Shrinkage and cross-validation

Shrink instead of choose

1 Shrinkage
2 Mixed models
3 Ridge and lasso
4 Cross-validation
5 CV traps
6 Classification



Week 5 recap

Kahoot-style warm-up

8 quick questions on last week.

Not graded.

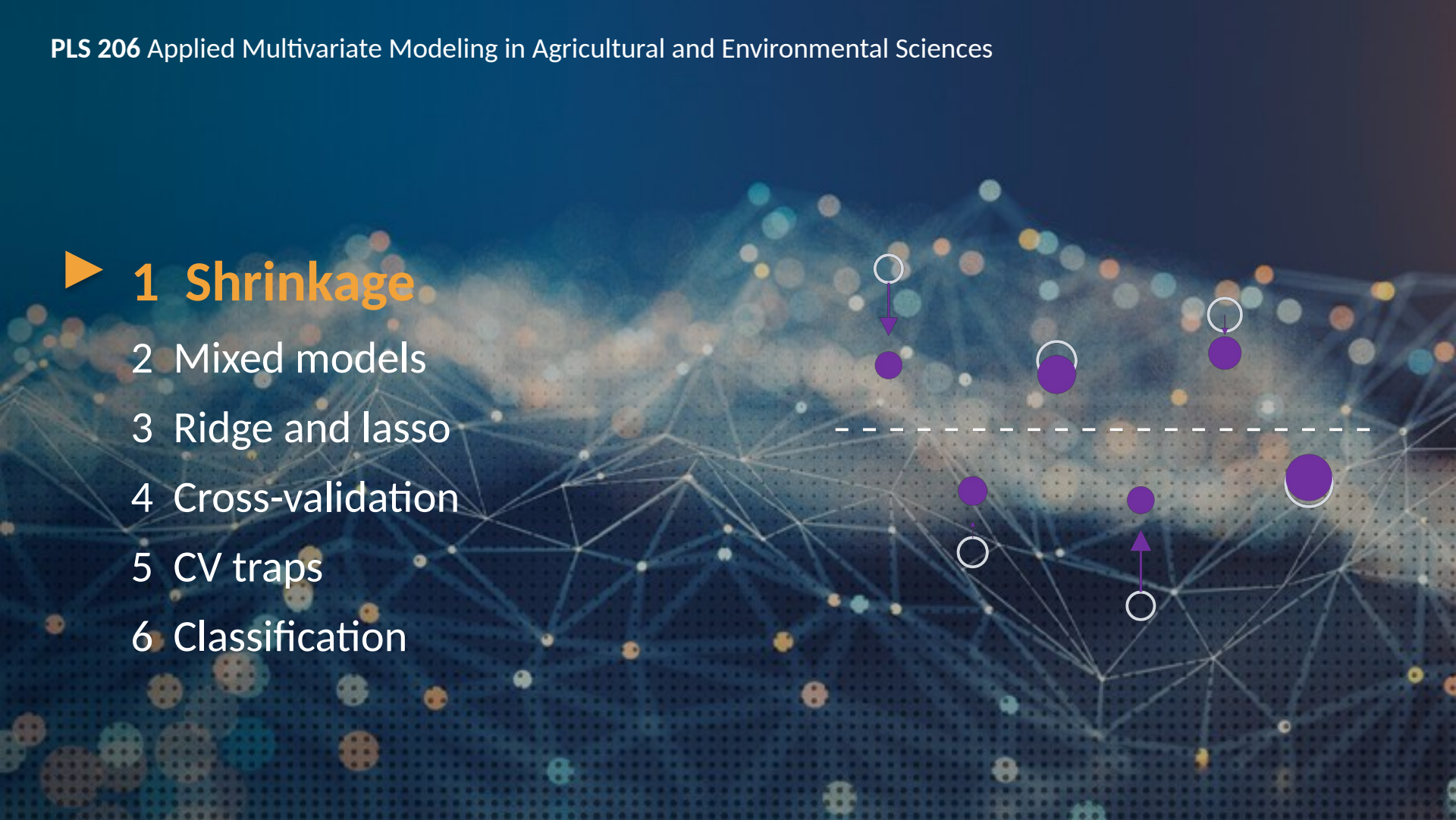
Use a pseudonym if you want.

Fast and correct wins.

Scan with your phone

[Join on your phone or laptop: greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w6-week5-recap]

[host screen]



1 **Shrinkage**

2 Mixed models

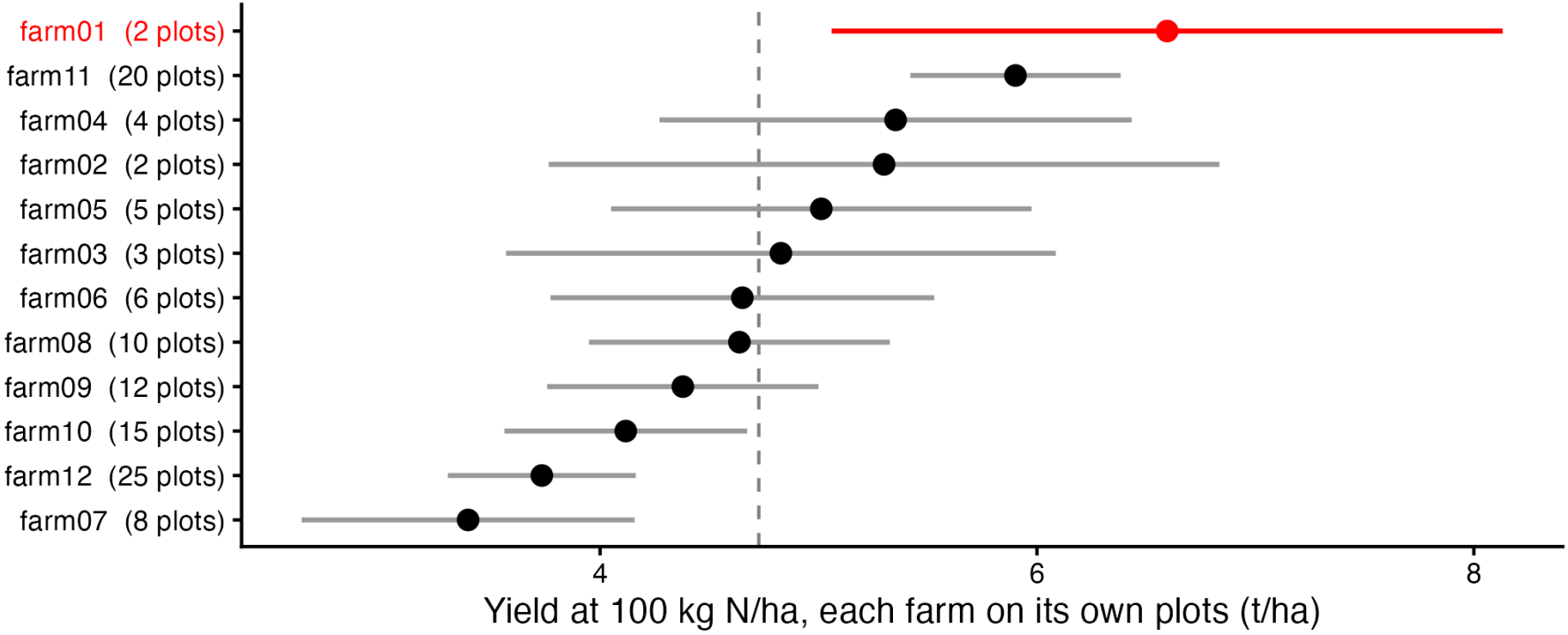
3 Ridge and lasso

4 Cross-validation

5 CV traps

6 Classification

Which farm is best?



farm01 tops the table: 6.60 t/ha, on **2 plots**

Live poll

Multiple choice

Next season, at 100 kg N/ha, which farm do you bet yields more?

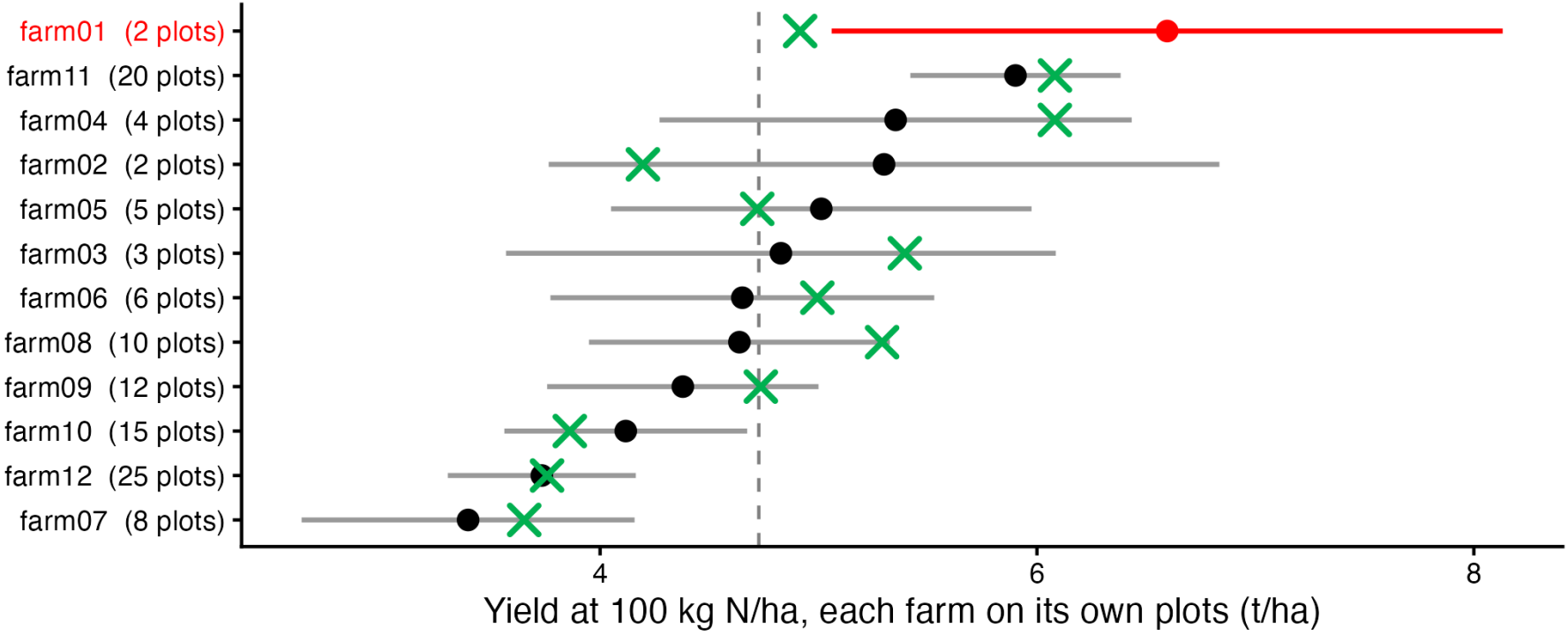
A farm01: 6.6 t/ha, 2 plots
B farm11: 5.9 t/ha, 20 plots
C No way to tell

Scan with your phone

[Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-best-farm](greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-best-farm)

[host screen]

The truth: the top farm was lucky



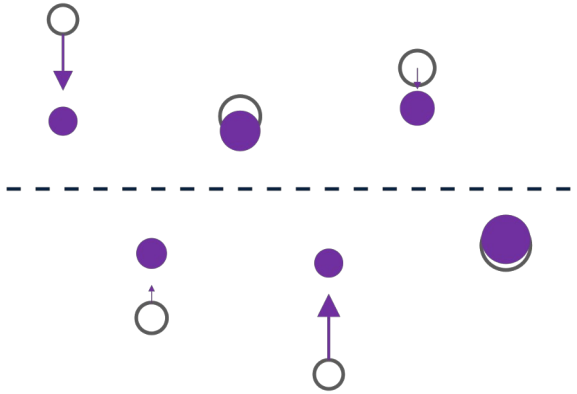
× = the truth (simulated). farm01: really **4.92 t/ha**, about average

Shrink toward the average

$$\text{shrunk} = \text{average} + w \times (\text{own} - \text{average})$$

$$\text{farm01:} \quad 4.73 + 0.49 \times (6.60 - 4.73) \approx 5.64 \text{ t/ha}$$

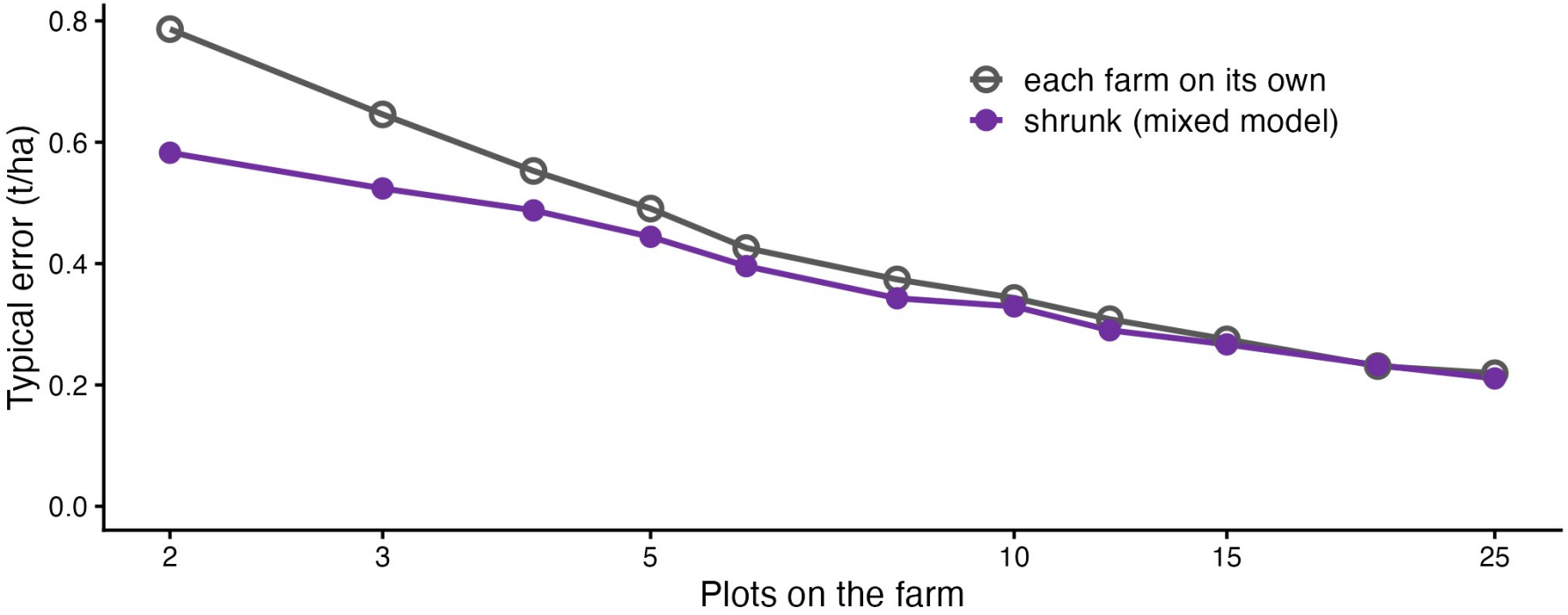
w: between 0 (trust the average) and 1 (trust the farm); grows with the farm's plots



Few plots: **small w**, pulled hard. Many plots: **w near 1**, barely moves.

