CSSS 569 Visualizing Data and Models

Lab 4: Advanced `ggplot2`

Brian Leung

Department of Political Science, UW

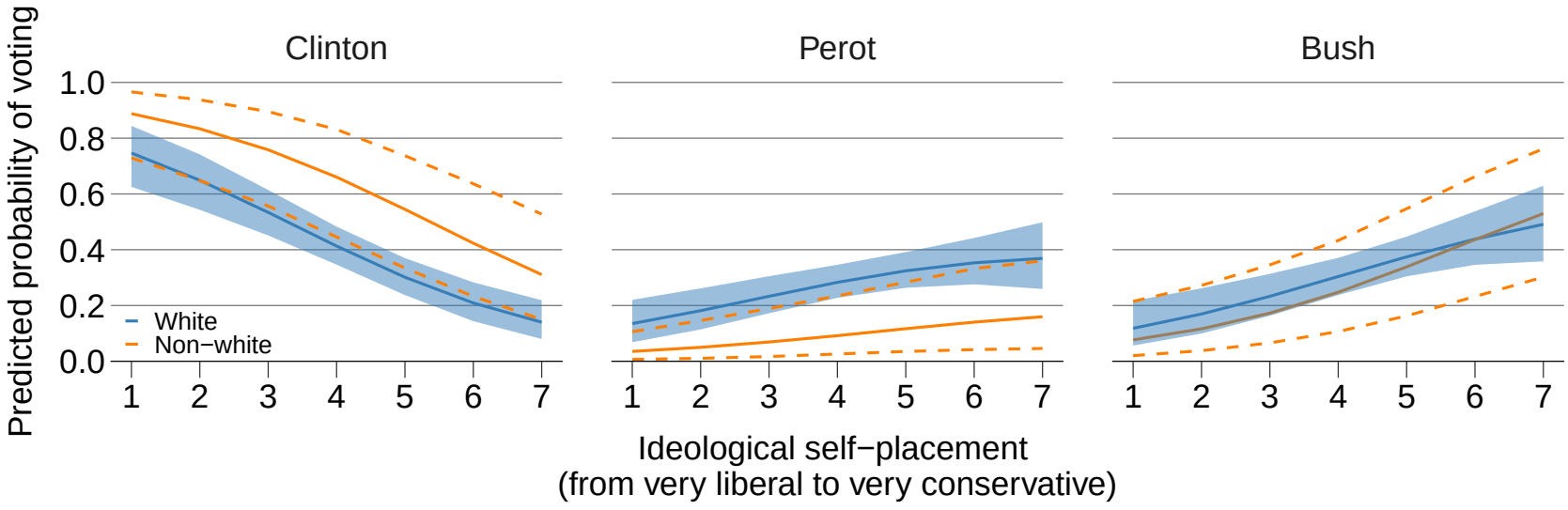
February 9, 2021

Introduction

- Recap of what we've covered last week
- Making a scatterplot from scratch in `ggplot2` (from Chris's slides)
  1. Decide on dimensions: aspect ratio, axis limits
  2. Add axis labels, plot titles
  3. Choose data markers: points, symbols, text
  4. Scaling & transformation, add ticks if needed
  5. Choose a color palette
  6. Add annotations: labels, arrows, notes
  7. Add best-fit line(s) & confidence intervals
  8. Add extra plots (e.g., rugs) to make a confection
  9. Repeat as small multiples (`facet_grid` and `facet_wrap`)
  - Next week we'll implement them using `tile`
- Unpack the inner working of `ggplot2`
  - data, aes(...), geom(..., inherit.aes = TRUE)
- Customized theme: theme_cavis.R
- Exercise to reproduce a graph

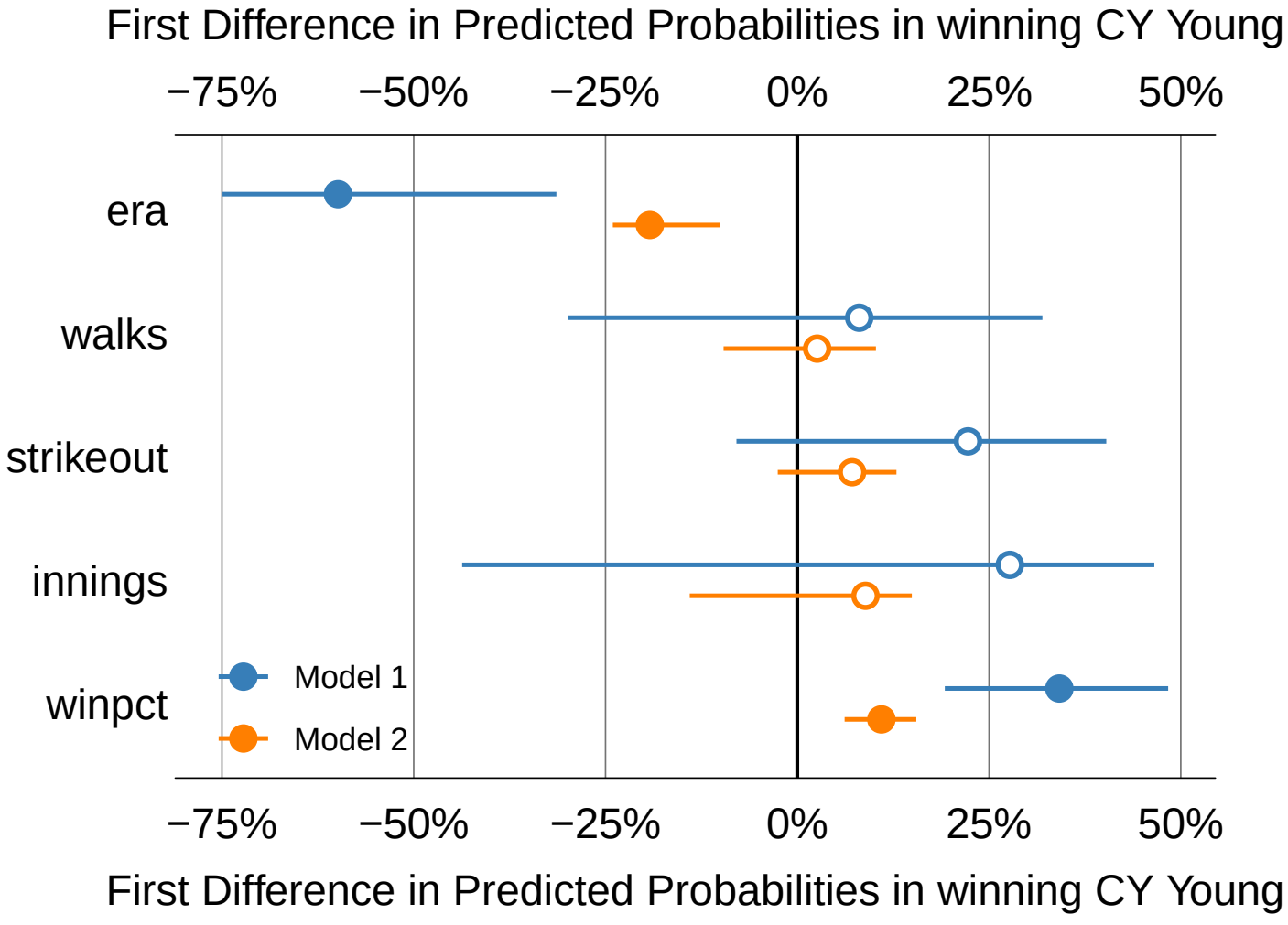
Roadmap for today

Today's lab is structured around three exercises:



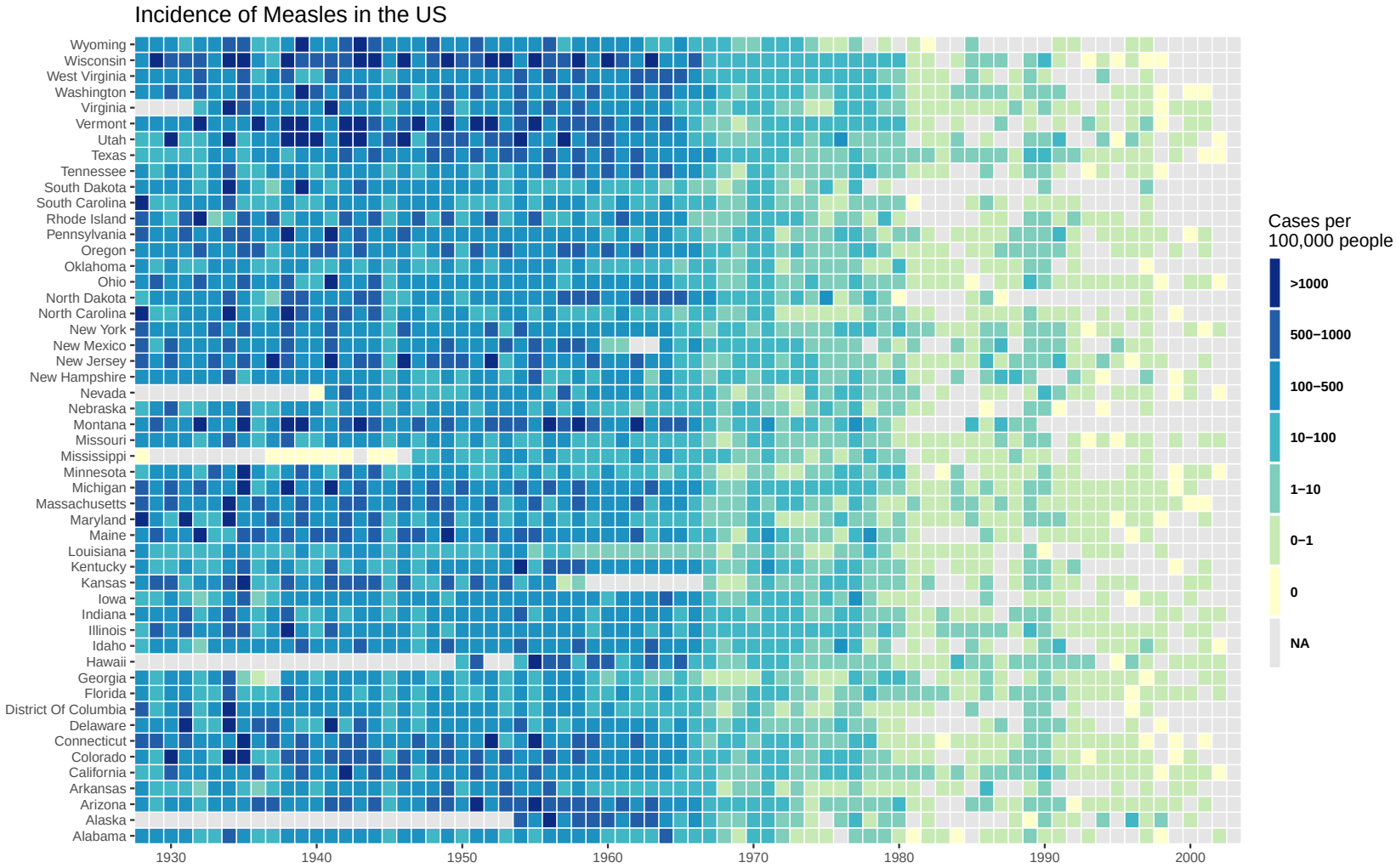
Roadmap for today

Today's lab is structured around three exercises:



Roadmap for today

Today's lab is structured around three exercises:



Roadmap for today

1. Last exercise: 1992 Presidential Election

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels
2. Ropeladder exercise: Cy Young award

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels
2. Ropeladder exercise: Cy Young award
   - `pivot_longer` and `pivot_wider`

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels
2. Ropeladder exercise: Cy Young award
   - `pivot_longer` and `pivot_wider`
   - Sorting using `fct_reorder`

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels
2. Ropeladder exercise: Cy Young award
   - `pivot_longer` and `pivot_wider`
   - Sorting using `fct_reorder`
   - Use of `scale_{...}`

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels
2. Ropeladder exercise: Cy Young award
   - `pivot_longer` and `pivot_wider`
   - Sorting using `fct_reorder`
   - Use of `scale_{...}`
3. Heatmap exercise: Measles in US

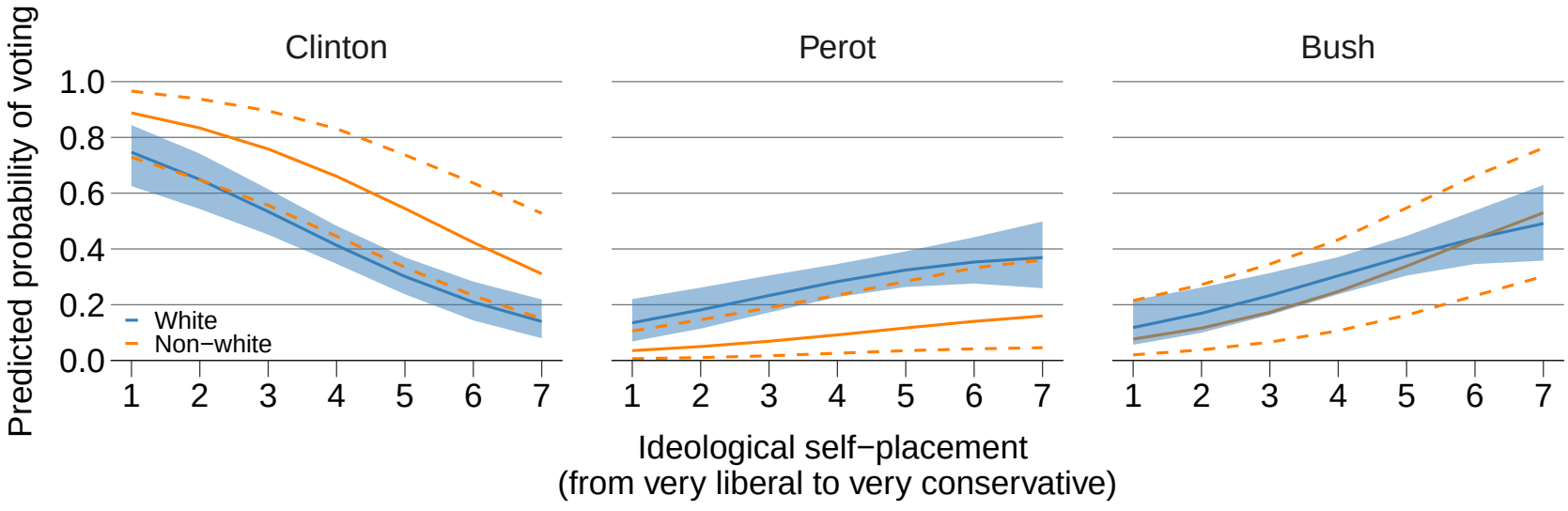
Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels
2. Ropeladder exercise: Cy Young award
   - `pivot_longer` and `pivot_wider`
   - Sorting using `fct_reorder`
   - Use of `scale_{...}`
3. Heatmap exercise: Measles in US
   - Use of `geom_tile` and various ways to `scale_color/fill_{...}`

Roadmap for today

1. Last exercise: 1992 Presidential Election
   - Use of `scale_{...}`
   - Use of `facet_grid` and facet-specific labels
2. Ropeladder exercise: Cy Young award
   - `pivot_longer` and `pivot_wider`
   - Sorting using `fct_reorder`
   - Use of `scale_{...}`
3. Heatmap exercise: Measles in US
   - Use of `geom_tile` and various ways to `scale_color/fill_{...}`
4. Highlight `ggplot2` extension packages (See more here)

Last exercise: 1992 Presidential Election

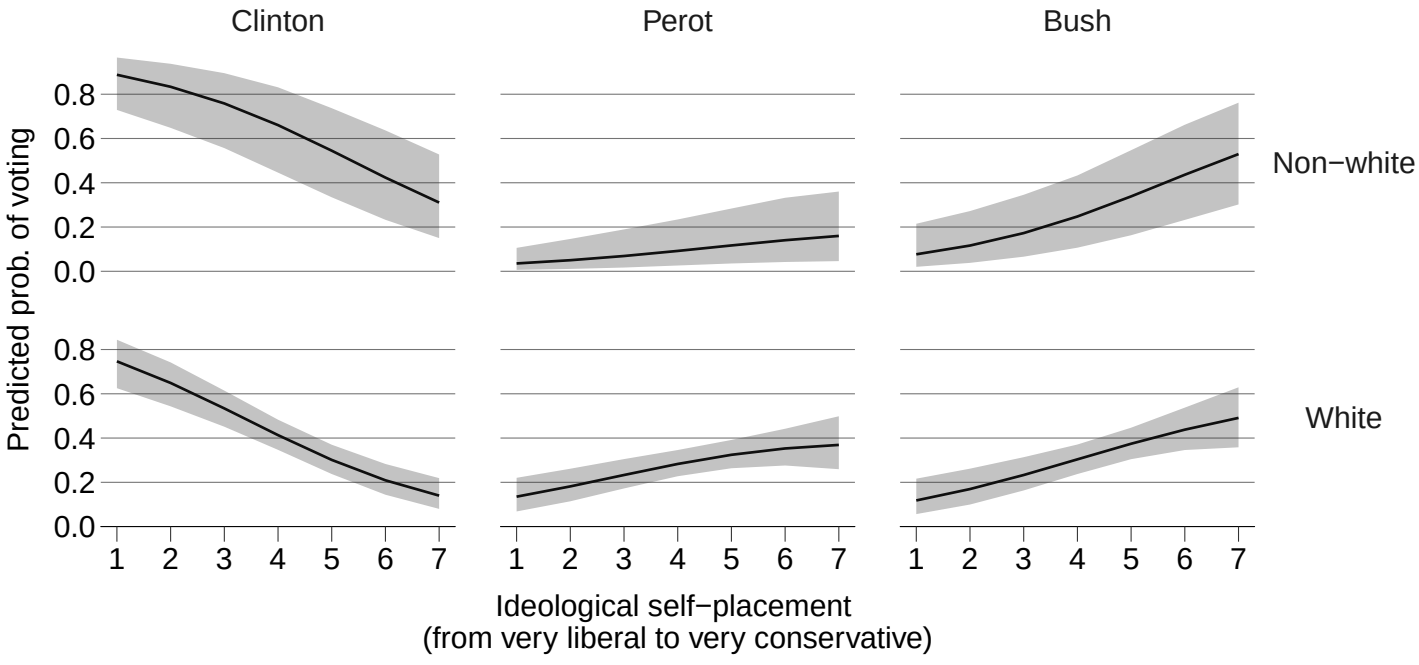


Last exercise: motivation

- There are many ways to do small multiples:

Last exercise: motivation

- There are many ways to do small multiples:
  - `plot + facet_grid(nonwhite ~ vote92)`

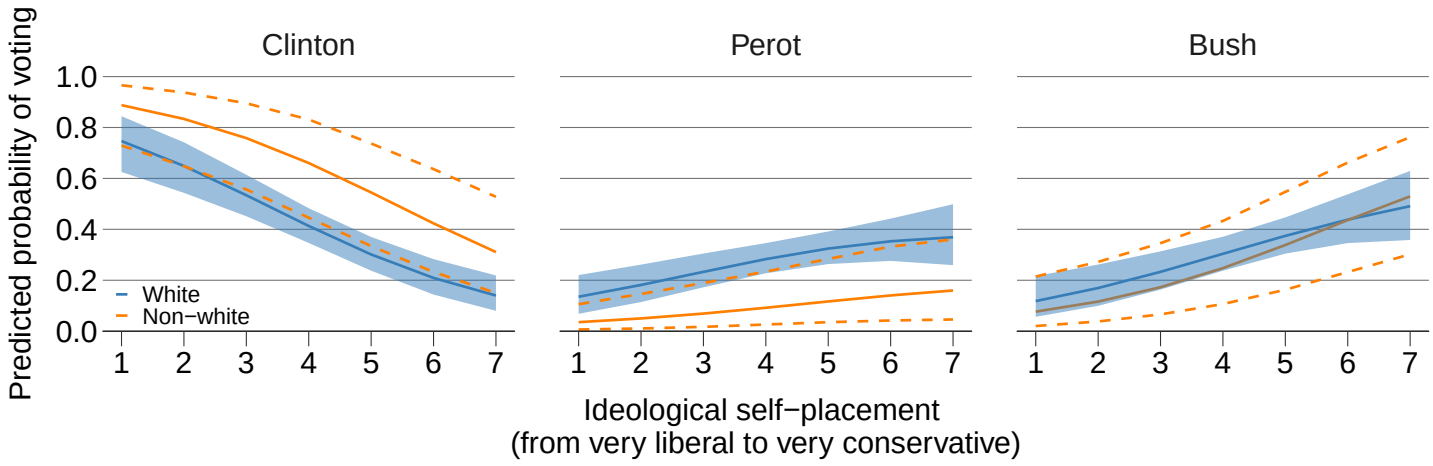


Last exercise: motivation

- Thoughtful juxtaposition facilitates meaningful comparison and provokes further inquiry

