

Non-linear regression

Bending the line

- 1 Bend the line
- 2 Splines
- 3 GAMs
- 4 Reading a GAM
- 5 GAM toolkit
- 6 Mechanistic models



Week 3 recap

Kahoot-style warm-up

10 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-w4-week3-recap](https://greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w4-week3-recap)

[host screen](#)

This week: five tools for curves

You want...

An optimum, or a test for a bend

How many polynomial degrees?

A curve inside lm / glm / lmer, fixed df

The shape, wiggleness chosen by the data

A parameter with biological meaning

Use

```
x + I(x^2)
```

```
poly(x, d)
```

```
ns(x, df)
```

```
gam(y ~ s(x))
```

```
nls()
```

Pick the tool by **the question you are asking**

► 1 Bend the line

2 Splines

3 GAMs

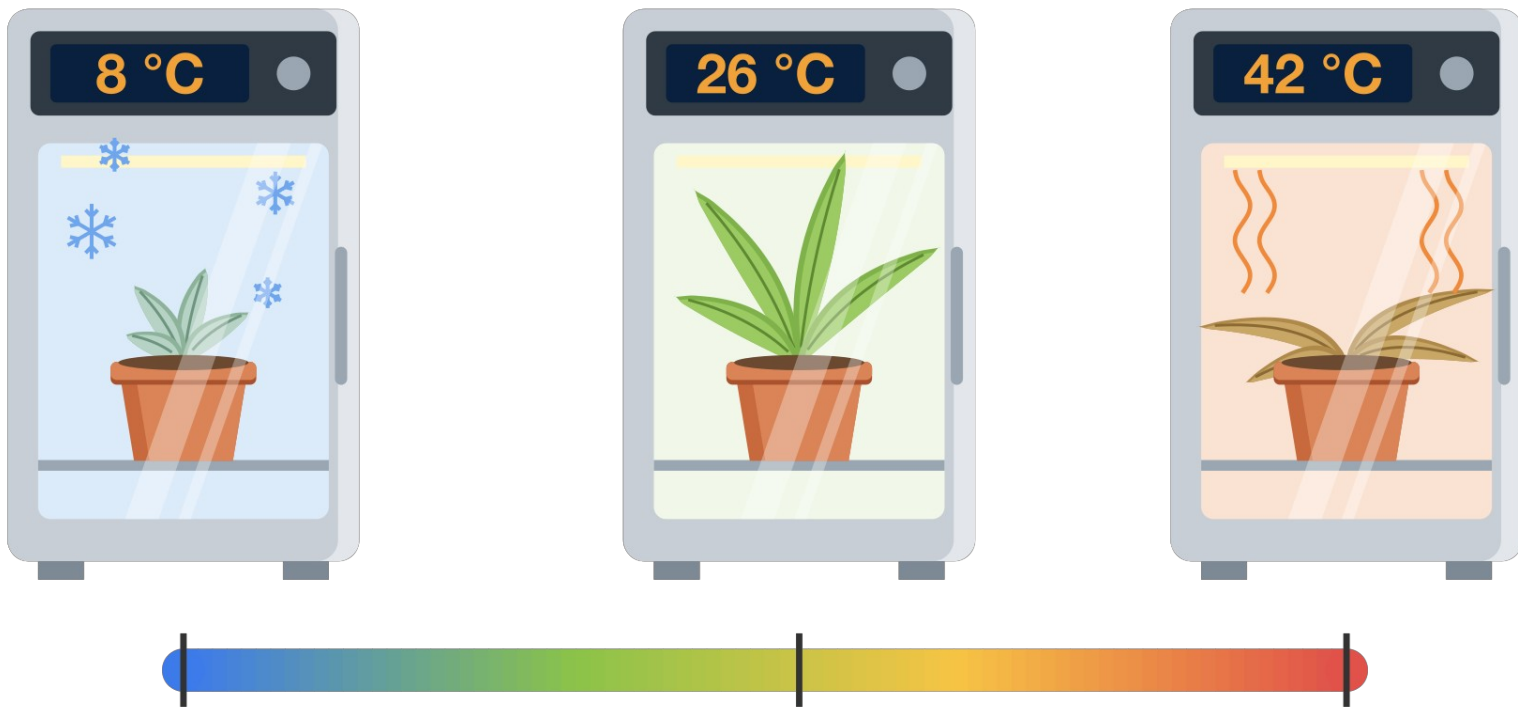
4 Reading a GAM

5 GAM toolkit

6 Mechanistic models



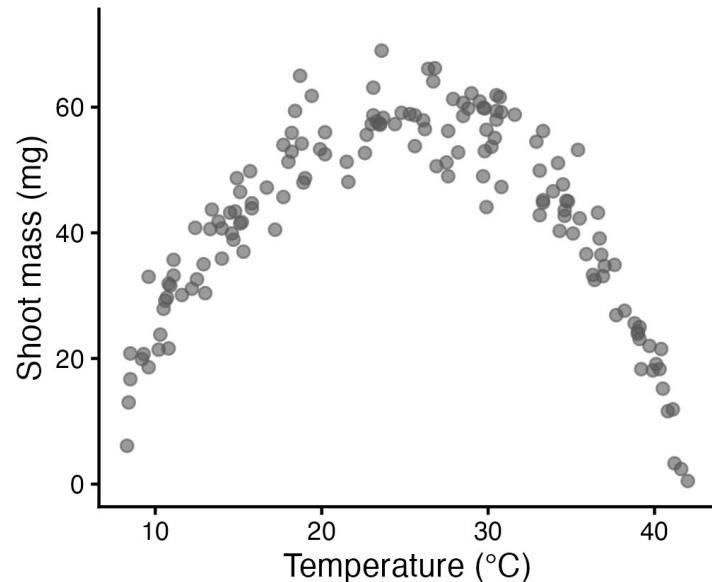
The experiment



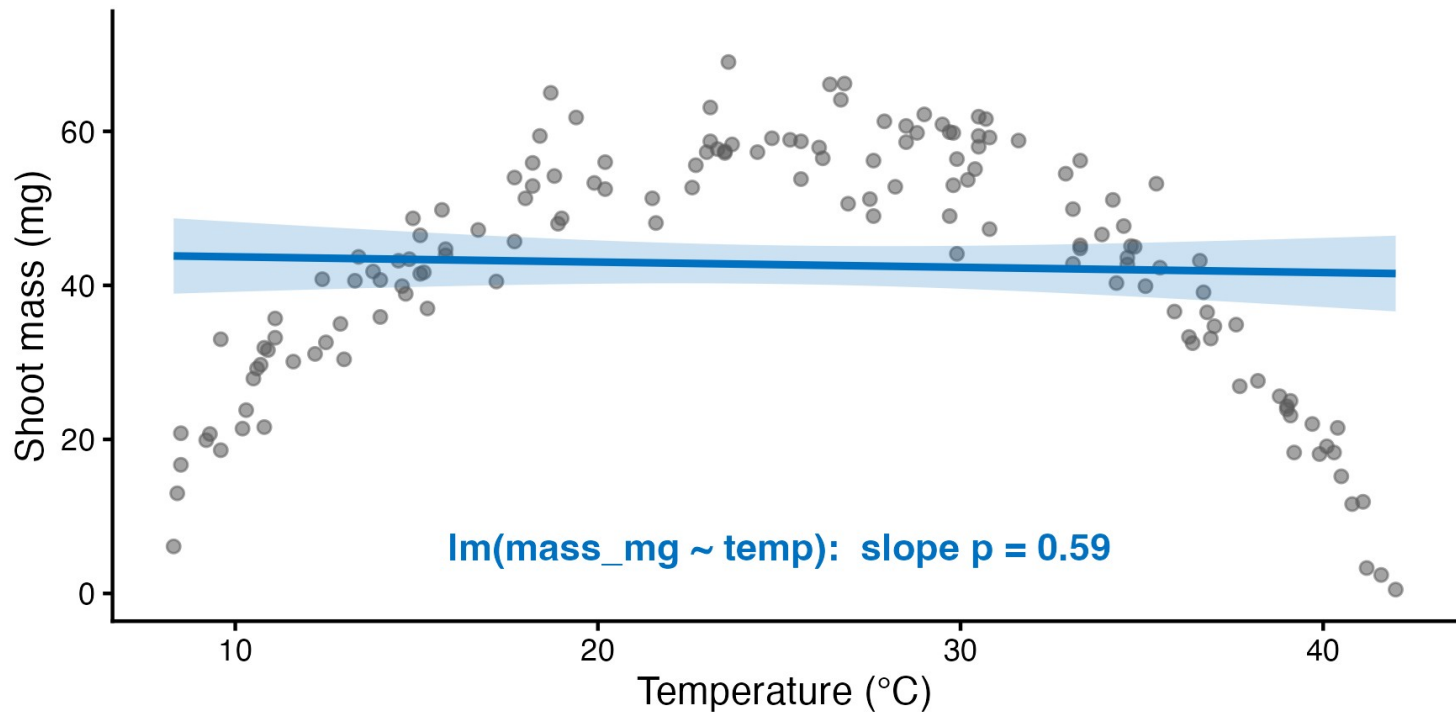
150 seedlings, 8 to 42 °C, 14 days. (Hypothetical, simulated data.)

Shoot mass vs temperature

```
> seedlings <- read.csv("https://greymonroe.github.io/PLS_206/data/seedling_temp.csv")
> std <- subset(seedlings, variety == "standard")
> head(std)
  pot variety temp mass_mg damaged
1   1 standard 12.9   35.0        1
2   2 standard 28.2   52.8        1
4   4 standard 15.1   46.5        0
5   5 standard 14.0   35.9        0
6   6 standard 28.5   60.7        0
9   9 standard 10.6   29.2        0
> nrow(std)
[1] 150 ← standard-variety pots
```



A straight line



Slope $p = 0.59$: does temperature not matter?

lm(mass_mg ~ temp)

```
> m_lin <- lm(mass_mg ~ temp, data = std)
> summary(m_lin)
```

Call:
lm(formula = mass_mg ~ temp, data = std)

Residuals:

Min	1Q	Median	3Q	Max
-41.038	-10.882	2.596	12.935	26.218

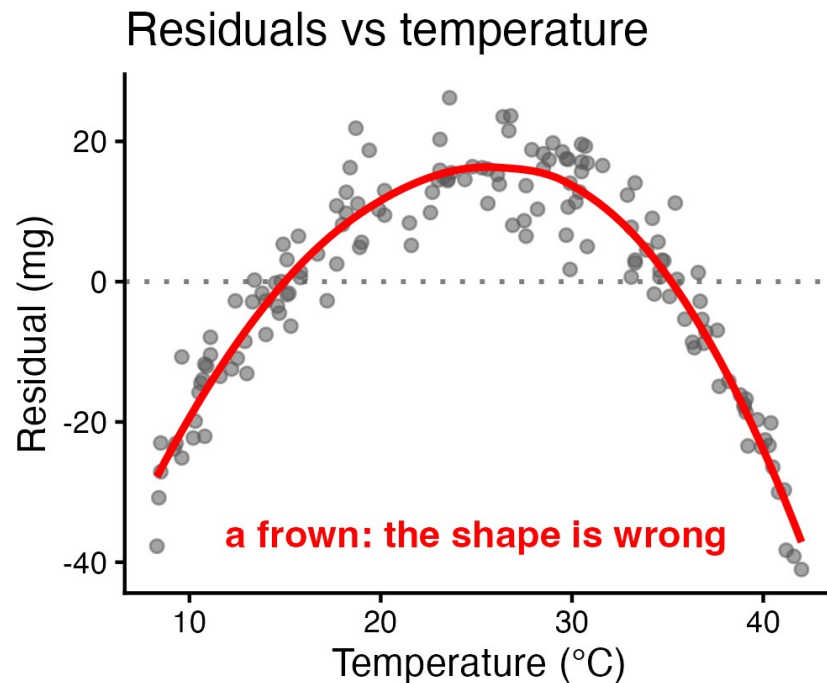
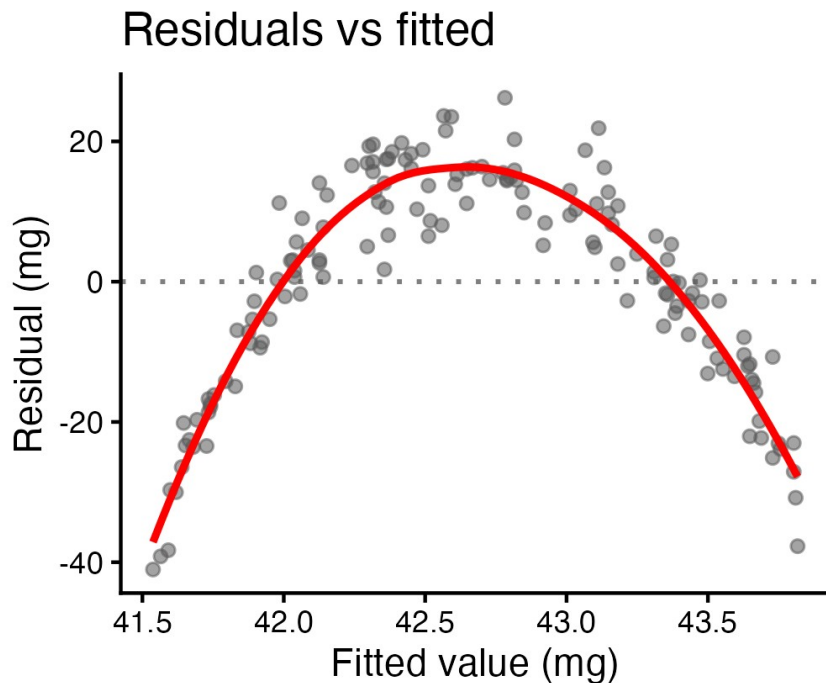
Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	44.37849	3.42270	12.966	<2e-16 ***
temp	-0.06764	0.12660	-0.534	0.594 ← slope p

Residual standard error: 15.54 on 148 degrees of freedom
Multiple R-squared: 0.001925, Adjusted R-squared: -0.004819
F-statistic: 0.2855 on 1 and 148 DF, p-value: 0.5939

$R^2 = 0.002$: the line explains almost nothing

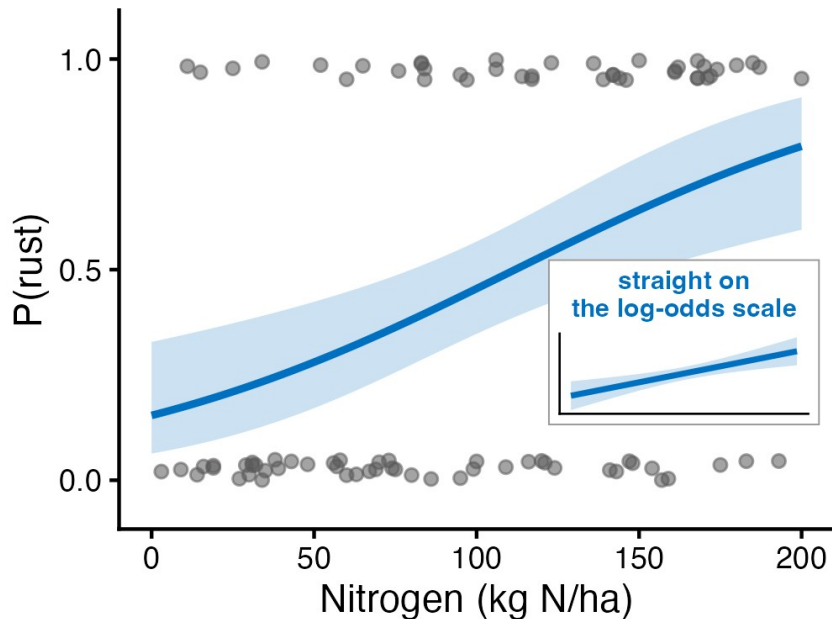
Residuals: always look



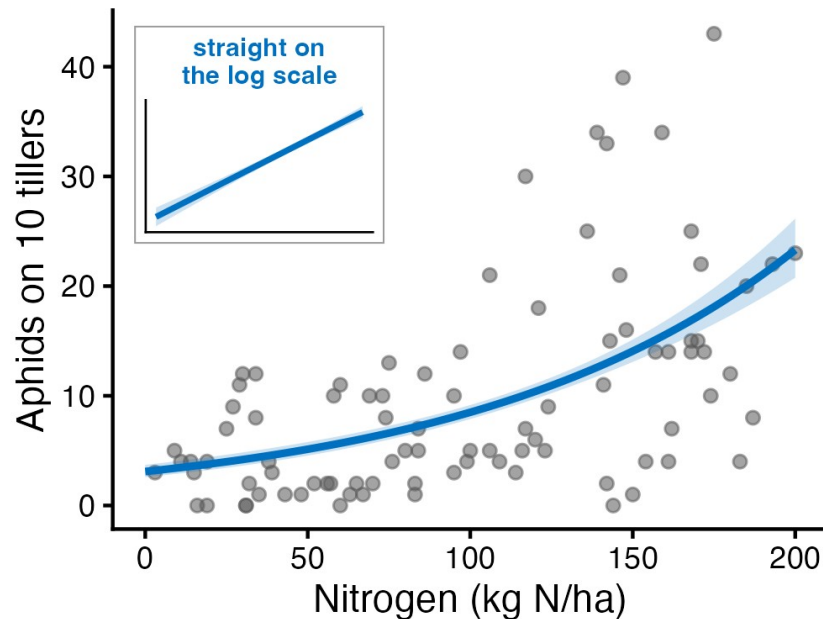
A frown in the residuals: the shape is wrong

You already fit curves last week

Logistic: an S-curve



Poisson: an exponential curve



Straight on the **link scale**, curved on the scale of y

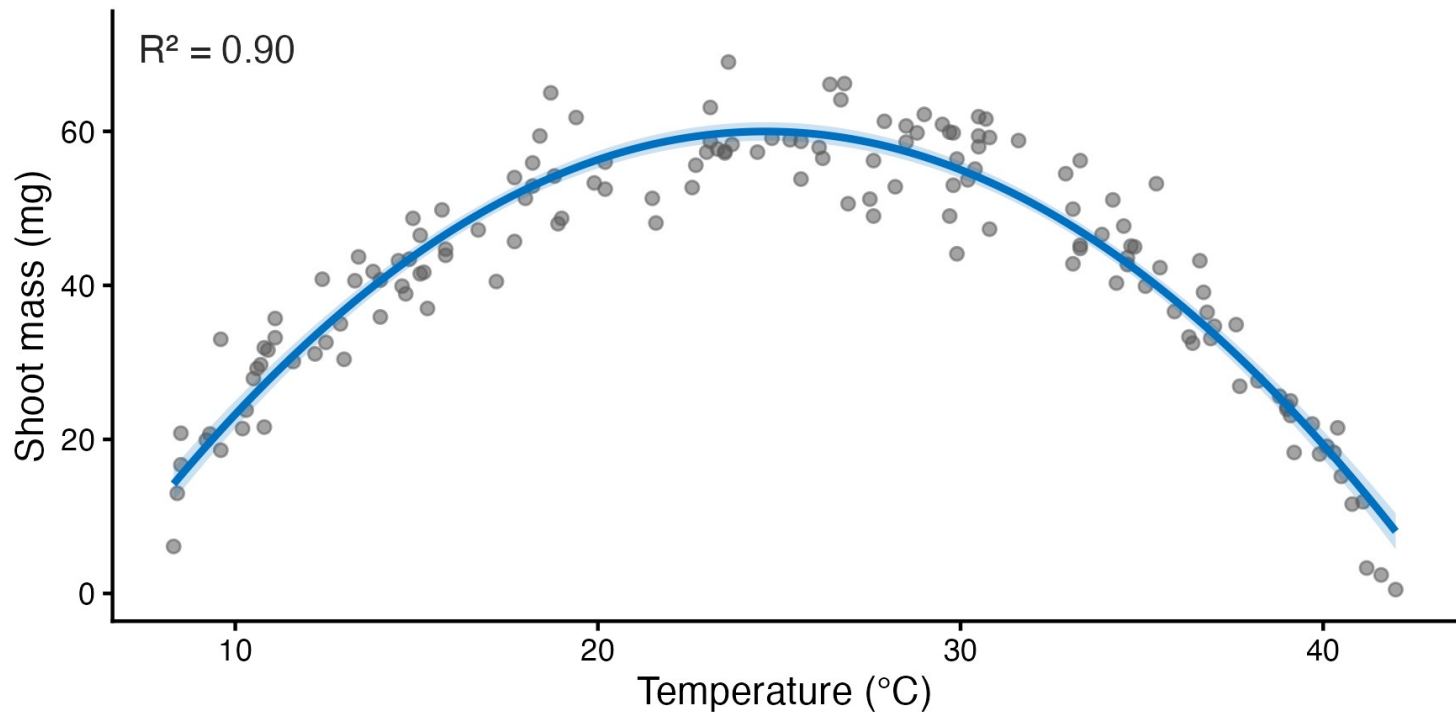
The whole trick: a new column

$$\text{mass} = \beta_0 + \beta_1 \text{temp} + \beta_2 \text{temp}^2$$

β_0 : intercept β_1, β_2 : coefficients that `lm()` estimates temp^2 : a new column made from temp

```
> head(cbind(temp = std$temp, temp2 = std$temp^2))
      temp temp2 ← a new column
[1,] 12.9 166.41 ← = 12.9 x 12.9
[2,] 28.2 795.24
[3,] 15.1 228.01
[4,] 14.0 196.00
[5,] 28.5 812.25
[6,] 10.6 112.36
```

One extra column: a curve



temp² bends the line into a parabola

lm(mass_mg ~ temp + I(temp^2))

```
> m_quad <- lm(mass_mg ~ temp + I(temp^2), data = std)
> summary(m_quad)
```

Call:

```
lm(formula = mass_mg ~ temp + I(temp^2), data = std)
```

Residuals:

Min	1Q	Median	3Q	Max
-11.0866	-2.8436	-0.5789	3.2740	13.1977

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-44.1887	2.5967	-17.02	<2e-16	***
temp	8.4626	0.2316	36.53	<2e-16	***
I(temp^2)	-0.1719	0.0046	-37.36	<2e-16	*** ← negative: bends down

Residual standard error: 4.812 on 147 degrees of freedom

Multiple R-squared: **0.9049** Adjusted R-squared: 0.9036

F-statistic: 699.5 on 2 and 147 DF, p-value: < 2.2e-16

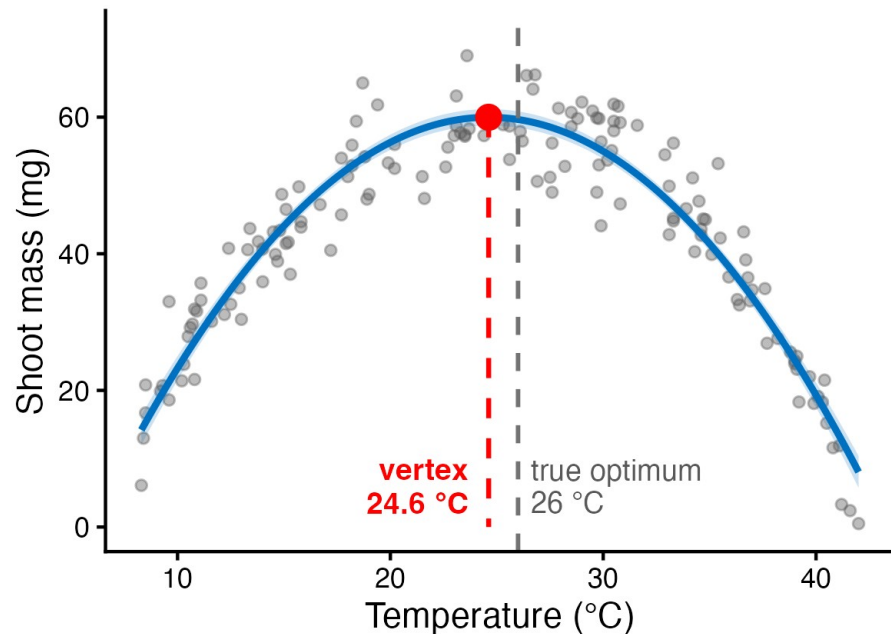
temp² is negative: the curve bends down

Top of the hill

$$\text{temp}^* = -\frac{\beta_1}{2\beta_2}$$
$$= -\frac{8.463}{2 \times (-0.1719)} = 24.6 \text{ } ^\circ\text{C}$$

temp*: the temperature at the top of the parabola

(where the slope is zero)



Estimated optimum **24.6 °C**; the truth is 26 °C

Live poll

Number: live histogram

Maize N trial:

$$\text{yield} = \beta_0 + 0.06 \cdot N - 0.0002 \cdot N^2$$

Optimal N rate (kg N/ha)?

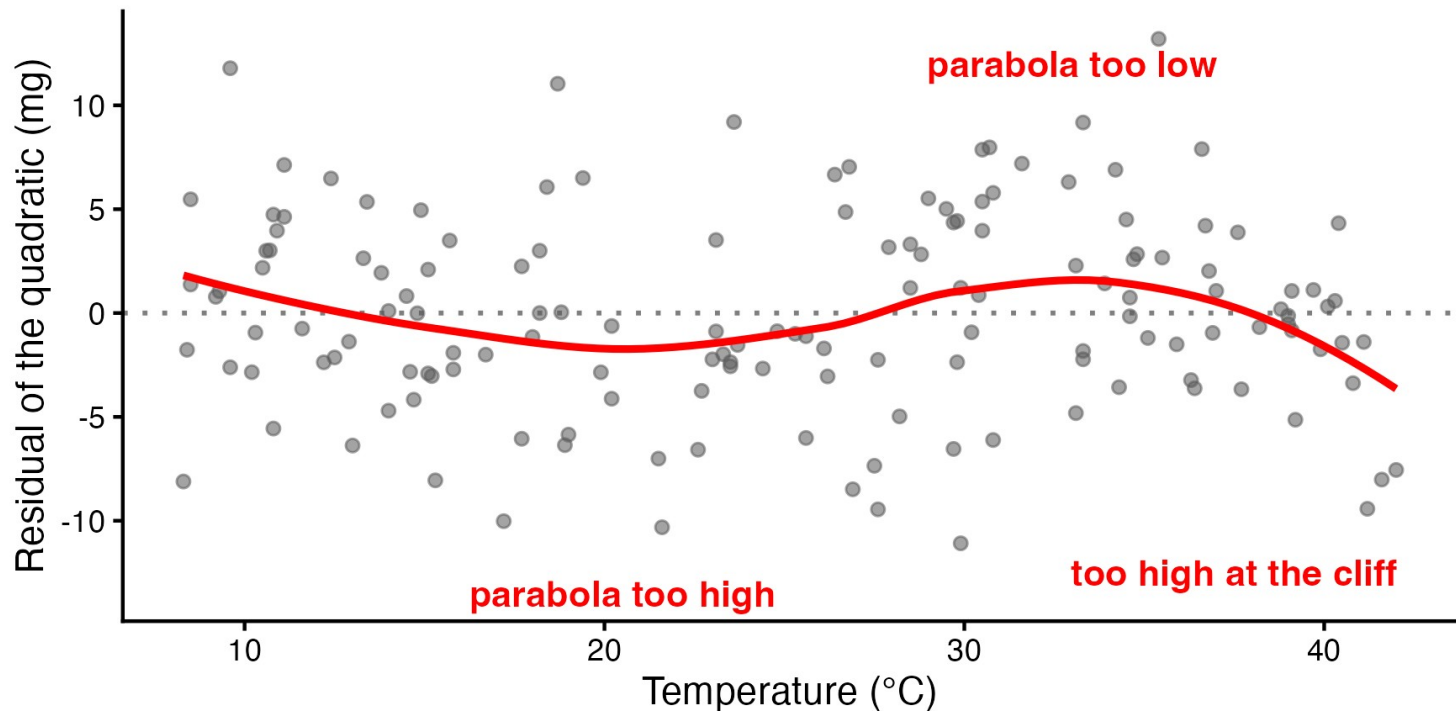


Scan with your phone

[Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w4-vertex](https://greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w4-vertex)

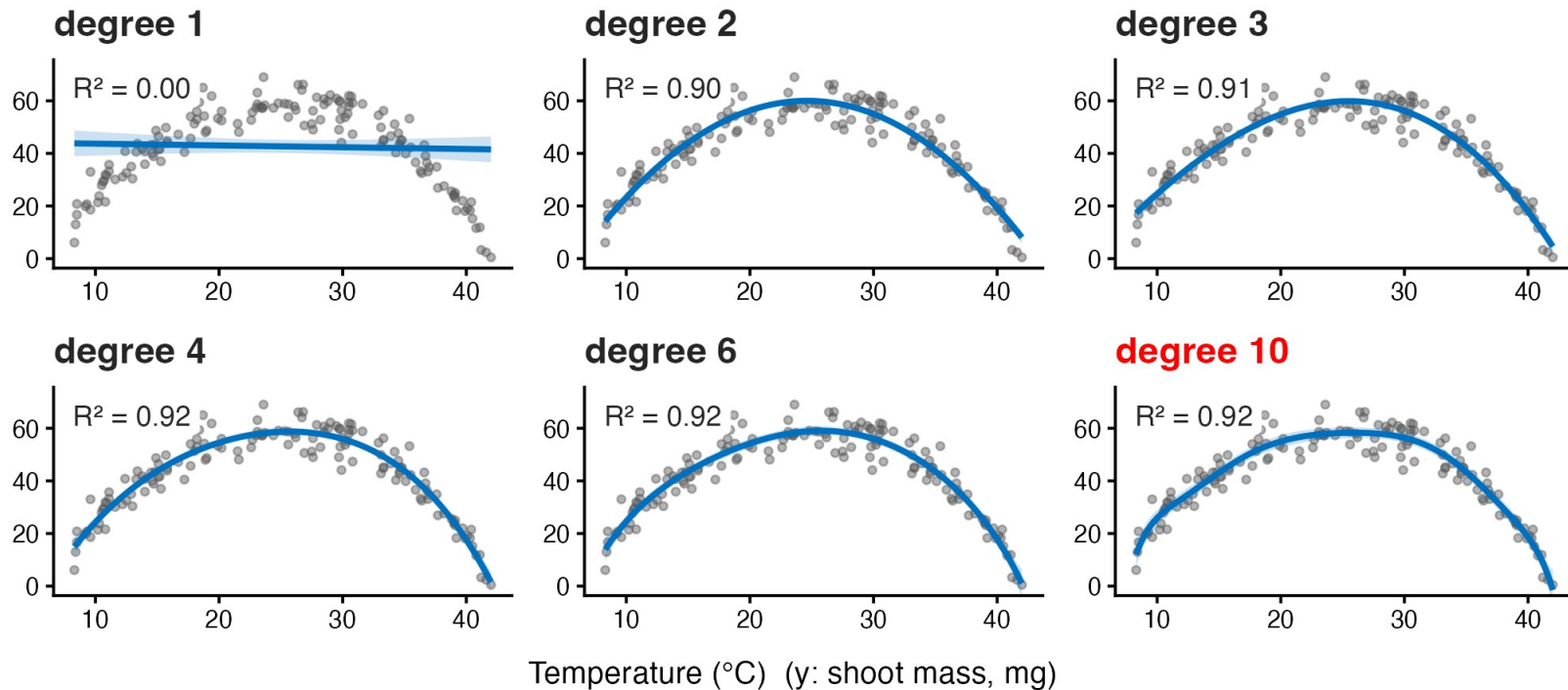
[host screen](#)

A parabola is symmetric



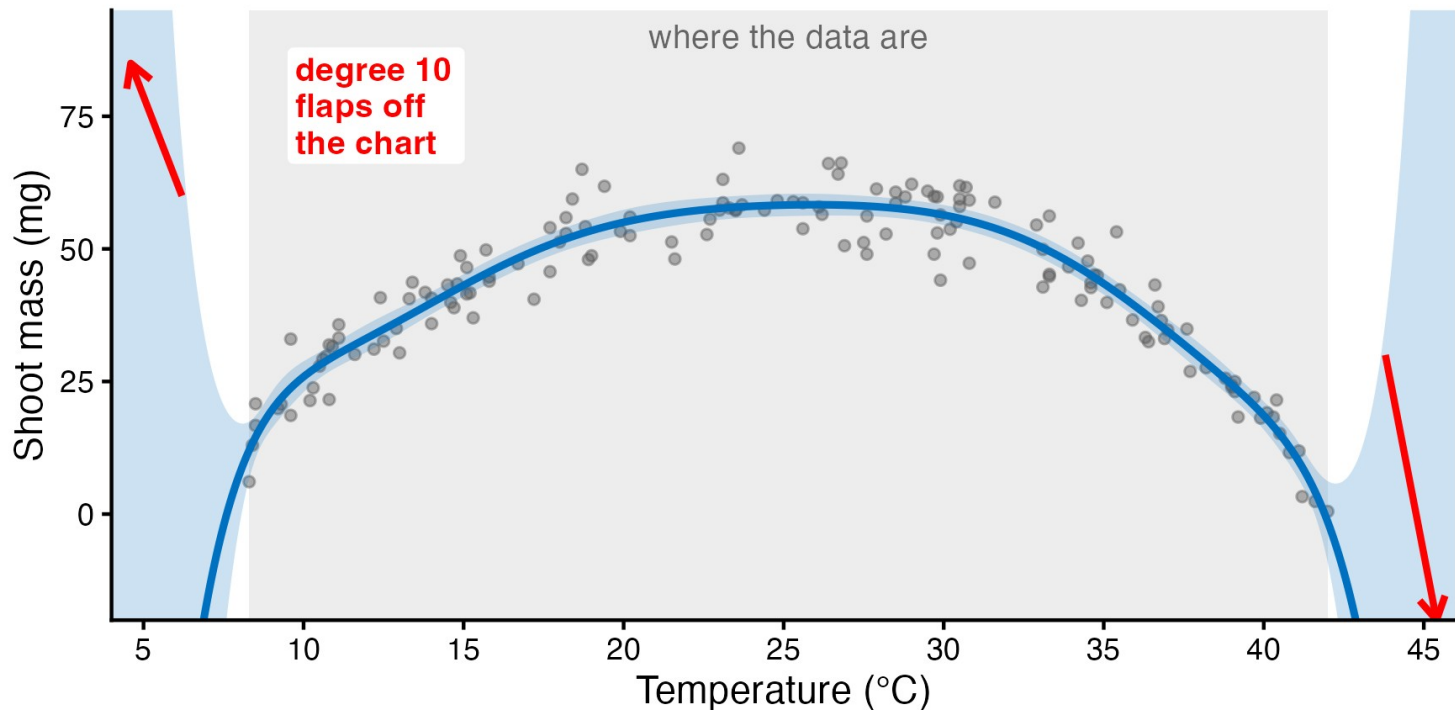
Seedling growth **isn't**: slow rise, fast crash

More columns: temp^3 , temp^4 , ...



