Biostatistics I: Linear Regression

Non-linear Effects

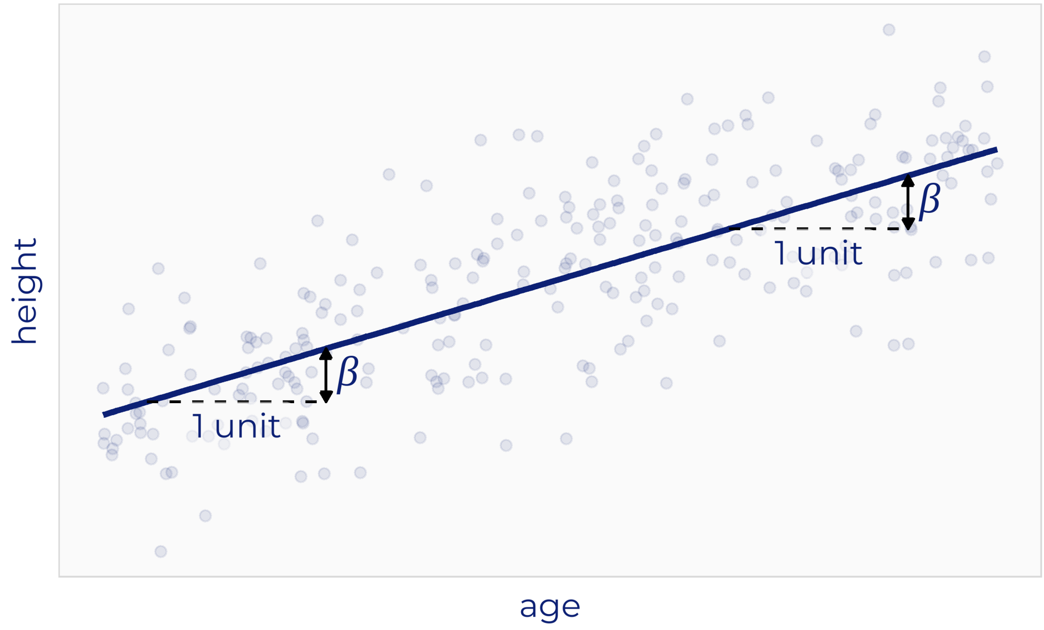
Nicole S. Erler
Department of Biostatistics, Erasmus Medical Center
n.erler@erasmusmc.nl
@N_Erler

Erasmus MC
University Medical Center Rotterdam

Linear Regression

Requirement for linear models:
The model is **linear in the regression coefficients** and the error term.

