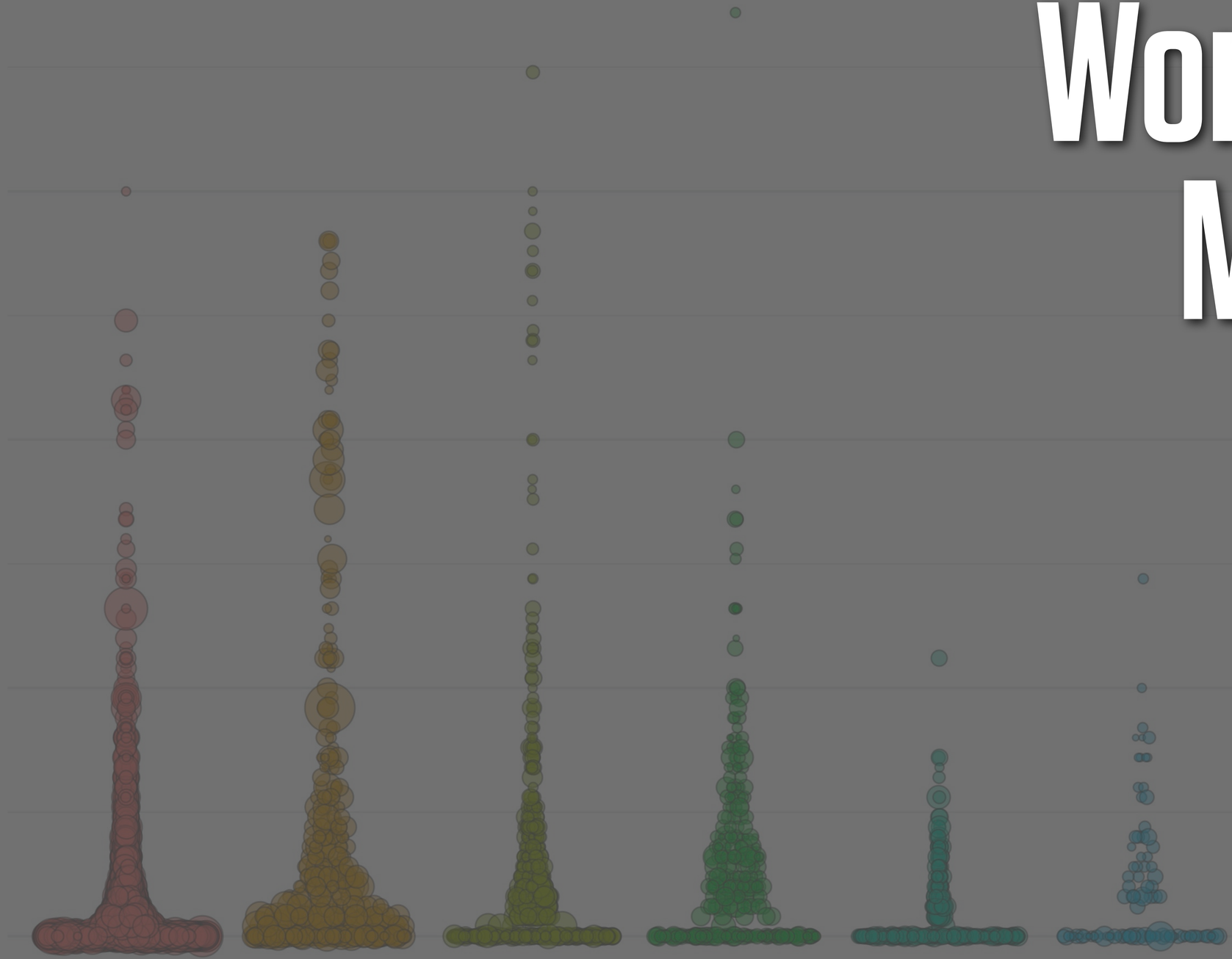
Data Visualization

rstudio::conf

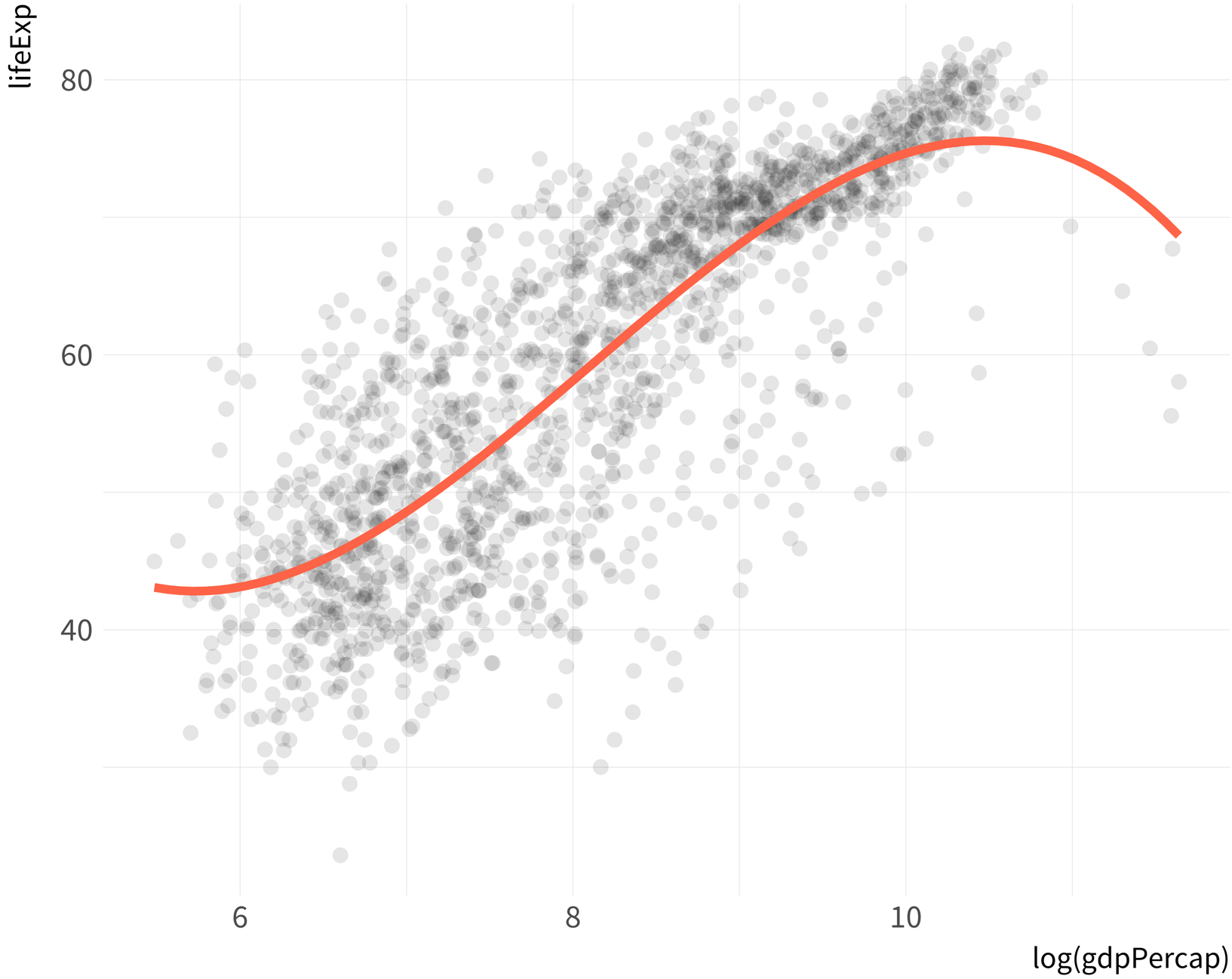
Work with Models



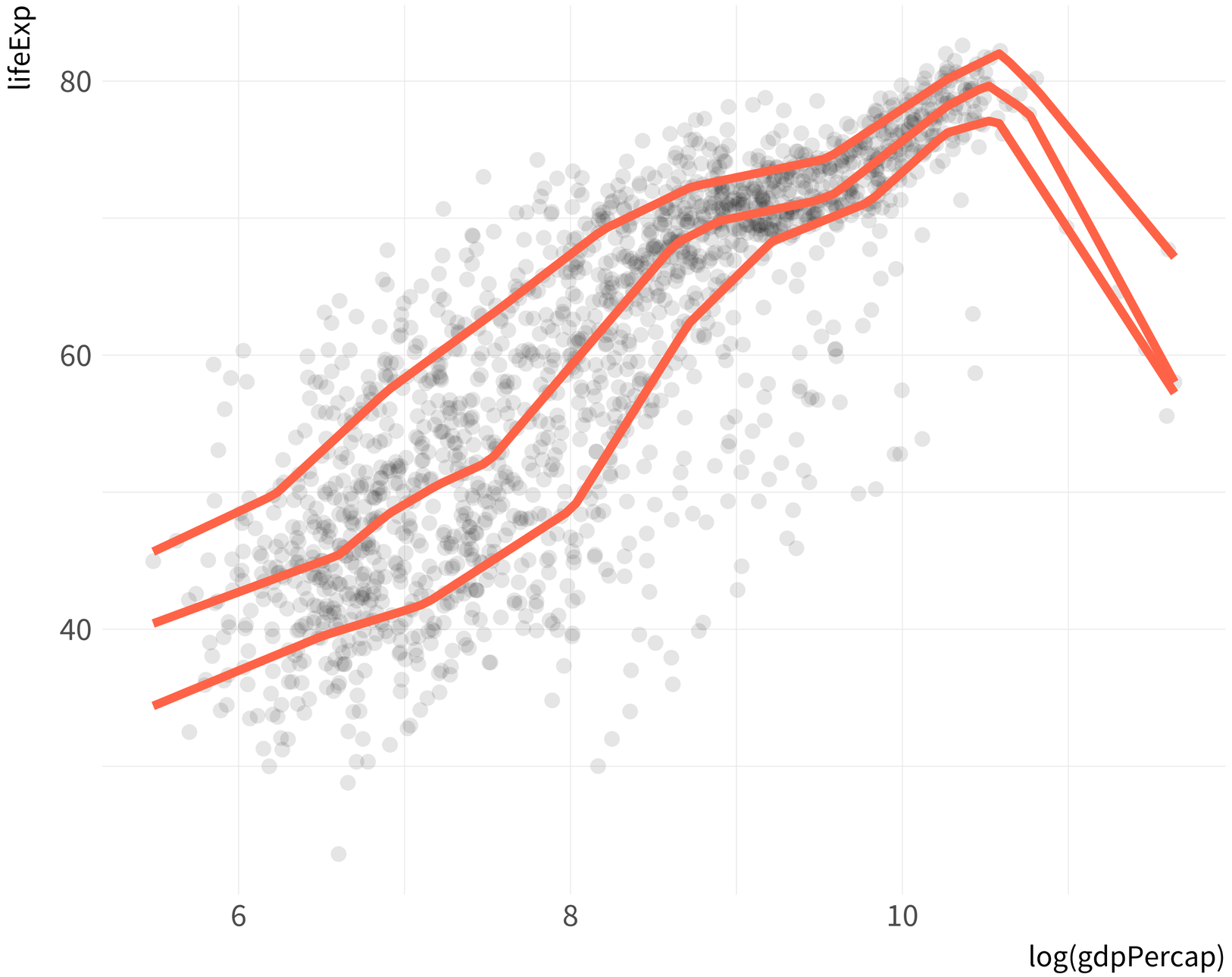
```r
p <- ggplot(data = gapminder,
            mapping = aes(x = log(gdpPercap), y = lifeExp))

p + geom_point(alpha=0.1) +
    geom_smooth(color = "tomato",
                fill="tomato",
                method = MASS::rlm) +
    geom_smooth(color = "steelblue",
                fill="steelblue",
                method = "lm")
```


lifeExp
80
60
40
6
8
10
log(gdpPercap)

```
p + geom_point(alpha=0.1) +
    geom_smooth(color = "tomato",
                method = "lm", size = 1.2,
                formula = y ~ splines::bs(x, 3),
                se = FALSE)
```


lifeExp
80
60
40
6
8
10
log(gdpPercap)

```
p + geom_point(alpha=0.15) +
    geom_smooth(color = "tomato",
                method = "lm",
                size = 1.2,
                formula = y ~ splines::bs(x, 3),
                se = FALSE)
```


lifeExp
80
60
40
6
8
10
log(gdpPercap)

LOOK INSIDE MODEL OBJECTS

```
gapminder
```

```
## # A tibble: 1,704 x 6
##        country continent  year lifeExp      pop gdpPercap
##         <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan      Asia  1952    28.8  8425333       779
##  2 Afghanistan      Asia  1957    30.3  9240934       821
##  3 Afghanistan      Asia  1962    32.0 10267083       853
##  4 Afghanistan      Asia  1967    34.0 11537966       836
##  5 Afghanistan      Asia  1972    36.1 13079460       740
##  6 Afghanistan      Asia  1977    38.4 14880372       786
##  7 Afghanistan      Asia  1982    39.9 12881816       978
##  8 Afghanistan      Asia  1987    40.8 13867957       852
##  9 Afghanistan      Asia  1992    41.7 16317921       649
## 10 Afghanistan      Asia  1997    41.8 22227415       635
## # ... with 1,694 more rows
```

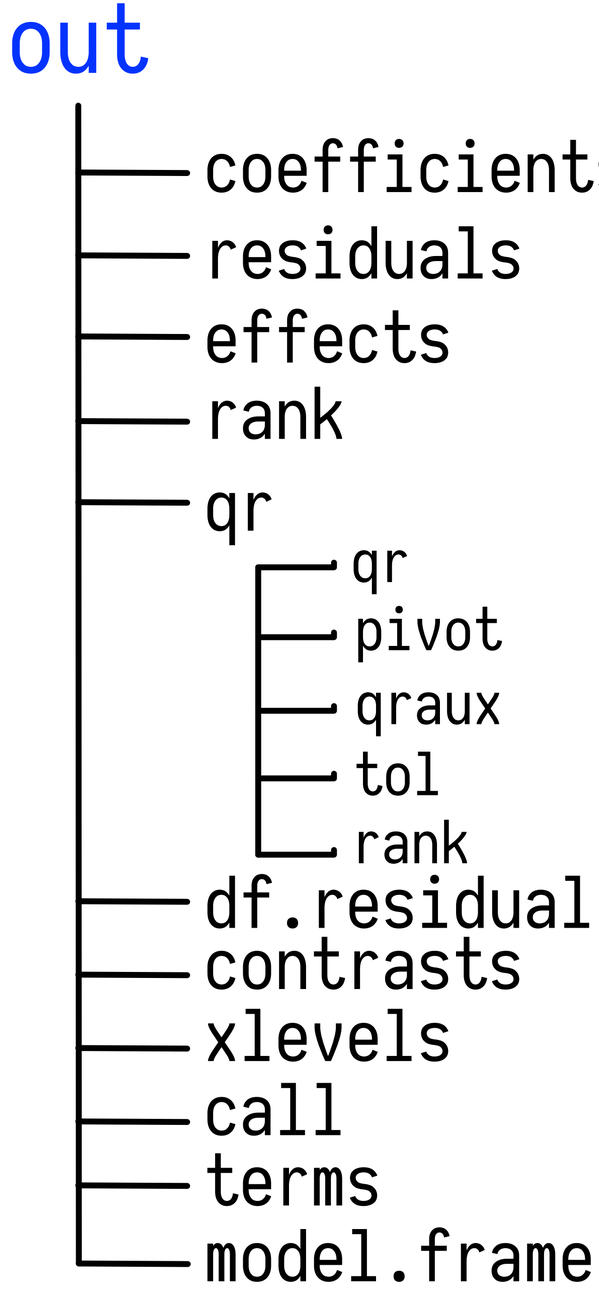
```
str(gapminder)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    1704 obs. of  6 variables:
## $ country : Factor w/ 142 levels "Afghanistan",..: 1 1 ...
## $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3
##    ...
## $ year : int 1952 1957 ...
## $ lifeExp : num 28.8 ...
## $ pop : int 8425333 9240934 ...
## $ gdpPercap: num 779 ...
```

```r
out <- lm(formula = lifeExp ~ gdpPercap + pop + continent,
          data = gapminder)
```

```r
summary(out)
```

```
##
## Call:
## lm(formula = lifeExp ~ gdpPercap + pop + continent, data = gapminder)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -49.16  -4.49   0.30   5.11  25.17
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.78e+01   3.40e-01  140.82   <2e-16 ***
## gdpPercap         4.50e-04   2.35e-05   19.16   <2e-16 ***
## pop               6.57e-09   1.98e-09    3.33    9e-04 ***
## continentAmericas 1.35e+01   6.00e-01   22.46   <2e-16 ***
## continentAsia     8.19e+00   5.71e-01   14.34   <2e-16 ***
## continentEurope   1.75e+01   6.25e-01   27.97   <2e-16 ***
## continentOceania  1.81e+01   1.78e+00   10.15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.37 on 1697 degrees of freedom
## Multiple R-squared:  0.582,  Adjusted R-squared:  0.581
## F-statistic:  394 on 6 and 1697 DF,  p-value: <2e-16
```




```
out$coefficients
out$residuals
out$df.residual
```

USE RSTUDIO'S OBJECT INSPECTOR TO POKE AROUND

GET MODEL-BASED GRAPHICS RIGHT

Present Findings in Substantive Terms

Present Findings in Substantive Terms

Show Your Degree of Confidence or Uncertainty

Present Findings in Substantive Terms

Show Your Degree of Confidence or Uncertainty

Show the Data When you Can

Notice that each of these points applies just as well to the presentation of data in any format: tables, models, text, whatever. Graphics are not special in this respect.