A farm's estimate is a **compromise** between its own plots and all the farms

Replay the trial 500 times



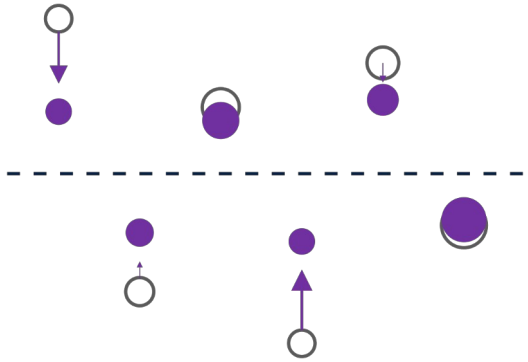
2-plot farms: error **0.79** → **0.58 t/ha**. 25 plots: **0.22** → **0.21**

Two ways to shrink this week

Mixed models — farm effects → the average farm

Ridge and lasso — slopes → zero

How much? — Wednesday: cross-validation



Both trade a little **bias** for much less **variance**

1 Shrinkage

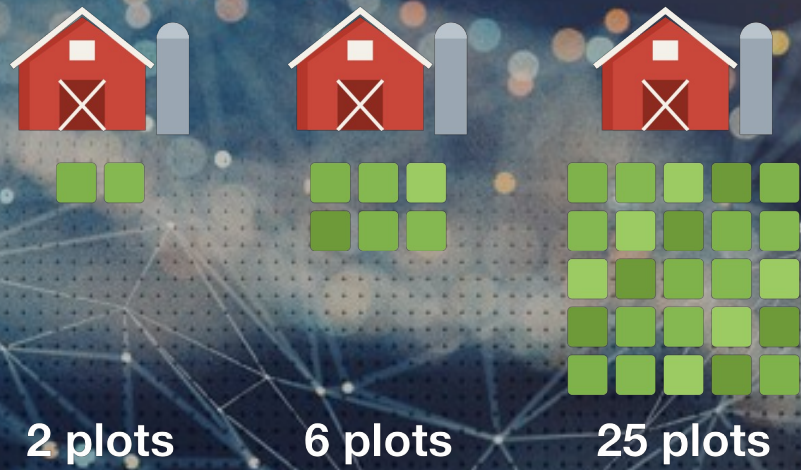
▶ **2 Mixed models**

3 Ridge and lasso

4 Cross-validation

5 CV traps

6 Classification



Where this goes

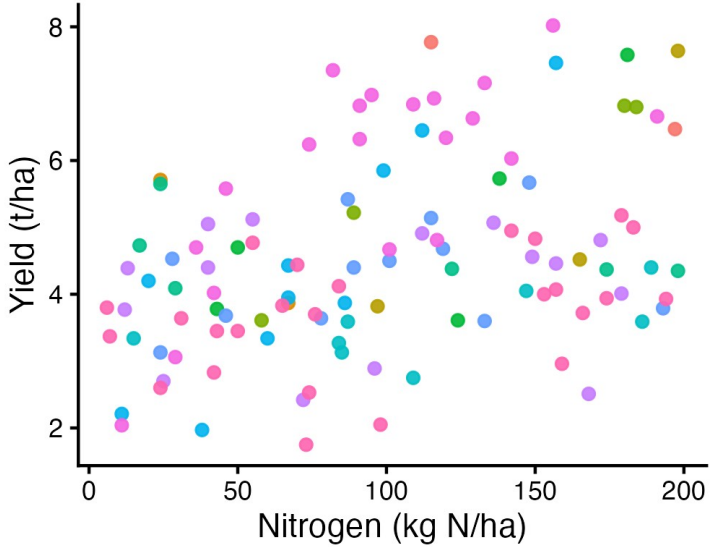
	independent	grouped
Normal y	lm()	lmer()
yes/no, counts	glm()	glmer()

Same formulas, plus **(1 | farm)**. Package lme4.

112 plots on 12 farms

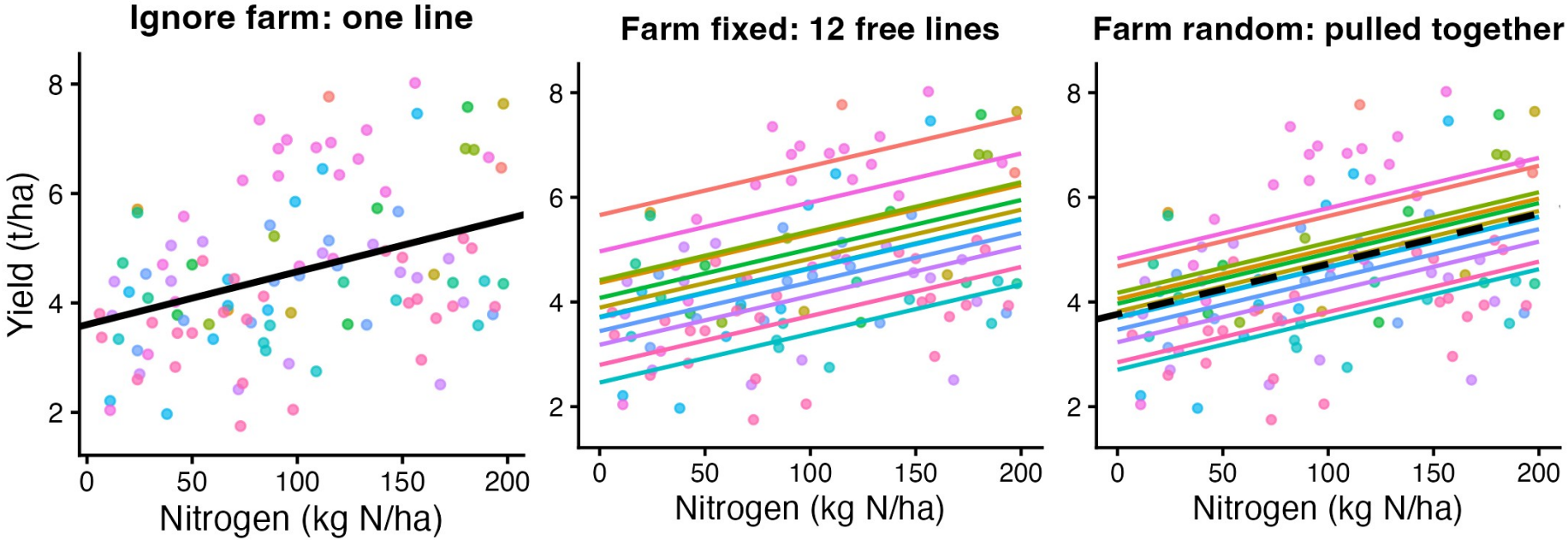
```
> farms <- read.csv("https://greymonroe.github.io/PLS_206/data/farm_plots.csv")
> head(farms, 4)
    farm cover nitrogen yield rust
1 farm01    no      115  7.77    1
2 farm01    no      197  6.47    0
3 farm02    no       24  5.71    0
4 farm02    no       67  3.87    1
> nrow(farms)
[1] 112  ← plots
```





Each farm has its own baseline. **2 to 25 plots per farm.**

Three ways to treat farm



Fixed: each farm on its own. **Random: farms are a sample of all farms.**

What 'random' means

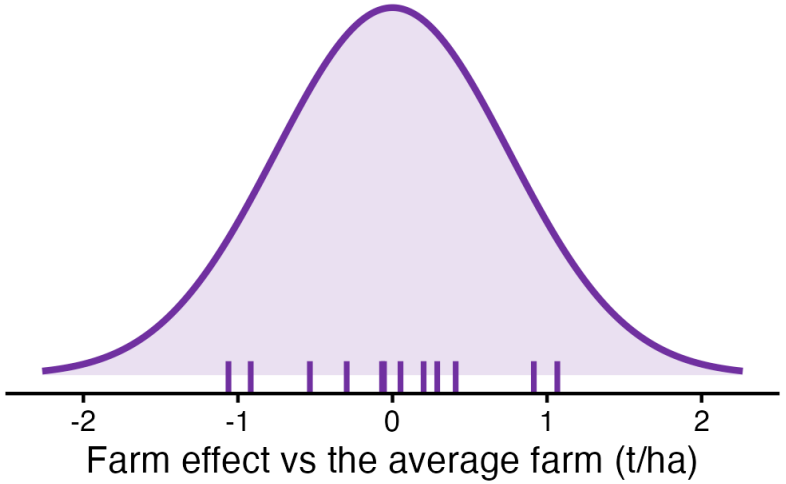
$$\text{yield}_{ij} = \beta_0 + u_j + \beta_1 \,\text{nitrogen}_{ij} + \varepsilon_{ij}$$

$$u_j \sim N(0,\ \sigma^2_{\text{farm}}) \qquad \varepsilon_{ij} \sim N(0,\ \sigma^2)$$

i: plot j: farm u_j: farm j's offset from the average farm

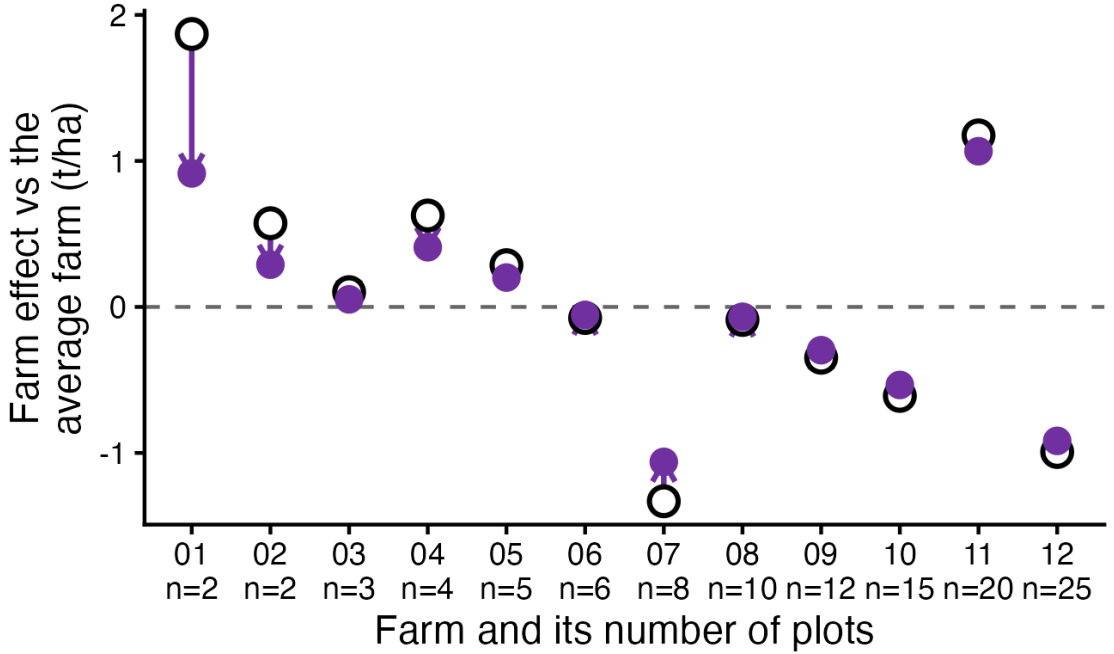
Fixed: estimate 11 farm differences.

