Regression III

Linear Model Visualization

Dave Armstrong

Goals of the Lecture

Discuss effective ways of testing and presenting effects in linear models

- Dummy variables
- Presenting and testing pairwise comparisons
- Quasi-variances
- Optimal Visual Testing Intervals
- Multiplicity Problem

Categorical Explanatory Variables

- *Linear* regression can be extended to accommodate categorical variables (*factors*) using *dummy variable regressors* (or *indicator variables*)
- Below a categorical variable is represented by a dummy regressor D, (coded 1 for one category, 0 for the other):

$$Y_i = \alpha + \beta X_i + \gamma D_i + \varepsilon_i$$

- This fits *two regression lines with the same slope but different intercepts*. In other words, the coefficient γ represents the constant separation between the two regression lines:
  - $Y_i = \alpha + \beta X_i + \gamma(0) + \varepsilon_i = \alpha + \beta X_i + \varepsilon_i$
  - $Y_i = \alpha + \beta X_i + \gamma(1) + \varepsilon_i = (\alpha + \gamma) + \beta X_i + \varepsilon_i$

Notes

Type notes here...

Categorical Explanatory Variables (2)

- In Figure (a) failure to account for a categorical variable (gender) does not produce significantly different results, either in terms of the intercept or the slope
- In Figure (b) the dummy regressor captures a significant difference in intercepts. More importantly, failing to include gender gives a negative slope for the relationship between education and income (dotted line) when in fact it should be positive for both men and women.

Notes

Type notes here...

Multi-Category Explanatory Variables

- Dummy regressors are easily extended to explanatory variables with more than two categories.
- A variable with m categories has $m-1$ regressors:
- As with the two-category case, one of the categories is a reference group (coded 0 for all dummy regressors).

Category	D_1	D_2
Blue Collar	1	0
Professional	0	1
White Collar	0	0

Notes

Type notes here…

Choosing the Reference Category

How do we choose the reference category?

- The choice of reference category is technically irrelevant - all choices produce exactly the same inferences.

Theory may suggest we compare to a particular category

- You should leave out the category in which you are most interested.

Notes

Type notes here...

Multi-Category Explanatory Variables (2)

- A model with one quantitative predictor (e.g., income) then takes the following form:

$$Y_i = \alpha + \beta X_i + \gamma_1 D_{i1} + \gamma_2 D_{i2} + \varepsilon_i$$

- This produces three *parallel regression lines*:

$$\begin{aligned} \text{Blue Collar: } Y_i &= (\alpha + \gamma_1) + \beta X_i + \varepsilon_i \\ \text{Professional: } Y_i &= (\alpha + \gamma_2) + \beta X_i + \varepsilon_i \\ \text{White Collar: } Y_i &= \alpha + \beta X_i + \varepsilon_i \end{aligned}$$

- Again, these lines are different only in terms of their intercepts

- i.e., the γ coefficients represent the constant distance between the regression lines. γ_1 and γ_2 are the differences between occupation types compared to `white collar`, when holding income constant.

Notes

Type notes here...

Dummy Variables in R

- in R, if categorical variables are properly specified as factors, dummy coding is done by default
- To specify a variable as a factor:

```
library(car)
data(Duncan)
contrasts(Duncan$type)
```

```
##      prof wc
## bc      0  0
## prof    1  0
## wc      0  1
```

- It is easy to change the reference category in R:

```
type2 <- relevel(Duncan$type, ref="wc")
contrasts(type2)
```

```
##      bc prof
## wc    0    0
## bc    1    0
## prof  0    1
```

Notes

Type notes here...

Effects of Dummy Variables in R (1)

```r
data(Duncan)
mod1<-lm(prestige~income+education+
    type, data=Duncan)
summary(mod1)
```

```
##
## Call:
## lm(formula = prestige ~ income + education + type, data = Duncan)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.890  -5.740  -1.754   5.442  28.972
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.18503    3.71377  -0.050  0.96051
## income        0.59755    0.08936   6.687 5.12e-08 ***
## education     0.34532    0.11361   3.040  0.00416 **
## typeprof     16.65751    6.99301   2.382  0.02206 *
## typewc      -14.66113    6.10877  -2.400  0.02114 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.744 on 40 degrees of freedom
## Multiple R-squared:  0.9131,    Adjusted R-squared:  0.9044
## F-statistic:   105 on 4 and 40 DF,  p-value: < 2.2e-16
```

Notes

Type notes here...