Present Findings in Substantive Terms

Show Your Degree of Confidence or Uncertainty

Show the Data When you Can

PREDICTIONS FROM MODELS: DIY Method

Behind the scenes, predict() and its methods do the work

Give predict() a model and new data, and if it has a **method** for that model **class**, it will produce y-hat estimates for the new data.

```r
min_gdp <- min(gapminder$gdpPercap)
max_gdp <- max(gapminder$gdpPercap)

med_pop <- median(gapminder$pop)


pred_df <- expand.grid(gdpPercap = (seq(from = min_gdp,
                                        to = max_gdp,
                                        length.out = 100)),
                       pop = med_pop,
                       continent = c("Africa", "Americas",
                                     "Asia", "Europe", "Oceania"))
```

Generate New Data with expand.grid()

```r
dim(pred_df)
## [1] 500   3
```

```r
head(pred_df)
##   gdpPercap     pop continent
## 1   241.166 7023596    Africa
## 2  1385.428 7023596    Africa
## 3  2529.690 7023596    Africa
## 4  3673.953 7023596    Africa
## 5  4818.215 7023596    Africa
## 6  5962.477 7023596    Africa
```

Notice that you are getting a table of data back —a data frame. Underneath we are still creating and operating on tables of tidy data.

```r
pred_out <- predict(object = out,
                    newdata = pred_df,
                    interval = "predict")

head(pred_out)
```

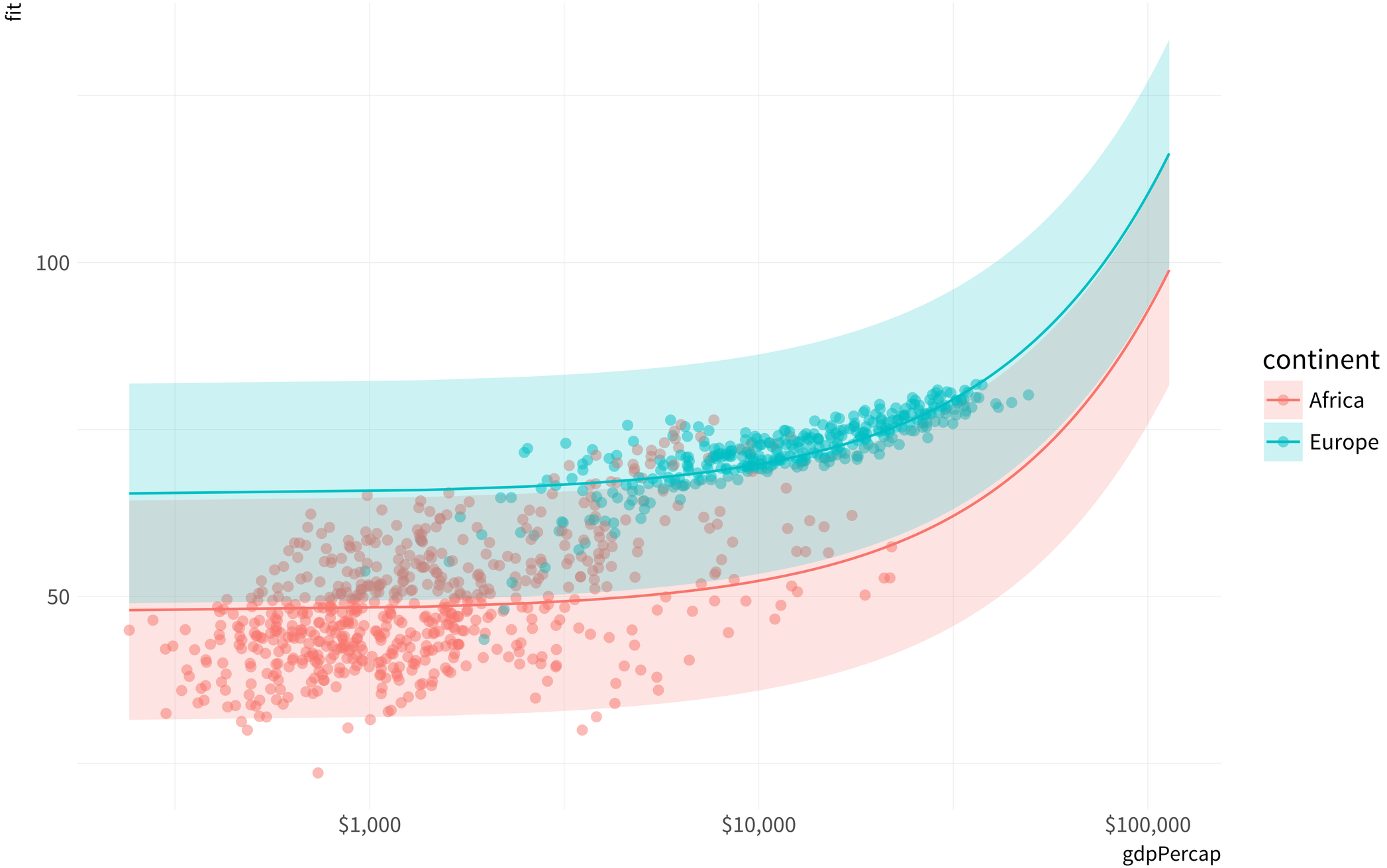
```
##        fit     lwr     upr
## 1 47.9686 31.5477 64.3895
## 2 48.4830 32.0623 64.9037
## 3 48.9973 32.5767 65.4180
## 4 49.5117 33.0909 65.9325
## 5 50.0260 33.6050 66.4471
## 6 50.5404 34.1189 66.9619
```

```
pred_df <- cbind(pred_df, pred_out)

head(pred_df)
```

```
##   gdpPercap      pop continent  fit  lwr  upr
## 1       241 7023596    Africa 48.0 31.5 64.4
## 2      1385 7023596    Africa 48.5 32.1 64.9
## 3      2530 7023596    Africa 49.0 32.6 65.4
## 4      3674 7023596    Africa 49.5 33.1 65.9
## 5      4818 7023596    Africa 50.0 33.6 66.4
## 6      5962 7023596    Africa 50.5 34.1 67.0
```

For "natural" data frames you should merge() or _join() by a common key, never cbind()

```r
p <- ggplot(data = subset(pred_df, continent %in% c("Europe",
                                "Africa")),
                  mapping = aes(x = gdpPercap,
                                y = fit,
                                ymin = lwr,
                                ymax = upr,
                                color = continent,
                                fill = continent,
                                group = continent))

p + geom_point(data = subset(gapminder,
                                continent %in% c("Europe", "Africa")),
                    mapping = aes(x = gdpPercap,
                                y = lifeExp,
                                color = continent),
                                alpha = 0.5,
                                inherit.aes = FALSE) +
    geom_line() +
    geom_ribbon(alpha = 0.2, color = FALSE) +
    scale_x_log10(labels = scales::dollar)
```


fit
100
50
$1,000
$10,000
$100,000
gdpPercap
continent
Africa
Europe

Model-specific methods do most of that work automatically

In practice, we will use predict() methods behind the scenes when working with models, rather than going through these steps manually.

USING broom TO TIDY UP MODELS

```
library(broom)
```

component level `tidy()`

observation level `augment()`

model level `glance()`

broom's functions turn information contained in model obects into tidy data that you can feed to your plots

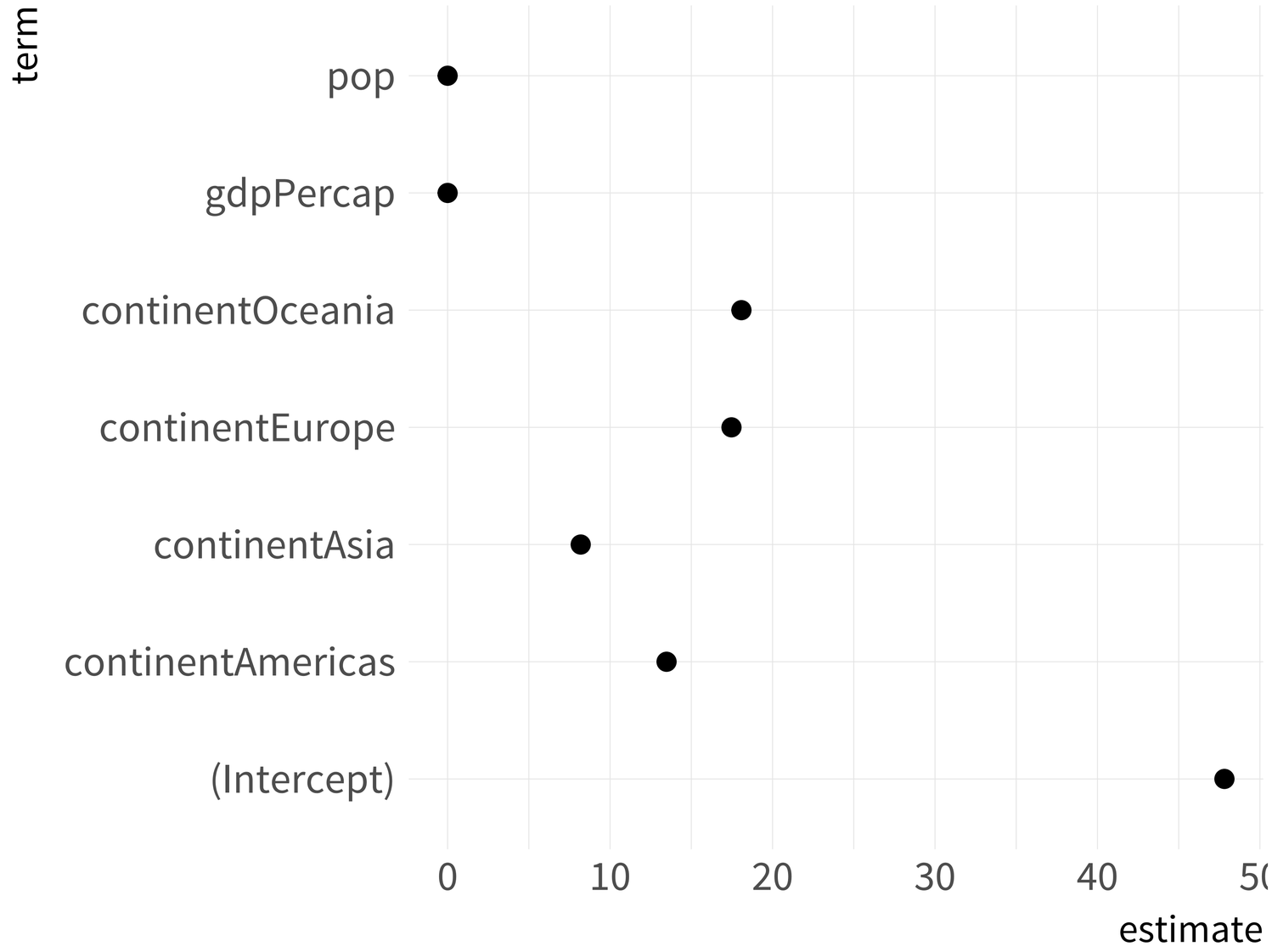
```
summary(out)
##
## Call:
## lm(formula = lifeExp ~ gdpPercap + pop + continent, data = gapminder)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.161  -4.486   0.297   5.110  25.175
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.781e+01  3.395e-01 140.819  < 2e-16 ***
## gdpPercap         4.495e-04  2.346e-05  19.158  < 2e-16 ***
## pop               6.570e-09  1.975e-09   3.326 0.000901 ***
## continentAmericas 1.348e+01  6.000e-01  22.458  < 2e-16 ***
## continentAsia     8.193e+00  5.712e-01  14.342  < 2e-16 ***
## continentEurope   1.747e+01  6.246e-01  27.973  < 2e-16 ***
## continentOceania  1.808e+01  1.782e+00  10.146  < 2e-16 ***
##
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.365 on 1697 degrees of freedom
## Multiple R-squared:  0.5821, Adjusted R-squared:  0.5806
## F-statistic: 393.9 on 6 and 1697 DF,  p-value: < 2.2e-16
```

component level

```
out_comp <- tidy(out)

out_comp %>% round_df()
```

```
##                 term estimate std.error statistic p.value
## 1        (Intercept)    47.81      0.34    140.82       0
## 2          gdpPercap     0.00      0.00     19.16       0
## 3                pop     0.00      0.00      3.33       0
## 4 continentAmericas    13.48      0.60     22.46       0
## 5     continentAsia     8.19      0.57     14.34       0
## 6   continentEurope    17.47      0.62     27.97       0
## 7  continentOceania    18.08      1.78     10.15       0
```

```
p <- ggplot(out_comp, mapping = aes(x = term,
                                    y = estimate))

p + geom_point() + coord_flip()
```

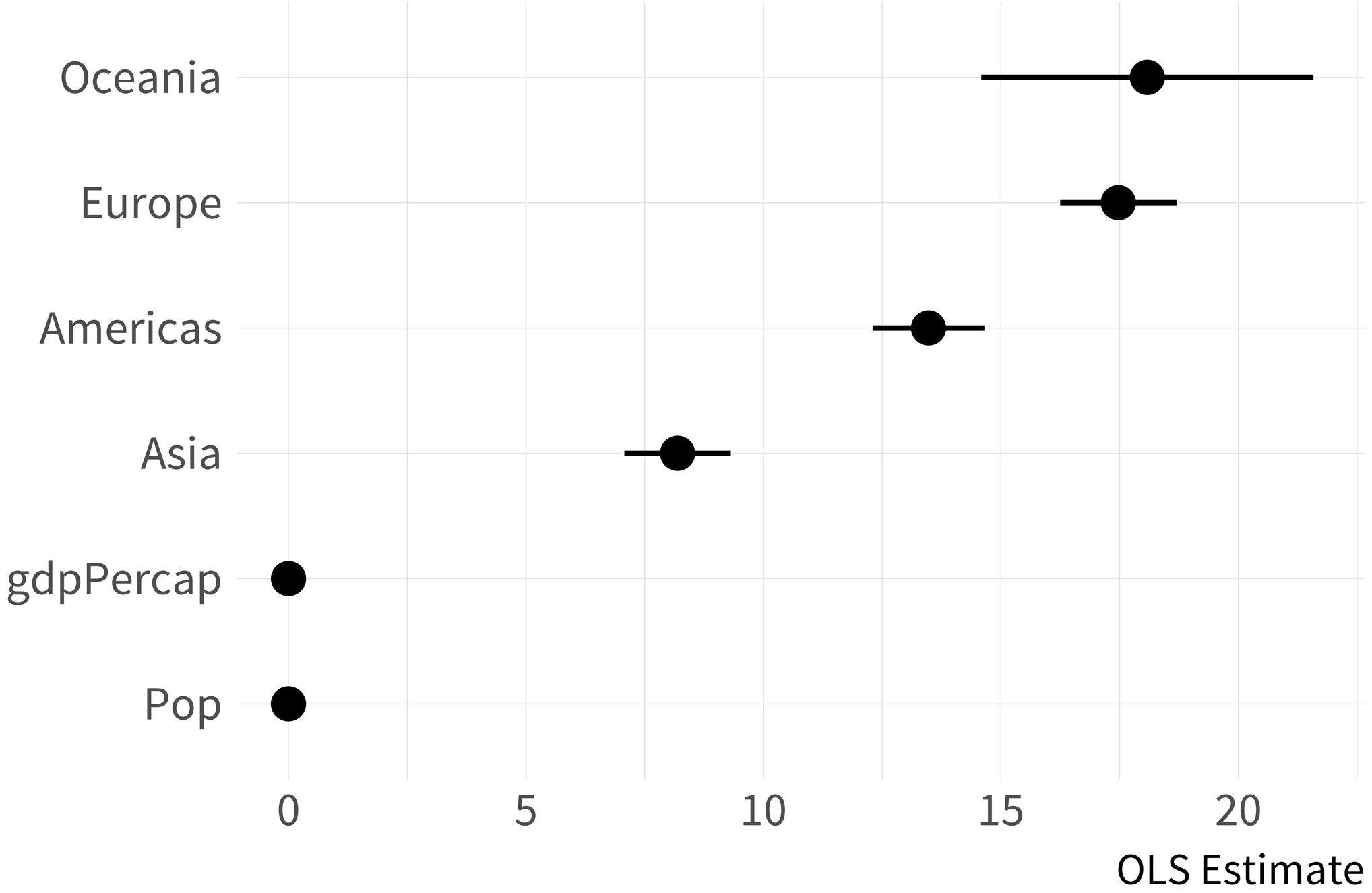

term
pop
gdpPercap
continentOceania
continentEurope
continentAsia
continentAmericas
(Intercept)
0
10
20
30
40
50
estimate

```
out_conf <- tidy(out, conf.int = TRUE)
out_conf %>% round_df()
```

```
##                  term estimate std.error statistic p.value conf.low conf.high
## 1         (Intercept)    47.81      0.34    140.82       0    47.15     48.48
## 2           gdpPercap     0.00      0.00     19.16       0     0.00      0.00
## 3                 pop     0.00      0.00      3.33       0     0.00      0.00
## 4 continentAmericas    13.48      0.60     22.46       0    12.30     14.65
## 5     continentAsia     8.19      0.57     14.34       0     7.07      9.31
## 6   continentEurope    17.47      0.62     27.97       0    16.25     18.70
## 7  continentOceania    18.08      1.78     10.15       0    14.59     21.58
```

```r
out_conf <- subset(out_conf, term %nin% "(Intercept)")

out_conf <- out_conf %>%
     mutate(nicelabs = prefix_strip(term, "continent"))
```

```
p <- ggplot(out_conf, mapping = aes(x = reorder(nicelabs, estimate),
                                    y = estimate,
                                    ymin = conf.low,
                                    ymax = conf.high))

p + geom_pointrange() + coord_flip() +
    labs(x="", y="OLS Estimate")
```


Oceania
Europe
Americas
Asia
gdpPercap
Pop
0
5
10
15
20
OLS Estimate

```
summary(out)
##
## Call:
## lm(formula = lifeExp ~ gdpPercap + pop + continent, data = gapminder)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.161  -4.486   0.297   5.110  25.175
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.781e+01  3.395e-01 140.819  < 2e-16 ***
## gdpPercap         4.495e-04  2.346e-05  19.158  < 2e-16 ***
## pop               6.570e-09  1.975e-09   3.326 0.000901 ***
## continentAmericas 1.348e+01  6.000e-01  22.458  < 2e-16 ***
## continentAsia     8.193e+00  5.712e-01  14.342  < 2e-16 ***
## continentEurope   1.747e+01  6.246e-01  27.973  < 2e-16 ***
## continentOceania  1.808e+01  1.782e+00  10.146  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.365 on 1697 degrees of freedom
## Multiple R-squared:  0.5821, Adjusted R-squared:  0.5806
## F-statistic: 393.9 on 6 and 1697 DF,  p-value: < 2.2e-16
```

observation level

- `.fitted` — The fitted values of the model.
- `.se.fit` — The standard errors of the fitted values.
- `.resid` — The residuals.
- `.hat` — The diagonal of the hat matrix.
- `.sigma` — An estimate of residual standard deviation when the corresponding observation is dropped from the model.
- `.cooksd` — Cook’s distance, a common regression diagnostic.
- `.std.resid` — The standardized residuals.

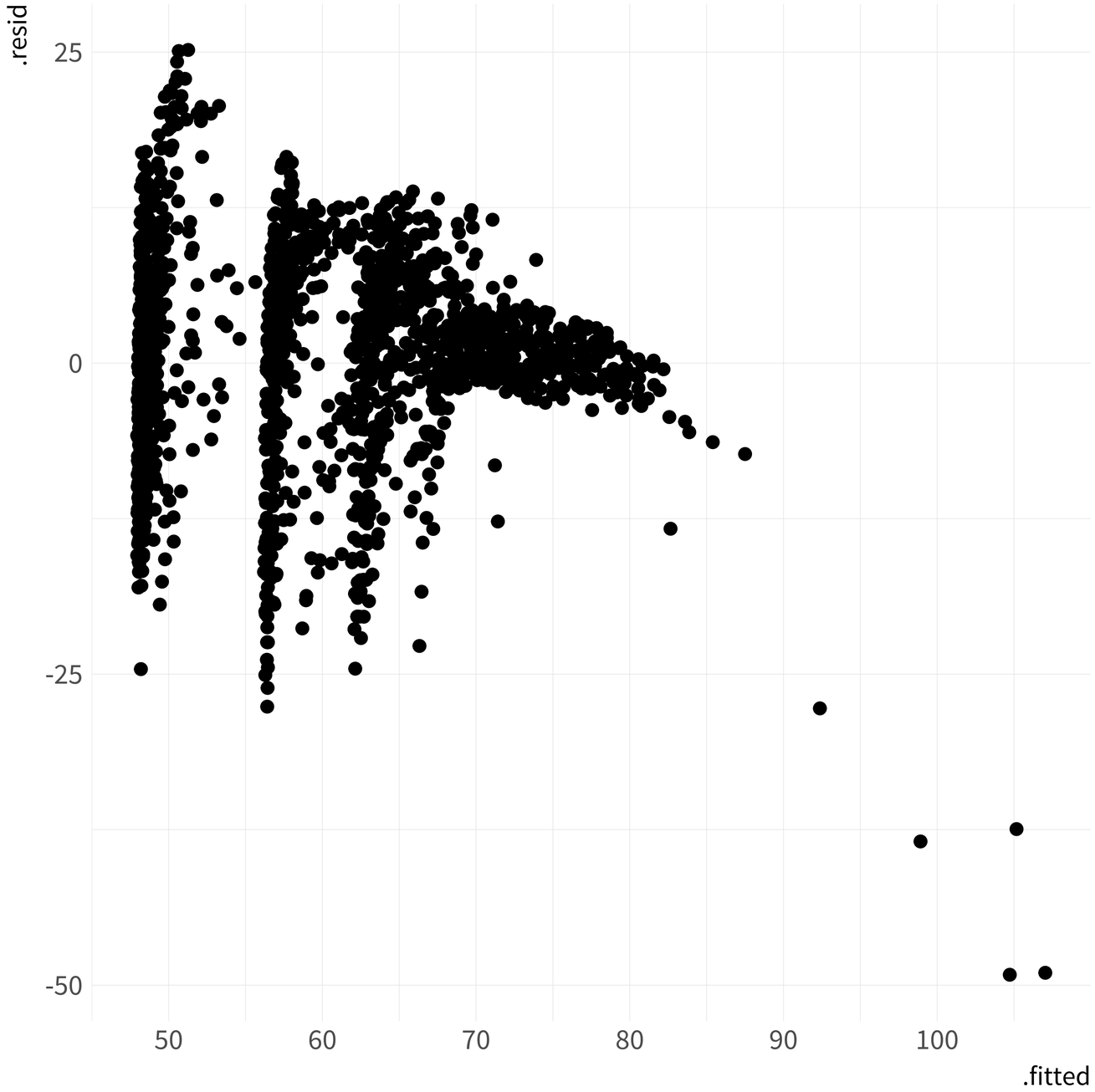
```
out_aug <- augment(out)
head(out_aug) %>% round_df()
```

```
##   lifeExp gdpPercap      pop continent .fitted .se.fit
## 1   28.80    779.45  8425333      Asia   56.41    0.47
## 2   30.33    820.85  9240934      Asia   56.44    0.47
## 3   32.00    853.10 10267083      Asia   56.46    0.47
## 4   34.02    836.20 11537966      Asia   56.46    0.47
## 5   36.09    739.98 13079460      Asia   56.43    0.47
## 6   38.44    786.11 14880372      Asia   56.46    0.47
##   .resid .hat .sigma .cooksd .std.resid
## 1 -27.61    0   8.34    0.01      -3.31
## 2 -26.10    0   8.34    0.00      -3.13
## 3 -24.46    0   8.35    0.00      -2.93
## 4 -22.44    0   8.35    0.00      -2.69
## 5 -20.34    0   8.35    0.00      -2.44
## 6 -18.02    0   8.36    0.00      -2.16
```

```
out_aug <- augment(out, data = gapminder)
head(out_aug) %>% round_df()
```

```
##       country continent year lifeExp      pop gdpPercap
## 1 Afghanistan      Asia 1952   28.80  8425333    779.45
## 2 Afghanistan      Asia 1957   30.33  9240934    820.85
## 3 Afghanistan      Asia 1962   32.00 10267083    853.10
## 4 Afghanistan      Asia 1967   34.02 11537966    836.20
## 5 Afghanistan      Asia 1972   36.09 13079460    739.98
## 6 Afghanistan      Asia 1977   38.44 14880372    786.11
##   .fitted .se.fit .resid .hat .sigma .cooksd .std.resid
## 1   56.41    0.47 -27.61    0   8.34    0.01       -3.31
## 2   56.44    0.47 -26.10    0   8.34    0.00       -3.13
## 3   56.46    0.47 -24.46    0   8.35    0.00       -2.93
## 4   56.46    0.47 -22.44    0   8.35    0.00       -2.69
## 5   56.43    0.47 -20.34    0   8.35    0.00       -2.44
## 6   56.46    0.47 -18.02    0   8.36    0.00       -2.16
```

```
p <- ggplot(data = out_aug,
            mapping = aes(x = .fitted, y = .resid))
p + geom_point()
```