Random: estimate one spread,
farm SD σ = 0.76 t/ha



'Random' = drawn from a **population of farms** we want to generalize to

Shrinkage: small farms borrow strength



farm fixed (own plots)

farm random (lmer)

$$w_j = \frac{\sigma^2_{\text{farm}}}{\sigma^2_{\text{farm}} + \sigma^2/n_j}$$

$n = 2$: 0.49 $n = 25$: 0.92

farm01: +1.87 on its own, **+0.91 t/ha** in the mixed model

In R: lmer()

```
> library(lme4)
> m_mixed <- lmer(yield ~ nitrogen + (1 | farm), data = farms)
> VarCorr(m_mixed)
 Groups   Name        Std.Dev.
 farm     (Intercept) 0.75542  ← σ_farm: farm-to-farm SD
 Residual             1.08355  ← σ: plot-to-plot SD
> round(coef(summary(m_mixed)), 4)
            Estimate Std. Error t value
(Intercept)   3.7653     0.3186 11.8181
nitrogen      0.0096     0.0019  5.1223
```

2 variances
instead of 12 intercepts

Same nitrogen slope:
0.0096 vs 0.0094

```
> m_fixed <- lm(yield ~ nitrogen + farm, data = farms)
> length(coef(m_fixed))
[1] 13  ← 12 farm intercepts + slope
> round(coef(summary(m_fixed))["nitrogen", ], 4)
  Estimate Std. Error    t value   Pr(>|t|)
    0.0094     0.0019     4.8882     0.0000  ← same slope
```

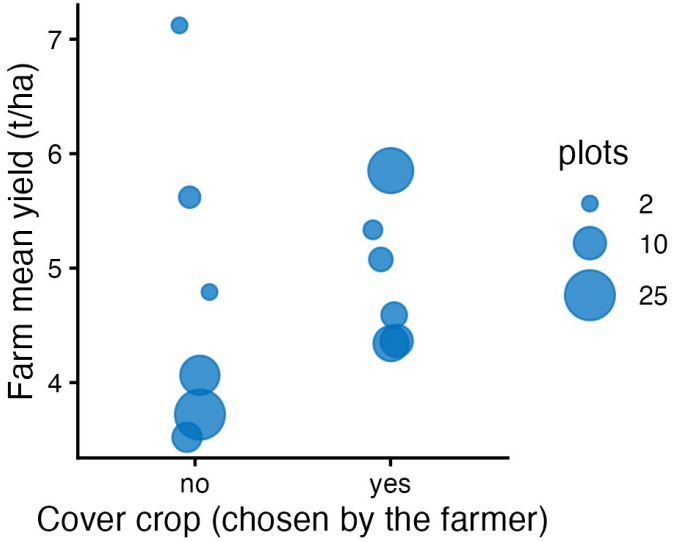
Nitrogen varies **within** farms, so the slope barely changes

p-values: load lmerTest instead

```
> library(lmerTest)   # instead of lme4
> m_mixed <- lmer(yield ~ nitrogen + (1 | farm), data = farms)
> signif(coef(summary(m_mixed)), 3)
            Estimate Std. Error    df t value Pr(>|t|)
(Intercept)  3.77000    0.31900  24.6   11.80 1.25e-11
nitrogen     0.00961    0.00188 105.0    5.12 1.38e-06  ← p
```

lmerTest = the same lmer(), plus df and p-values (Satterthwaite)

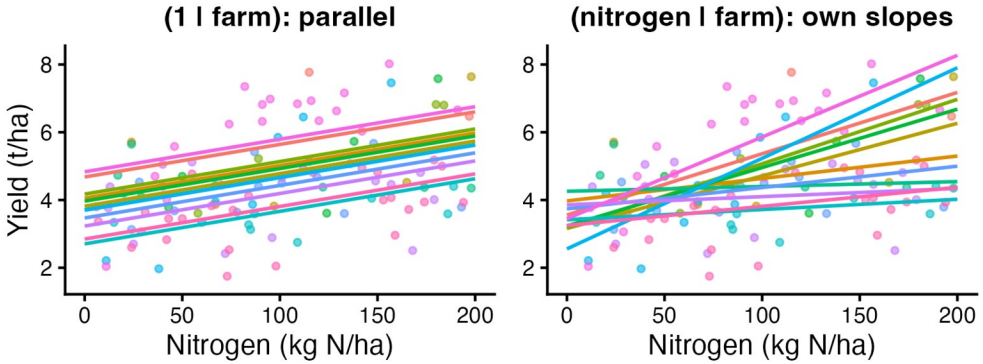
Why it matters: a treatment chosen per farm




```
> m_plots <- lm(yield ~ nitrogen + cover, data = farms)
> cf <- coef(summary(m_plots))
> signif(cf["coveryes", , drop = FALSE], 3)
          Estimate Std. Error t value Pr(>|t|)
coveryes      1.02      0.235    4.35 3.04e-05  ← lm
> m_cover <- lmer(yield ~ nitrogen + cover + (1 | farm),
+                 data = farms)
> cf <- coef(summary(m_cover))
> signif(cf["coveryes", , drop = FALSE], 3)
          Estimate Std. Error   df t value Pr(>|t|)
coveryes     0.417      0.497 8.32   0.838    0.425  ← lmer
```

lm(): t = 4.4, p < 0.001. lmer(): p = 0.43. **The truth: no effect.**

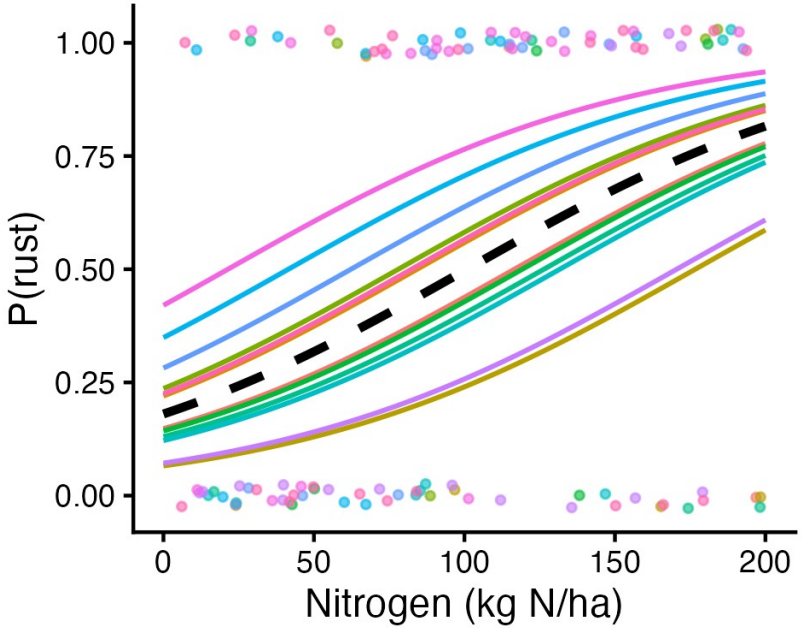
Do farms respond differently? Random slopes




```
> farms$n100 <- farms$nitrogen / 100   # per 100 kg N
> m_int   <- lmer(yield ~ n100 + (1 | farm), data = farms)
> m_slope <- lmer(yield ~ n100 + (n100 | farm), data = farms)
> a <- anova(m_int, m_slope, refit = FALSE)
> a[, c("npar", "AIC", "Chisq", "Df", "Pr(>Chisq)")]
        npar    AIC  Chisq Df Pr(>Chisq)
m_int      4 362.51
m_slope    6 348.91 17.602  2  0.0001506 ***
```

(n100 | farm): each farm its own slope, also shrunk. Rescale N so lmer converges.

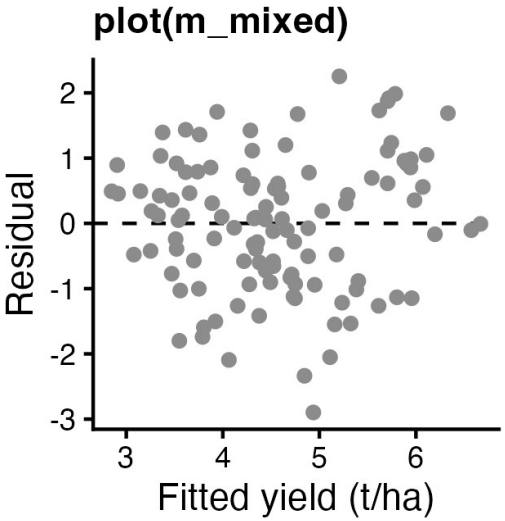
Yes/no and counts: glmer()

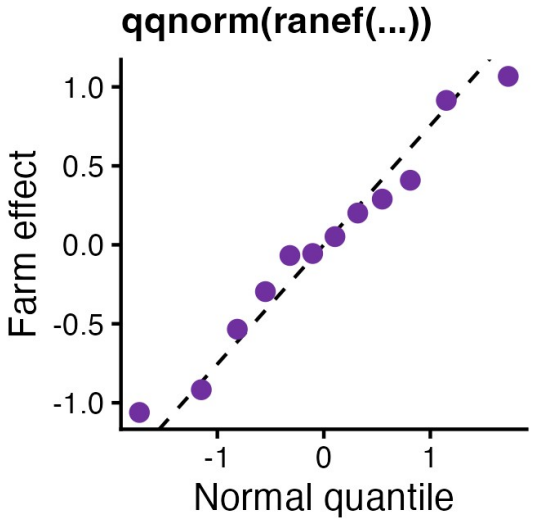



```
> m_rust <- glmer(rust ~ nitrogen + (1 | farm), data = farms,
+                 family = binomial)
> round(coef(summary(m_rust)), 4)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   -1.514     0.6048 -2.5034   0.0123
nitrogen       0.015     0.0044  3.3943   0.0007
> exp(10 * fixef(m_rust)["nitrogen"])   # odds ratio per 10 kg N
nitrogen
1.162025  ← odds ×1.16 per 10 kg N
```

glm() + **(1 | farm)**: odds of rust ×1.16 per 10 kg N, p < 0.001

Checks





as for lm()	**residual plot**
farm effects	**roughly Normal**
groups	**5 or more**

Fewer than 5 groups? Use a fixed factor: one spread from 3 farms is a guess

Live poll

Multiple choice

4 wheat varieties, each once in each of 8 blocks. What goes in (1 | …)?

A variety
B block
C both
D neither

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-fixed-random

host screen

Fixed or random?

you want each level's own effect	fixed (these 4 varieties)
levels sample a population	random (blocks, farms, years)
fewer than 5 levels	fixed
treatment applied to whole groups	group must be random
y is 0/1 or a count	`glmer(..., family = binomial)`

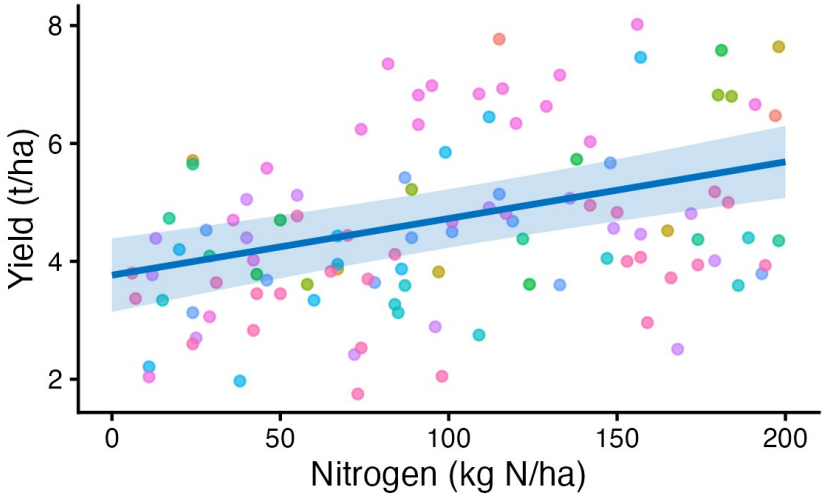
Reporting a mixed model

Methods

"We fit a linear mixed model of yield on nitrogen with **farm as a random intercept** (112 plots on 12 farms; lme4, p-values from lmerTest)."

Results

"Yield increased with nitrogen (**0.96 t/ha per 100 kg N**, $p < 0.001$; Fig. 1). Farm-to-farm SD 0.76 t/ha; plot-to-plot SD 1.08 t/ha."



Methods: the model and the groups. **Results**: the effect, both SDs, the plot

1 Shrinkage
2 Mixed models
3 **Ridge and lasso**
4 Cross-validation
5 CV traps
6 Classification



A random effect is a price on big offsets

$$\text{mixed model:} \quad \sum (y_i - \hat{y}_i)^2 + \frac{\sigma^2}{\sigma^2_{\text{farm}}} \sum_j u_j^2$$

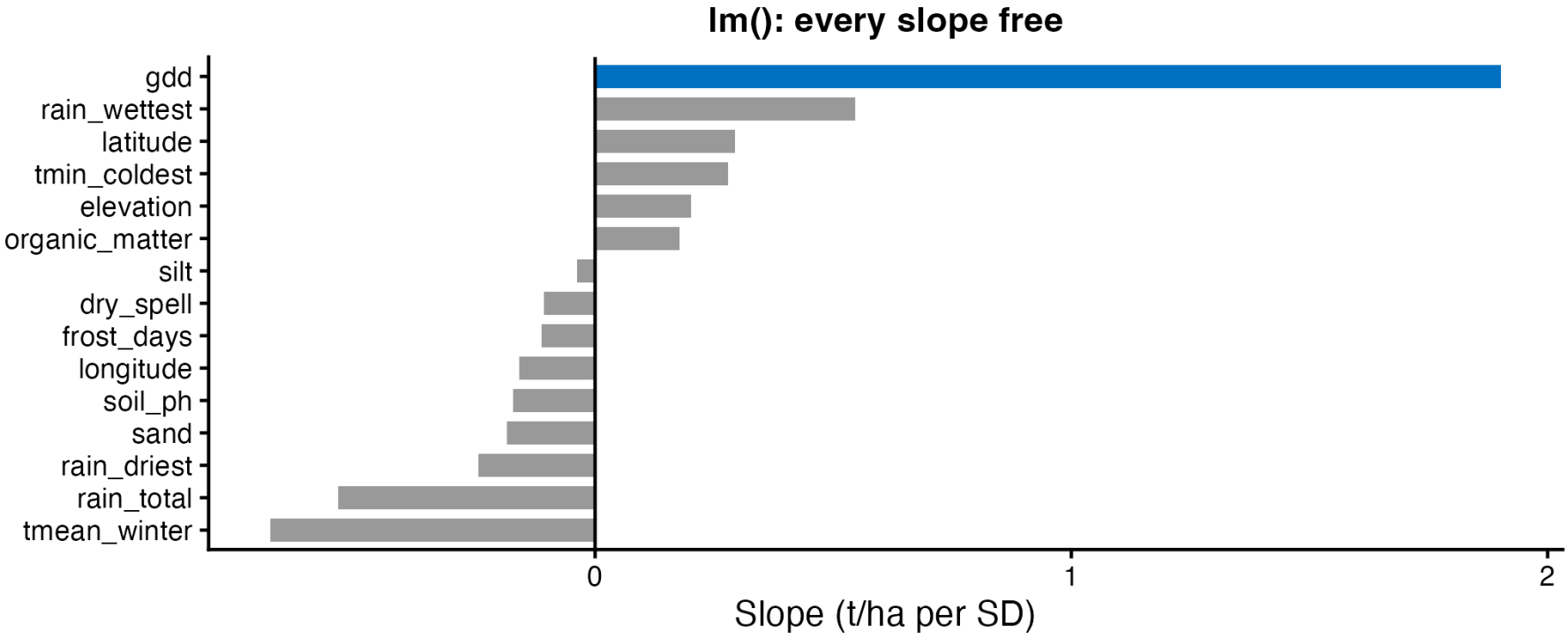
$$\text{ridge:} \quad \sum (y_i - \hat{y}_i)^2 + \lambda \sum_k \beta_k^2$$

first term: misfit second: a price on big farm offsets (u_j) or big slopes (β_k)

lmer sets its own price from the data: $\sigma^2/\sigma^2_{\text{farm}} = 1.08^2/0.76^2 = 2.1$

Mixed models price big **farm offsets**. Ridge prices big **slopes**.