Last exercise: motivation

- Thoughtful juxtaposition facilitates meaningful comparison and provokes further inquiry
  - Sometimes, data overlapping might be an interesting phenomenon



Last exercise: 1992 Presidential Election

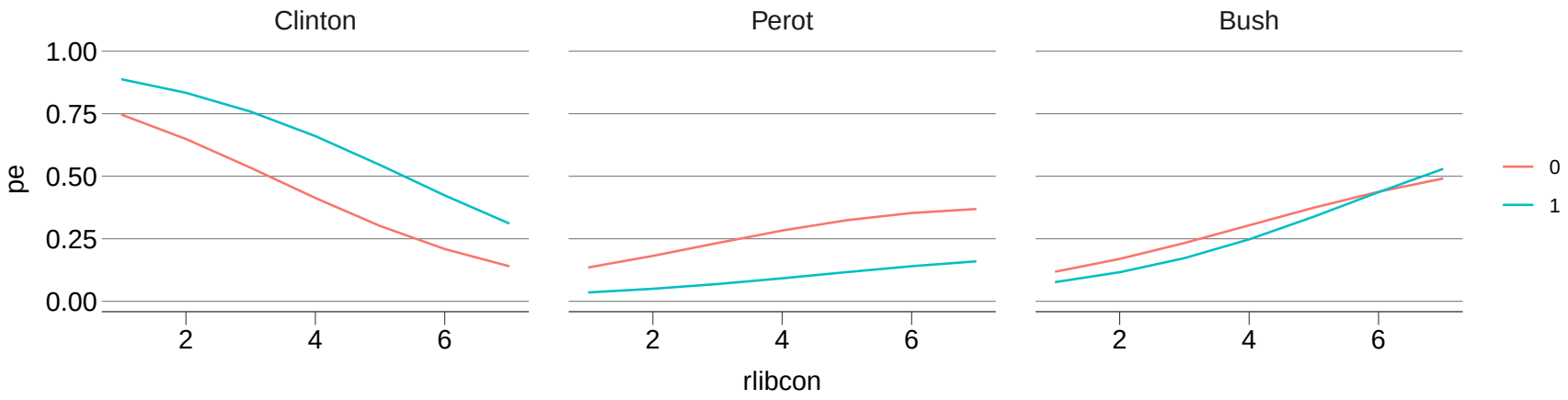
```r
### Prerequisite
# Load package
library(tidyverse)
library(RColorBrewer)

# Load data
presVoteEV <- read_csv("data/presVoteEV.csv")

# Load theme
source("theme/theme_cavis.R")

# Get nice color
brewer <- brewer.pal(9, "Set1")
blue <- brewer[2]
orange <- brewer[5]
```

Last exercise: 1992 Presidential Election

```r
# Factorize variables
presVoteEV <- presVoteEV %>%
  mutate(
    nonwhite = factor(nonwhite),
    vote92 = factor(vote92, levels = c("Clinton", "Perot", "Bush"))
  )
```

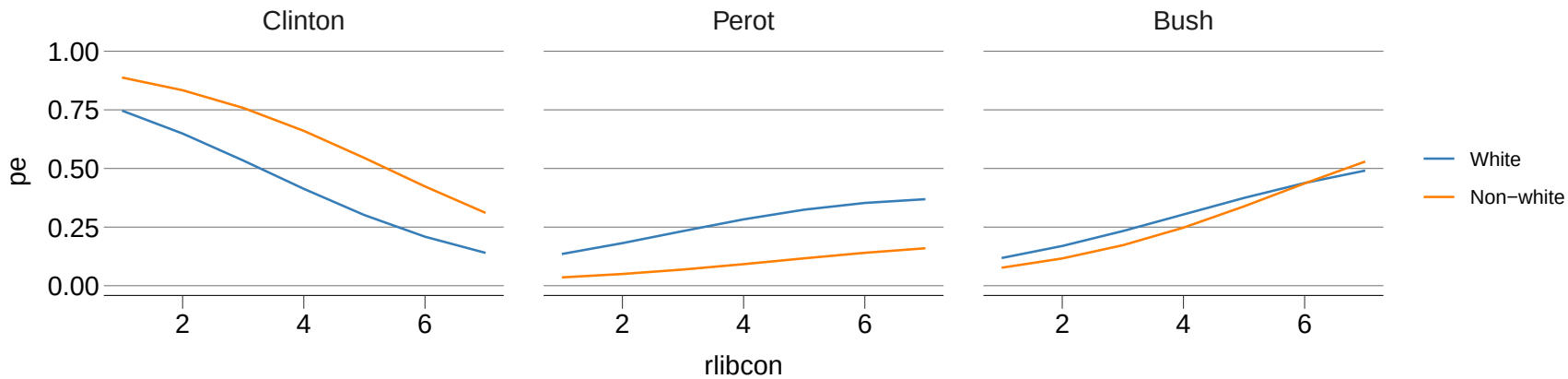
Last exercise: 1992 Presidential Election

```
p <-
  ggplot(presVoteEV, aes(x = rlibcon,
                         y = pe, ymin = lower, ymax = upper,
                         color = nonwhite, fill = nonwhite)) +
  facet_grid(~ vote92) +
  geom_line() +
  theme_cavis_hgrid

print(p)
```



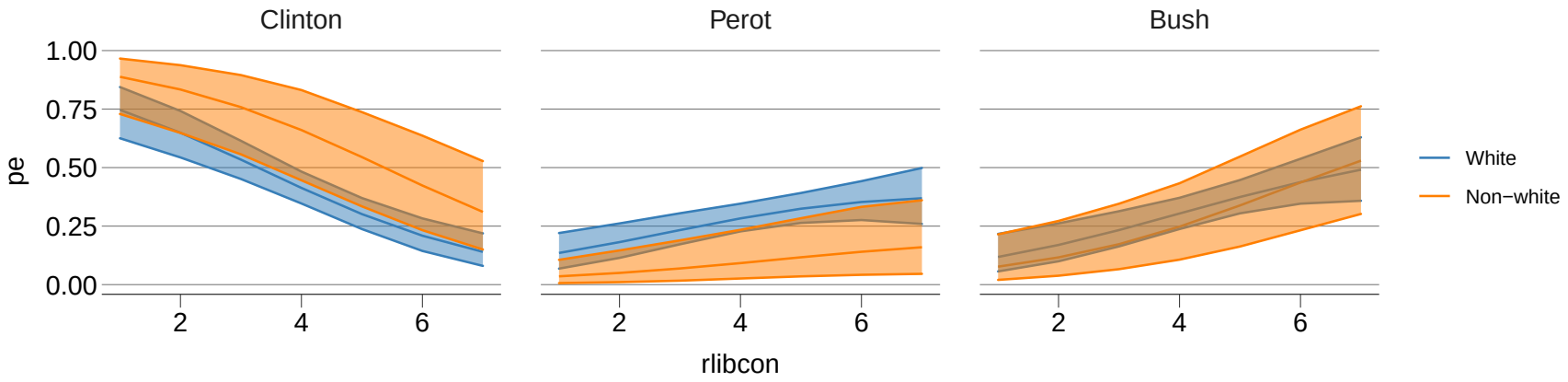
Last exercise: 1992 Presidential Election

```
p <- p +
  scale_color_manual(values = c(blue, orange),
                     labels = c("White", "Non-white"))

print(p)
```



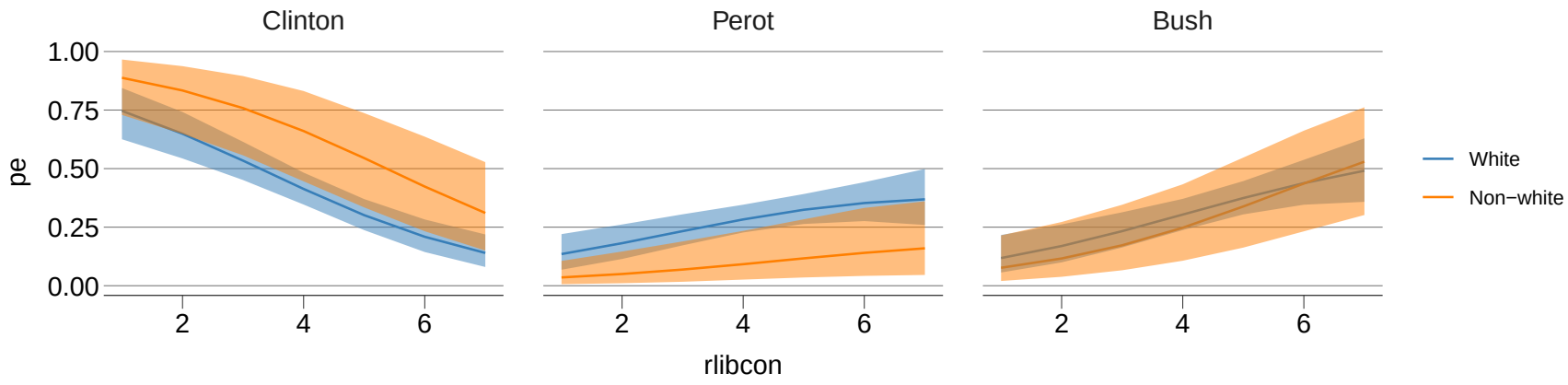
Last exercise: 1992 Presidential Election

```
p +
  geom_ribbon(alpha = 0.5, show.legend = FALSE) +
  scale_fill_manual(values = c(blue, orange))
```



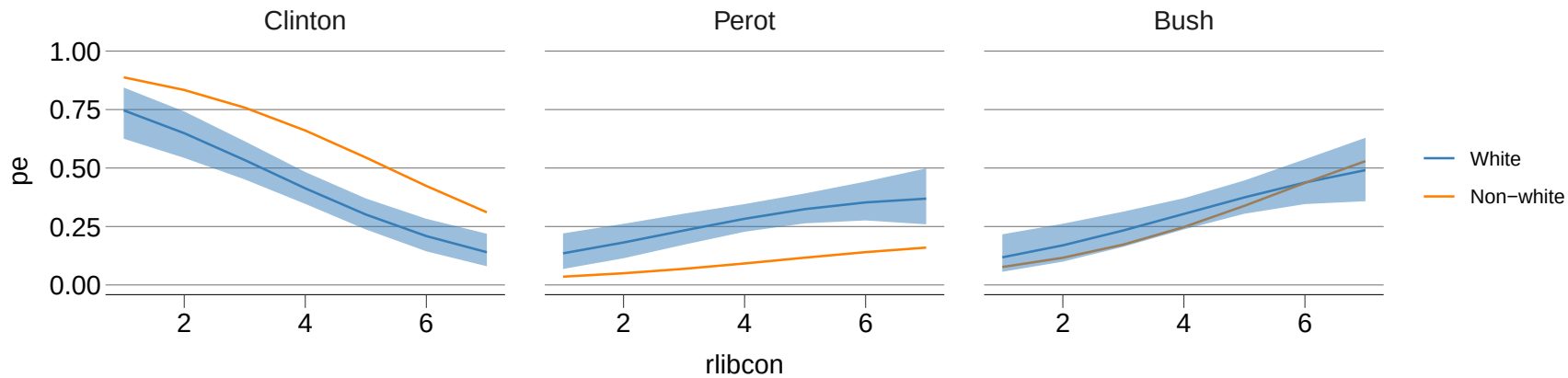
Last exercise: 1992 Presidential Election

```
p +
  geom_ribbon(alpha = 0.5, linetype = 0, show.legend = FALSE) +
  scale_fill_manual(values = c(blue, orange))
```



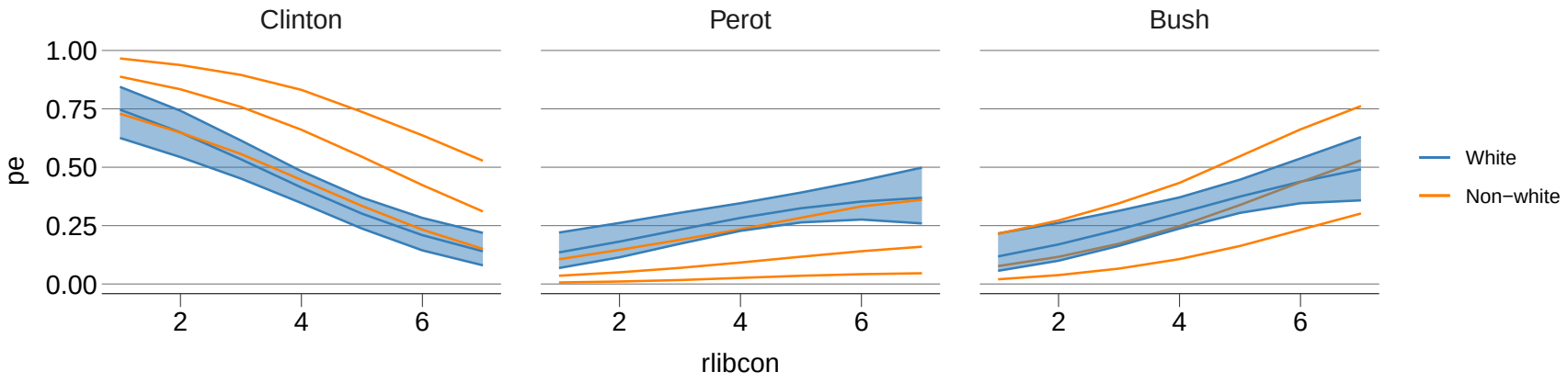
Last exercise: 1992 Presidential Election

```
p <- p +
  geom_ribbon(alpha = 0.5, linetype = 0, show.legend = FALSE) +
  scale_fill_manual(values = c(blue, NA))

print(p)
```



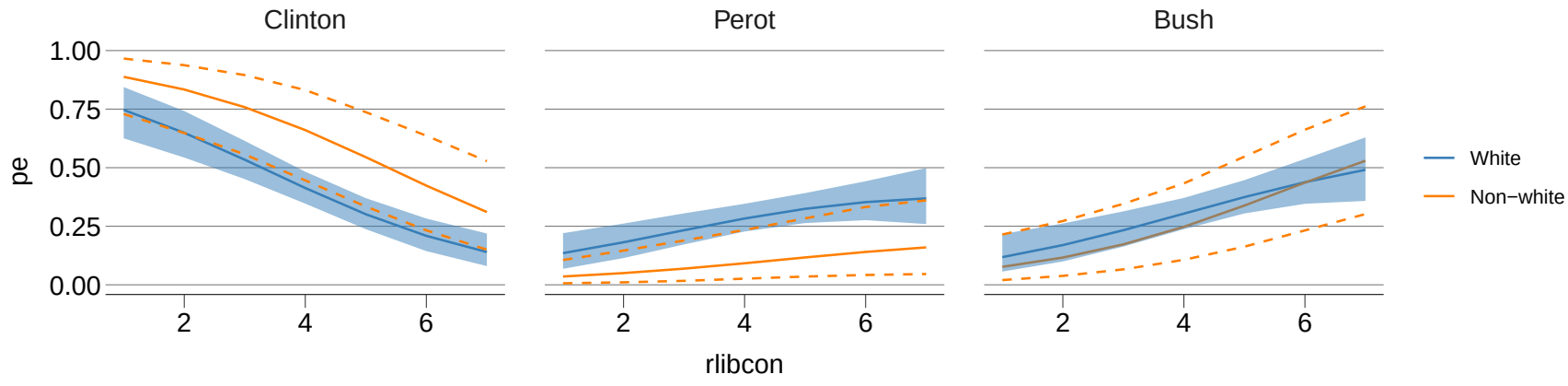
Last exercise: 1992 Presidential Election

```
p +
  geom_line(aes(y = upper)) +
  geom_line(aes(y = lower))
```



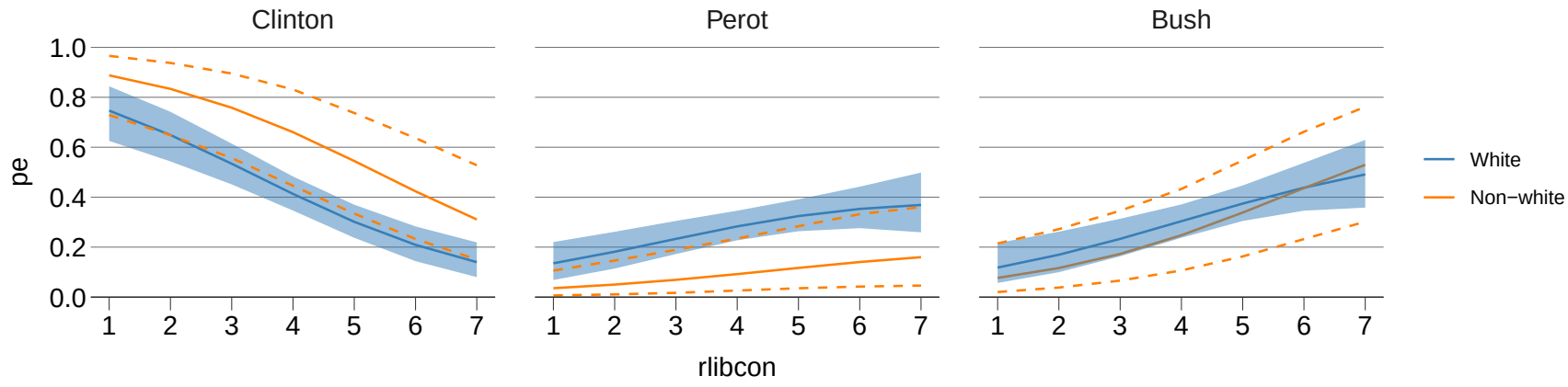
Last exercise: 1992 Presidential Election

```
p <- p +
  geom_line(aes(y = upper, linetype = nonwhite), show.legend = FALSE) +
  geom_line(aes(y = lower, linetype = nonwhite), show.legend = FALSE) +
  scale_linetype_manual(values = c(0, 2)) # 0 = blank; 2 = dashed

print(p)
```



Last exercise: 1992 Presidential Election

```
p <- p +
  scale_x_continuous(breaks = 1:7) +
  scale_y_continuous(breaks = seq(0, 1, 0.2),
                     limits = c(0, 1),
                     expand = c(0, 0))

print(p)
```



Last exercise: 1992 Presidential Election

```
p <- p +
  theme(legend.position = c(0.06, 0.13),
        legend.key.size = unit(0.2, "cm")) +
  labs(y = "Predicted prob. of voting",
       x = "Ideological self-placement")

print(p)
```

Last exercise: 1992 Presidential Election

Full code to reproduce the graph

```
p <- ggplot(presVoteEV, aes(x = rlibcon, y = pe,
                            color = nonwhite, fill = nonwhite)) +
  # Small multiples by candidates
  facet_grid(~ vote92) +
  # Point estimates lines
  geom_line() +
  scale_color_manual(values = c(blue, orange),
                     labels = c("White", "Non-white")) +
  # CIs for white voters
  geom_ribbon(aes(ymin = lower, ymax = upper),
              alpha = 0.5, linetype = 0, show.legend = FALSE) +
  scale_fill_manual(values = c(blue, NA)) +
  # CIs for non-white voters
  geom_line(aes(y = upper, linetype = nonwhite), show.legend = FALSE) +
  geom_line(aes(y = lower, linetype = nonwhite), show.legend = FALSE) +
  scale_linetype_manual(values = c(0, 2)) +
  # Other adjustments
  scale_x_continuous(breaks = 1:7) +
  scale_y_continuous(breaks = seq(0, 1, 0.2), limits = c(0, 1),
                     expand = c(0, 0)) +
  theme_cavis_hgrid +
  theme(legend.position = c(0.06, 0.13), legend.key.size = unit(0.2, "cm")) +
  labs(y = "Predicted prob. of voting", x = "Ideological self-placement")
```