Each extra power lets the curve bend **once more**

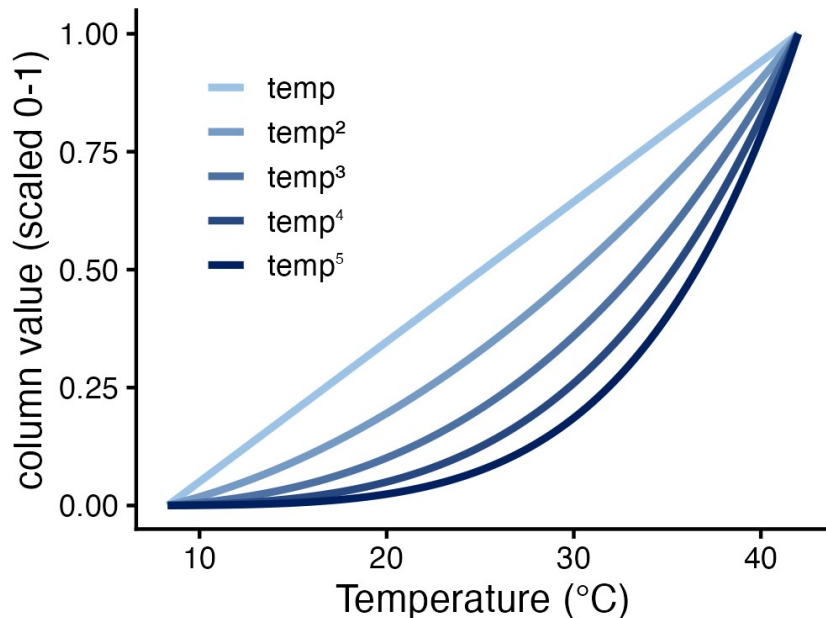
High-degree polynomials flap at the edges



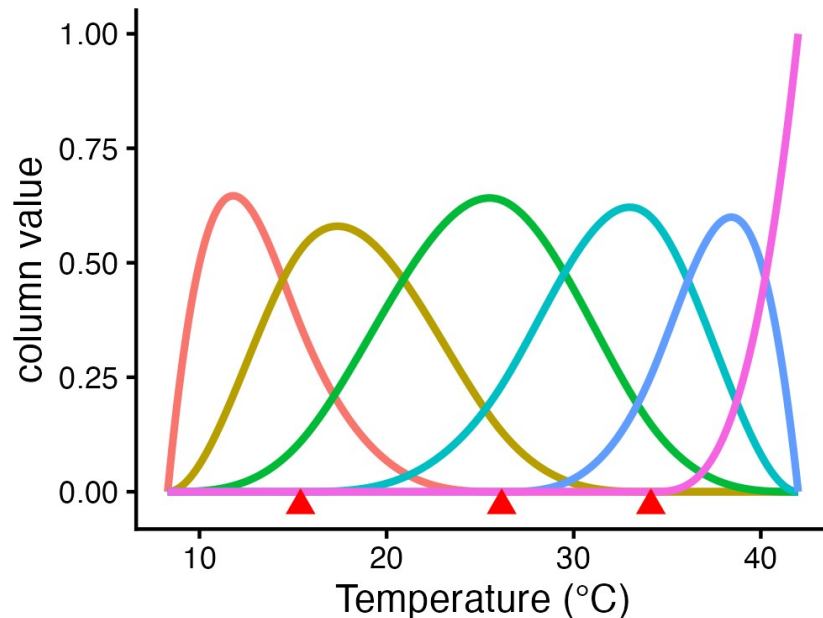
Step outside the data and the prediction **explodes**

Global columns vs local columns

Polynomial columns: global



Spline columns: local

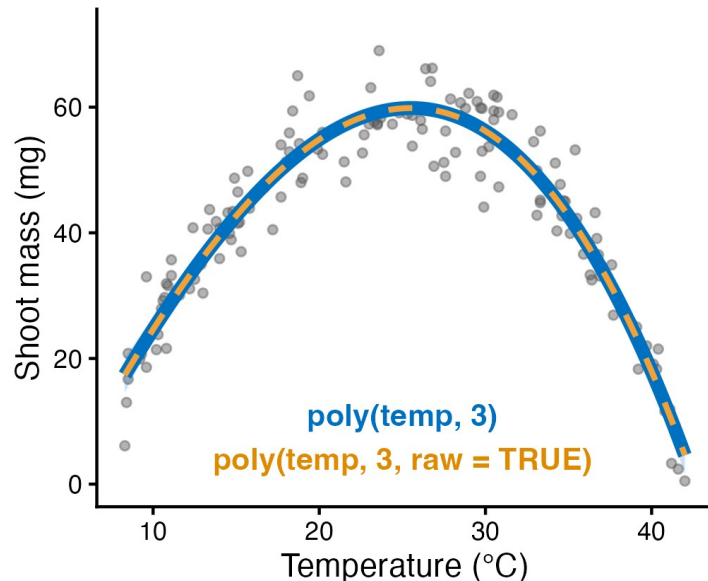


Polynomial columns are global; **spline** columns are local

poly(): raw or orthogonal?

```
> m_orth <- lm(mass_mg ~ poly(temp, 3), data = std)
> round(coef(m_orth), 3)
      (Intercept) poly(temp, 3)1 poly(temp, 3)2 poly(temp, 3)3
          42.680      -8.301      -179.788      -16.005
```

```
> m_raw <- lm(mass_mg ~ poly(temp, 3, raw = TRUE), data = std)
> signif(coef(m_raw), 3)
      (Intercept) poly(temp, 3, raw = TRUE)1
          -23.90000           5.41000
poly(temp, 3, raw = TRUE)2 poly(temp, 3, raw = TRUE)3
          -0.03790           -0.00178
> all.equal(fitted(m_orth), fitted(m_raw))
[1] TRUE ← same fitted values
```



Same curve, different coefficients. **raw = TRUE** if you will read them

Live coding: lines, residuals, quadratics



`Week4_live_1_curves.R`

Part 1: sections 1 to 4

- 1 The data
- 2 A straight line and its residuals
- 3 A quadratic: one new column
- 4 Higher polynomials

1 Bend the line

▶ 2 Splines

3 GAMs

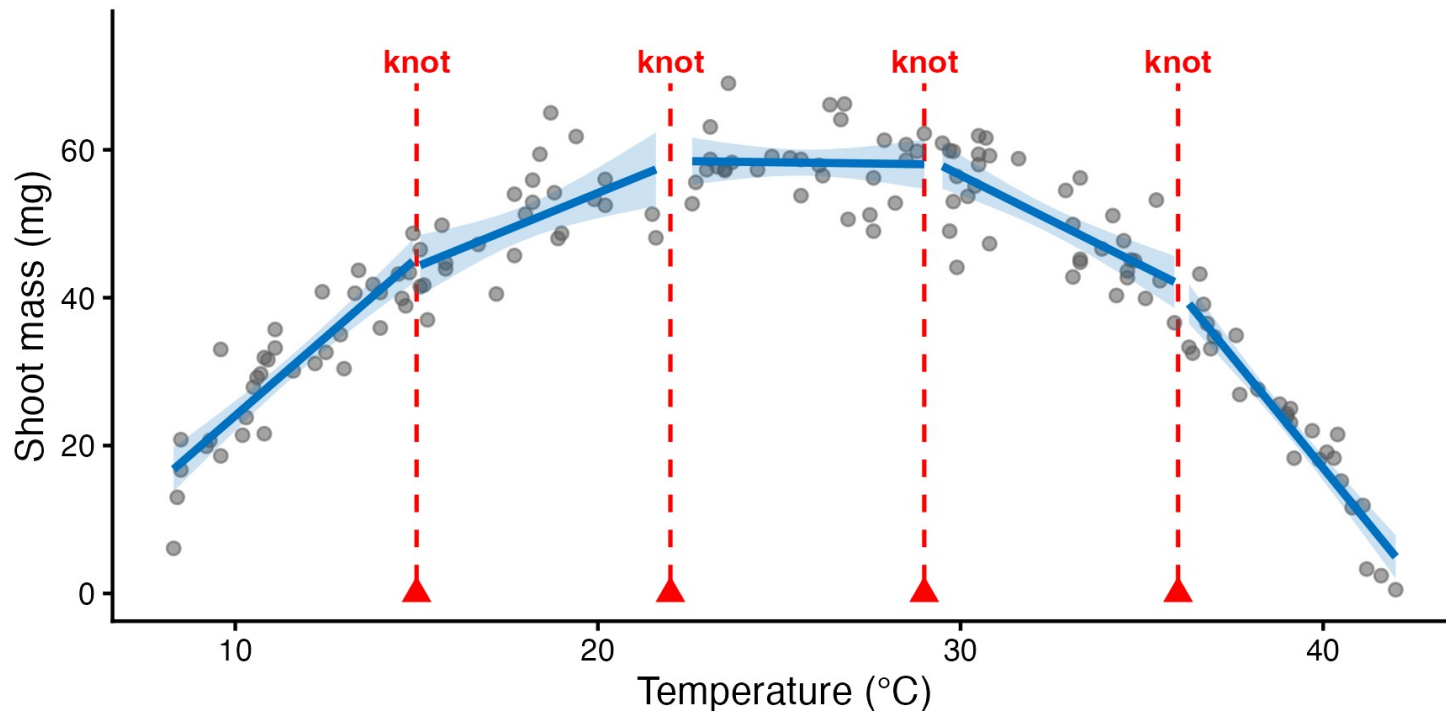
4 Reading a GAM

5 GAM toolkit

6 Mechanistic models

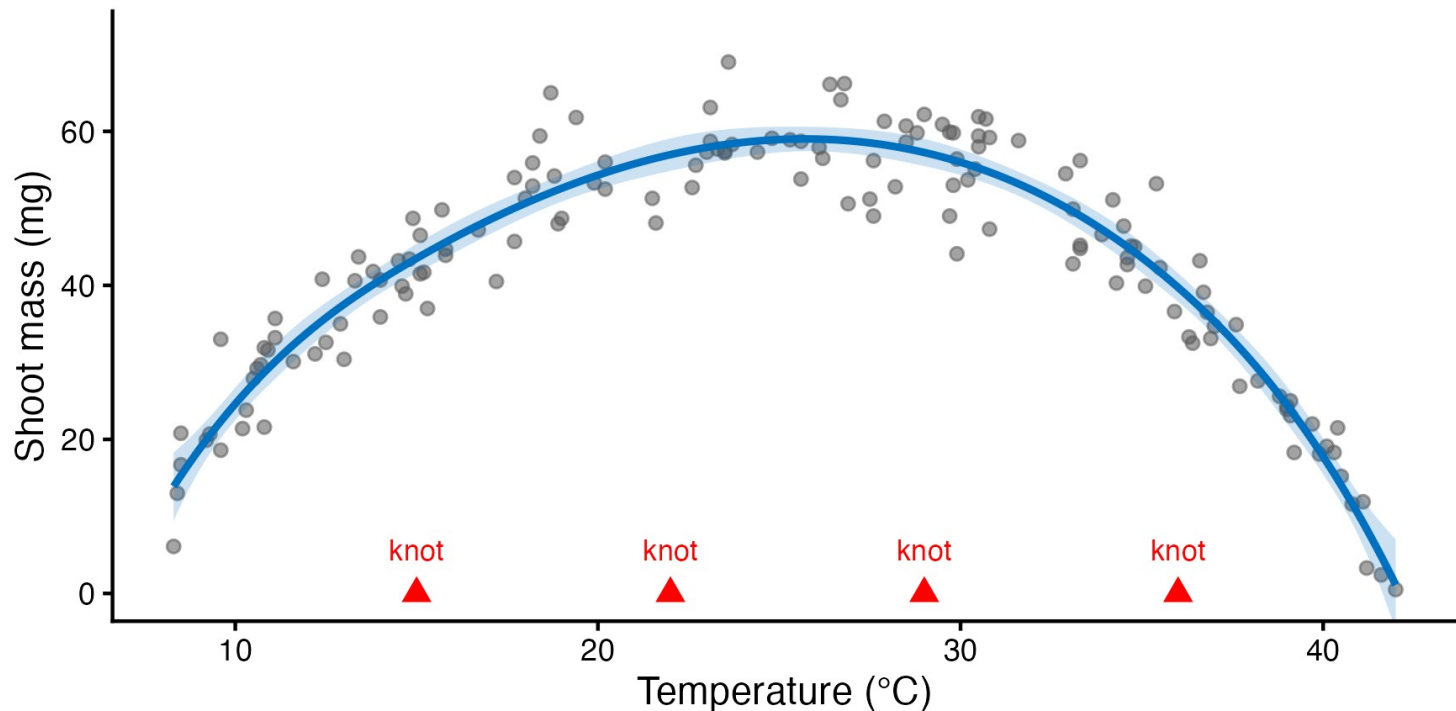


Fit the curve in pieces



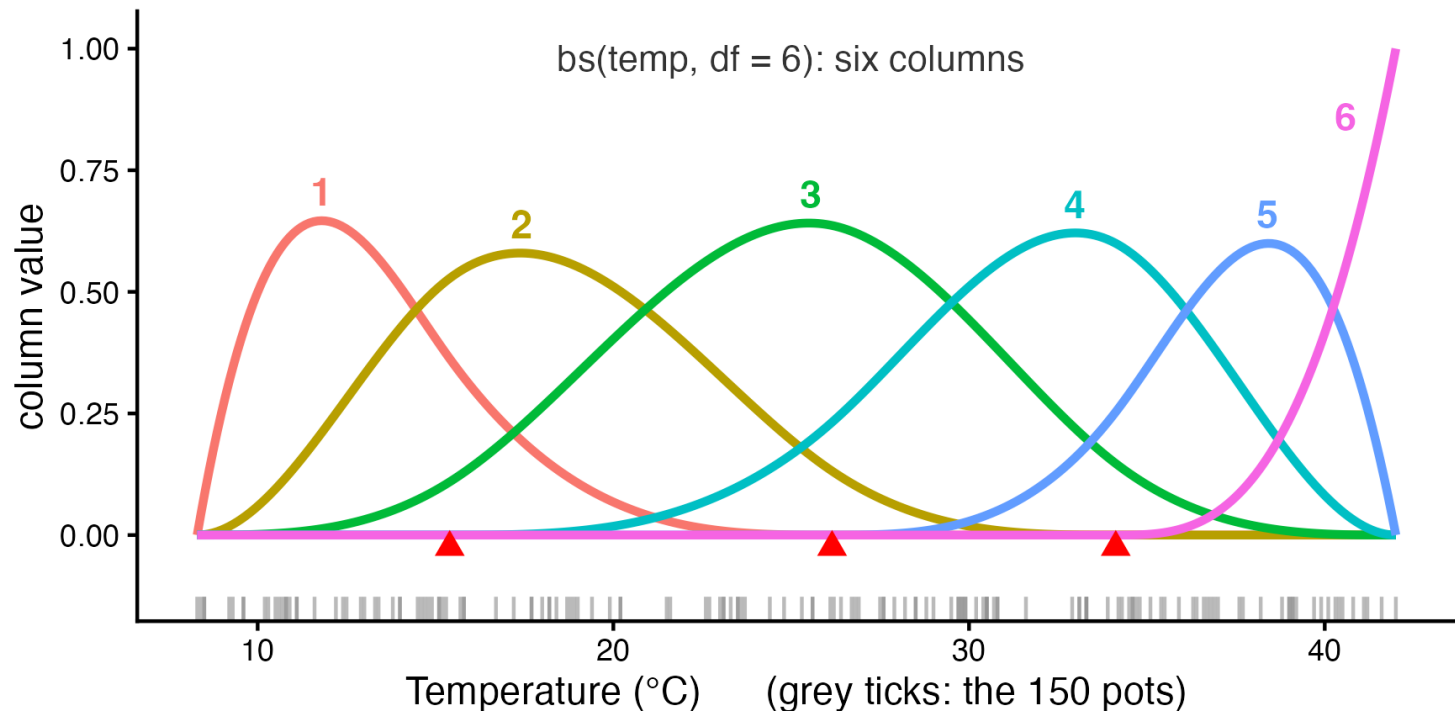
Cut x at **knots**: each piece listens only to its own data

Join the pieces smoothly



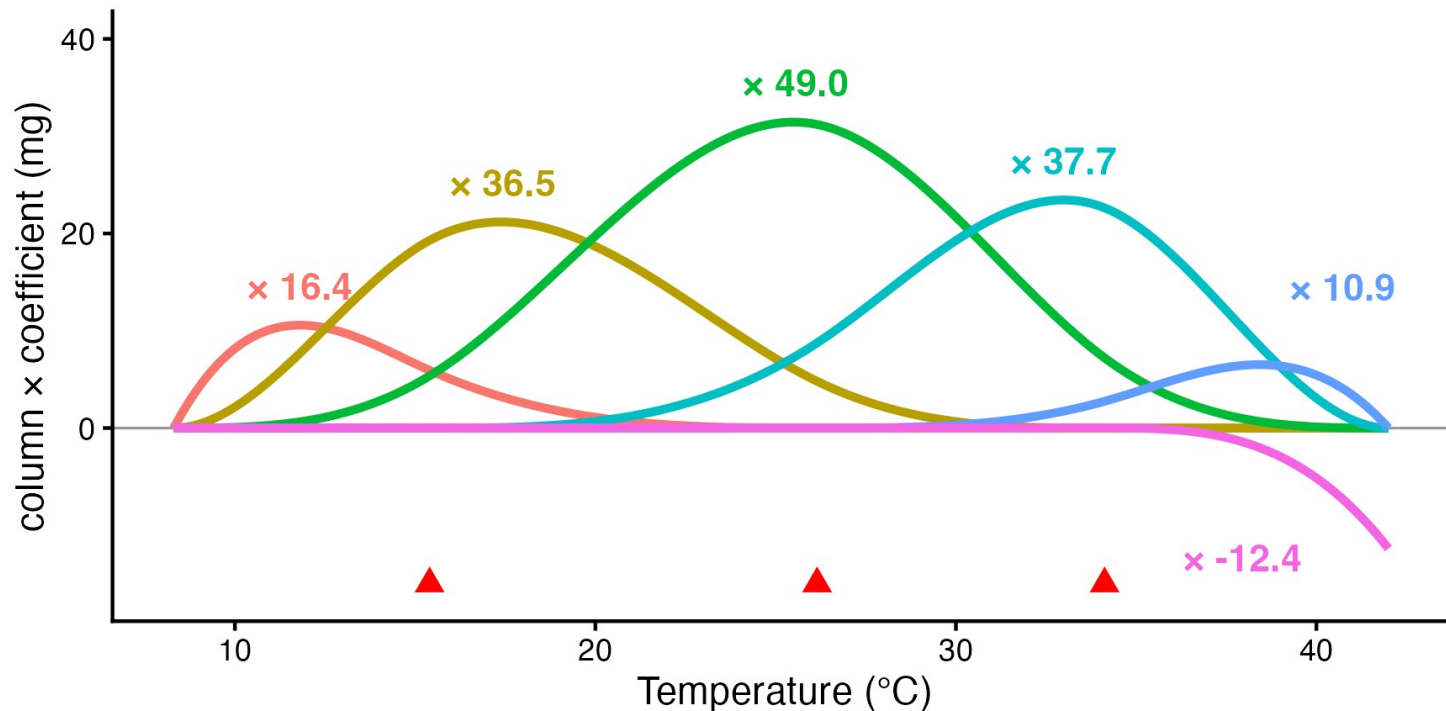
Same height, slope and curvature **at every knot**

Each column is a bump



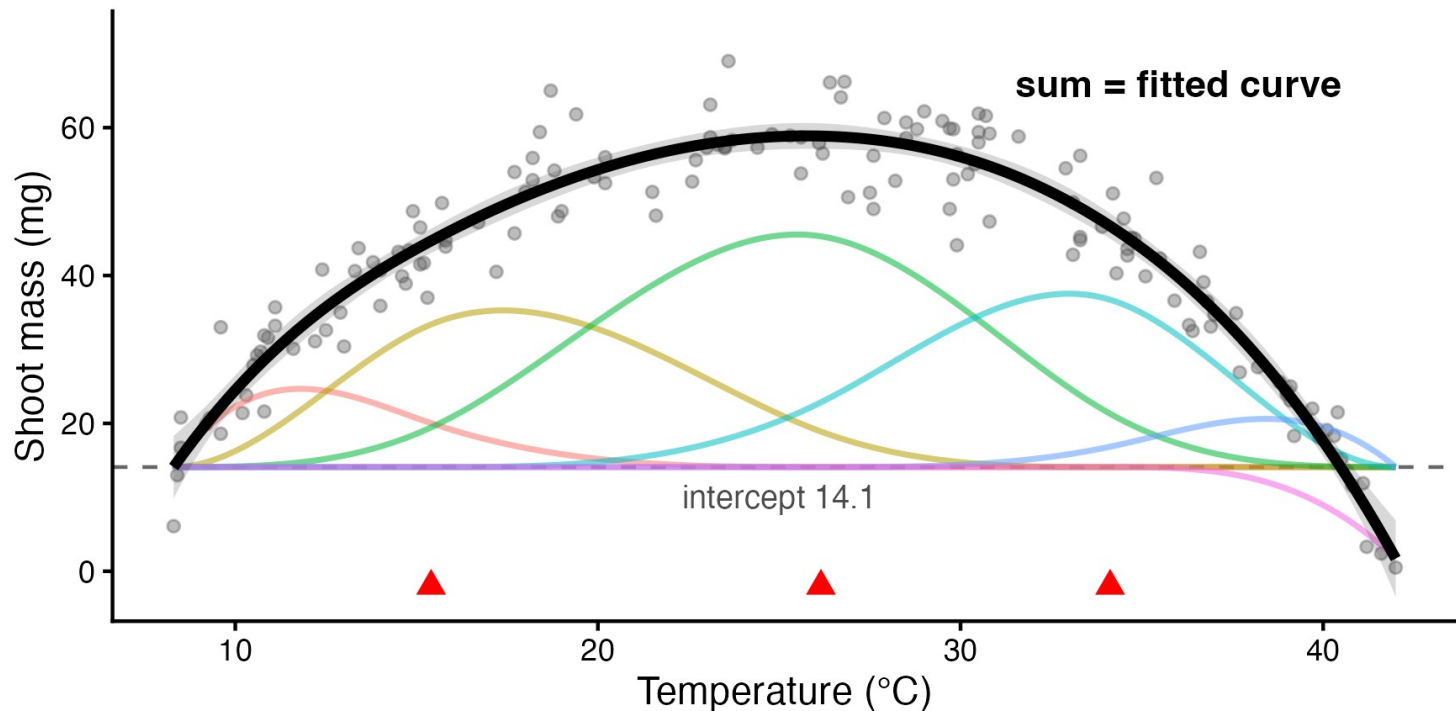
bs(temp, df = 6): six new columns, each a local bump

lm() gives each bump a weight



Column $\times$ coefficient: **tall where seedlings grow fast**

Add them up: that's the curve



New columns, one coefficient each, **summed**

Same machine: lm() with six new columns

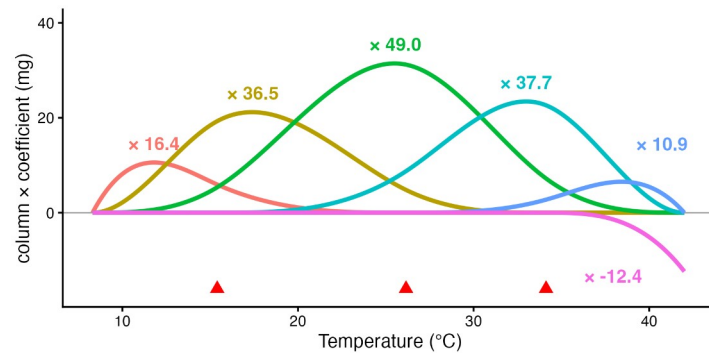
```
> library(splines)
> X <- bs(std$temp, df = 6)
> round(head(X), 2)
```

	1	2	3	4	5	6
[1,]	0.61	0.32	0.03	0.00	0.00	0
[2,]	0.00	0.05	0.56	0.38	0.00	0
[3,]	0.39	0.51	0.10	0.00	0.00	0
[4,]	0.51	0.42	0.06	0.00	0.00	0
[5,]	0.00	0.05	0.54	0.40	0.01	0
[6,]	0.58	0.10	0.00	0.00	0.00	0

← pot 1: bumps 1–3 on

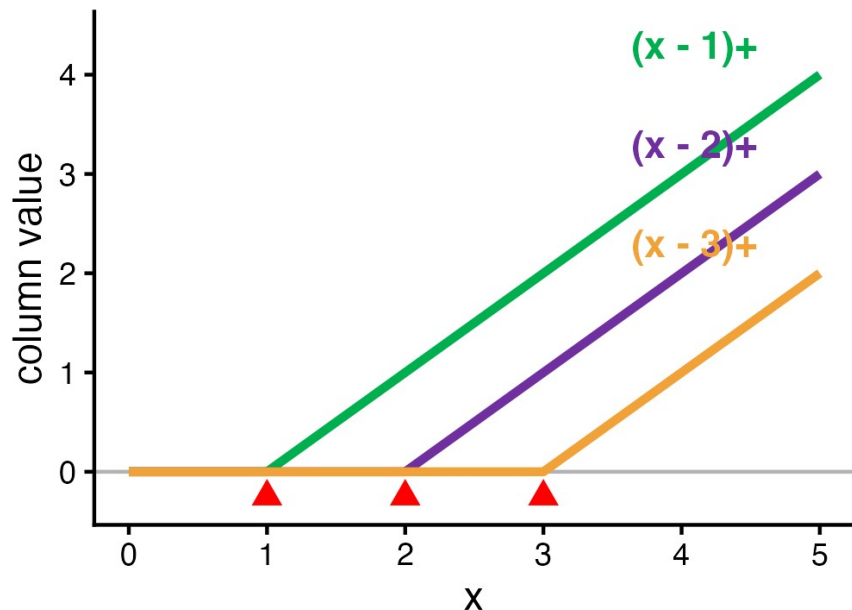
```
> m_bs <- lm(mass_mg ~ bs(temp, df = 6), data = std)
> round(coef(m_bs), 1)
```

	(Intercept)	bs(temp, df = 6)1	bs(temp, df = 6)2	bs(temp, df = 6)3	bs(temp, df = 6)4	bs(temp, df = 6)5	bs(temp, df = 6)6
	14.1	16.4	36.5	49.0	37.7	10.9	-12.4

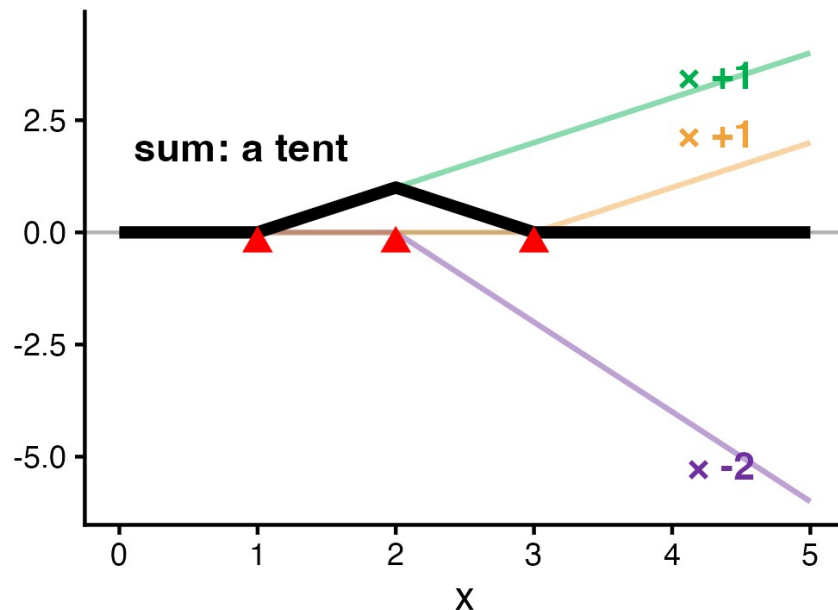


Where do the bumps come from?

