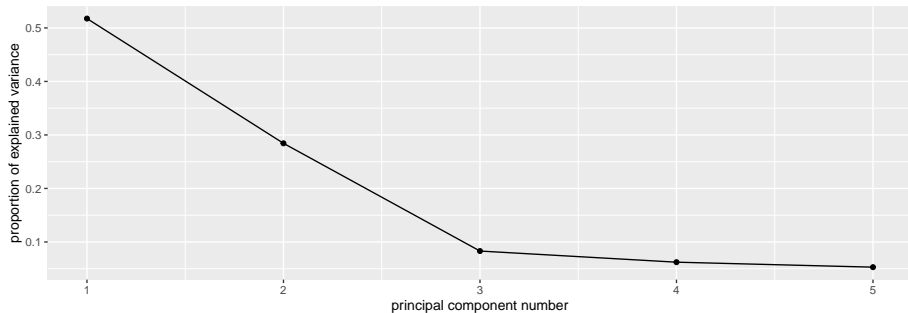
test	para	sent	word	add	dots
para	1.000	0.722	0.714	0.203	0.095
sent	0.722	1.000	0.685	0.246	0.181
word	0.714	0.685	1.000	0.170	0.113
add	0.203	0.246	0.170	1.000	0.585
dots	0.095	0.181	0.113	0.585	1.000

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.

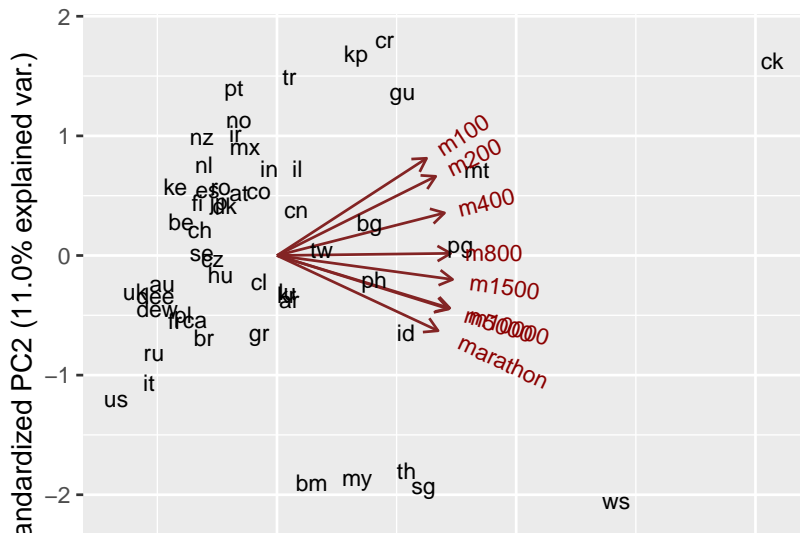
Track running records revisited

Read the data, run principal components, get biplot:

```
my_url <- "http://www.utsc.utoronto.ca/~butler/d29/men_track_f  
track <- read_table(my_url)  
track %>% select(where(is.numeric)) -> track_num  
track.pc <- princomp(track_num, cor = T)  
g2 <- ggbiplot(track.pc, labels = track$country)
```

The biplot

g2



Factor Analysis

Benefit of rotation

- 100m and marathon arrows almost perpendicular, but components don't match anything much:
- sprinting: bottom left and top right
- distance running: top left and bottom right.
- Can we arrange things so that components (factors) correspond to something meaningful?

Track records by factor analysis

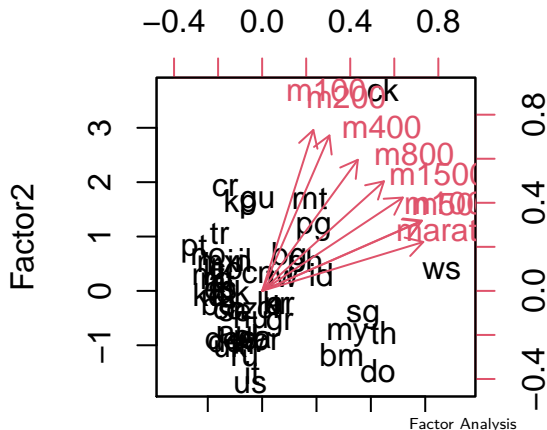
Obtain factor scores (have actual data):

```
track %>%  
  select(where(is.numeric)) %>%  
  factanal(2, scores = "r") -> track.f
```

Track data biplot

Not so nice-looking:

```
biplot(track.f$scores, track.f$loadings,
       xlab = track$country
)
```



Comments

- This time 100m “up” (factor 2), marathon “right” (factor 1).
- Countries most negative on factor 2 good at sprinting.
- Countries most negative on factor 1 good at distance running.