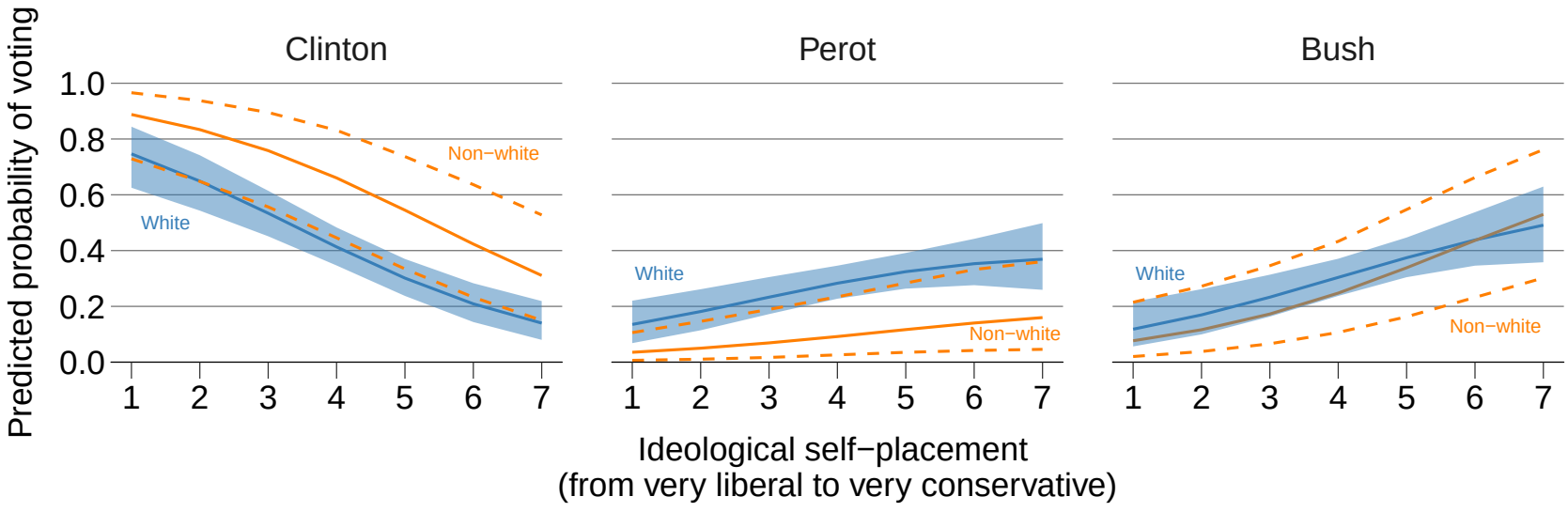
Last exercise: 1992 Presidential Election

Alternative way: create subsets of data and localize data source for each geom layer

```
whiteData <- filter(presVoteEV, nonwhite == 0)
nonwhiteData<- filter(presVoteEV, nonwhite == 1)

ggplot(mapping = aes(x = rlibcon, y = pe, ymin = lower, ymax = upper)) +
  # Small multiples by candidates
  facet_grid(~ vote92) +
  # For white voters
  geom_line(data = whiteData, aes(colour = blue)) +
  geom_ribbon(data = whiteData, fill = blue, linetype = 0, alpha = 0.5) +
  # For non-white voters
  geom_line(data = nonwhiteData, aes(colour = orange)) +
  geom_line(data = nonwhiteData, aes(y = lower), colour = orange, linetype = 2)
  geom_line(data = nonwhiteData, aes(y = upper), colour = orange, linetype = 2)
  # To create a legend that recognizes color strings
  scale_color_identity(labels = c("White", "Non-white"), guide = "legend") +
  # Other adjustments
  scale_x_continuous(breaks = 1:7) +
  scale_y_continuous(breaks = seq(0, 1, 0.2), limits = c(0, 1),
                     expand = c(0, 0)) +
  theme_cavis_hgrid +
  theme(legend.position = c(0.07, 0.13), legend.key.size = unit(0.2, "cm")) +
  labs(y = "Predicted prob. of voting", x = "Ideological self-placement")
```

Last exercise: 1992 Presidential Election

Bonus: You can also create facet-specific labels instead of a generic legend



Last exercise: 1992 Presidential Election

To create facet-specific labels, you need a separate dataframe with the x and y coordinates, labels, and the variable used in `facet_wrap` for identification

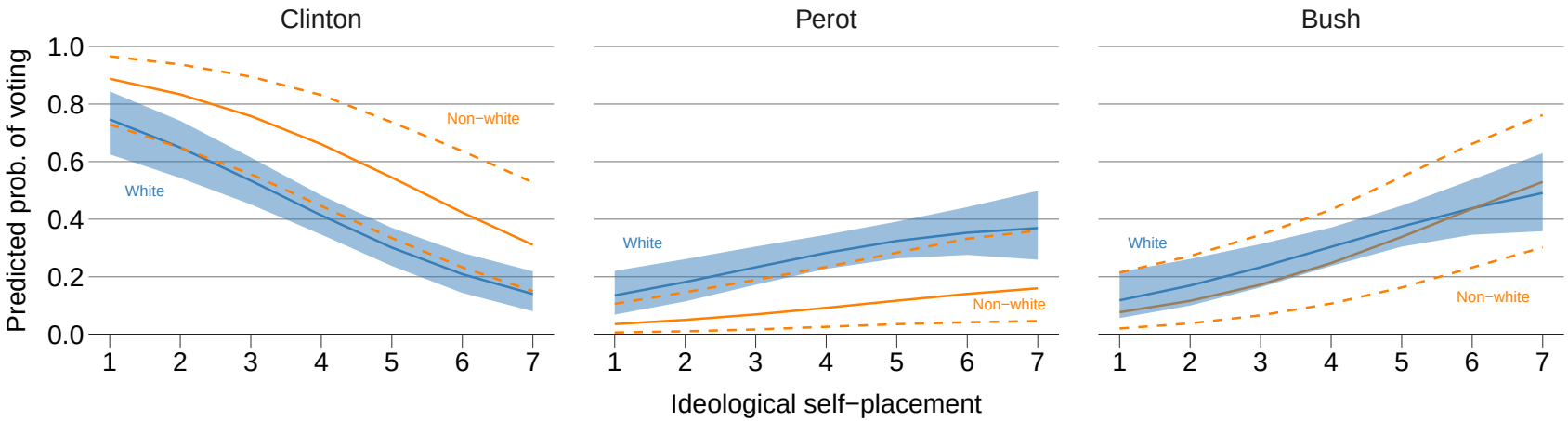
```
candidates <- c("Clinton", "Perot", "Bush")

facet_labels <-
  tibble(vote92 = rep(candidates, each = 2),
         nonwhite = rep(c(0, 1), 3),
         label = rep(c("White", "Non-white"), 3),
         x_coord = c(1.5, 6.3, 1.4, 6.6, 1.4, 6.3),
         y_coord = c(0.5, 0.75, 0.32, 0.105, 0.32, 0.13)) %>%
  mutate(vote92 = factor(vote92, levels = candidates),
         nonwhite = factor(nonwhite))

print(facet_labels)
```

```
## # A tibble: 6 x 5
##   vote92  nonwhite label     x_coord y_coord
##   <fct>   <fct>    <chr>       <dbl>   <dbl>
## 1 Clinton 0        White         1.5   0.5
## 2 Clinton 1        Non-white     6.3   0.75
## 3 Perot   0        White         1.4   0.32
## 4 Perot   1        Non-white     6.6   0.105
## 5 Bush    0        White         1.4   0.32
## 6 Bush    1        Non-white     6.3   0.13
```

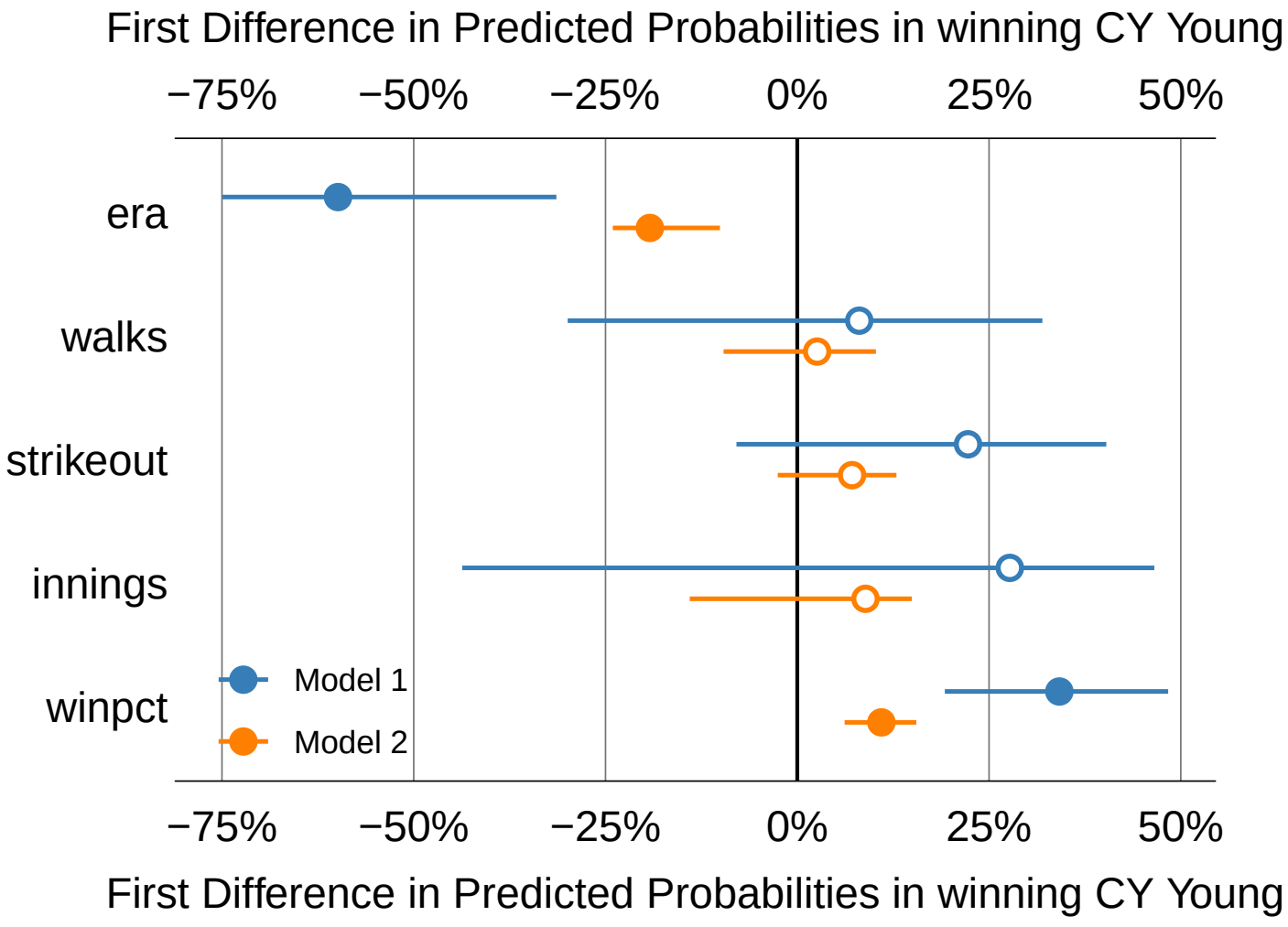
Last exercise: 1992 Presidential Election

To create facet-specific labels, you need a separate dataframe with the x and y coordinates, labels, and the variable used in `facet_wrap` for identification

```
p +
  geom_text(data = facet_labels,
            aes(x = x_coord,
                y = y_coord,
                label = label,
                color = nonwhite),
            size = 2.5) +
  theme(legend.position = "none")
```



Ropeladder exercise: Cy Young award



Ropeladder exercise: Cy Young award

- Model results `cyyoungFD.csv` can be found on the course website
  - Background: North American baseball pitchers from 1980 to 2002 competing for the Cy Young Award
  - Outcome: binary; winning the Cy Young Award or not
  - Model: logistic regression
  - Note: estimated quantities of interests (e.g. first differences) are obtained via `simcf`, `Zelig`, or `ggeffects` packages
  - Variables in the model (not that they are important. . .):

Rows/Columns	Explanation
winpct	The percentage of games which the pitcher personally won
era	The number of runs the pitcher allows per 9 innings
strikeout	The number of strikeouts the pitcher collected over a season
innings	The number of innings (periods) a pitcher played during the season
walks	The number of walks the pitcher collected over a season
pe	The first difference in expected prob. of winning Cy Young Award
lower	Lower bound of the 95% confidence intervals
upper	Upper bound of the 95% confidence intervals

Ropeladder exercise: Cy Young award

```
cyyoungFD <- read_csv("data/cyyoungFD.csv")
cyyoungFD
```

```
## # A tibble: 5 x 7
##   covariate   pe_m1 lower_m1 upper_m1   pe_m2 lower_m2 upper_m2
##   <chr>       <dbl>    <dbl>    <dbl>   <dbl>    <dbl>    <dbl>
## 1 winpct     0.342    0.192     0.484  0.198    0.111     0.280
## 2 era       -0.599   -0.749    -0.314 -0.347   -0.434    -0.182
## 3 strikeout  0.223   -0.0793    0.403  0.129   -0.0459    0.233
## 4 innings    0.277   -0.437     0.465  0.160   -0.253     0.269
## 5 walks      0.0808  -0.299     0.320  0.0468  -0.173     0.185
```

Ropeladder exercise: Cy Young award

- Major challenges:
  1. How do we reformat the data in a "tidy way"?
  2. How do we create a variable that classifies statistically significant covariates from non-significant ones?
     - How can we indicate statistical non-significance using white-filled circles?
  3. How do we plot the model results as a ropeladder?
  4. How can we sort the covariates according to their effect sizes?
  5. How can we juxtapose two models' results effectively?

Ropeladder exercise: Cy Young award

1. How do we reformat the data in a "tidy way"?
   - Motivation: create a column `model` with values {m1, m2} such that we can map `model` to aesthetics
     - i.e. `aes(colour = model)`
   - We need to learn `pivot_longer` and `pivot_wider` (available in the recent `tidyr` versions)
     - Update by `update.packages("tidyverse")`

```
## # A tibble: 10 x 5
##    covariate model      pe   lower  upper
##    <chr>     <chr>   <dbl>   <dbl>  <dbl>
##  1 winpct    m1     0.342   0.192   0.484
##  2 era       m1    -0.599  -0.749  -0.314
##  3 strikeout m1     0.223  -0.0793  0.403
##  4 innings   m1     0.277  -0.437   0.465
##  5 walks     m1     0.0808 -0.299   0.320
##  6 winpct    m2     0.198   0.111   0.280
##  7 era       m2    -0.347  -0.434  -0.182
##  8 strikeout m2     0.129  -0.0459  0.233
##  9 innings   m2     0.160  -0.253   0.269
## 10 walks     m2     0.0468 -0.173   0.185
```

Ropeladder exercise: Cy Young award

1. How do we reformat the data in a "tidy way"?

```
print(cyyoungFD)
```

```
## # A tibble: 5 x 7
##   covariate   pe_m1 lower_m1 upper_m1   pe_m2 lower_m2 upper_m2
##   <chr>       <dbl>    <dbl>    <dbl>   <dbl>    <dbl>    <dbl>
## 1 winpct     0.342    0.192     0.484  0.198    0.111     0.280
## 2 era       -0.599   -0.749    -0.314 -0.347   -0.434    -0.182
## 3 strikeout  0.223   -0.0793    0.403  0.129   -0.0459    0.233
## 4 innings    0.277   -0.437     0.465  0.160   -0.253     0.269
## 5 walks      0.0808  -0.299     0.320  0.0468  -0.173     0.185
```

Ropeladder exercise: Cy Young award

1. How do we reformat the data in a "tidy way"?

```
cyyoungFD %>%
  pivot_longer(cols = pe_m1:upper_m2,
               names_to = "col_names",
               values_to = "value")
```

```
## # A tibble: 30 x 3
##    covariate col_names  value
##    <chr>     <chr>      <dbl>
##  1 winpct    pe_m1      0.342
##  2 winpct    lower_m1   0.192
##  3 winpct    upper_m1   0.484
##  4 winpct    pe_m2      0.198
##  5 winpct    lower_m2   0.111
##  6 winpct    upper_m2   0.280
##  7 era       pe_m1     -0.599
##  8 era       lower_m1  -0.749
##  9 era       upper_m1  -0.314
## 10 era       pe_m2     -0.347
## # ... with 20 more rows
```

Ropeladder exercise: Cy Young award

1. How do we reformat the data in a "tidy way"?

```
cyyoungFD %>%
  pivot_longer(cols = pe_m1:upper_m2,
               names_to = "col_names",
               values_to = "value") %>%
  separate(col_names, into = c("stat", "model"), sep = "_")
```

```
## # A tibble: 30 x 4
##    covariate stat  model  value
##    <chr>     <chr> <chr>  <dbl>
##  1 winpct    pe    m1     0.342
##  2 winpct    lower m1     0.192
##  3 winpct    upper m1     0.484
##  4 winpct    pe    m2     0.198
##  5 winpct    lower m2     0.111
##  6 winpct    upper m2     0.280
##  7 era       pe    m1    -0.599
##  8 era       lower m1    -0.749
##  9 era       upper m1    -0.314
## 10 era       pe    m2    -0.347
## # ... with 20 more rows
```

Ropeladder exercise: Cy Young award

1. How do we reformat the data in a "tidy way"?

```
cyyoungFD %>%
  pivot_longer(cols = pe_m1:upper_m2,
               names_to = "col_names",
               values_to = "value") %>%
  separate(col_names, into = c("stat", "model"), sep = "_") %>%
  pivot_wider(names_from = stat, values_from = value)
```

```
## # A tibble: 10 x 5
##    covariate model      pe   lower  upper
##    <chr>     <chr>   <dbl>   <dbl>  <dbl>
##  1 winpct    m1      0.342   0.192   0.484
##  2 winpct    m2      0.198   0.111   0.280
##  3 era       m1     -0.599  -0.749  -0.314
##  4 era       m2     -0.347  -0.434  -0.182
##  5 strikeout m1      0.223  -0.0793  0.403
##  6 strikeout m2      0.129  -0.0459  0.233
##  7 innings   m1      0.277  -0.437   0.465
##  8 innings   m2      0.160  -0.253   0.269
##  9 walks     m1      0.0808 -0.299   0.320
## 10 walks     m2      0.0468 -0.173   0.185
```

Ropeladder exercise: Cy Young award

1. How do we reformat the data in a "tidy way"?

```
cyyoungFD <- cyyoungFD %>%
  pivot_longer(cols = pe_m1:upper_m2,
               names_to = "col_names",
               values_to = "value") %>%
  separate(col_names, into = c("stat", "model"), sep = "_") %>%
  pivot_wider(names_from = stat, values_from = value) %>%
  arrange(model)

print(cyyoungFD)
```

```
## # A tibble: 10 x 5
##    covariate model      pe   lower  upper
##    <chr>     <chr>   <dbl>   <dbl>  <dbl>
##  1 winpct    m1      0.342   0.192   0.484
##  2 era       m1     -0.599  -0.749  -0.314
##  3 strikeout m1      0.223  -0.0793  0.403
##  4 innings   m1      0.277  -0.437   0.465
##  5 walks     m1      0.0808 -0.299   0.320
##  6 winpct    m2      0.198   0.111   0.280
##  7 era       m2     -0.347  -0.434  -0.182
##  8 strikeout m2      0.129  -0.0459  0.233
##  9 innings   m2      0.160  -0.253   0.269
## 10 walks     m2      0.0468 -0.173   0.185
```