.resid
25
0
-25
-50
50
60
70
80
90
100
.fitted

```
summary(out)
##
## Call:
## lm(formula = lifeExp ~ gdpPercap + pop + continent, data = gapminder)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.161  -4.486   0.297   5.110  25.175
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.781e+01  3.395e-01 140.819  < 2e-16 ***
## gdpPercap         4.495e-04  2.346e-05  19.158  < 2e-16 ***
## pop               6.570e-09  1.975e-09   3.326 0.000901 ***
## continentAmericas 1.348e+01  6.000e-01  22.458  < 2e-16 ***
## continentAsia     8.193e+00  5.712e-01  14.342  < 2e-16 ***
## continentEurope   1.747e+01  6.246e-01  27.973  < 2e-16 ***
## continentOceania  1.808e+01  1.782e+00  10.146  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.365 on 1697 degrees of freedom
## Multiple R-squared:  0.5821, Adjusted R-squared:  0.5806
## F-statistic: 393.9 on 6 and 1697 DF,  p-value: < 2.2e-16
```

model level

```
glance(out) %>% round_df()
```

```
##   r.squared adj.r.squared sigma statistic p.value df
## 1      0.58          0.58  8.37    393.91       0  7
##     logLik     AIC     BIC deviance df.residual
## 1 -6033.83 12083.6 12127.2   118754        1697
```

```
library(survival)
```

```
> head(lung)
```

	inst	time	status	age	sex	ph.ecog	ph.karno	pat.karno	meal.cal	wt.loss
1	3	306	2	74	1	1	90	100	1175	NA
2	3	455	2	68	1	0	90	90	1225	15
3	3	1010	1	56	1	0	90	90	NA	15
4	5	210	2	57	1	1	90	60	1150	11
5	1	883	2	60	1	0	100	90	NA	0
6	12	1022	1	74	1	1	50	80	513	0

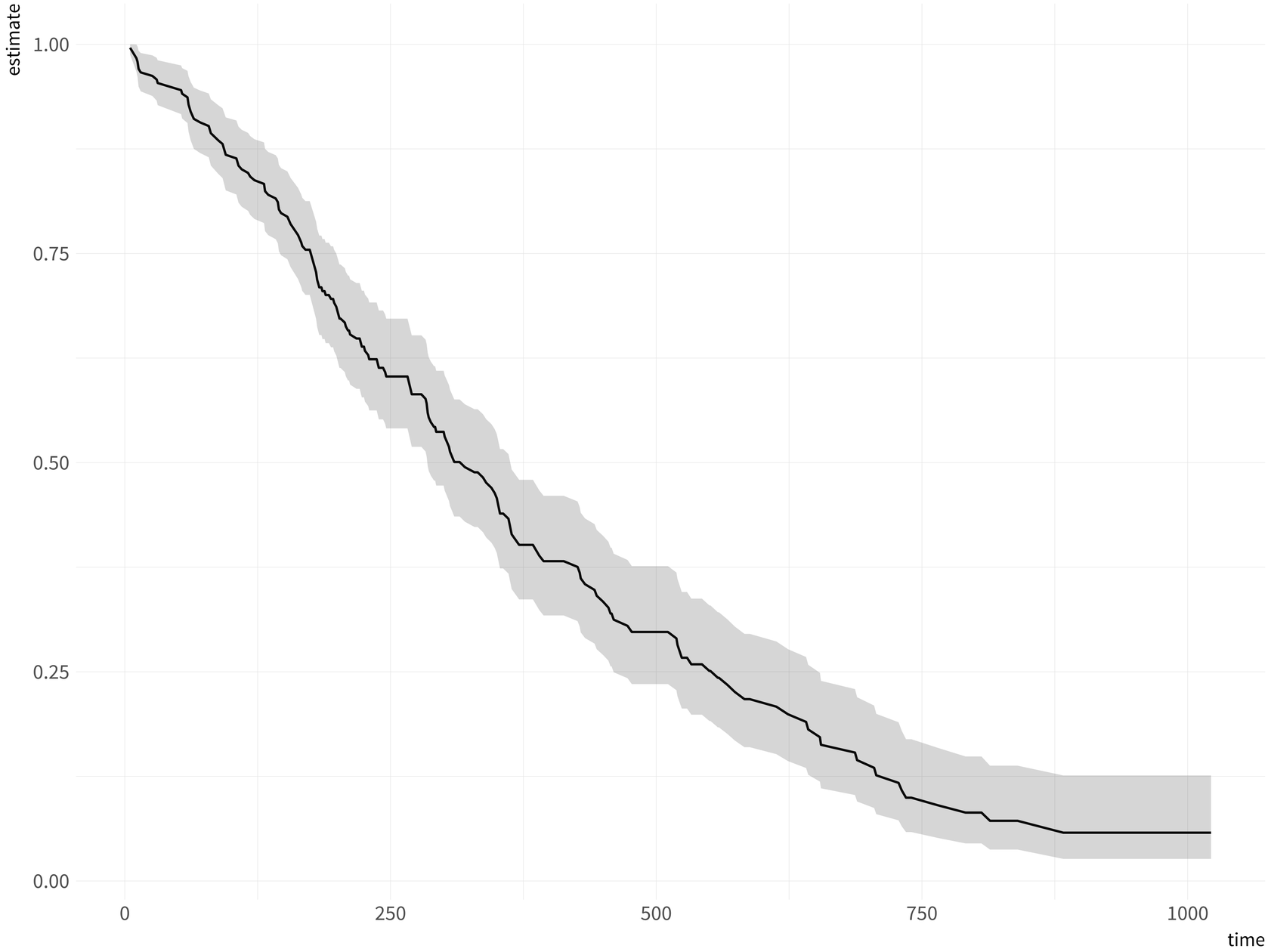
```
> tail(lung)
```

	inst	time	status	age	sex	ph.ecog	ph.karno	pat.karno	meal.cal	wt.loss
223	1	116	2	76	1	1	80	80	NA	0
224	1	188	1	77	1	1	80	60	NA	3
225	13	191	1	39	1	0	90	90	2350	-5
226	32	105	1	75	2	2	60	70	1025	5
227	6	174	1	66	1	1	90	100	1075	1
228	22	177	1	58	2	1	80	90	1060	0

```
out_cph <- coxph(Surv(time, status) ~ age + sex, data = lung)

> summary(out_cph)
Call:
coxph(formula = Surv(time, status) ~ age + sex, data = lung)

  n= 228, number of events= 165

        coef exp(coef)  se(coef)      z Pr(>|z|)
age  0.017045  1.017191  0.009223  1.848  0.06459 .
sex -0.513219  0.598566  0.167458 -3.065  0.00218 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

    exp(coef) exp(-coef) lower .95 upper .95
age    1.0172     0.9831    0.9990    1.0357
sex    0.5986     1.6707    0.4311    0.8311

Concordance= 0.603  (se = 0.025 )
Likelihood ratio test= 14.12  on 2 df,   p=9e-04
Wald test            = 13.47  on 2 df,   p=0.001
Score (logrank) test = 13.72  on 2 df,   p=0.001
```

```
out_surv <- survfit(out_cph)

> out_surv

Call: survfit(formula = out_cph)

      n  events  median 0.95LCL 0.95UCL
    228     165     320     285     363

> summary(out_surv)
Call: survfit(formula = out_cph)

 time n.risk n.event survival std.err lower 95% CI upper 95% CI
    5    228       1   0.9958 0.00417        0.9877        1.000
   11    227       3   0.9833 0.00831        0.9671        1.000
   12    224       1   0.9791 0.00928        0.9611        0.997
   13    223       2   0.9706 0.01096        0.9494        0.992
   15    221       1   0.9664 0.01170        0.9438        0.990
   26    220       1   0.9622 0.01240        0.9382        0.987
```

```r
out_tidy <- tidy(out_surv)

p <- ggplot(data = out_tidy, mapping = aes(time, estimate))

p + geom_line() +
    geom_ribbon(mapping = aes(ymin = conf.low,
                              ymax = conf.high),
                              alpha = .2)
```


estimate
1.00
0.75
0.50
0.25
0.00
0
250
500
750
1000
time

Grouped Analysis with broom

```r
eu77 <- gapminder %>% filter(continent == "Europe", year == 1977)

eu77
```

```
## # A tibble: 30 x 6
##                  country continent  year lifeExp      pop
##                   <fctr>    <fctr> <int>   <dbl>    <int>
##  1                Albania    Europe  1977   68.93  2509048
##  2                Austria    Europe  1977   72.17  7568430
##  3                Belgium    Europe  1977   72.80  9821800
##  4 Bosnia and Herzegovina    Europe  1977   69.86  4086000
##  5               Bulgaria    Europe  1977   70.81  8797022
##  6                Croatia    Europe  1977   70.64  4318673
##  7         Czech Republic    Europe  1977   70.71 10161915
##  8                Denmark    Europe  1977   74.69  5088419
##  9                Finland    Europe  1977   72.52  4738902
## 10                 France    Europe  1977   73.83 53165019
## # ... with 20 more rows, and 1 more variables:
## #   gdpPercap <dbl>
```

```r
fit <- lm(lifeExp ~ log(gdpPercap), data = eu77)

summary(fit)
```

```
##
## Call:
## lm(formula = lifeExp ~ log(gdpPercap), data = eu77)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.496 -1.031  0.093  1.176  3.712
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      29.489      7.161    4.12  0.00031 ***
## log(gdpPercap)    4.488      0.756    5.94  2.2e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.11 on 28 degrees of freedom
## Multiple R-squared:  0.557,  Adjusted R-squared:  0.541
## F-statistic: 35.2 on 1 and 28 DF,  p-value: 2.17e-06
```

```
out_le <- gapminder %>%
    group_by(continent, year) %>%
    nest()
```

```
out_le
```

```
## # A tibble: 60 x 3
##    continent  year                data
##       <fctr> <int>              <list>
##  1      Asia  1952 <tibble [33 x 4]>
##  2      Asia  1957 <tibble [33 x 4]>
##  3      Asia  1962 <tibble [33 x 4]>
##  4      Asia  1967 <tibble [33 x 4]>
##  5      Asia  1972 <tibble [33 x 4]>
##  6      Asia  1977 <tibble [33 x 4]>
##  7      Asia  1982 <tibble [33 x 4]>
##  8      Asia  1987 <tibble [33 x 4]>
##  9      Asia  1992 <tibble [33 x 4]>
## 10      Asia  1997 <tibble [33 x 4]>
## # ... with 50 more rows
```

```
out_le %>% filter(continent == "Europe" & year == 1977) %>%
       unnest(cols = c(data))
```

```
## # A tibble: 30 x 6
##    continent  year                  country lifeExp      pop
##       <fctr> <int>                   <fctr>   <dbl>    <int>
##  1    Europe  1977                  Albania   68.93  2509048
##  2    Europe  1977                  Austria   72.17  7568430
##  3    Europe  1977                  Belgium   72.80  9821800
##  4    Europe  1977 Bosnia and Herzegovina   69.86  4086000
##  5    Europe  1977                 Bulgaria   70.81  8797022
##  6    Europe  1977                  Croatia   70.64  4318673
##  7    Europe  1977           Czech Republic   70.71 10161915
##  8    Europe  1977                  Denmark   74.69  5088419
##  9    Europe  1977                  Finland   72.52  4738902
## 10    Europe  1977                   France   73.83 53165019
## # ... with 20 more rows, and 1 more variables:
## #   gdpPercap <dbl>
```

```r
fit_ols <- function(df) {
    lm(lifeExp ~ log(gdpPercap), data = df)
}

out_le <- gapminder %>%
    group_by(continent, year) %>%
    nest() %>%
    mutate(model = map(data, fit_ols))
```

```
out_le
```

```
## # A tibble: 60 x 4
##    continent  year                data    model
##       <fctr> <int>              <list>   <list>
##  1      Asia  1952 <tibble [33 x 4]> <S3: lm>
##  2      Asia  1957 <tibble [33 x 4]> <S3: lm>
##  3      Asia  1962 <tibble [33 x 4]> <S3: lm>
##  4      Asia  1967 <tibble [33 x 4]> <S3: lm>
##  5      Asia  1972 <tibble [33 x 4]> <S3: lm>
##  6      Asia  1977 <tibble [33 x 4]> <S3: lm>
##  7      Asia  1982 <tibble [33 x 4]> <S3: lm>
##  8      Asia  1987 <tibble [33 x 4]> <S3: lm>
##  9      Asia  1992 <tibble [33 x 4]> <S3: lm>
## 10      Asia  1997 <tibble [33 x 4]> <S3: lm>
## # ... with 50 more rows
```

```r
fit_ols <- function(df) {
    lm(lifeExp ~ log(gdpPercap), data = df)
}

out_tidy <- gapminder %>%
    group_by(continent, year) %>%
    nest() %>%
    mutate(model = map(data, fit_ols),
           tidied = map(model, tidy))

out_tidy %>% ungroup() %>% sample_n(5)
```

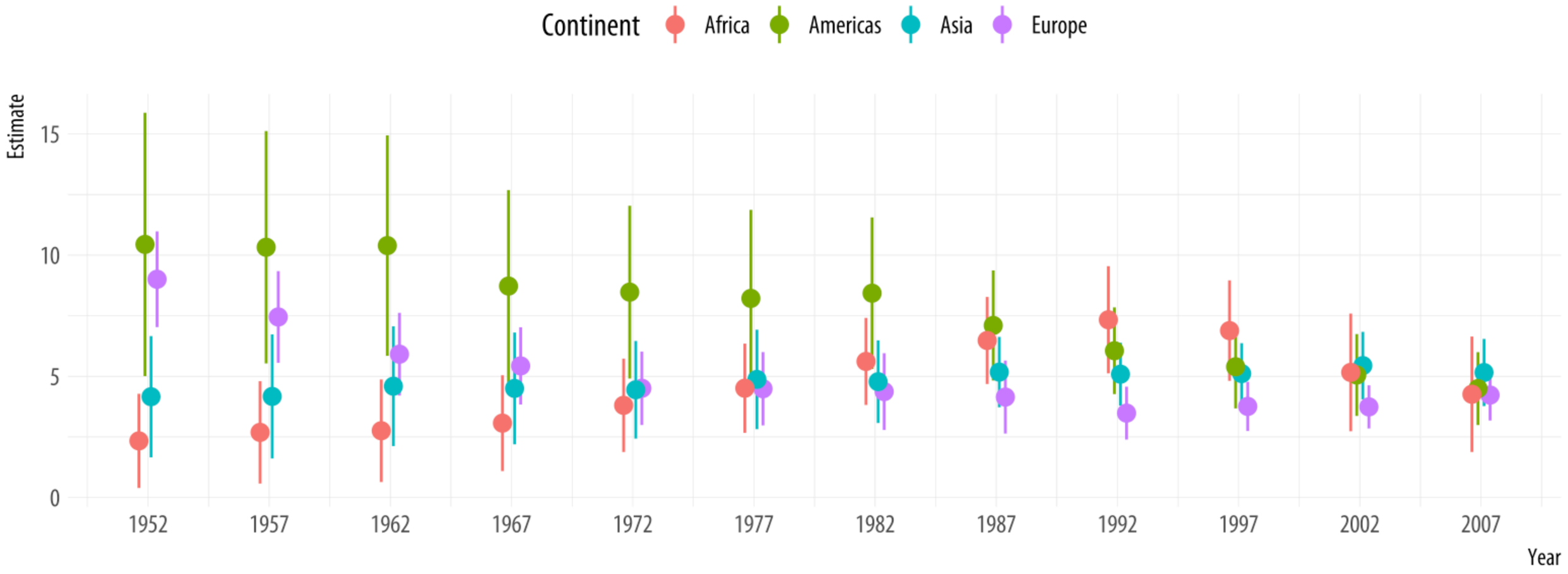
```r
fit_ols <- function(df) {
    lm(lifeExp ~ log(gdpPercap), data = df)
}

out_tidy <- gapminder %>%
    group_by(continent, year) %>%
    nest() %>%
    mutate(model = map(data, fit_ols),
           tidied = map(model, tidy)) %>%
    unnest(cols = c(tidied))

out_tidy %>% ungroup() %>% sample_n(5)
```

```r
fit_ols <- function(df) {
    lm(lifeExp ~ log(gdpPercap), data = df)
}

out_tidy <- gapminder %>%
    group_by(continent, year) %>%
    nest() %>%
    mutate(model = map(data, fit_ols),
           tidied = map(model, tidy)) %>%
    unnest(cols = c(tidied)) %>%
    filter(term %nin% "(Intercept)" &
           continent %nin% "Oceania")

out_tidy %>% ungroup() %>% sample_n(5)
```

```
out_tidy %>% ungroup() %>% sample_n(5)
```

```
## # A tibble: 5 x 7
##   continent  year                term estimate std.error
##      <fctr> <int>               <chr>    <dbl>     <dbl>
## 1    Europe  1982 log(gdpPercap)     4.37     0.791
## 2    Europe  1987 log(gdpPercap)     4.14     0.752
## 3    Africa  1972 log(gdpPercap)     3.80     0.962
## 4    Europe  1967 log(gdpPercap)     5.43     0.796
## 5      Asia  1952 log(gdpPercap)     4.16     1.251
## # ... with 2 more variables: statistic <dbl>, p.value <dbl>
```

```r
p <- ggplot(data = out_tidy,
            mapping = aes(x = year, y = estimate,
                          ymin = estimate - 2*std.error,
                          ymax = estimate + 2*std.error,
                          group = continent,
                          color = continent))

p + geom_pointrange(position = position_dodge(width = 1)) +
    scale_x_continuous(breaks = unique(gapminder$year)) +
    theme(legend.position = "top") +
    labs(x = "Year", y = "Estimate", color = "Continent")
```

