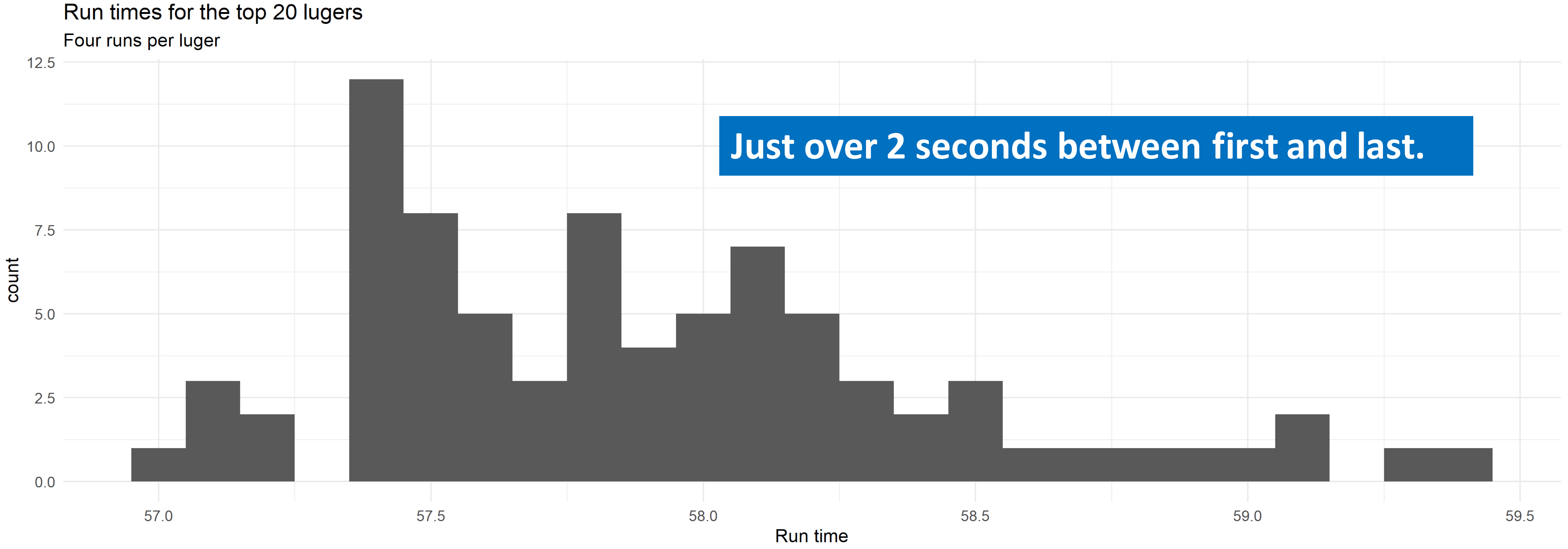
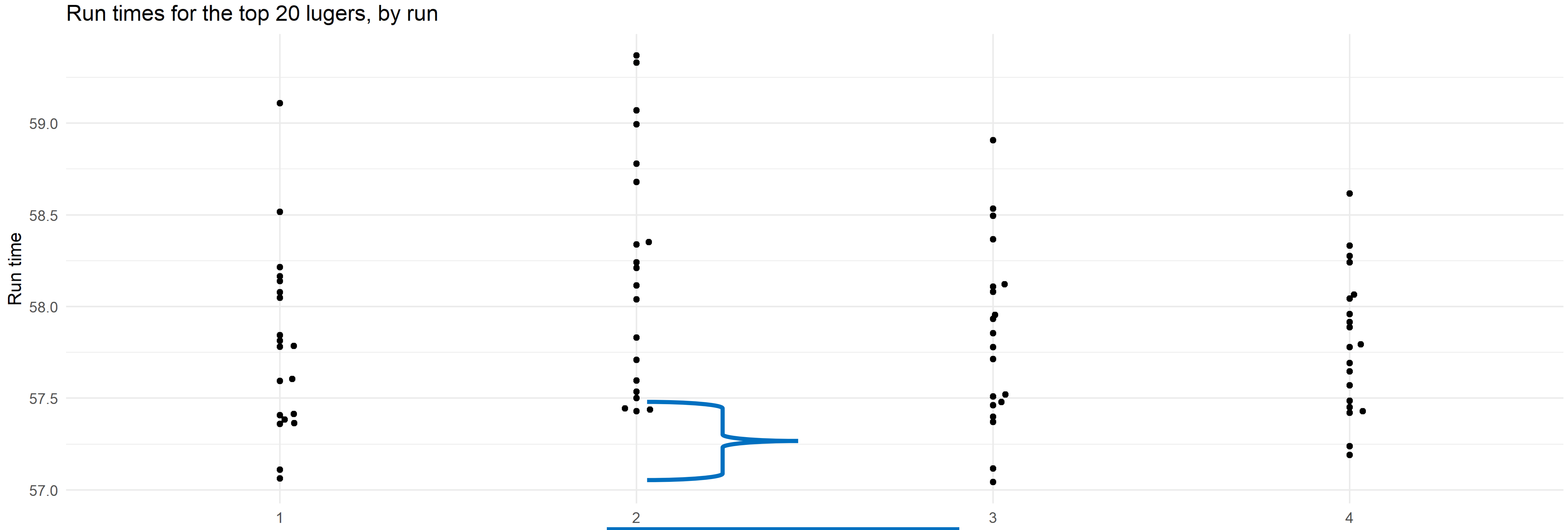
Higher Learning