Ropeladder exercise: Cy Young award

2. How do we create a variable that classifies statistically significant covariates from non-significant ones?
   - How do we know if a covariate's effect is statistically significant or not?

```
print(cyyoungFD)
```

```
## # A tibble: 10 x 5
##    covariate model      pe   lower  upper
##    <chr>     <chr>   <dbl>   <dbl>  <dbl>
##  1 winpct    m1      0.342   0.192   0.484
##  2 era       m1     -0.599  -0.749  -0.314
##  3 strikeout m1      0.223  -0.0793  0.403
##  4 innings   m1      0.277  -0.437   0.465
##  5 walks     m1      0.0808 -0.299   0.320
##  6 winpct    m2      0.198   0.111   0.280
##  7 era       m2     -0.347  -0.434  -0.182
##  8 strikeout m2      0.129  -0.0459  0.233
##  9 innings   m2      0.160  -0.253   0.269
## 10 walks     m2      0.0468 -0.173   0.185
```

Ropeladder exercise: Cy Young award

2. How do we create a variable that classifies statistically significant covariates from non-significant ones?

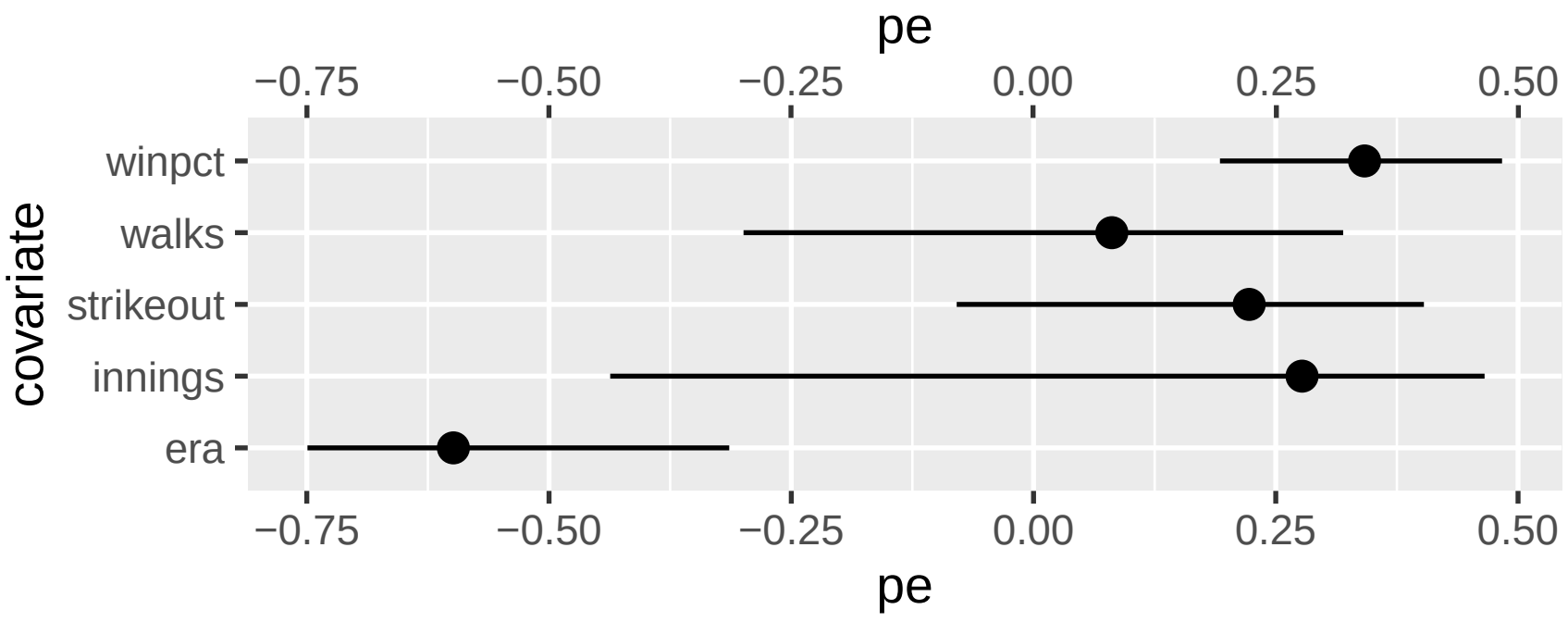
```
cyyoungFD <- cyyoungFD %>%
  mutate(
    signif = case_when(
      lower > 0 & upper > 0 ~ TRUE, # Both bounds are above zero = signif
      lower < 0 & upper < 0 ~ TRUE, # Both bounds are below zero = signif
      TRUE ~ FALSE)                 # Everything else is not signif
  )
print(cyyoungFD)
```

```
## # A tibble: 10 x 6
##    covariate model      pe   lower  upper signif
##    <chr>     <chr>   <dbl>   <dbl>  <dbl> <lgl>
##  1 winpct    m1      0.342   0.192  0.484 TRUE
##  2 era       m1     -0.599  -0.749 -0.314 TRUE
##  3 strikeout m1      0.223  -0.0793 0.403 FALSE
##  4 innings   m1      0.277  -0.437  0.465 FALSE
##  5 walks     m1      0.0808 -0.299  0.320 FALSE
##  6 winpct    m2      0.198   0.111  0.280 TRUE
##  7 era       m2     -0.347  -0.434 -0.182 TRUE
##  8 strikeout m2      0.129  -0.0459 0.233 FALSE
##  9 innings   m2      0.160  -0.253  0.269 FALSE
## 10 walks     m2      0.0468 -0.173  0.185 FALSE
```

Ropeladder exercise: Cy Young award

3. How do we plot the model results as a ropeladder?
   - Practice: focus on model 1 and replicate the graph below
   - Hints: check out `geom_pointrange()`

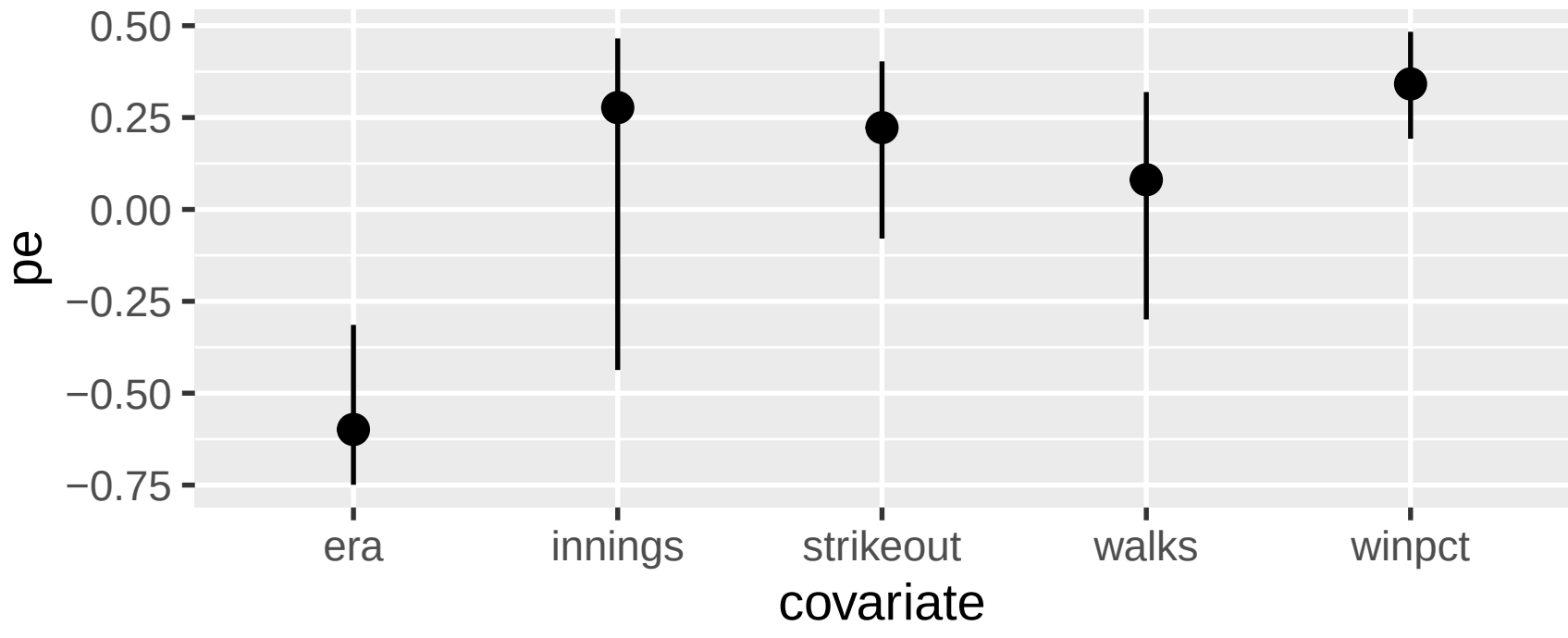
```
cyyoungFD %>%
  filter(model == "m1") %>%
  ggplot(...)
```



Ropeladder exercise: Cy Young award

3. How do we plot the model results as a ropeladder?
   - Old method: `coord_flip()`

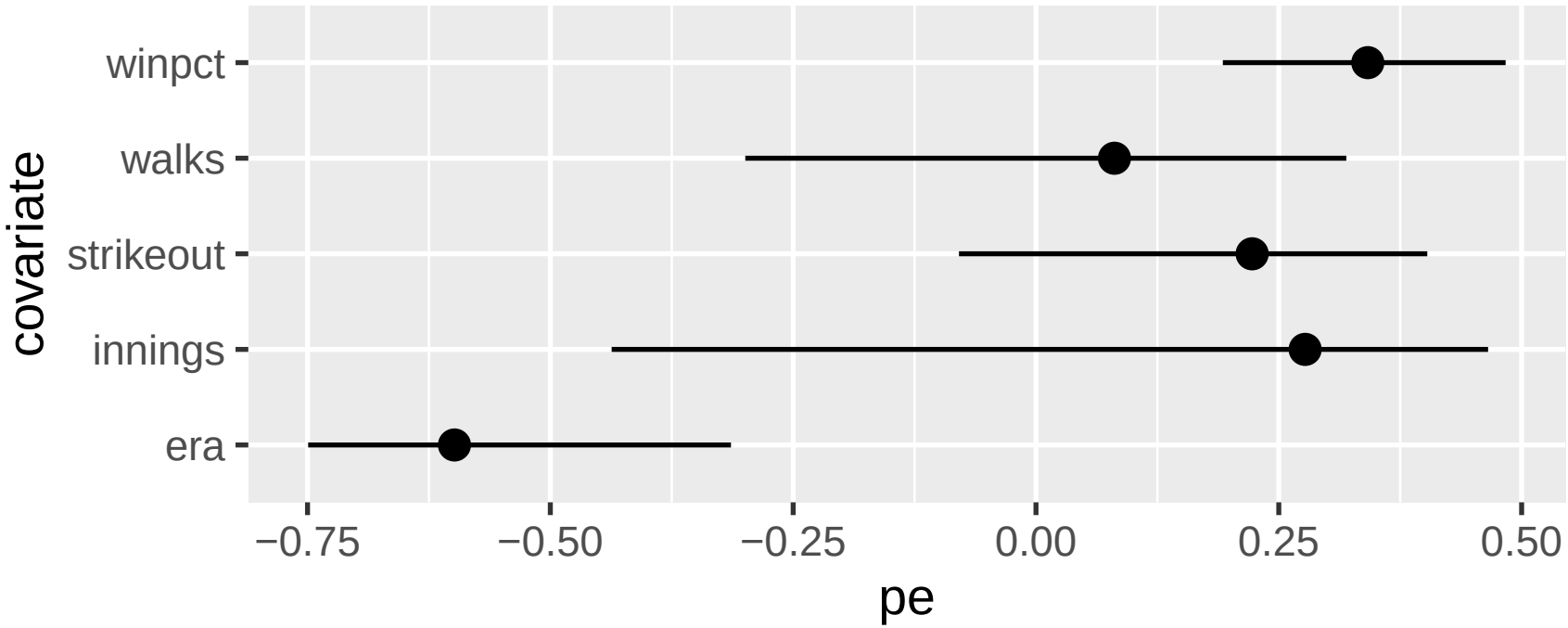
```
cyyoungFD %>%
  filter(model == "m1") %>%
  ggplot(aes(x = covariate, y = pe, ymax = upper, ymin = lower)) +
  geom_pointrange()
```



Ropeladder exercise: Cy Young award

3. How do we plot the model results as a ropeladder?
   - Old method: `coord_flip()`

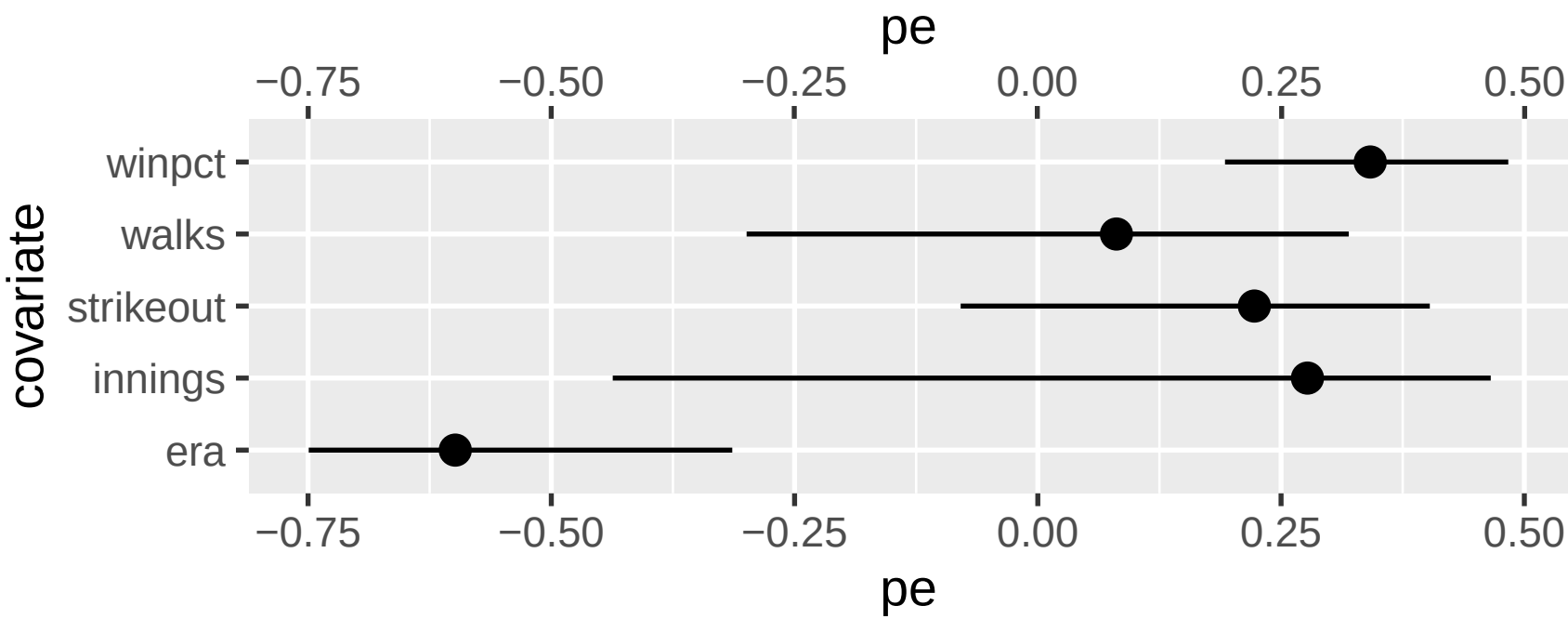
```
cyyoungFD %>%
  filter(model == "m1") %>%
  ggplot(aes(x = covariate, y = pe, ymax = upper, ymin = lower)) +
  geom_pointrange() +
  coord_flip()
```



Ropeladder exercise: Cy Young award

3. How do we plot the model results as a ropeladder?
   - Old method: `coord_flip()`

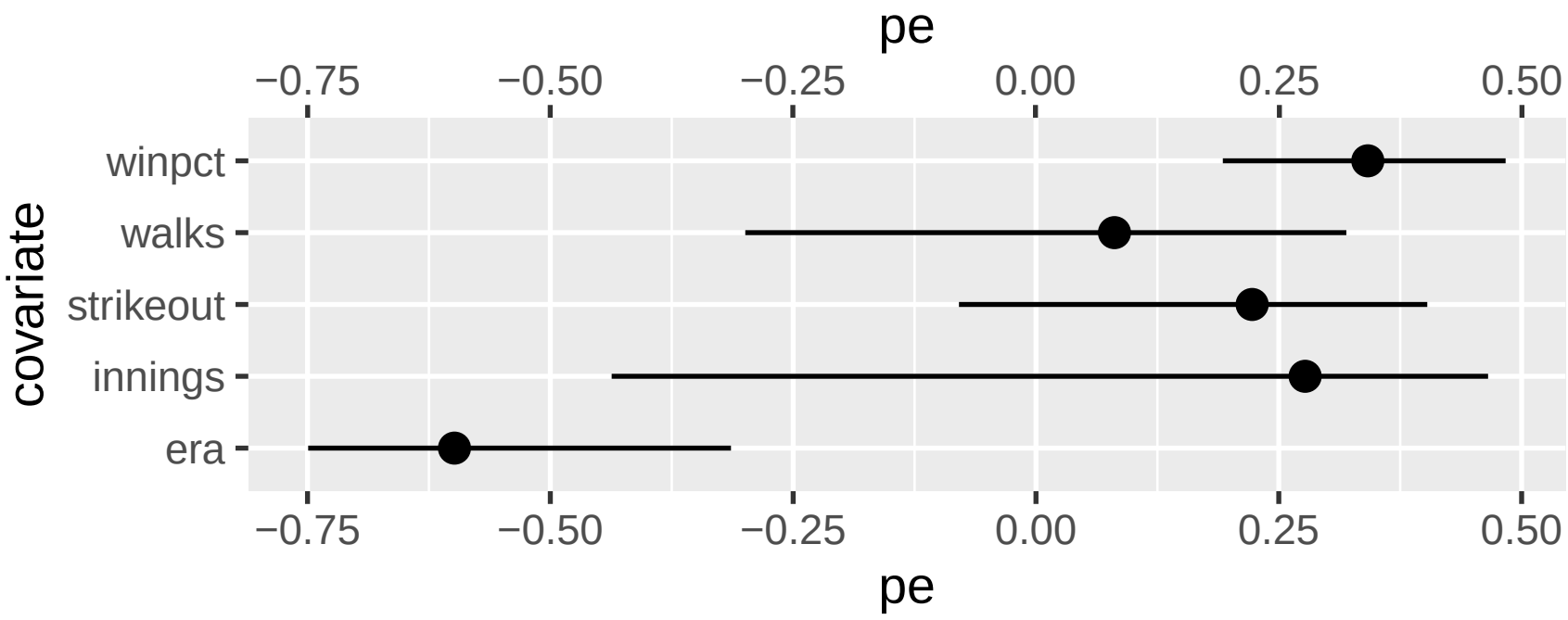
```
cyyoungFD %>%
  filter(model == "m1") %>%
  ggplot(aes(x = covariate, y = pe, ymax = upper, ymin = lower)) +
  geom_pointrange() +
  coord_flip() +
  scale_y_continuous(sec.axis = dup_axis(~ .))
```



Ropeladder exercise: Cy Young award

3. How do we plot the model results as a ropeladder?
   - New method: native `xmax` and `xmin` inputs in new `ggplot2`
   - But problems with legend keys displayed in wrong orientation