Three ramp columns

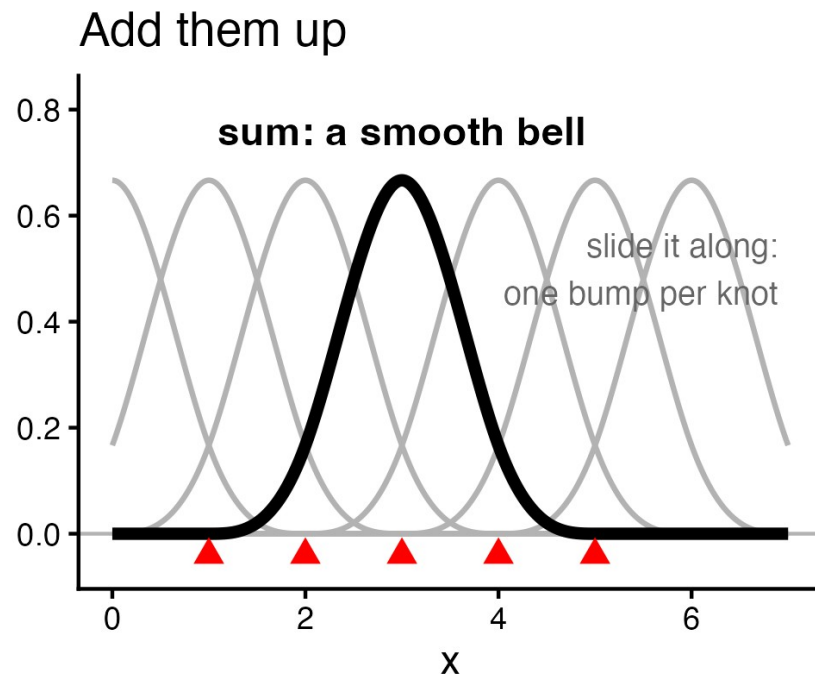
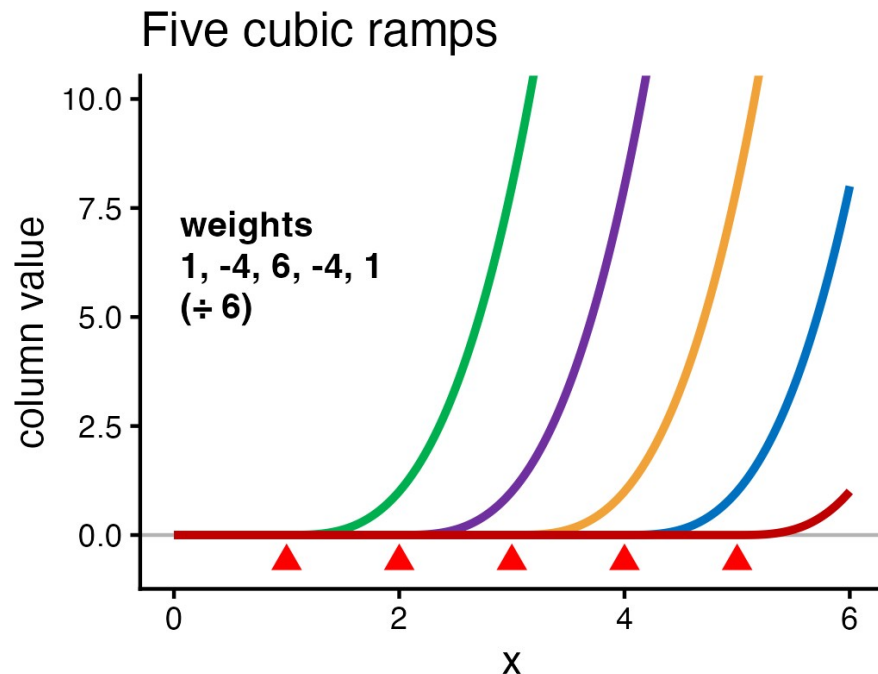


Weight them +1, -2, +1, add up



Ramp columns that switch on at knots. **Weights +1, -2, +1 make a tent.**

The smooth version: cubic ramps make a bell



Five cubic ramps $\times (1, -4, 6, -4, 1) \div 6 =$ **one smooth bump**. That is a B-spline.

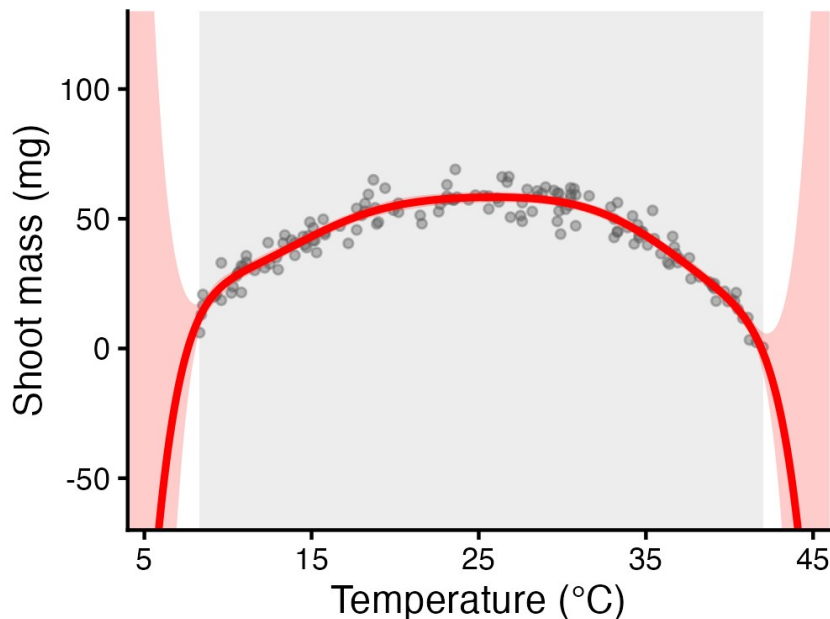
A spline by hand: x , x^2 , x^3 and one ramp per knot

```
> ramp <- function(x, knot) pmax(x - knot, 0)^3
> m_hand <- lm(mass_mg ~ temp + I(temp^2) + I(temp^3) +
+             ramp(temp, 18.8) + ramp(temp, 30.5),
+             data = std)
> m_bs <- lm(mass_mg ~ bs(temp, knots = c(18.8, 30.5)),
+           data = std)
> all.equal(fitted(m_hand), fitted(m_bs))
[1] TRUE ← identical fitted values
> summary(m_hand)$r.squared
[1] 0.9157797
> summary(m_bs)$r.squared
[1] 0.9157797 ← same R-squared
```

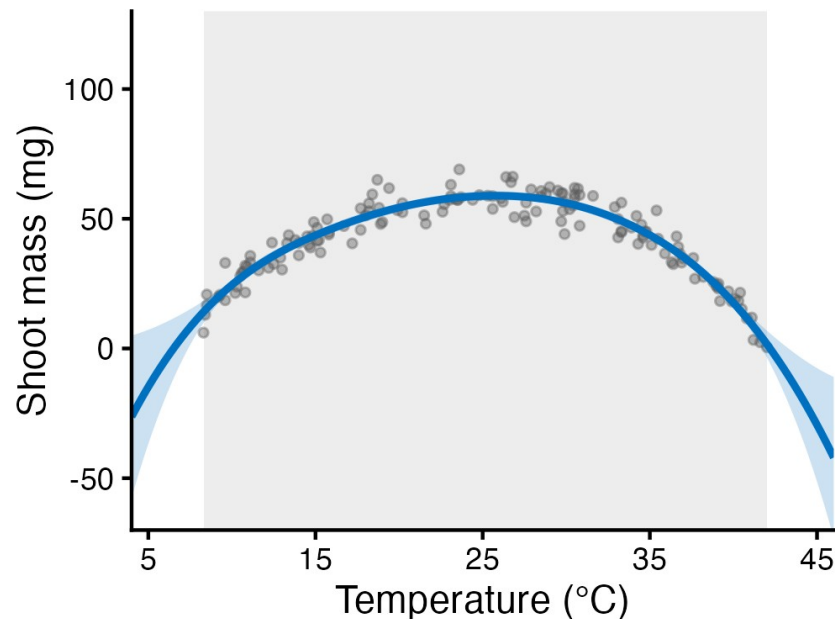
Ramps or bumps: **the same fitted curve**; `bs()` just uses tidier columns

Local: the edges stay calm

Polynomial, degree 10



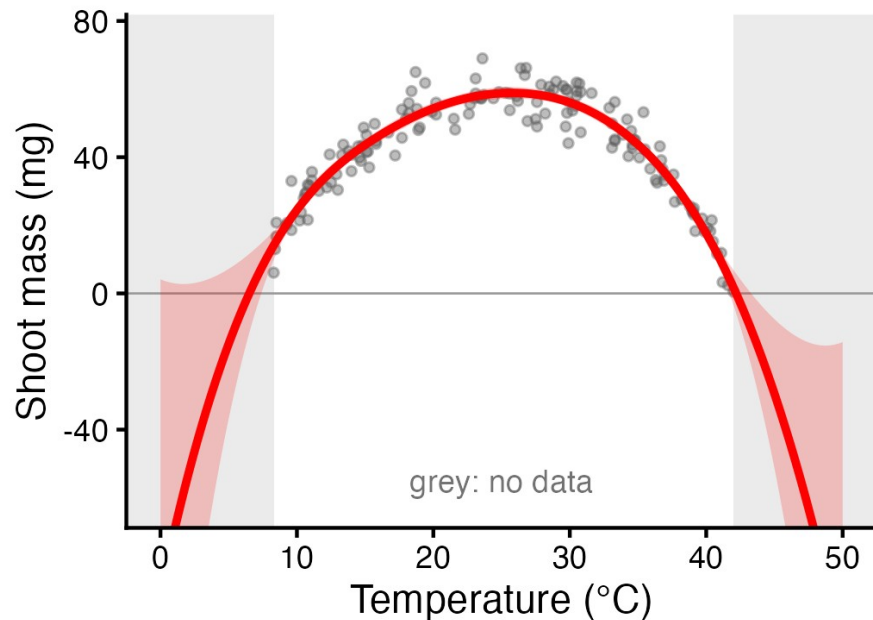
Spline, bs(df = 6)



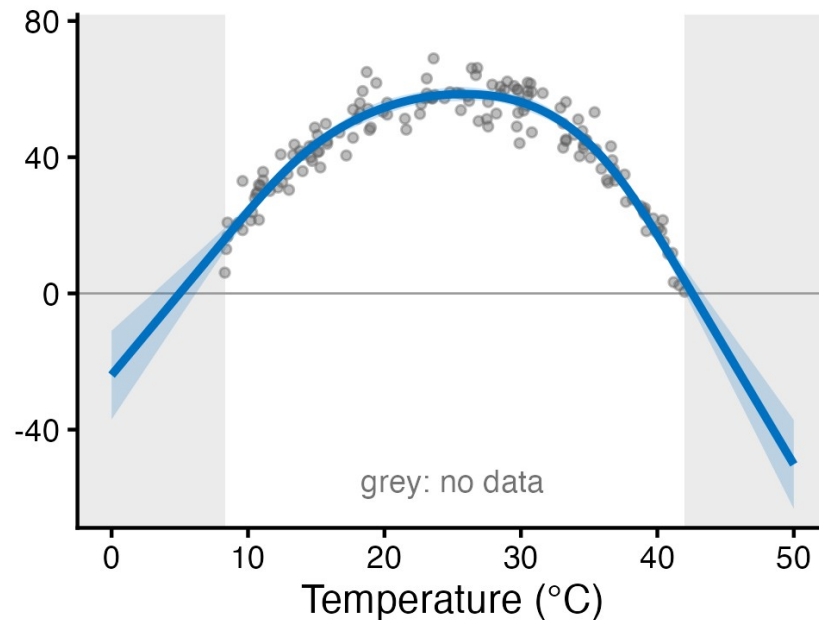
Polynomial flaps; the **spline** stays calm

Natural splines: straight past the data

bs(): bends past the data



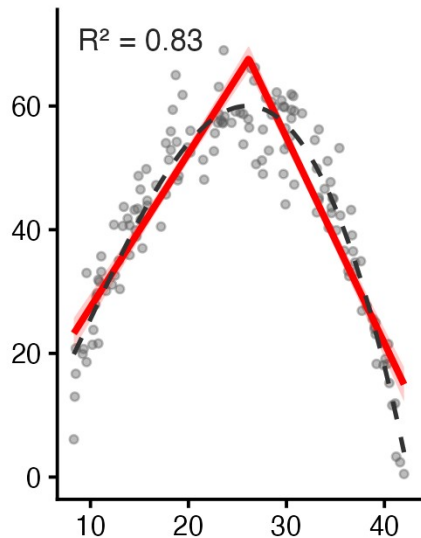
ns(): straight past the data



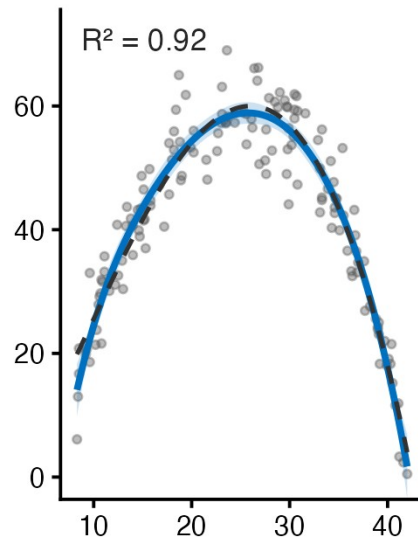
ns(): same fit inside, straight lines outside. **Drops into lm, glm, lmer.**

How many knots?

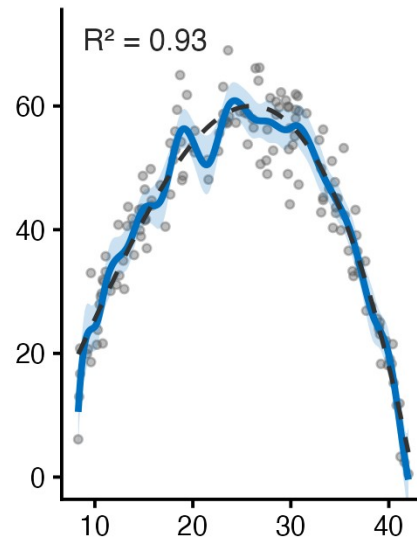
2 straight pieces: stiff



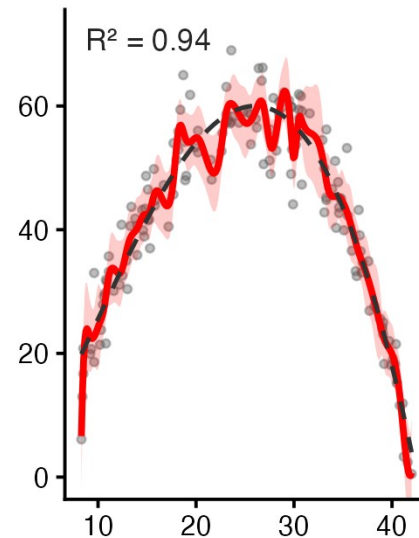
6 columns



20 columns



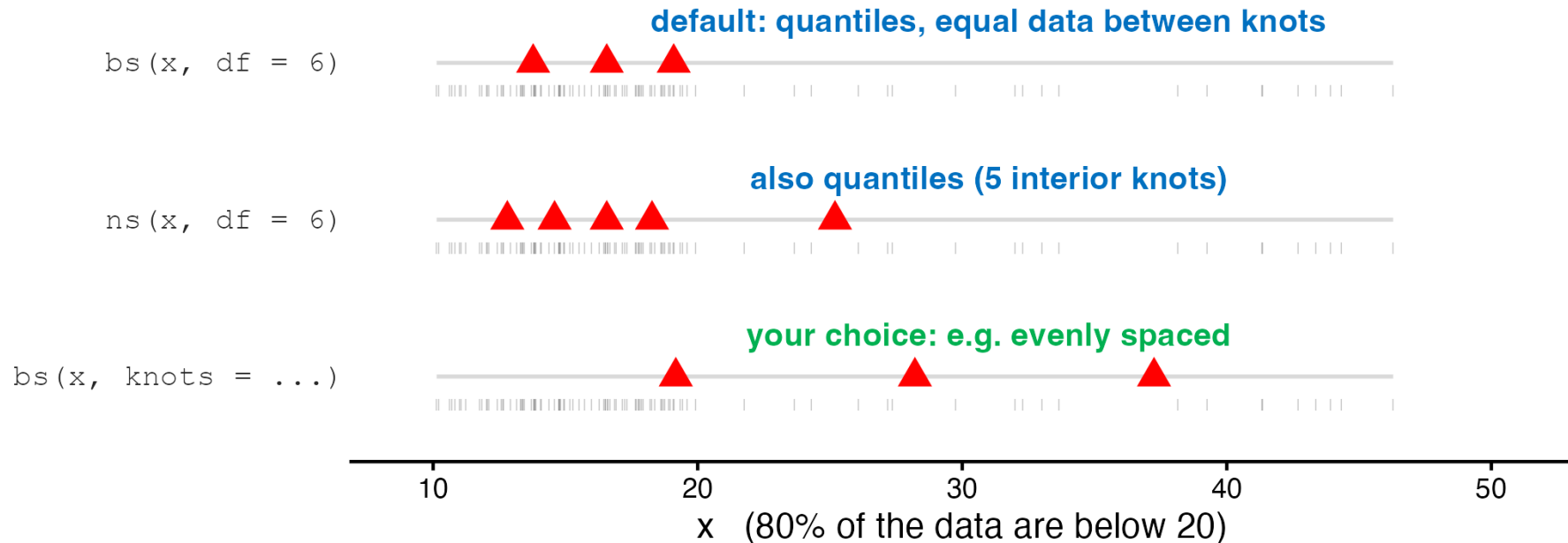
40 columns: wiggly



Temperature (°C) (y: shoot mass, mg; dashed = the true curve)

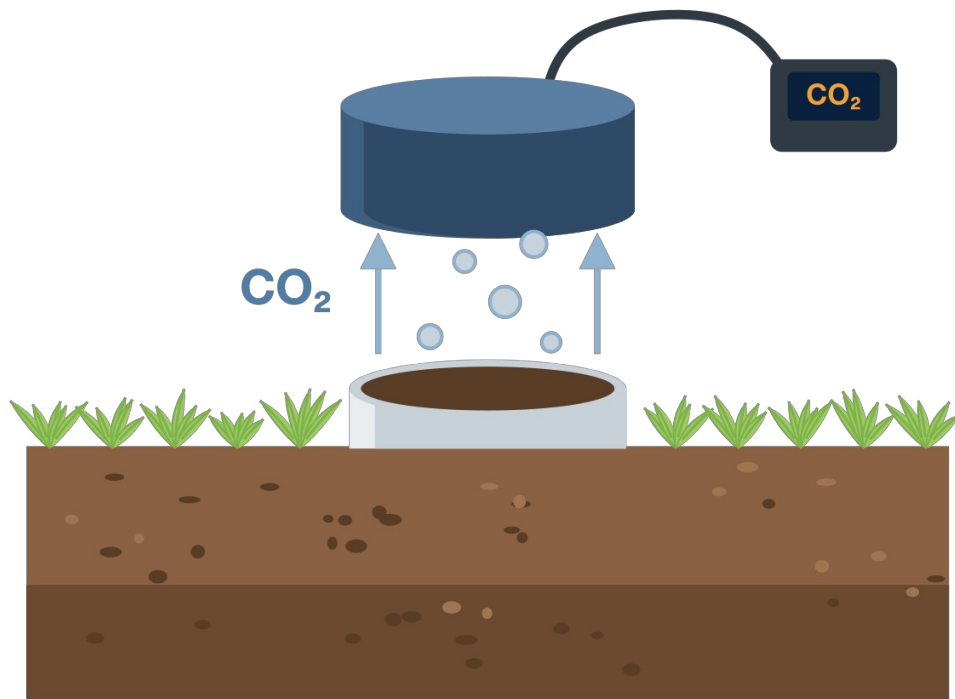
Too few: stiff. Too many: wiggly.

Where do the knots go?



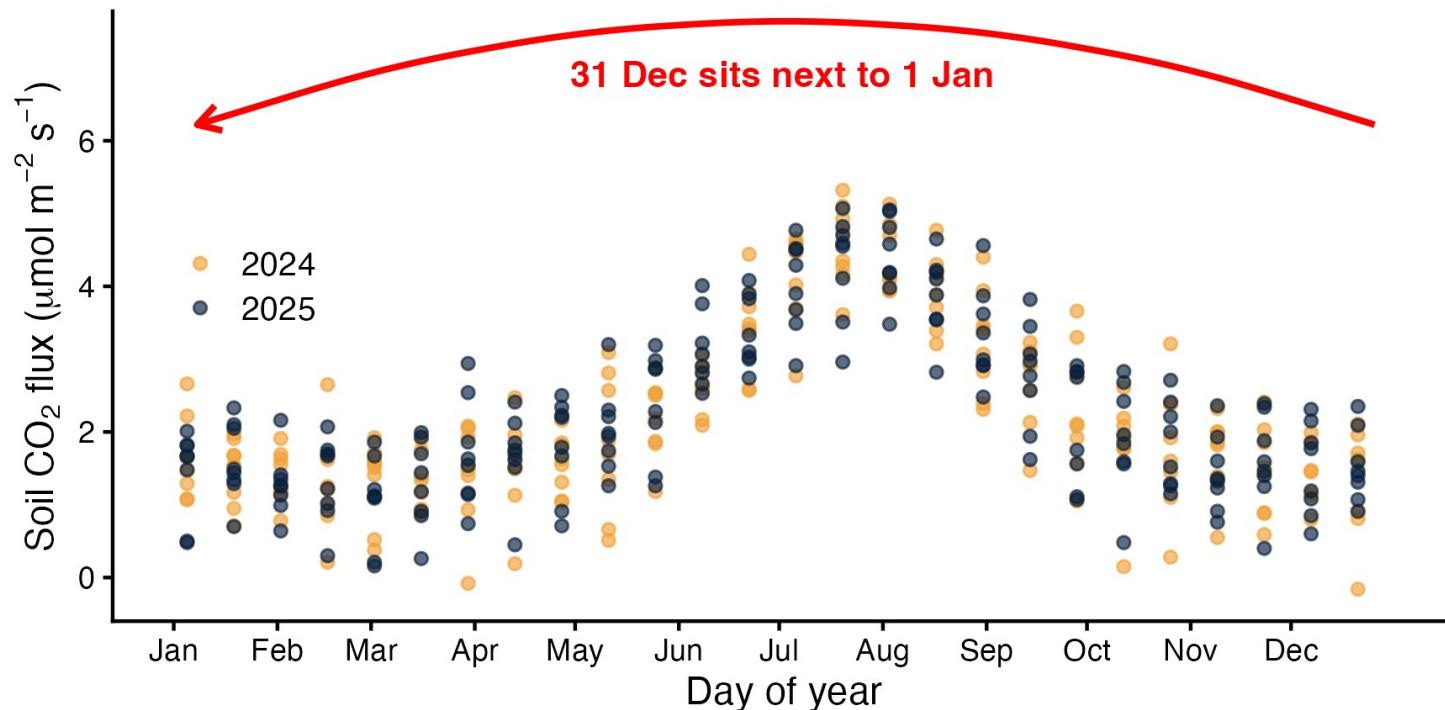
Default: quantiles, equal data between knots. Or place them yourself with **knots =**

A second experiment: a cycle



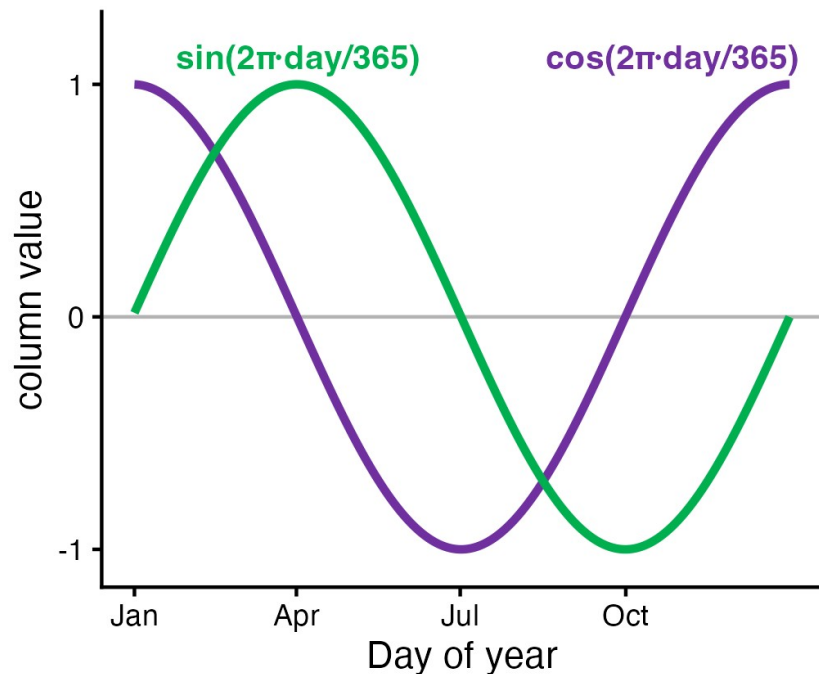
8 soil collars, every two weeks, two years. (Hypothetical, simulated data.)

Soil respiration through the year



31 December sits next to **1 January**

Cycles: sin and cos columns



$$\text{flux} = \beta_0 + \beta_1 \sin\left(\frac{2\pi \text{ day}}{365}\right) + \beta_2 \cos\left(\frac{2\pi \text{ day}}{365}\right)$$

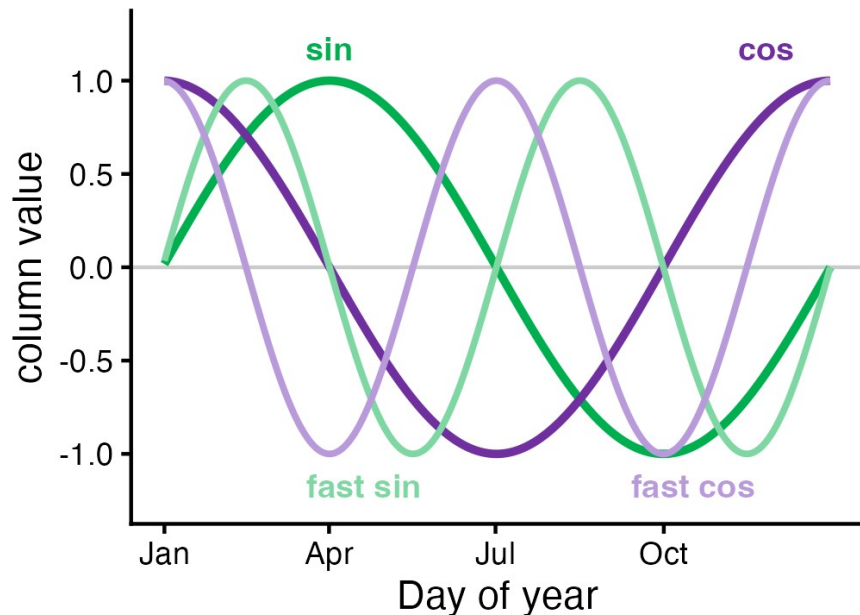
day: day of the year (1 to 365)

$2\pi \text{ day}/365$: how far round the year, as an angle

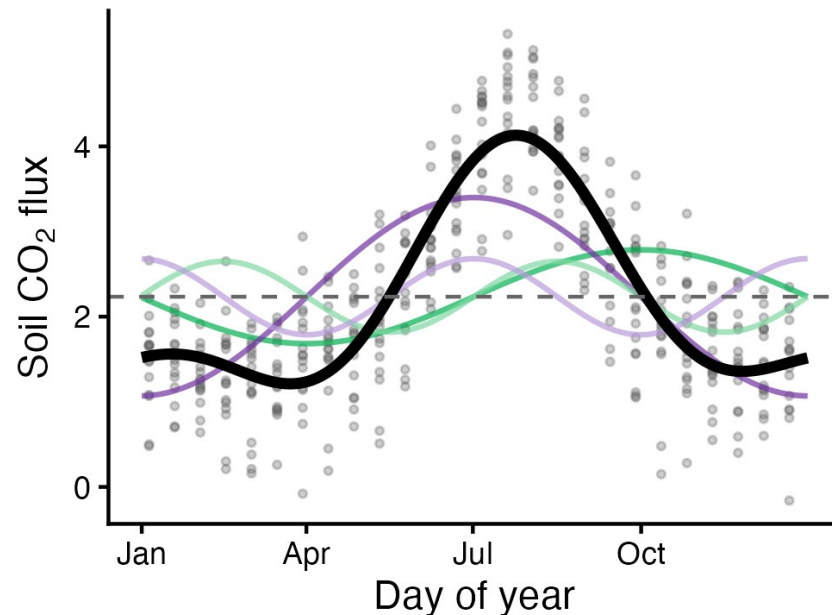
β_1, β_2 : weights that $\text{lm}()$ estimates

A second pair, twice as fast

A second pair, twice as fast



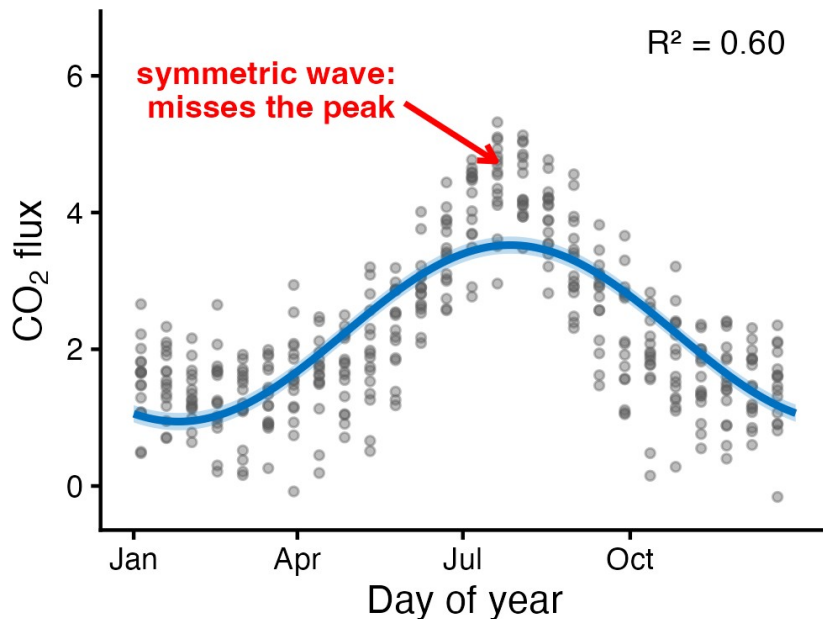
× weights, add up: the leaning wave



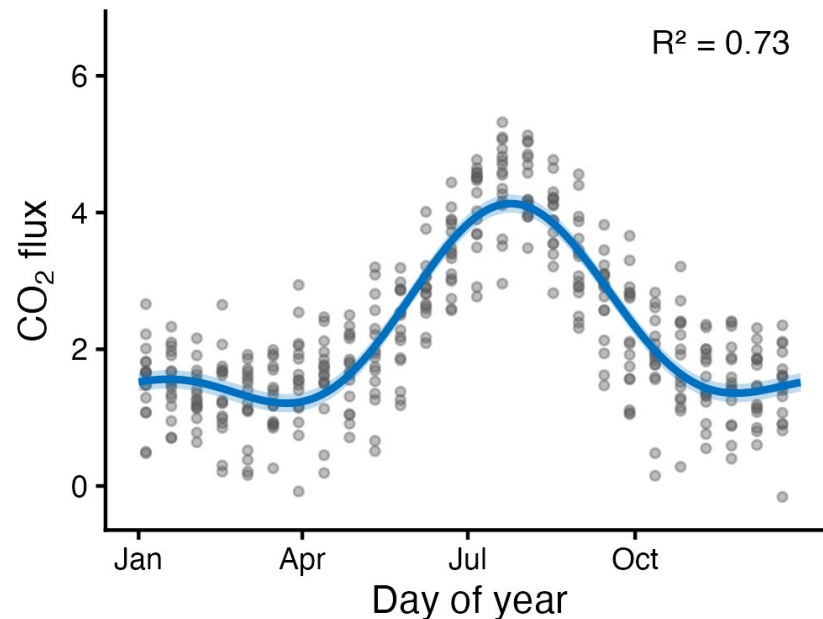
Four columns, four weights, **summed**: the same trick as the spline bumps