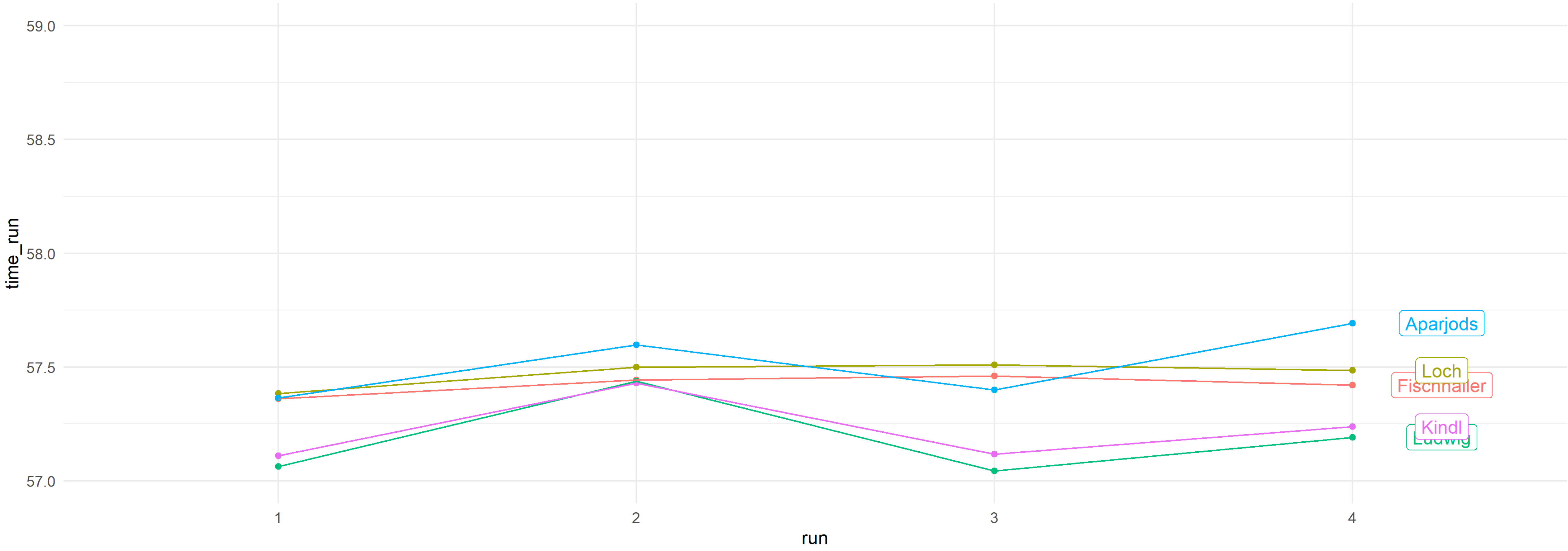
Hierarchical linear models



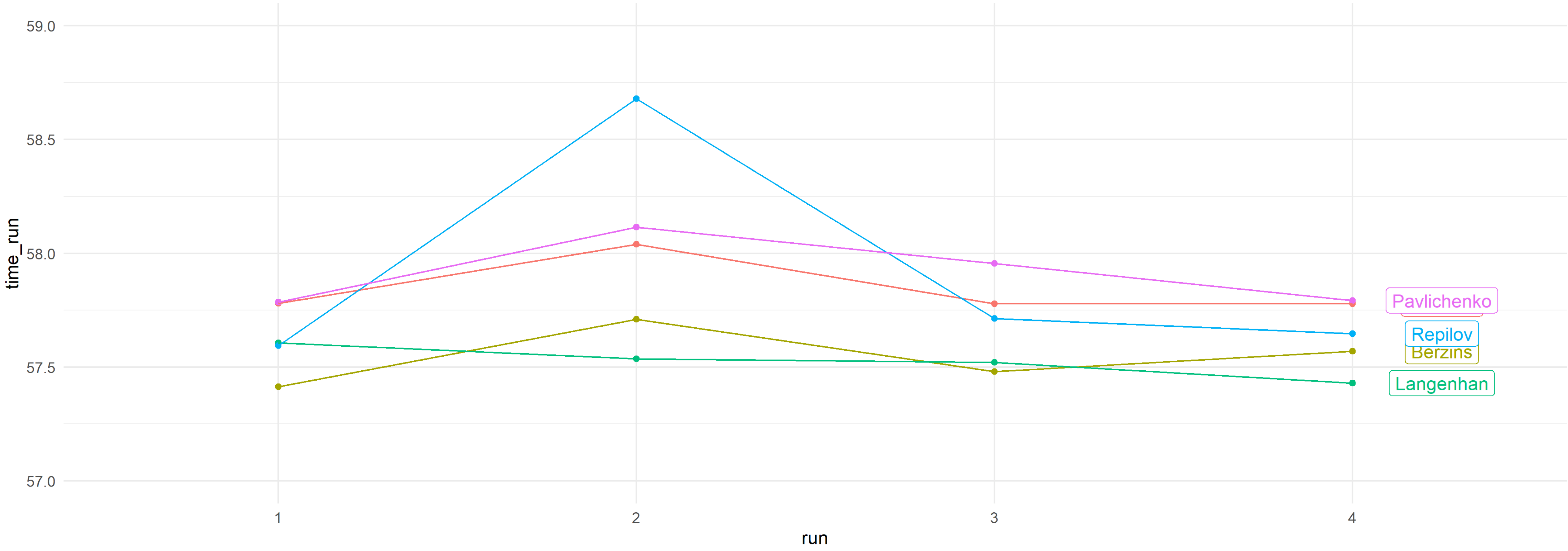
Luge times follow a skewed distribution. Low upside, higher downside.



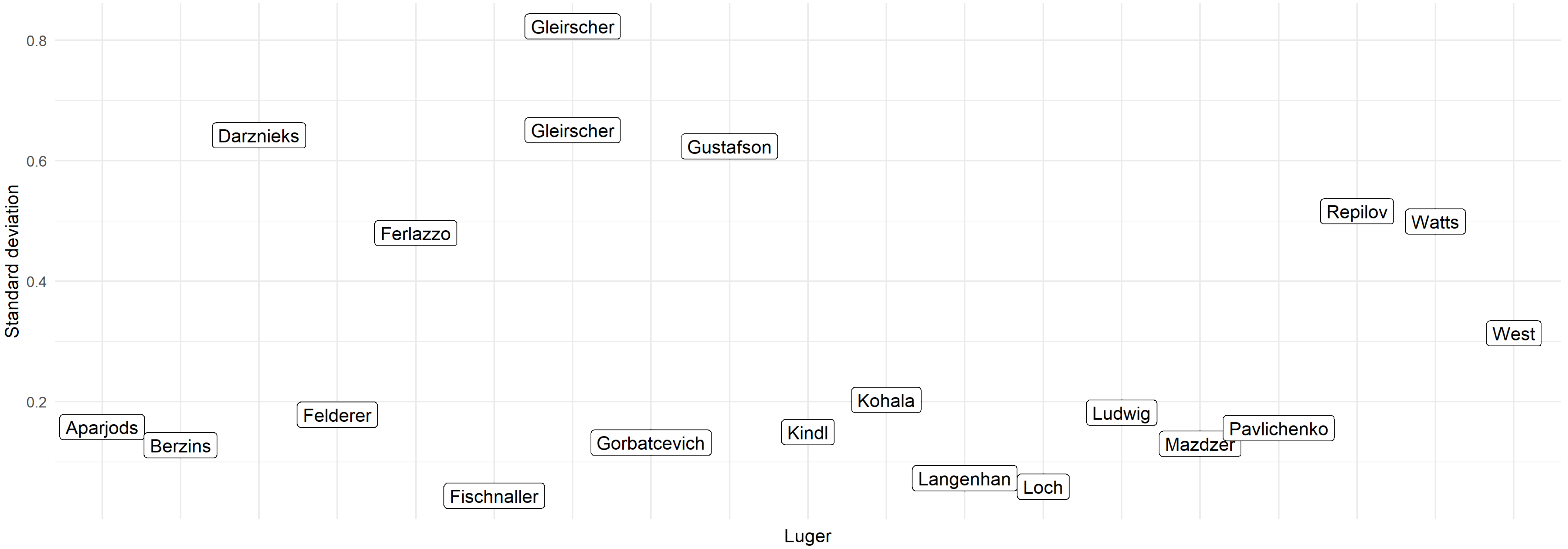
Everyone was a bit slower for run 2.