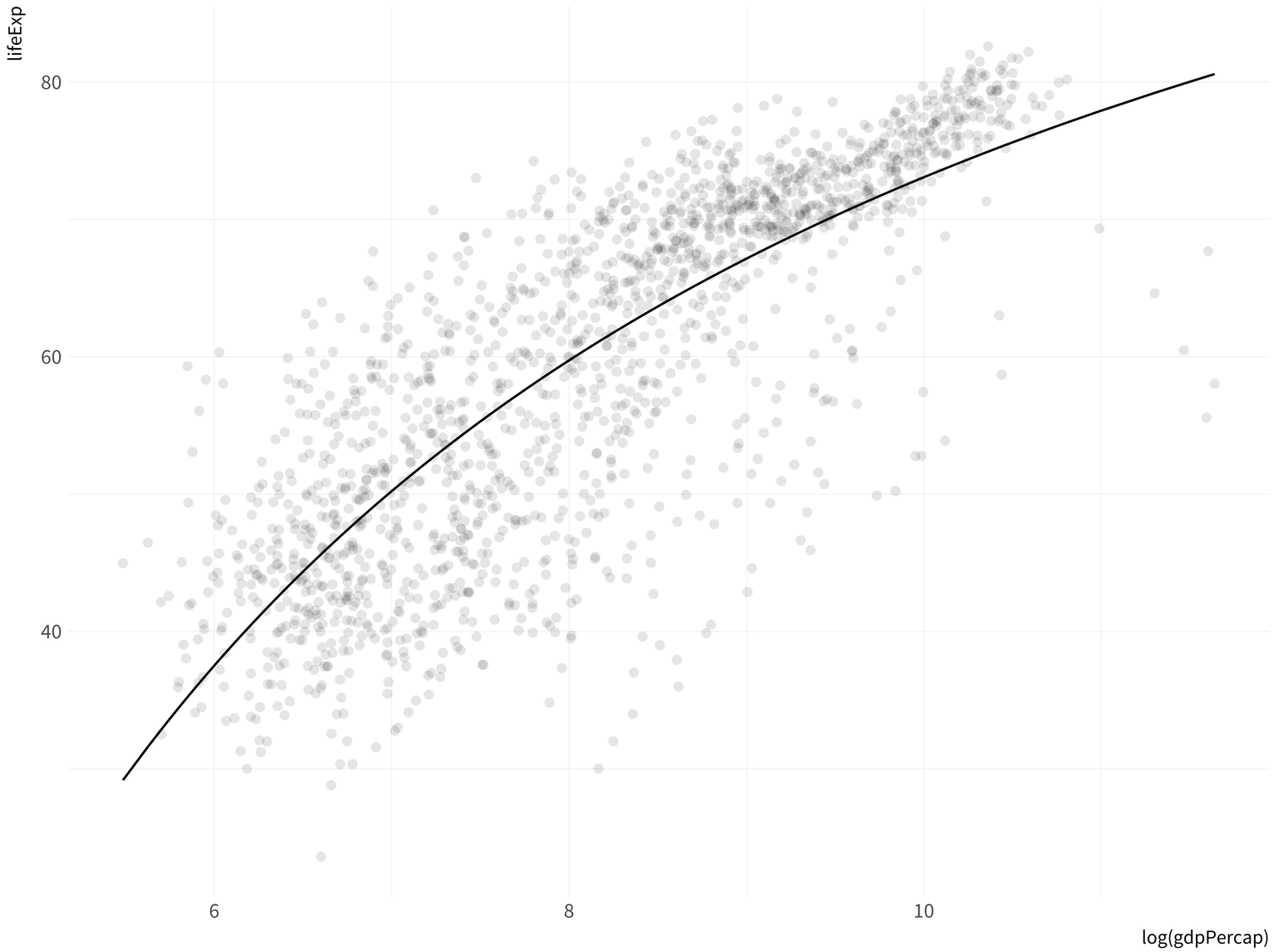

Continent
Africa
Americas
Asia
Europe
Estimate
15
10
5
0
1952
1957
1962
1967
1972
1977
1982
1987
1992
1997
2002
2007
Year

```
out_nls <- nls(lifeExp ~ k / log(gdpPercap) + b,
               data = gapminder, start=list(k=1, b=0))
summary(out_nls)
```

```
##
## Formula: lifeExp ~ k/log(gdpPercap) + b
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## k  -533.48       9.60   -55.5   <2e-16 ***
## b   126.42       1.22   103.7   <2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.7 on 1702 degrees of freedom
##
## Number of iterations to convergence: 1
## Achieved convergence tolerance: 3.44e-09
```

```
p <- ggplot(data = gapminder,
            mapping = aes(x = log(gdpPercap), y = lifeExp))

p + geom_point(alpha=0.1) +
    geom_line(aes(y=predict(out_nls)))
```

Predicted values from one nls model

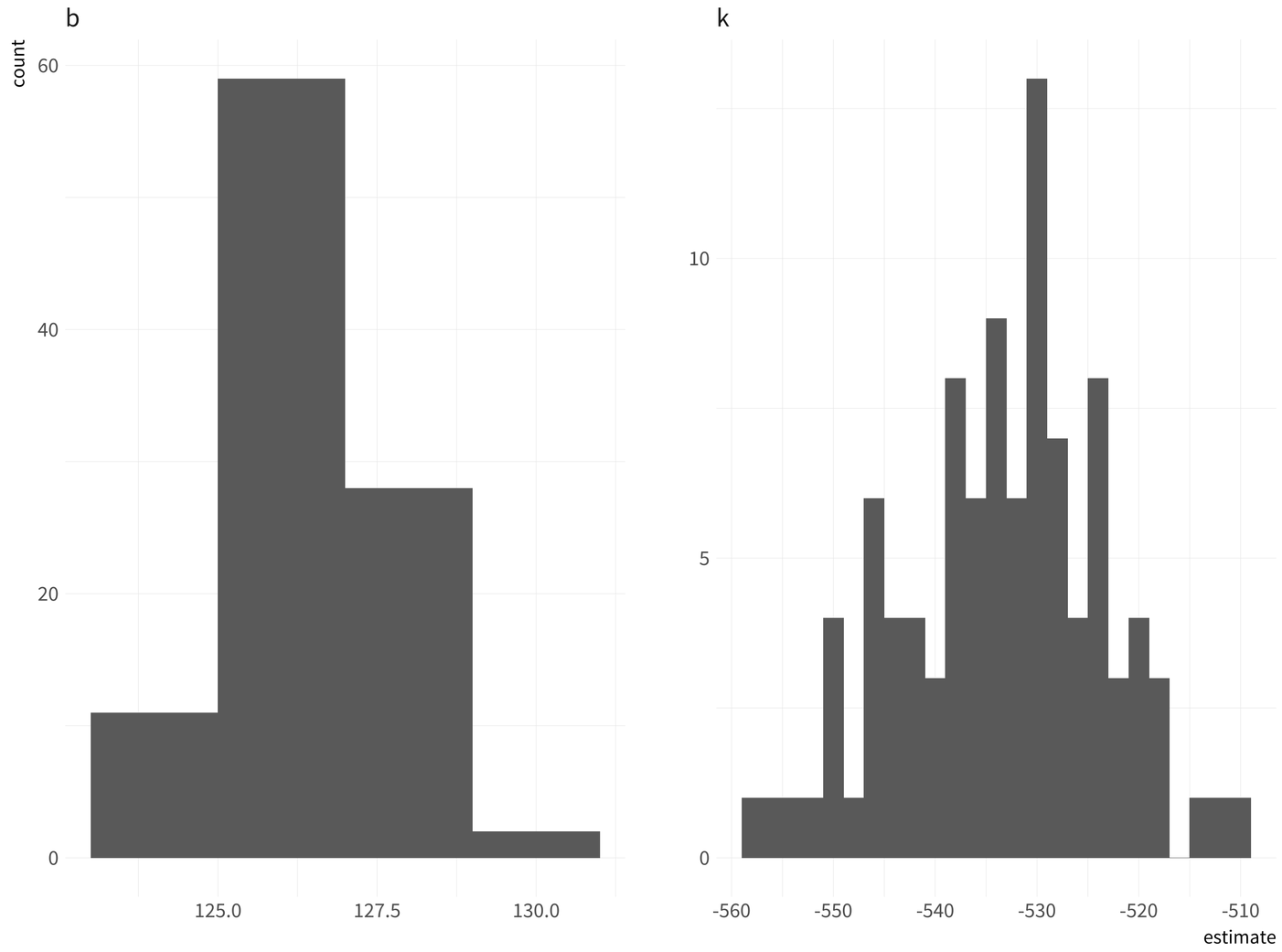

lifeExp
80
60
40
6
8
10
log(gdpPercap)

```
set.seed(112016)
out_bootnls <- gapminder %>% bootstrap(100) %>%
    do(tidy(nls(lifeExp ~ k / log(gdpPercap) + b, .,
                start=list(k=1, b=0))))

out_bootnls
```

```
## Source: local data frame [200 x 6]
## Groups: replicate [100]
##
## # A tibble: 200 x 6
##    replicate  term estimate std.error statistic p.value
##        <int> <chr>    <dbl>     <dbl>     <dbl>   <dbl>
##  1         1     k -529.199   9.98608  -52.9937       0
##  2         1     b  125.863   1.26055   99.8476       0
##  3         2     k -526.546   9.83457  -53.5403       0
##  4         2     b  125.335   1.24874  100.3694       0
##  5         3     k -521.846   9.71175  -53.7335       0
##  6         3     b  125.066   1.23666  101.1323       0
##  7         4     k -528.829   9.83972  -53.7443       0
##  8         4     b  125.915   1.23730  101.7665       0
##  9         5     k -517.459   9.68335  -53.4380       0
## 10         5     b  124.575   1.23188  101.1261       0
## # ... with 190 more rows
```

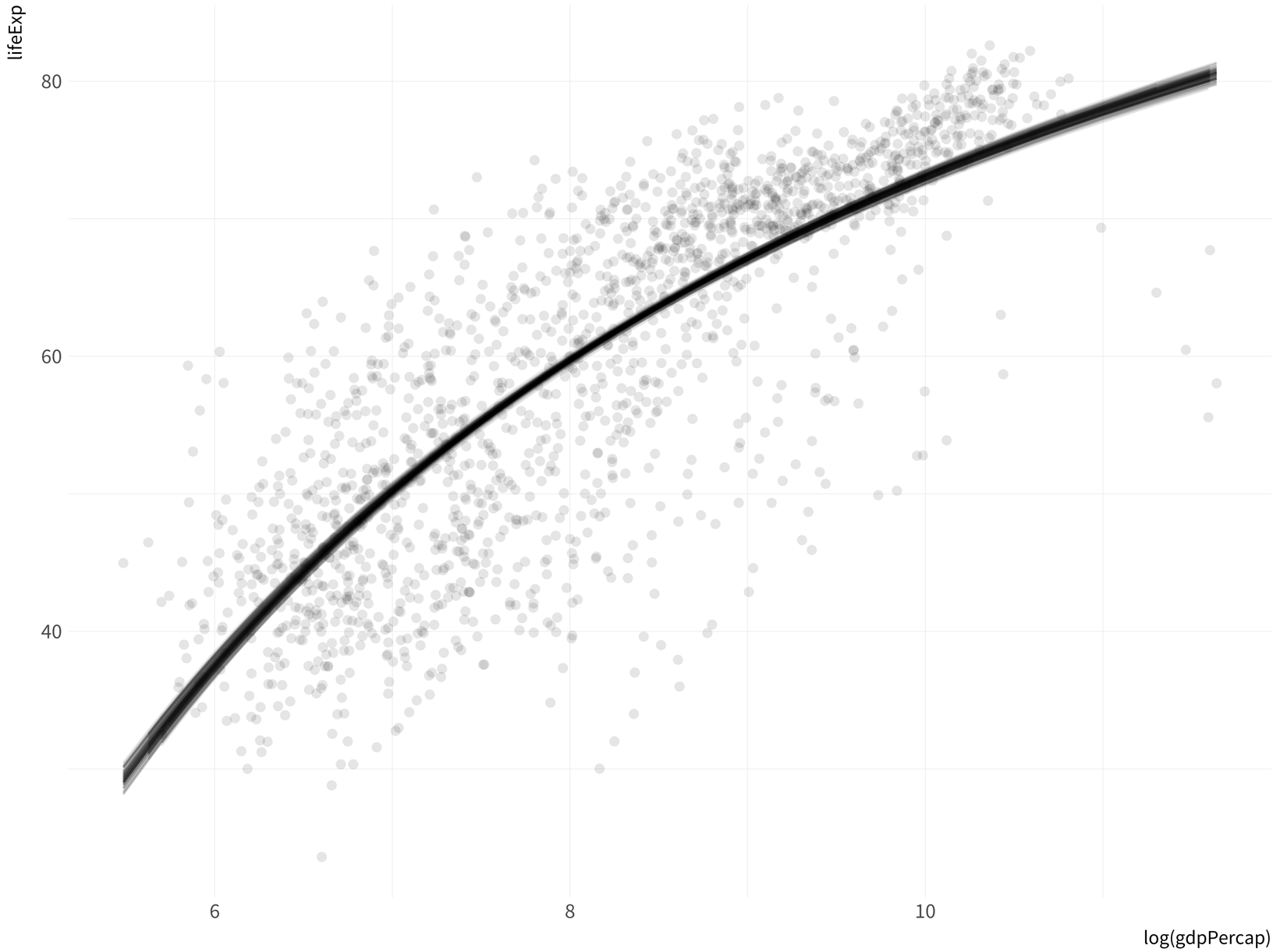
tidy() output from one hundred bootstrap replicates of this model

```r
p <- ggplot(out_bootnls, mapping = aes(estimate))
p + geom_histogram(binwidth=2) +
    facet_wrap(~ term, scales="free")
```



```
out_bootnls <- gapminder %>% bootstrap(100) %>%
    do(augment(nls(lifeExp ~ k / log(gdpPercap) + b, .,
                   start=list(k=1, b=0)), .))

p <-  ggplot(data = gapminder,
             mapping = aes(x = log(gdpPercap), y = lifeExp))

p + geom_point(alpha = 0.1) +
    geom_line(data = out_bootnls,
              mapping = aes(y=.fitted,
                            group=replicate), alpha=0.1)
```

Using augment() output from one hundred bootstrap replicates of the model


lifeExp
80
60
40
6
8
10
log(gdpPercap)

```
aug.bootspline <- gapminder %>% group_by(continent) %>%
    bootstrap(100, by_group = TRUE) %>%
    do(augment(nls(lifeExp ~ k / log(gdpPercap) + b, .,
                   start=list(k=1, b=0)), .))

p <-  ggplot(data = gapminder,
             mapping = aes(x = log(gdpPercap),
                           y = lifeExp, color = continent))

p + geom_point(alpha = 0.05, color = "black") +
    geom_line(data = aug.bootspline,
              mapping = aes(y = .fitted,
                            group = replicate), alpha=0.05) +
    facet_wrap(~ continent, nrow=1) +
    guides(fill=FALSE, color=FALSE)
```


Africa
Americas
Asia
Europe
Oceania
lifeExp
80
60
40
6
8
10
log(gdpPercap)

Grouped Analysis with broom: PCA example

midwest data

```
> midwest
# A tibble: 437 x 28
      PID county state  area poptotal popdensity popwhite popblack popamerindian popasian popother percwhite
    <int> <chr>  <chr> <dbl>    <int>      <dbl>    <int>    <int>          <int>    <int>    <int>     <dbl>
 1    561 ADAMS  IL    0.052    66090      1271.    63917     1702             98      249      124      96.7
 2    562 ALEXA… IL    0.014    10626       759      7054     3496             19       48        9      66.4
 3    563 BOND   IL    0.022    14991       681.    14477      429             35       16       34      96.6
 4    564 BOONE  IL    0.017    30806      1812.    29344      127             46      150     1139      95.3
 5    565 BROWN  IL    0.018     5836       324.     5264      547             14        5        6      90.2
 6    566 BUREAU IL    0.05     35688       714.    35157       50             65      195      221      98.5
 7    567 CALHO… IL    0.017     5322       313.     5298        1              8       15        0      99.5
 8    568 CARRO… IL    0.027    16805       622.    16519      111             30       61       84      98.3
 9    569 CASS   IL    0.024    13437       560.    13384       16              8       23        6      99.6
10    570 CHAMP… IL    0.058   173025      2983.   146506    16559            331     8033     1596      84.7
# … with 427 more rows, and 16 more variables: percblack <dbl>, percamerindan <dbl>, percasian <dbl>,
#   percother <dbl>, popadults <int>, perchsd <dbl>, percollege <dbl>, percprof <dbl>, poppovertyknown <int>,
#   percpovertyknown <dbl>, percbelowpoverty <dbl>, percchildbelowpovert <dbl>, percadultpoverty <dbl>,
#   percelderlypoverty <dbl>, inmetro <int>, category <chr>
```

```
mw_pca <- midwest %>%
    group_by(state) %>%
    select_if(is.numeric) %>%
    select(-PID)
```

Group by state

Choose only the numeric variables

Drop the id variable

```
mw_pca
```

```
## # A tibble: 437 x 25
## # Groups:   state [5]
##    state  area poptotal popdensity popwhite popblack popamerindian popasian
##    <chr> <dbl>    <int>      <dbl>    <int>    <int>          <int>    <int>
##  1 IL    0.052    66090      1271.    63917     1702             98      249
##  2 IL    0.014    10626       759      7054     3496             19       48
##  3 IL    0.022    14991       681.    14477      429             35       16
##  4 IL    0.017    30806      1812.    29344      127             46      150
##  5 IL    0.018     5836       324.     5264      547             14        5
##  6 IL    0.05     35688       714.    35157       50             65      195
##  7 IL    0.017     5322       313.     5298        1              8       15
##  8 IL    0.027    16805       622.    16519      111             30       61
##  9 IL    0.024    13437       560.    13384       16              8       23
## 10 IL    0.058   173025      2983.   146506    16559            331     8033

## #   percwhite <dbl>, percblack <dbl>, percamerindan <dbl>,
## #   percasian <dbl>, percother <dbl>, popadults <int>, perchsd <dbl>,
## #   percollege <dbl>, percprof <dbl>, poppovertyknown <int>,
## #   percpovertyknown <dbl>, percbelowpoverty <dbl>,
## #   percchildbelowpovert <dbl>, percadultpoverty <dbl>,
## #   percelderlypoverty <dbl>, inmetro <int>
```

```
do_pca <- function(df){
  prcomp(df,
         center = TRUE, scale = TRUE)
}
```

A function to run a PCA

```
out_pca <- mw_pca %>%
    ungroup() %>%
    select(-state) %>%
    do_pca()
```

Try it on the full, ungrouped dataset first

```
out_pca
## Standard deviations (1, .., p=24):
##  [1] 3.098601e+00 2.209601e+00 1.649472e+00 1.192893e+00 1.121586e+00
##  [6] 8.977640e-01 8.859441e-01 8.194829e-01 6.921234e-01 5.649742e-01
## [11] 5.439431e-01 4.854143e-01 3.799956e-01 3.583348e-01 3.094795e-01
## [16] 2.500930e-01 2.087861e-01 1.924391e-01 9.654481e-02 3.473006e-02
## [21] 1.328238e-02 3.862376e-03 2.886151e-09 5.193302e-16
##
## Rotation (n x k) = (24 x 24):
##                                PC1           PC2           PC3          PC4
## area                   -0.026194151  0.014996226 -0.103615981  0.25080525
## poptotal               -0.312990340  0.051517565  0.110772033  0.05437383
## popdensity             -0.292047651  0.023597548  0.045576790 -0.10421671
## popwhite               -0.314102501  0.023751463  0.081172077  0.02779283
```