More harmonics, more shape

1 sin/cos pair

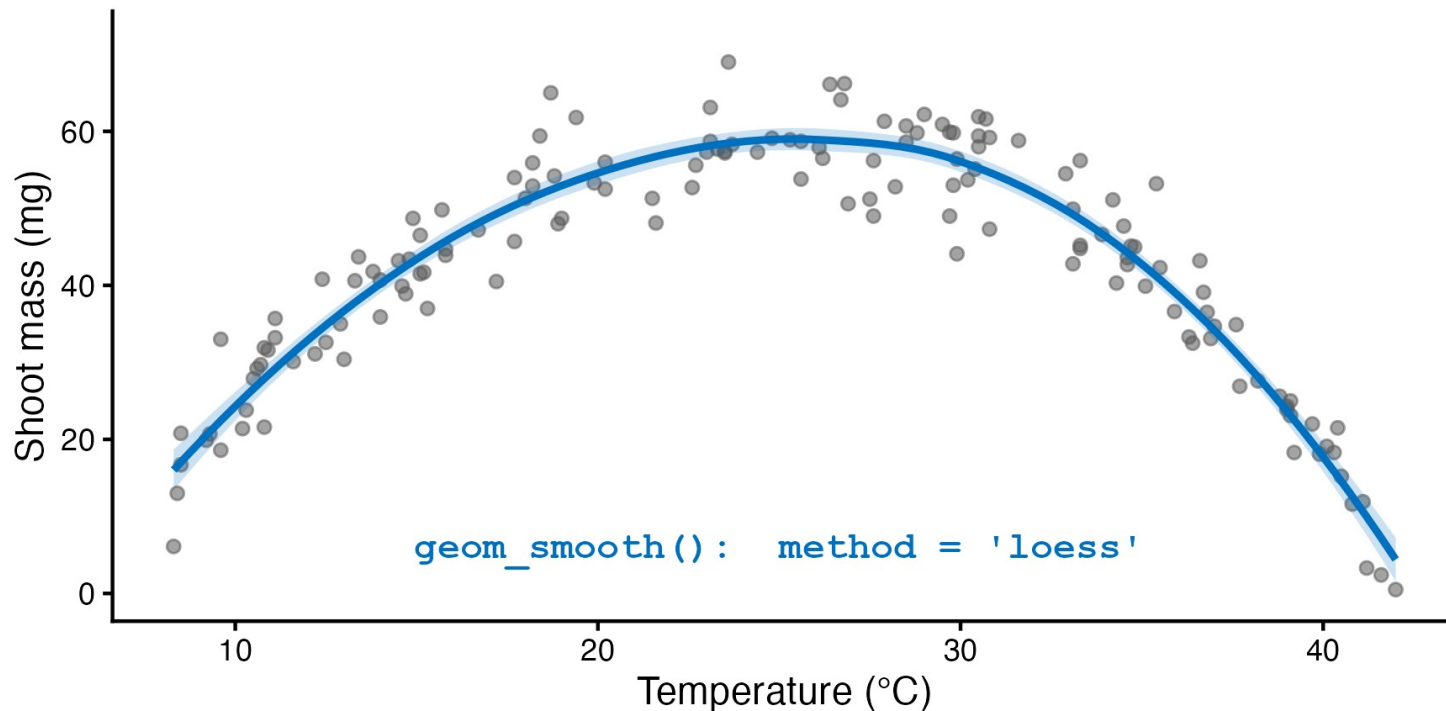


2 pairs: the wave can lean



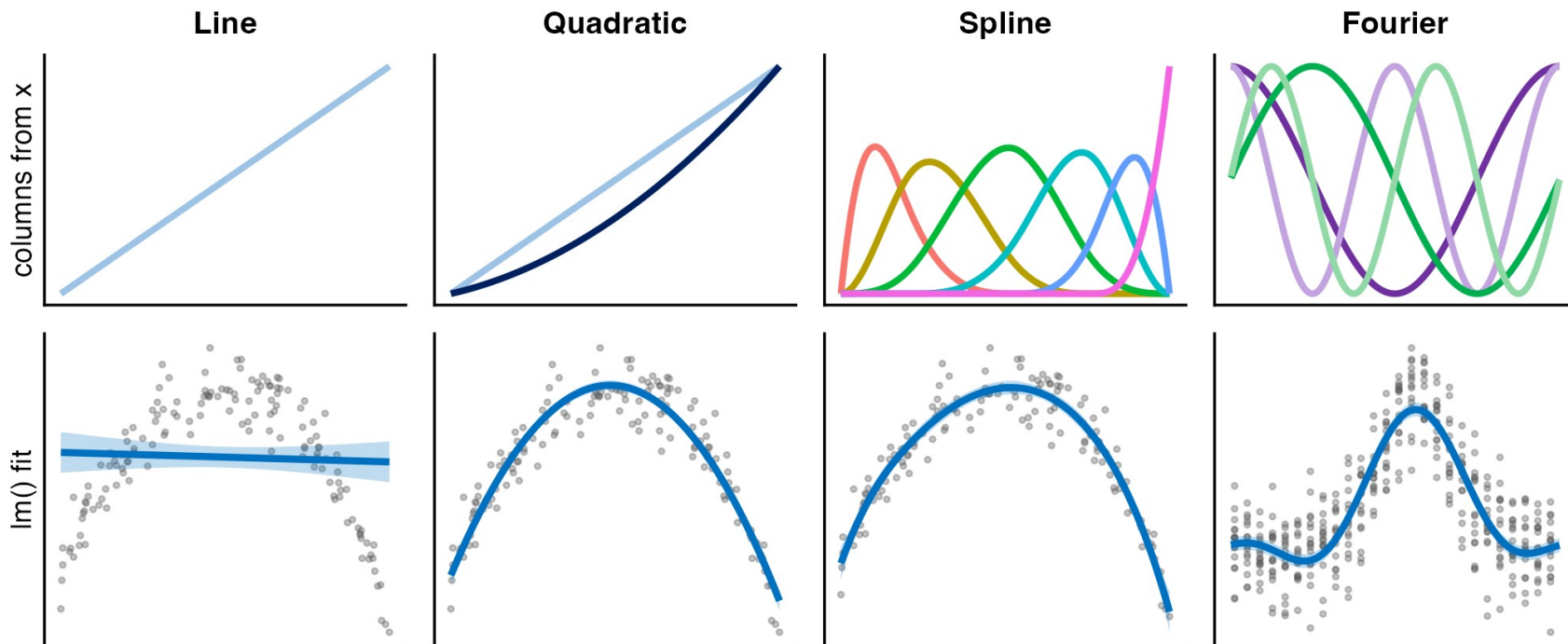
A second sin/cos pair lets the wave lean: R^2 0.60 → 0.73

What `geom_smooth()` draws: loess



loess is **for looking**, not for inference

Every curve so far



New columns + `lm()`. The open question: how flexible?

1 Bend the line

2 Splines

▶ **3 GAMs**

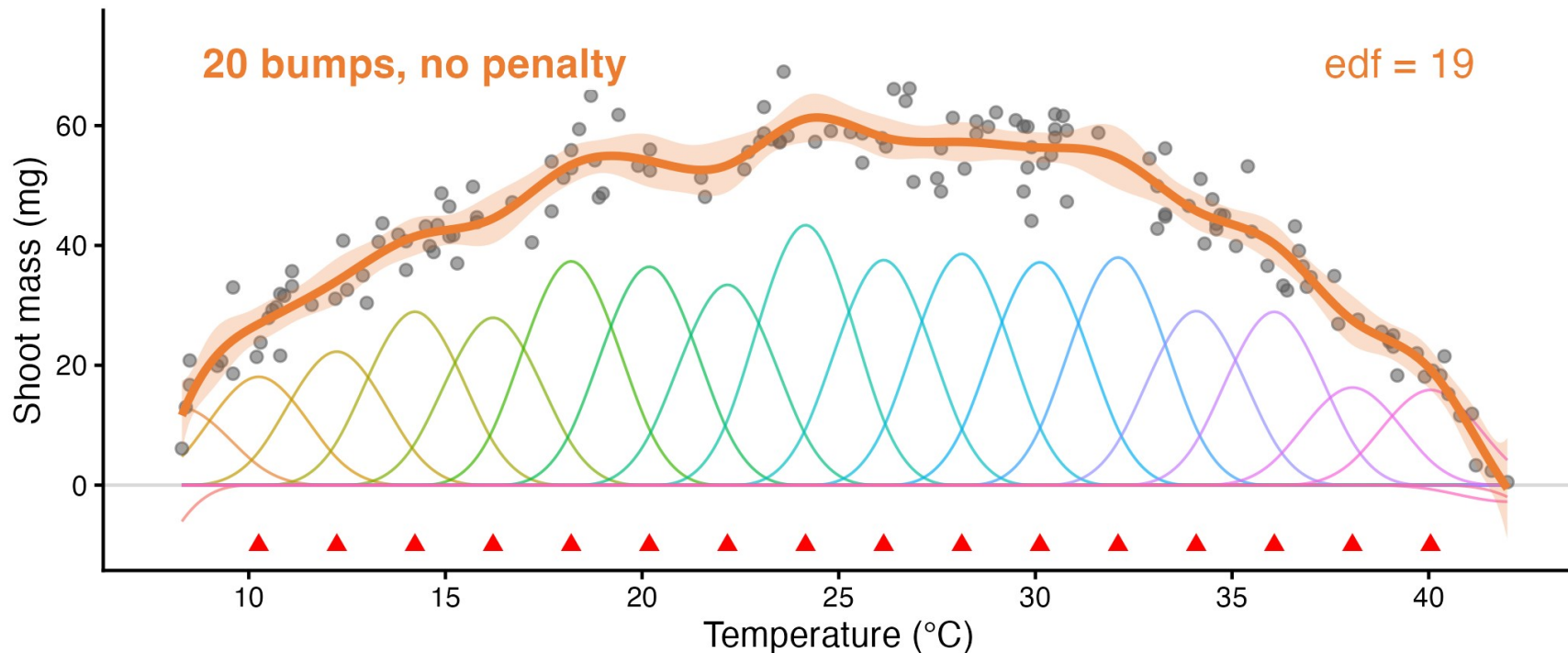
4 Reading a GAM

5 GAM toolkit

6 Mechanistic models



Lots of bumps, no brakes



Twenty bumps, no penalty: it **fits the noise**

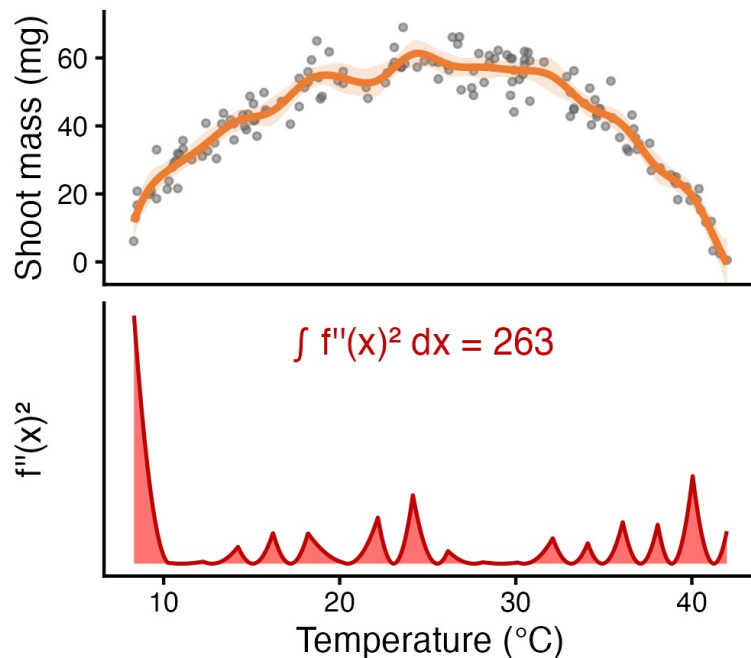
Fit the data, but pay for every wiggle

$$\text{minimise} \quad \underbrace{\sum_i (y_i - f(x_i))^2}_{\text{fit the data}} + \lambda \underbrace{\int f''(x)^2 dx}_{\text{wiggleness}}$$

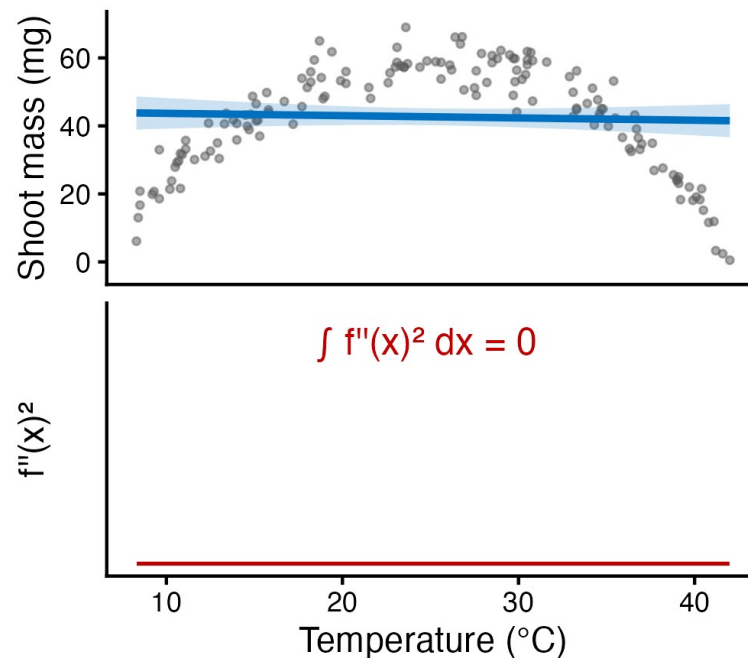
f : the fitted curve f'' : its curvature (second derivative) λ : the price of a wiggle

Wiggleness = how much the curve bends

Wiggly curve

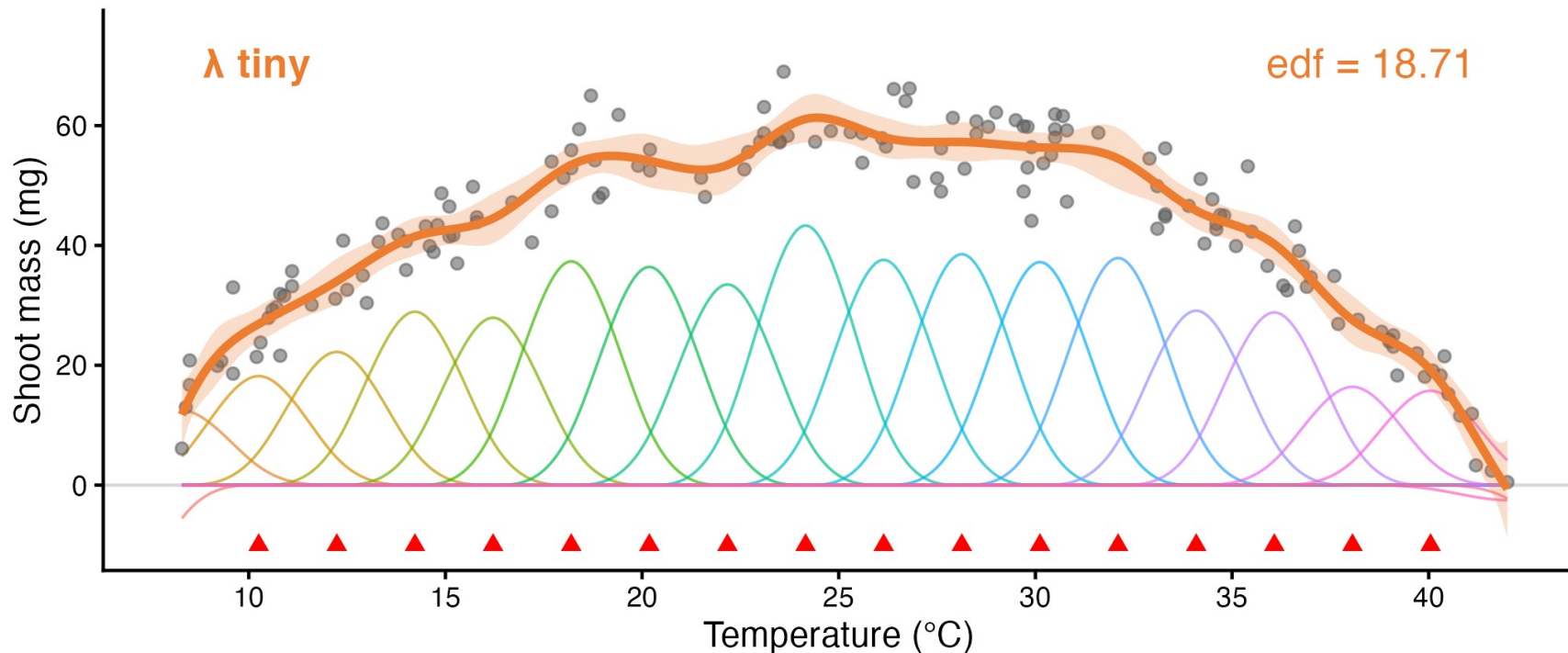


Straight line



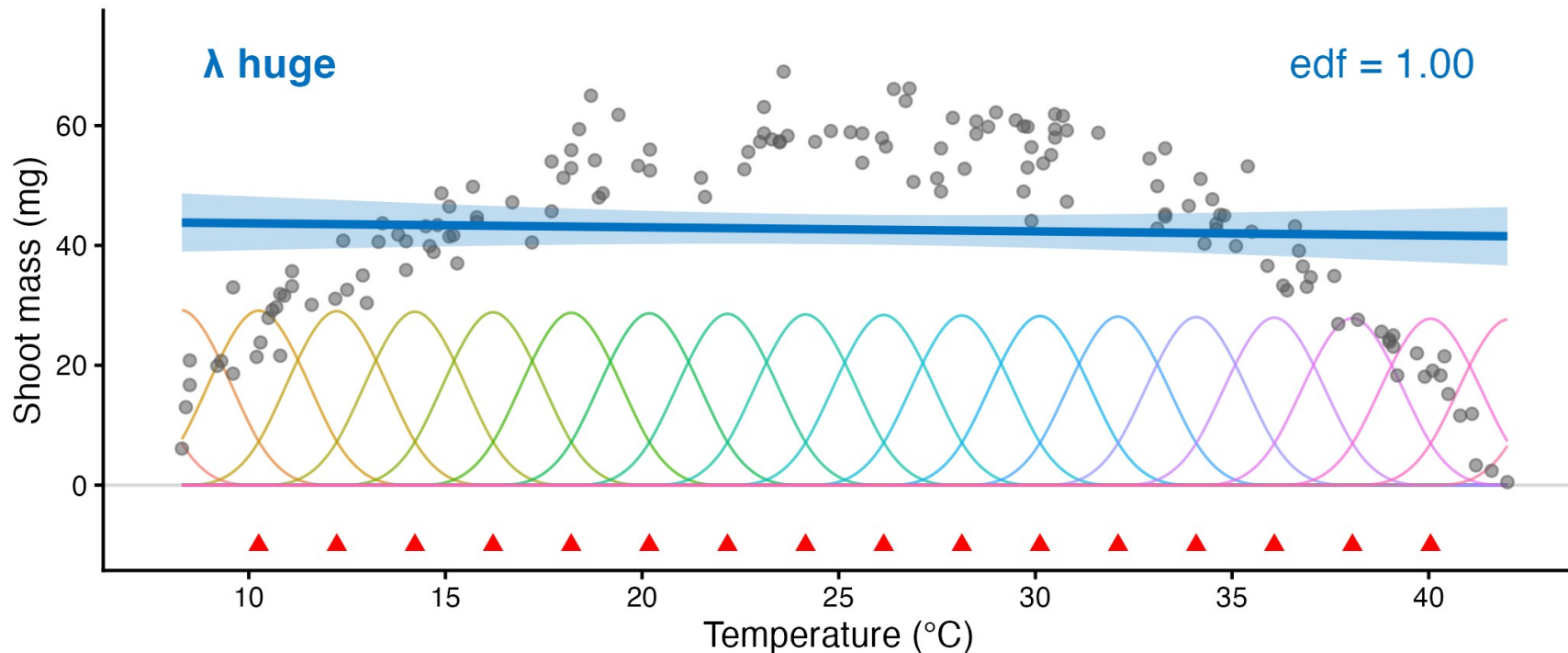
A straight line has **zero** wiggleness

λ tiny: wiggly



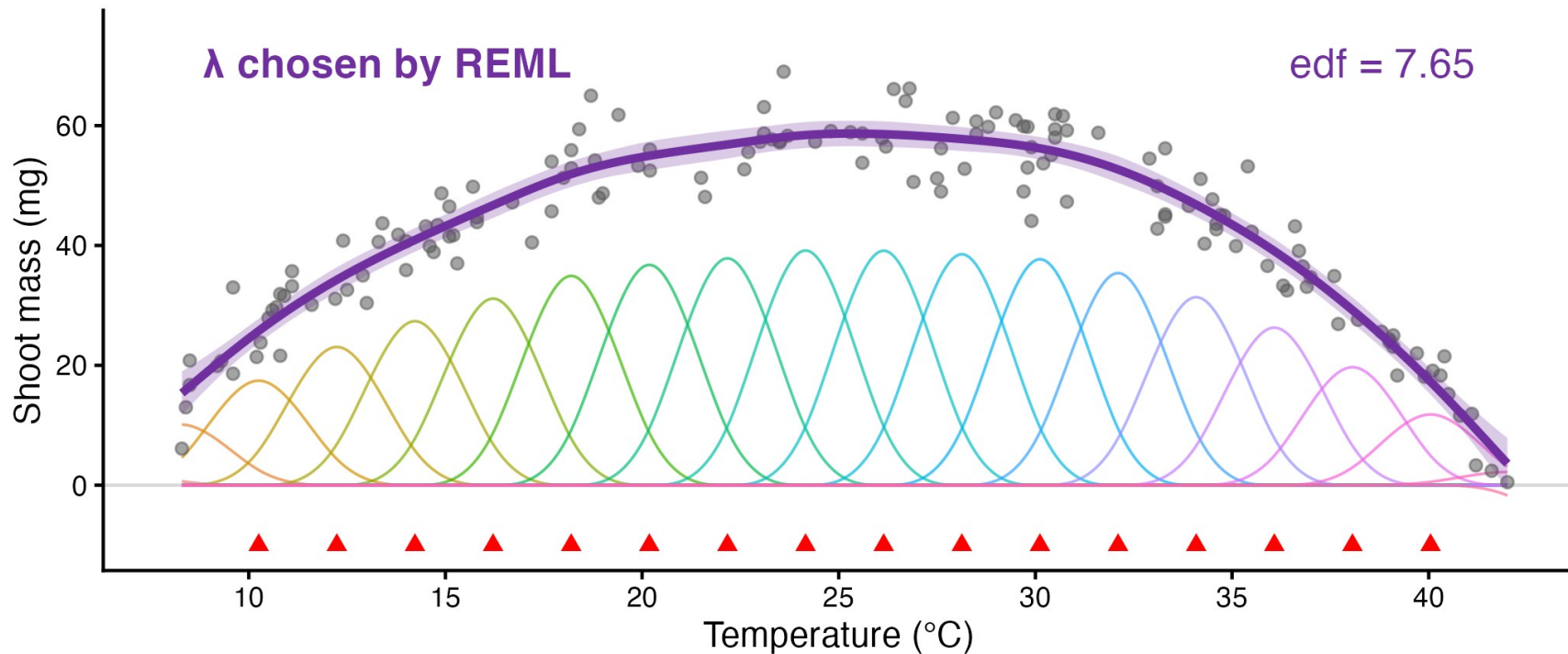
λ tiny: wiggles are free (edf = 18.7)

λ huge: a straight line



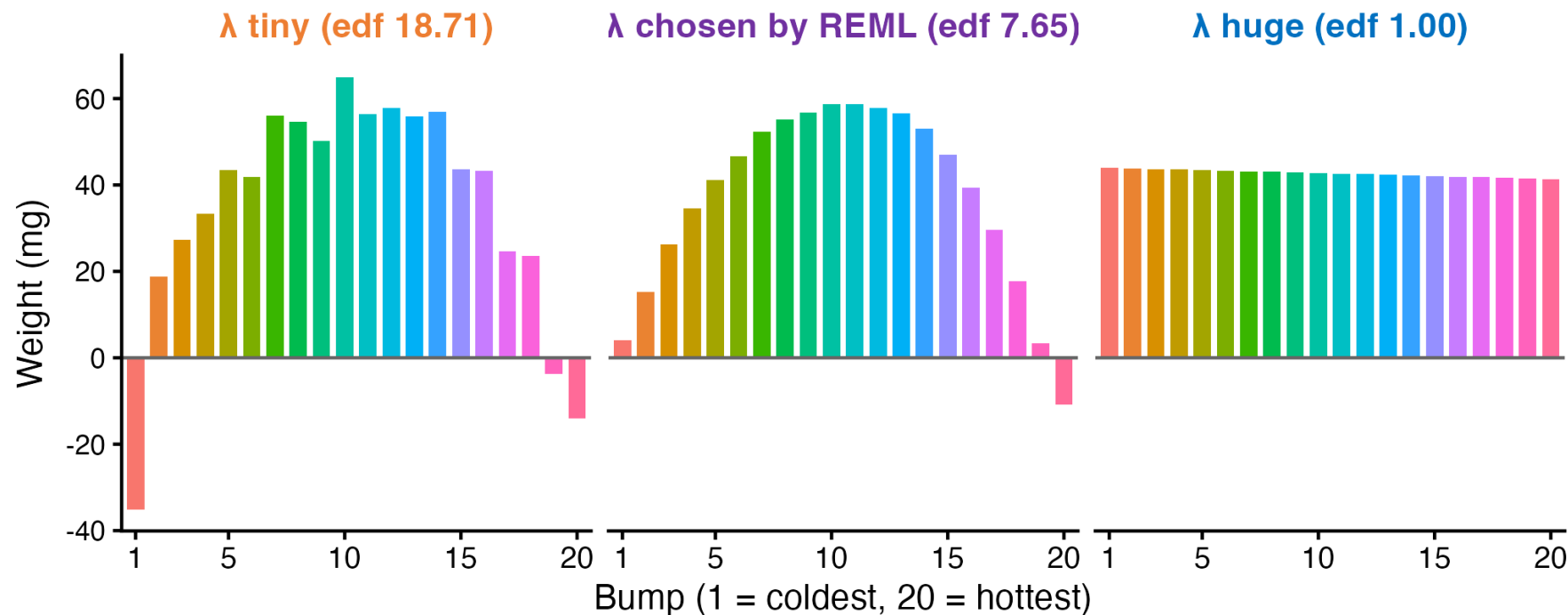
λ huge: any bend is too expensive (edf = 1)

λ chosen from the data



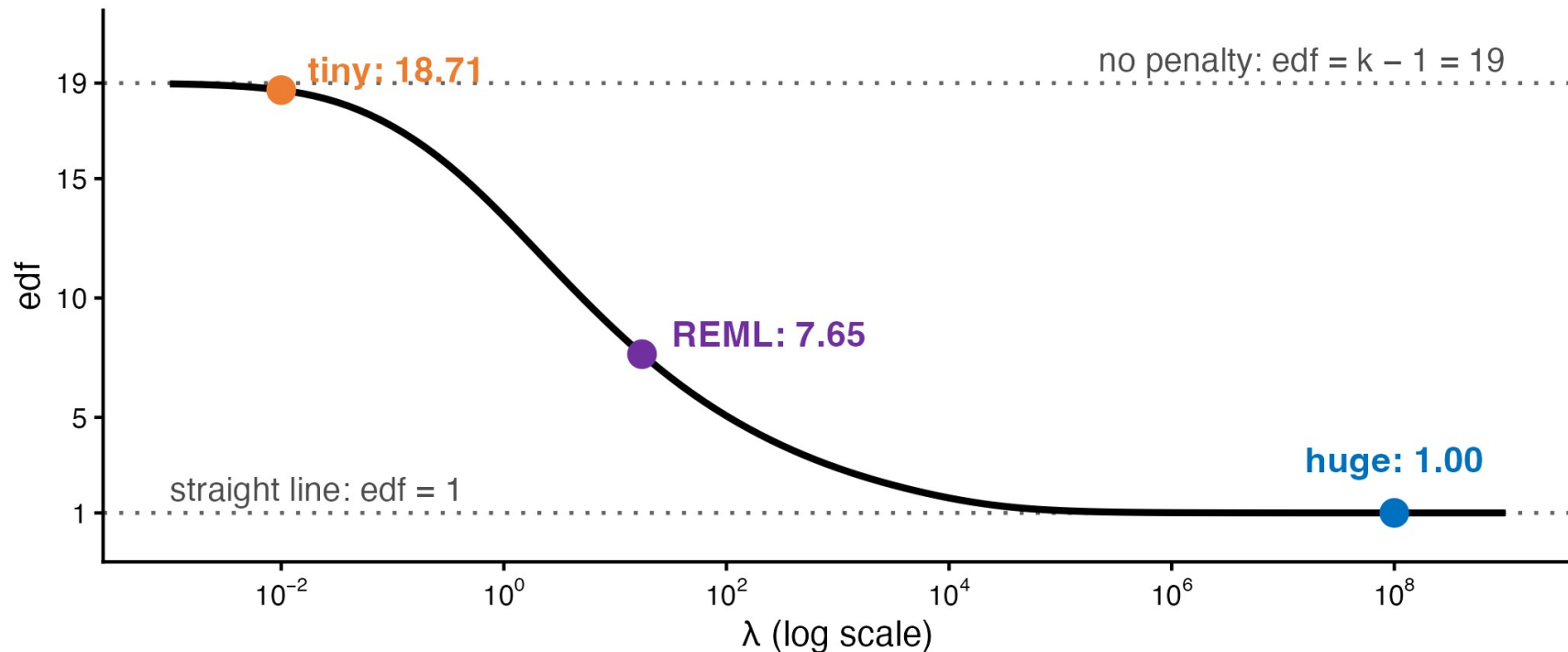
λ chosen by REML: edf = 7.63

The penalty changes the weights



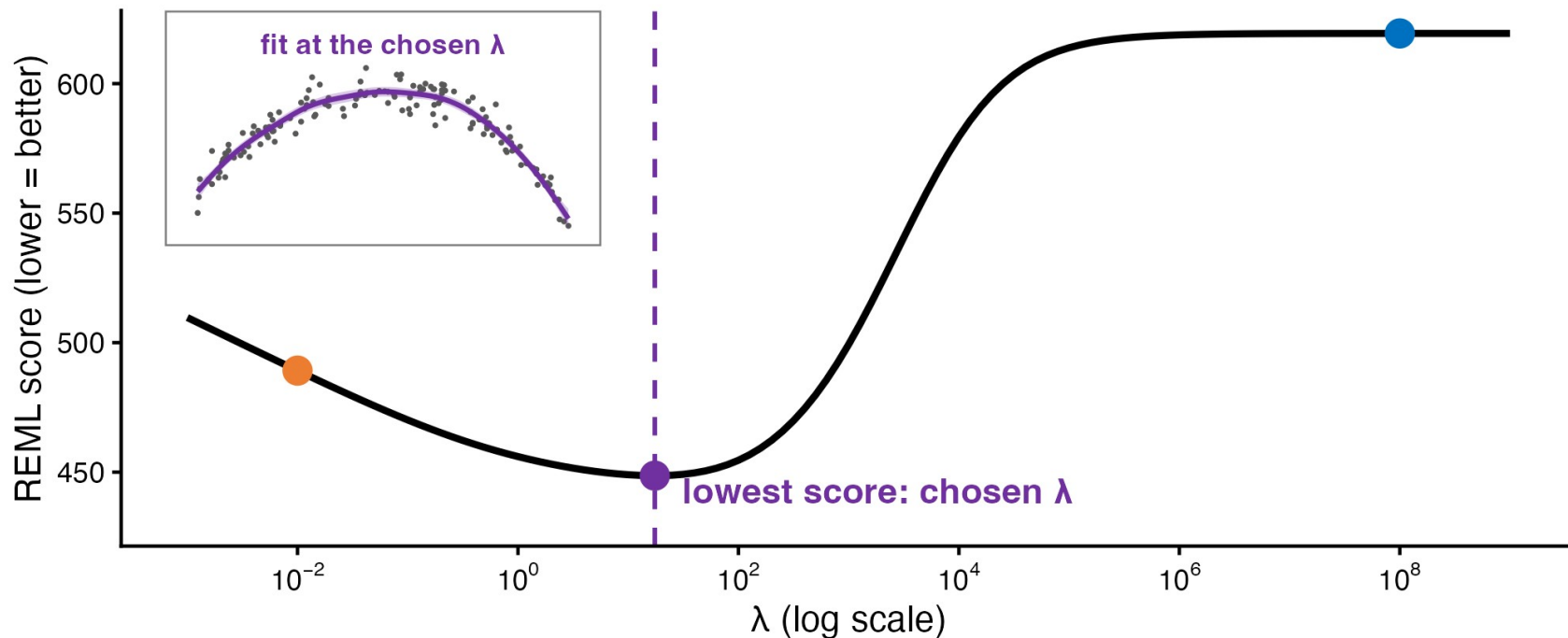
Same bumps, **different weights**: λ shrinks them toward a line

edf: how many parameters the curve is worth



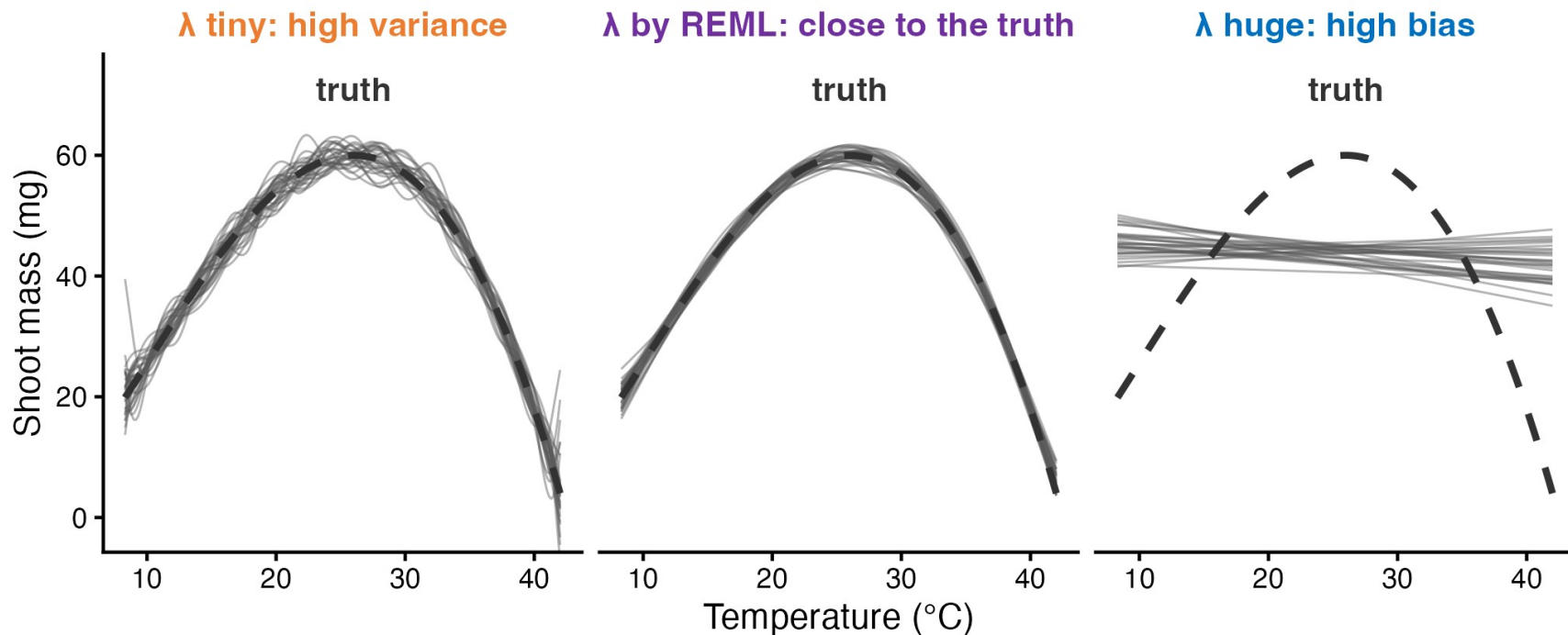
edf = 1: a straight line. **edf = 19**: no penalty

REML picks λ



Always write **method = "REML"**

Too wiggly: variance. Too stiff: bias.