Back to 15 predictors, one real driver



lm(): tmean_winter gets **-0.68 t/ha per SD**. Truth: only **gdd** matters

Ridge: least squares plus a price on slopes

$$\text{minimize} \quad \sum_i (y_i - \hat{y}_i)^2 \; + \; \lambda \sum_k \beta_k^2$$

first term: least squares, the lm() fit λ: how much each unit of slope costs

$\lambda = 0$: plain lm() $\lambda \to \infty$: every slope 0, predict the mean

A big slope must **earn its price** by fitting the data much better

Scale the predictors first

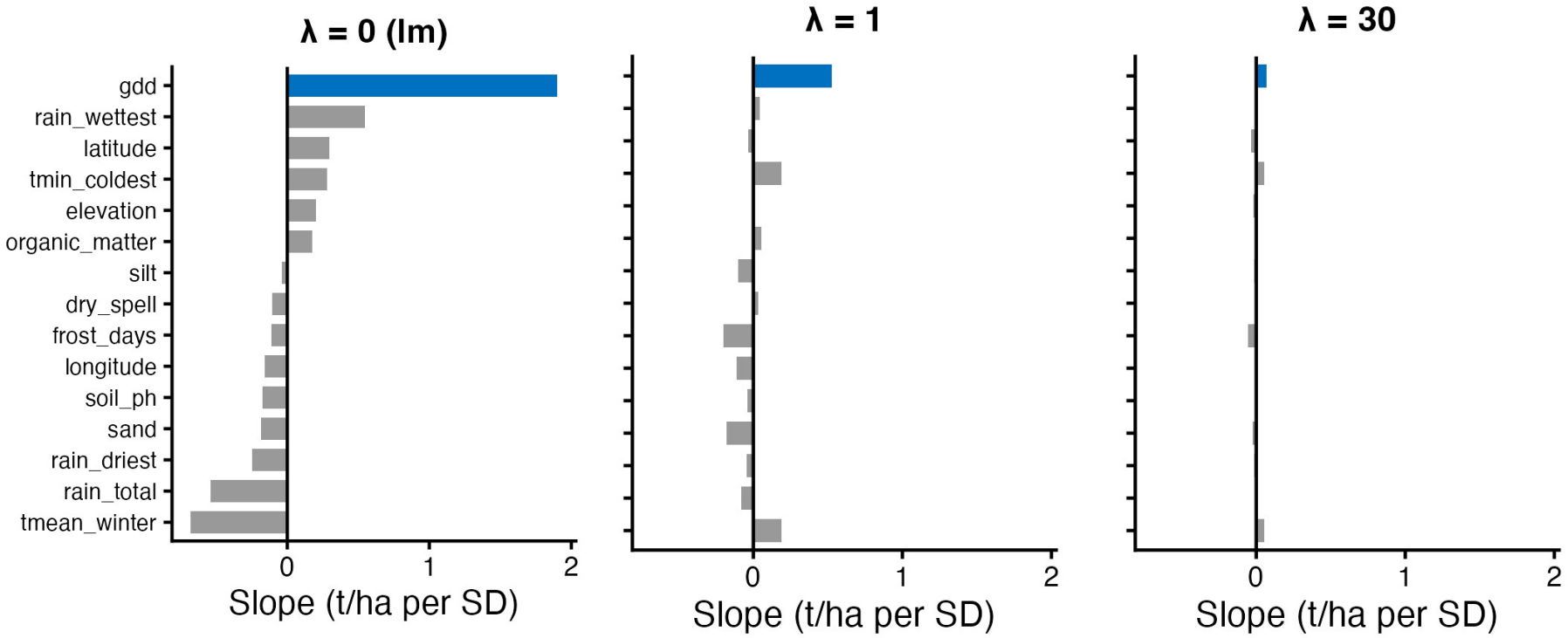
```
> round(apply(cc[, c("gdd", "rain_total", "soil_ph")], 2, sd),
1)
       gdd rain_total    soil_ph
     203.0      160.0        0.5  ← very different units
```

The price is per unit of slope, and units are arbitrary.

scale(): every predictor in **SD units**, so each pays the same price.

Always scale before ridge or lasso (glmnet does it for you by default)

Turn the dial: λ



λ = 1: gdd 0.52, junk under 0.20. λ = 30: **everything near 0**

Live poll

Multiple choice

Ridge with an enormous λ.

What does it predict for every farm?

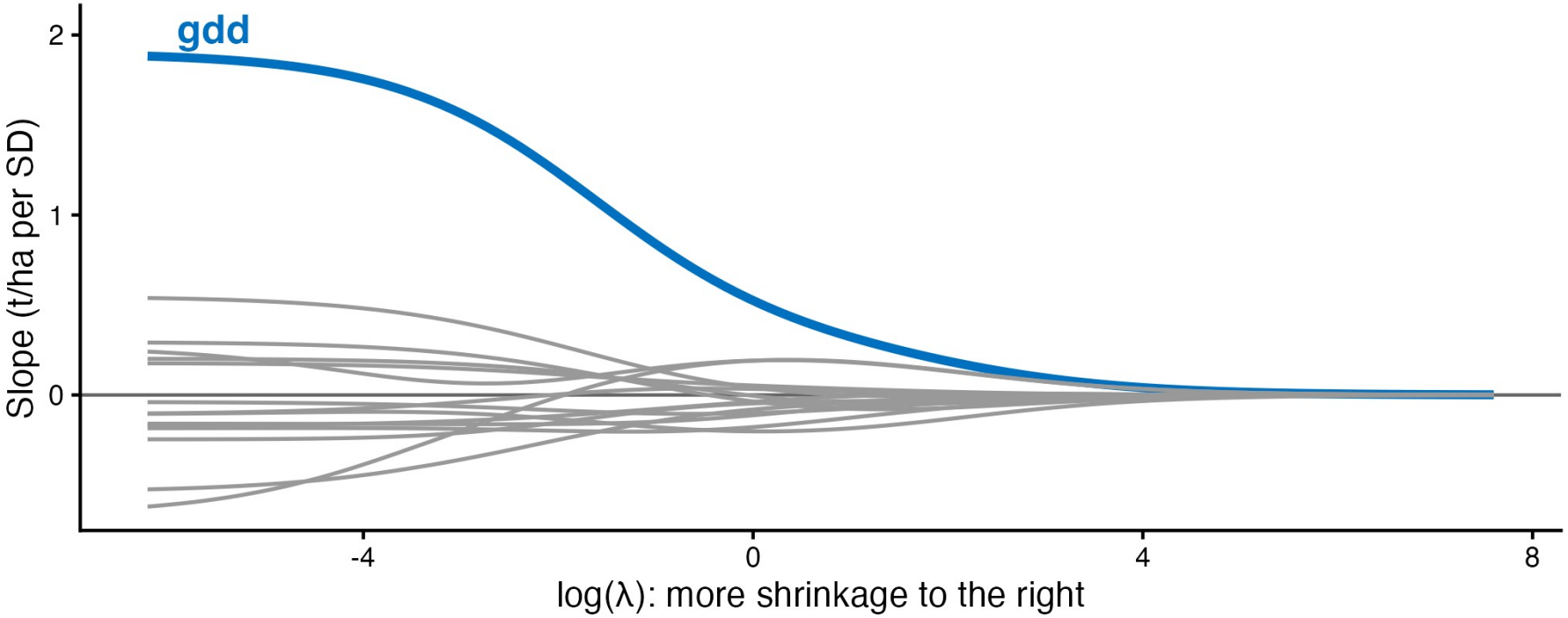
A The mean biomass
B Zero biomass
C The same as lm()
D gdd alone

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-lambda-huge

host screen

Ridge paths: every slope shrinks, none hits zero



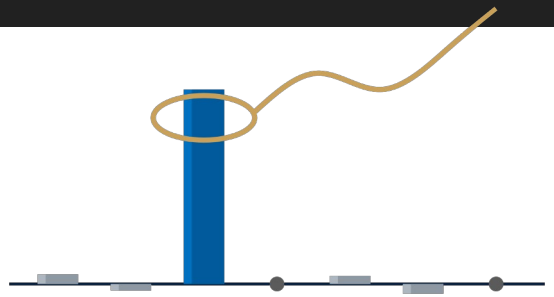
gdd shrinks slowest. The others fade but **never reach exactly 0**

Lasso: price the absolute slope

ridge: $\sum_i (y_i - \hat{y}_i)^2 + \lambda \sum_k \beta_k^2$

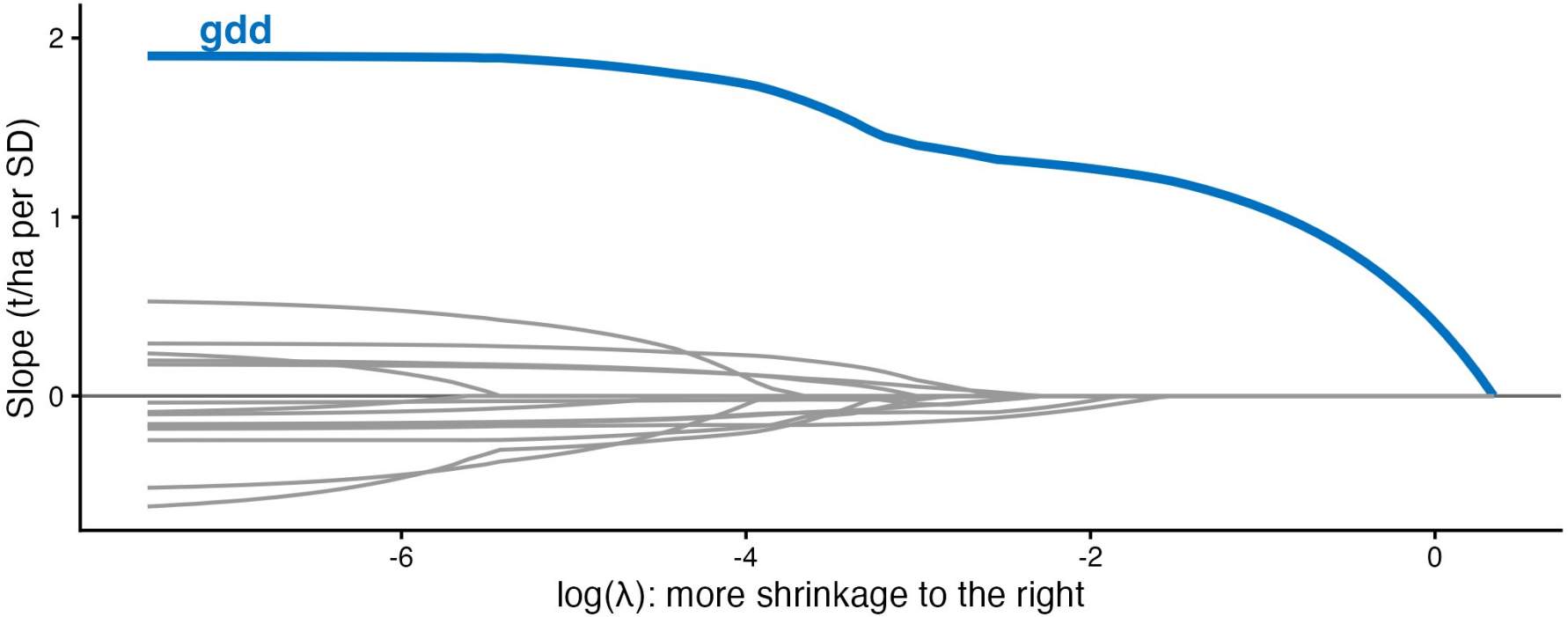
lasso: $\sum_i (y_i - \hat{y}_i)^2 + \lambda \sum_k |\beta_k|$

square vs absolute value: small slopes cost almost nothing in ridge, but full price in lasso