What we get back when we run prcomp()

Name	Type	Value
out_pca	list [5] (S3: prcomp)	List of length 5
sdev	double [24]	3.099 2.210 1.649 1.193 1.122 0.898 ...
rotation	double [24 x 24]	-2.62e-02 -3.13e-01 -2.92e-01 -3.14e-01 -2.88e-01 -2.69e-01 1.50e-02 5.15e-02 ...
center	double [24]	3.32e-02 9.61e+04 3.10e+03 8.18e+04 1.10e+04 3.43e+02 ...
scale	[illegible]	1.47e-02 2.98e+05 7.66e+03 2.00e+05 7.90e+04 8.69e+02 ...
x	double [437 x 24]	5.25e-01 -3.69e-02 1.07e+00 -3.82e-01 9.76e-01 7.17e-01 2.00e-01 8.45e+00 ...

```r
do_pca <- function(df){
  out <- prcomp(df, center = TRUE, scale = TRUE)
  out$x <- data.frame(out$x)
  out
}
```

Workaround for CRAN version of broom

Try it on the full, ungrouped dataset first

```r
out_pca <- mw_pca %>%
    ungroup() %>%
    select(-state) %>%
    do_pca()
```

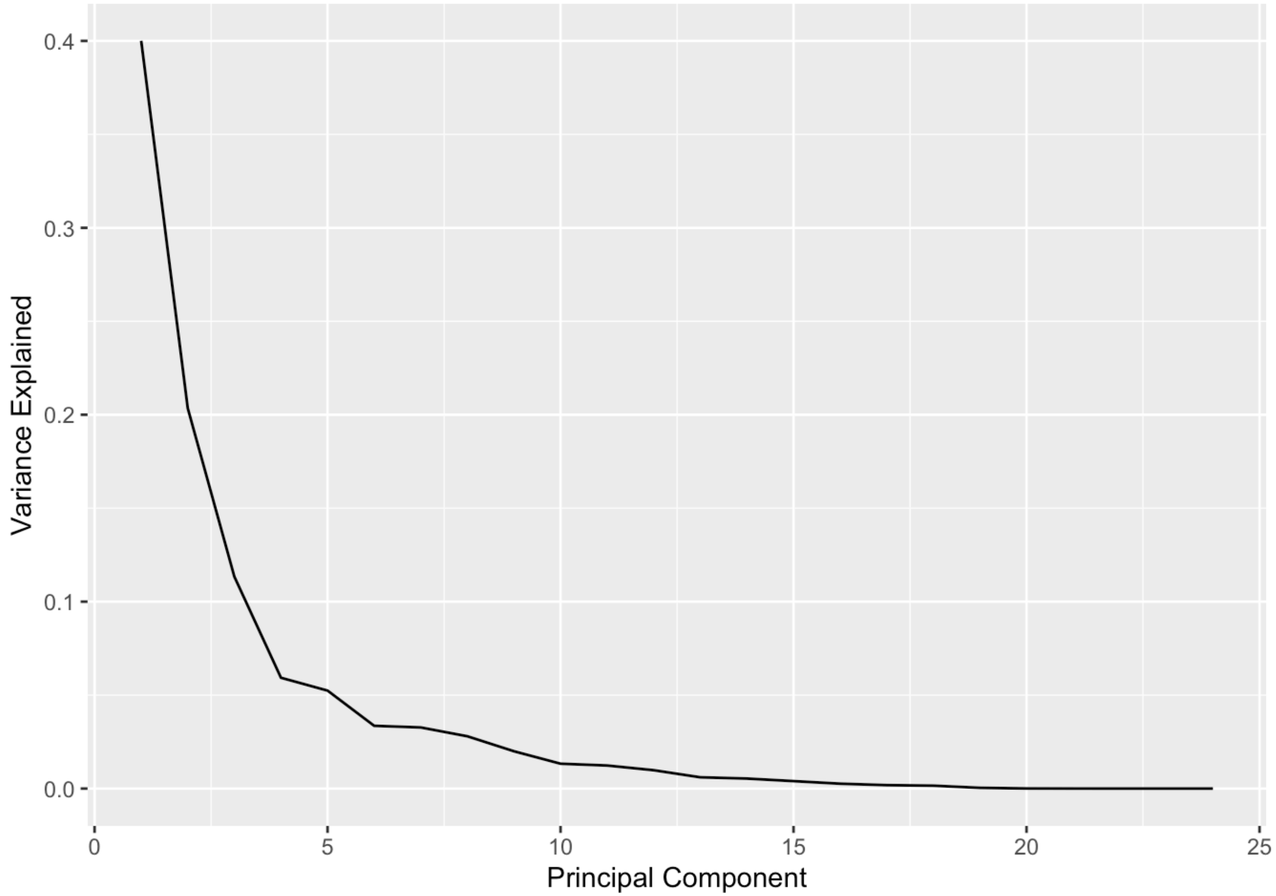
```r
out_pca
## Standard deviations (1, .., p=24):
##  [1] 3.098601e+00 2.209601e+00 1.649472e+00 1.192893e+00 1.121586e+00
##  [6] 8.977640e-01 8.859441e-01 8.194829e-01 6.921234e-01 5.649742e-01
## [11] 5.439431e-01 4.854143e-01 3.799956e-01 3.583348e-01 3.094795e-01
## [16] 2.500930e-01 2.087861e-01 1.924391e-01 9.654481e-02 3.473006e-02
## [21] 1.328238e-02 3.862376e-03 2.886151e-09 5.193302e-16
##
## Rotation (n x k) = (24 x 24):
##                                PC1           PC2           PC3          PC4
## area                   -0.026194151  0.014996226 -0.103615981  0.25080525
## poptotal               -0.312990340  0.051517565  0.110772033  0.05437383
## popdensity             -0.292047651  0.023597548  0.045576790 -0.10421671
## popwhite               -0.314102501  0.023751463  0.081172077  0.02779283
```

Name	Type	Value
out_pca	list [5] (S3: prcomp)	List of length 5
sdev	double [24]	3.099 2.210 1.649 1.193 1.122 0.898 ...
rotation	double [24 x 24]	-2.62e-02 -3.13e-01 -2.92e-01 -3.14e-01 -2.88e-01 -2.69e-01
center	double [24]	3.32e-02 9.61e+04 3.10e+03 8.18e+04 1.10e+04 3.43e+02 ...
scale	double [24]	1.47e-02 2.98e+05 7.66e+03 2.00e+05 7.90e+04 8.69e+02 ...
x	list [437 x 24] (S3: data.frame	A data.frame with 437 rows and 24 columns

```
summary(out_pca)
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6    PC7
## Standard deviation     3.0986 2.2096 1.6495 1.19289 1.12159 0.89776 0.8859
## Proportion of Variance 0.4001 0.2034 0.1134 0.05929 0.05241 0.03358 0.0327
## Cumulative Proportion  0.4001 0.6035 0.7168 0.77614 0.82856 0.86214 0.8948
##                            PC8     PC9   PC10    PC11    PC12    PC13
## Standard deviation     0.81948 0.69212 0.5650 0.54394 0.48541 0.38000
## Proportion of Variance 0.02798 0.01996 0.0133 0.01233 0.00982 0.00602
## Cumulative Proportion  0.92283 0.94278 0.9561 0.96841 0.97823 0.98425
##                           PC14    PC15    PC16    PC17    PC18    PC19
## Standard deviation     0.35833 0.30948 0.25009 0.20879 0.19244 0.09654
## Proportion of Variance 0.00535 0.00399 0.00261 0.00182 0.00154 0.00039
## Cumulative Proportion  0.98960 0.99359 0.99619 0.99801 0.99955 0.99994
##                           PC20    PC21     PC22      PC23      PC24
## Standard deviation     0.03473 0.01328 0.003862 2.886e-09 5.193e-16
## Proportion of Variance 0.00005 0.00001 0.000000 0.000e+00 0.000e+00
## Cumulative Proportion  0.99999 1.00000 1.000000 1.000e+00 1.000e+00
```

Tidy and Plot

```
tidy_pca <- tidy(out_pca, matrix = "pcs")

tidy_pca
## # A tibble: 24 x 4
##       PC std.dev percent cumulative
##    <dbl>   <dbl>   <dbl>      <dbl>
##  1     1    3.10   0.400      0.400
##  2     2    2.21   0.203      0.603
##  3     3    1.65   0.113      0.717
##  4     4    1.19   0.0593     0.776
##  5     5    1.12   0.0524     0.829
##  6     6    0.898  0.0336     0.862
##  7     7    0.886  0.0327     0.895
##  8     8    0.819  0.0280     0.923
##  9     9    0.692  0.0200     0.943
## 10    10    0.565  0.0133     0.956
```




```
tidy_pca %>%
    ggplot(aes(x = PC, y = percent)) +
    geom_line() +
    labs(x = "Principal Component", y = "Variance Explained")
```

Nest the data by state ...

```
mw_pca <- mw_pca %>%
    group_by(state) %>%
    nest()
```

```
mw_pca
## # A tibble: 5 x 2
## # Groups:   state [5]
##   state             data
##   <chr> <list<df[,24]>>
## 1 IL          [102 x 24]
## 2 IN           [92 x 24]
## 3 MI           [83 x 24]
## 4 OH           [88 x 24]
## 5 WI           [72 x 24]
```

... and run a PCA on each row

```
state_pca <- mw_pca %>%
    mutate(pca = map(data, do_pca))
```

```
state_pca
## # A tibble: 5 x 3
## # Groups:   state [5]
##   state             data pca
##   <chr> <list<df[,24]>> <list>
## 1 IL          [102 x 24] <prcomp>
## 2 IN           [92 x 24] <prcomp>
## 3 MI           [83 x 24] <prcomp>
## 4 OH           [88 x 24] <prcomp>
## 5 WI           [72 x 24] <prcomp>
```

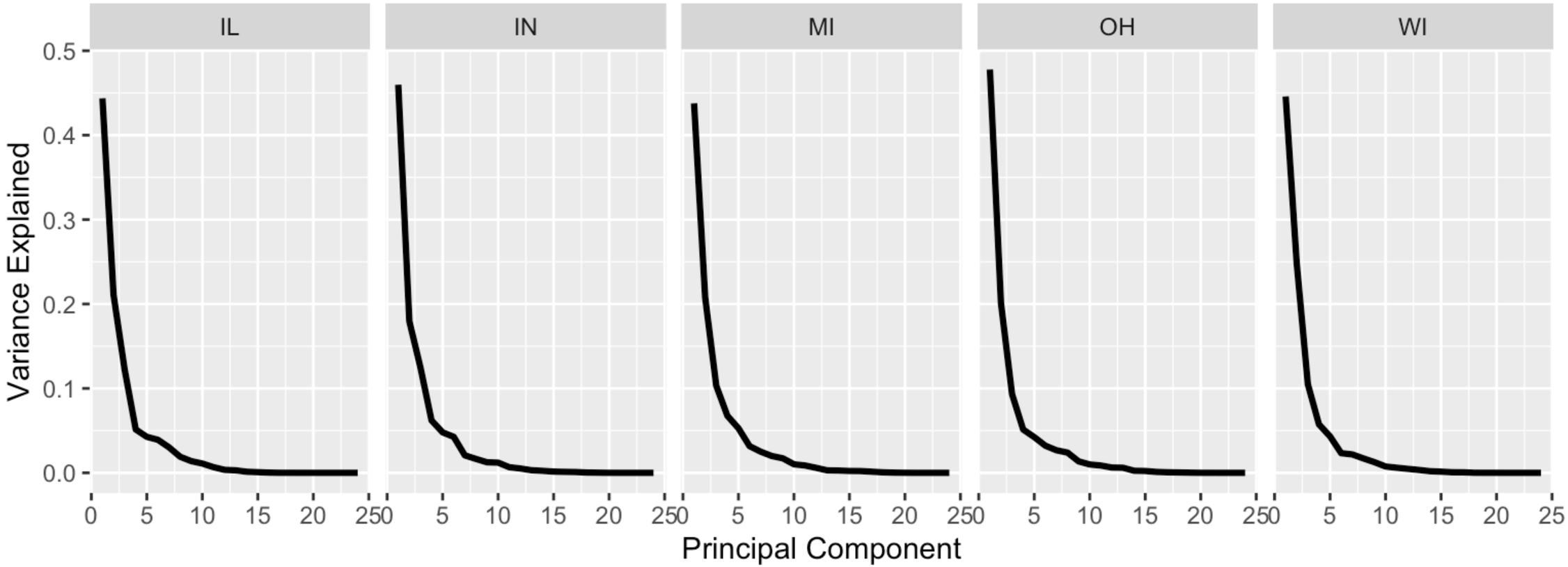
Tidy the grouped PCAs in the same mutate() call

```
do_tidy <- function(pr){
    broom::tidy(pr, matrix = "pcs")
}
state_pca  <- mw_pca %>%
    mutate(pca = map(data, do_pca),
           pcs = map(pca, do_tidy))

state_pca
```

```
## # A tibble: 5 x 4
## # Groups:   state [5]
##   state             data pca       pcs
##   <chr> <list<df[,24]>> <list>    <list>
## 1 IL          [102 x 24] <prcomp> <tibble [24 x 4]>
## 2 IN           [92 x 24] <prcomp> <tibble [24 x 4]>
## 3 MI           [83 x 24] <prcomp> <tibble [24 x 4]>
## 4 OH           [88 x 24] <prcomp> <tibble [24 x 4]>
## 5 WI           [72 x 24] <prcomp> <tibble [24 x 4]>
```

... And plot the results

```
state_pca %>%
    unnest(cols = c(pcs)) %>%
    ggplot(aes(x = PC, y = percent)) +
    geom_line(size = 1.1) +
    facet_wrap(~ state, nrow = 1) +
    labs(x = "Principal Component",
         y = "Variance Explained")
```



We can augment() as well as tidy()

```
do_aug <- function(pr){
    broom::augment(pr)
}
```

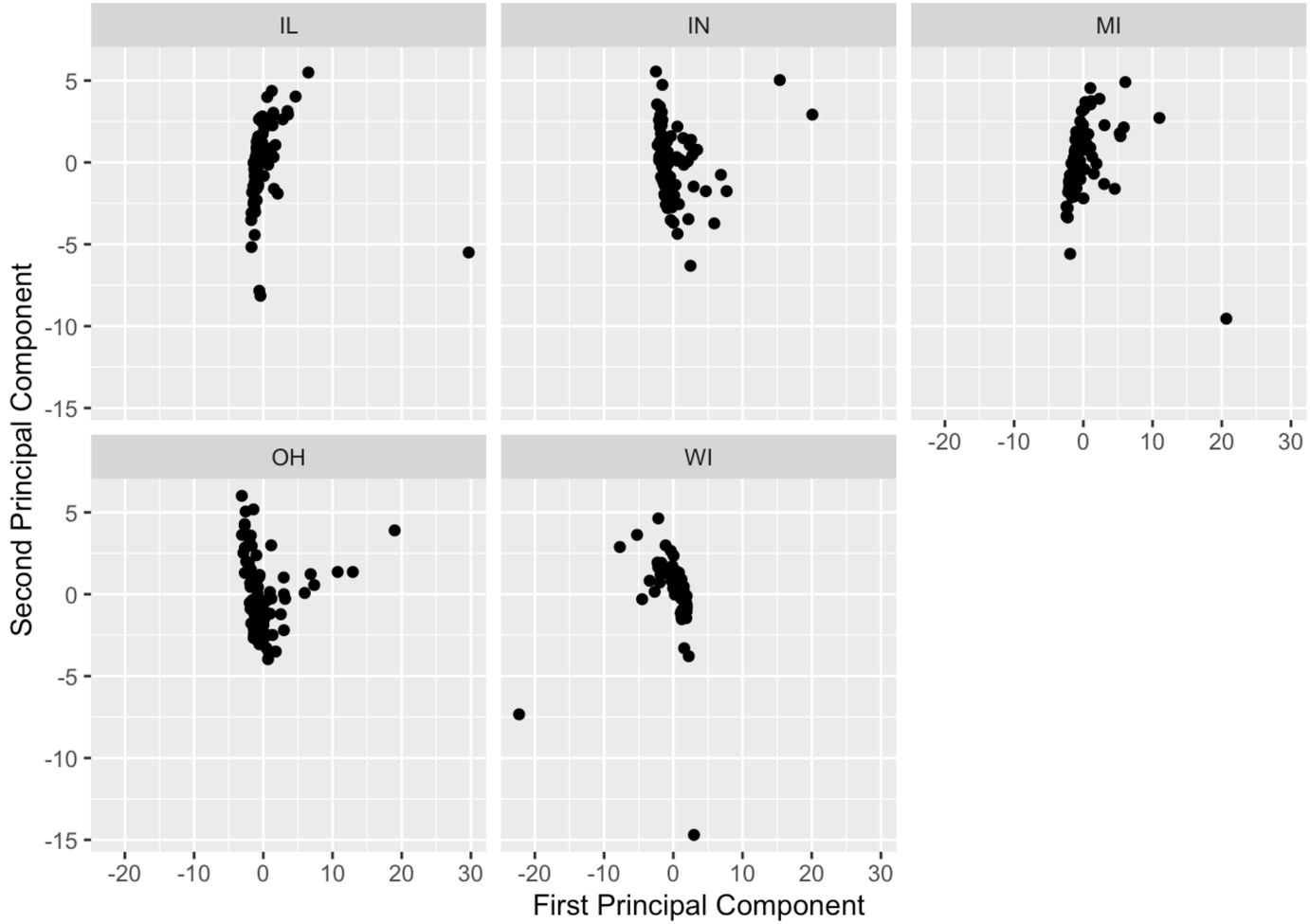
```
state_pca  <- mw_pca %>%
    mutate(pca = map(data, do_pca),
           pcs = map(pca, do_tidy),
           fitted = map(pca, do_aug))
```

```
state_pca
## # A tibble: 5 x 5
## # Groups:   state [5]
##   state             data pca      pcs                fitted
##   <chr> <list<df[, 24]>> <list>   <list>             <list>
## 1 IL          [102 x 24] <prcomp> <tibble [24 x 4]>  tibble [102 x 25]>
## 2 IN           [92 x 24] <prcomp> <tibble [24 x 4]>  tibble [92 x 25]>
## 3 MI           [83 x 24] <prcomp> <tibble [24 x 4]>  tibble [83 x 25]>
## 4 OH           [88 x 24] <prcomp> <tibble [24 x 4]>  tibble [88 x 25]>
## 5 WI           [72 x 24] <prcomp> <tibble [24 x 4]>  tibble [72 x 25]>
```

Nesting allows us to tidily store results of varying dimensions in the same tibble