λ tiny: high variance. **λ huge:** high bias. **REML:** close to the truth

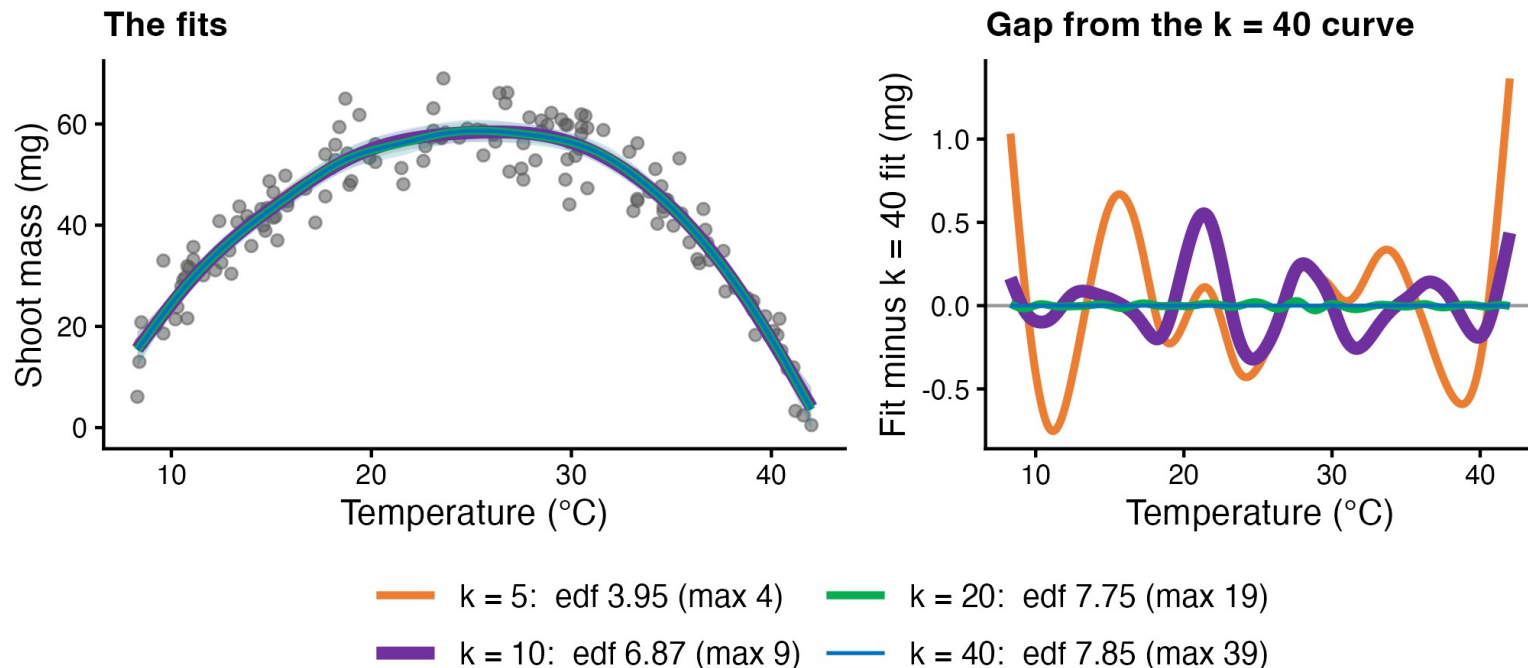
Same idea in Week 6: ridge regression

$$\text{GAM: } \sum_i (y_i - f(x_i))^2 + \lambda \int f''(x)^2 dx$$

$$\text{ridge (Week 6): } \sum_i (y_i - \hat{y}_i)^2 + \lambda \sum_j \beta_j^2$$

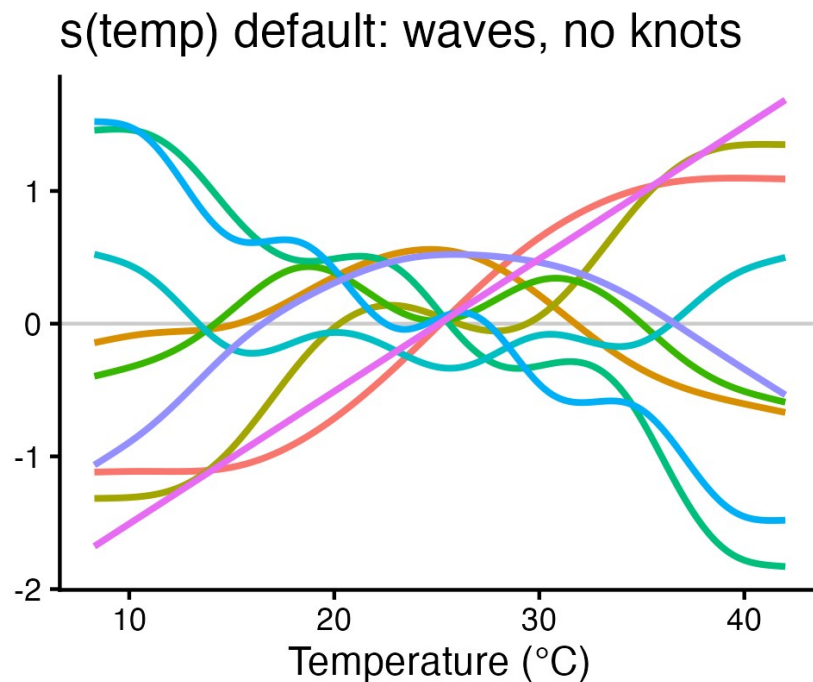
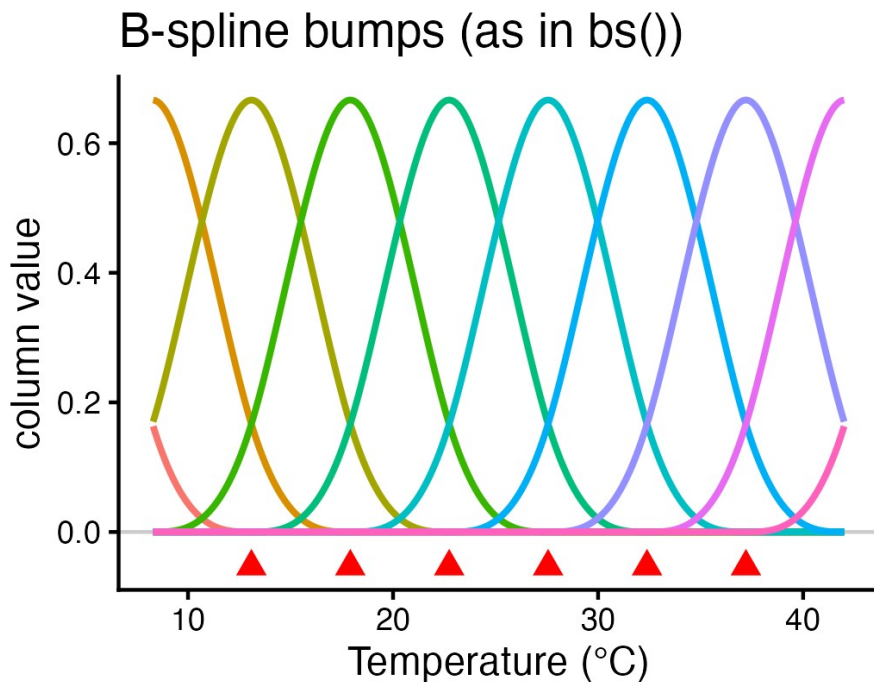
$\hat{y}_i$: fitted value β_j : the slopes (not the intercept) λ : how hard to shrink, chosen from the data

k is a ceiling, not the wiggleness



k = maximum wiggleness. **The penalty** decides the actual wiggleness.

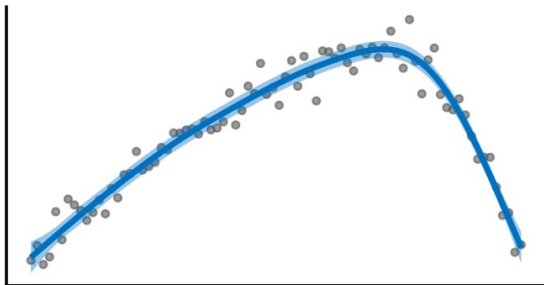
The default `s()`: waves, not bumps



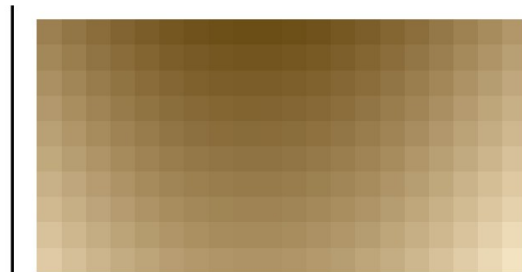
Bumps or waves: **same trick, nearly the same curve** (edf 6.42 vs 6.87)

When would you reach for a GAM?

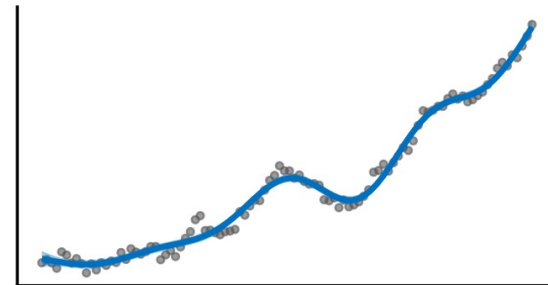
Shape: optimum, cliff



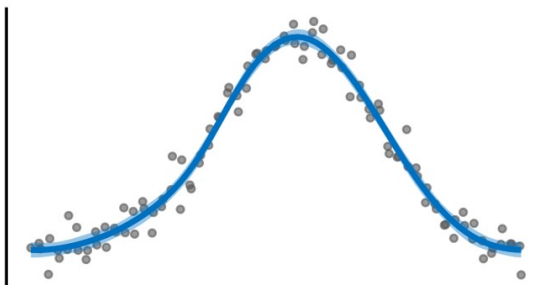
Curved nuisance: a field



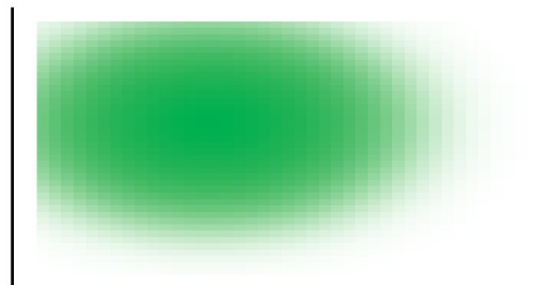
Trend over time



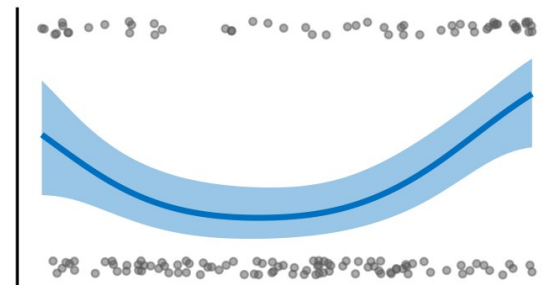
Cycle: Dec meets Jan



Space: maps



Yes/no or counts



Whenever the curve matters and **you have no equation for it**

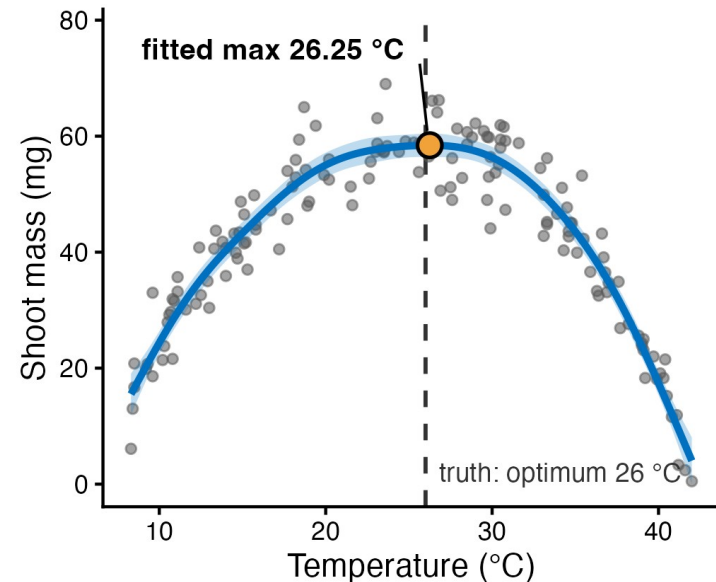
Your first GAM

```
> library(mgcv)
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML")
> summary(m_gam)
```

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value
s(temp)	6.869	7.98	192.4	<2e-16 ***

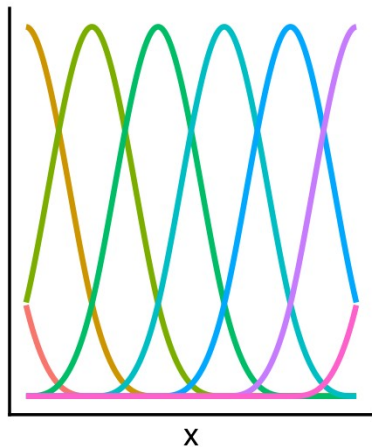
← edf



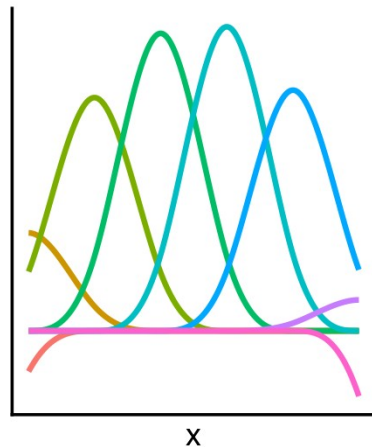
Fitted optimum **26.25 °C**; the truth is 26 °C

One idea, four tools

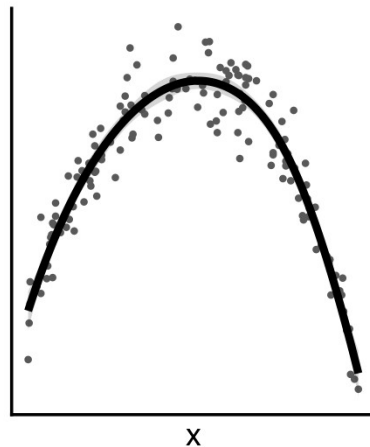
1. Columns from x



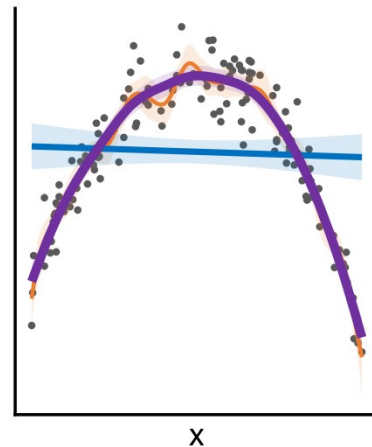
2. A weight each



3. Add them up



4. λ tames wiggles



Polynomials, splines, Fourier terms and GAMs: **columns $\times$ weights, summed**

Live coding: splines and a first GAM



`Week4_live_1_curves.R`

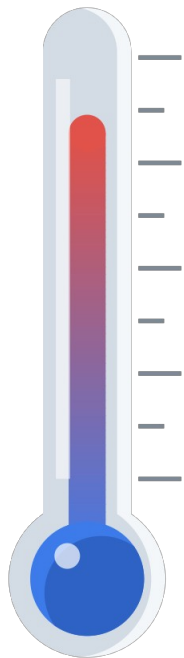
Part 2: sections 5 to 7

5 A spline: bumps as columns

6 Cycles: sin and cos columns

7 A first GAM

Next class



Reading a GAM summary, line by line

Choosing k

The GAM toolkit

Models whose parameters mean something

Problem Set 4 due Monday 26 October, 1:10 PM

Non-linear regression, part 2

1 Bend the line

2 Splines

3 GAMs

▶ **4 Reading a GAM**

5 GAM toolkit

6 Mechanistic models



Recap

Kahoot-style warm-up

10 quick questions on curves, splines and GAMs.

Not graded.

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-w4-monday-recap](https://greymonroe.github.io/live-poll-test/quiz-play.html?quiz=pls206-w4-monday-recap)

[host screen](#)

The summary, line by line

```
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML")  
> summary(m_gam)
```

```
Family: gaussian  
Link function: identity
```

```
Formula:  
mass_mg ~ s(temp)
```

```
Parametric coefficients:  
              Estimate Std. Error t value Pr(>|t|)  
(Intercept)  42.6800    0.3765   113.4   <2e-16 ***  
---
```

```
Approximate significance of smooth terms:  
              edf Ref.df      F p-value  
s(temp) 6.869    7.98 192.4   <2e-16 ***  
---
```

```
R-sq.(adj) = 0.912  Deviance explained = 91.6%  
-REML = 450.4  Scale est. = 21.261    n = 150
```

A GAM is a GLM with curves

```
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML") ← a smooth of temp  
> summary(m_gam)
```

```
Family: gaussian ← normal errors  
Link function: identity ← no transformation
```

```
Formula:  
mass_mg ~ s(temp) ← a smooth of temp
```

```
Parametric coefficients:  
              Estimate Std. Error t value Pr(>|t|)  
(Intercept)  42.6800    0.3765   113.4   <2e-16 ***  
---
```

```
Approximate significance of smooth terms:  
              edf Ref.df      F p-value  
s(temp) 6.869    7.98 192.4   <2e-16 ***  
---
```

```
R-sq.(adj) = 0.912  Deviance explained = 91.6%  
-REML = 450.4  Scale est. = 21.261    n = 150
```

The intercept is the mean of y

```
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML")
> summary(m_gam)
```

```
Family: gaussian
Link function: identity
```

```
Formula:
mass_mg ~ s(temp)
```

```
Parametric coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	42.6800	0.3765	113.4	<2e-16 ***

← = mean of y

```
Approximate significance of smooth terms:
```

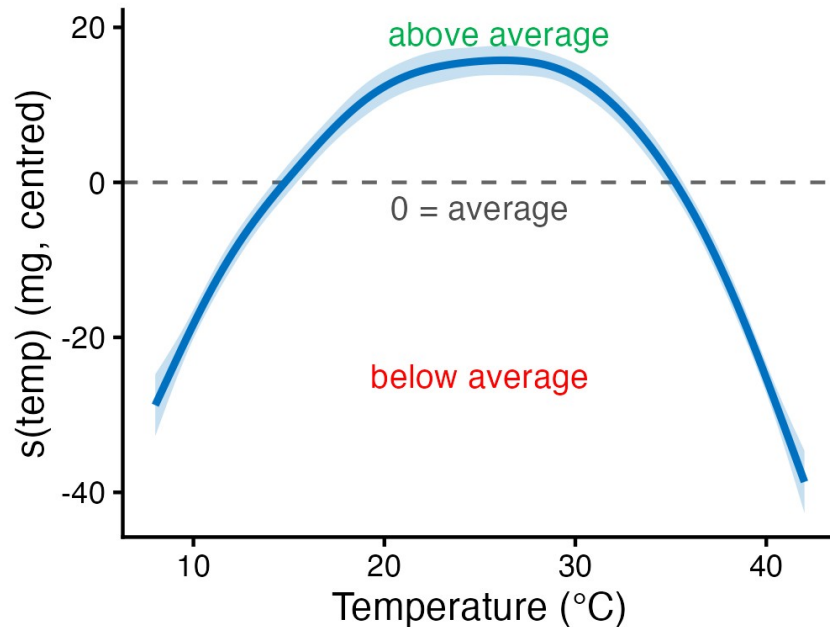
	edf	Ref.df	F	p-value
s(temp)	6.869	7.98	192.4	<2e-16 ***

```
R-sq.(adj) = 0.912   Deviance explained = 91.6%
-REML = 450.4   Scale est. = 21.261   n = 150
```

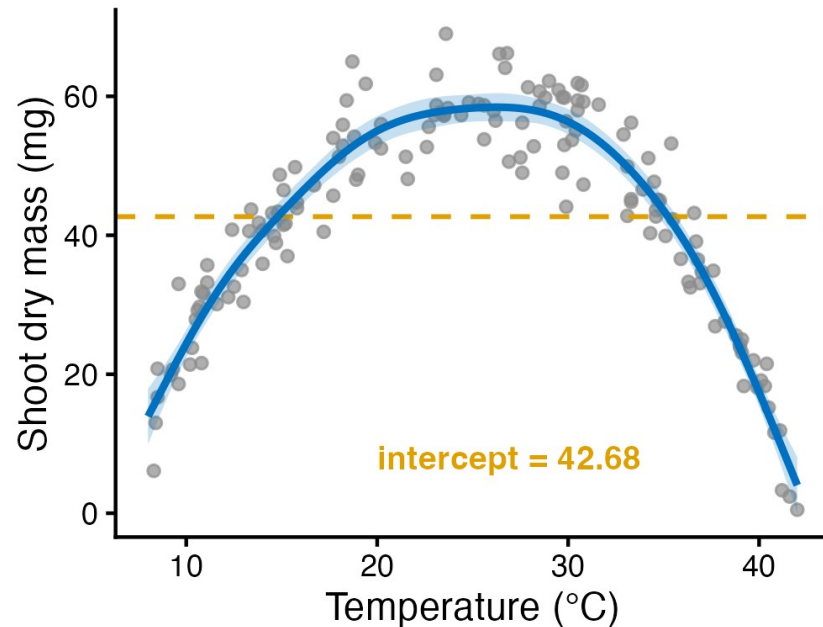
```
> mean(std$mass_mg)
[1] 42.68 ← same as the intercept
```

Smooths are centred at zero

The smooth on its own



+ intercept: the fitted curve



smooth + **intercept** = the fitted curve

edf: the curve's flexibility

```
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML")
> summary(m_gam)

Family: gaussian
Link function: identity

Formula:
mass_mg ~ s(temp)

Parametric coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.6800    0.3765   113.4   <2e-16 ***
---

Approximate significance of smooth terms:
              edf Ref.df      F p-value
s(temp)  6.869   7.98 192.4   <2e-16 *** ← edf
---

R-sq.(adj) =  0.912   Deviance explained = 91.6%
-REML =  450.4   Scale est. = 21.261      n = 150
```

edf = 6.87: worth about 7 parameters. 1 = a straight line

The test: is the smooth flat?

```
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML")  
> summary(m_gam)
```

```
Family: gaussian  
Link function: identity
```

```
Formula:  
mass_mg ~ s(temp)
```

```
Parametric coefficients:  
              Estimate Std. Error t value Pr(>|t|)  
(Intercept)  42.6800     0.3765   113.4   <2e-16 ***  
---
```

```
Approximate significance of smooth terms:  
              edf Ref.df      F p-value  
s(temp) 6.869   7.98 192.4 <2e-16 *** ← is it flat? no  
---
```

```
R-sq.(adj) = 0.912  Deviance explained = 91.6%  
-REML = 450.4  Scale est. = 21.261    n = 150
```

Ref.df, F, p-value: could this curve be zero everywhere?

How much variation the curve captures

```
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML")
> summary(m_gam)
```

Family: gaussian
Link function: identity

Formula:
mass_mg ~ s(temp)

Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	42.6800	0.3765	113.4	<2e-16 ***

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value
s(temp)	6.869	7.98	192.4	<2e-16 ***

R-sq.(adj) = 0.912 Deviance explained = 91.6% ← variation captured
-REML = 450.4 Scale est. = 21.261 n = 150

For binomial or Poisson GAMs, report **deviance explained**

REML score, residual variance, n

```
> m_gam <- gam(mass_mg ~ s(temp), data = std, method = "REML")  
> summary(m_gam)
```

```
Family: gaussian  
Link function: identity
```

```
Formula:  
mass_mg ~ s(temp)
```

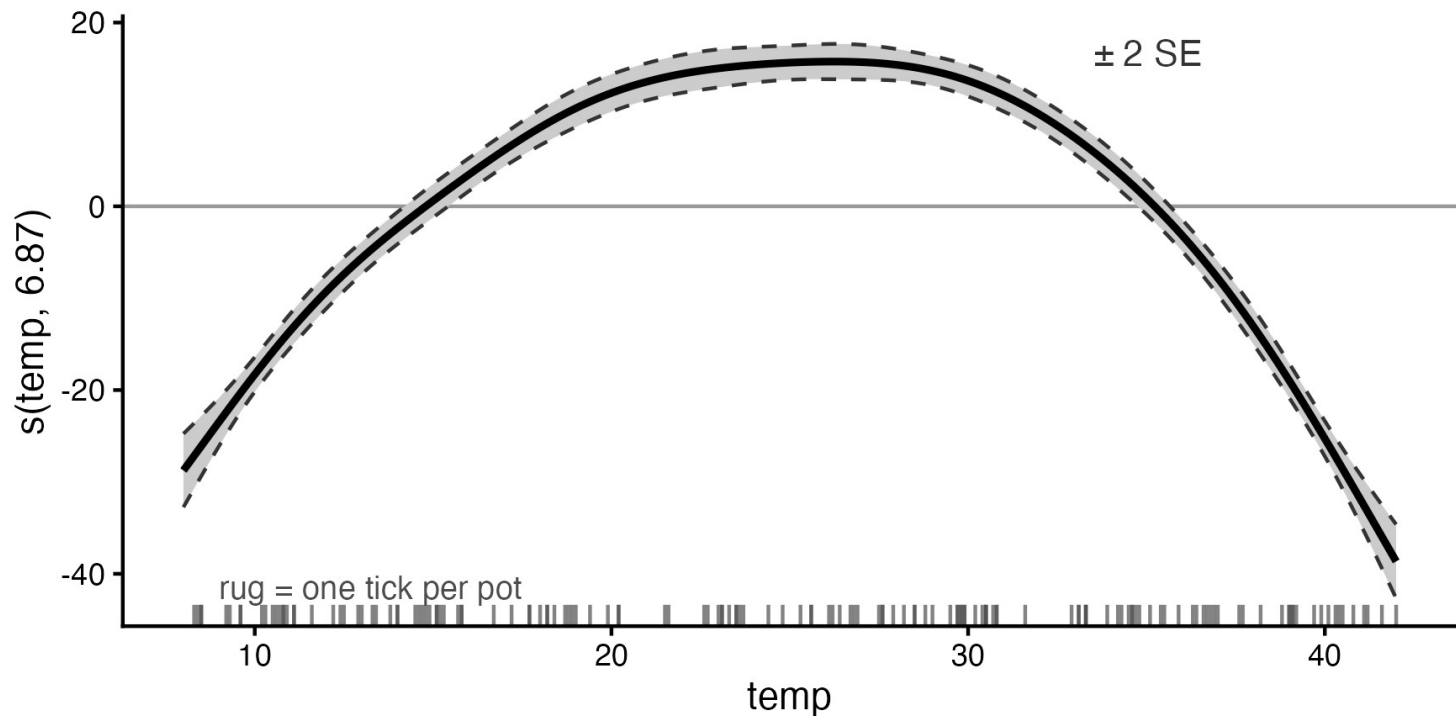
```
Parametric coefficients:  
              Estimate Std. Error t value Pr(>|t|)  
(Intercept)  42.6800    0.3765   113.4   <2e-16 ***  
---
```

```
Approximate significance of smooth terms:  
              edf Ref.df      F p-value  
s(temp)  6.869    7.98 192.4   <2e-16 ***  
---
```