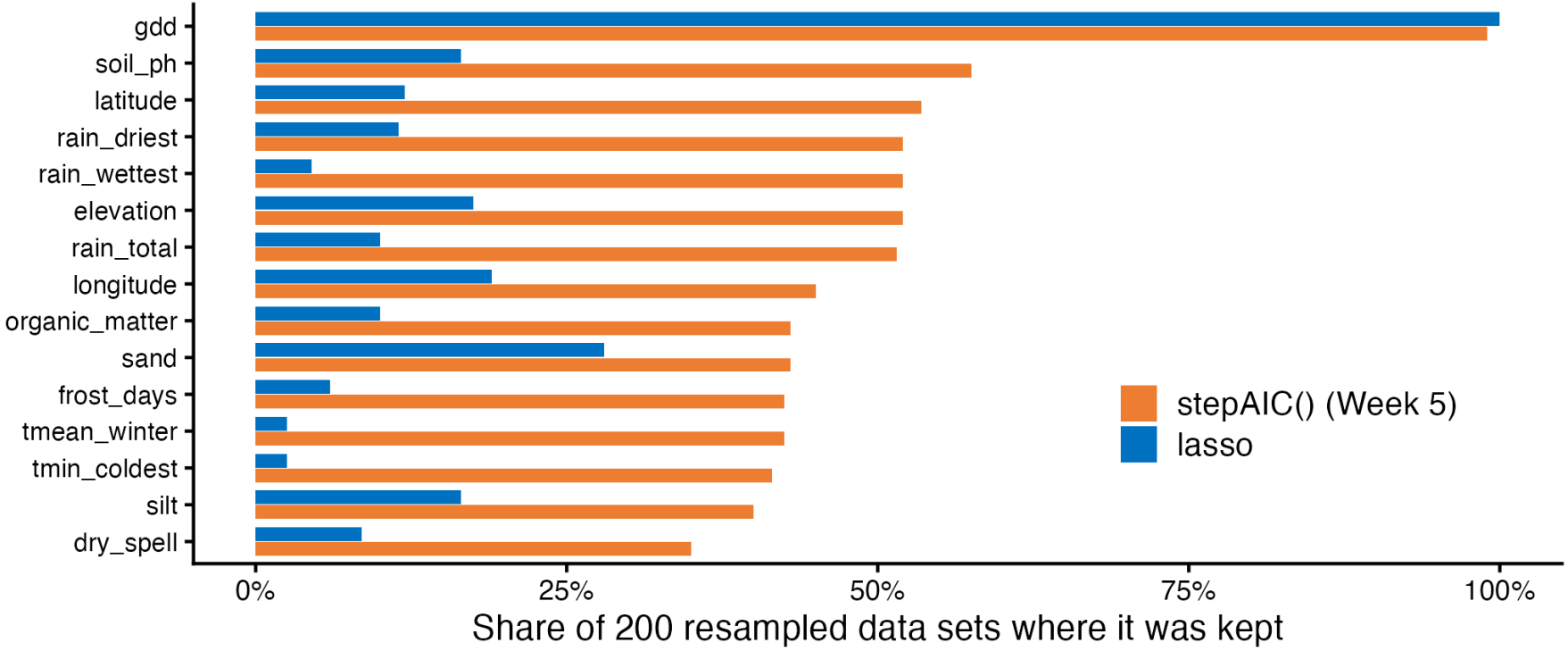
The lasso: Tibshirani 1996, J. Royal Statistical Society B 58: 267-288

Lasso paths: slopes hit exactly zero



Turn up λ and predictors drop out one by one. **gdd is the last one standing**

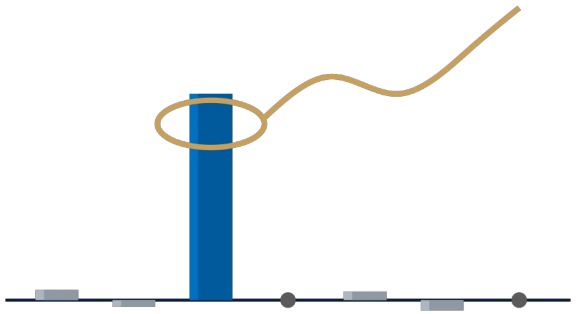
Lasso vs stepwise, 200 resampled data sets



gdd alone: **lasso 97 of 200**, **stepAIC 0 of 200**

Ridge, lasso, elastic net

ridge (alpha = 0)	**shrinks all; keeps all; good with many small effects**
lasso (alpha = 1)	**shrinks and drops; picks one of a correlated pair**
elastic net (0 < alpha < 1)	**a mix: drops some, shares among correlated ones**



glmnet: one function, the **alpha** argument chooses

In R: glmnet()

```
> library(glmnet)
> X <- scale(as.matrix(cc[, -1]))     # predictors on one scale
> y <- cc$biomass
> fit_lasso <- glmnet(X, y, alpha = 1)  # alpha = 0 for ridge
> round(coef(fit_lasso, s = 300), 1)    # slopes at lambda = 300
16 x 1 sparse Matrix of class "dgCMatrix"
                s=300
(Intercept)    4252.9
latitude          .  ← exactly 0
longitude         .
elevation         .
gdd            1117.0  ← the only slope left
tmean_winter      .
tmin_coldest      .
frost_days        .
rain_total        .
rain_wettest      .
rain_driest       .
dry_spell         .
sand              .
silt              .
soil_ph           .
organic_matter    .
```

x: a **matrix**
not a formula

λ = 300: 1 slope left

. = exactly zero: the lasso dropped it

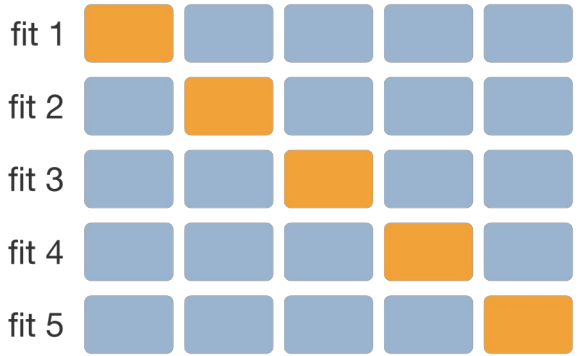
Shrunk slopes have no p-values

good for	**prediction, choosing predictors**
not for	**p-values, confidence intervals**
why	**biased toward 0 on purpose**

To test an idea: **lm() / lmer() and AIC**. To predict: **shrink**

Three ways to shrink

mixed model	**group effects → average group; amount set by the variances**
ridge	**slopes → 0; amount set by λ**
lasso	**slopes → exactly 0; amount set by λ**



How do we choose λ? **Cross-validation**, Wednesday

Live coding: shrinkage



`Week6_live_1_shrink.R`

Part 1

1 Farms: lm vs lmer, shrinkage
2 lmerTest, cover crops
3 Random slopes, glmer
4 Ridge and lasso with glmnet

Shrinkage and cross-validation, part 2

1 Shrinkage
2 Mixed models
3 Ridge and lasso
▶ **4 Cross-validation**
5 CV traps
6 Classification



Monday recap

Kahoot-style warm-up

8 quick questions on Monday.

Not graded.

Use a pseudonym if you want.

Fast and correct wins.

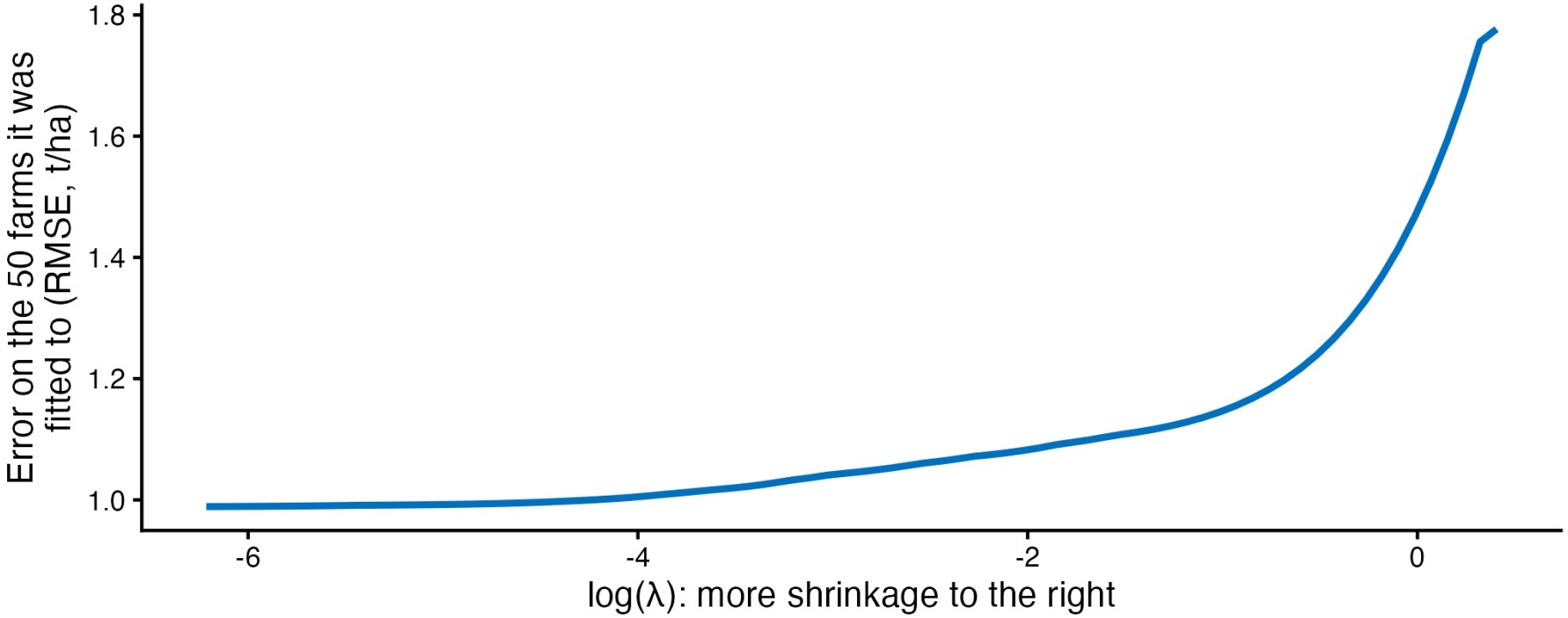


Scan with your phone

[Join on your phone or laptop: greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w6-monday-recap](greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w6-monday-recap)

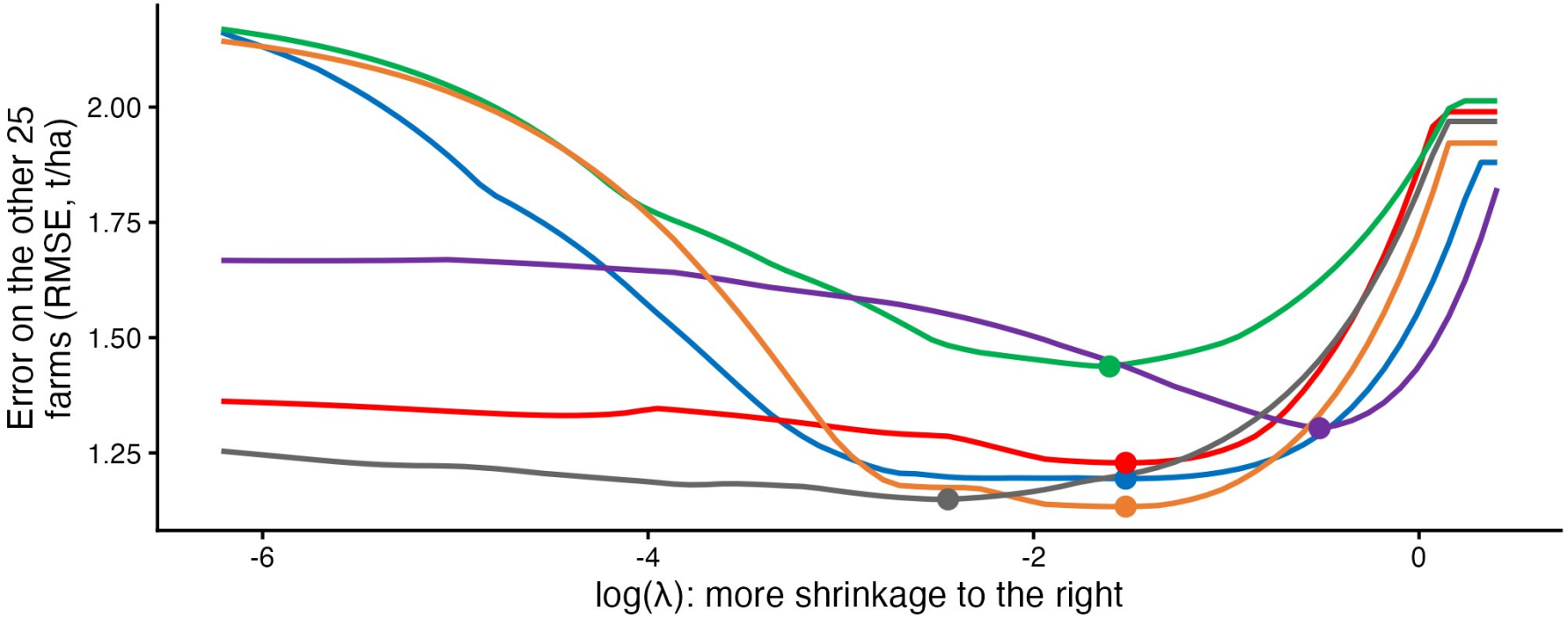
[host screen]