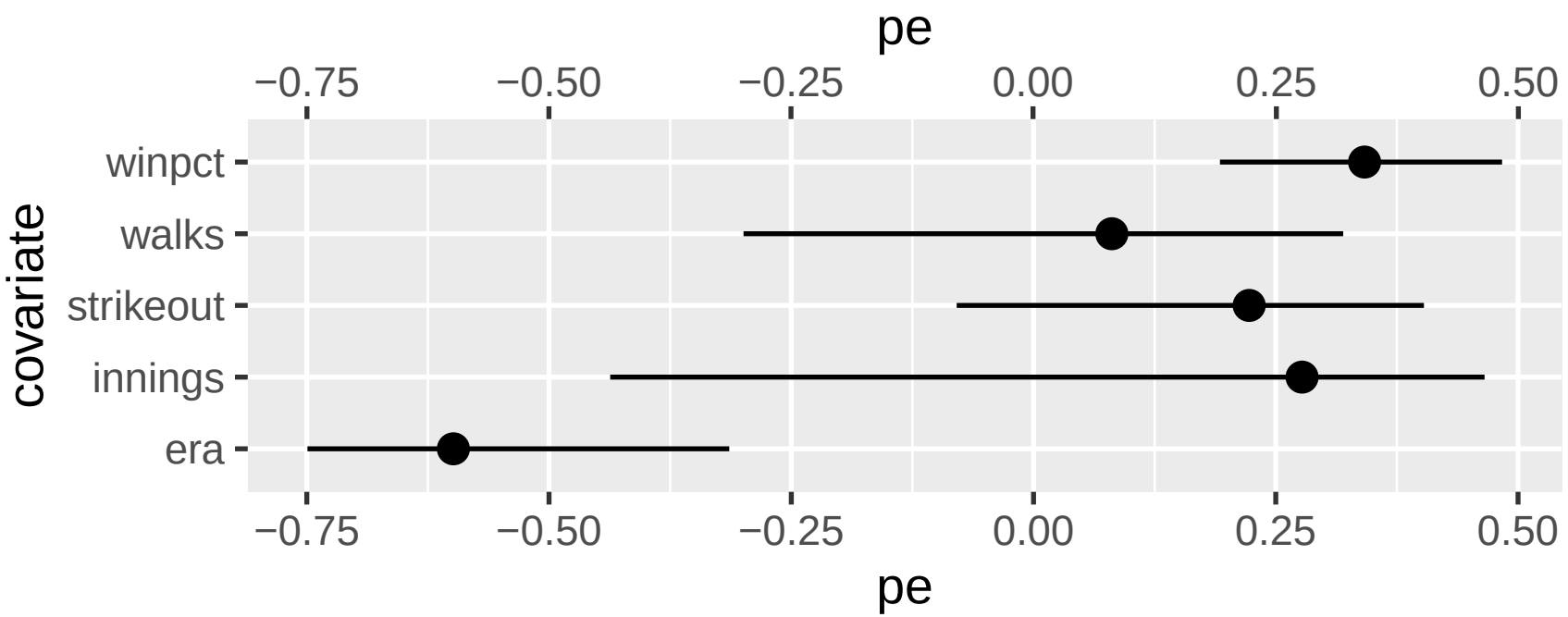
```
cyyoungFD %>%
  filter(model == "m1") %>%
  ggplot(aes(y = covariate, x = pe, xmax = upper, xmin = lower)) +
  geom_pointrange() +
  scale_x_continuous(sec.axis = dup_axis(~ .))
```



Ropeladder exercise: Cy Young award

3. How do we plot the model results as a ropeladder?
   - Preferred method: `ggstance` package implements horizontal versions of different geoms with correctly oriented legend keys

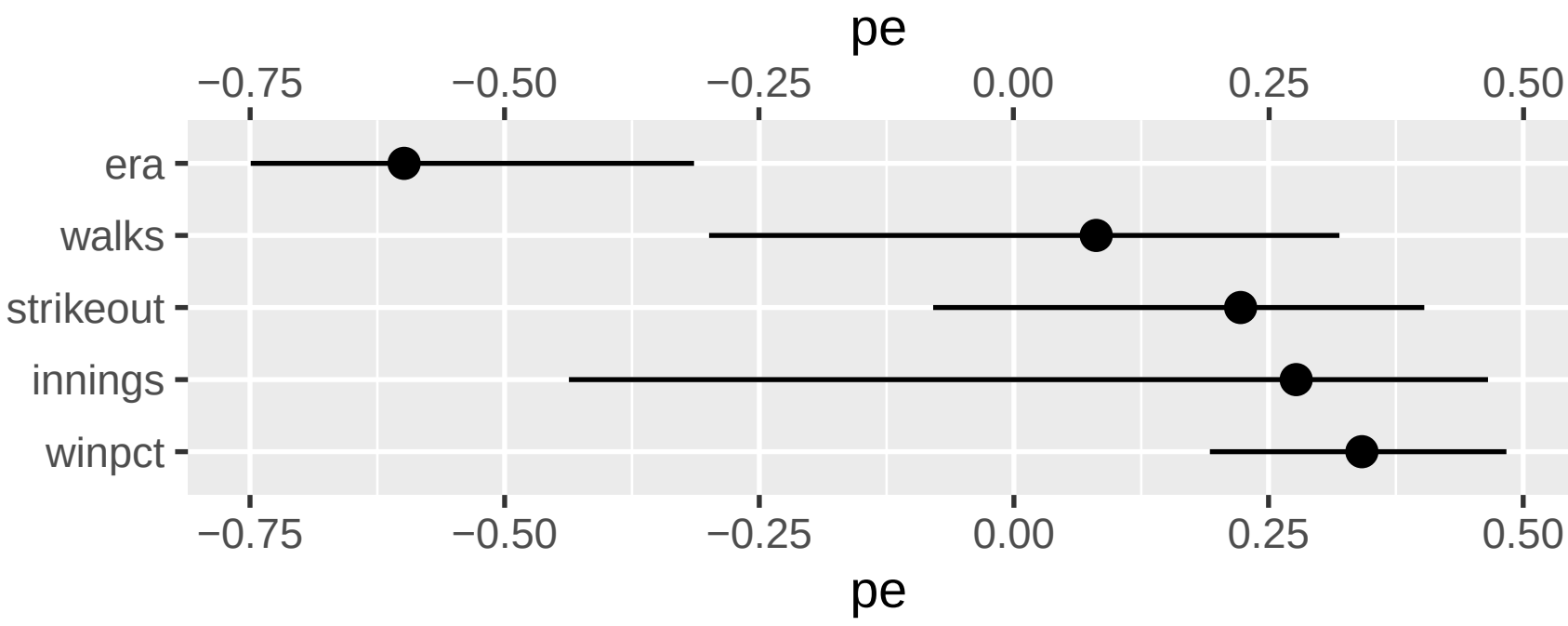
```
library(ggstance)

cyyoungFD %>%
  filter(model == "m1") %>%
  ggplot(aes(y = covariate, x = pe, xmax = upper, xmin = lower)) +
  geom_pointrangeh() + # note the "h" at the end
  scale_x_continuous(sec.axis = dup_axis(~ .))
```



Ropeladder exercise: Cy Young award

4. How can we sort the covariates according to their effect sizes?
   - `fct_reorder()` from `forcats` package (in `tidyverse`)

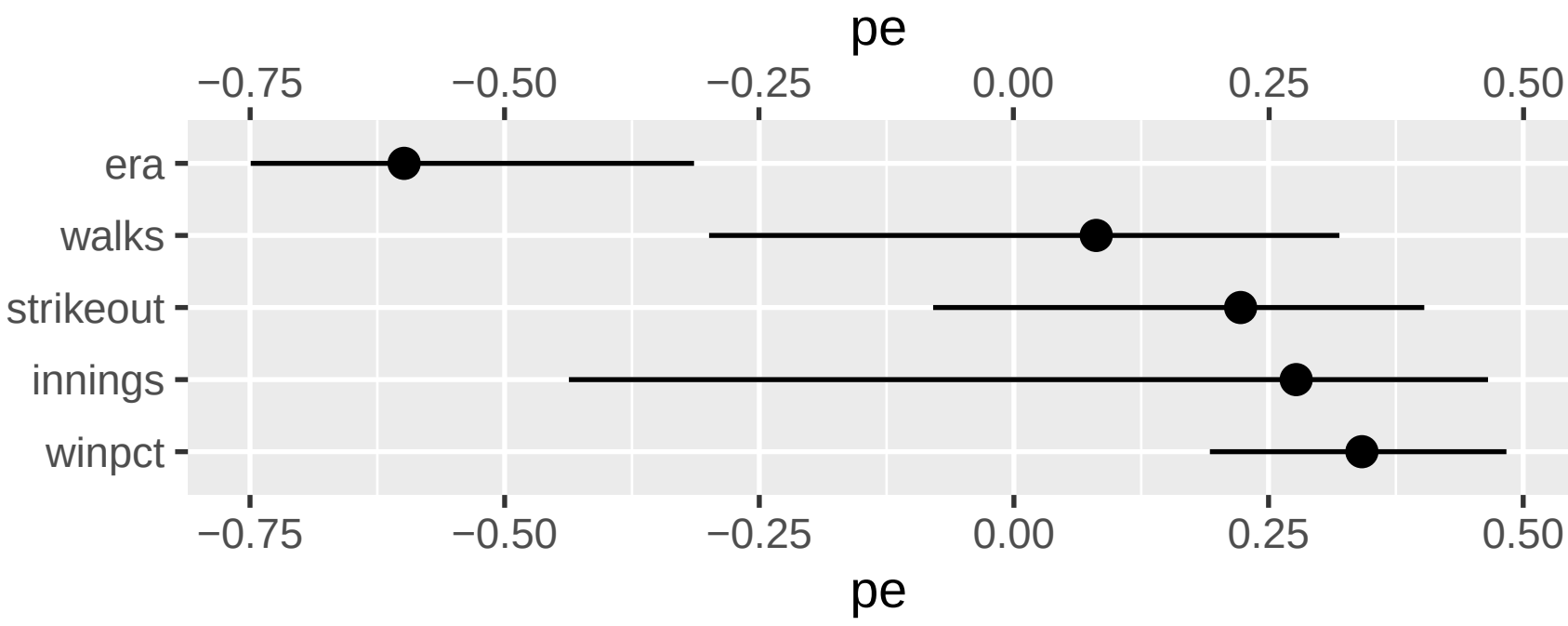
```
cyyoungFD %>%
  filter(model == "m1") %>%
  mutate(covariate = fct_reorder(covariate, pe, .desc = TRUE)) %>%
  ggplot(aes(y = covariate, x = pe, xmax = upper, xmin = lower)) +
  geom_pointrangeh() +
  scale_x_continuous(sec.axis = dup_axis(~ .)) +
  labs(y = NULL)
```



Ropeladder exercise: Cy Young award

4. How can we sort the covariates according to their effect sizes?
   - `fct_reorder()` from `forcats` package (in `tidyverse`)

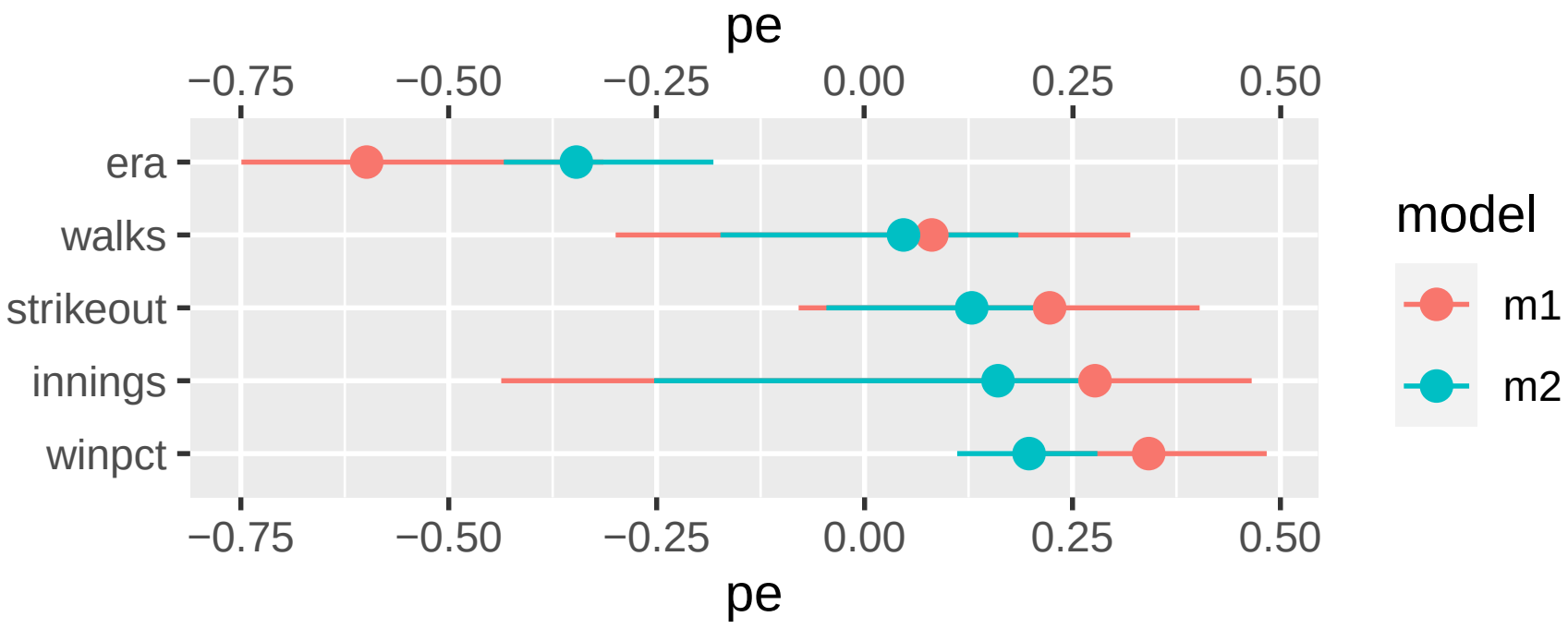
```
cyyoungFD %>%
  filter(model == "m1") %>%
  ggplot(aes(y = fct_reorder(covariate, pe, .desc = TRUE),
             x = pe, xmax = upper, xmin = lower)) +
  geom_pointrangeh() +
  scale_x_continuous(sec.axis = dup_axis(~ .)) +
  labs(y = NULL)
```



Ropeladder exercise: Cy Young award

5. How can we juxtapose two models' results effectively?
   - If you simply map `model` to colour...

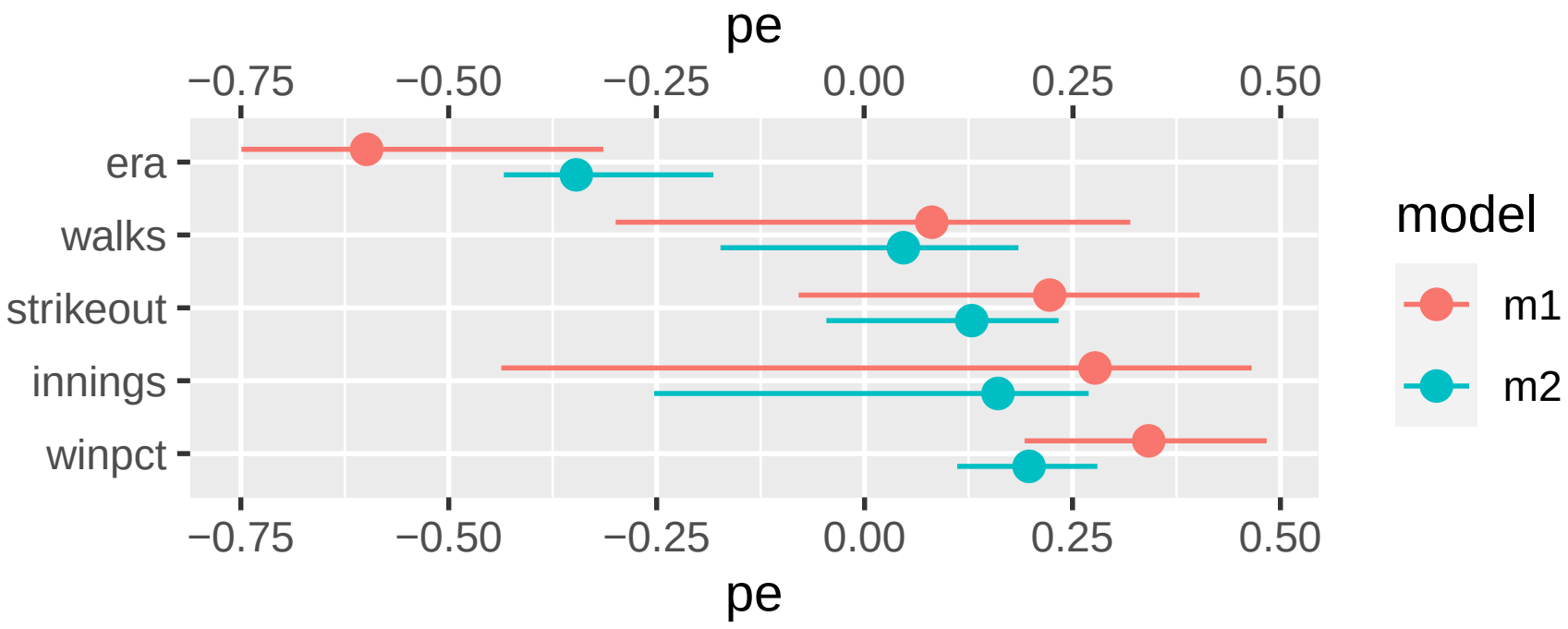
```
ggplot(cyyoungFD, aes(y = fct_reorder(covariate, pe, .desc = TRUE),
                      x = pe, xmax = upper, xmin = lower,
                      colour = model)) +
  geom_pointrangeh() +
  scale_x_continuous(sec.axis = dup_axis(~ .)) +
  labs(y = NULL)
```



Ropeladder exercise: Cy Young award

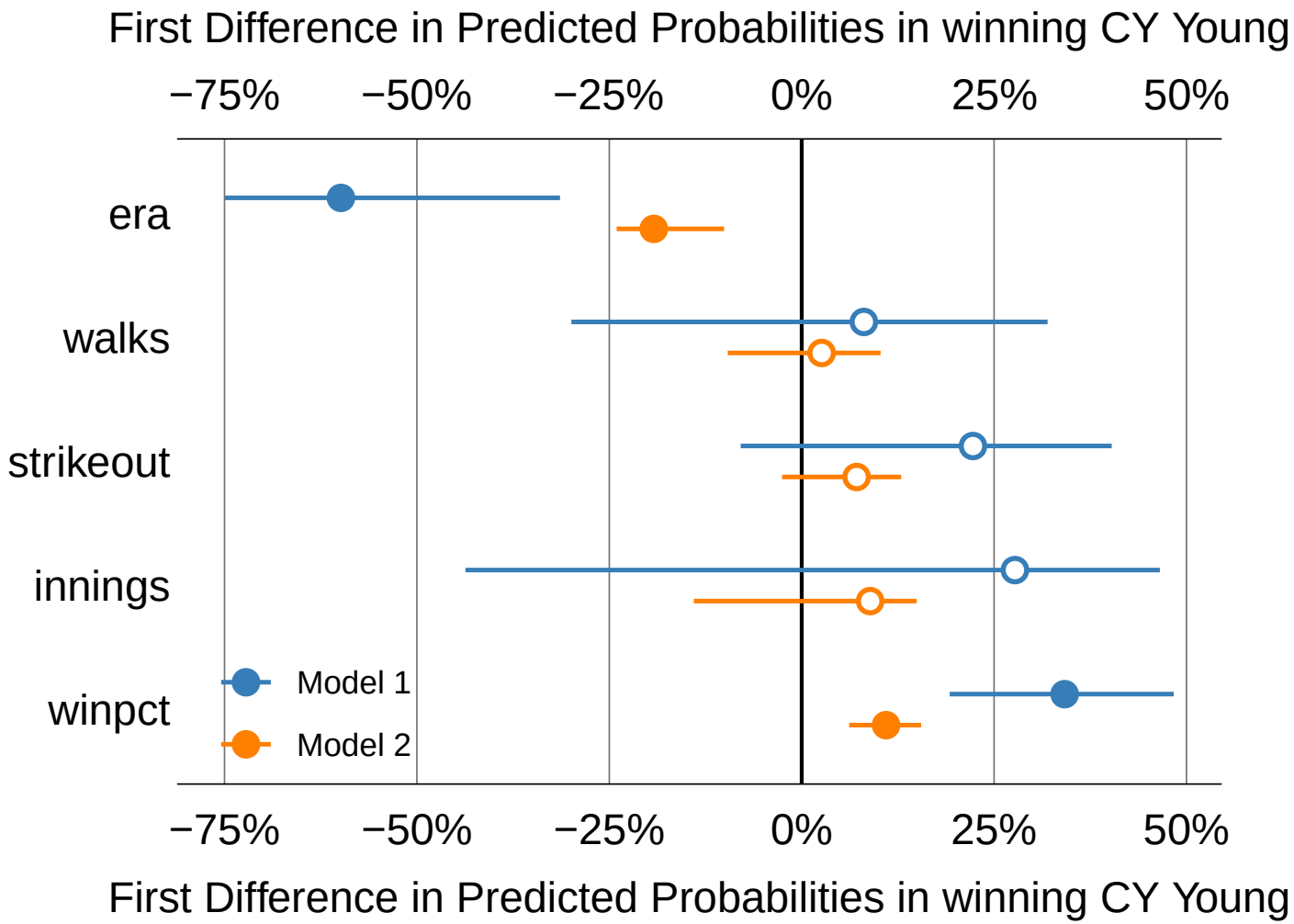
5. How can we juxtapose two models' results effectively?
   - Use `position_dodge2v` inside `geom_pointrange()`

```
ggplot(cyyoungFD, aes(y = fct_reorder(covariate, pe, .desc = TRUE),
                      x = pe, xmax = upper, xmin = lower,
                      colour = model)) +
  geom_pointrangeh(position = position_dodge2v(height = 0.7)) +
  scale_x_continuous(sec.axis = dup_axis(~ .)) +
  labs(y = NULL)
```