```
R-sq.(adj) = 0.912   Deviance explained = 91.6%  
-REML = 450.4   Scale est. = 21.261   n = 150
```

Scale est. = residual variance: $\sqrt{21.3} \approx 4.6$ mg scatter per pot

plot(m): the partial effect

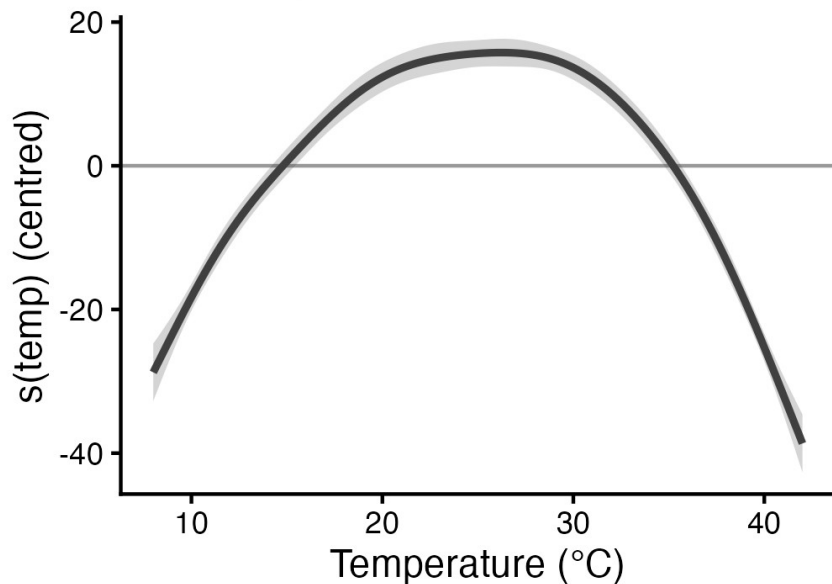


Centred at 0, on the link scale, ± 2 SE, a rug for the data

For a figure: `predict()`

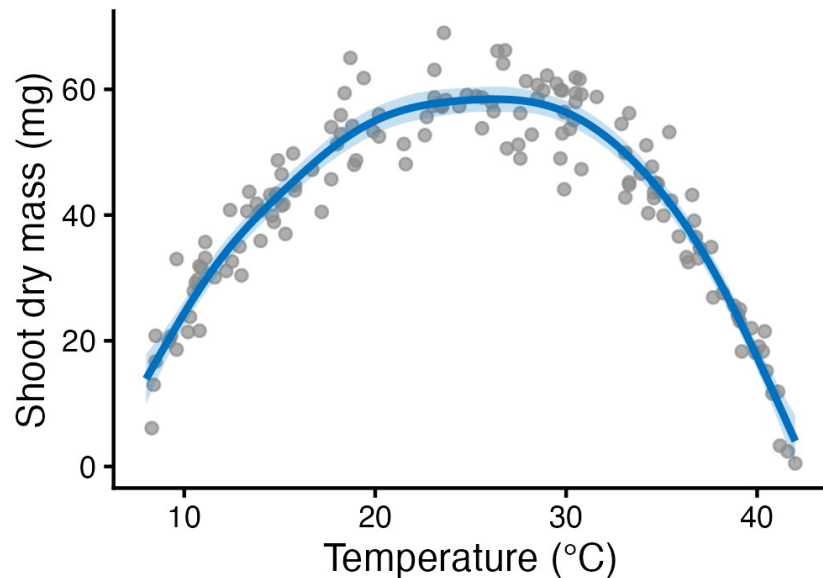
`plot(m): partial effect`

centred at 0, ± 2 SE



`predict(m, se.fit = TRUE)`

data + fit + 95% CI, in mg



`predict(m, se.fit = TRUE)`: back on the scale of y, with the data

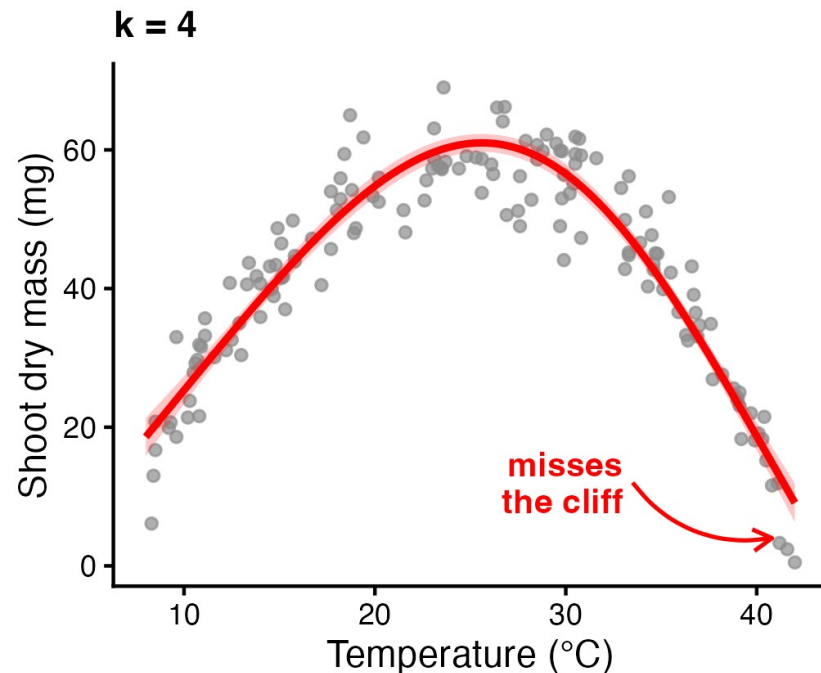
k too small: gam.check() complains

```
> m4 <- gam(mass_mg ~ s(temp, k = 4), data = std,  
+           method = "REML")  
> k.check(m4)
```

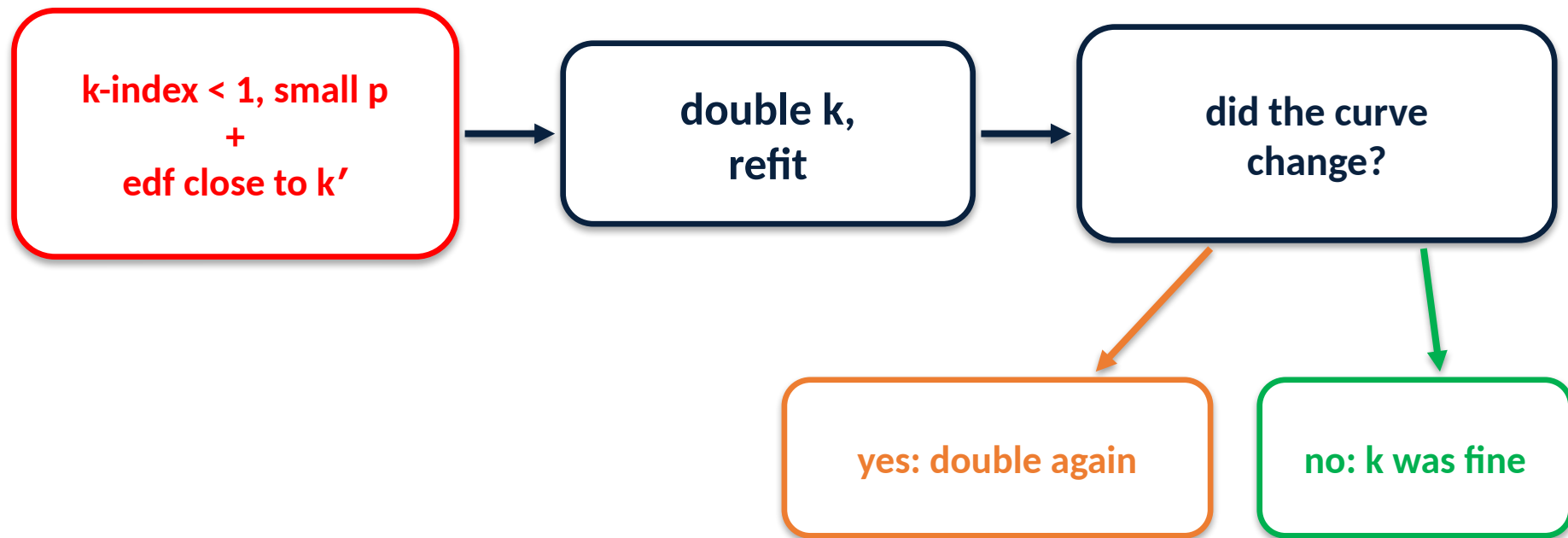
	k'	edf	k-index	p-value
s(temp)	3	2.987211	0.7854077	0.0025

k-index < 1 and **small p**

edf 2.99 vs **k' = 3**: at the ceiling



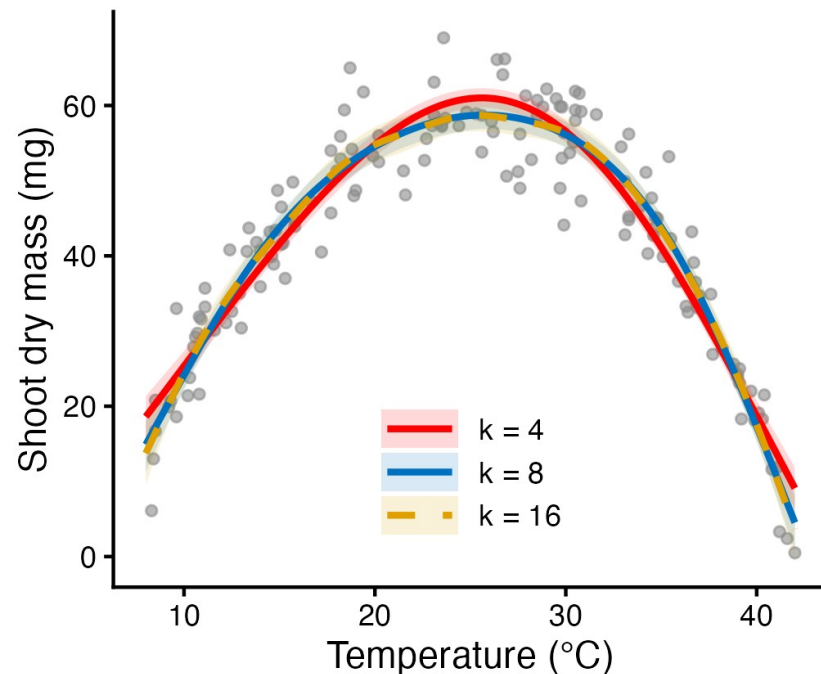
The k check: double it and see



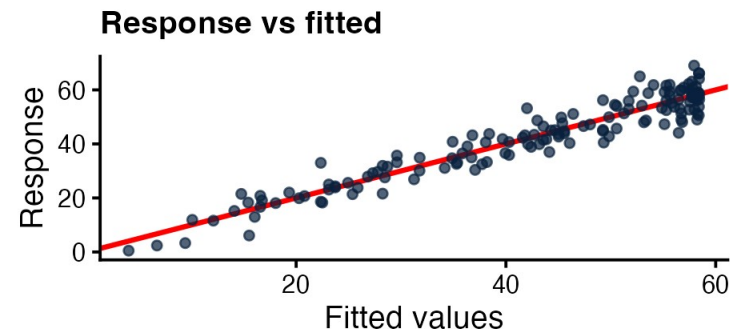
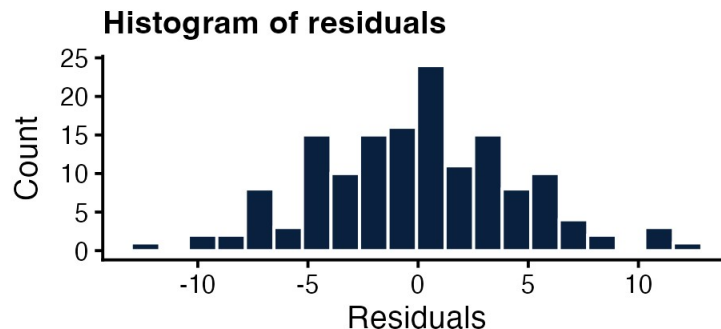
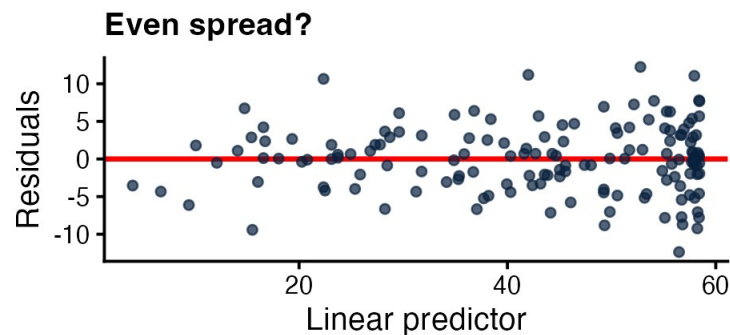
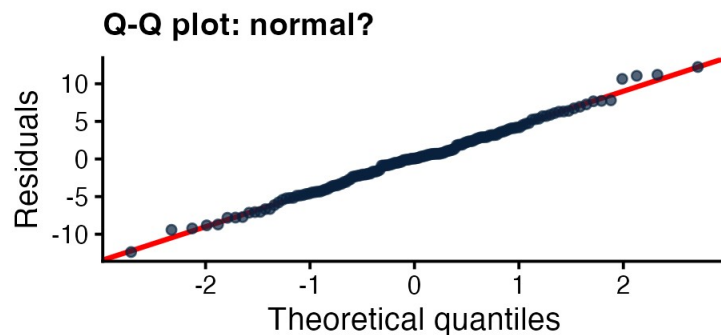
k = 8 and k = 16: the same curve

```
> m8 <- gam(mass_mg ~ s(temp, k = 8), data = std,  
+           method = "REML")  
> m16 <- gam(mass_mg ~ s(temp, k = 16), data = std,  
+            method = "REML")  
> c(k4 = summary(m4)$edf, k8 = summary(m8)$edf,  
+   k16 = summary(m16)$edf)  
      k4      k8      k16  
2.987211 6.056363 7.617236
```

The **curve** is the test, not the edf. Stop at 8.

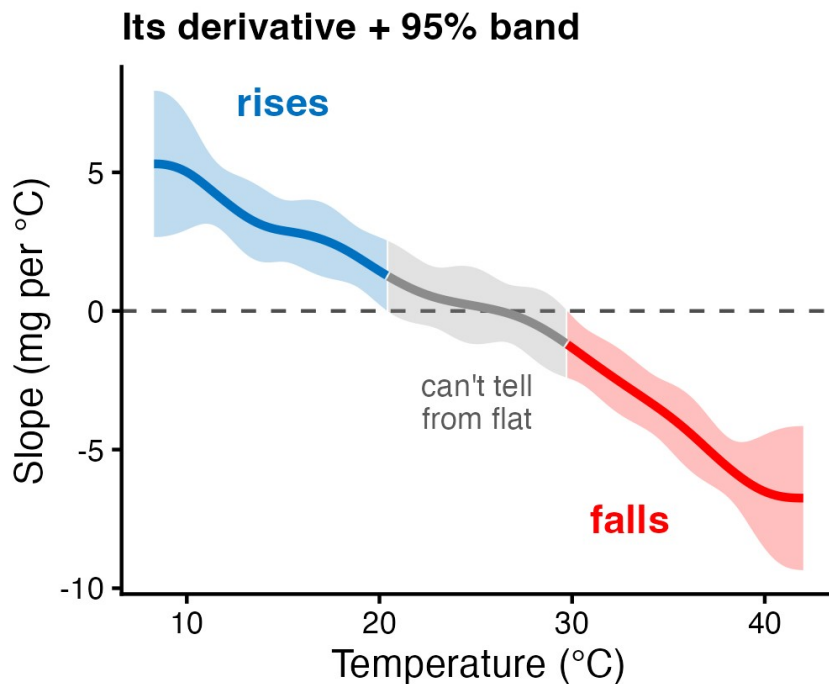
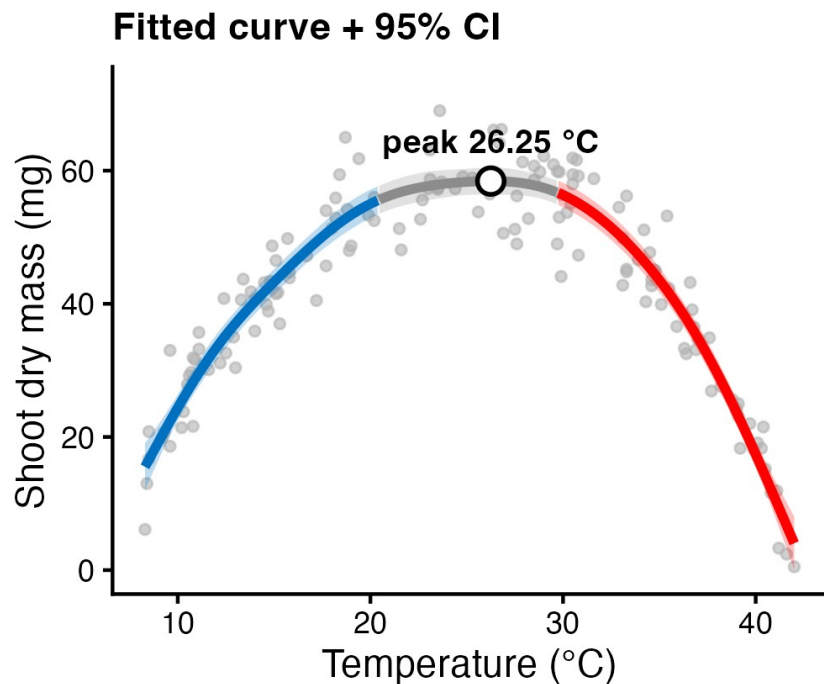


Same residual checks as lm()



A GAM fixes the **shape**, not unequal spread or the wrong family

No single slope



Show where it **rises** and where it **falls**

Reporting a GAM

Methods

Shoot mass was modelled as a smooth function of temperature with a GAM (mgcv, REML, $k = 10$).

Results

Temperature had a strong non-linear effect on shoot mass (edf = 6.87, $F = 192.4$, $p < 0.001$; deviance explained 91.6%), peaking at 26.25 °C (Fig. 1).

The **figure** is the result

Live coding: summary, plot, gam.check



`Week4_live_2_gam.R`

Part 1: sections 1 to 3

- 1 The summary, line by line
- 2 Plotting a GAM
- 3 Is k big enough?

1 Bend the line

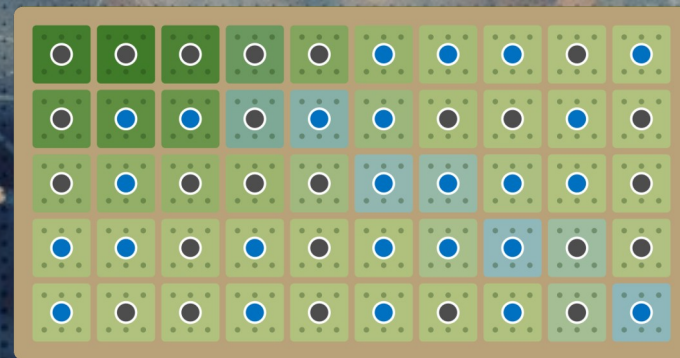
2 Splines

3 GAMs

4 Reading a GAM

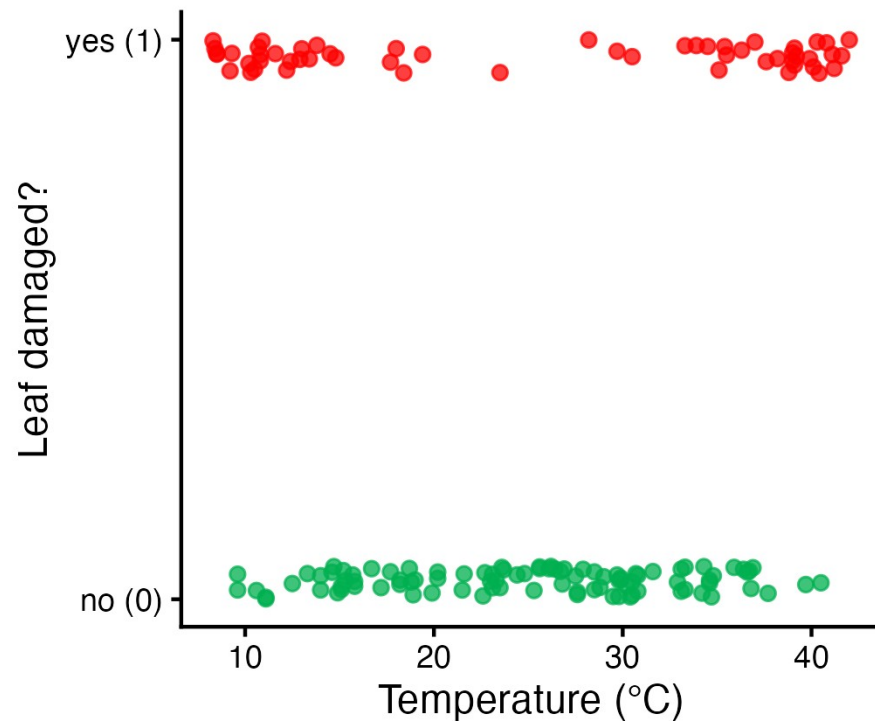
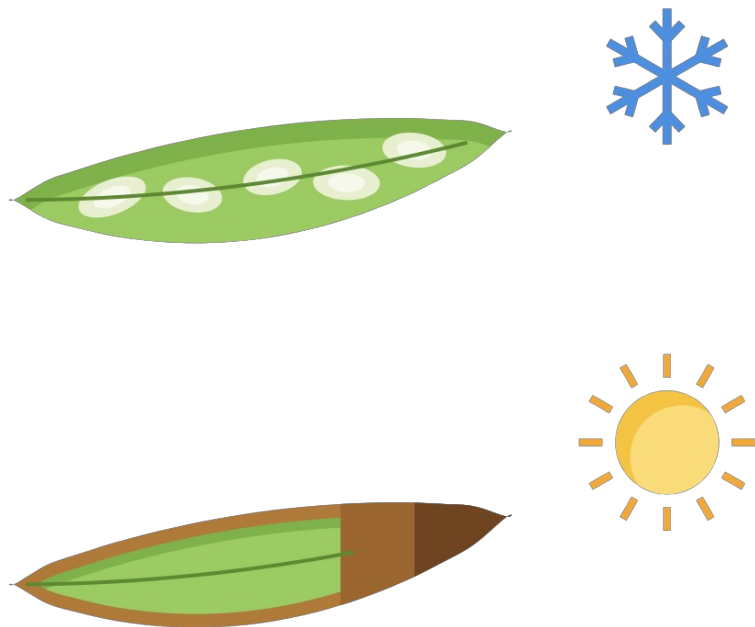
► **5 GAM toolkit**

6 Mechanistic models



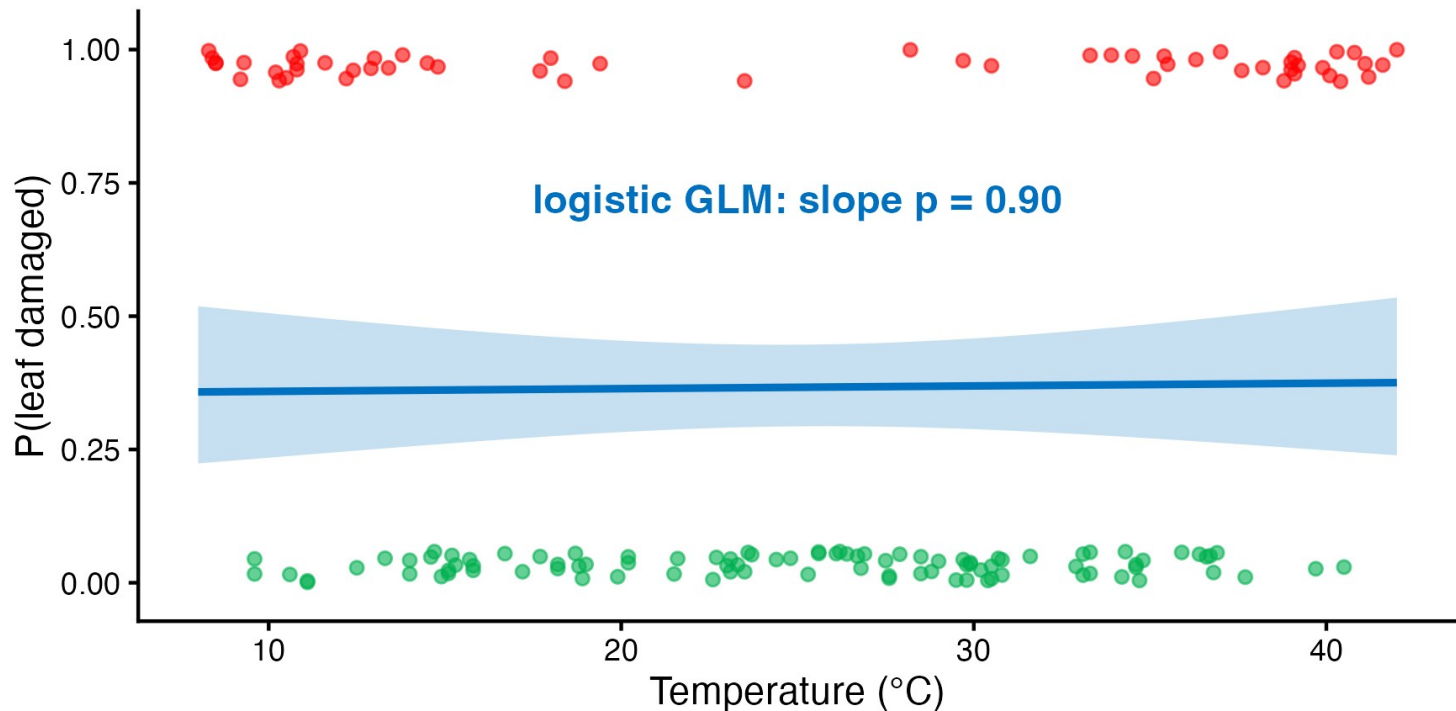
○ wildtype ● edited

Same pots: was the leaf damaged?



A **yes** / **no** response: chilling at one end, scorch at the other

Logistic GLM: misses the U



Straight on the logit scale: slope $p = 0.9$

family = binomial: a logistic GAM

```
> m_dmg <- gam(damaged ~ s(temp), family = binomial, data = std,
+             method = "REML")
> summary(m_dmg)
```

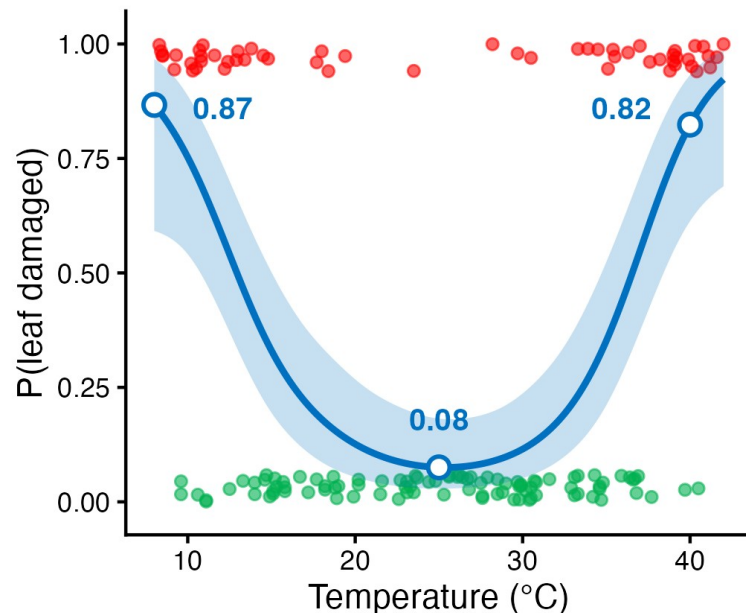
```
Family: binomial ← yes/no
Link function: logit
```

```
Formula:
damaged ~ s(temp)
```

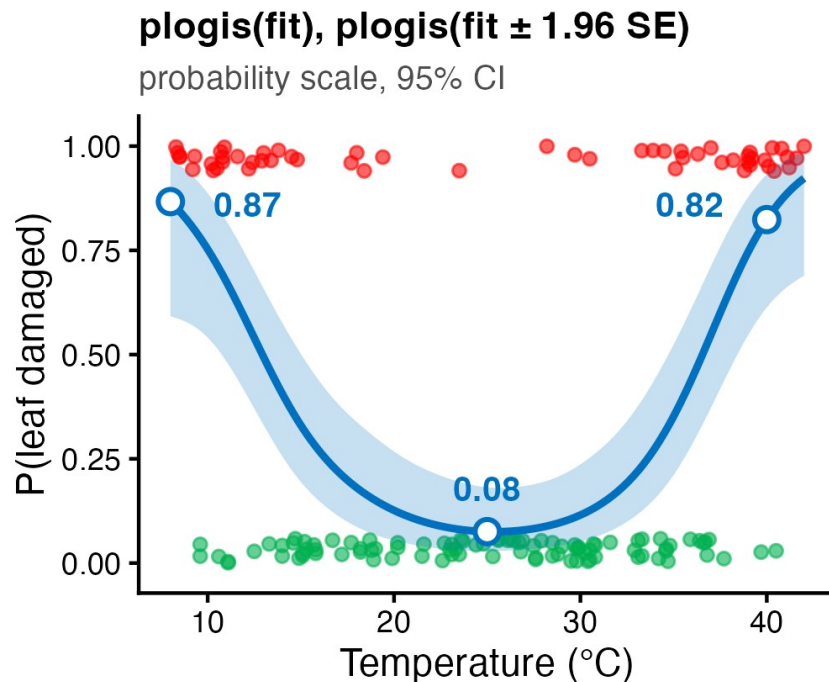
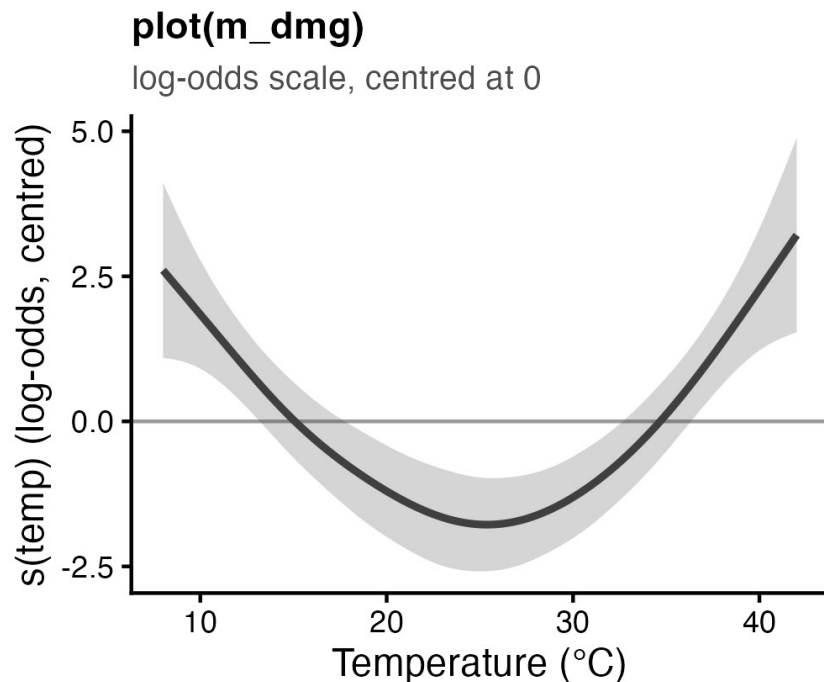
```
Parametric coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.7303     0.2178   -3.353 0.000798 ***
---
```

```
Approximate significance of smooth terms:
              edf Ref.df Chi.sq p-value
s(temp) 3.471  4.303   38.8 1.1e-07 *** ← a curve, not flat
---
```