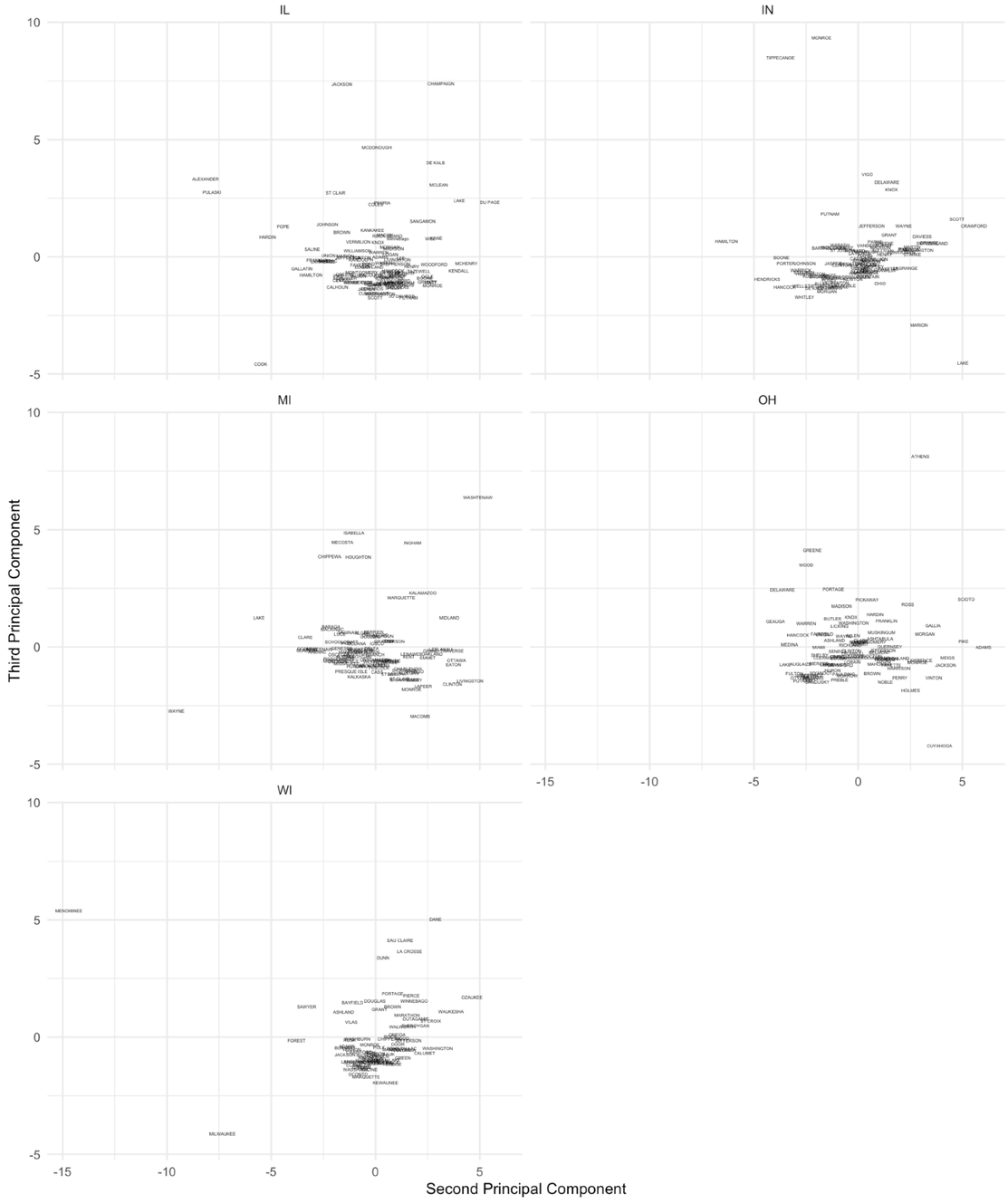
Unnest a sub-table to plot the augmented output

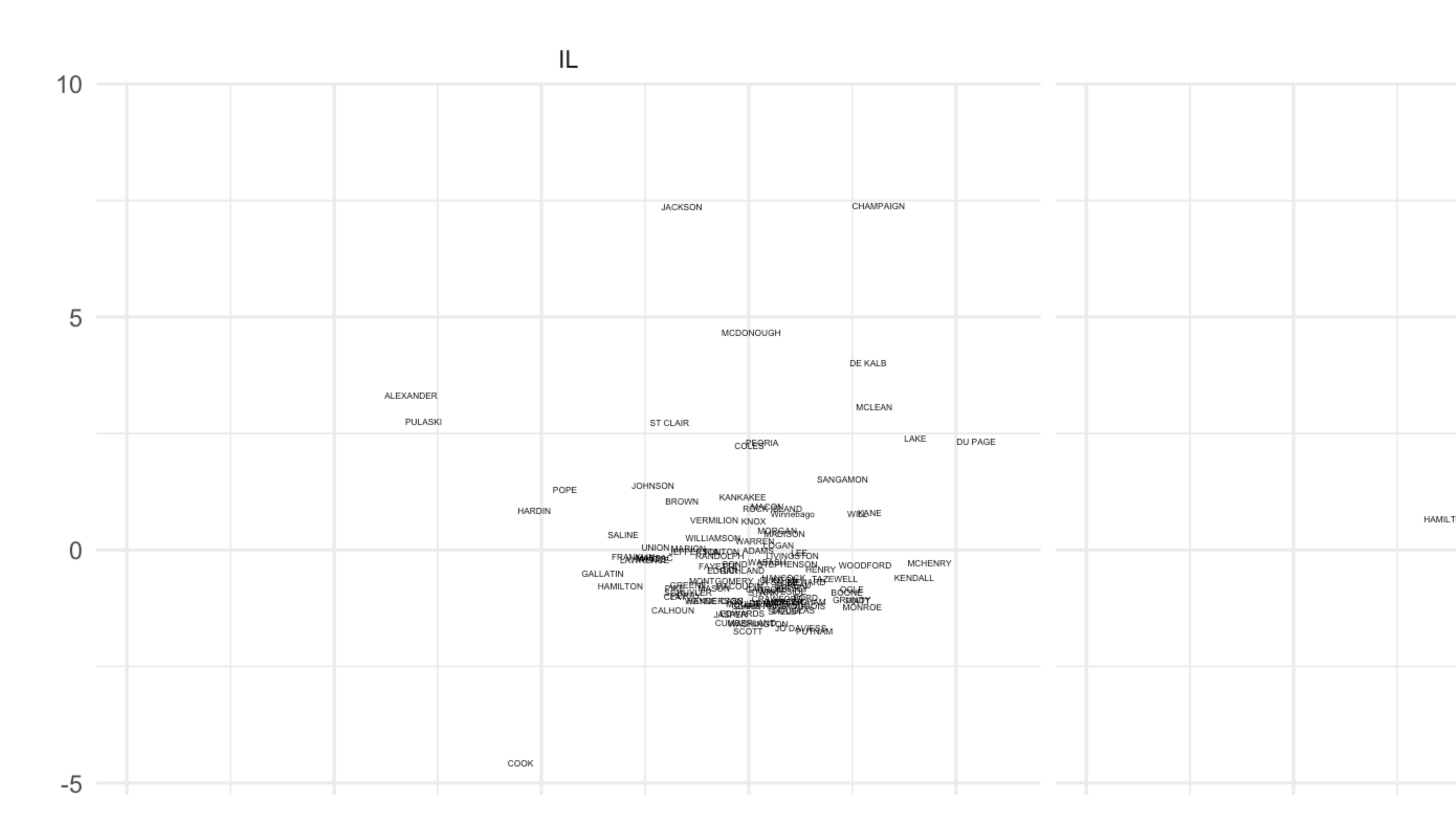
```
state_pca %>%
    unnest(cols = c(fitted)) %>%
    ggplot(aes(x = .fittedPC1,
               y = .fittedPC2)) +
    geom_point() +
    facet_wrap(~ state) +
    labs(x = "First Principal Component",
         y = "Second Principal Component")
```



From start to finish in a single breath

```
midwest %>%
    group_by(state) %>%
    select_if(is.numeric) %>%
    select(-PID) %>%
    nest() %>%
    mutate(pca = map(data, do_pca),
           pcs = map(pca, do_tidy),
           fitted = map(pca, do_aug)) %>%
    unnest(cols = c(fitted)) %>%
    add_column(county = midwest$county) %>%
    ggplot(mapping = aes(x = .fittedPC2,
                y = .fittedPC3,
                label = county)) +
    geom_text(size = 1.1) +
    labs(x = "Second Principal Component",
         y = "Third Principal Component") +
    theme_minimal() + facet_wrap(~ state, ncol = 2)
```




IL
10
5
0
-5
JACKSON
CHAMPAIGN
MCDONOUGH
DE KALB
ALEXANDER
MCLEAN
PULASKI
ST CLAIR
LAKE
DU PAGE
PEORIA
COLES
SANGAMON
POPE
JOHNSON
KANKAKEE
BROWN
HARDIN
KANE
VERMILION
KNOX
SALINE
WILLIAMSON
WARREN
UNION
ADAMS
LEE
RANDOLPH
WOODFORD
MCHENRY
GALLATIN
HENRY
KENDALL
HAMILTON
MONROE
CALHOUN
SCOTT
PUTNAM
COOK

PCA detour

```r
install.packages("imager")
library(imager)
library(here)
```

```r
img <- load.image(here("assets/elvis-nixon.jpeg"))
str(img)
```

```
##  'cimg' num [1:800, 1:633, 1, 1] 0.914 0.929 0.91 0.906 0.898 ...
```

```r
dim(img)
```

```
## [1] 800 633   1   1
```

```r
img_df_long <- as.data.frame(img)
```

```r
head(img_df_long)
```

```
##   x y     value
## 1 1 1 0.9137255
## 2 2 1 0.9294118
## 3 3 1 0.9098039
## 4 4 1 0.9058824
## 5 5 1 0.8980392
## 6 6 1 0.8862745
```

```r
img_df <- tidyr::pivot_wider(img_df_long,
                             names_from = y,
                             values_from = value)
```

```r
dim(img_df)
```

```
## [1] 800 634
```

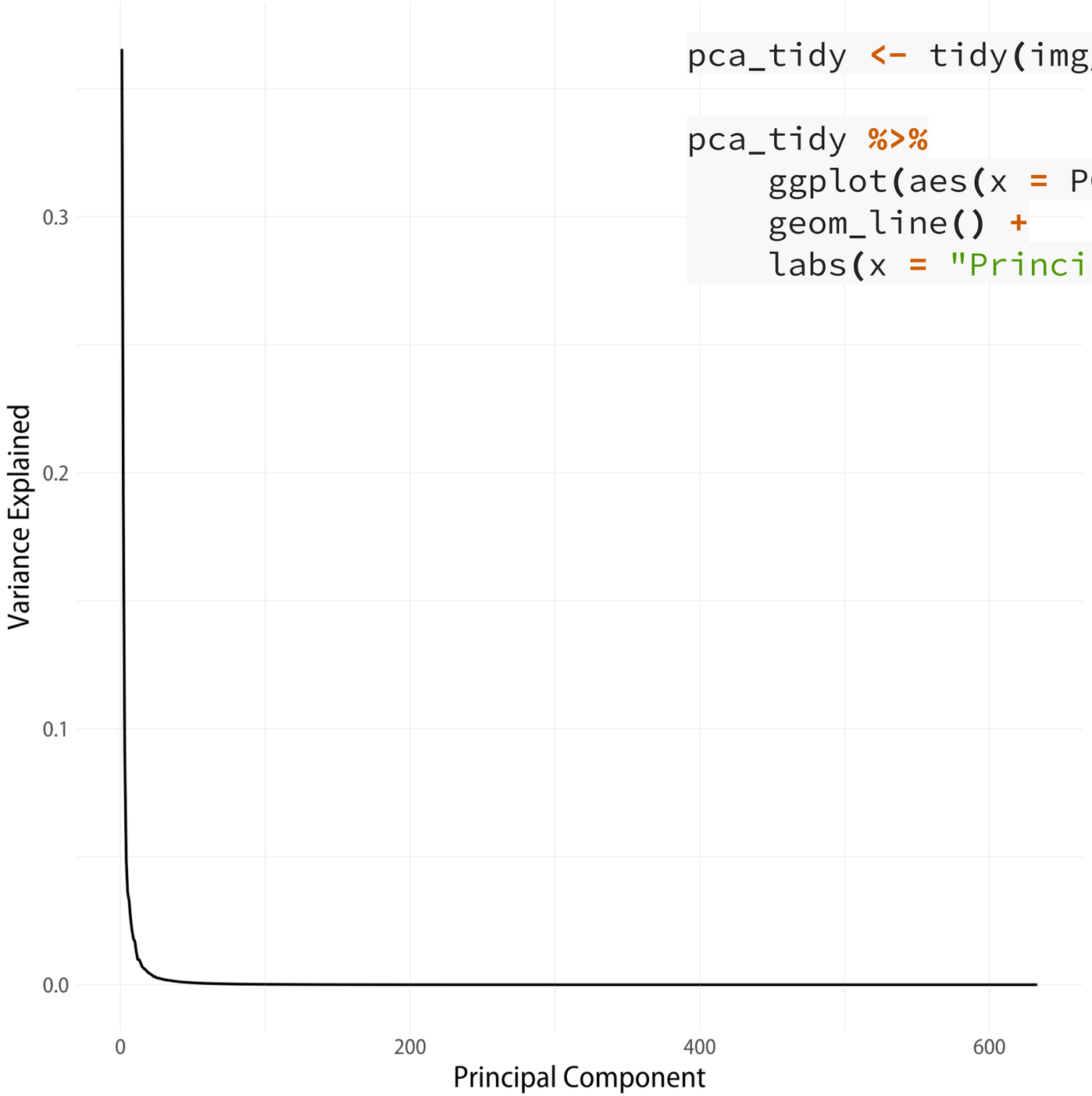
```r
img_df[1:5, 1:5]
```

```
## # A tibble: 5 x 5
##       x   `1`   `2`   `3`   `4`
##   <int> <dbl> <dbl> <dbl> <dbl>
## 1     1 0.914 0.914 0.914 0.910
## 2     2 0.929 0.929 0.925 0.918
## 3     3 0.910 0.910 0.902 0.894
## 4     4 0.906 0.902 0.898 0.894
## 5     5 0.898 0.894 0.890 0.886
```

```r
img_pca <- img_df %>%
  dplyr::select(-x) %>%
  prcomp(scale = TRUE, center = TRUE)
```

```r
summary(img_pca)
```

```
## Importance of components:
##                            PC1      PC2      PC3      PC4      PC5      PC6
## Standard deviation     15.2124 10.9823 7.54308 5.57239 4.77759 4.55531
## Proportion of Variance  0.3656  0.1905 0.08989 0.04905 0.03606 0.03278
## Cumulative Proportion   0.3656  0.5561 0.64601 0.69506 0.73112 0.76391
##                            PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     4.0649 3.66116 3.36891 3.27698 2.82984 2.49643
## Proportion of Variance 0.0261 0.02118 0.01793 0.01696 0.01265 0.00985
## Cumulative Proportion  0.7900 0.81118 0.82911 0.84608 0.85873 0.86857

## [A lot more components]

##                           PC628    PC629    PC630    PC631    PC632
## Standard deviation     0.001125 0.001104 0.001097 0.001037 0.000993
## Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.000000
## Cumulative Proportion  1.000000 1.000000 1.000000 1.000000 1.000000
##                           PC633
## Standard deviation     0.0009215
## Proportion of Variance 0.0000000
## Cumulative Proportion  1.0000000
```




```r
pca_tidy <- tidy(img_pca, matrix = "pcs")

pca_tidy %>%
    ggplot(aes(x = PC, y = percent)) +
    geom_line() +
    labs(x = "Principal Component", y = "Variance Explained")
```

```r
names(img_pca)
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

```r
reverse_pca <- function(n_comp = 20, pca_object = img_pca){

 recon <- pca_object$x[, 1:n_comp] %*% t(pca_object$rotation[, 1:n_comp])

 if(all(pca_object$scale != FALSE)){
   recon <- scale(recon, center = FALSE,
                  scale = 1/pca_object$scale)
  }
  if(all(pca_object$center != FALSE)){
   recon <- scale(recon, scale = FALSE,
                         center = -1 * pca_object$center)
  }

 recon_df <- data.frame(cbind(1:nrow(recon), recon))
  colnames(recon_df) <- c("x", 1:(ncol(recon_df)-1))

 recon_df_long <- recon_df %>%
    tidyr::pivot_longer(cols = -x, names_to = "y",
                        values_to = "value") %>%
    mutate(y = as.numeric(y)) %>%
    arrange(y) %>%
    as.data.frame()

  recon_df_long
}
```

Multiply the matrix of rotated data by the transpose of the loadings to get back to a matrix of original data values

Remove any scaling

Remove any centering

Make the matrix of results into a data frame we can work with

Convert the data frame to a long-format tibble

Return the tidied object

```
n_pcs <- c(2:5, 10, 20, 50, 100)
names(n_pcs) <- paste("First", n_pcs,
                      "Components", sep = "_")
```

The sequence of # components we want: 2, 3, 4, 5, 10, 20, 50, 100.

```
recovered_imgs <- map_dfr(n_pcs,
                          reverse_pca,
                          .id = "pcs") %>%
  mutate(pcs = stringr::str_replace_all(pcs, "_", " "),
         pcs = factor(pcs, levels = unique(pcs),
                      ordered = TRUE))
```

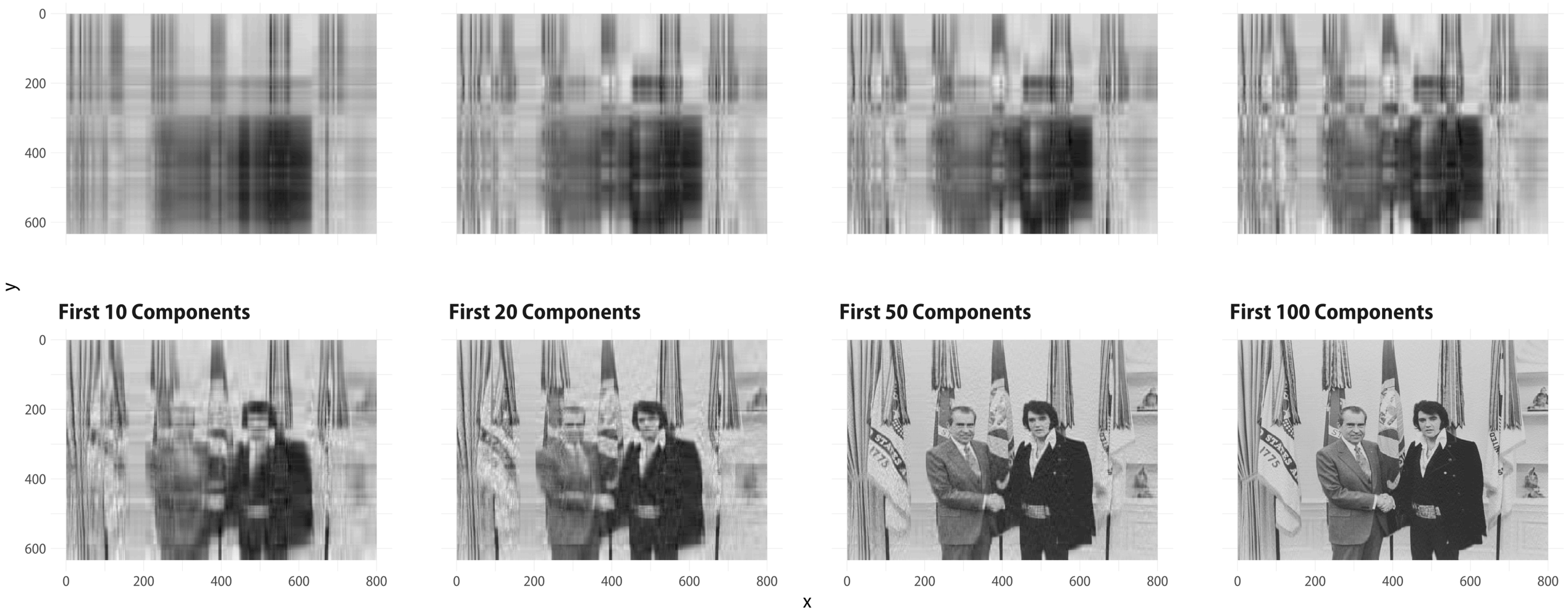
Use map_dfr() to feed this sequence to reverse_pca(). Keep track of the sequence with an id variable called 'pcs' that we convert to a factor.