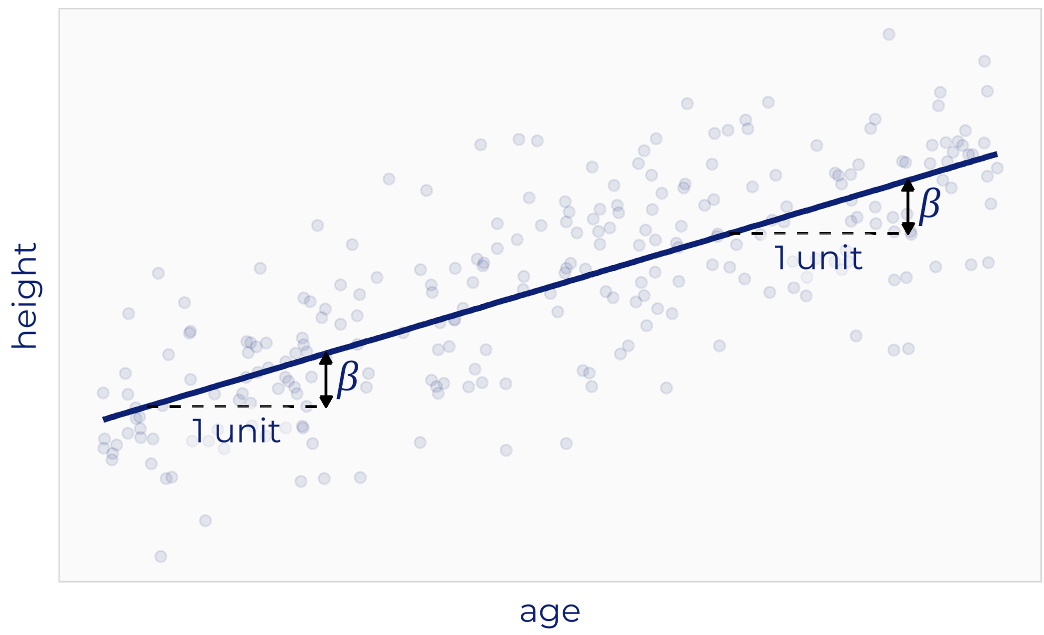
linear model



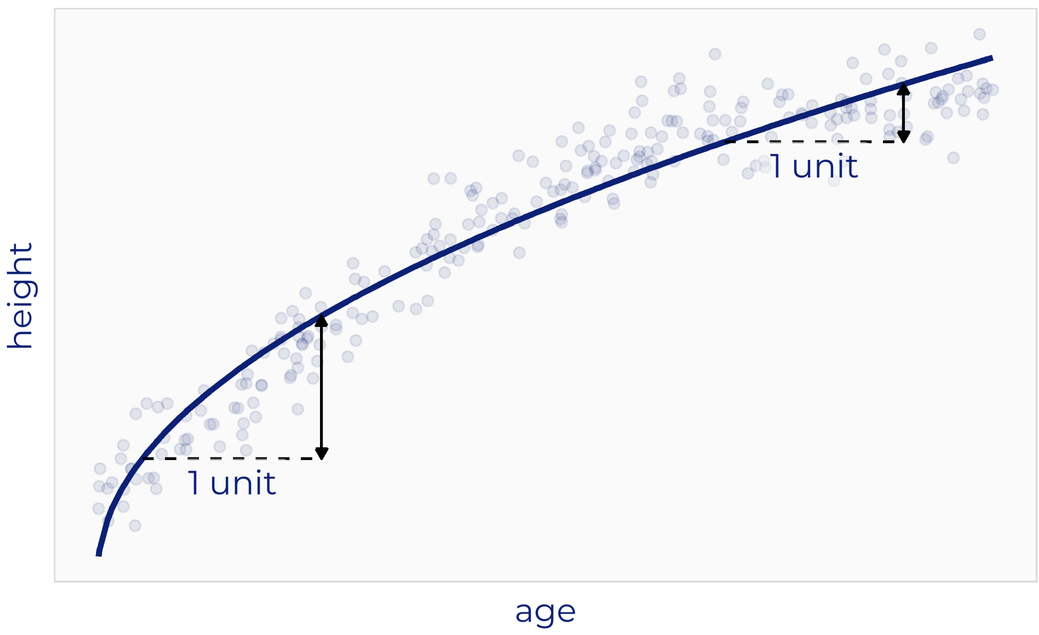
Linear Regression

Requirement for linear models:
The model is **linear in the regression coefficients** and the error term.