Effects of Dummy Variables in R (2)

- The `lm` output suggests that the categorical variable `type` has a strong effect on `prestige`.
- The incremental F-test confirms this finding

```
Anova(mod1)
```

```
## Anova Table (Type II tests)
##
## Response: prestige
##           Sum Sq Df F value    Pr(>F)
## income    4246.1  1 44.7201 5.124e-08 ***
## education  877.2  1  9.2388  0.004164 **
## type      3708.7  2 19.5302 1.208e-06 ***
## Residuals 3798.0 40
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notes

Type notes here...

The Reference Category Problem

Typically categorical variables in statistical models are reported in contrast to a reference category

- It is then difficult to make inferences about differences between categories aside from the reference category

Typical solutions:

- Refit the model with a different reference category
- Report the full variance-covariance matrix for the estimated parameters. A *standard error* between any two dummy regressors could then be easily calculated:

$$\mathrm{var}(aX + bY) = a^2\mathrm{var}(X) + b^2\mathrm{var}(Y) + 2ab\mathrm{cov}(X, Y)$$

For a categorical variable with p levels, this would require reporting $\frac{p(p-1)}{2}$ covariances, making it difficult to do so if only because of space constraints.

Notes

Type notes here...

Calculating Different Contrasts

It is straightforward to calculate all pairwise comparisons.

```r
data(Ornstein, package="carData")
omod <- lm(interlocks ~
    nation + sector + log2(assets),
    data=Ornstein)
library(multcomp)
summary(glht(omod, linfct=mcp(nation = "Tukey")))
```

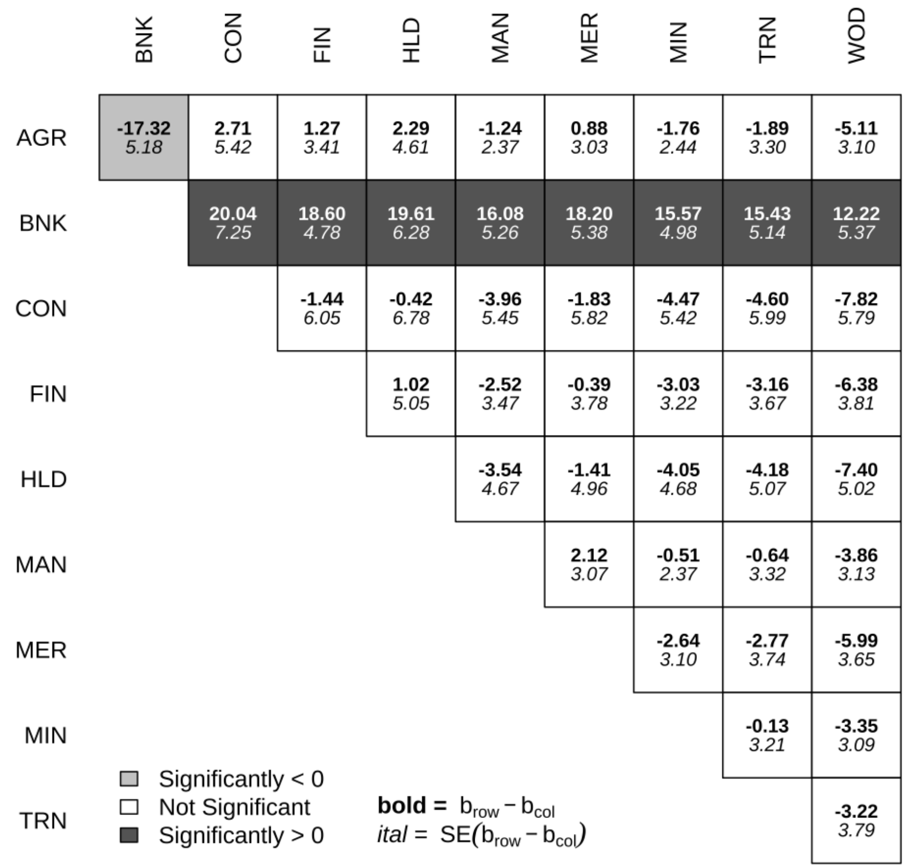
```
##
##      Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: lm(formula = interlocks ~ nation + sector + log2(assets), data = Ornstein)
##
## Linear Hypotheses:
##                 Estimate Std. Error t value Pr(>|t|)
## OTH - CAN == 0    -3.053      3.087  -0.989    0.745
## UK - CAN == 0     -5.329      3.071  -1.735    0.294
## US - CAN == 0     -8.491      1.717  -4.944   <0.001 ***
## UK - OTH == 0     -2.276      3.865  -0.589    0.932
## US - OTH == 0     -5.438      3.018  -1.802    0.262
## US - UK == 0      -3.162      3.028  -1.044    0.711
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Notes