Which λ? Not the one that fits best



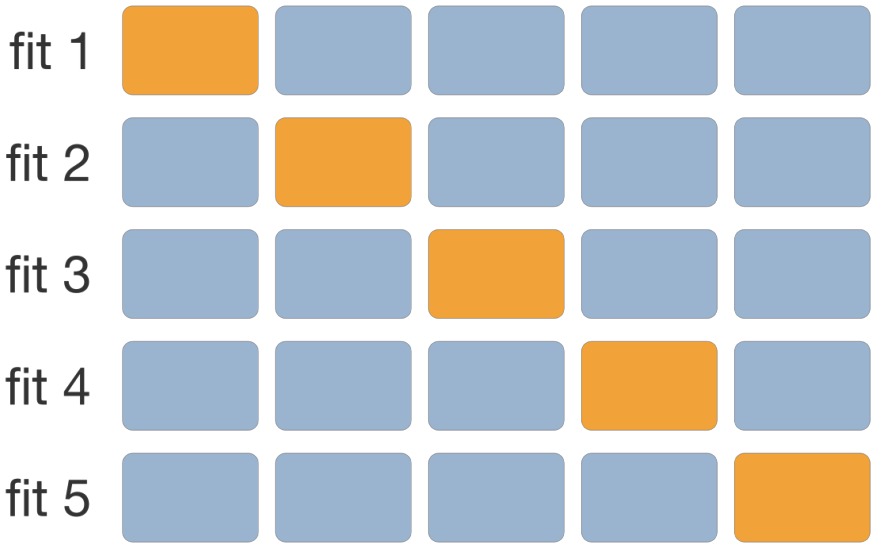
On the farms it was fitted to, **λ = 0 always wins**: back to plain lm()

Hold out half: the answer depends on the split



Six random 25/25 splits, **six different best λ** (dots)

k-fold: every farm gets a turn



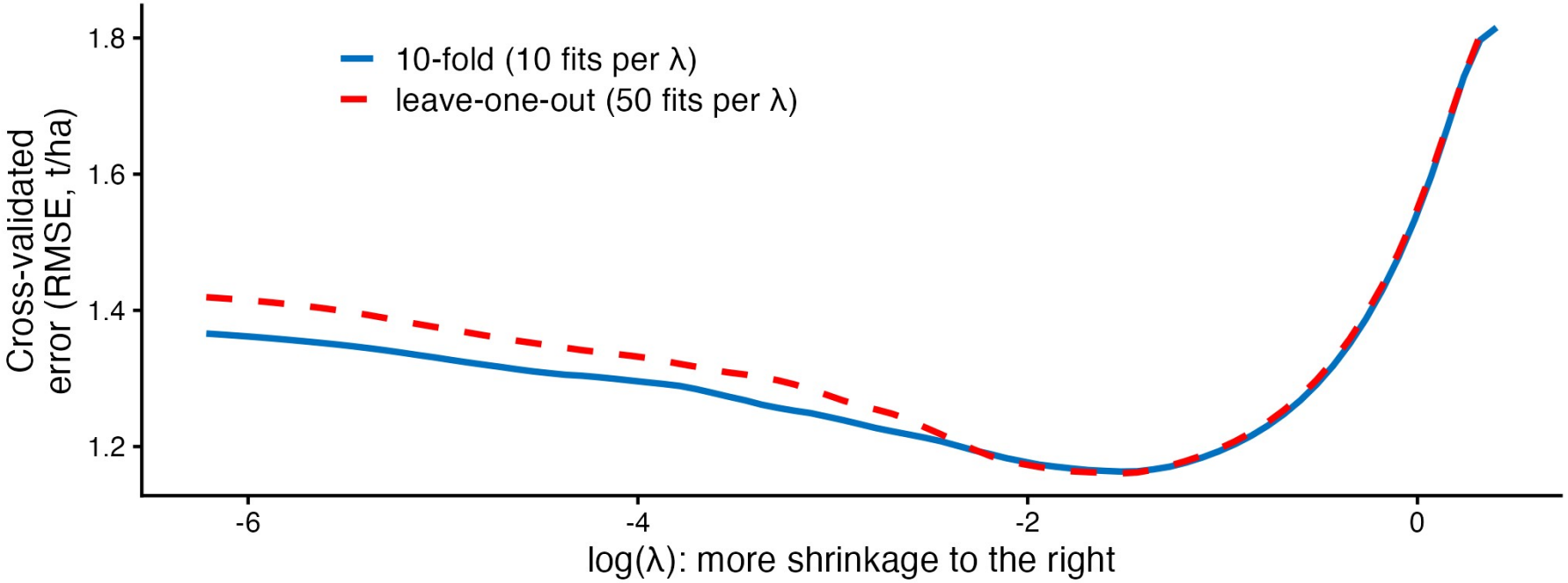
1 Split the farms into k folds (here 5)

2 Fit on k − 1 folds, predict the **one left out**

3 Repeat until every fold has been left out once; **average the errors**

Every farm is predicted once, by a model that **never saw it**

Leave-one-out: k = n



Leave-one-out and 10-fold agree here. Week 5: **AIC ≈ leave-one-out**

Live poll

Multiple choice

10-fold CV on the 50 farms. For ONE value of λ, how many times is the model fitted?

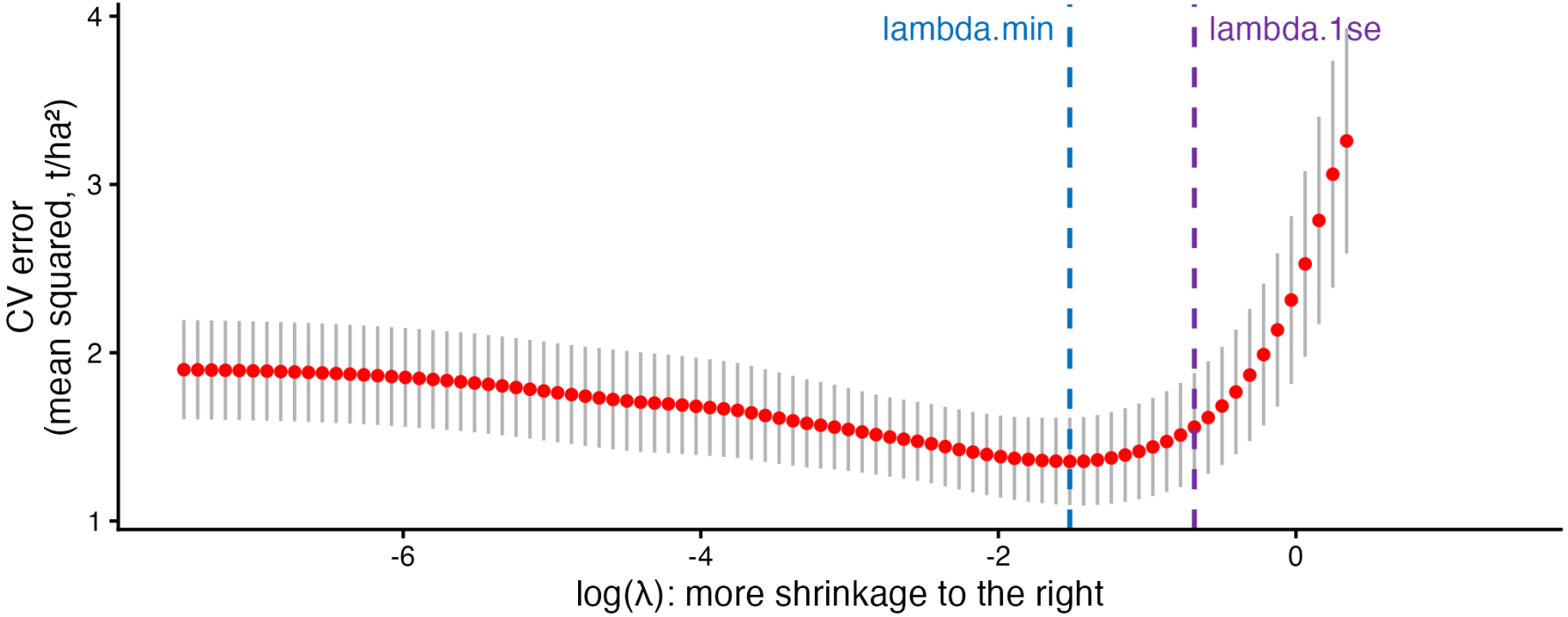
A 1
B 5
C 10
D 50

Scan with your phone

Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-folds

host screen

cv.glmnet(): the CV error along λ



lambda.min: lowest CV error. **lambda.1se**: simplest within 1 SE of it

lambda.min or lambda.1se?

```
> library(glmnet)
> set.seed(1)
> cv <- cv.glmnet(X, y, alpha = 1, nfolds = 10)
> cv
Call:  cv.glmnet(x = X, y = y, nfolds = 10, alpha = 1)
Measure: Mean-Squared Error
    Lambda Index Measure     SE Nonzero
min 0.2187    21   1.354 0.2601       1  ← best CV error
1se 0.5052    12   1.558 0.3212       1  ← simplest within 1 SE
```

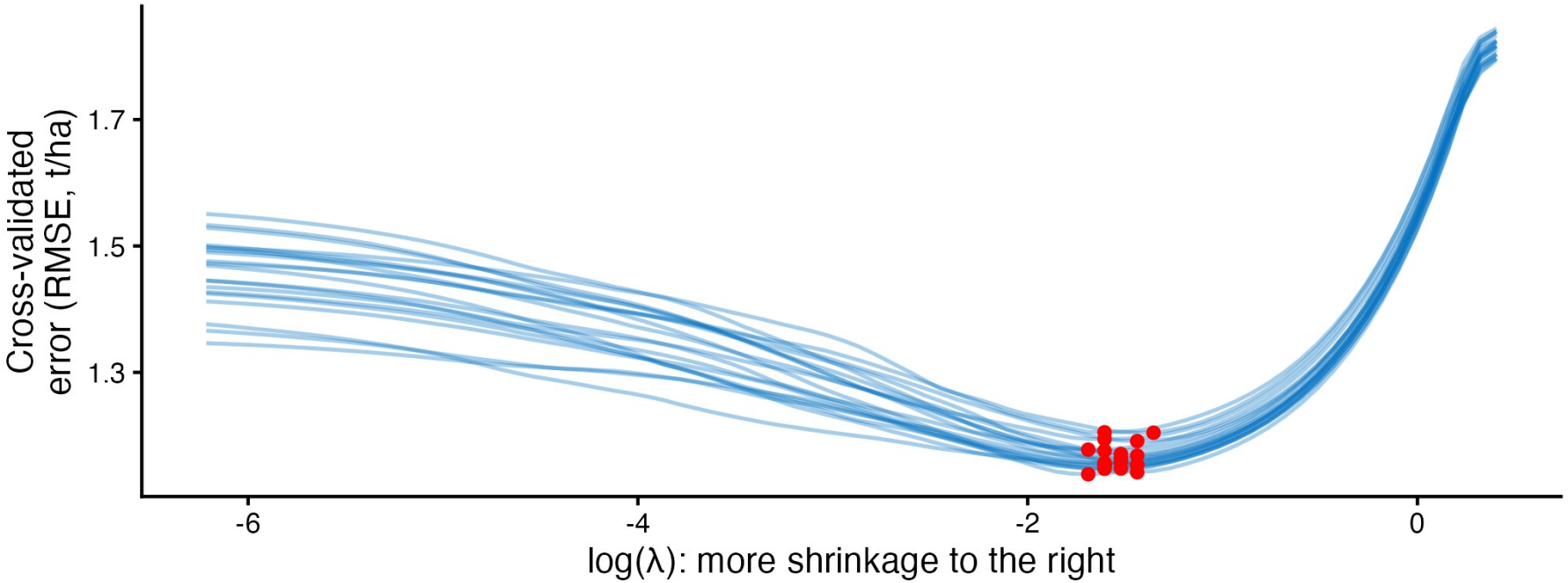
```
> b <- cbind(min = coef(cv, s = "lambda.min")[, 1],
+            se1 = coef(cv, s = "lambda.1se")[, 1])
> round(b[rowSums(b != 0) > 0, ], 3)
                min   se1
(Intercept) 4.253 4.253
gdd         1.199 0.910  ← shrunk more
```

predict **min**

short, stable list **1se**

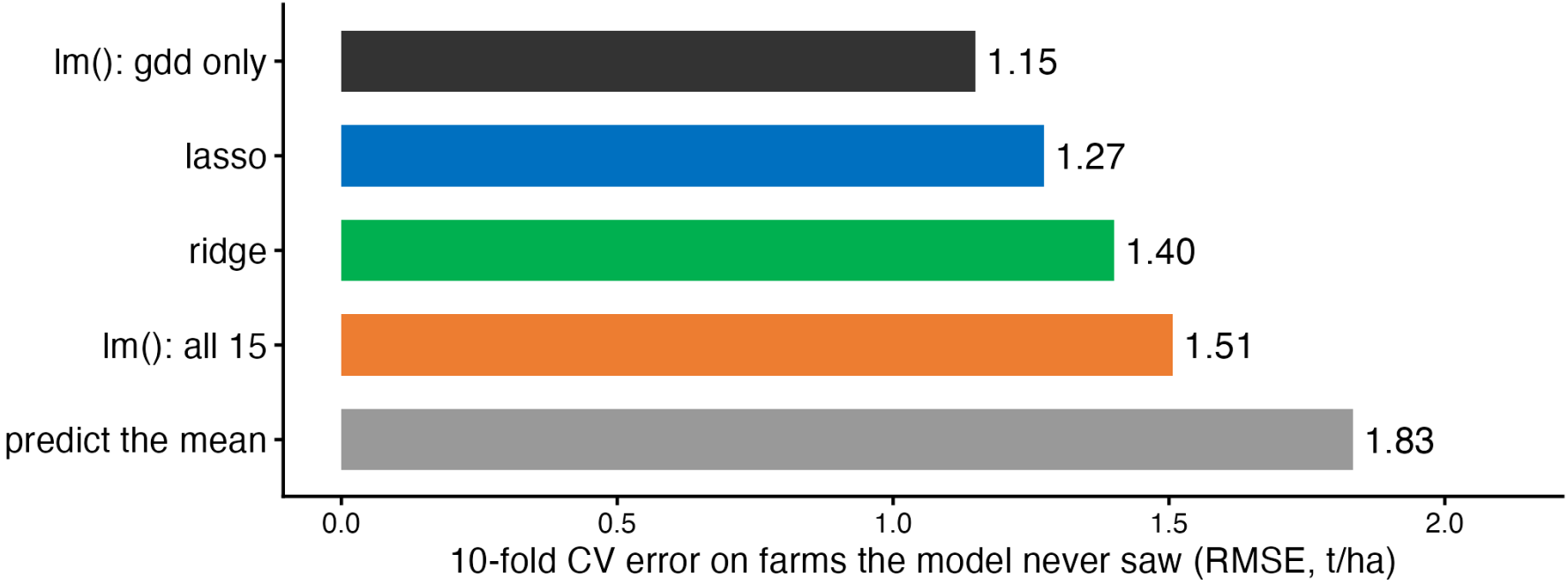
Both keep only gdd. Its slope: lm 1.90, min 1.20, 1se **0.91**