linear model

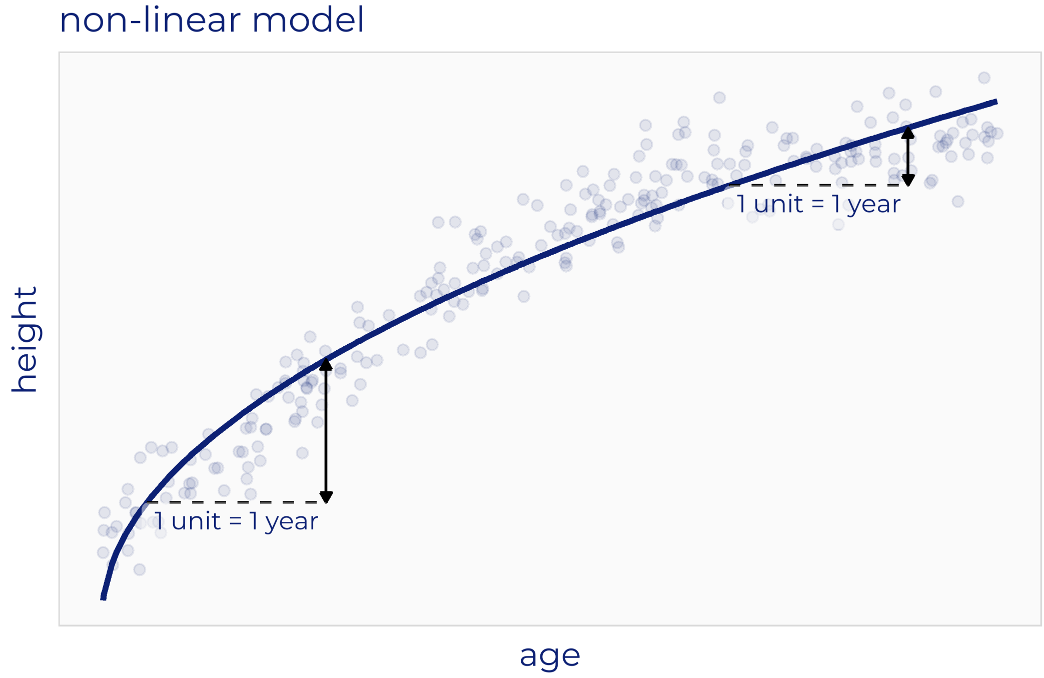


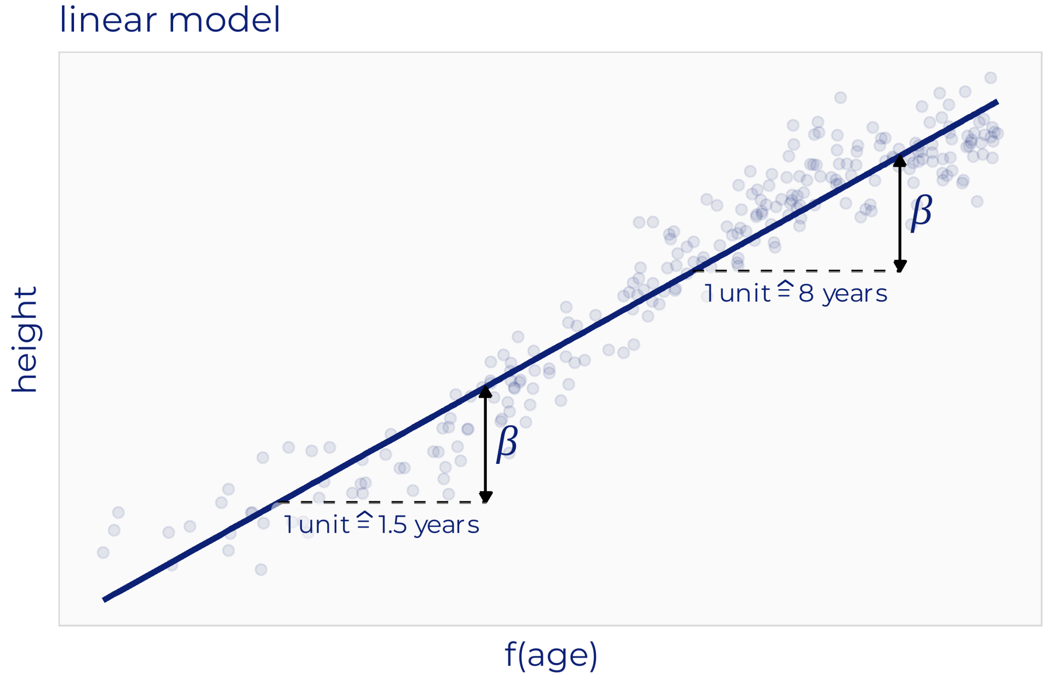
non-linear model



Linearity and Nonlinear Effects

$$\text{height}_i = \beta_0 + f(\beta_1)\text{age}_i + \varepsilon_i \qquad\qquad \text{height}_i = \beta_0 + \beta_1 f(\text{age}_i) + \varepsilon_i$$





As long as we can write the model as $y_i = f(\mathbf{x}_i)^\top \boldsymbol{\beta} + \varepsilon_i$ we have a linear model.