Type notes here...

factorplot

```r
library(factorplot)
ofp <- factorplot(
    omod,
    factor.variable="sector")
plot(ofp)
```



	BNK	CON	FIN	HLD	MAN	MER	MIN	TRN	WOD
AGR	**-17.32** *5.18*	**2.71** *5.42*	**1.27** *3.41*	**2.29** *4.61*	**-1.24** *2.37*	**0.88** *3.03*	**-1.76** *2.44*	**-1.89** *3.30*	**-5.11** *3.10*
BNK		**20.04** *7.25*	**18.60** *4.78*	**19.61** *6.28*	**16.08** *5.26*	**18.20** *5.38*	**15.57** *4.98*	**15.43** *5.14*	**12.22** *5.37*
CON			**-1.44** *6.05*	**-0.42** *6.78*	**-3.96** *5.45*	**-1.83** *5.82*	**-4.47** *5.42*	**-4.60** *5.99*	**-7.82** *5.79*
FIN				**1.02** *5.05*	**-2.52** *3.47*	**-0.39** *3.78*	**-3.03** *3.22*	**-3.16** *3.67*	**-6.38** *3.81*
HLD					**-3.54** *4.67*	**-1.41** *4.96*	**-4.05** *4.68*	**-4.18** *5.07*	**-7.40** *5.02*
MAN						**2.12** *3.07*	**-0.51** *2.37*	**-0.64** *3.32*	**-3.86** *3.13*
MER							**-2.64** *3.10*	**-2.77** *3.74*	**-5.99** *3.65*
MIN								**-0.13** *3.21*	**-3.35** *3.09*
TRN									**-3.22** *3.79*

▨ Significantly < 0
□ Not Significant
■ Significantly > 0

bold = $b_{row} - b_{col}$
ital = $SE(b_{row} - b_{col})$

Notes

Type notes here...

sigplot

I also recently developed a different solution that's based on the D3.js library.

- One way of using the function is by giving it a model object (that works with the `ggpredict()` function) and a model term.
- Another way of interacting with it is by giving it output from a Bayesian model.
  - This could be output generated by something like BUGS, JAGS or Stan.
  - It could also be data generated by parametric bootstrap from models estimated in the Frequentist contexs. In this case, the model would be assuming flat priors over the support of the model parameters.

This plot is interactive - so doesn't translate as well in print, but scales better than the `factorplot()` output.

- install with `remotes::install_github("davidaarmstrong/daviz")`

Notes

Type notes here...

sigplot 2

```r
library(daviz)
library(r2d3)
omod2 <- lm(interlocks ~
    nation + sector,
    data=Ornstein)

sigd3(omod2, "sector",
      fname="sector_plot.html",
      return_iFrame = TRUE)
```

Notes

Type notes here...

Quasi-Variances

Assuming that the dummy variables d_j represent the $j = 0, \ldots, J$ categories of the variable x, we could estimate the model: $y = b_0 + b_1 d_1 + b_2 d_2 + \ldots + b_j d_j + \mathbf{Zg} + e$.