Rotated factor loadings

```
track.f$loadings
```

```
##  
## Loadings:  
##           Factor1 Factor2  
## m100      0.291    0.914  
## m200      0.382    0.882  
## m400      0.543    0.744  
## m800      0.691    0.622  
## m1500     0.799    0.530  
## m5000     0.901    0.394  
## m10000    0.907    0.399  
## marathon 0.915    0.278  
##  
##           Factor1 Factor2  
## SS loadings      4.112    3.225  
## Proportion Var   0.514    0.403  
## Cumulative Var   0.514    0.917
```

Which countries are good at sprinting or distance running?

Make a data frame with the countries and scores in:

```
scores <- data.frame(  
  country = track$country,  
  track.f$scores  
)  
scores %>% slice(1:6)
```

country	Factor1	Factor2
ar	0.3363378	-0.2651512
au	-0.4939579	-0.8121335
at	-0.7419991	0.1764151
be	-0.7960275	-0.2388525
bm	1.4654159	-1.1704466
br	0.0778016	-0.8871291

The best sprinting countries

Most negative on factor 2:

```
scores %>%
```

```
  arrange(Factor2) %>%
```

```
  left_join(iso, by = c("country" = "ISO2")) %>%
```

```
  select(Country, Factor1, Factor2) %>%
```

```
  slice(1:10)
```

Country	Factor1	Factor2
United States of America	-0.2194270	-1.7251036
Italy	-0.1843670	-1.4990521
Dominican Republic	2.1290655	-1.4666402
Russian Federation	-0.3247311	-1.2236590
Bermuda	1.4654159	-1.1704466
United Kingdom	-0.5896906	-1.0139983
France	-0.2530185	-0.9519162
West Germany	-0.4674888	-0.9079005
Canada	-0.1369016	-0.8920777
Brazil	0.0778016	-0.8871291

The best distance-running countries

Most negative on factor 1:

```
scores %>%  
  arrange(Factor1) %>%  
  left_join(iso, by = c("country" = "ISO2")) %>%  
  select(Country, Factor1, Factor2) %>%  
  slice(1:10)
```

Country	Factor1	Factor2
Portugal	-1.2509805	0.7836689
Norway	-0.9920727	0.6229956
New Zealand	-0.9813348	0.2660349
Kenya	-0.9749696	-0.0709948
Iran, Islamic Republic of	-0.9231505	0.5027121
Netherlands	-0.9078661	0.2394820
Romania	-0.8178386	0.1855500
Mexico	-0.8096291	0.5144676
Finland	-0.8094725	-0.0570522
Belgium	-0.7960275	-0.2388525

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

Some of the data

```
my_url <- "http://www.utsc.utoronto.ca/~butler/d29/factor.txt"
bem <- read_tsv(my_url)
bem
```

	help- subfu	pre- liant	def- bel	yield- ing	chee- ful	in- dpt	ath- let	as- shy	str- sert	force- pers	af- ful	flat- fect	ter	an- a- loyal	fem- i- nineth	sym- pa- thy	sen- si- tive	und- stand	com- passer	lead- absoot		
1	7	7	5	5	7	7	7	1	7	7	2	7	4	7	1	2	6	3	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
3	7	6	4	4	5	5	2	3	4	4	3	5	2	7	5	6	5	2	4	6	6	4
4	6	6	7	4	6	6	3	4	4	3	3	5	3	7	6	6	5	2	6	6	6	2
5	6	6	7	4	7	7	7	2	7	7	5	7	4	7	7	4	7	3	7	6	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
8	6	4	6	6	6	3	1	3	3	4	1	7	7	6	4	4	7	3	6	6	6	1
9	7	6	7	5	6	7	5	2	5	6	6	7	3	7	7	6	6	4	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
11	7	4	7	4	7	5	2	1	5	6	4	7	7	7	7	5	5	1	5	6	6	4
13	7	7	7	4	6	7	1	3	6	6	4	6	5	7	7	5	7	6	6	7	7	5
14	7	7	5	5	7	1	3	5	6	7	1	7	7	7	3	5	7	5	7	7	7	2
15	7	7	7	7	7	7	7	1	7	7	7	7	5	7	7	7	7	1	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
25	6	6	7	4	4	7	7	1	5	5	7	6	1	7	4	4	7	5	4	7	5	4
26	6	6	5	5	7	6	2	2	5	5	4	4	5	7	7	4	6	2	6	6	5	6