Ropeladder exercise: Cy Young award

- Remaining challenge: How can we indicate statistical non-significance using white-filled circles?
- Practice time: reproduce the following graph as much as you can

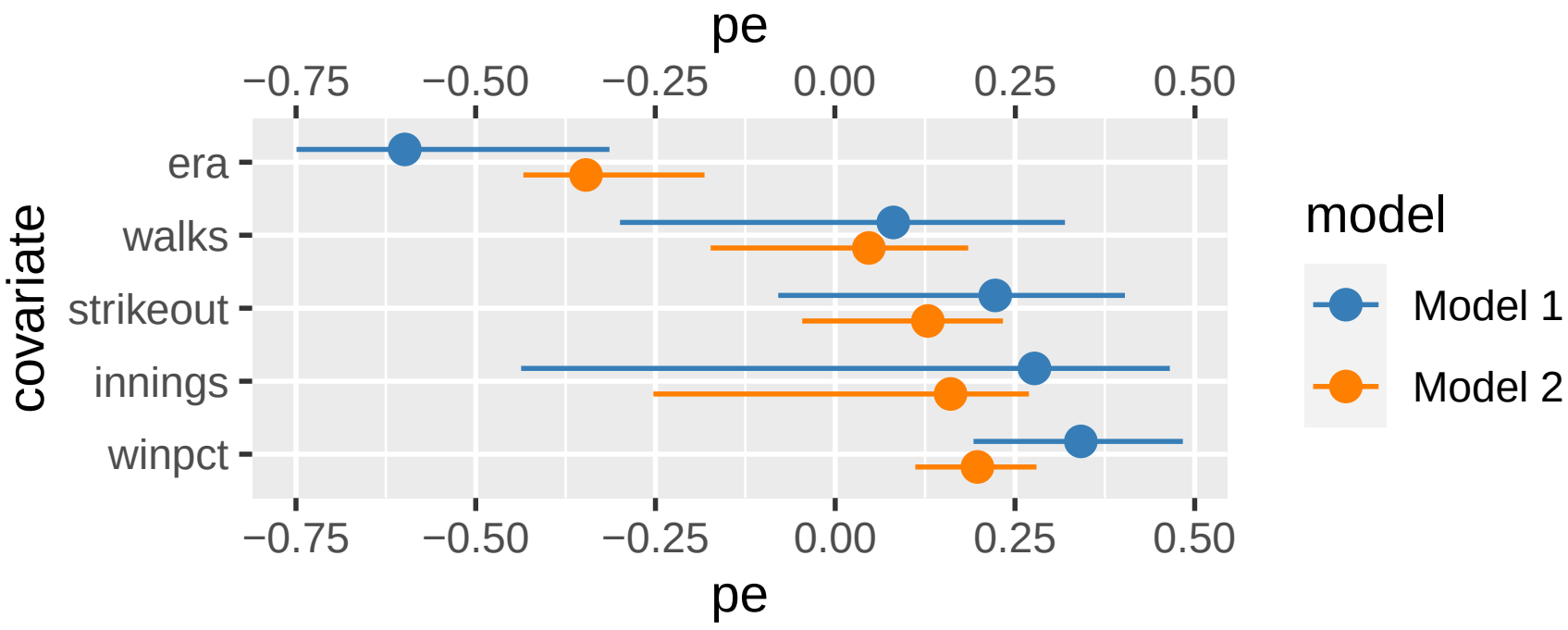


Ropeladder exercise: Cy Young award

```
# Wrangle data before visualization
cyyoungFD <- cyyoungFD %>%
  mutate(covariate = fct_reorder(covariate, pe, .desc = TRUE),
         model = case_when(model == "m1" ~ "Model 1",
                           model == "m2" ~ "Model 2"))

# Get nice colors
brewer <- brewer.pal(9, "Set1")
blue <- brewer[2]
orange <- brewer[5]
```

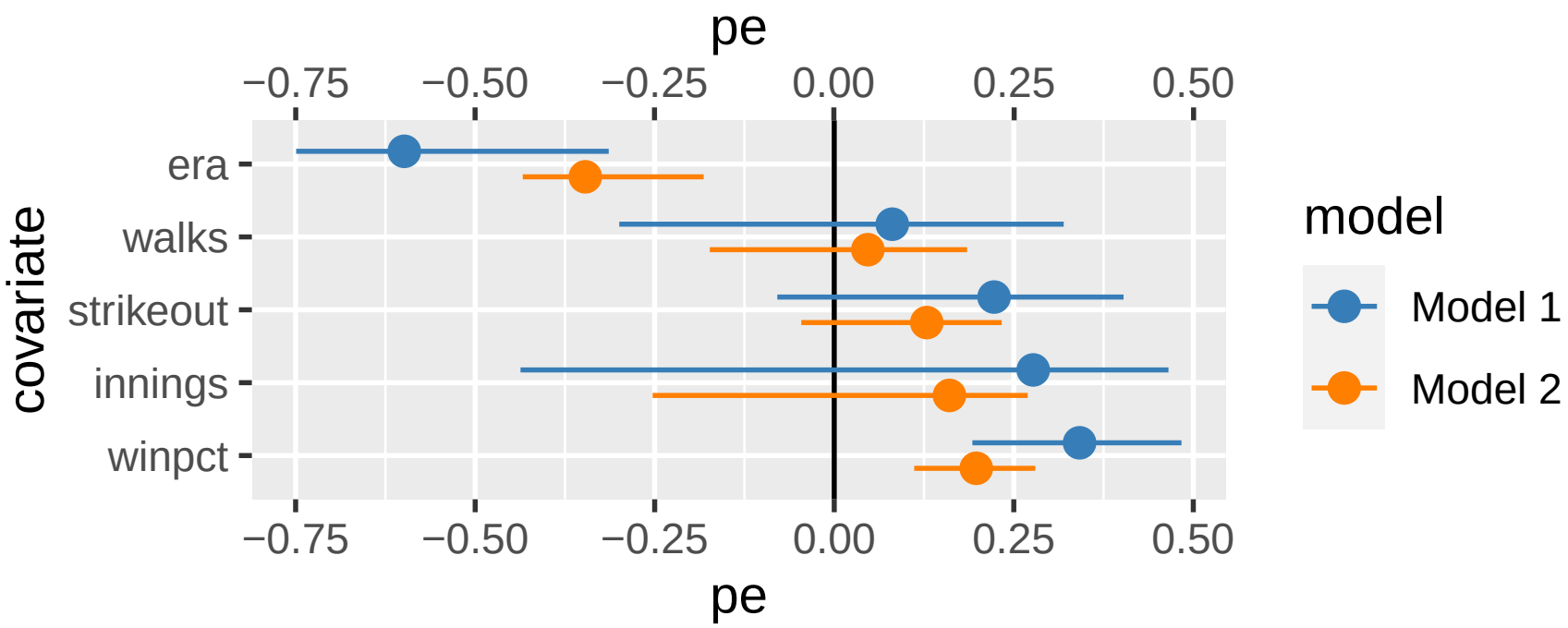
Ropeladder exercise: Cy Young award

```
ggplot(cyyoungFD, aes(y = covariate, x = pe, xmax = upper, xmin = lower,
                      colour = model)) +
  geom_pointrangeh(position = position_dodge2v(height = 0.7)) +
  scale_colour_manual(values = c(blue, orange)) +
  scale_x_continuous(sec.axis = dup_axis(~ .))
```



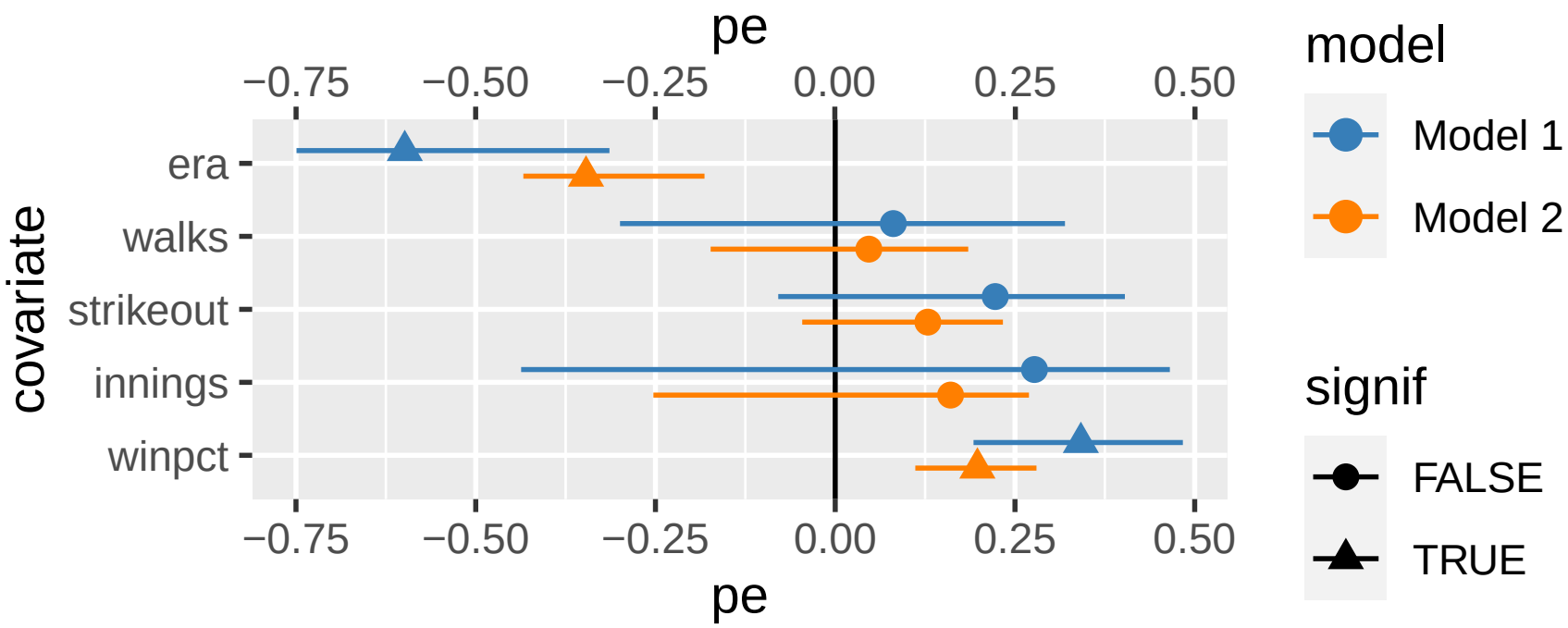
Ropeladder exercise: Cy Young award

```
ggplot(cyyoungFD, aes(y = covariate, x = pe, xmax = upper, xmin = lower,
                      colour = model)) +
  geom_vline(xintercept = 0) +
  geom_pointrangeh(position = position_dodge2v(height = 0.7)) +
  scale_colour_manual(values = c(blue, orange)) +
  scale_x_continuous(sec.axis = dup_axis(~ .))
```



Ropeladder exercise: Cy Young award

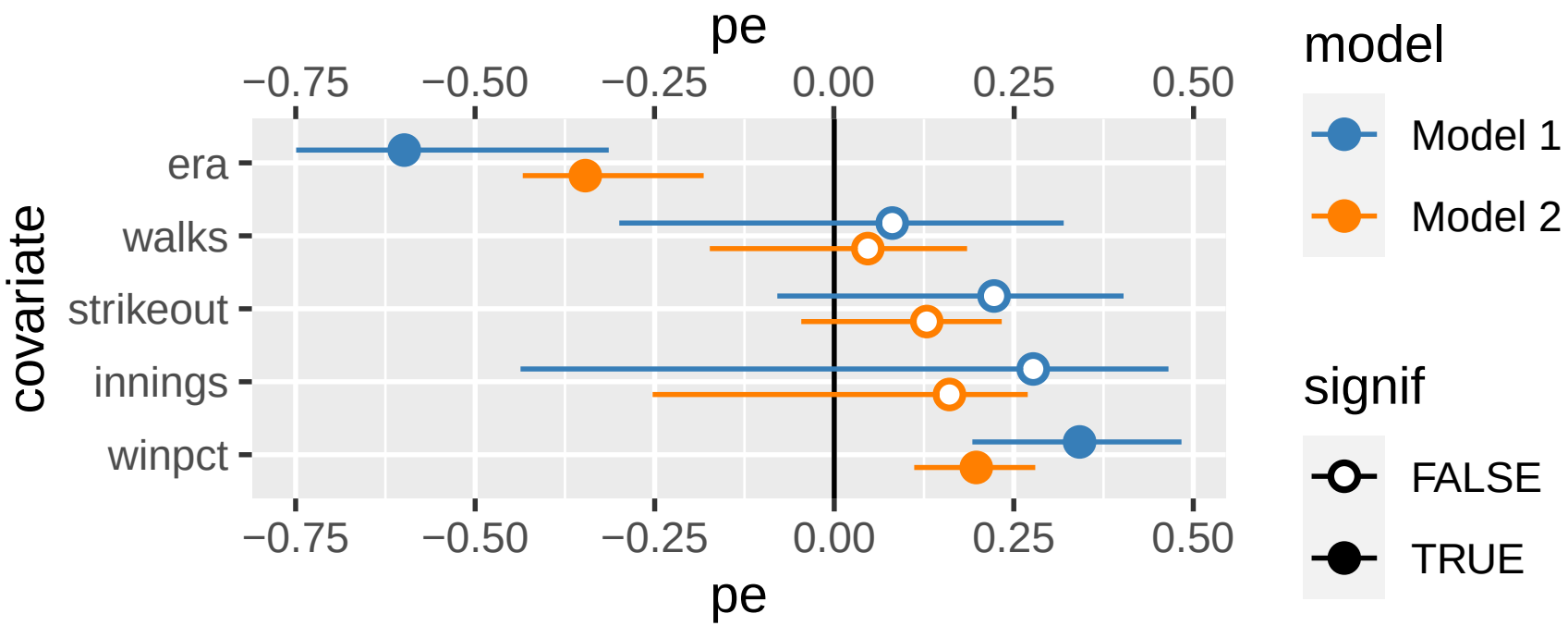
```
ggplot(cyyoungFD, aes(y = covariate, x = pe, xmax = upper, xmin = lower,
                      colour = model, shape = signif, fill = signif)) +
  geom_vline(xintercept = 0) +
  geom_pointrangeh(position = position_dodge2v(height = 0.7)) +
  scale_colour_manual(values = c(blue, orange)) +
  scale_x_continuous(sec.axis = dup_axis(~ .))
```



Ropeladder exercise: Cy Young award

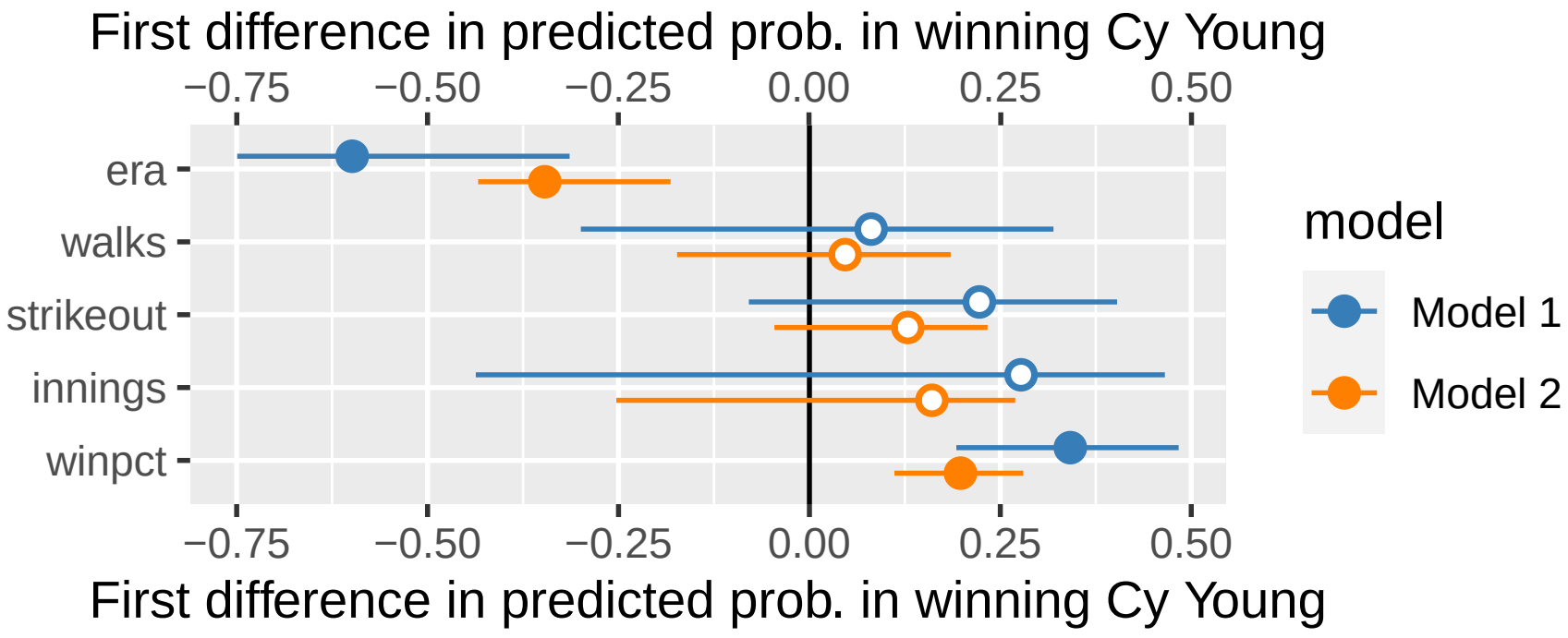
- See all possible shapes here
  - 21 = fillable circle; 19 = solid (non-fillable) circle

```
ggplot(cyyoungFD, aes(y = covariate, x = pe, xmax = upper, xmin = lower,
                      colour = model, shape = signif, fill = signif)) +
  geom_vline(xintercept = 0) +
  geom_pointrangeh(position = position_dodge2v(height = 0.7)) +
  scale_colour_manual(values = c(blue, orange)) +
  scale_shape_manual(values = c(21, 19)) +    #Non-signif-> fillable circle(21)
  scale_fill_manual(values = c("white", NA)) +#Non-signif-> fill it w. white
  scale_x_continuous(sec.axis = dup_axis(~ .))
```



Ropeladder exercise: Cy Young award

```
ggplot(cyyoungFD, aes(y = covariate, x = pe, xmax = upper, xmin = lower,
                      colour = model, shape = signif, fill = signif)) +
  geom_vline(xintercept = 0) +
  geom_pointrangeh(position = position_dodge2v(height = 0.7)) +
  scale_colour_manual(values = c(blue, orange)) +
  scale_shape_manual(values = c(21, 19)) +
  scale_fill_manual(values = c("white", NA)) +
  scale_x_continuous(sec.axis = dup_axis(~ .)) +
  guides(shape = "none", fill = "none") +
  labs(y = NULL, x = "First difference in predicted prob. in winning Cy Young")
```