- To find the p-value for the comparison of b_1 to b_2, we would need to calculate:

$$t_{1,2} = \frac{b_1 - b_2}{\sqrt{var(b_1) + var(b_2) - 2cov(b_1, b_2)}}$$

or more generally:

$$t_{j,k} = \frac{b_j - b_k}{\sqrt{var(b_j) + var(b_k) - 2cov(b_j, b_k)}}$$

Notes

Type notes here...

Quasi-variances (2)

Imagine that we could replace:

$$t_{j,k} = \frac{b_j - b_k}{\sqrt{var(b_j) + var(b_k) - 2cov(b_j, b_k)}}$$

with

$$t_{j,k} \approx \frac{b_j - b_k}{\sqrt{q_j + q_k}}$$

The q terms are the quasi-variances.

- They can be presented along side (or instead of) conventional standard errors.

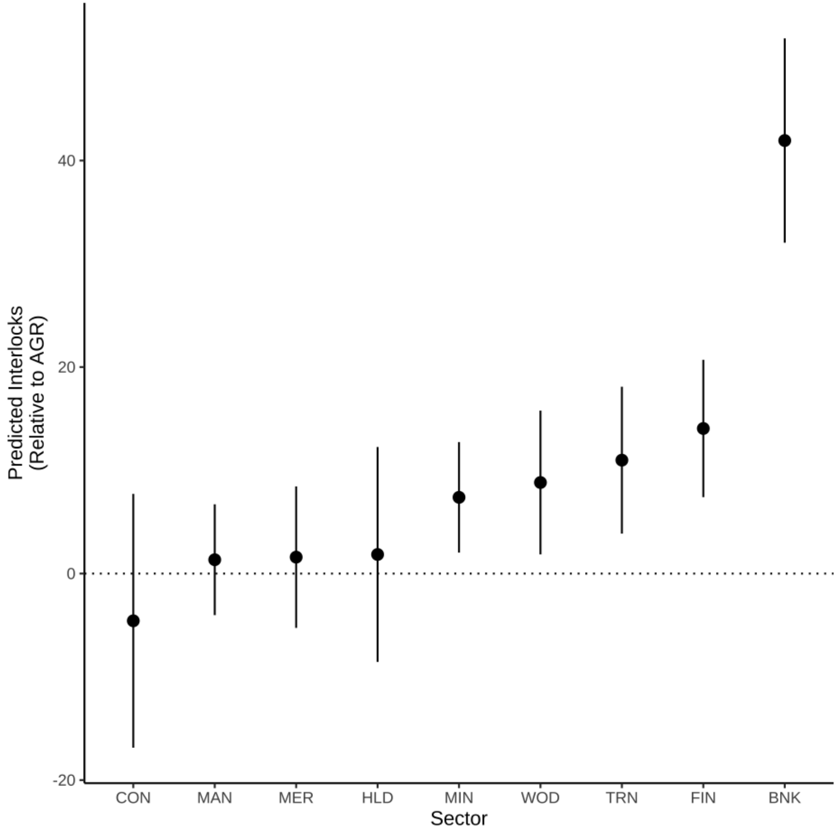
Notes

Type notes here...

Optimal Visual Testing Confidence Intervals

Consider the Ornstein model example from above. A static effect plot would look as follows:

```
b <- omod2$coef[5:13]
v <- vcov(omod2)[5:13,5:13]
plot_dat <- tibble(
  sector = factor(2:10,
                  levels=1:10,
                  labels=levels(Ornstein$sector)),
  b = unname(b),
  se = sqrt(diag(v)),
  lwr = b - qt(.975,
               omod2$df.residual)*se,
  upr = b + qt(.975,
               omod2$df.residual)*se)
ggplot(plot_dat, aes(x=reorder(sector, b, mean),
                     y=b,
                     ymin=lwr, ymax=upr)) +
  geom_pointrange() +
  geom_hline(yintercept=0, linetype=3) +
  theme_classic() +
  labs(x="Sector",
       y="Predicted Interlocks\n(Relative to AGR)")
```



Notes

Type notes here...

Optimal Visual Testing Confidence Intervals (2)

Why use 95% confidence intervals?

- Displays the non-rejectable null hypothesis values for the parameter of interest.
- Manifestly unhelpful if we want to use the confidence intervals for testing hypotheses about differences across parameters.

Some have suggested 84% confidence intervals as a good alternative.