Folds are random: so is λ



20 different fold assignments: lambda.min from 0.18 to 0.26. **set.seed()** and report it

Compare whole recipes on unseen farms



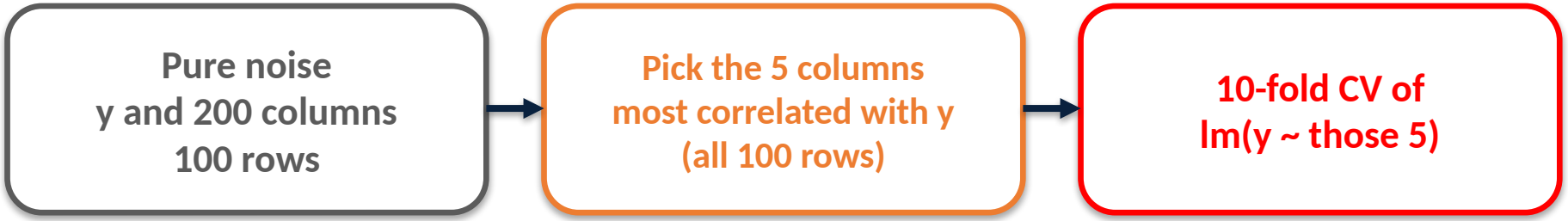
gdd alone 1.15, lasso 1.27, **all 15 1.51** t/ha. λ chosen inside each fold

1 Shrinkage
2 Mixed models
3 Ridge and lasso
4 Cross-validation
▶ **5 CV traps**
6 Classification





Trap: choose predictors first, then cross-validate



Nothing is related to anything. The honest answer: **no better than predicting the mean.**

A common recipe in papers: **screen, then validate**

Live poll

Multiple choice

Pure noise. Pick the 5 best of 200 columns using all rows, then 10-fold CV.

Compared with predicting the mean, the CV error is…

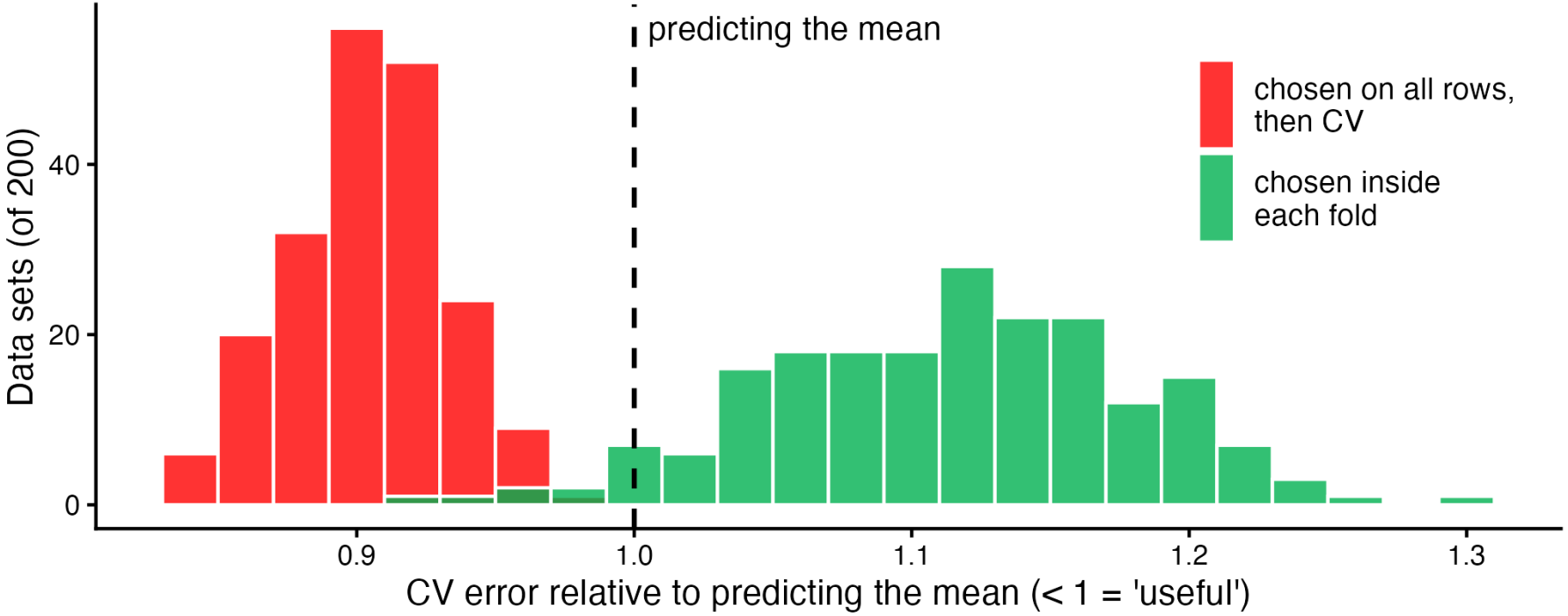
A Lower: looks useful
B About the same
C Higher

Scan with your phone

[Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-noise-cv](greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-noise-cv)

[host screen]

Screened outside the folds: fooled every time



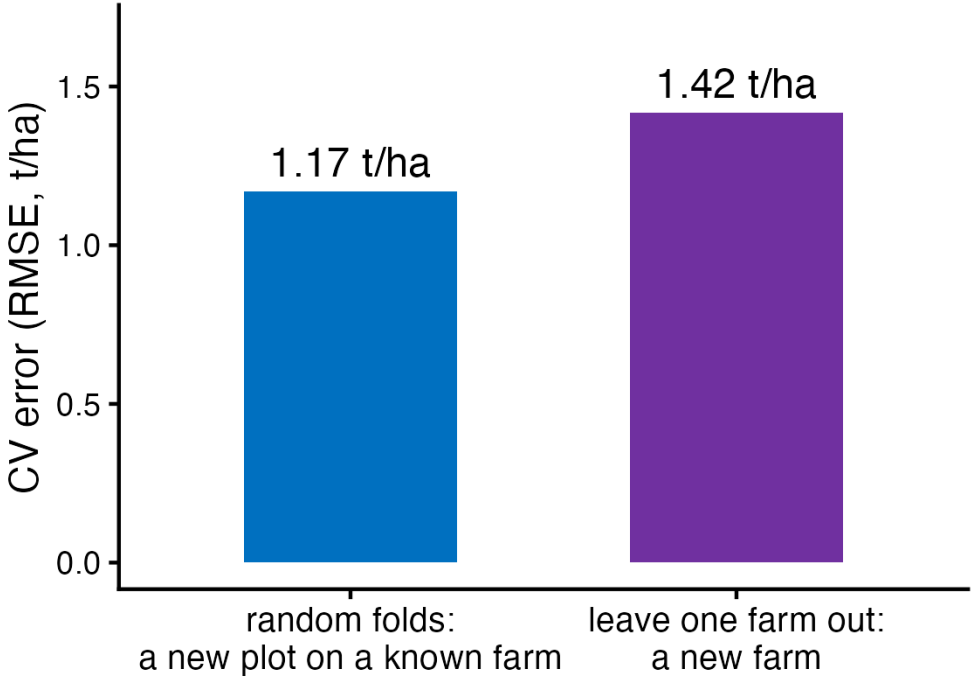
Outside: beats the mean in 100% of 200 data sets. **Inside**: 5%

The rule: steps that use y go inside the folds

choosing predictors	**inside each fold**
choosing λ	**inside (cv.glmnet within each fold)**
choosing a transformation	**inside**
a final test set	**touch it once, at the end**

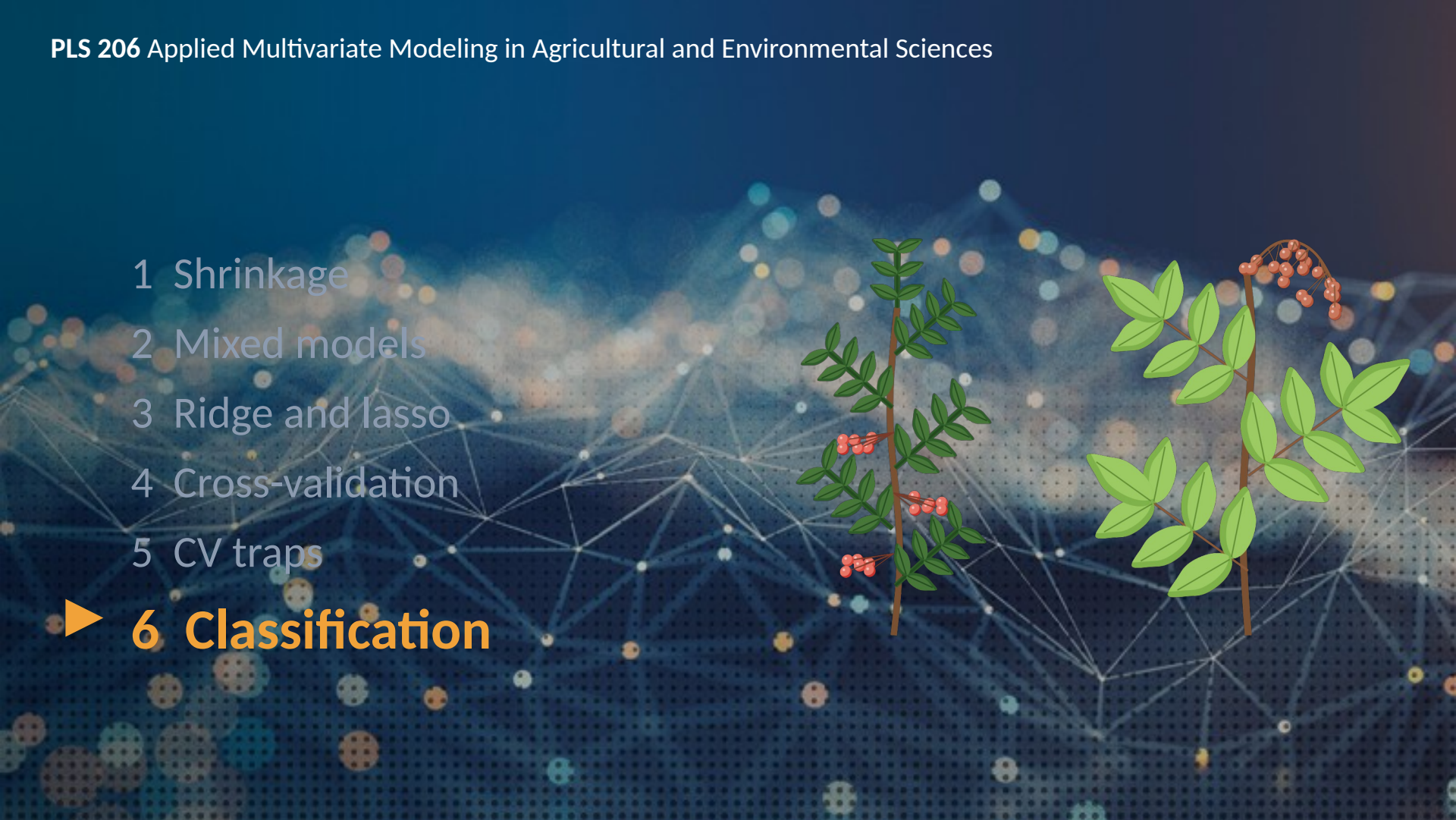
If a step looked at the left-out rows' y, they were **not left out**