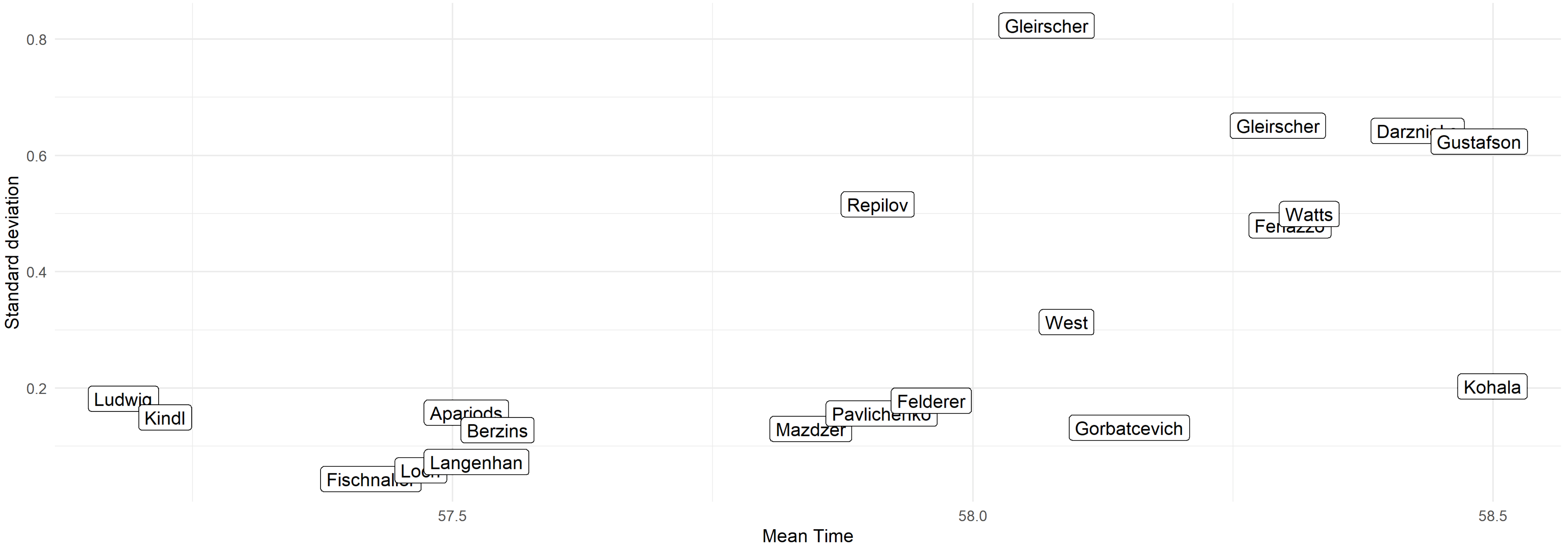
The top 5 finishers are pretty consistent!



Still consistent, but a bit more variation.

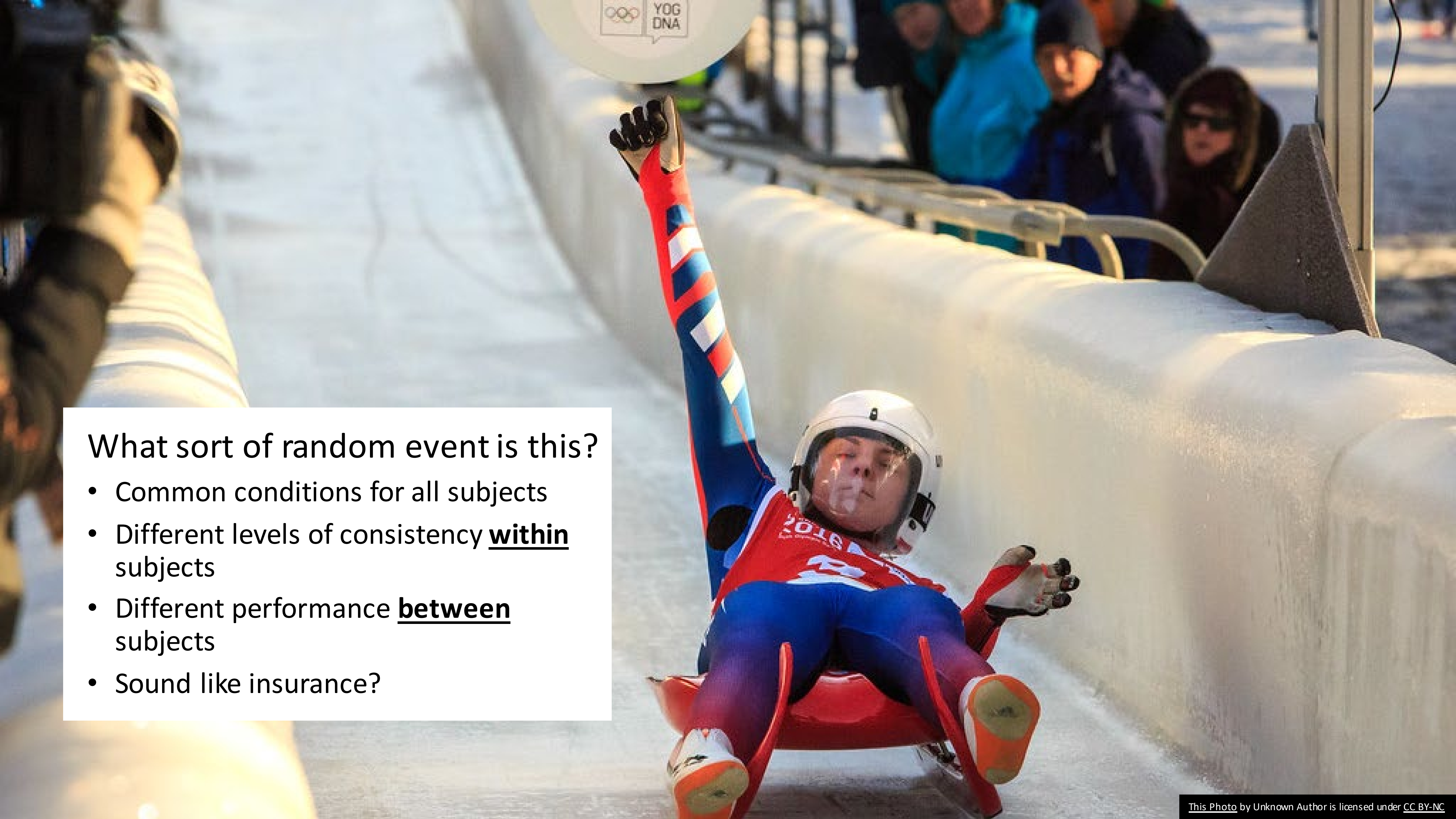


Standard deviations don’t deviate that much!



The top finishers tends to have low standard deviation. Consistent runs -> no mistakes -> faster time, though other factors also at play.

Put differently, low standard deviation means that I can more easily predict how fast they will perform.



What sort of random event is this?

- Common conditions for all subjects
- Different levels of consistency **within** subjects
- Different performance **between** subjects
- Sound like insurance?

How can we predict run 4 using runs 1-3?

- Use the sample average for all athletes?
- Use the sample average for the individual athlete?
- ¿Por qué no los dos? Why not both?

If you think this sounds like credibility that’s only because you’ve been paying attention.

Here’s how this looks using math …

$$\widehat{y_{ij}^{p}} = \widehat{\beta_0^{p}}$$

One estimator to rule them all

The estimators have the same symbols and subscripts, NOT the same values!

$$\widehat{y_{ij}^{I}} = \widehat{\beta_j^{I}}$$

$$\widehat{y_{ij}^{m}} = \widehat{\beta_0^{m}} + \widehat{\beta_j^{m}}$$

A little of this, a little of that

$$\widehat{y_{ij}^{m}} = \widehat{\beta_0^{m}} + \widehat{\beta_j^{m}}$$

$$\widehat{y_{ij}^{m}} = Z * \widehat{\beta_j^{I}} + (1 - Z) * \widehat{\beta_0^{p}}$$

$$Z_j \approx \frac{n_j}{n_j + {\sigma_j^2}/{\sigma_g^2}}$$

This is greatest accuracy credibility!

$$Z_j \approx \frac{n_j}{n_j + \sigma_j^2 / \sigma_g^2}$$

- As n_j increases, so does the credibility
- As the variance with group j increases, the credibility decreases
- As the variance across the groups increases, the credibility increases

Often in practice, actuaries tend to focus on n. This is important, but if you are not considering the other two elements, you are ignoring a substantial amount of information!