- 84% works more often than 95%, but not always.

Why not just optimize this - find the best confidence level such that whether confidence intervals overlap represents to the greatest degree possible the actual testing results?

Notes

Type notes here...

Implementation

To use the function, you'll need to install the `psre` package from my github:

```
remotes::install_github("davidaarmstrong/psre")
```

You can then find the optimal visual testing confidence intervals with:

```
library(psre)
o <- optCL(omod2,
           varname="sector",
           add_ref=TRUE,
           grid_range=c(.5,.99))
```

```
o[c("opt_levels", "opt_errors", "lev_errors", "err_dat")]
```

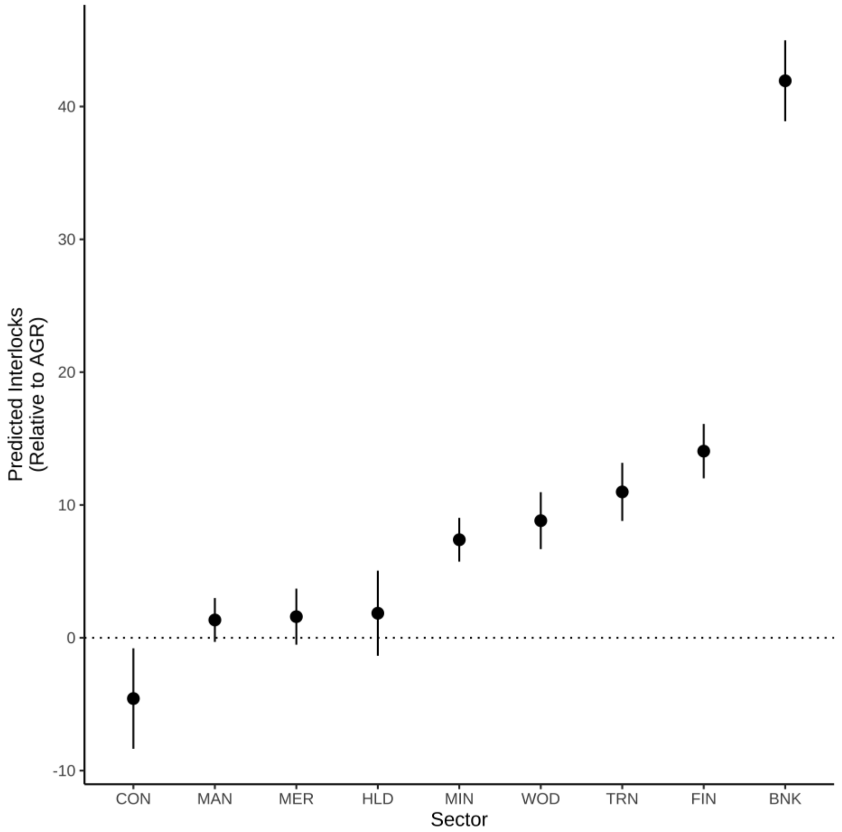
```
## $opt_levels
## [1] 0.7276768
##
## $opt_errors
## [1] 0.02222222
##
## $lev_errors
## [1] 0.2
##
## $err_dat
## # A tibble: 1 x 20
## # Rowwise:
##    cat1  cat2    b1    b2    v1    v2   vt1   vt2 cov12 comp_var  diff
##   <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>    <dbl> <dbl>
## 1     3     8 -4.57  7.38  38.9  7.37  38.9  7.37  4.19     37.9 -12.0
## # … with 8 more variables: p <dbl>, lb1 <dbl>, ub1 <dbl>, lb2 <dbl>, ub
## #   sig <dbl>, olap <dbl>, crit <dbl>
```

Notes

Type notes here...