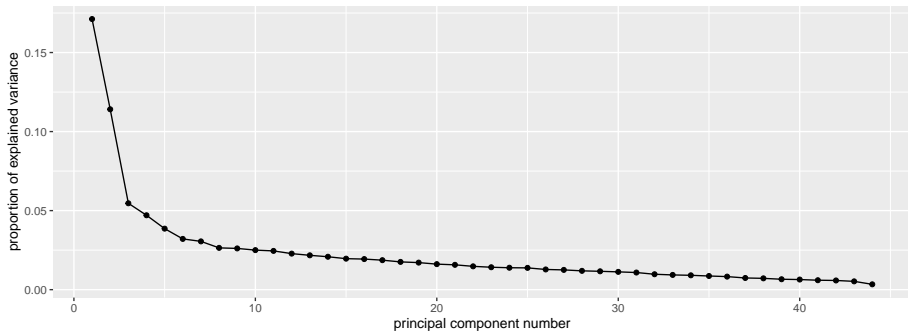
Principal components first

...to decide on number of factors:

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

The scree plot

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

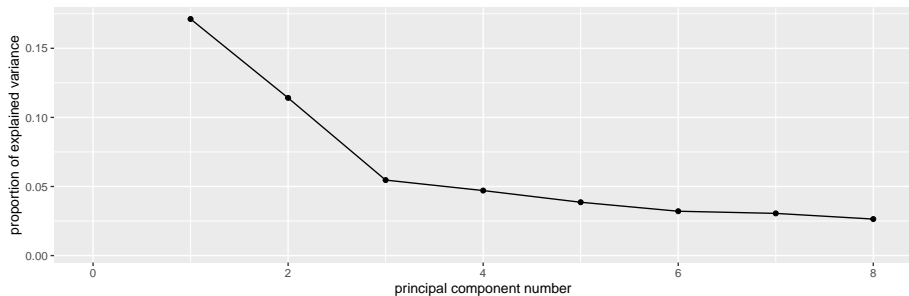


- No obvious elbow.

Zoom in to search for elbow

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

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



but is 2 really good?

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

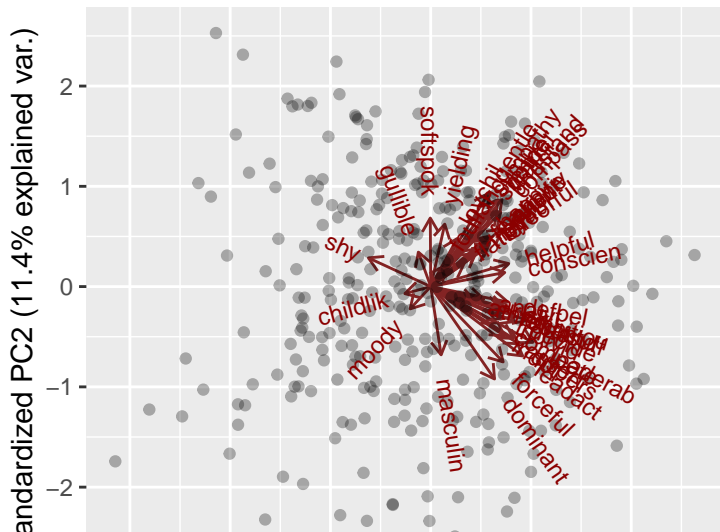
```
## Importance of components:
##               Comp.1    Comp.2
## Standard deviation    2.7444993 2.2405789
## Proportion of Variance 0.1711881 0.1140953
## Cumulative Proportion 0.1711881 0.2852834
##               Comp.3    Comp.4
## Standard deviation    1.55049106 1.43886350
## Proportion of Variance 0.05463688 0.04705291
## Cumulative Proportion 0.33992029 0.38697320
##               Comp.5    Comp.6
## Standard deviation    1.30318840 1.18837867
## Proportion of Variance 0.03859773 0.03209645
## Cumulative Proportion 0.42557093 0.45766738
##               Comp.7    Comp.8
## Standard deviation    1.15919129 1.07838912
## Proportion of Variance 0.03053919 0.02643007
## Cumulative Proportion 0.48820657 0.51463664
##               Comp.9    Comp.10
## Standard deviation    1.07120568 1.04901318
## Proportion of Variance 0.02607913 0.02500974
## Cumulative Proportion 0.54071577 0.56572551
##               Comp.11    Comp.12
```