```
R-sq.(adj) = 0.335   Deviance explained = 28.7%
-REML = 75.359   Scale est. = 1           n = 150
```

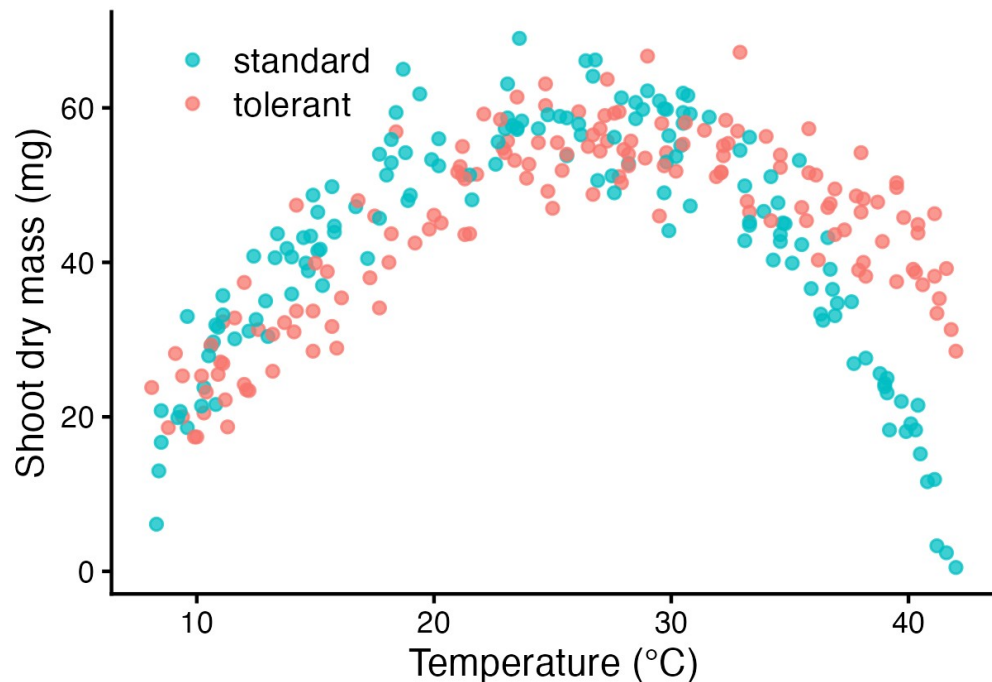
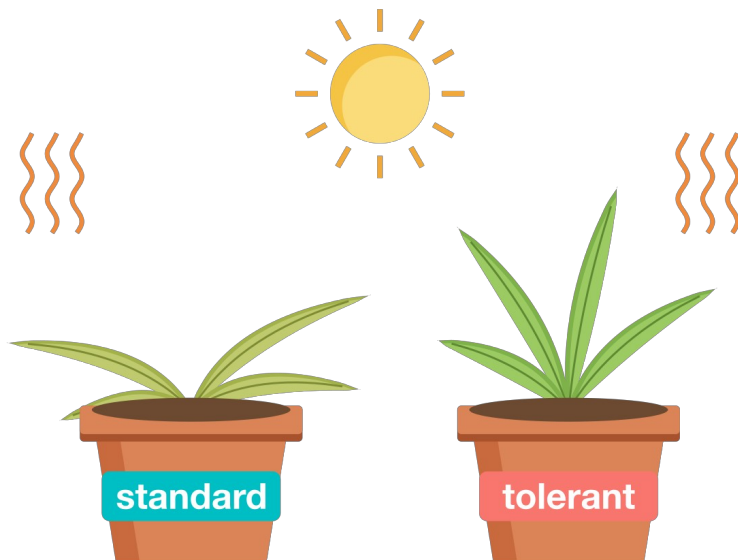


plot() shows log-odds



Build the band on the link scale, then **plogis()**

Two varieties: two curves?



The **tolerant** line peaks later and keeps growing in the heat

by = variety: one smooth per variety

```
> seedlings$variety <- factor(seedlings$variety)
> m_by <- gam(mass_mg ~ variety + s(temp, by = variety),
+           data = seedlings, method = "REML")
> summary(m_by)
```

Family: gaussian
Link function: identity

Formula:
mass_mg ~ variety + s(temp, by = variety)

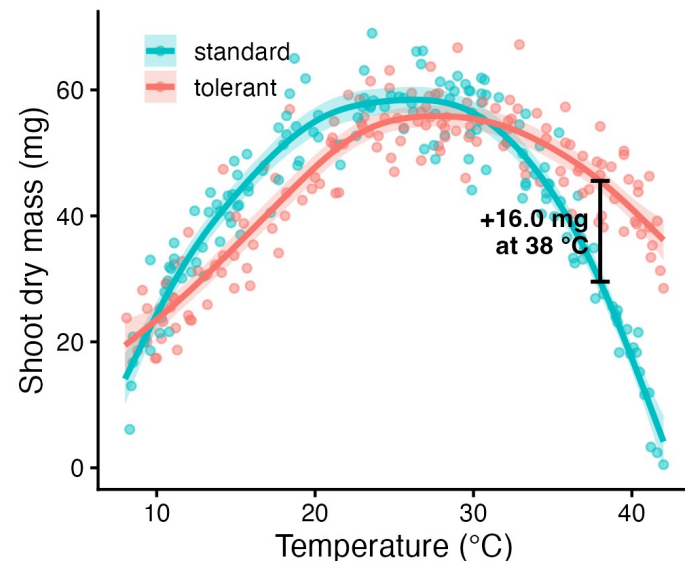
Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	42.5874	0.3916	108.759	< 2e-16 ***
varietytolerant	1.8393	0.5535	3.323	0.00101 ** ← level

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value
s(temp):varietystandard	6.761	7.885	181.7	<2e-16 ***
s(temp):varietytolerant	5.473	6.609	116.6	<2e-16 ***

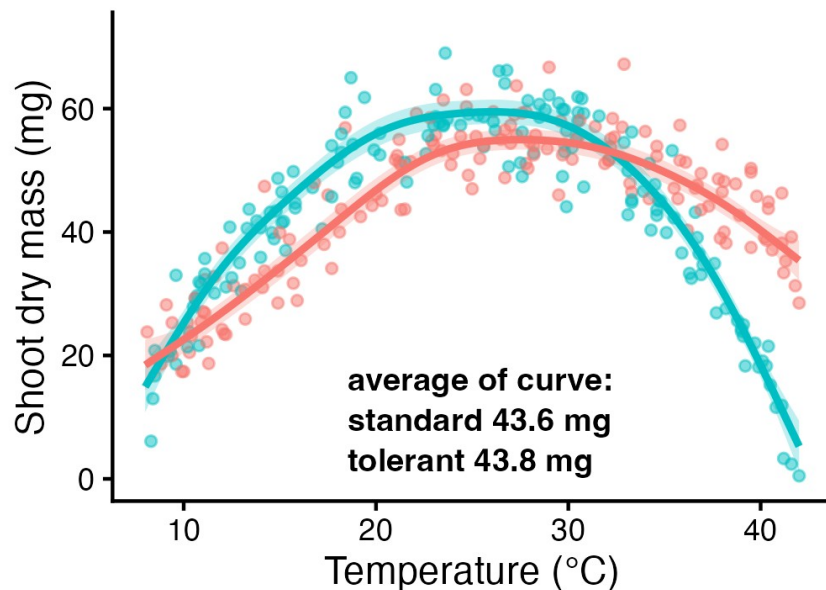
R-sq.(adj) = 0.881 Deviance explained = 88.6%
-REML = 907.63 Scale est. = 22.777 n = 300



by = needs the factor too

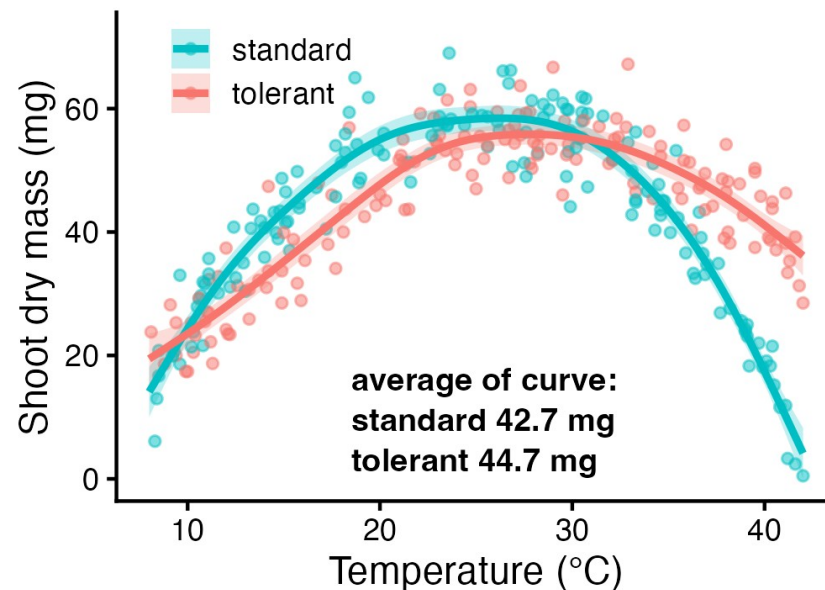
```
mass ~ s(temp, by = variety)
```

no variety term: wrong



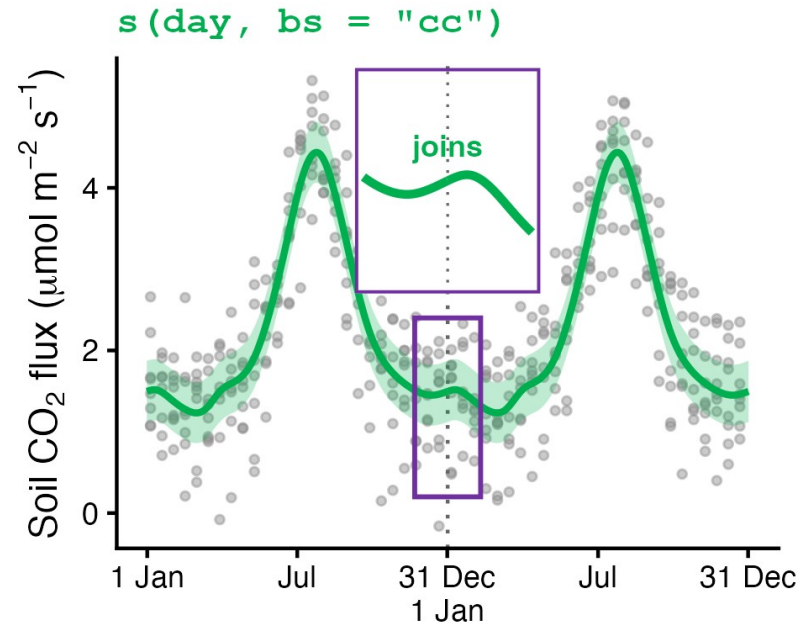
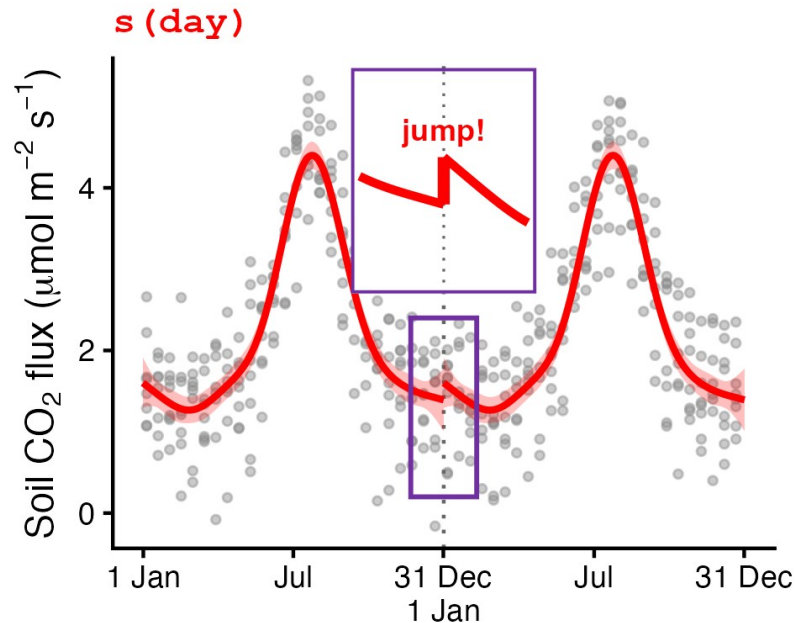
```
mass ~ variety + s(temp, by = variety)
```

with the main effect: right



variety + s(temp, by = variety): always add the factor itself

bs = "cc": the year wraps around



A cyclic smooth **joins up** between 31 December and 1 January

bs = "re": a random effect

```
> soil$collar <- factor(soil$collar)
> m_soil <- gam(flux ~ s(day, bs = "cc", k = 20) +
+             s(collar, bs = "re"),
+             data = soil, method = "REML",
+             knots = list(day = c(0.5, 365.5)))
> anova(m_soil)
```

Family: gaussian
Link function: identity

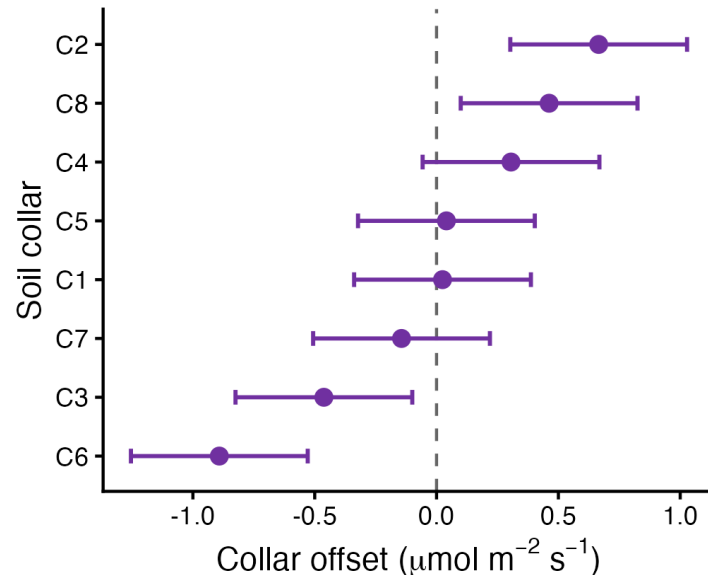
Formula:
flux ~ s(day, bs = "cc", k = 20) + s(collar, bs = "re")

Approximate significance of smooth terms:

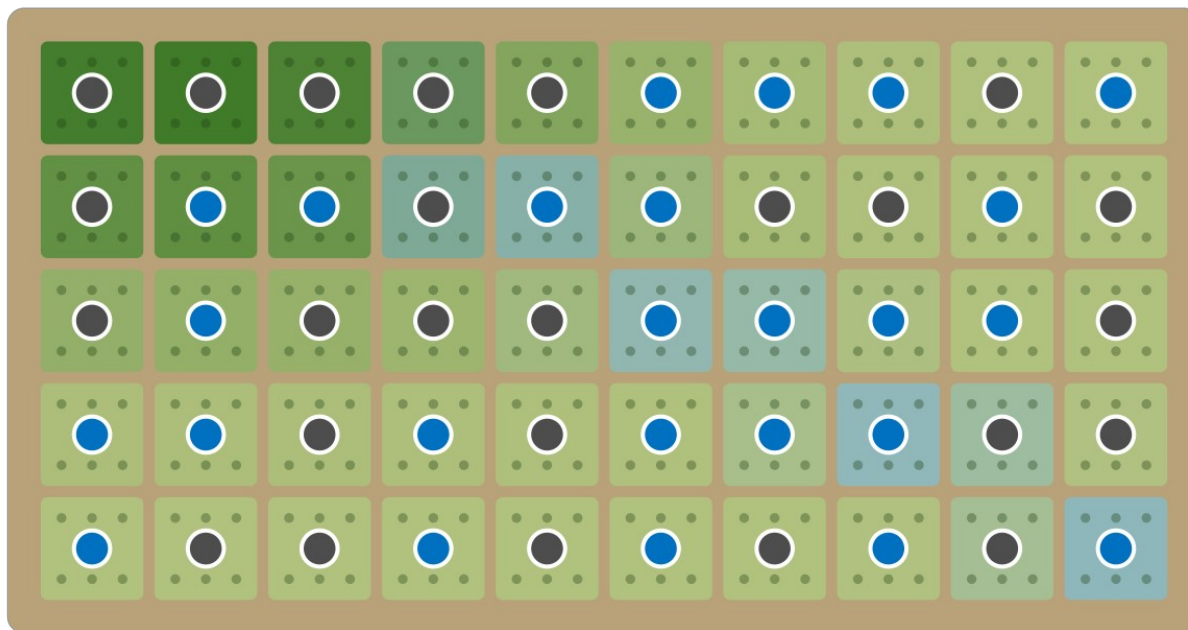
	edf	Ref.df	F	p-value	
s(day)	12.104	18.000	187.7	<2e-16	
s(collar)	6.933	7.000	103.8	<2e-16	← random intercept

```
> round(gam.vcomp(m_soil)$vc, 3)
```

	std.dev	lower	upper	
s(day)	0.004	0.003	0.007	
s(collar)	0.507	0.299	0.860	← collar SD
scale	0.359	0.335	0.385	



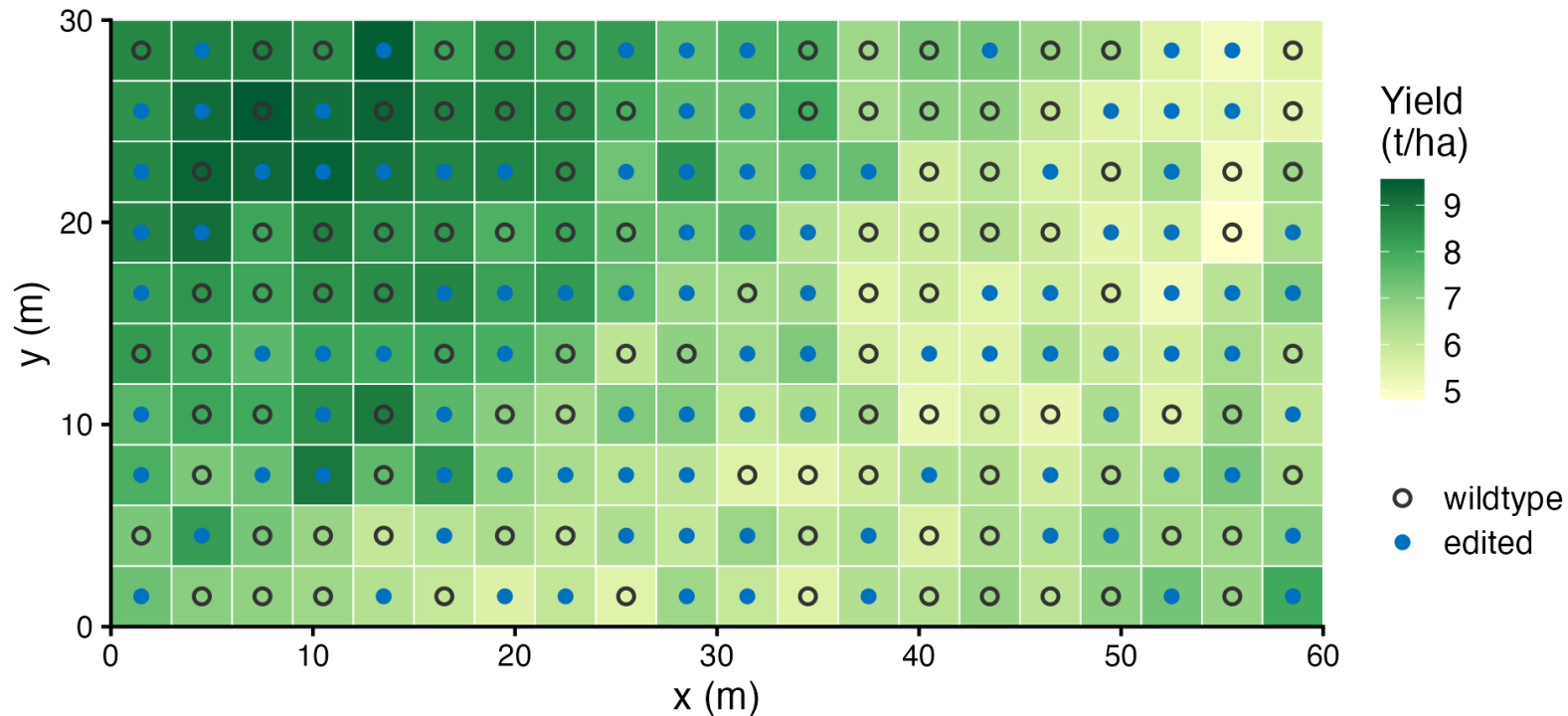
A rice field trial



● wildtype ● edited

Edited line vs wild type, 200 plots. (Hypothetical, simulated data.)

The field is not uniform



The gradient is **much bigger** than any genotype difference

Ignore the field

```
> field$genotype <- factor(field$genotype,  
+                           levels = c("wildtype", "edited"))  
> m_naive <- lm(yield ~ genotype, data = field)  
> summary(m_naive)
```

Call:

```
lm(formula = yield ~ genotype, data = field)
```

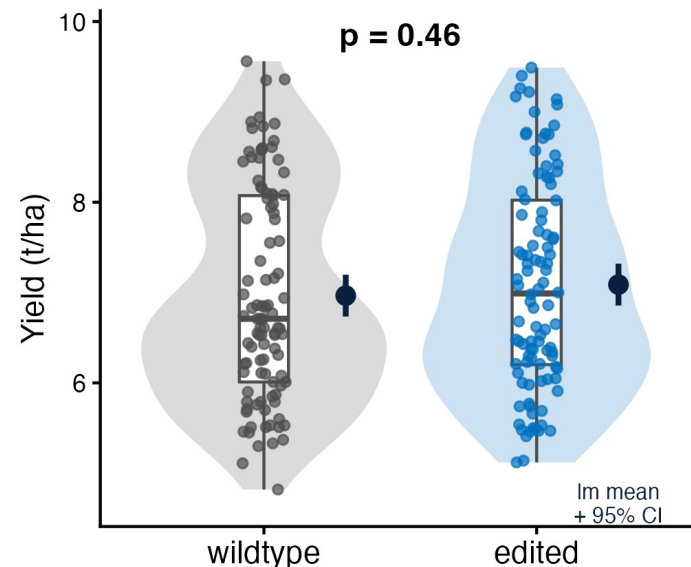
Residuals:

	Min	1Q	Median	3Q	Max
	-2.146	-0.913	-0.227	1.042	2.594

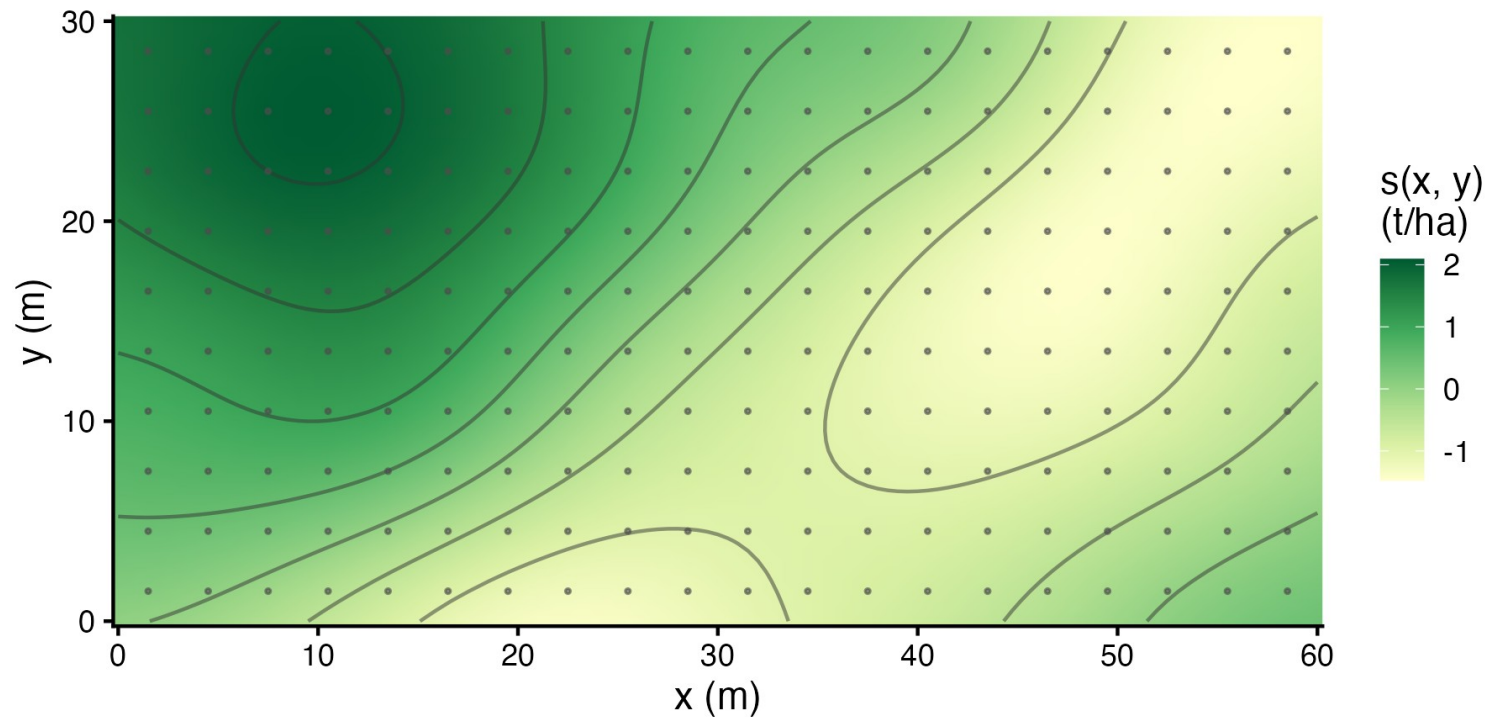
Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	6.9660	0.1174	59.317	<2e-16 ***
genotypeedited	0.1220	0.1661	0.735	0.463

Residual standard error: 1.174 on 198 degrees of freedom
Multiple R-squared: 0.002718, Adjusted R-squared: -0.002319
F-statistic: 0.5396 on 1 and 198 DF, p-value: 0.4635



$s(x, y)$: a smooth surface



A smooth of **both coordinates**: a fertility map learned from the data

Model the field: the effect appears