Implementation (2)

```
plot_dat <- tibble(
  sector = factor(2:10,
                levels=1:10,
                labels=levels(Ornstein$sector)),
  b = unname(b),
  se = sqrt(diag(v)),
  lwr = b - qt(mean(o$opt_levels),
             omod2$df.residual)*se,
  upr = b + qt(mean(o$opt_levels),
             omod2$df.residual)*se)
ggplot(plot_dat, aes(x=reorder(sector, b, mean),
                     y=b,
                     ymin=lwr, ymax=upr)) +
  geom_pointrange() +
  geom_hline(yintercept=0, linetype=3) +
  theme_classic() +
  labs(x="Sector",
       y="Predicted Interlocks\n(Relative to AGR)")
```



NB: Optimal Visual Testing Intervals used ($\approx 73\%$) to identify 95% tests. Even though the construction and mining intervals do not overlap, their difference is not

Notes

Type notes here...

Multiplicity Problem

Usually, we choose to control Type I error rates when we test hypotheses, by evaluating a hypothesis, H, at a pre-specified significance level, α.

- Assume two hypotheses, $H = \{H_1, H_2\}$, both of which are true, and we are testing them independently, each at level $\alpha = 0.05$.
- The probability of not rejecting either hypothesis is $(1 - \alpha)^2 = 0.9025$
    - The probability of falsely rejecting at least one test is $1 - (1 - \alpha)^2 = 0.0975$,
- The probability of falsely rejecting at least one test among a set of m tests $H = \{H_1, \ldots, H_m\}$ is $1 - (1 - \alpha)^m$.

Notes

Type notes here...

Actual Type I Error Rates with Multiple Testing

Notes

Type notes here...

Controlling for Multiple Testing

Hypotheses	Not Rejected	Rejected	Total
True	U	V	m_0
False	T	S	$m - m_0$
Total	W	R	m

Notes

Type notes here...

Extending Type I Error to Multiple Tests

- Per-comparison Error Rate: $\mathrm{PCER} = \frac{E(V)}{m}$ is the expected proportion of Type I errors among m comparisons. If tested independently, $\mathrm{PCER} = \frac{\alpha m_0}{m} \leq \alpha$
- Family-wise Error Rate: $\mathrm{FWER} = P(V > 0)$ is the probability of committing at least one Type I error.
  - Most commonly used measure, good when number of comparisons is moderate or where strong evidence is needed.
  - FWER approaches 1 as number of comparisons increases without a multiplicity adjustment
  - FWER reduces to the Type I error rate α when $m = 1$
  - A less strict version $\mathrm{gFWER} = P(V > k)$, where the probability of making some small number (k) of Type I errors is acceptable.
- False Discovery Rate: If $Q = \frac{V}{R}$, the proportion of false rejections among all rejections. $FDR = E\left(\frac{V}{R} R > 0\right) P(R > 0)$. Extensions here abound and is an area of active research.

Notes

Type notes here...

Strong vs. Weak Control

- Control of Type I error rate is considered *weak* if the Type I error rate is controlled only under the global null hypothesis (i.e., assuming $H_1, \dots, H_m$ are all true)
- Control of Type I error rate is considered *strong* if the Type I error rate is controlled under any configuration of true null hypotheses (except for the null set).
- Controlling FWER in the strong sense is the most stringent (i.e., conservative) test.

Notes

Type notes here...

Single-step vs. Stepwise Procedures

- In single-step procedures, the information about rejecting or not rejecting one hypothesis does not enter into the decision for another. (Example: Bonferroni)
- In stepwise procedures (different from and decidedly less controversial than "stepwise regression"), hypotheses are ordered (in a potentially data-dependent fashion) and either:
    - In a step-down procedure, hypotheses are rejected until the first non-rejection and then all others are retained. (Example: Holm)
    - In a setp-up procedure, hypotheses are retained until the first rejection then all others are rejected. (Example: Hochberg)

Notes

Type notes here...

Adjusted p-values

p-values can be calculated adjusting for any multiple comparison procedure mentioned above. The adjusted p-value for test i (call them q_i) take the form:

$$q_i = \inf \{\alpha \in (0,1) | H_i \text{ is rejected at level } \alpha\}$$

- To control FWER in the strong sense, Bonferroni (single-step), Holm (step-down) and Hochberg (step-up) are options, though Holm's method is known to dominate Bonferroni's under a set of minimally restrictive assumptions.

- To control FDR, Benjamini-Hochberg (BH) works under the assumption of independent tests and Benjamini-Yekuteli (BY) works when independence cannot be assumed.

Notes

Type notes here...