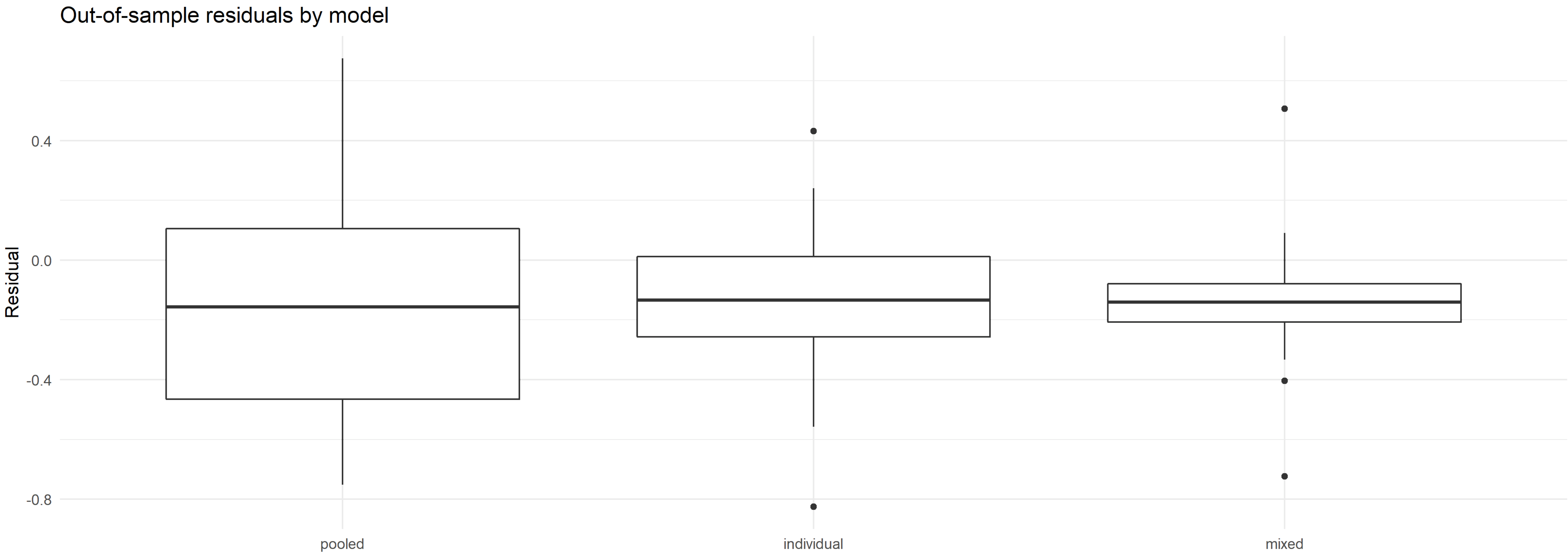
… and here's how this looks in R

```
fit_pooled <- lm(
  time_run ~ 1
  , data = tbl_train
)

fit_individual <- lm(
  time_run ~ 0 + name_full
  , data = tbl_train
)

fit_mixed <- lme4::lmer(
  time_run ~ 1 + (1 | name_full)
  , data = tbl_train
)
```

Could have used a 1 here. Then the coefficients would have been differences from some base athlete.

Wacky new notation here

More on formulas in a bit. If you're not familiar with R, hang tight.



Note that the mean residual is NOT zero. Out-of-sample performance is not guaranteed to be unbiased.

	Mean absolute error	Root mean squared error
Pooled	0.334	0.401
Individual	0.217	0.299
Mixed	0.201	0.263

Single summaries show that out of sample model performance for the mixed model beats the pooled and individual models.

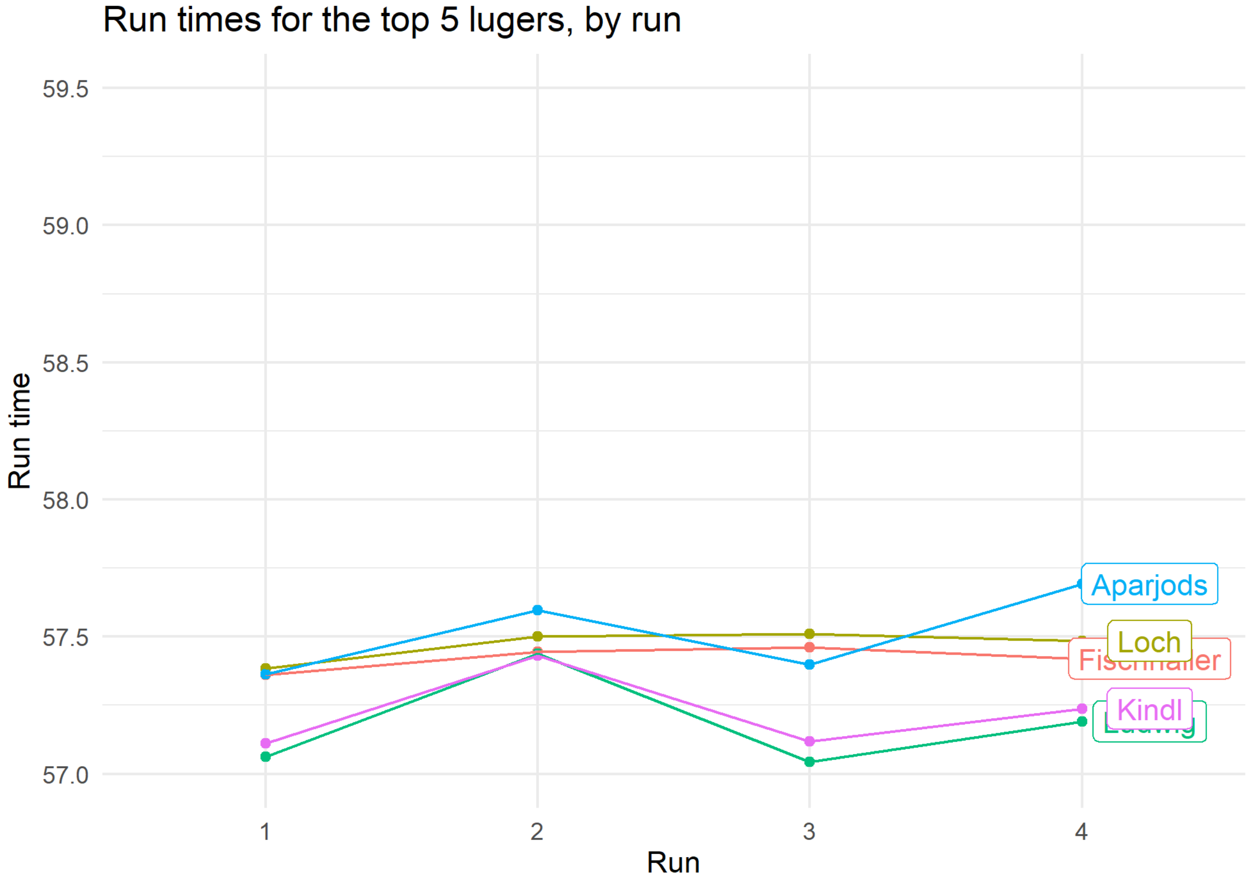
Perhaps we just got lucky?