Modelling Non-linear Associations

Non-linear associations between response and continuous covariates can be modelled by **transforming the covariate**, i.e.,

$$y_i = f(\mathbf{x}_i)^\top \boldsymbol{\beta} + \varepsilon_i$$

But:

- The regression coefficient corresponds to a **1 unit change in the transformed covariate**, not to a 1 unit change in the original covariate.
- We cannot represent the effect of the covariate on its original scale by a single number.

Modelling Non-linear Associations

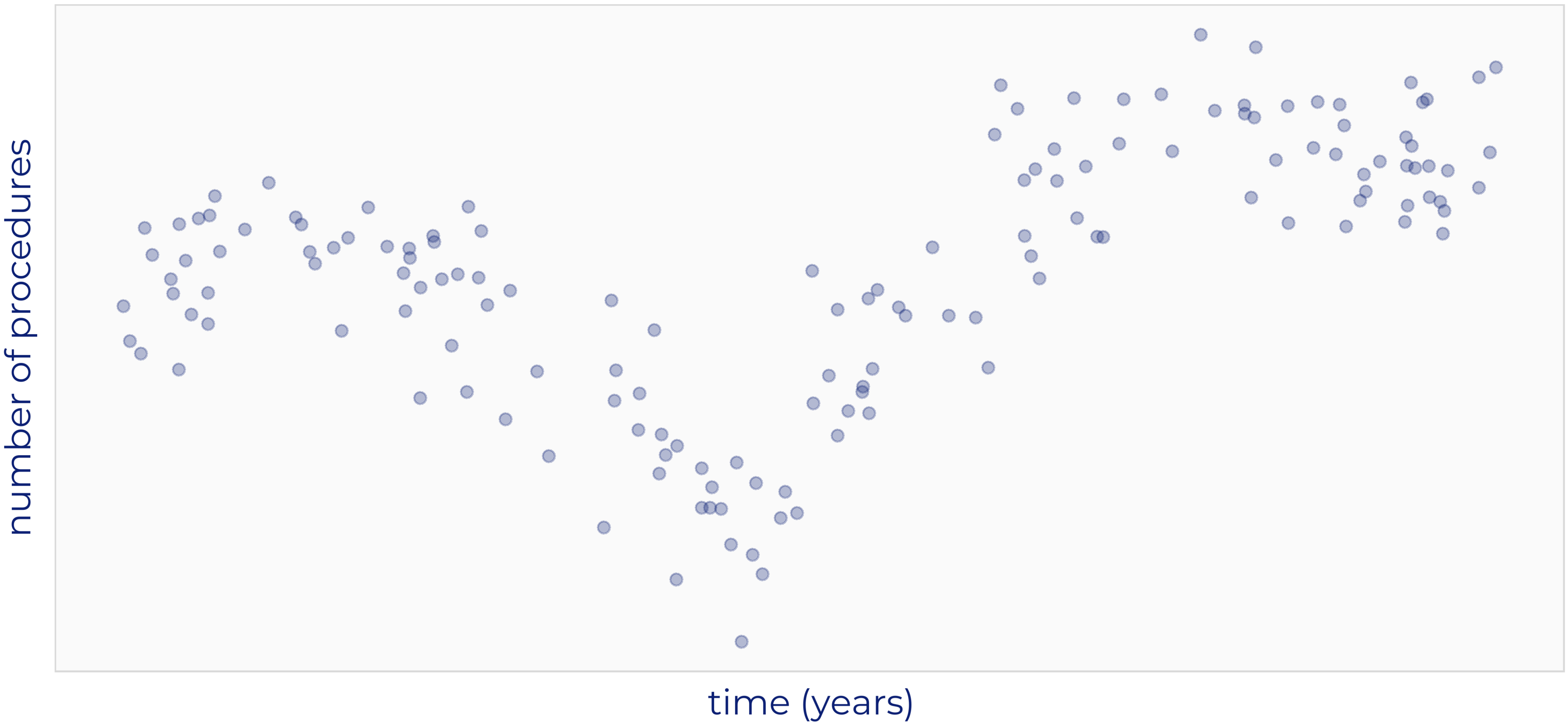
A **transformation of the response** also results in a non-linear association between the original response and the covariates, i.e.,