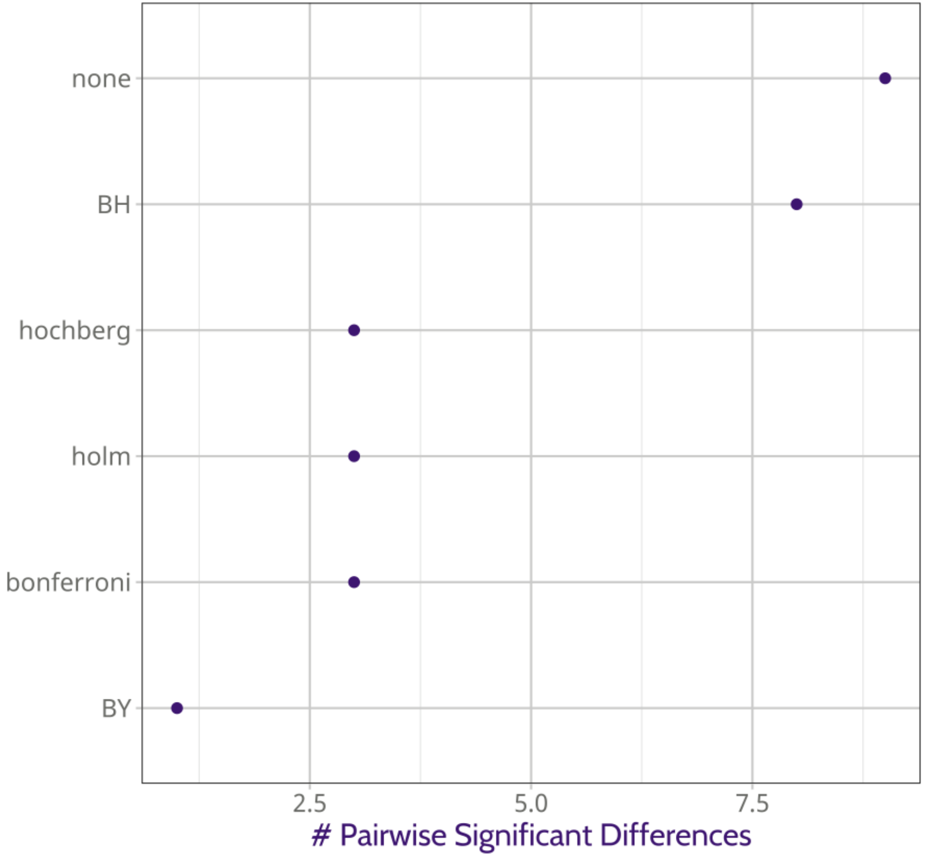
Multiplicity Correction

- Above, we tested 45 hypotheses simultaneously, so 5\% (or ≈ 2) could will be significant by chance''.
  - ○ The Holm correction sets the α for the entire set of tests equal to the desired rate by setting the α for each individual test to $\frac{\alpha}{n-i+1}$ where n is the number of comparisons and i is the rank-order of the p-value. Compare this to the Bonferroni p-value of $\frac{\alpha}{n}$.

Notes

Type notes here...

Different Corrections



Notes

Type notes here...

Factorplot Summary

```
summary(ofp2)
```

```
##     sig+ sig- insig
## AGR    0    1     8
## BNK    3    0     6
## CON    0    0     9
## FIN    0    1     8
## HLD    0    0     9
## MAN    0    0     9
## MER    0    1     8
## MIN    0    0     9
## TRN    0    0     9
## WOD    0    0     9
```

```
print(ofp2, sig=T)
```

```
##             Difference    SE p.val
## AGR - BNK    -17.323 5.185 0.042
## BNK - FIN     18.597 4.784 0.006
## BNK - MER     18.203 5.377 0.037
```

Notes

Type notes here...