Trap: rows that come in groups





Fold by the unit you will predict: a new farm, a new year, a new site



1 Shrinkage

2 Mixed models

3 Ridge and lasso

4 Cross-validation

5 CV traps

▶ **6 Classification**

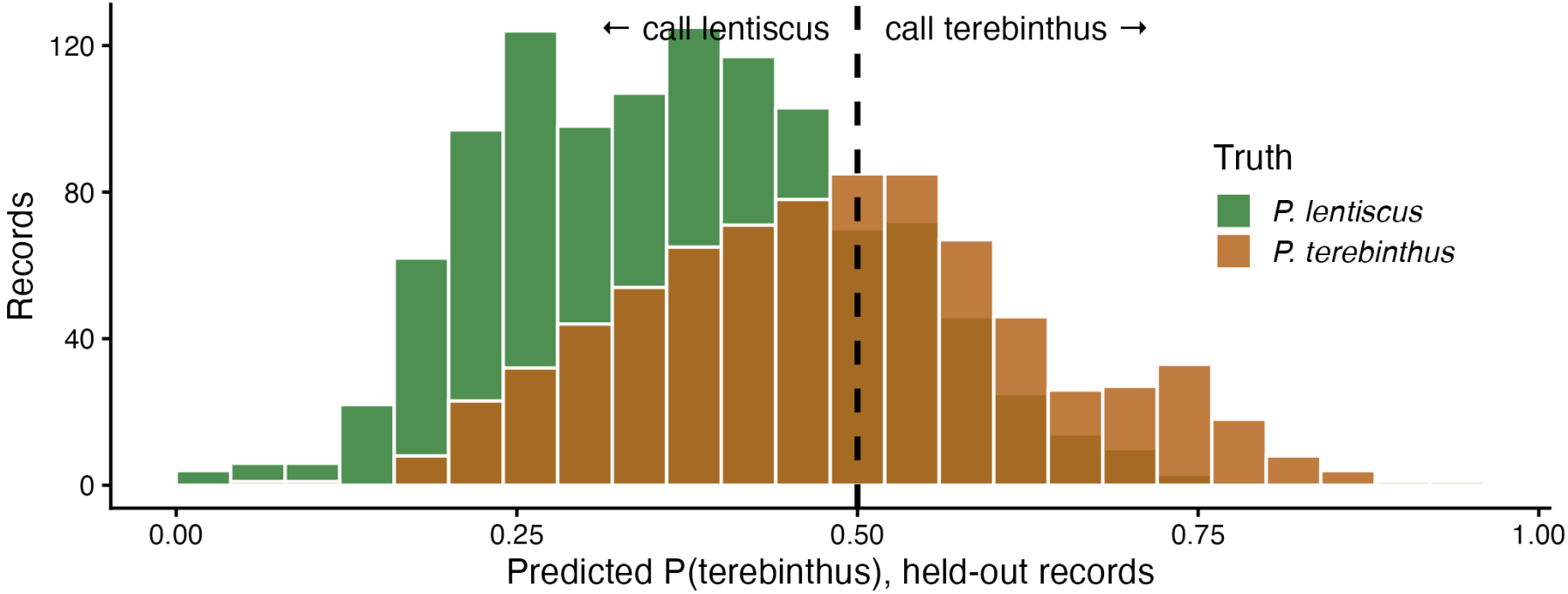
Predict the species from climate

```
> Xp <- scale(as.matrix(pis[, c(paste0("bio", 1:19),
"effort")]))
> set.seed(6)
> test <- sample(nrow(pis), round(0.3 * nrow(pis)))  # 30% held
out
> cvp <- cv.glmnet(Xp[-test, ], pis$y[-test], family =
"binomial",
+                  alpha = 1)
> b <- coef(cvp, s = "lambda.1se")[, 1]
> round(b[b != 0], 2)
(Intercept)        bio3        bio4        bio5        bio6
      -0.37        0.00        0.12       -0.05       -0.46  ← warmer winters: less terebinthus
        bio8        bio9       bio11       bio13       bio15
      -0.20        0.01       -0.08        0.06        0.11
     effort
       0.14
```



Logistic lasso on 20 predictors keeps **10**. 1,890 records held out

A probability, then a call



The model gives a **probability**. A call needs a **threshold** (here 0.5)

The confusion matrix

```
> p_hat <- predict(cvp, Xp[test, ], s = "lambda.1se", type =
"response")
> call <- ifelse(p_hat > 0.5, "terebinthus", "lentiscus")
> table(predicted = call, true = pis$species_short[test])
             true
predicted     lentiscus terebinthus
  lentiscus         911         417  ← right
  terebinthus       201         361  ← right
```

accuracy 67% called right

sensitivity 46% of terebinthus found

specificity 82% of lentiscus found

Sensitivity and **specificity** say which mistakes the model makes

Live poll

Multiple choice

41% of the held-out records are terebinthus.

A 'model' that always answers lentiscus is right...

A 0% of the time
B 41%
C 50%
D 59%

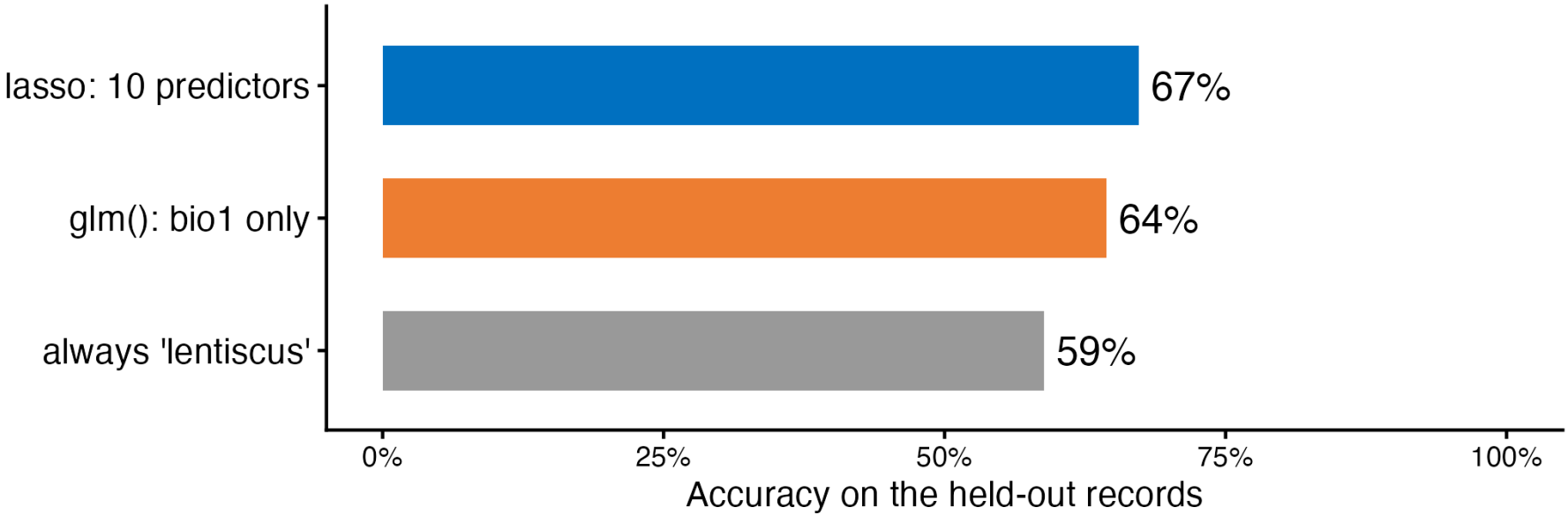


Scan with your phone

[Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-baseline](greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w6-baseline)

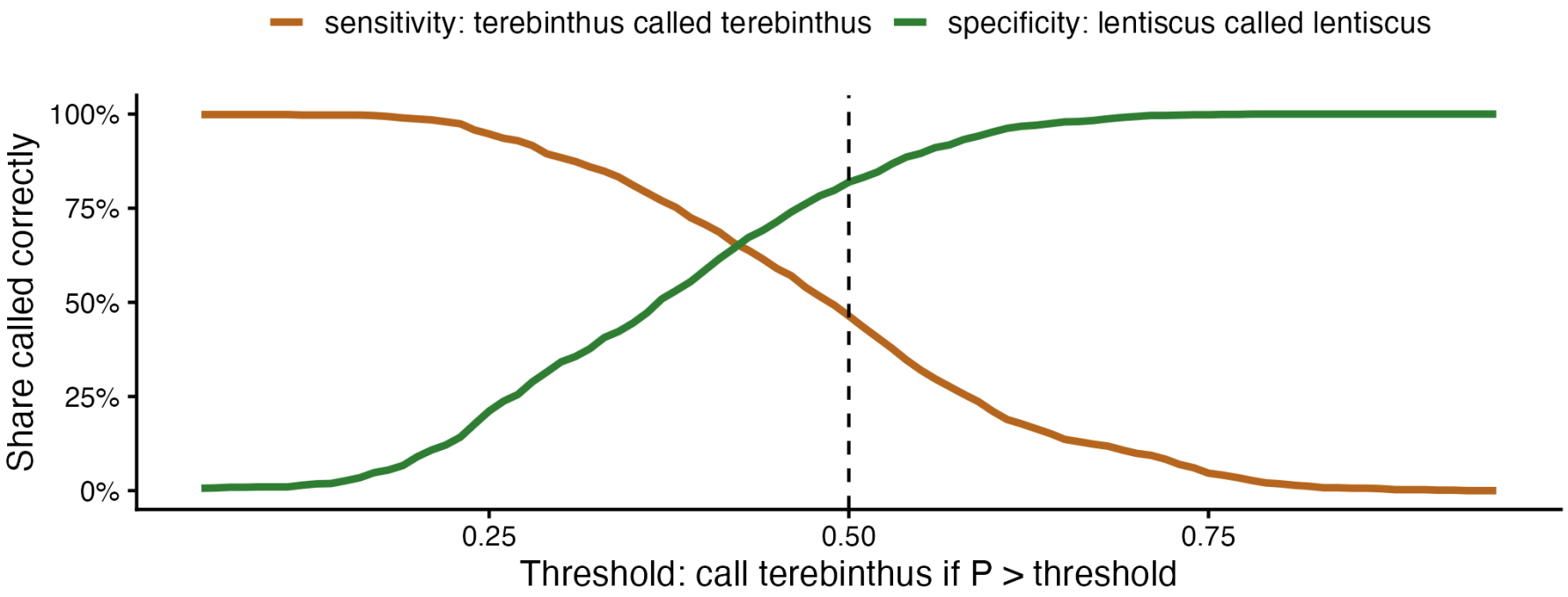
[host screen]

Accuracy needs a baseline



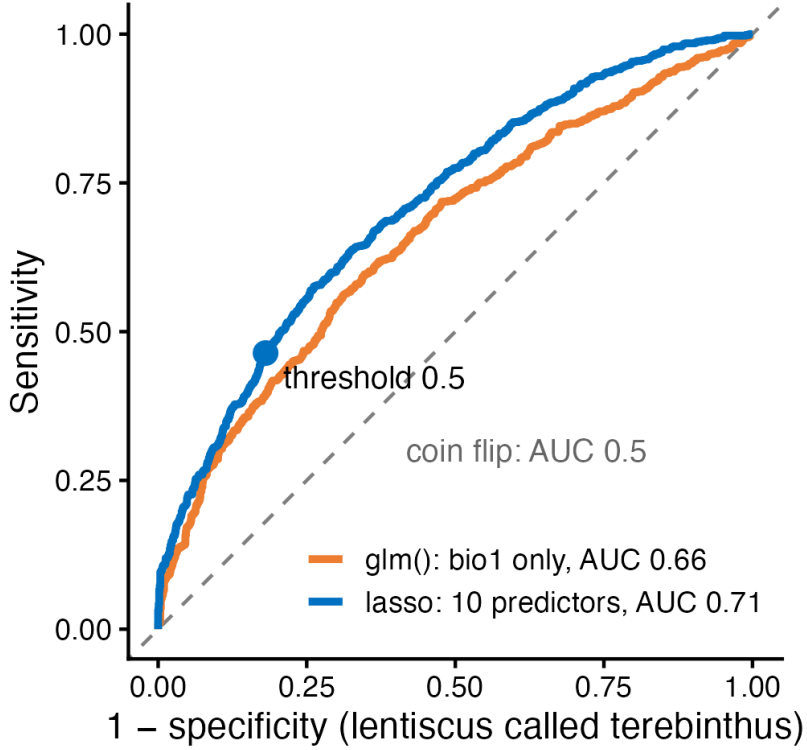
Always 'lentiscus': **59%**. The lasso adds **8 points**

Move the threshold: trade one mistake for the other



Threshold 0.3: sensitivity 88%, specificity 34%. **Which mistake costs more?**

ROC curve: every threshold at once



AUC 0.5 a coin flip

AUC 1 perfect

AUC P(a random terebinthus scores above a random lentiscus)

AUC: **bio1 alone 0.66**, **lasso 0.71** on held-out records

AUC by cross-validation

```
> library(pROC)
> auc(roc(pis$y[test], as.numeric(p_hat)))
Area under the curve: 0.7132  ← held-out records
> set.seed(6)
> cv_auc <- cv.glmnet(Xp[-test, ], pis$y[-test], family =
"binomial",
+                     type.measure = "auc")
> max(cv_auc$cvm)           # CV AUC on the training records
[1] 0.6986302  ← CV estimate
```

CV on the training records **0.70**, the untouched test set **0.71**: they agree

A number y: RMSE, MAE, R^2 on unseen data

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2} = 1.15 \text{ t/ha}$$

$$\text{MAE} = \frac{1}{n}\sum|y_i - \hat{y}_i| = 0.99 \text{ t/ha}$$

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} = 0.58$$

$\hat{y}_i$: the prediction for farm i from a model that never saw farm i (10-fold CV, lm(biomass $\sim$ gdd))

R^2 on the fitted farms 0.63; on unseen farms **0.58**

The week in one table

grouped data	`lmer(y ~ x + (1 | group))`
p-values for it	`library(lmerTest)`
many predictors	`glmnet(X, y, alpha = 0 or 1)`
choose λ	`cv.glmnet(): lambda.min / 1se`
judge predictions	k-fold CV; steps that use y inside folds
yes/no	baseline, sensitivity, specificity, AUC

Problem Set 6 due Monday 9 November, 1:10 PM

Live coding: cross-validation



`Week6_live_2_cv.R`

Part 2

1 k-fold CV by hand

2 cv.glmnet(): min and 1se

3 The screening trap

4 Leave one farm out

5 Pistacia: confusion matrix, ROC, AUC