$$f(y_i) = \mathbf{x}_i^\top \boldsymbol{\beta} + \varepsilon_i$$

But:

- The regression coefficients represent a **change in $f(\mathbf{y})$**, not in $\mathbf{y}$.
- Only some transformations result in a direct interpretation with regards to a change in $\mathbf{y}$ (e.g., the $\log$).
- This **affects** the interpretation of **all covariates** in the model.

Complex Non-linear Forms



Complex Non-linear Forms

Transformations of covariates may result in **multiple terms**.

For example,

$$y_i = f(\mathbf{x}_i)^\top \boldsymbol{\beta} + \varepsilon_i,$$

could use a function

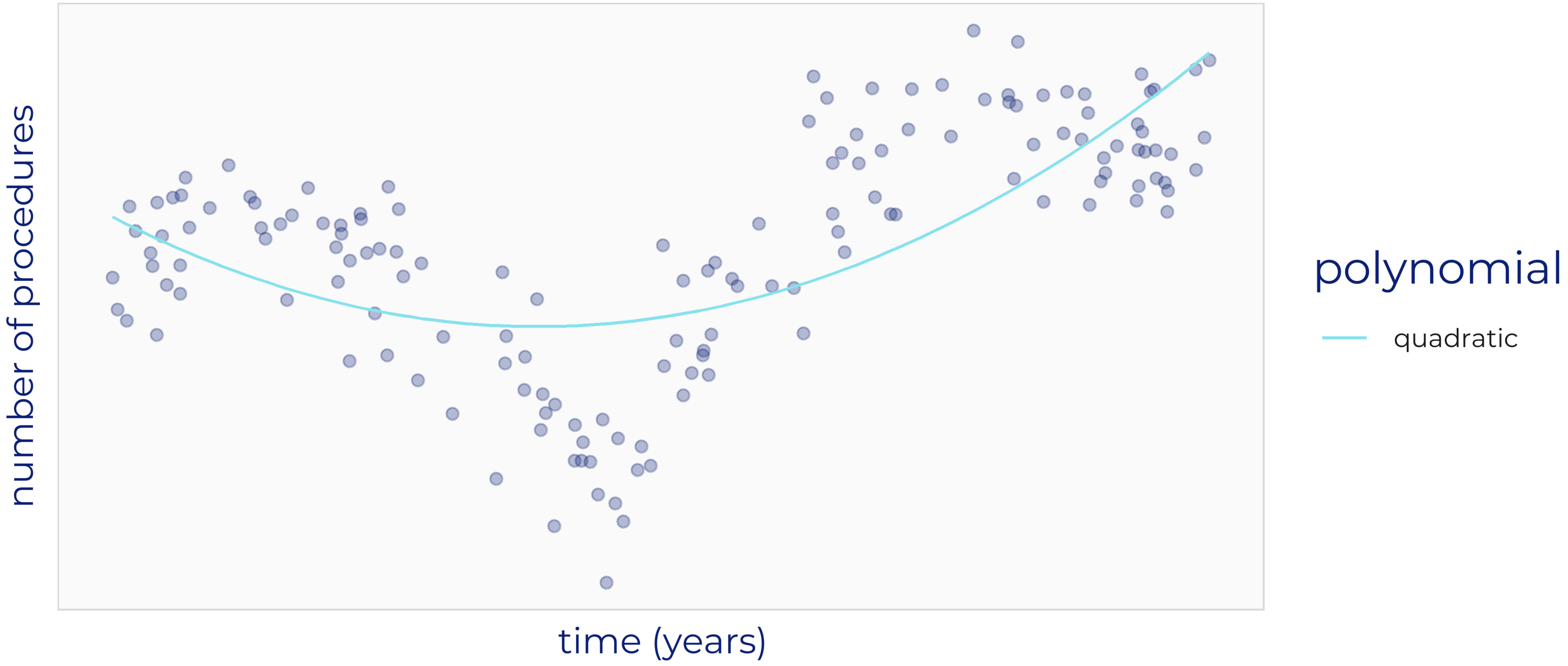
$$f(x_i) = \beta_1 x_i + \beta_2 x_i^2$$