OVT Adjustment

```r
o2 <- optCL(omod2,
           varname="sector",
           add_ref=TRUE,
           grid_range=c(.5,.99),
           adjust="holm")
```

```r
o2[c("opt_levels", "opt_errors", "lev_errors", "err_dat")]
```

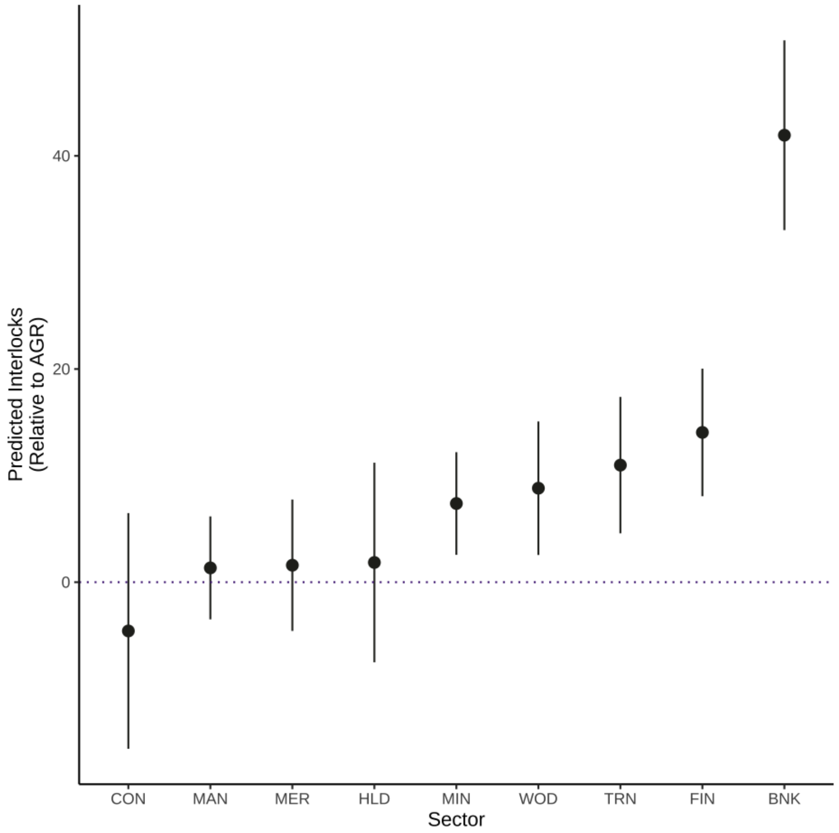
```
## $opt_levels
## [1] 0.9500000 0.9504040 0.9553535 0.9603030 0.9900000
## 
## $opt_errors
## [1] 0.06666667
## 
## $lev_errors
## [1] 0.06666667
## 
## $err_dat
## # A tibble: 3 x 20
## # Rowwise: 
##    cat1  cat2    b1    b2    v1    v2   vt1   vt2 cov12 comp_var   dif
##   <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>    <dbl>  <dbl
## 1     1     8     0  7.38     0  7.37     0  7.37     0     7.37  -7.38
## 2     1     9     0 11.0      0 13.0      0 13.0      0    13.0  -11.0
## 3     1    10     0  8.82     0 12.5      0 12.5      0    12.5   -8.82
## # … with 8 more variables: p <dbl>, lb1 <dbl>, ub1 <dbl>, lb2 <dbl>, ub
## #   sig <dbl>, olap <dbl>, crit <dbl>
```

Notes

Type notes here...

Implementation (2)

```
plot_dat <- tibble(
  sector = factor(2:10,
                  levels=1:10,
                  labels=levels(Ornstein$sector)),
  b = unname(b),
  se = sqrt(diag(v)),
  lwr = b - qt(mean(o2$opt_levels),
               omod2$df.residual)*se,
  upr = b + qt(mean(o2$opt_levels),
               omod2$df.residual)*se)
ggplot(plot_dat, aes(x=reorder(sector, b, mean),
                     y=b,
                     ymin=lwr, ymax=upr)) +
  geom_pointrange() +
  geom_hline(yintercept=0, linetype=3) +
  theme_classic() +
  labs(x="Sector",
       y="Predicted Interlocks\n(Relative to AGR)")
```



NB: Optimal Visual Testing Intervals used ($\approx 96\%$) to identify 95% tests. Even though the AGR does not overlap with MIN, TER or WOD, they are not statistically

Notes

Type notes here...

Tomorrow

- Interactions
- Relative Importance