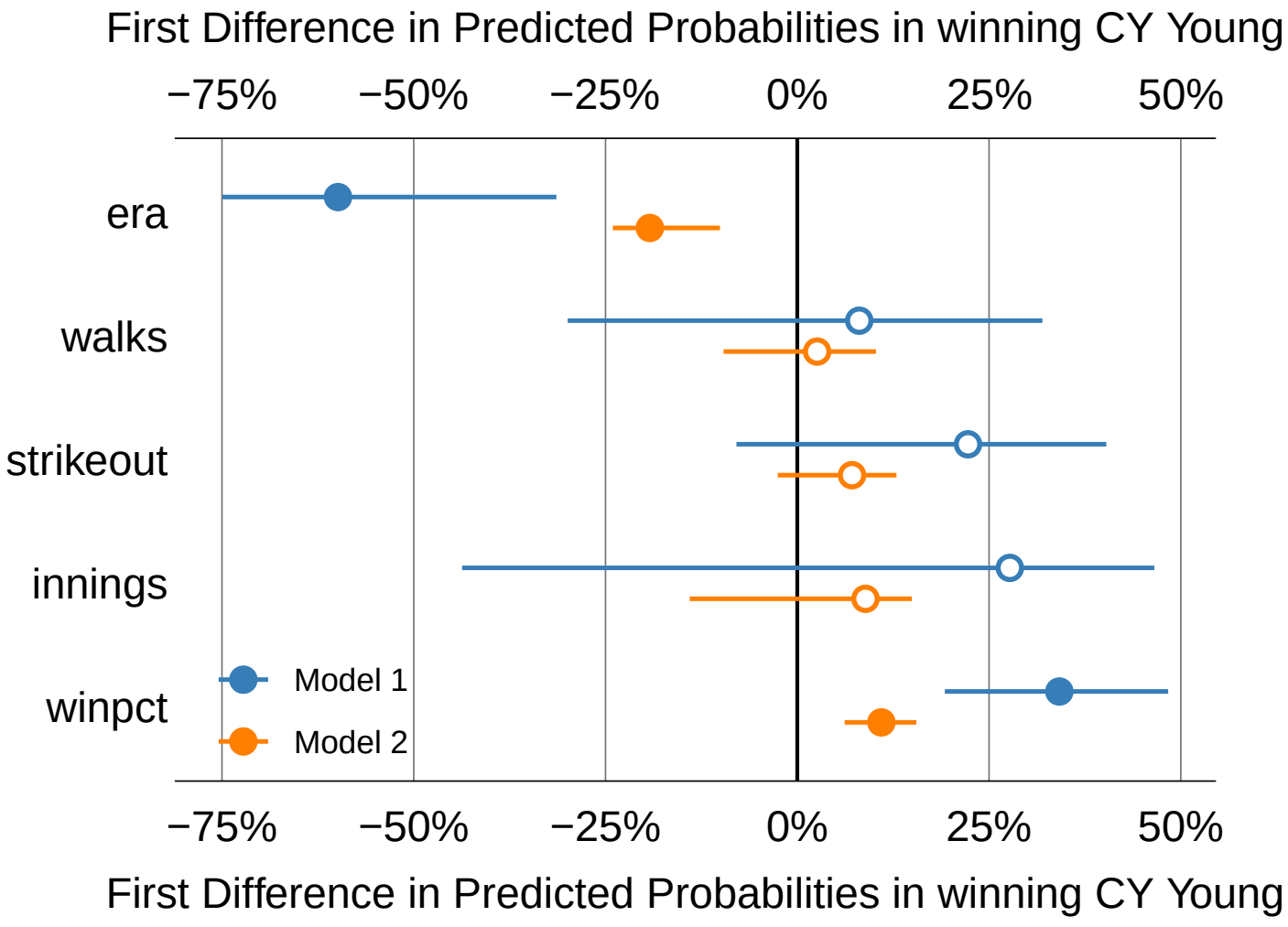


Ropeladder exercise: Cy Young award

Full code:

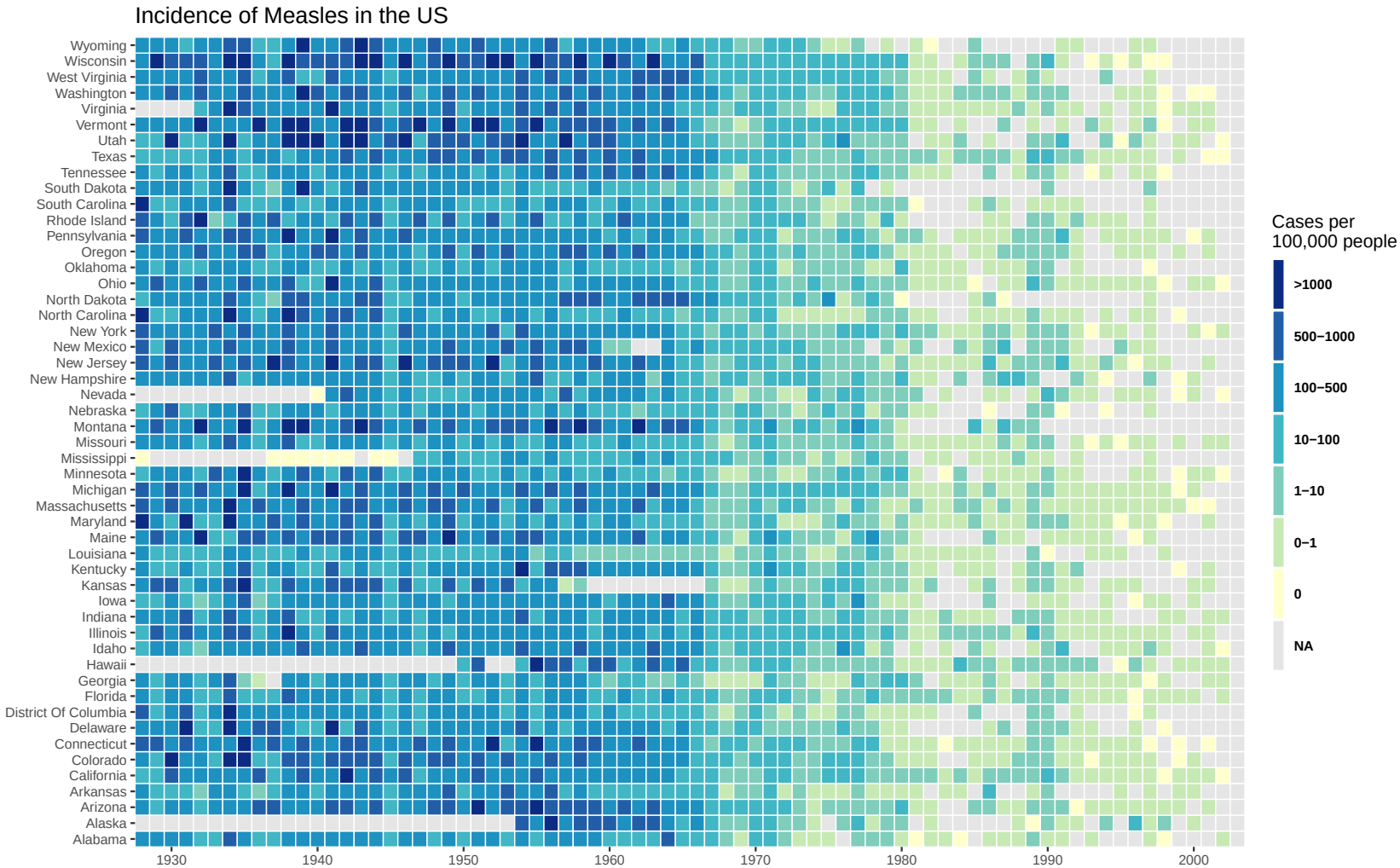
```
ggplot(cyyoungFD, aes(y = covariate, x = pe, xmax = upper, xmin = lower,
                      colour = model, shape = signif, fill = signif)) +
  geom_vline(xintercept = 0) +
  geom_pointrangeh(position = position_dodge2v(height = 0.7)) +
  scale_colour_manual(values = c(blue, orange)) +
  scale_shape_manual(values = c(21, 19)) +
  scale_fill_manual(values = c("white", NA)) +
  scale_x_continuous(sec.axis = dup_axis(~ .), labels = scales::percent) +
  guides(shape = "none", fill = "none") +
  labs(y = NULL, x = "First difference in predicted prob. in winning Cy Young") +
  theme_cavis_vgrid +
  theme(legend.position = c(0.13, 0.125), axis.ticks.x = element_blank())
```

Ropeladder exercise: Cy Young award



Heatmap exercise: scaling colour and fill

- This example is taken from Francis (2019)
  - Data: Measles level 1 incidence (cases per 100,000 people)
  - Coverage: 51 US states; 76 years (3,876 observations)



Heatmap exercise: scaling colour and fill

```
# Load data
measles <- read_csv("data/measles.csv")
```

```
##
## -- Column specification --------------------------------
## cols(
##   year = col_double(),
##   state = col_character(),
##   count = col_double(),
##   countCat = col_character()
## )
```

```
head(measles)
```

```
## # A tibble: 6 x 4
##    year state      count countCat
##   <dbl> <chr>      <dbl> <chr>
## 1  1928 Alabama    335.  100-500
## 2  1928 Alaska      NA   <NA>
```

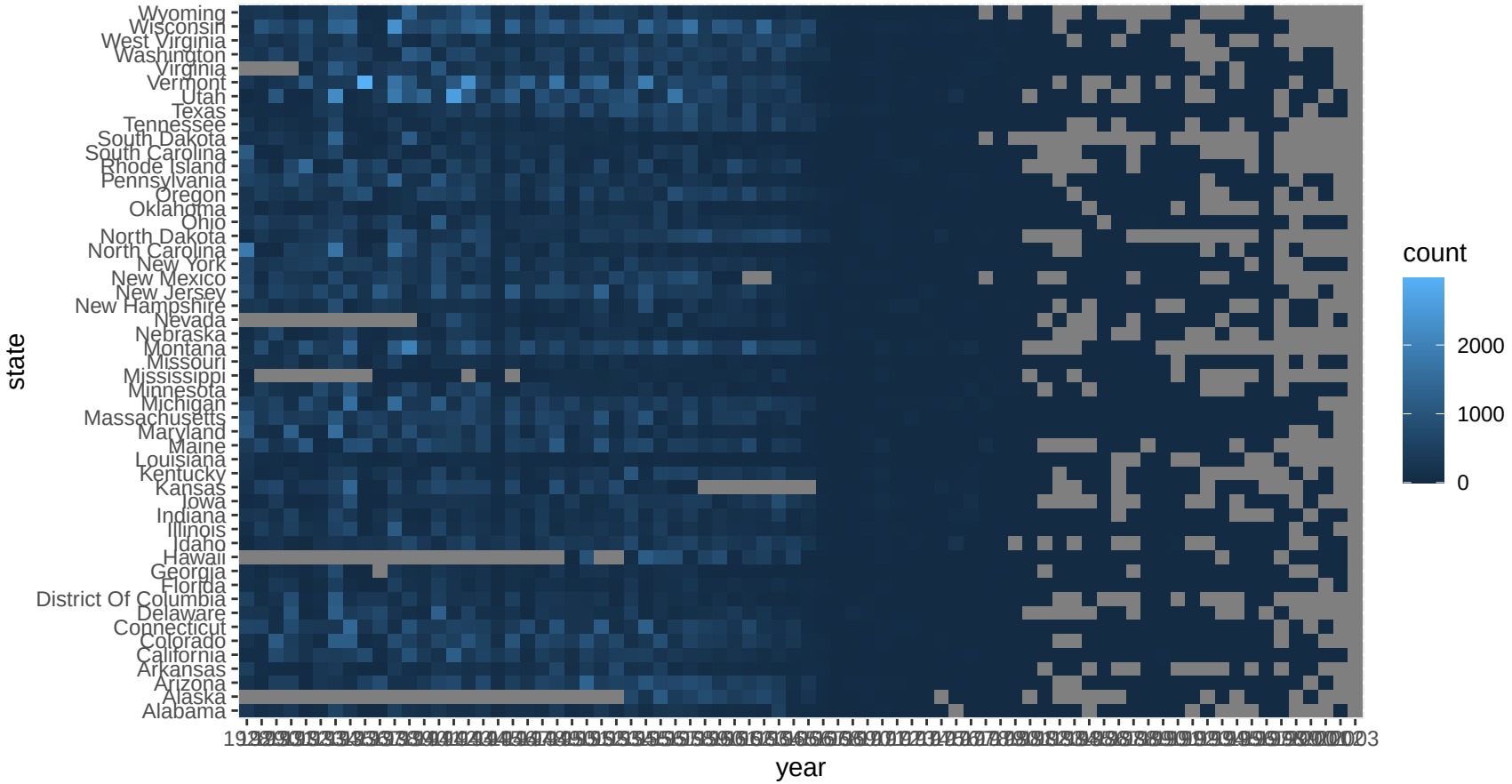
Heatmap exercise: scaling colour and fill

```r
# Factorize variables
levels <- rev(c("0", "0-1", "1-10", "10-100", "100-500", "500-1000", ">1000"))

measles <- measles %>%
  mutate(
    year = factor(year),
    state = factor(state),
    countCat = factor(countCat, levels = levels)
  )

head(measles)
```

```
## # A tibble: 6 x 4
##   year  state      count countCat
##   <fct> <fct>      <dbl> <fct>
## 1 1928  Alabama    335.  100-500
## 2 1928  Alaska      NA   <NA>
## 3 1928  Arizona    201.  100-500
## 4 1928  Arkansas   482.  100-500
## 5 1928  California  69.2 10-100
## 6 1928  Colorado   207.  100-500
```

Heatmap exercise: scaling colour and fill

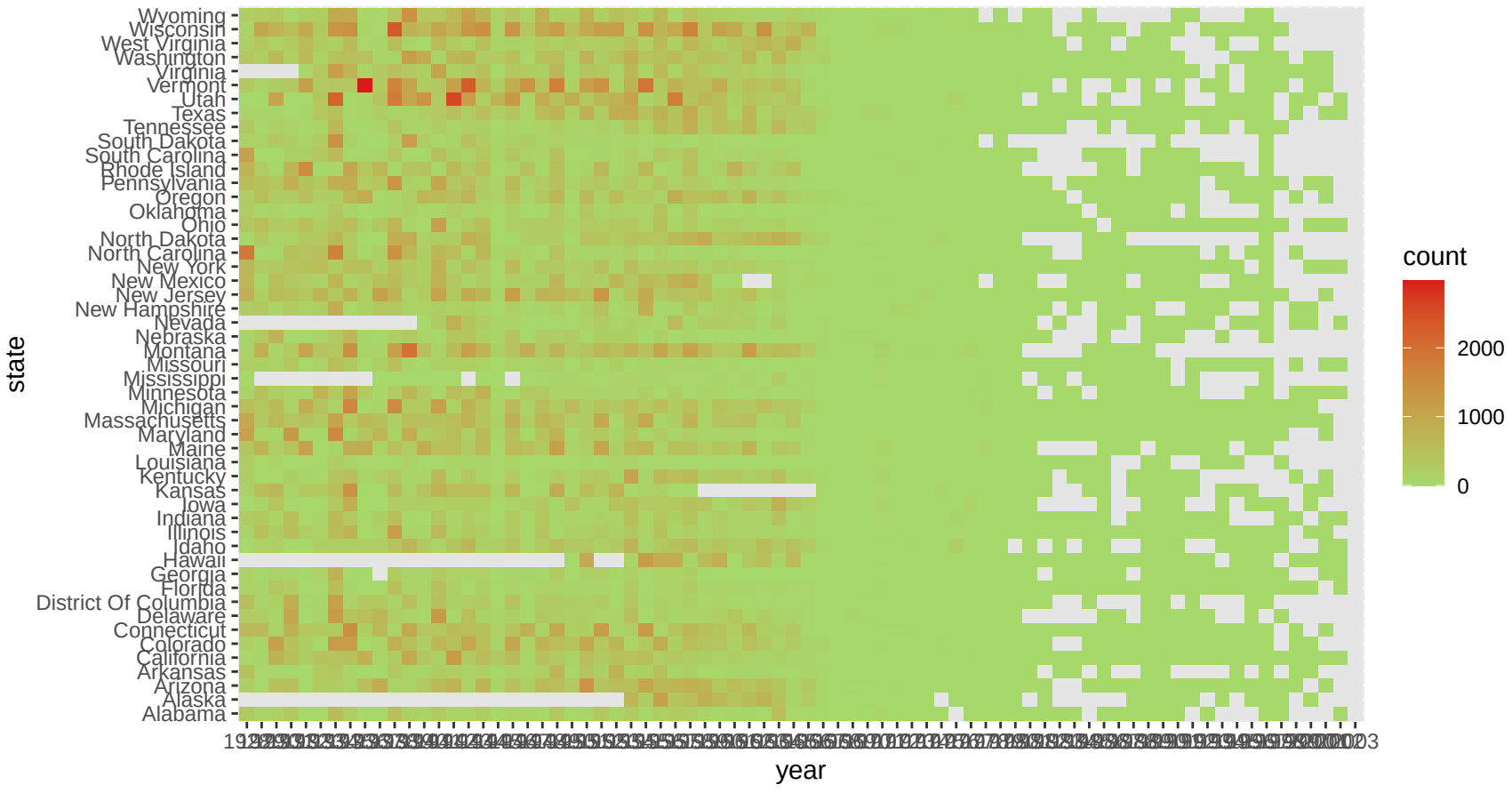
```
ggplot(measles, aes(x = year, y = state, fill = count))+
  geom_tile()
```



Heatmap exercise: scaling colour and fill

- For continuous color values, use `scale_fill_gradient`
  - To pick colors, reference here

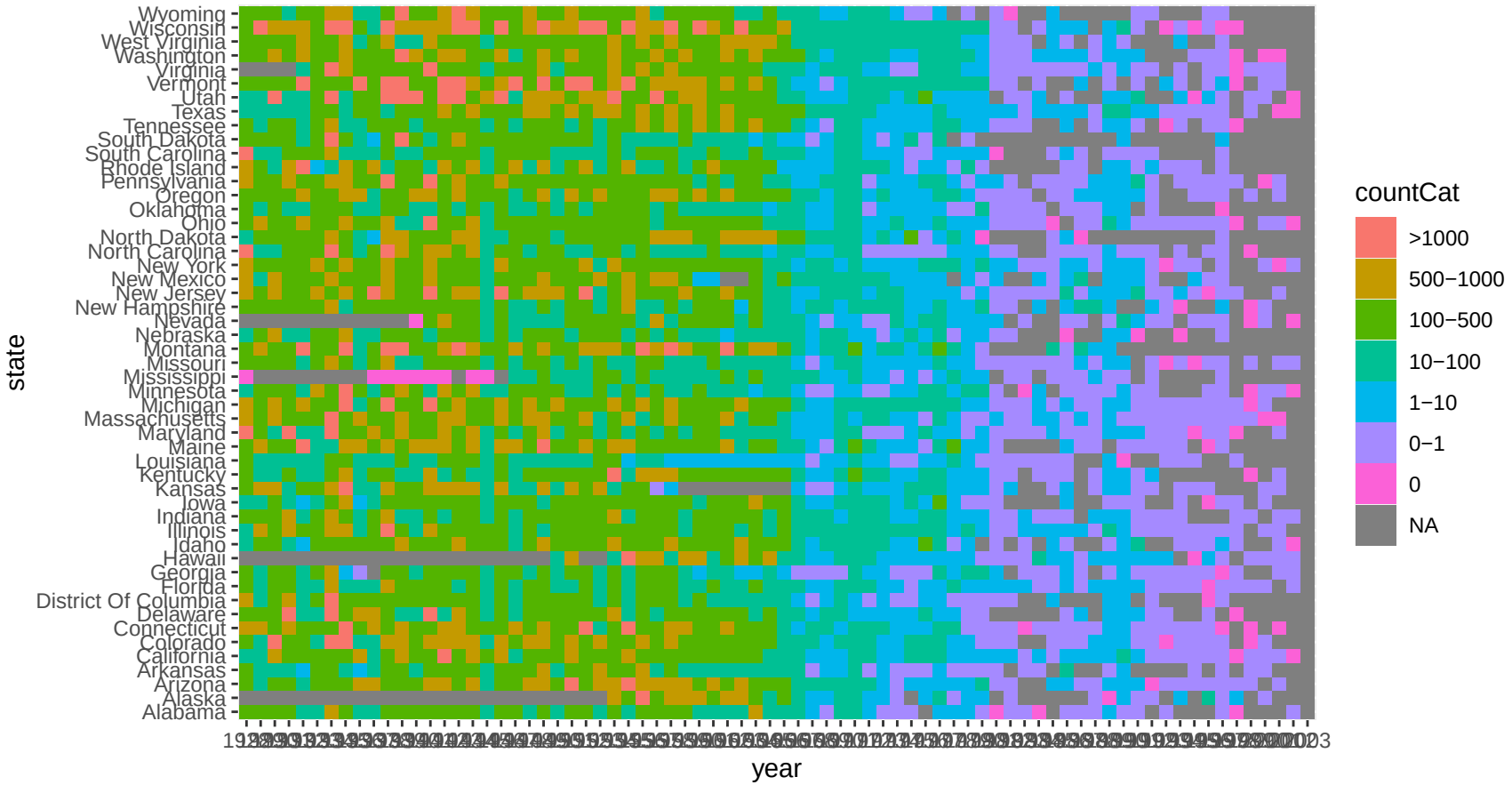
```
ggplot(measles, aes(x = year, y = state, fill = count)) +
  geom_tile() +
  scale_fill_gradient(high = "#d7191c", low = "#a6d96a", na.value = "grey90")
```