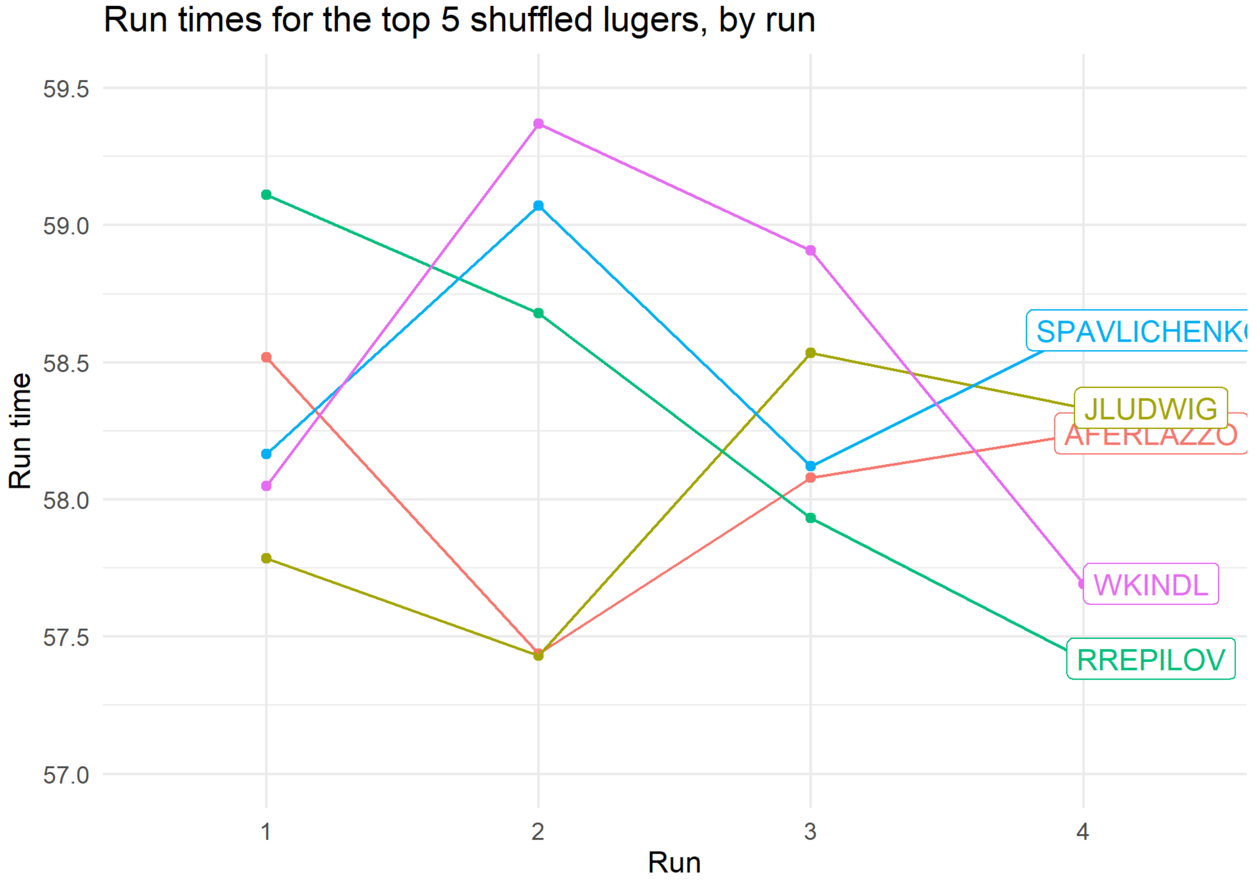
- Does the act of splitting into subsamples guarantee better model performance?
- If so, we could split into any subgroups of the same size and get comparable benefits.
- If subgroups are the cause, then it’s simply statistical hokum, there’s no wisdom needed before we a model, nor gained afterwards.

Shuffle the competitors

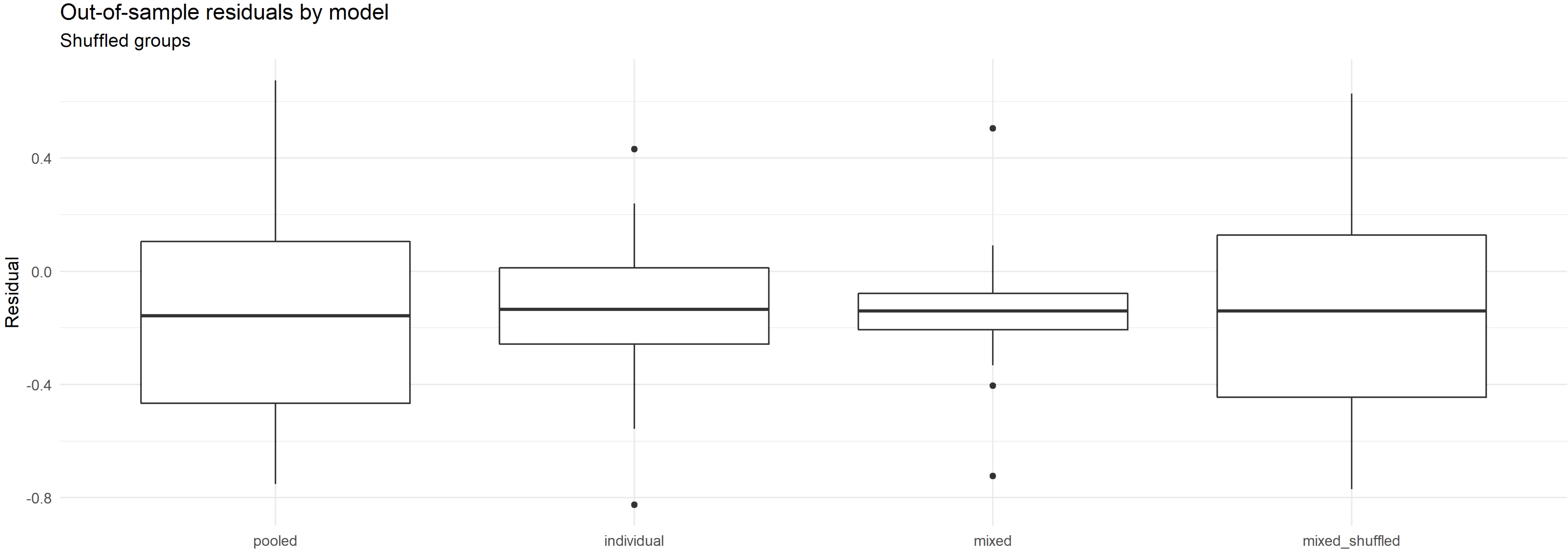
	noc	name_last	name_full	name_first	run	sex	time_run	shuffled
1	LAT	Darznieks	ADARZNIEKS	Arturs	1	men	58.166	SPAVLICHENKO
2	LAT	Darznieks	ADARZNIEKS	Arturs	2	men	59.370	WKINDL
3	LAT	Darznieks	ADARZNIEKS	Arturs	3	men	57.932	RREPILOV
4	AUS	Ferlazzo	AFERLAZZO	Alexander	1	men	58.216	SKOHALA
5	AUS	Ferlazzo	AFERLAZZO	Alexander	2	men	58.994	AGORBATCEVICH
6	AUS	Ferlazzo	AFERLAZZO	Alexander	3	men	58.122	SPAVLICHENKO
7	ROC	Gorbatcevich	AGORBATCEVICH	Aleksandr	1	men	58.139	RWATTS
8	ROC	Gorbatcevich	AGORBATCEVICH	Aleksandr	2	men	58.339	DGLEIRSCHER
9	ROC	Gorbatcevich	AGORBATCEVICH	Aleksandr	3	men	58.080	AFERLAZZO

If you think that shuffling the names is a big deal, you're saying that the credibility of three observations is measurable and meaningful!