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 10 o'clock or 7 o'clock.
- Suggests factor analysis with rotation will get interpretable factors (rotate to 6 o'clock and 9 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  
## ...
```

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

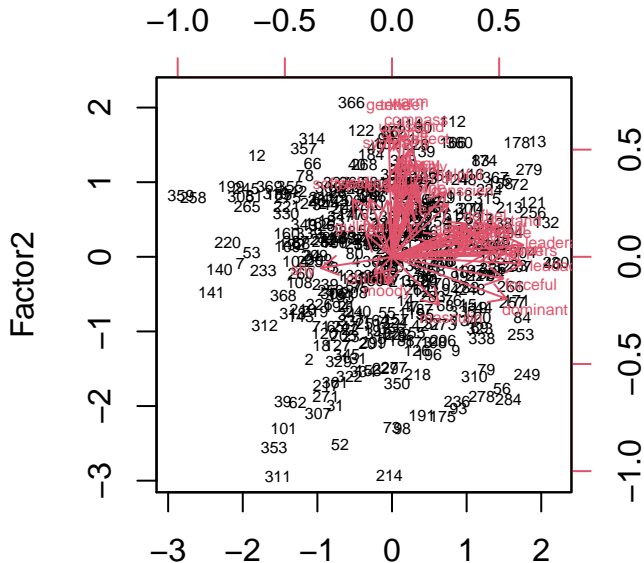
Plotting the two factors

- 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
573	335	cheerful	4	0.1523718	0.3705305
349	206	feminine	6	0.1132019	0.3227962
37	26	sympathy	3	0.0230146	0.5256654
418	237	compete	2	0.4504188	0.0532073
589	345	loyal	6	0.1512127	0.4166622
479	270	shy	3	-0.4144579	-0.0653728
553	322	athlet	3	0.2670788	0.0755429
202	114	softspok	6	-0.2303283	0.3362171
98	60	lovchil	7	-0.0270572	0.3268500
527	306	leadact	1	0.7627129	-0.0406672
311	181	analyt	4	0.2949559	0.1270016
397	223	yielding	4	-0.1309965	0.3376293

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

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

Factor 5

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

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

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.

Section 1

Confirmatory factor analysis}

Confirmatory factor analysis

- Exploratory: what do data suggest as hidden underlying factors (in terms of variables observed)?
- Confirmatory: have *theory* about how underlying factors depend on observed variables; test whether theory supported by data:
- does theory provide *some* explanation (better than nothing)
- can we do better?
- Also can compare two theories about factors: is more complicated one significantly better than simpler one?

Children and tests again

- Previously had this correlation matrix of test scores (based on 145 children):

```
km
##          para  sent  word   add  dots
## [1,]  1.000  0.722  0.714  0.203  0.095
## [2,]  0.722  1.000  0.685  0.246  0.181
## [3,]  0.714  0.685  1.000  0.170  0.113
## [4,]  0.203  0.246  0.170  1.000  0.585
## [5,]  0.095  0.181  0.113  0.585  1.000
```

- Will use package `lavaan` for confirmatory analysis.
- Can use actual data or correlation matrix.
- Latter (a bit) more work, as we see.

Two or three steps

- Make sure correlation matrix (if needed) is handy.
- Specify factor model (from theory)
- Fit factor model: does it fit acceptably?

Terminology

- Thing you cannot observe called **latent variable**.
- Thing you *can* observe called **manifest variable**.
- Model predicts latent variables from manifest variables.
 - asserts a relationship between latent and manifest.
- We need to invent names for the latent variables.

Specifying a factor model

- Model with one factor including all the tests:

```
test.model.1 <- "ability=~para+sent+word+add+dots"
```

- and a model that we really believe, that there are two factors, a verbal and a mathematical:

```
test.model.2 <- "verbal=~para+sent+word  
                math=~add+dots"
```

- Note the format: really all one line between single quotes, but putting it on several lines makes the layout clearer.
- Also note special notation =~ for “this latent variable depends on these observed variables”.

Fitting a 1-factor model

- Need to specify model, correlation matrix, n like this:

```
fit1 <- cfa(test.model.1,
  sample.cov = km,
  sample.nobs = 145
)
```

- Has summary, or briefer version like this:

```
fit1

## lavaan 0.6-8 ended normally after 16 iterations
##
##      Estimator                  ML
##      Optimization method      NLMINB
##      Number of model parameters      10
##
##      Number of observations          145
##
## Model Test User Model:
##
##      Test statistic                  59.886
```

Two-factor model

```
fit2 <- cfa(test.model.2, sample.cov = km, sample.nobs = 145)
fit2
```

```
## lavaan 0.6-8 ended normally after 25 iterations
##
##      Estimator                      ML
##      Optimization method          NLMINB
##      Number of model parameters      11
##
##      Number of observations          145
##
## Model Test User Model:
##
##      Test statistic                  2.951
##      Degrees of freedom              4
##      P-value (Chi-square)            0.566
```

- This fits OK: 2-factor model supported by the data.
- 1-factor model did not fit. We really need 2 factors.
- Same conclusion as from `factanal` earlier.

Comparing models

- Use anova as if this were a regression:

```
anova(fit1, fit2)
```

	Df	AIC	BIC	Chisq	Chisq diff	Df diff	Pr(>Chisq)
fit2	4	1776.673	1809.417	2.950949	NA	NA	NA
fit1	5	1831.608	1861.375	59.886210	56.93526	1	0

- 2-factor model fits significantly better than 1-factor.
- No surprise!

Track and field data, yet again

- cfa works easier on actual data, such as the running records:

```
track %>% print(n = 6)
```

```
## # A tibble: 55 x 9
##   m100  m200  m400  m800 m1500 m5000 m10000 marathon
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>    <dbl>
## 1  10.4  20.8  46.8  1.81  3.7   14.0   29.4    138.
## 2  10.3  20.1  44.8  1.74  3.57  13.3   27.7    128.
## 3  10.4  20.8  46.8  1.79  3.6   13.3   27.7    136.
## 4  10.3  20.7  45.0  1.73  3.6   13.2   27.4    130.
## 5  10.3  20.6  45.9  1.8   3.75  14.7   30.6    147.
## 6  10.2  20.4  45.2  1.73  3.66  13.6   28.6    133.
## # ... with 49 more rows, and 1 more variable: country <chr>
```

- Specify factor model. Factors seemed to be “sprinting” (up to 800m) and “distance running” (beyond):

```
track.model <- "sprint=~m100+m200+m400+m800
               distance=~m1500+m5000+m10000+marathon"
```

Fit and examine the model

- Fit the model. The observed variables are on different scales, so we should standardize them first via `std.ov`:

```
track.1 <- track %>%
  select(-country) %>%
  cfa(track.model, data = ., std.ov = T)
track.1
```

```
## lavaan 0.6-8 ended normally after 59 iterations
```

```
##
```

##	Estimator	ML
##	Optimization method	NLMINB
##	Number of model parameters	17

```
##
```

##	Number of observations	55
----	------------------------	----

```
##
```

```
## Model Test User Model:
```

```
##
```

##	Test statistic	87.608
##	Degrees of freedom	19
##	P-value (Chi-square)	0.000

- This fits badly. Can we do better?

Factor model 2

- Define factor model:

```
track.model.2 <- "sprint=~m100+m200+m400  
                 middle=~m800+m1500  
                 distance=~m5000+m10000+marathon"
```

- Fit:

```
track %>%  
  select(-country) %>%  
  cfa(track.model.2, data = ., std.ov = T) -> track.2
```

Examine

```
track.2
```

```
## lavaan 0.6-8 ended normally after 72 iterations
##
##      Estimator                      ML
##      Optimization method          NLMINB
##      Number of model parameters    19
##
##      Number of observations        55
##
## Model Test User Model:
##
##      Test statistic                40.089
##      Degrees of freedom            17
##      P-value (Chi-square)          0.001
```

- Fits marginally better, though still badly.

Comparing the two models

- Second model doesn't fit well, but is it better than first?

```
anova(track.1, track.2)
```

	Df	AIC	BIC	Chisq	Chisq diff	Df diff	Pr(>Chisq)
track.2	17	535.4894	573.6288	40.08919	NA	NA	NA
track.1	19	579.0083	613.1329	87.60804	47.51885	2	0

- Oh yes, a lot better.