Heatmap exercise: scaling colour and fill

- Plot the categorical variable, `countCat`, instead

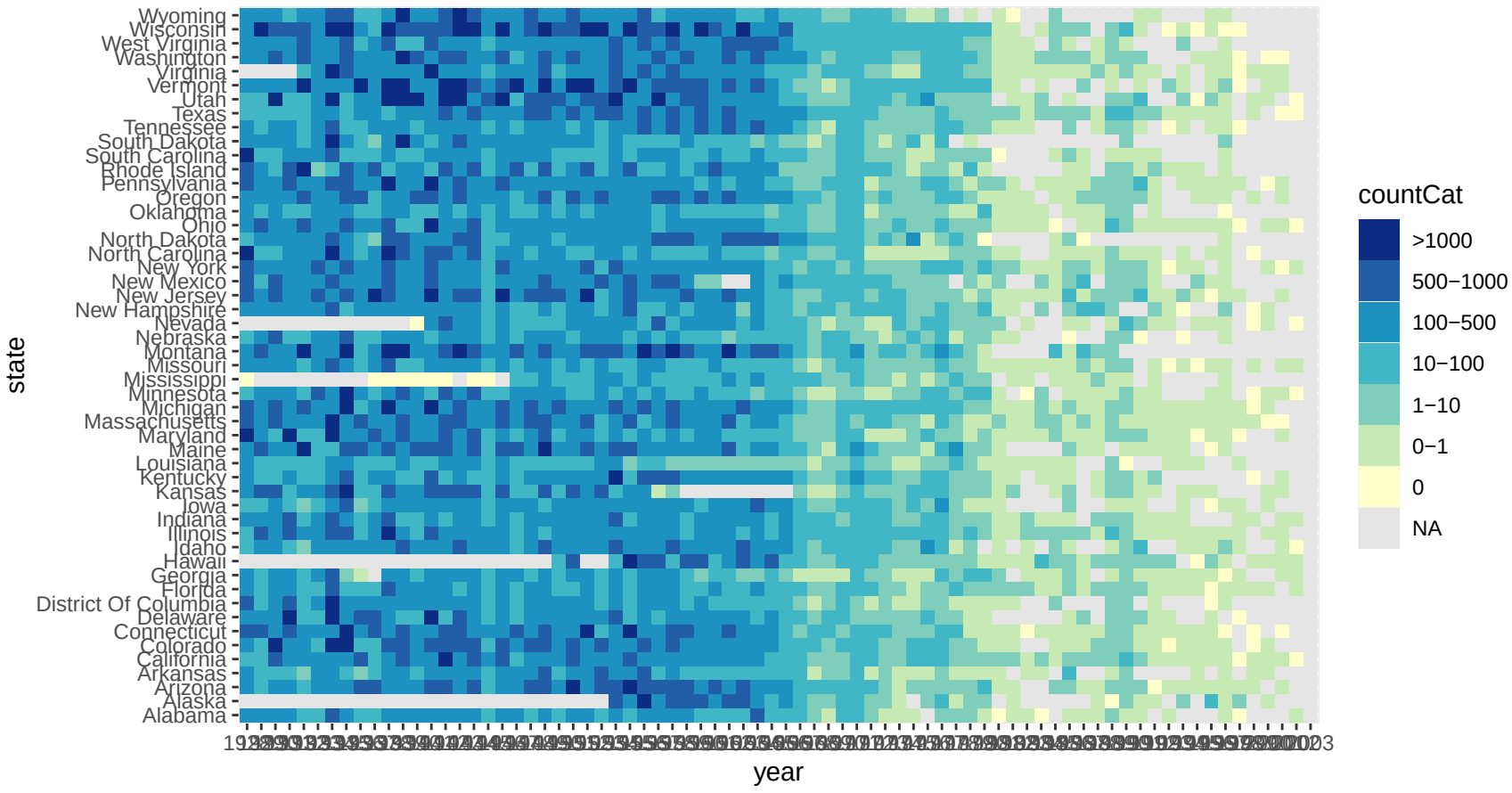
```
ggplot(measles, aes(x = year, y = state, fill = countCat)) +
  geom_tile()
```



Heatmap exercise: scaling colour and fill

- For categorical color values, use `scale_fill_brewer`
  - Run `RColorBrewer::display.brewer.all()` for all palettes

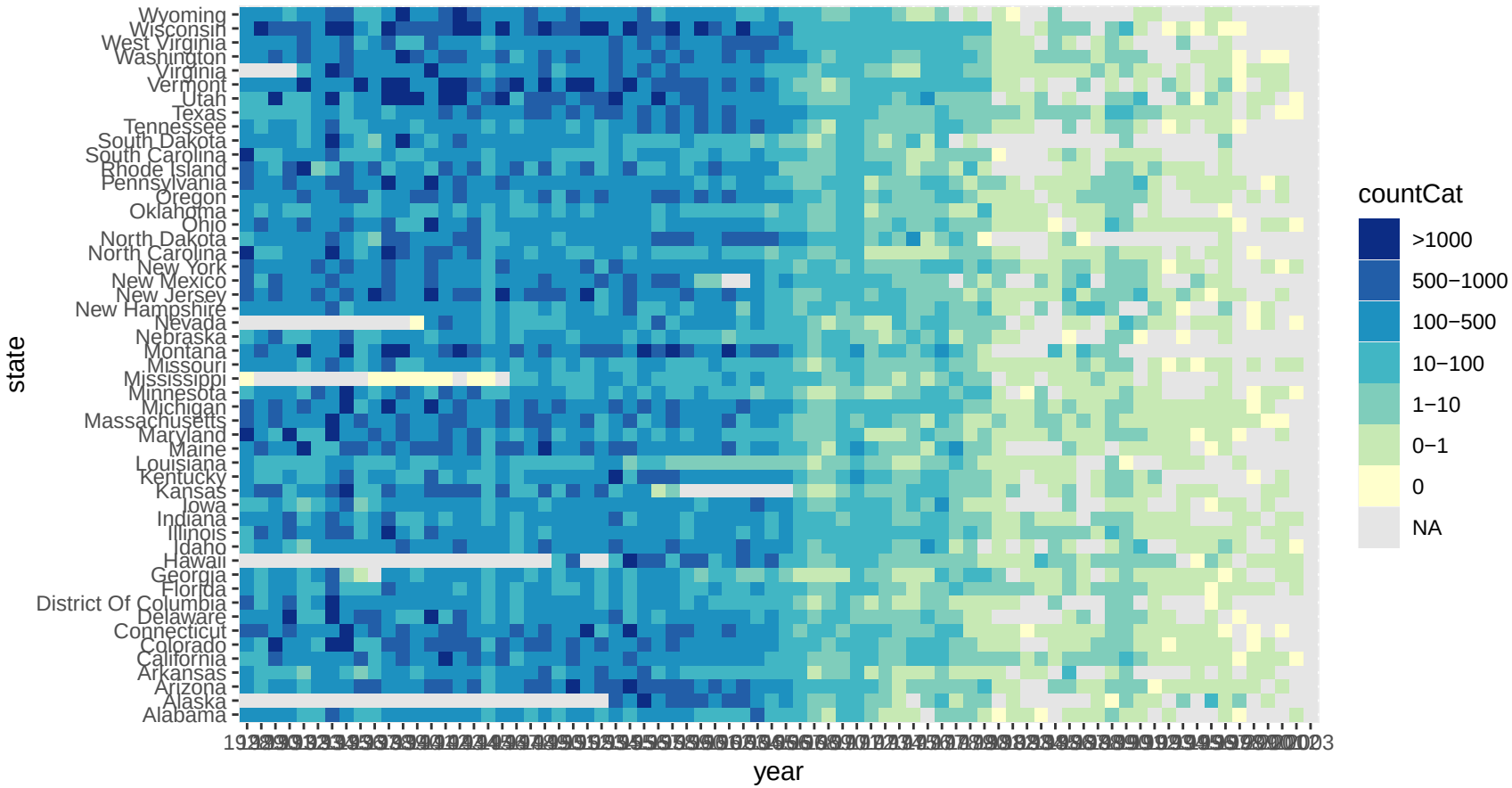
```
ggplot(measles, aes(x = year, y = state, fill = countCat)) +
  geom_tile() +
  scale_fill_brewer(palette = "YlGnBu", direction = -1, na.value = "grey90")
```



Heatmap exercise: scaling colour and fill

- Alternatively, use `RColorBrewer` and `scale_fill_manual`

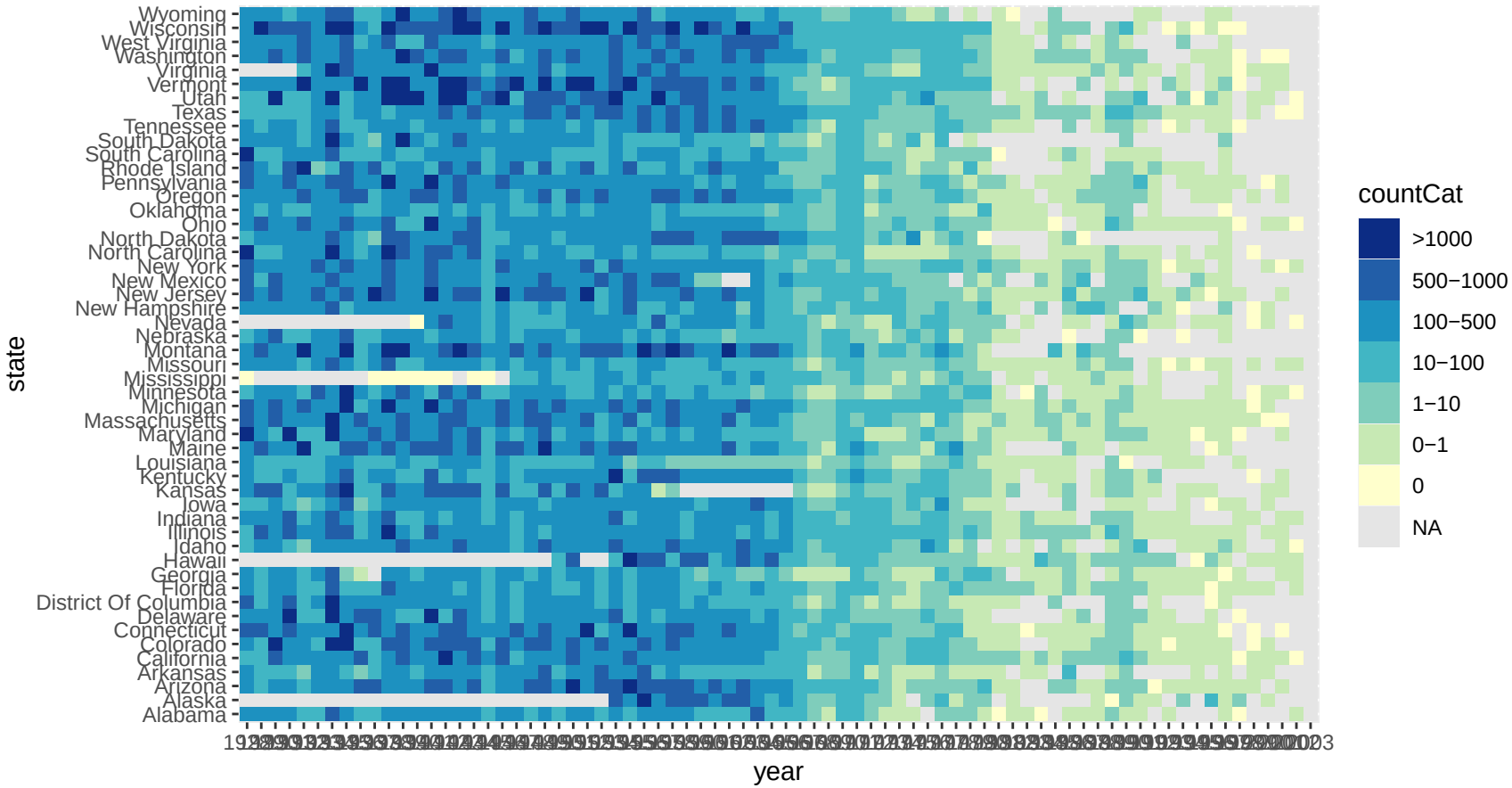
```
blues <- rev(brewer.pal(7, "YlGnBu"))

ggplot(measles, aes(x = year, y = state, fill = countCat)) +
  geom_tile() +
  scale_fill_manual(values = blues, na.value = "grey90")
```



Heatmap exercise: scaling colour and fill

- Use cluster analysis to sort the states

```
ggplot(measles, aes(x = year, y = state, fill = countCat)) +
  geom_tile() +
  scale_fill_manual(values = blues, na.value = "grey90")
```

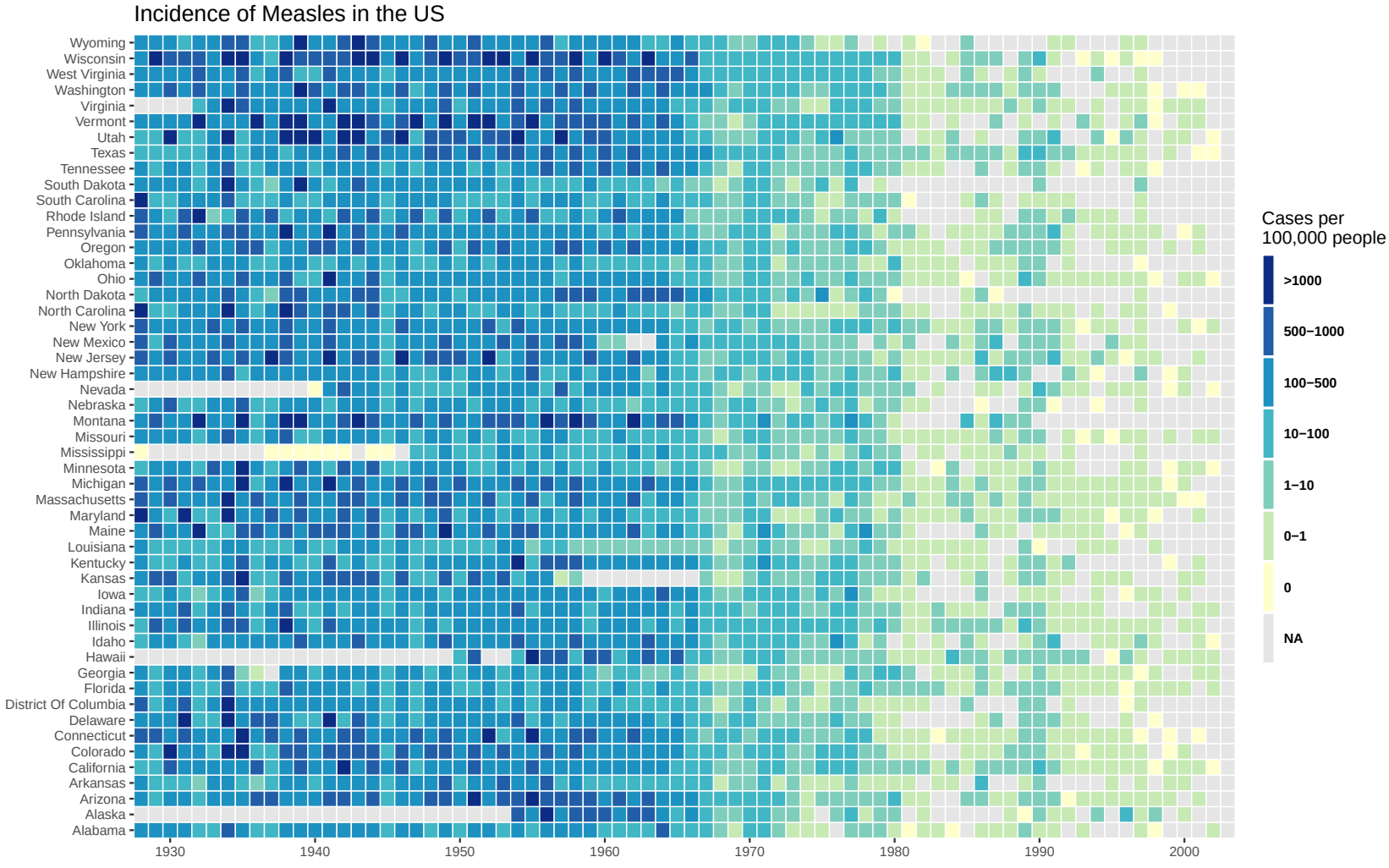


Heatmap exercise: scaling colour and fill

- Final product; see more details here:

```
hm <-
  ggplot(measles, aes(x = year, y = state, fill = countCat)) +
  # add border white colour of line thickness 0.25
  geom_tile(colour = "white", size = 0.25) +
  # scale fill with "YlGnBu" palette
  scale_fill_brewer(palette = "YlGnBu", direction = -1, na.value = "grey90") +
  # define new breaks on x-axis
  scale_x_discrete(expand = c(0, 0), breaks = seq(1930, 2000, 10)) +
  # remove extra space
  scale_y_discrete(expand = c(0, 0)) +
  # set a base size for all fonts
  theme_grey(base_size = 8) +
  # theme options
  theme(
    # bold font for legend text
    legend.text = element_text(face = "bold"),
    # set thickness of axis ticks
    axis.ticks = element_line(size = 0.4),
    # remove plot background
    plot.background = element_blank(),
    # remove plot border
    panel.border = element_blank(),
    # reshape the legend keys
    legend.key.height = grid::unit(0.8, "cm"),
    legend.key.width = grid::unit(0.2, "cm")
  ) +
  # legend title
  guides(fill = guide_legend(title = "Cases per\n100,000 people")) +
  # remove x and y axis labels; define title
  labs(x = NULL, y = NULL, title = "Incidence of Measles in the US")
```

Heatmap exercise: scaling colour and fill



ggplot2 extension packages

- With ggplot2 as the core package, an ecosystem of supporting packages have been developed

ggplot2 extension packages

- With ggplot2 as the core package, an ecosystem of supporting packages have been developed
  - See the gallery here

ggplot2 extension packages

- With ggplot2 as the core package, an ecosystem of supporting packages have been developed
  - See the gallery here
- I highlight several packages

ggplot2 extension packages

- With ggplot2 as the core package, an ecosystem of supporting packages have been developed
  - See the gallery here
- I highlight several packages
  - We've used: ggrepel, ggstance

ggplot2 extension packages

- With ggplot2 as the core package, an ecosystem of supporting packages have been developed
  - See the gallery here
- I highlight several packages
  - We've used: ggrepel, ggstance
  - gghighlight

ggplot2 extension packages

- With ggplot2 as the core package, an ecosystem of supporting packages have been developed
  - See the gallery here
- I highlight several packages
  - We've used: ggrepel, ggstance
  - gghighlight
  - ggridges

ggplot2 extension packages

- With ggplot2 as the core package, an ecosystem of supporting packages have been developed
  - See the gallery here
- I highlight several packages
  - We've used: ggrepel, ggstance
  - gghighlight
  - ggridges
  - cowplot