The shuffled competitors show higher variation. Recall what this does to Z.



Shuffling the names turns out to be a pretty big deal!

	Mean absolute error	Root mean squared error
Pooled	0.334	0.401
Individual	0.217	0.299
Mixed	0.201	0.263
Shuffled	0.337	0.404

Out of sample model performance for the mixed model beats the pooled and individual models.

How did we do that?

Hello
my name is

- Fixed and random effects
- Hierarchical
- Multilevel
- Linear mixed

Traditional linear model

$$y_i = X_i\beta + \epsilon$$

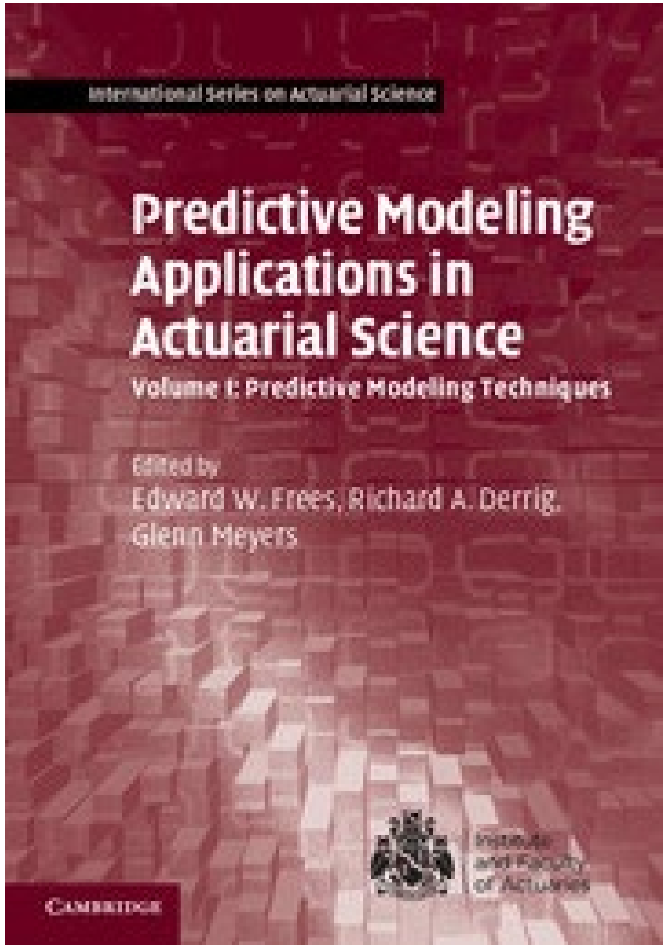
$$\epsilon \sim N(0, \sigma^2)$$

Mixed model

$$y_{ij} = X_i\beta + Z_j u + \epsilon$$

$$\epsilon \sim N(0, \sigma^2)$$

$$u \sim N(0, D)$$



Notation is slightly altered from that in "Predictive Modeling Applications in Actuarial Science

Fixed effects – same coefficients across the sample

$$y_{ij} = X_i\beta + Z_j u + \epsilon$$

Random effects – coefficients vary by group

Formulas in R

Formula	What it means
`y ~ 1`	Response is best estimated by the sample mean
`y ~ 1 + x`	Response is best estimated by a deviation from the mean which depends on a scaled difference of a predictor
`y ~ 1 + x1 + x2`	Response is best estimated by a deviation from the mean which depends on a scaled difference of two predictors
`y ~ 1 + (1 | group)`	Adjust the sample mean based on characteristics of the group
`y ~ 1 + (x | group)`	The constant is fixed for the whole population, but the rate of adjustment depends on the group.
`y ~ 1 + (1 + x | group)`	Both the constant and the rate of adjustment depend on the group.

The forgoing formulas are appropriate for the R package `lme4`. The formula interface for `nlme` is different!

Once again, here's how to do this in R.

```r
fit_pooled <- lm(
  time_run ~ 1
  , data = tbl_train
)

fit_individual <- lm(
  time_run ~ 0 + name_full
  , data = tbl_train
)

fit_mixed <- lme4::lmer(
  time_run ~ 1 + (1 | name_full)
  , data = tbl_train
)
```

And how to get the output

```
fit_mixed <- lme4::lmer(
  time_run ~ 1 + (1 | name_full)
  , data = tbl_train
)

summary(fit_mixed)

lme4::fixef(fit_mixed)
lme4::ranef(fit_mixed)
```

Loss reserving

Before we begin

- Loss reserving is a linear model. Repeat it until you believe it, too.
- Loss reserving is a linear model.
- Great background:
  - "Unbiased Loss Development Factors" – by Daniel Murphy
  - "Chain-Ladder Bias: Its Reason and Meaning" – by Leigh Halliwell
  - "Best Estimates for Reserves" – by Glen Barnett and Ben Zehnwirth
  - "Testing the Assumptions of Age-to-Age Factors" – by Gary Venter
- Loss reserving is a linear model.