```
p <- ggplot(data = recovered_imgs,
            mapping = aes(x = x, y = y, fill = value))
p_out <- p + geom_raster() +
  scale_y_reverse() +
  scale_fill_gradient(low = "black", high = "white") +
  facet_wrap(~ pcs, ncol = 4) +
  guides(fill = FALSE) +
  labs(title = "Recovering the content of an 800x600 pixel image\n
                from a Principal Components Analysis of its pixels") +
  theme(strip.text = element_text(face = "bold", size = rel(1.2)),
        plot.title = element_text(size = rel(1.5)))
```

Plot the results, faceting by pcs. Note use of scale_y_reverse()

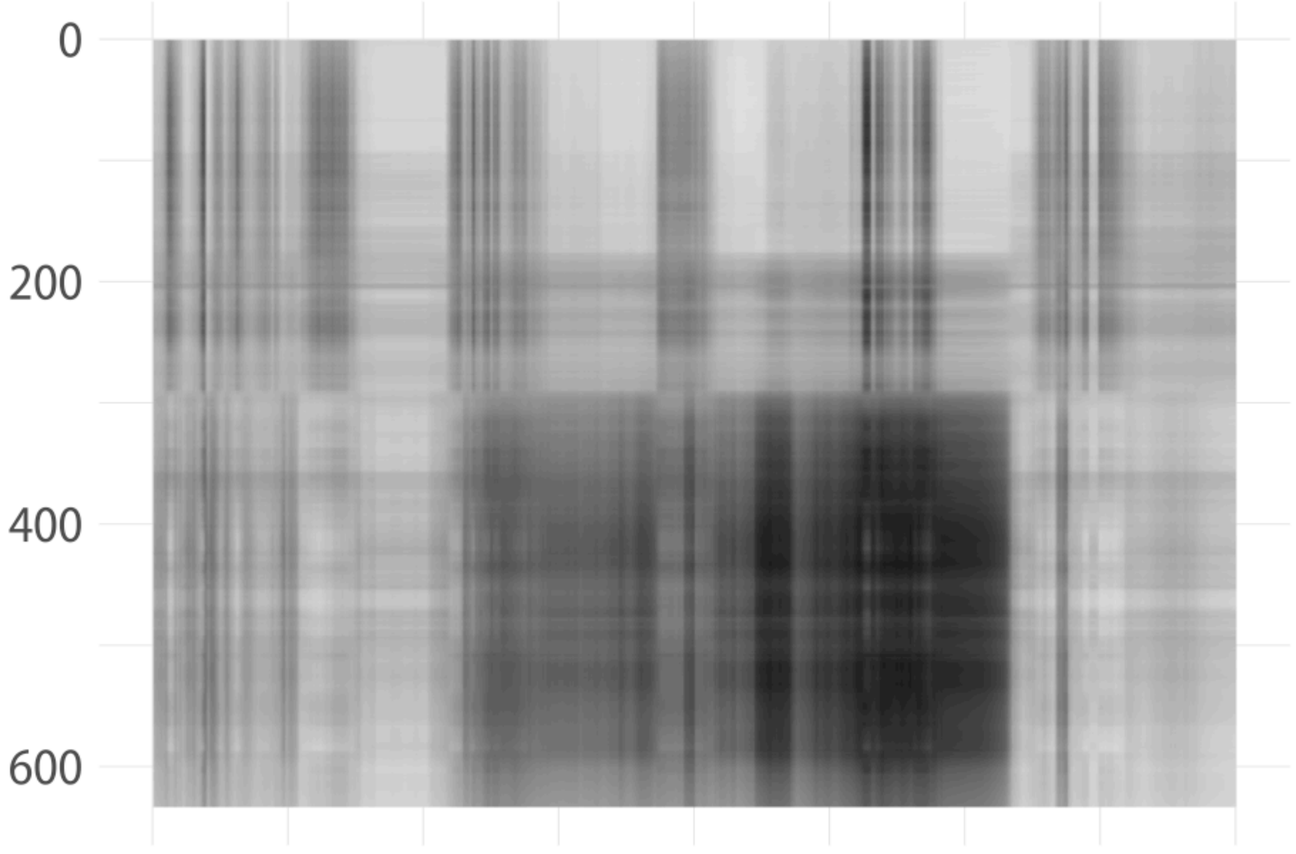
```
p_out
```

Recovering the content of an 800x600 pixel image from a Principal Components Analysis of its pixels



Recovering the content of an 800x600 pixel image from a Principal Components Analysis of its pixels

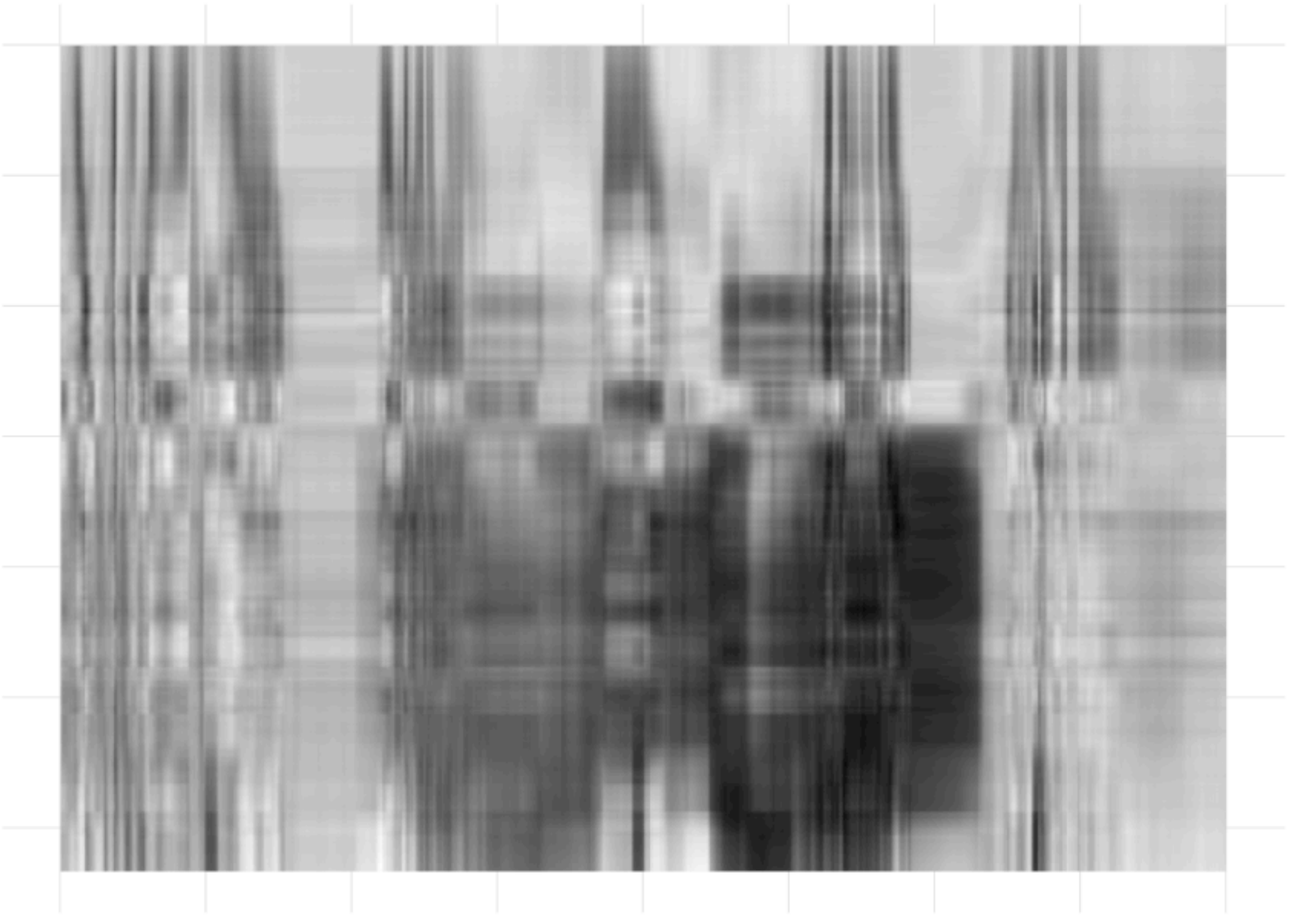
First 2 Components

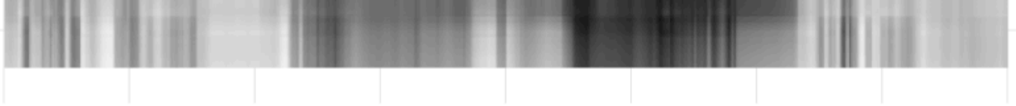


First 3 Components

onents

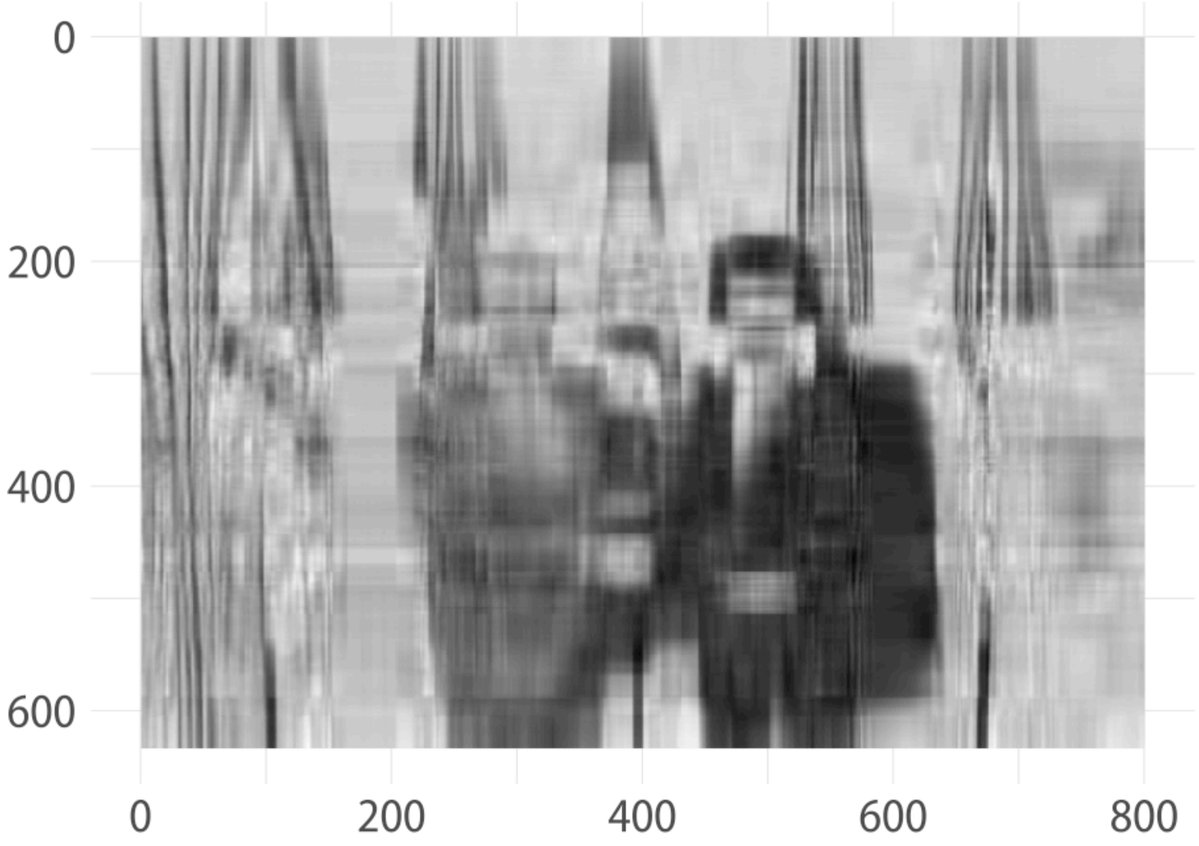
First 5 Components





y

First 10 Components



First 20 Components

First 50 Components

First 100 Components

X

Plot Marginal Effects

```r
library(margins)


gss_sm$polviews_m <- relevel(gss_sm$polviews,
                             ref = "Moderate")

out_bo <- glm(obama ~ polviews_m + sex*race,
              family = "binomial", data = gss_sm)

summary(out_bo)
```

```
##
## Call:
## glm(formula = obama ~ polviews_m + sex * race, family = "binomial",
##     data = gss_sm)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.905  -0.554   0.177   0.542   2.244
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      0.29649    0.13409    2.21   0.0270 *
## polviews_mExtremely Liberal      2.37295    0.52504    4.52  6.2e-06 ***
## polviews_mLiberal                2.60003    0.35667    7.29  3.1e-13 ***
## polviews_mSlightly Liberal       1.29317    0.24843    5.21  1.9e-07 ***
## polviews_mSlightly Conservative -1.35528    0.18129   -7.48  7.7e-14 ***
## polviews_mConservative          -2.34746    0.20038  -11.71  < 2e-16 ***
## polviews_mExtremely Conservative -2.72738    0.38721   -7.04  1.9e-12 ***
## sexFemale                        0.25487    0.14537    1.75   0.0796 .
## raceBlack                        3.84953    0.50132    7.68  1.6e-14 ***
## raceOther                       -0.00214    0.43576    0.00   0.9961
## sexFemale:raceBlack             -0.19751    0.66007   -0.30   0.7648
## sexFemale:raceOther              1.57483    0.58766    2.68   0.0074 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2247.9  on 1697  degrees of freedom
## Residual deviance: 1345.9  on 1686  degrees of freedom
##   (1169 observations deleted due to missingness)
## AIC: 1370
##
## Number of Fisher Scoring iterations: 6
```

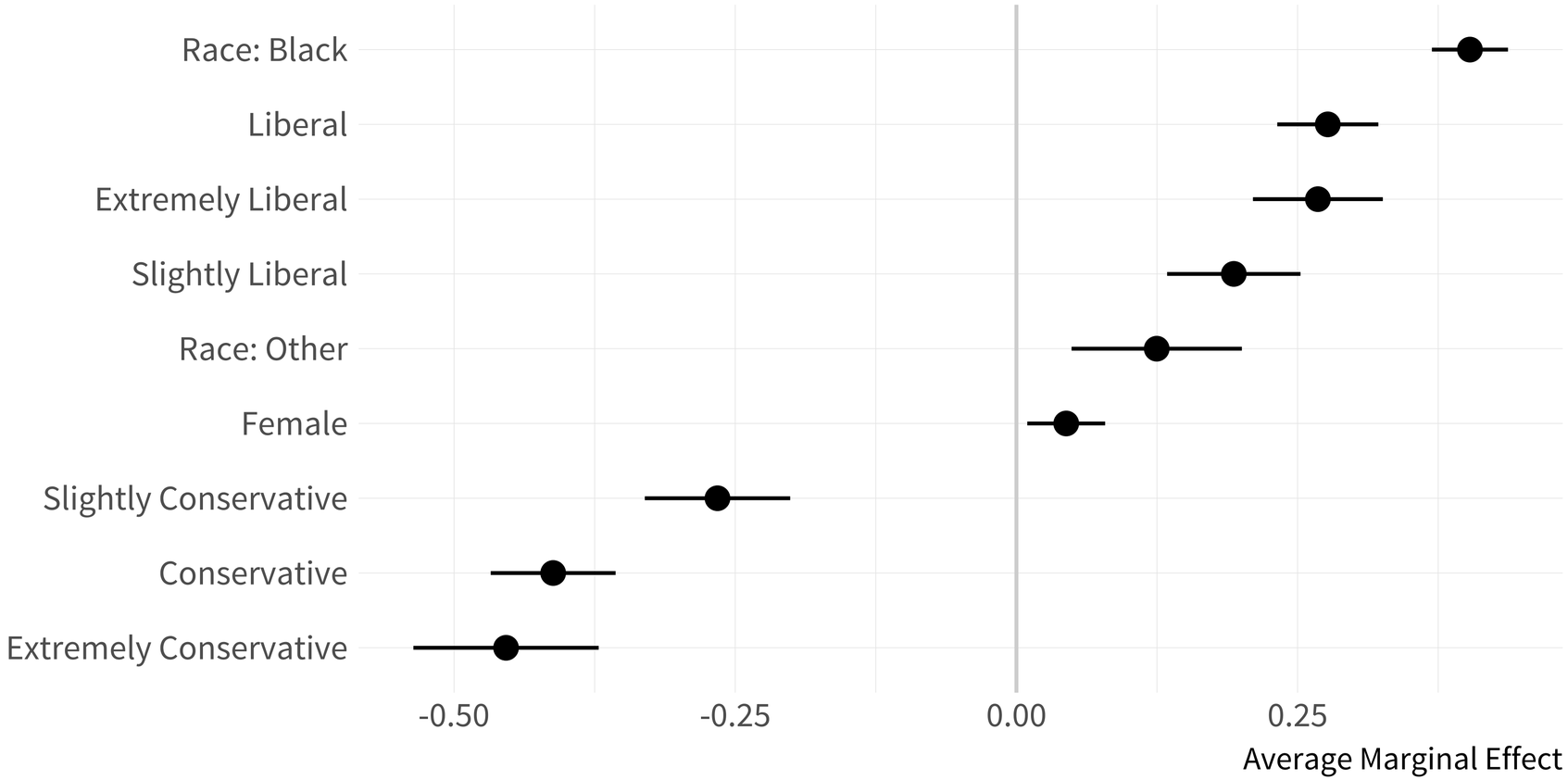
```
bo_m <- margins(out_bo)
summary(bo_m)
```

```
##                                factor     AME     SE         z      p   lower   upper
##            polviews_mConservative -0.4119 0.0283 -14.5394 0.0000 -0.4674 -0.3564
##  polviews_mExtremely Conservative -0.4538 0.0420 -10.7971 0.0000 -0.5361 -0.3714
##       polviews_mExtremely Liberal  0.2681 0.0295   9.0996 0.0000  0.2103  0.3258
##                   polviews_mLiberal  0.2768 0.0229  12.0736 0.0000  0.2319  0.3218
##   polviews_mSlightly Conservative -0.2658 0.0330  -8.0596 0.0000 -0.3304 -0.2011
##        polviews_mSlightly Liberal  0.1933 0.0303   6.3896 0.0000  0.1340  0.2526
##                         raceBlack  0.4032 0.0173  23.3568 0.0000  0.3694  0.4371
##                         raceOther  0.1247 0.0386   3.2297 0.0012  0.0490  0.2005
##                         sexFemale  0.0443 0.0177   2.5073 0.0122  0.0097  0.0789
```

```r
bo_gg <- data.frame(summary(bo_m))
prefixes <- c("polviews_m", "sex")
bo_gg$factor <- prefix_strip(bo_gg$factor, prefixes)
bo_gg$factor <- prefix_replace(bo_gg$factor, "race", "Race: ")

bo_gg %>% select(factor, AME, lower, upper)

bo_gg$thresh <- bb
```

```
##                    factor     AME    lower   upper
## 1            Conservative -0.4119 -0.46741 -0.3564
## 2 Extremely Conservative -0.4538 -0.53614 -0.3714
## 3      Extremely Liberal  0.2681  0.21032  0.3258
## 4                Liberal  0.2768  0.23190  0.3218
## 5  Slightly Conservative -0.2658 -0.33042 -0.2011
## 6        Slightly Liberal  0.1933  0.13400  0.2526
## 7             Race: Black  0.4032  0.36939  0.4371
## 8             Race: Other  0.1247  0.04904  0.2005
## 9                  Female  0.0443  0.00967  0.0789
```

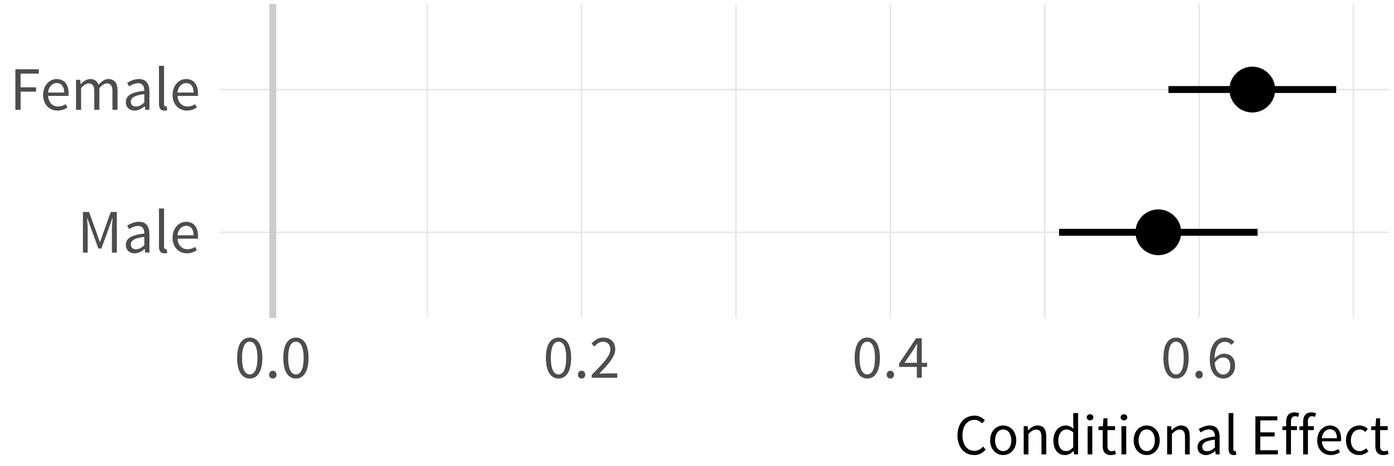
```r
p <- ggplot(data = bo_gg, aes(x = reorder(factor, AME),
                              y = AME, ymin = lower, ymax = upper))

p + geom_hline(yintercept = 0, color = "gray80") +
    geom_pointrange() + coord_flip() +
    labs(x = NULL, y = "Average Marginal Effect")
```