```
> m_sp <- gam(yield ~ genotype + s(x_m, y_m), data = field,
+             method = "REML")
> summary(m_sp)
```

Family: gaussian
Link function: identity

Formula:
yield ~ genotype + s(x_m, y_m)

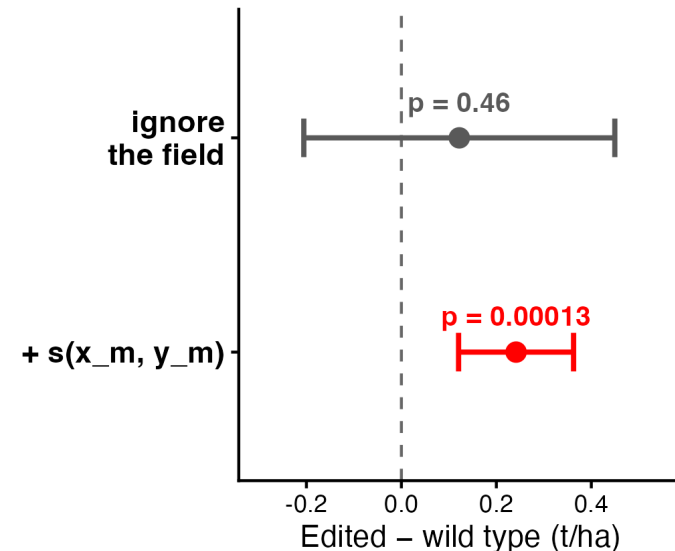
Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	6.90622	0.04242	162.79	< 2e-16 ***
genotypeedited	0.24156	0.06178	3.91	0.000132 ***

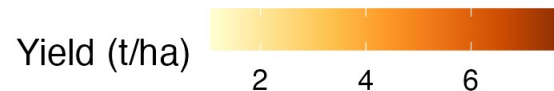
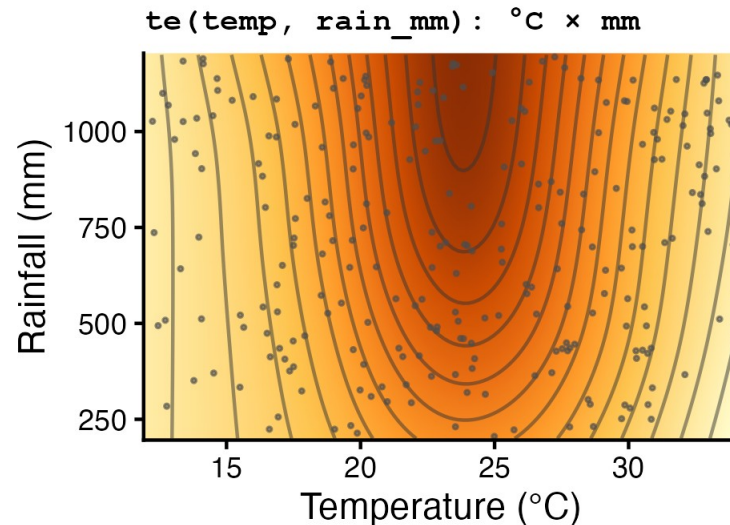
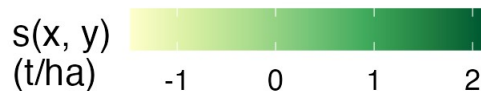
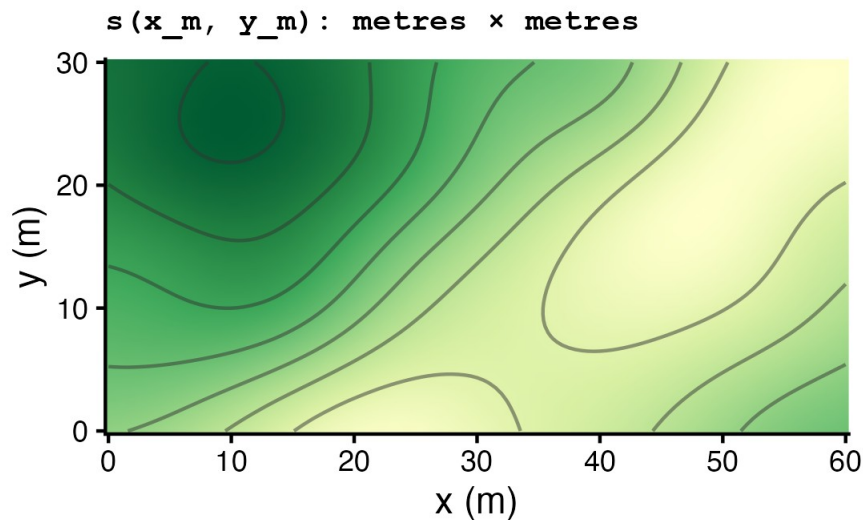
Approximate significance of smooth terms:

	edf	Ref.df	F	p-value
s(x_m,y_m)	21.39	25.87	54.56	<2e-16 ***

R-sq.(adj) = 0.877 Deviance explained = 89.1%
-REML = 137.08 Scale est. = 0.16911 n = 200



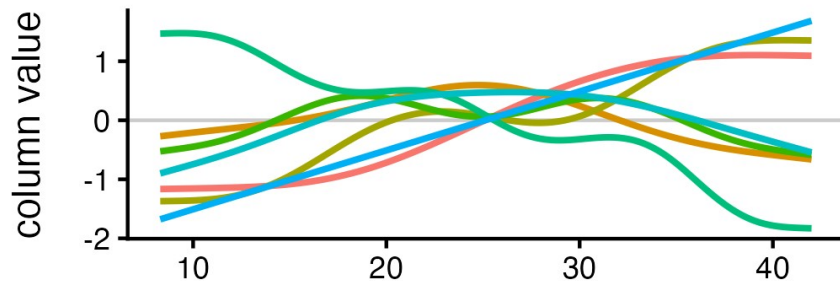
Same units: $s(x, y)$. Different units: $te(x, z)$.



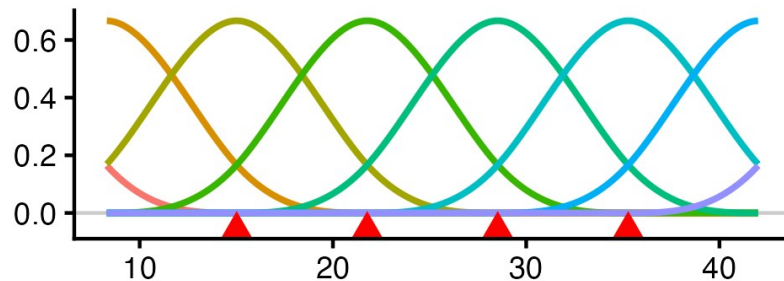
`te()` gives each direction its own smoothness

$s(x, bs = \dots)$: which columns?

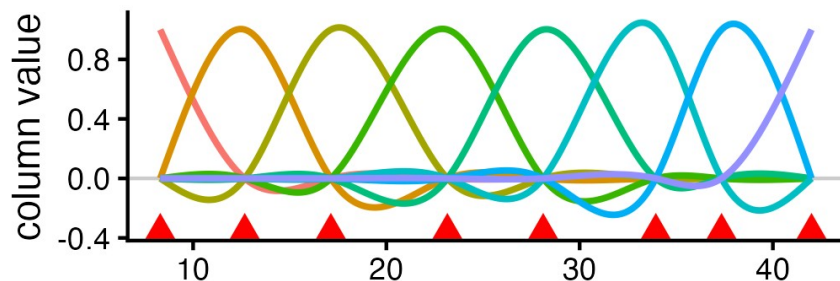
default (tp): waves



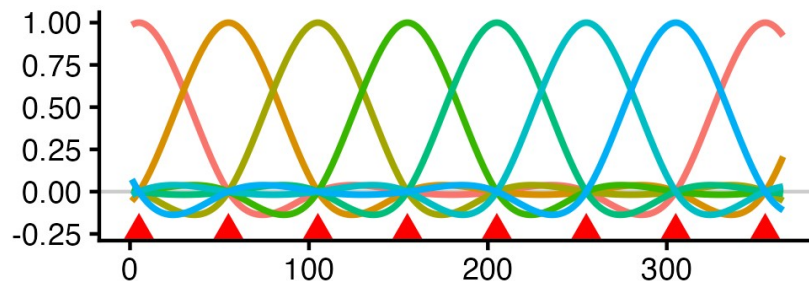
"bs" / "ps": B-spline bumps



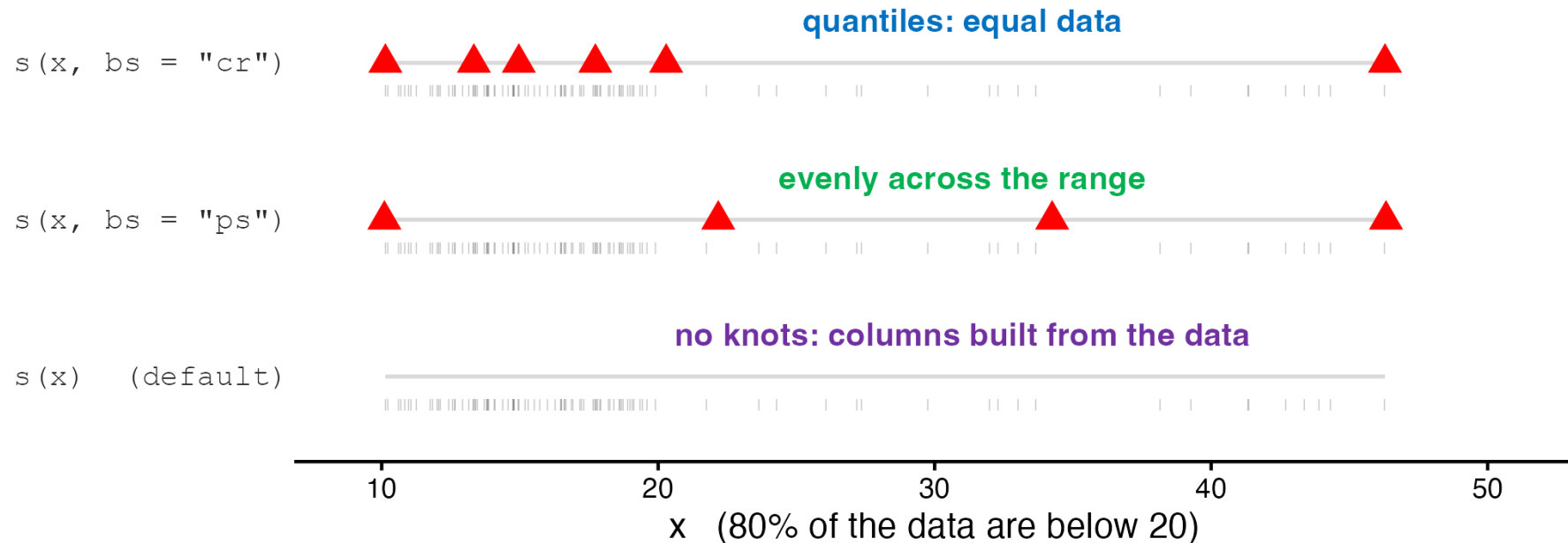
"cr": 1 at its knot, 0 at the others



"cc": like cr, but the ends wrap



Where `s()` puts its knots



cr: quantiles. **ps**: evenly spaced. **Default `s()`**: no knots.

Live poll

Multiple choice

You want the N rate where yield stops increasing, with a 95% CI. Which tool?

- A GAM: $s(N)$
- B polynomial
- C nls() with a plateau
- D loess



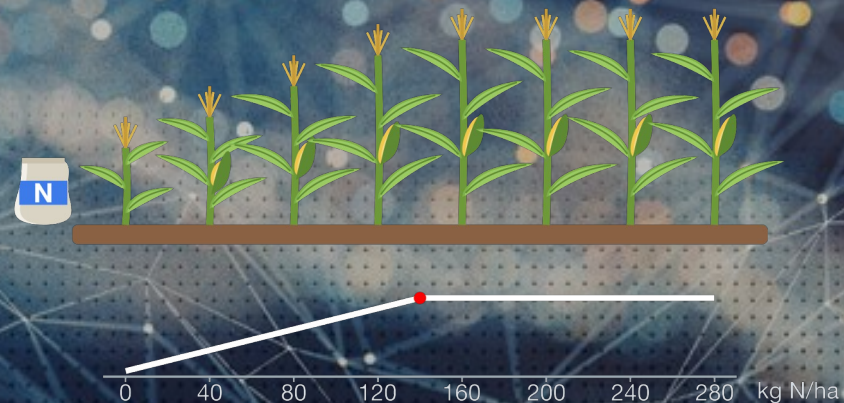
Scan with your phone

[Join on your phone or laptop: greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w4-describe-or-estimate](https://greymonroe.github.io/live-poll-test/submit.html?poll=pls206-w4-describe-or-estimate)

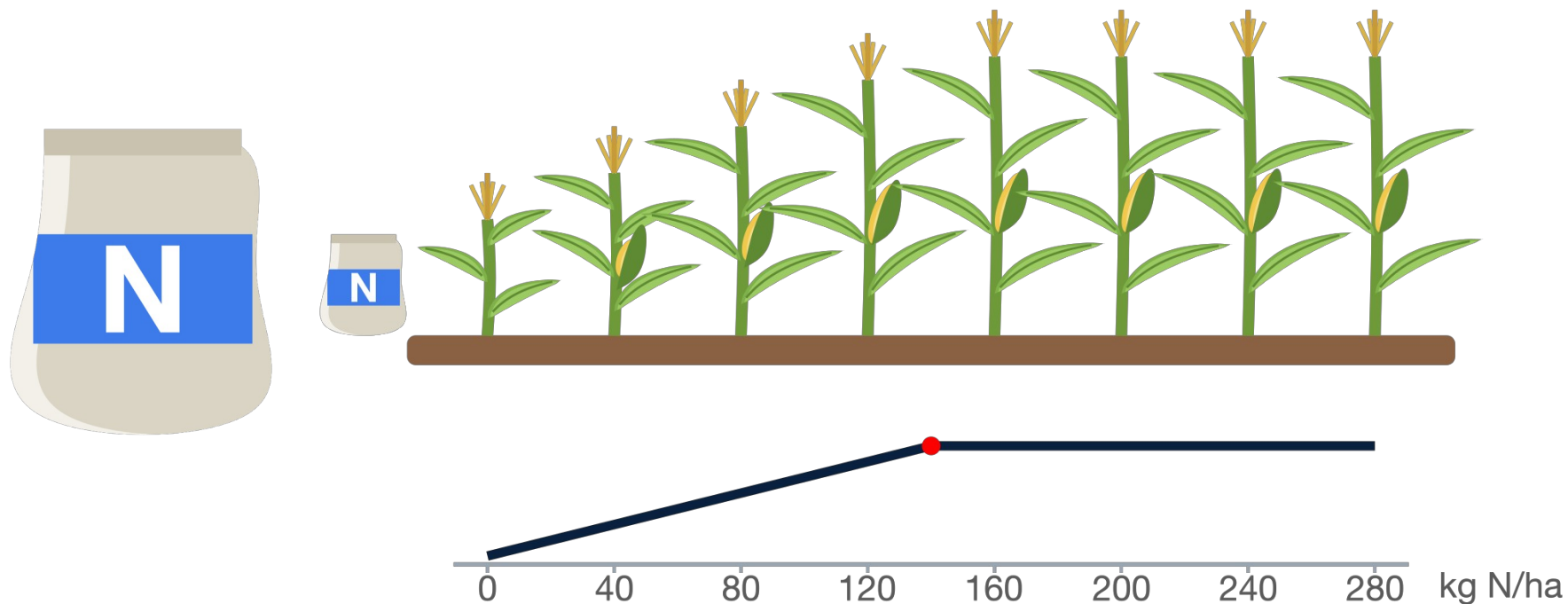
[host screen](#)

- 1 Bend the line
- 2 Splines
- 3 GAMs
- 4 Reading a GAM
- 5 GAM toolkit

► 6 Mechanistic models

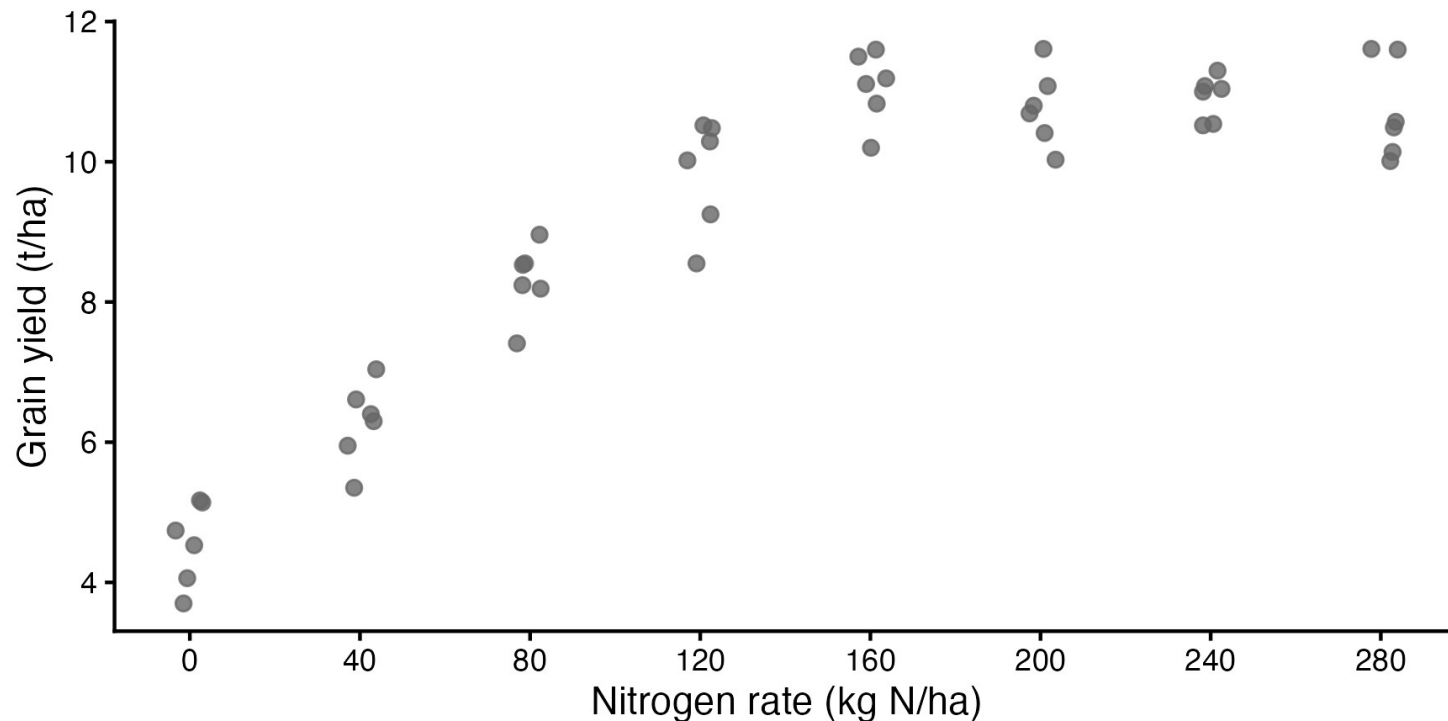


A maize nitrogen trial



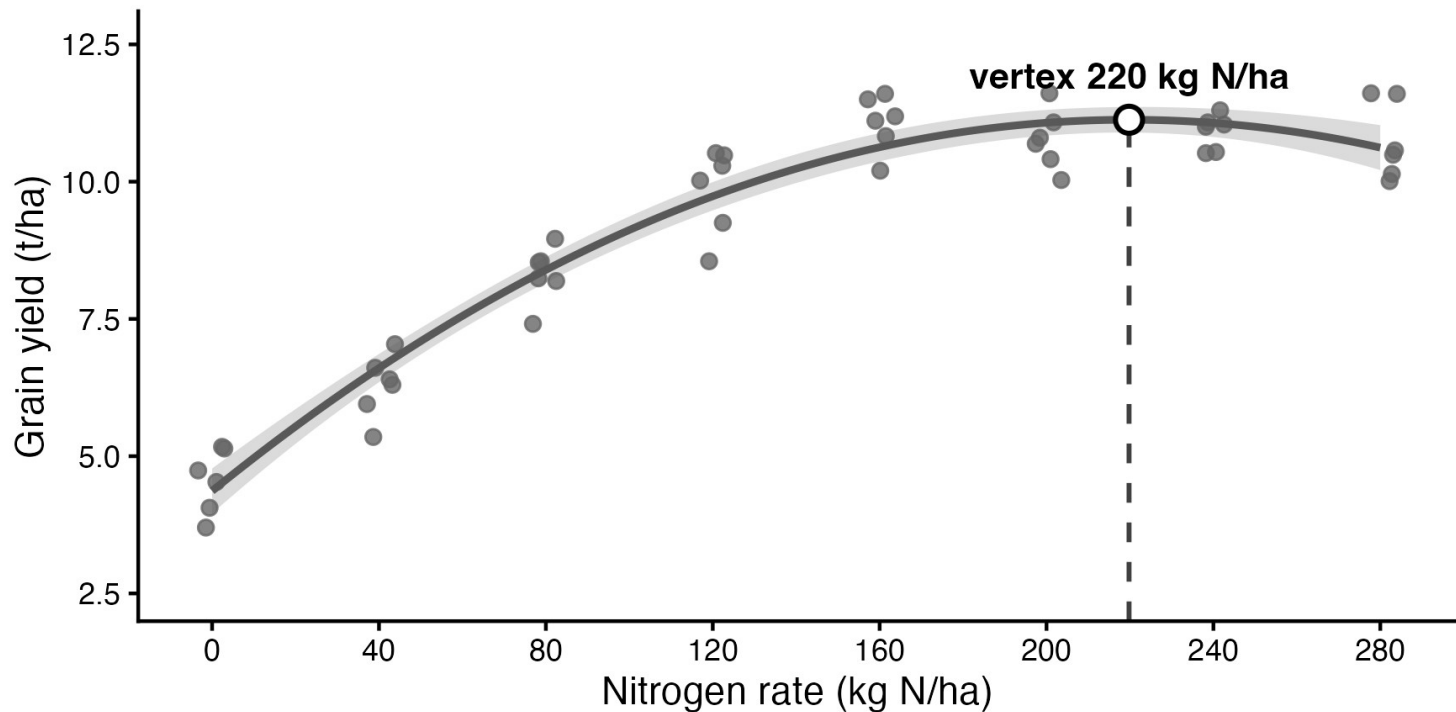
8 N rates, 6 plots each. (Hypothetical, simulated data.)

How much nitrogen is enough?



A **ramp**, then a **plateau**

The quadratic's optimum



Quadratic vertex: 220 kg N/ha

A model whose parameters mean something

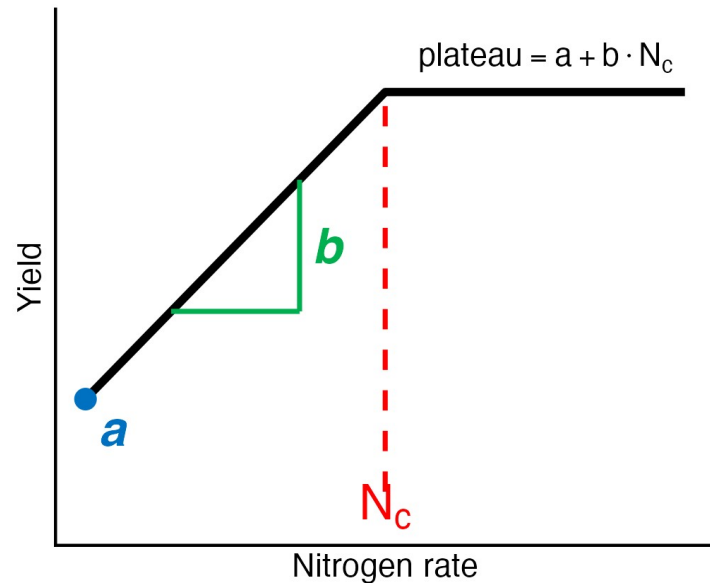
$$\text{yield} = a + b \cdot \min(N, N_c)$$

$$= 4.56 + 0.0448 \cdot \min(N, 140.9)$$

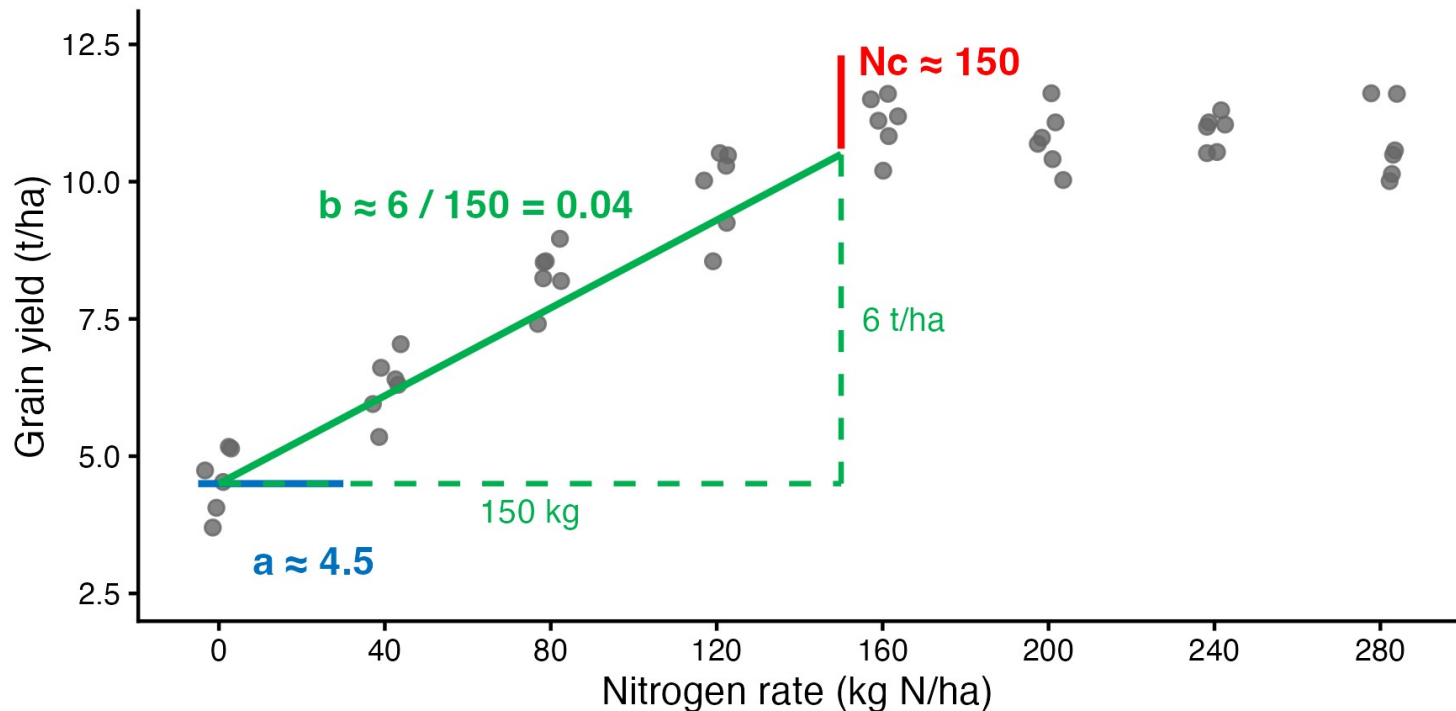
$$\text{plateau} = a + b \cdot N_c = 10.87 \text{ t/ha}$$

a : yield with no N b : t/ha per kg N

N_c : where the plateau starts N : N rate, kg N/ha



nls() needs starting values



Read them off the plot: $a \approx 4.5$, $b \approx 0.04$, $N_c \approx 150$

An error? Check your starting values

```
> try(nls(yield ~ a + b * pmin(N_rate, Nc), data = maize,  
+       start = list(a = 5, b = 0.04, Nc = 400))  
Error in nlsModel(formula, mf, start, wts, scaleOffset = scOff,  
nDcentral = nDcntr) :  
singular gradient matrix at initial parameter estimates
```

A break point beyond the data: **nls()** cannot start

nls(): the fitted plateau model

```
> m_lp <- nls(yield ~ a + b * pmin(N_rate, Nc), data = maize,  
+           start = list(a = 5, b = 0.04, Nc = 150))  
> summary(m_lp)
```

Formula: yield ~ a + b * pmin(N_rate, Nc)

Parameters:

	Estimate	Std. Error	t value	Pr(> t)	
a	4.561e+00	1.926e-01	23.68	<2e-16 ***	
b	4.481e-02	2.574e-03	17.41	<2e-16 ***	
Nc	1.409e+02	5.898e+00	23.89	<2e-16 ***	← Nc, kg N/ha

Residual standard error: 0.564 on 45 degrees of freedom

Number of iterations to convergence: 2

Achieved convergence tolerance: 2.187e-09

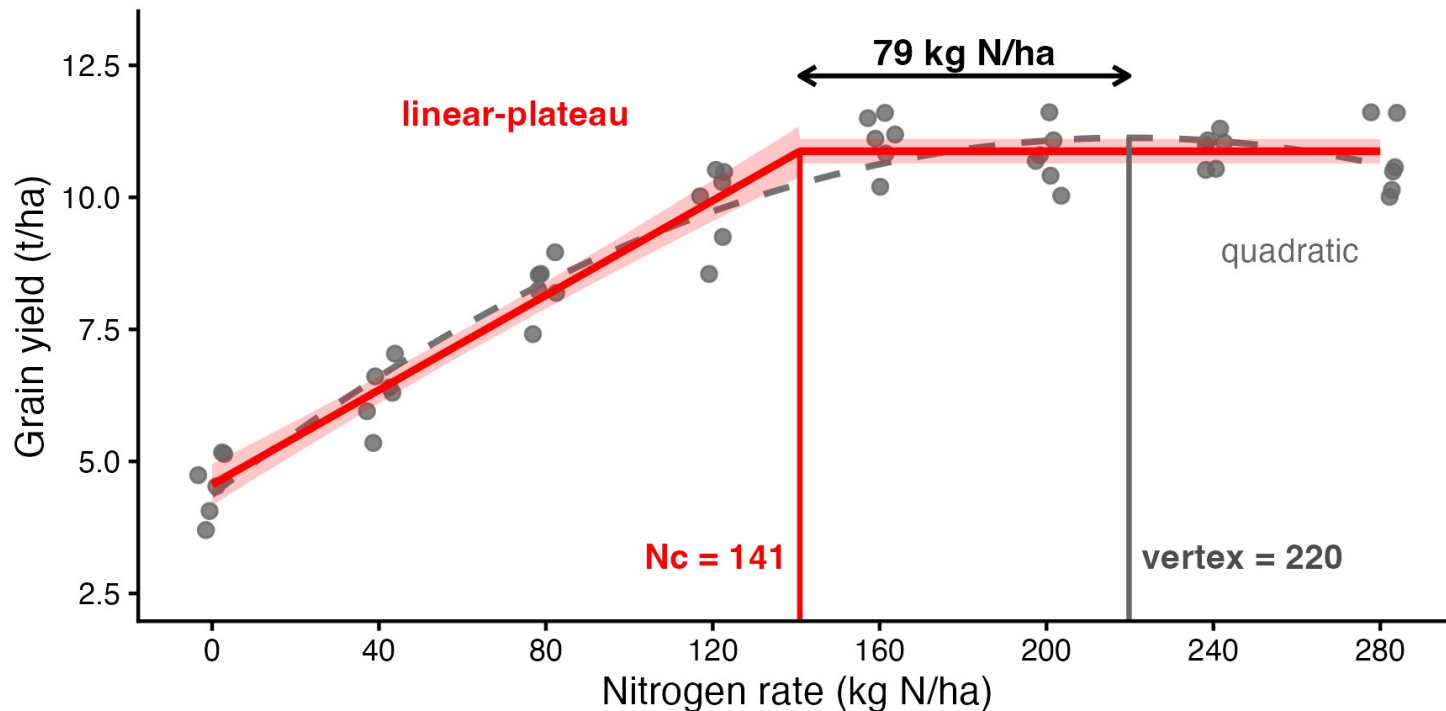
Nc = 141 kg N/ha; a = 4.56 t/ha; b = 0.045 t/ha per kg N

A confidence interval for Nc

```
> confint(m_lp)
Waiting for profiling to be done...
              2.5%          97.5%
a      4.17268886    4.9486445
b      0.03962376    0.0499929
Nc 129.96182740 153.9820056 ← 95% CI
```

95% CI for Nc: **130 to 154 kg N/ha**

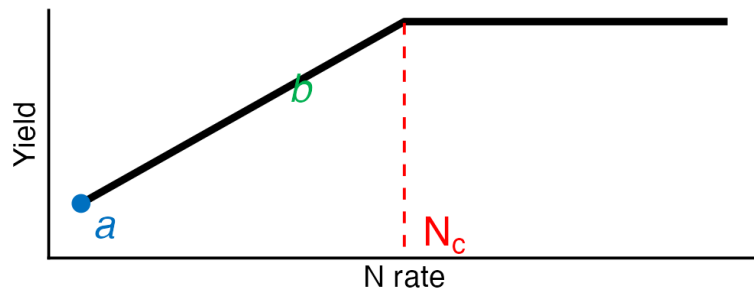
The quadratic overshoots



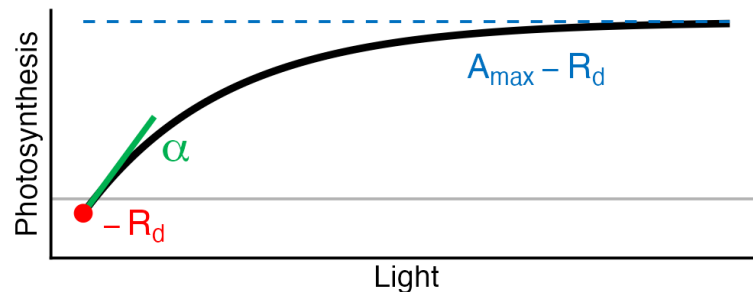
Nc = 141 vs vertex = 220: 79 kg N/ha apart

A shape for every mechanism

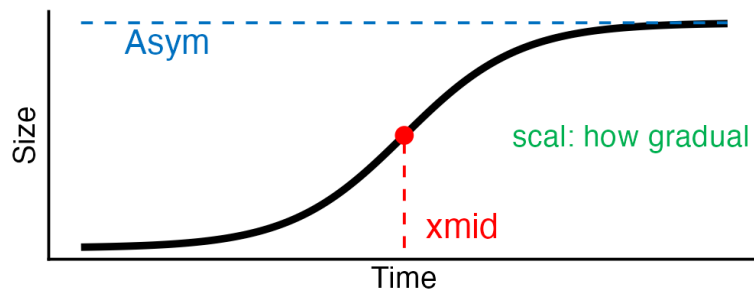
Linear-plateau: yield vs N



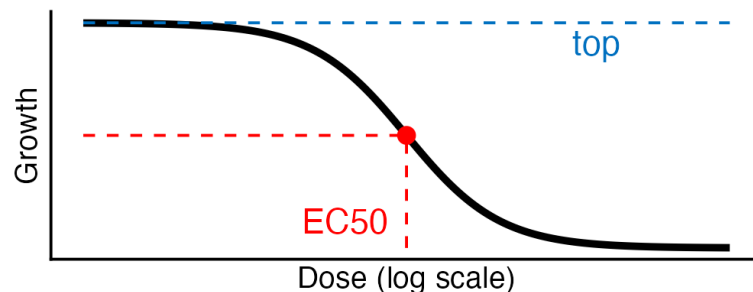
Light response: photosynthesis vs light



Logistic growth: size vs time



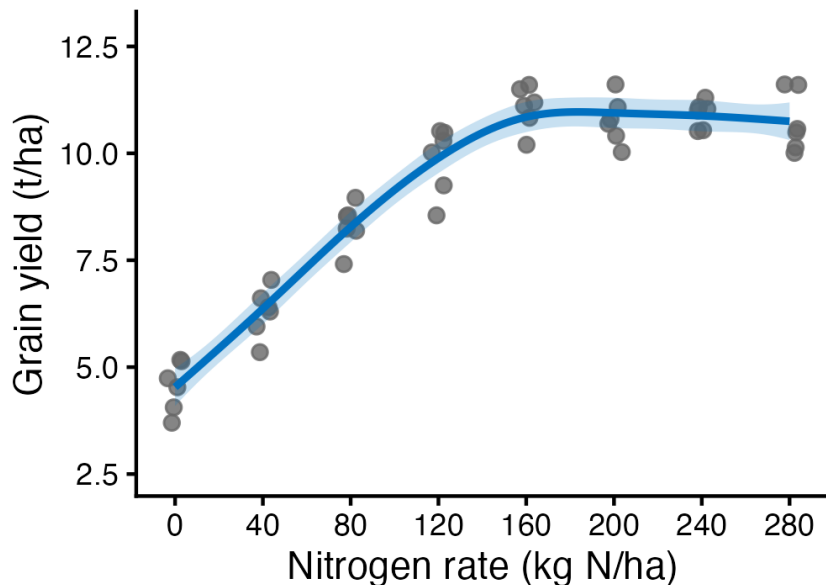
Dose-response: growth vs dose



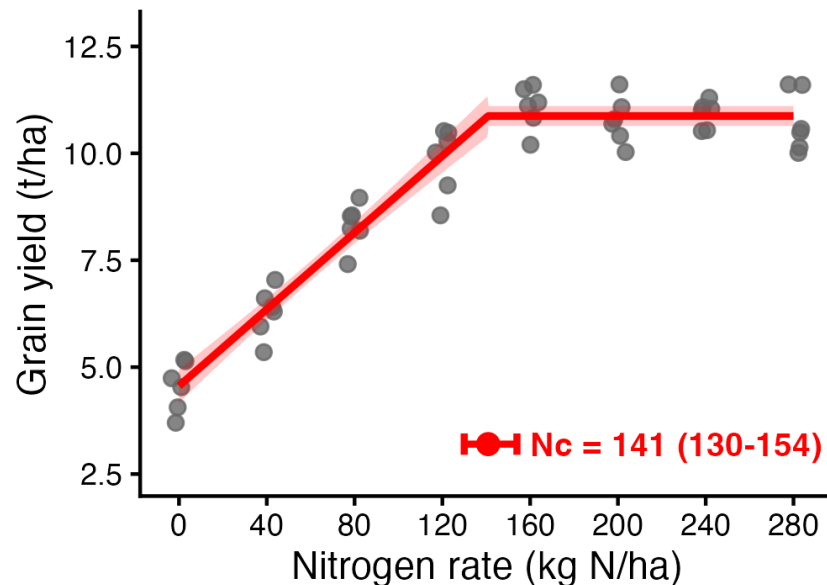
Each parameter answers a **biological question**

Describe a shape, or estimate a parameter?

`gam(yield ~ s(N_rate, k = 8))`
describes the shape



`nls(yield ~ a + b * pmin(N_rate, Nc))`
estimates N_c , with a 95% CI



GAM describes a shape. **nls()** estimates a parameter.

Live coding: by =, the field, nls



`Week4_live_2_gam.R`

Part 2: sections 4 to 6

- 4 A binomial GAM; one curve per variety
- 5 The field: model the gradient
- 6 Maize: a plateau model with `nls()`

Which tool?

Is it curved?

plot the residuals

Known shape, want its parameters

`nls()`

Unknown shape

`gam(y ~ s(x), method = "REML")`

Yes/no or counts

`family = binomial / poisson`

One curve per group

`g + s(x, by = g)`

A cycle

`bs = "cc"`

Space, or two predictors

`s(x, y) / te(x, z)`

Is k big enough?

`gam.check(); double k`