$$\hat{y} = 0 + \beta x$$

Most reserving actuaries

$$\hat{y} = \beta_0 + \beta_1 x$$

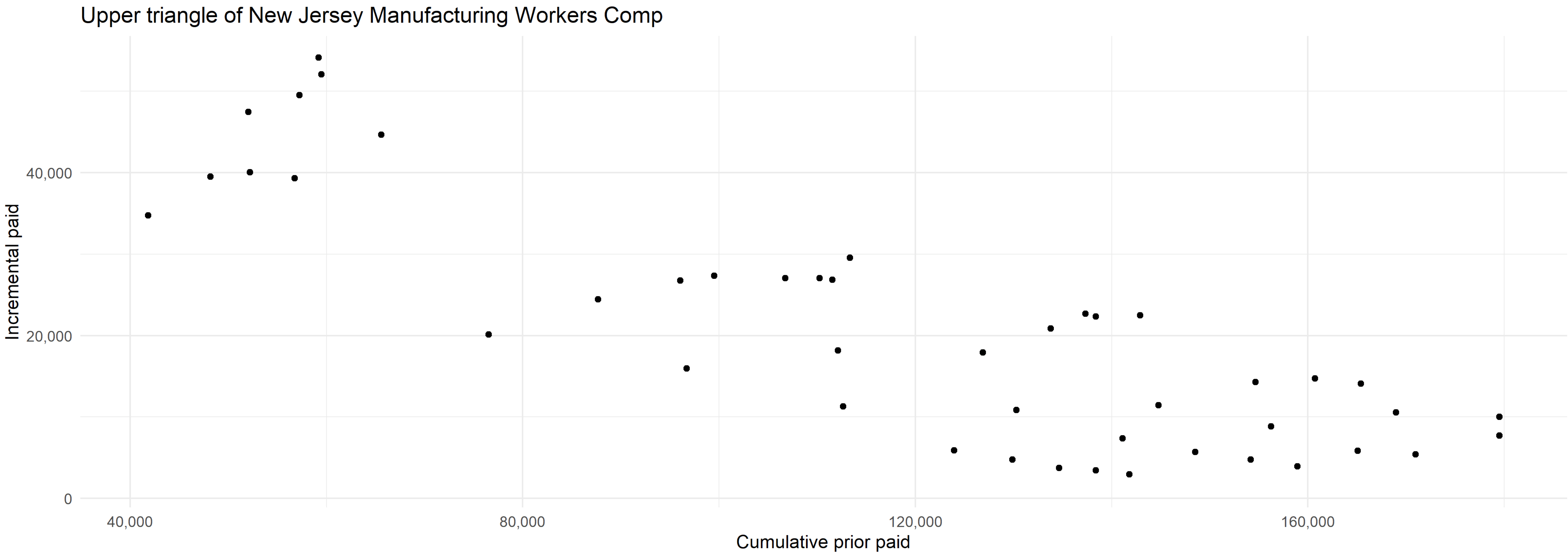
Clever reserving actuaries

$$\hat{y} = \beta_0 + \beta_1 x + Z_i x$$

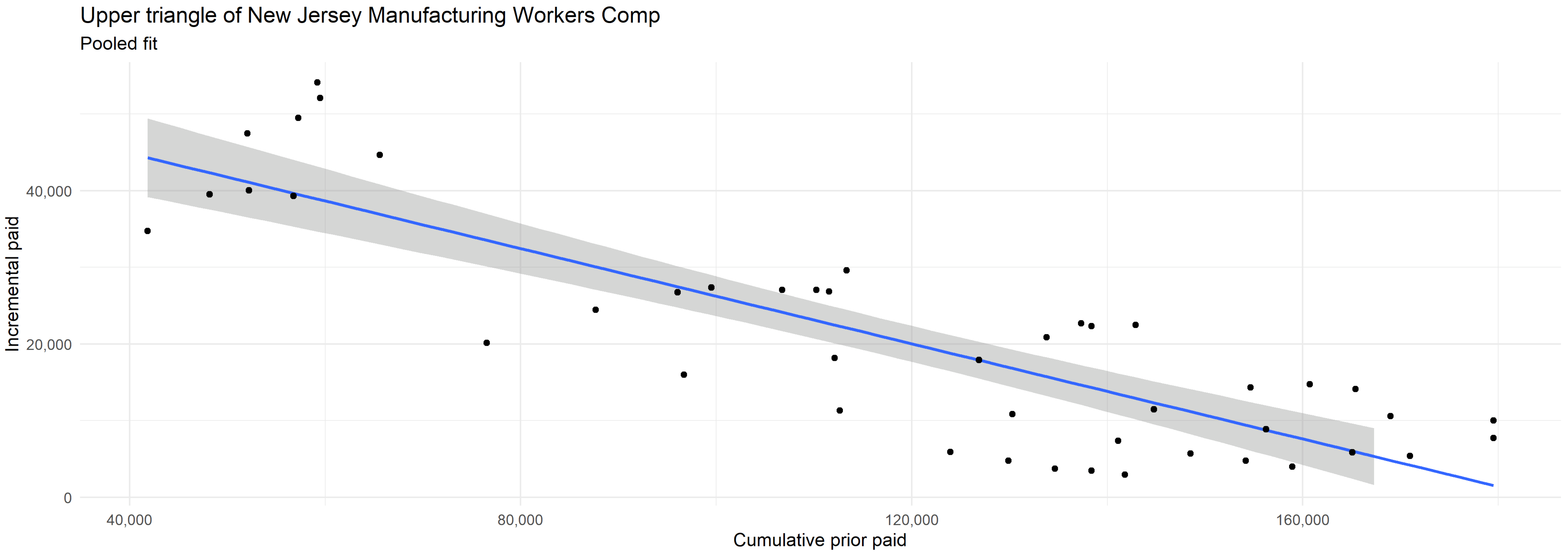
What we're about to do

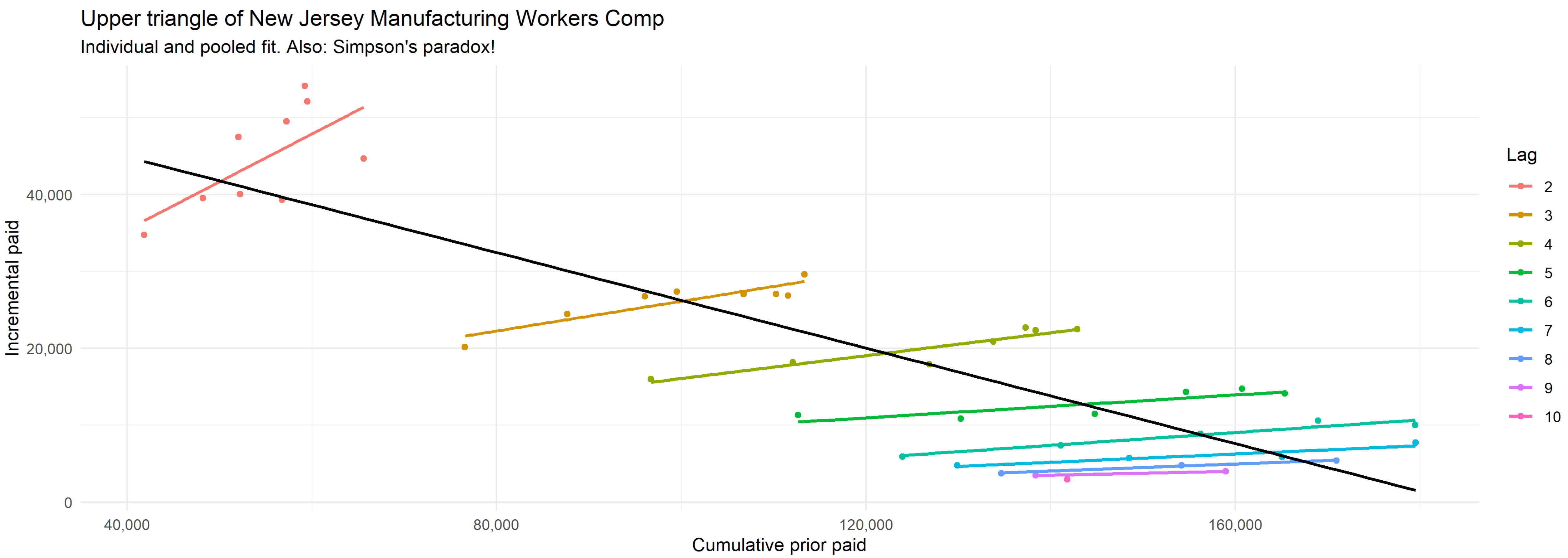
- Different equation for each development age (or is there?)
- Traditional view does not use an intercept. Several decades ago, Dan Murphy relaxed that assumption. You should consider it.
- Most actuaries take the cumulative paid or cumulative incurred as the response. Leigh Halliwell didn't and neither should you.