```
pv_cp <- cplot(out_bo, x = "sex", draw = FALSE)
pv_cp
```

```
##    xvals yvals upper lower
## 1   Male 0.574 0.638 0.509
## 2 Female 0.634 0.689 0.580
```

```
p <- ggplot(data = pv_cp, aes(x = reorder(xvals, yvals),
                              y = yvals, ymin = lower, ymax = upper))
p + geom_hline(yintercept = 0, color = "gray80") +
    geom_pointrange() + coord_flip() +
    labs(x = NULL, y = "Conditional Effect")
```



PLOTS FROM COMPLEX SURVEYS

```
library(survey)
library(srvyr)
```

```r
options(survey.lonely.psu = "adjust")
options(na.action="na.pass")

gss_wt <- subset(gss_lon, year > 1974) %>%
    mutate(stratvar = interaction(year, vstrat)) %>%
    as_survey_design(ids = vpsu,
                     strata = stratvar,
                     weights = wtssall,
                     nest = TRUE)
```

```
out_grp
```

```
## # A tibble: 150 x 5
##      year   race          degree   prop prop_se
##     <dbl> <fctr>          <fctr>  <dbl>   <dbl>
##  1  1976  White Lt High School 0.3283 0.01595
##  2  1976  Black Lt High School 0.5620 0.06112
##  3  1976  Other Lt High School 0.2500 0.15167
##  4  1976  White    High School 0.5183 0.01617
##  5  1976  Black    High School 0.3372 0.04759
##  6  1976  Other    High School 0.4167 0.16739
##  7  1976  White Junior College 0.0129 0.00298
##  8  1976  Black Junior College 0.0426 0.01931
##  9  1976  Other Junior College 0.0000 0.00000
## 10  1976  White        Bachelor 0.1012 0.00960
## # ... with 140 more rows
```

```r
out_mrg <- gss_wt %>%
    filter(year %in% seq(1976, 2016, by = 4)) %>%
    mutate(racedeg = interaction(race, degree)) %>%
    group_by(year, racedeg) %>%
    summarize(prop = survey_mean(na.rm = TRUE))

out_mrg
```

```
out_mrg
```

```
## # A tibble: 150 x 4
##     year                racedeg    prop prop_se
##    <dbl>                 <fctr>   <dbl>   <dbl>
##  1  1976 White.Lt High School 0.29823 0.01463
##  2  1976 Black.Lt High School 0.04711 0.00840
##  3  1976 Other.Lt High School 0.00195 0.00138
##  4  1976    White.High School 0.47078 0.01597
##  5  1976    Black.High School 0.02826 0.00594
##  6  1976    Other.High School 0.00325 0.00166
##  7  1976 White.Junior College 0.01170 0.00268
##  8  1976 Black.Junior College 0.00357 0.00162
##  9  1976 Other.Junior College 0.00000 0.00000
## 10  1976       White.Bachelor 0.09194 0.00888
## # ... with 140 more rows
```

```r
out_mrg <- gss_wt %>%
    filter(year %in% seq(1976, 2016, by = 4)) %>%
    mutate(racedeg = interaction(race, degree)) %>%
    group_by(year, racedeg) %>%
    summarize(prop = survey_mean(na.rm = TRUE)) %>%
    separate(racedeg, sep = "\\.", into = c("race", "degree"))

out_mrg
```

```
   out_mrg
```

```
## # A tibble: 150 x 5
##     year  race          degree    prop prop_se
##  * <dbl> <chr>          <chr>    <dbl>   <dbl>
##  1  1976 White Lt High School 0.29823 0.01463
##  2  1976 Black Lt High School 0.04711 0.00840
##  3  1976 Other Lt High School 0.00195 0.00138
##  4  1976 White    High School 0.47078 0.01597
##  5  1976 Black    High School 0.02826 0.00594
##  6  1976 Other    High School 0.00325 0.00166
##  7  1976 White Junior College 0.01170 0.00268
##  8  1976 Black Junior College 0.00357 0.00162
##  9  1976 Other Junior College 0.00000 0.00000
## 10  1976 White       Bachelor 0.09194 0.00888
## # ... with 140 more rows
```

```
p <- ggplot(data = subset(out_grp, race %nin% "Other"),
            mapping = aes(x = degree, y = prop,
                          ymin = prop - 2*prop_se,
                          ymax = prop + 2*prop_se,
                          fill = race,
                          color = race,
                          group = race))

dodge <- position_dodge(width=0.9)
```

```r
p + geom_col(position = dodge, alpha = 0.2) +
    geom_errorbar(position = dodge, width = 0.2) +
    scale_x_discrete(labels = scales::wrap_format(10)) +
    scale_y_continuous(labels = scales::percent) +
    scale_color_brewer(type = "qual", palette = "Dark2") +
    scale_fill_brewer(type = "qual", palette = "Dark2") +
    labs(title = "Educational Attainment by Race",
         subtitle = "GSS 1976-2016",
         fill = "Race",
         color = "Race",
         x = NULL, y = "Percent") +
    facet_wrap(~ year, ncol = 2) +
    theme(legend.position = "top")
```

Again, while ggplot puts this graph within your reach, think carefully before making it! This sort of multiple-comparison bar chart is often ineffective.

```r
p <- ggplot(data = subset(out_grp, race %nin% "Other"),
            mapping = aes(x = year, y = prop,
                          ymin = prop - 2*prop_se,
                          ymax = prop + 2*prop_se,
                          fill = race, color = race,
                          group = race))

p + geom_ribbon(alpha = 0.3, aes(color = NULL)) +
    geom_line() +
    facet_wrap(~ degree, ncol = 3) +
    scale_y_continuous(labels = scales::percent) +
    scale_color_brewer(type = "qual", palette = "Dark2") +
    scale_fill_brewer(type = "qual", palette = "Dark2") +
    labs(title = "Educational Attainment by Race",
         subtitle = "GSS 1976-2016", fill = "Race",
         color = "Race", x = NULL, y = "Percent") +
    theme(legend.position = "top")
```

Educational Attainment by Race

GSS 1976-2016

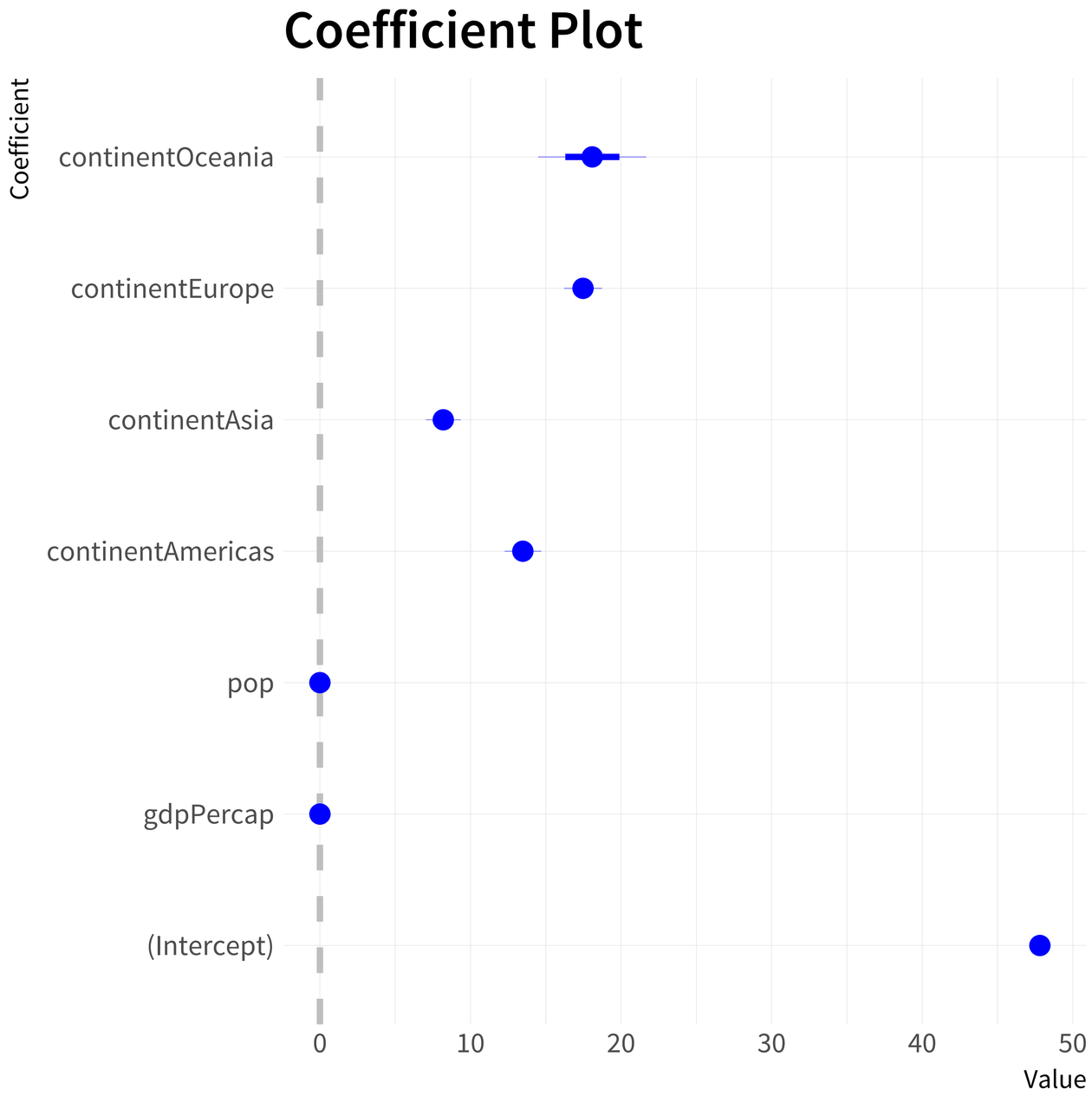
ALTERNATIVES FOR WORKING WITH MODELS

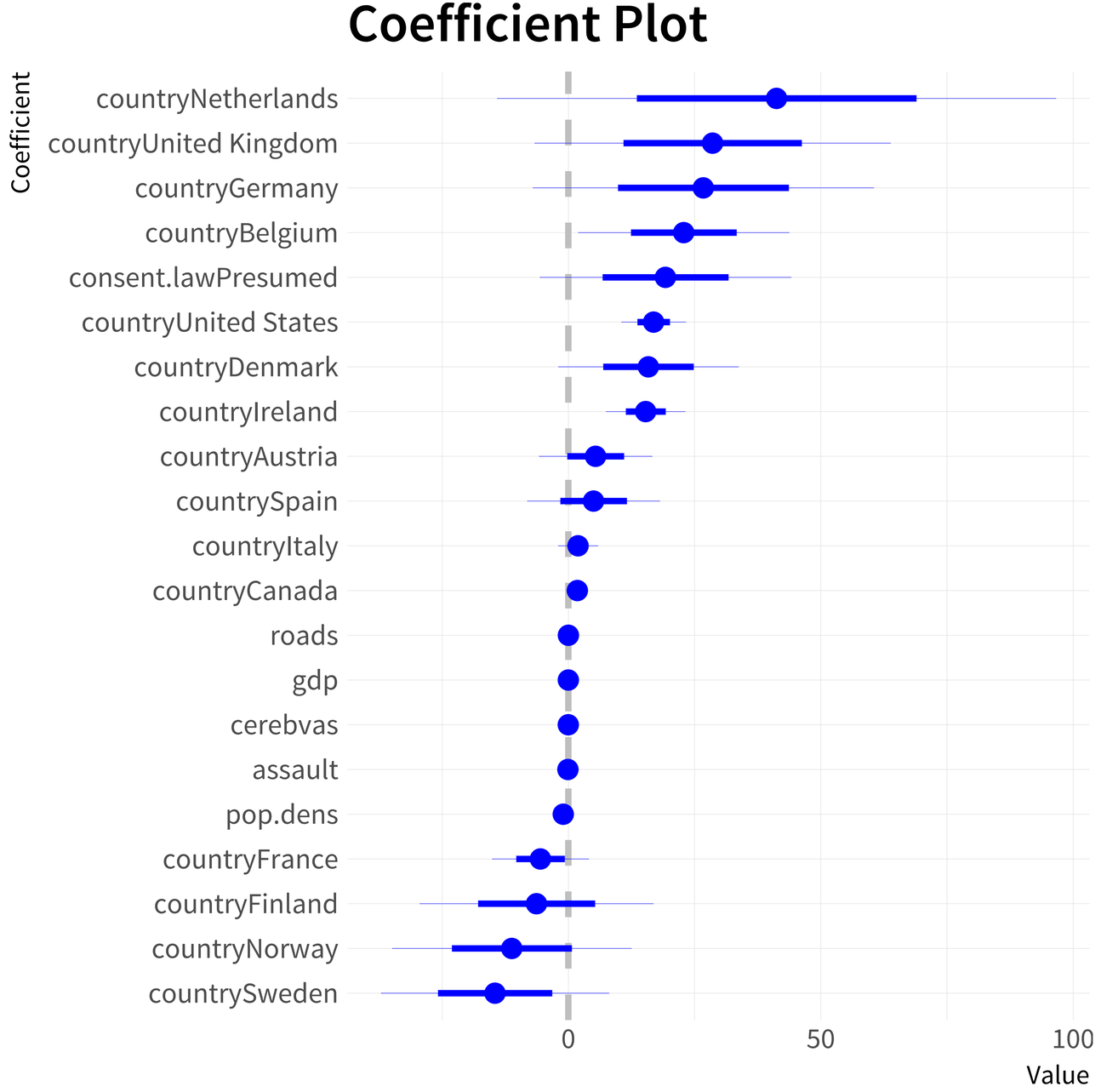
```r
out <- lm(formula = lifeExp ~ gdpPercap + pop + continent,
          data = gapminder)

plot(out, ask=FALSE)
```

Default plot() methods

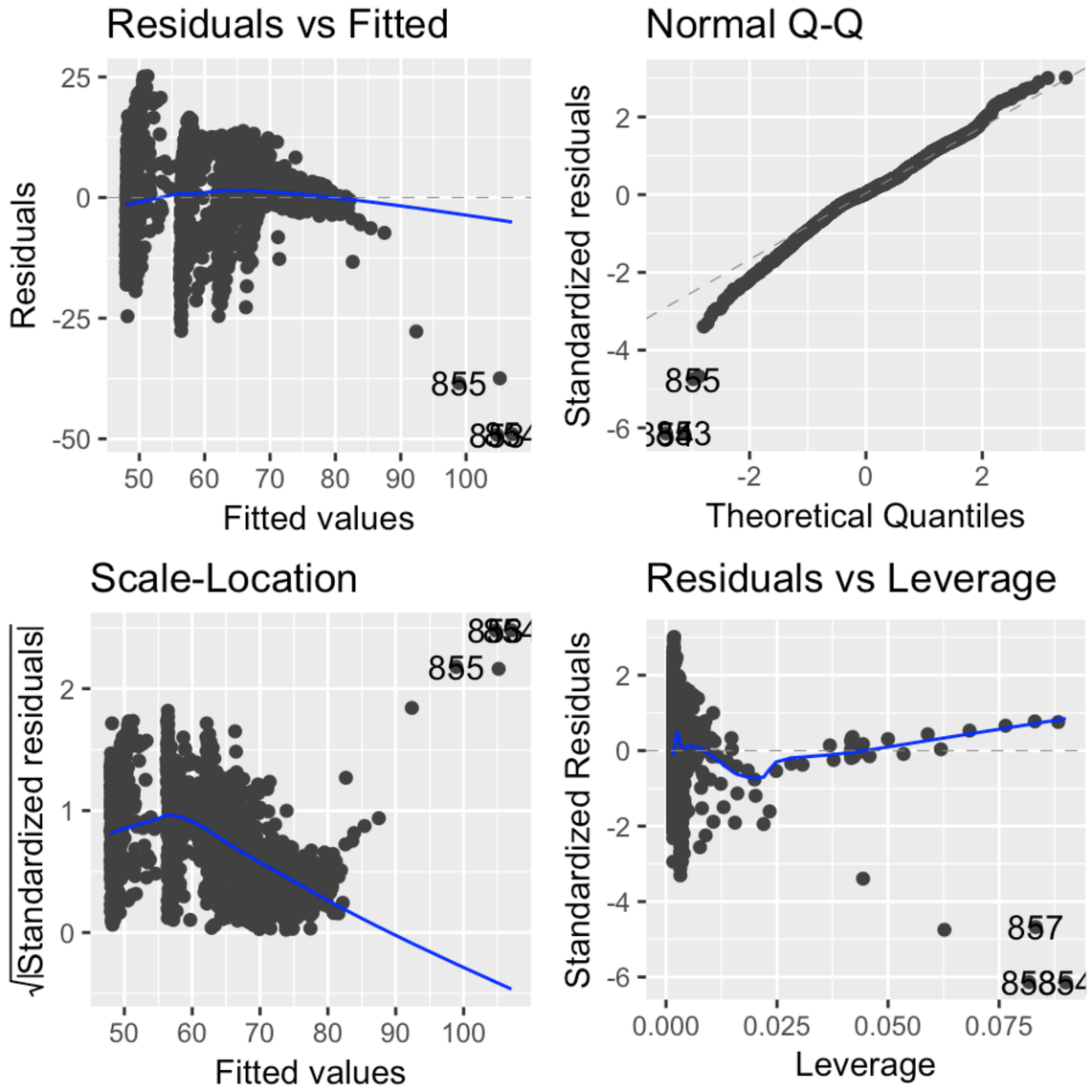
```r
library(coefplot)

out <- lm(formula = lifeExp ~ gdpPercap + pop + continent,
          data = gapminder)

out2 <- lm(formula = donors ~  gdp + roads + cerebvas + assault +
                       pop.dens + consent.law + country,
           data = organdata)

coefplot(out)
coefplot(out2, sort = "magnitude", intercept = FALSE)
```

The coefplot() library


Coefficient Plot
Coefficient
continentOceania
continentEurope
continentAsia
continentAmericas
pop
gdpPercap
(Intercept)
0
10
20
30
40
50
Value


Coefficient Plot
Coefficient
countryNetherlands
countryUnited Kingdom
countryGermany
countryBelgium
consent.lawPresumed
countryUnited States
countryDenmark
countryIreland
countryAustria
countrySpain
countryItaly
countryCanada
roads
gdp
cerebvas
assault
pop.dens
countryFrance
countryFinland
countryNorway
countrySweden
0
50
100
Value

```r
library(ggfortify)
autoplot(out)
```



The ggfortify() library