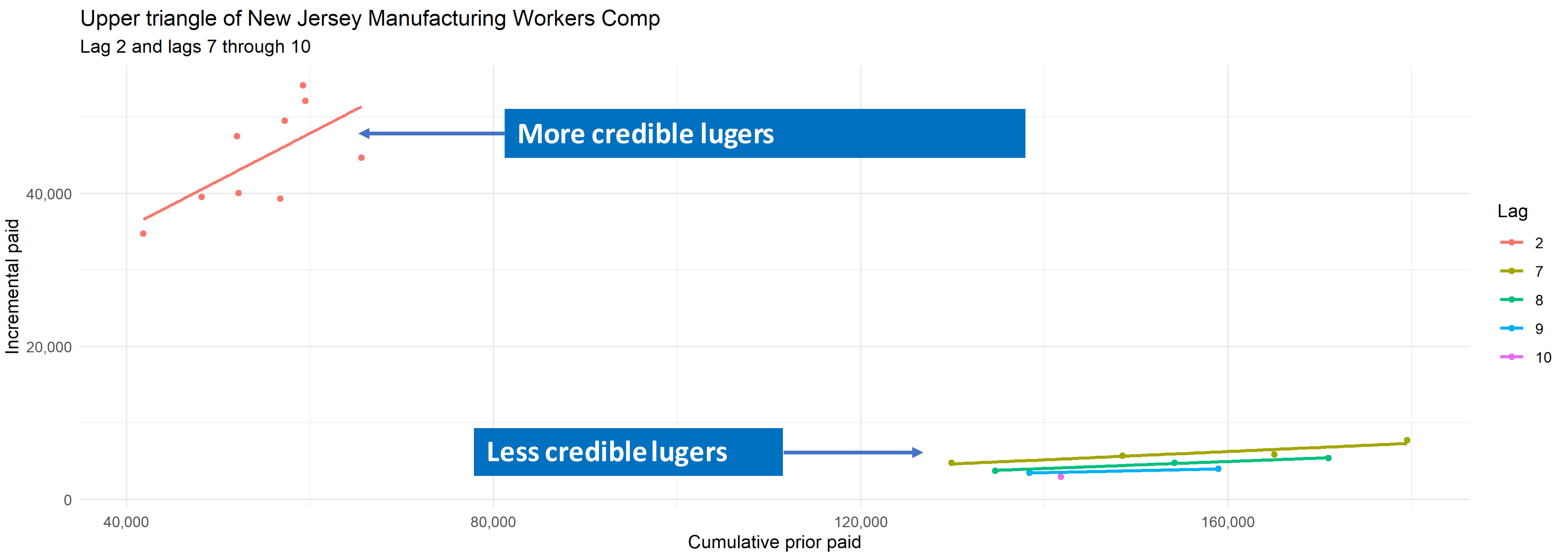
- Loss reserving is a linear model.
- Loss reserving is a linear model.
- Loss reserving is a linear model.



Upper triangle of New Jersey Manufacturing Workers Comp
Incremental paid
40,000
20,000
0
40,000
80,000
120,000
160,000
Cumulative prior paid


Upper triangle of New Jersey Manufacturing Workers Comp
Pooled fit
40,000
20,000
0
Incremental paid
40,000
80,000
120,000
160,000
Cumulative prior paid


Upper triangle of New Jersey Manufacturing Workers Comp
Individual and pooled fit. Also: Simpson's paradox!
Incremental paid
0
20,000
40,000
Cumulative prior paid
40,000
80,000
120,000
160,000
Lag
2
3
4
5
6
7
8
9
10



Upper triangle of New Jersey Manufacturing Workers Comp
Lag 2 and lags 7 through 10
More credible lugers
Less credible lugers
Incremental paid
0
20,000
40,000
40,000
80,000
120,000
160,000
Cumulative prior paid
Lag
2
7
8
9
10

```r
fit_njm_no_intercept <- lm(
  incremental_paid ~ 0 + prior_paid:lag
  , data = tbl_njm_upper %>% filter(Lag != 1)
)

fit_njm_no_intercept_tail <- lm(
  incremental_paid ~ 0 + prior_paid:lag_tail
  , data = tbl_njm_upper %>% filter(Lag != 1)
)

fit_njm_intercept <- lm(
  incremental_paid ~ 1 + lag_tail + prior_paid:lag_tail
  , data = tbl_njm_upper %>% filter(Lag != 1)
)

fit_njm_no_intercept_mixed <- lmer(
  incremental_paid ~  0 + prior_paid + (0 + prior_paid | lag_tail)
  , data = tbl_njm_upper %>% filter(Lag != 1)
)
```

Interaction -> different LDF by lag

Our grouped tail factor

Include an intercept

This model didn't fit! Uh-oh.

	Mean absolute error	Root mean squared error
No intercept	963	1,272
Pooled tail	1,424	1,687
Intercept	1,718	2,267
Mixed	??	??

Testing only uses development year 1998.
What's going on?

What’s going on

- 45 sample points is not a lot of data. Our luge example had 33% more observations.
- The model with an intercept will – for this arrangement of data! – lead to a model which overfits.
  - Theory is great. Must be supported by diagnostics for each data set
- We just got unlucky with this data set?
  - Unlikely, but let’s check it out

```
> summary(fit_njm_intercept)

Call:
lm(formula = incremental_paid ~ 1 + lag_tail + prior_paid:lag_tail,
    data = tbl_njm_upper %>% filter(Lag != 1))

Residuals:
    Min      1Q  Median      3Q     Max 
-6712.7 -1026.1    16.7   881.7  6742.4 

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)    
(Intercept)              1.058e+04  7.323e+03   1.445   0.1578    
lag_tail3               -3.777e+03  1.067e+04  -0.354   0.7255    
lag_tail4               -9.390e+03  1.102e+04  -0.852   0.4004    
lag_tail5               -8.563e+03  1.127e+04  -0.760   0.4528    
lag_tail6               -1.481e+04  1.182e+04  -1.253   0.2189    
lag_tailtail            -1.534e+04  1.098e+04  -1.397   0.1717    
lag_tail2:prior_paid     6.215e-01  1.328e-01   4.679 4.74e-05 ***
lag_tail3:prior_paid     1.930e-01  7.682e-02   2.513   0.0170 *  
lag_tail4:prior_paid     1.487e-01  6.446e-02   2.307   0.0275 *  
lag_tail5:prior_paid     7.441e-02  5.873e-02   1.267   0.2140    
lag_tail6:prior_paid     8.294e-02  5.977e-02   1.388   0.1746    
lag_tailtail:prior_paid  6.303e-02  5.350e-02   1.178   0.2472    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2639 on 33 degrees of freedom
Multiple R-squared:  0.9758,    Adjusted R-squared:  0.9678 
F-statistic: 121.1 on 11 and 33 DF,  p-value: < 2.2e-16
```

Most of these parameters aren’t all that good.

Get more data

```r
fit_wc_no_intercept <- lm(
  incremental_paid ~ 0 + prior_paid:Lag
  , data = tbl_wc_upper %>% filter(Lag != 1)
)

fit_wc_no_intercept_tail <- lm(
  incremental_paid ~ 0 + prior_paid:lag_tail
  , data = tbl_wc_upper %>% filter(Lag != 1)
)

fit_wc_mixed <- lmer(
  incremental_paid ~ 0 + prior_paid:lag_tail + (0 + prior_paid:lag_tail | Company)
  , data = tbl_wc_upper %>% filter(Lag != 1)
)
```

	Mean absolute error	Root mean squared error
No intercept	253	949
Pooled tail	269	971
Mixed	218	807

With more data, the mixed model outperforms

Nota bene

- We’re cheating a bit here. We are only able to look at out-of-sample performance because we waited ten years for it. (Hat tip -> Glenn Meyers and Peng Shi!)
- Cross validation on the upper triangle is the only way we can estimate OOS performance in the here and now.

Conclusion

What have we learned

- Mixed effects models may be viewed as a particular implementation of credibility
- Credibility is about much more than sample size
- Easy to explore in R using the nlme package
- Theory must answer to model diagnostics
- Reserve losses at different levels of granularity

Thank you!

Any questions?