or even

$$f(x_i) = \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 x_i^4 + \beta_5 x_i^5 + \ldots$$

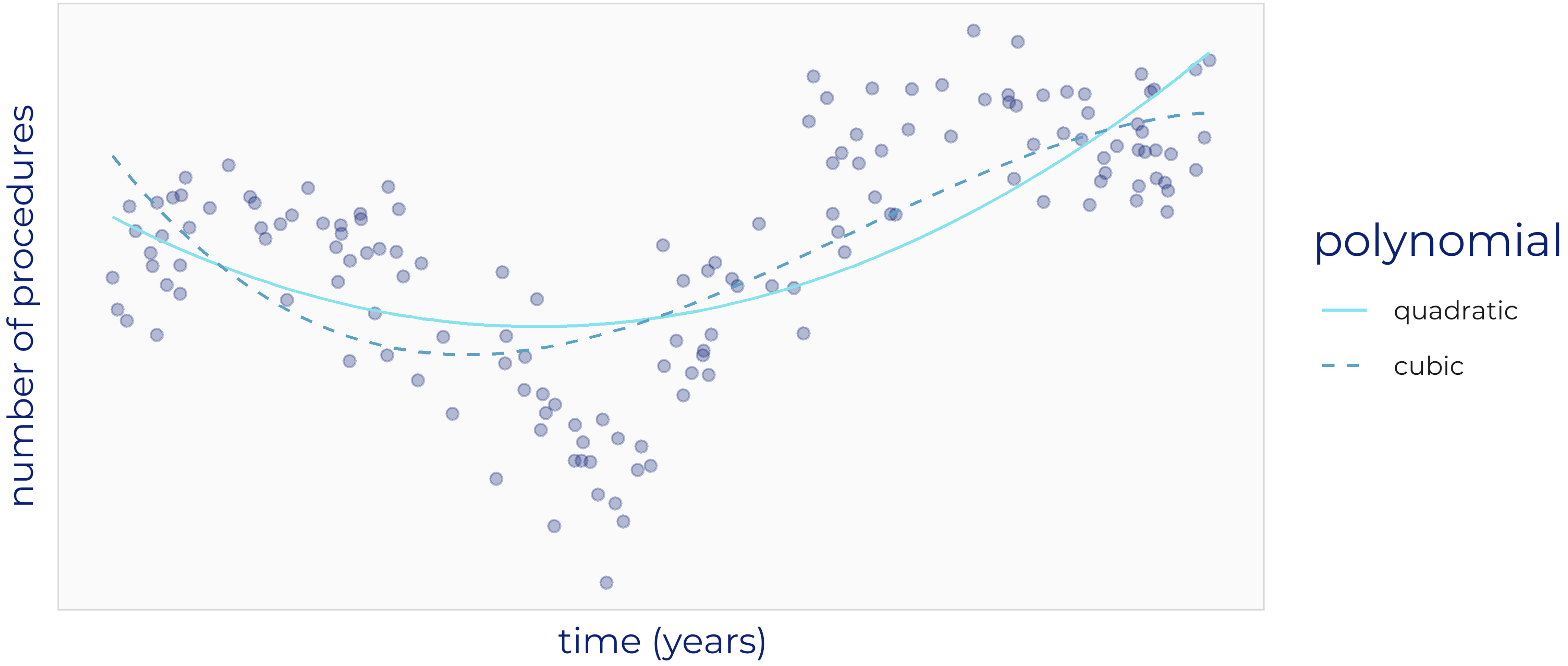
Complex Non-linear Forms: Polynomials

Polynomials are very flexible:



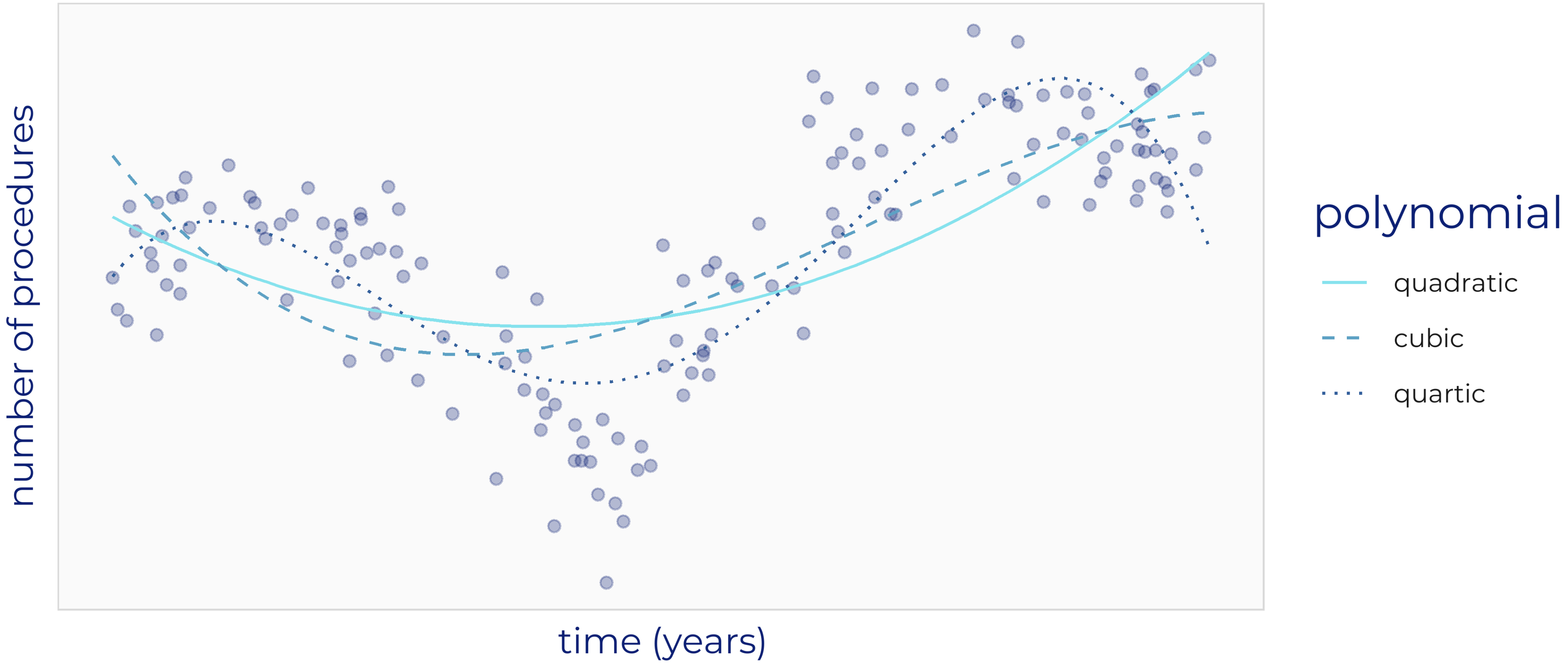
Complex Non-linear Forms: Polynomials

Polynomials are very flexible:



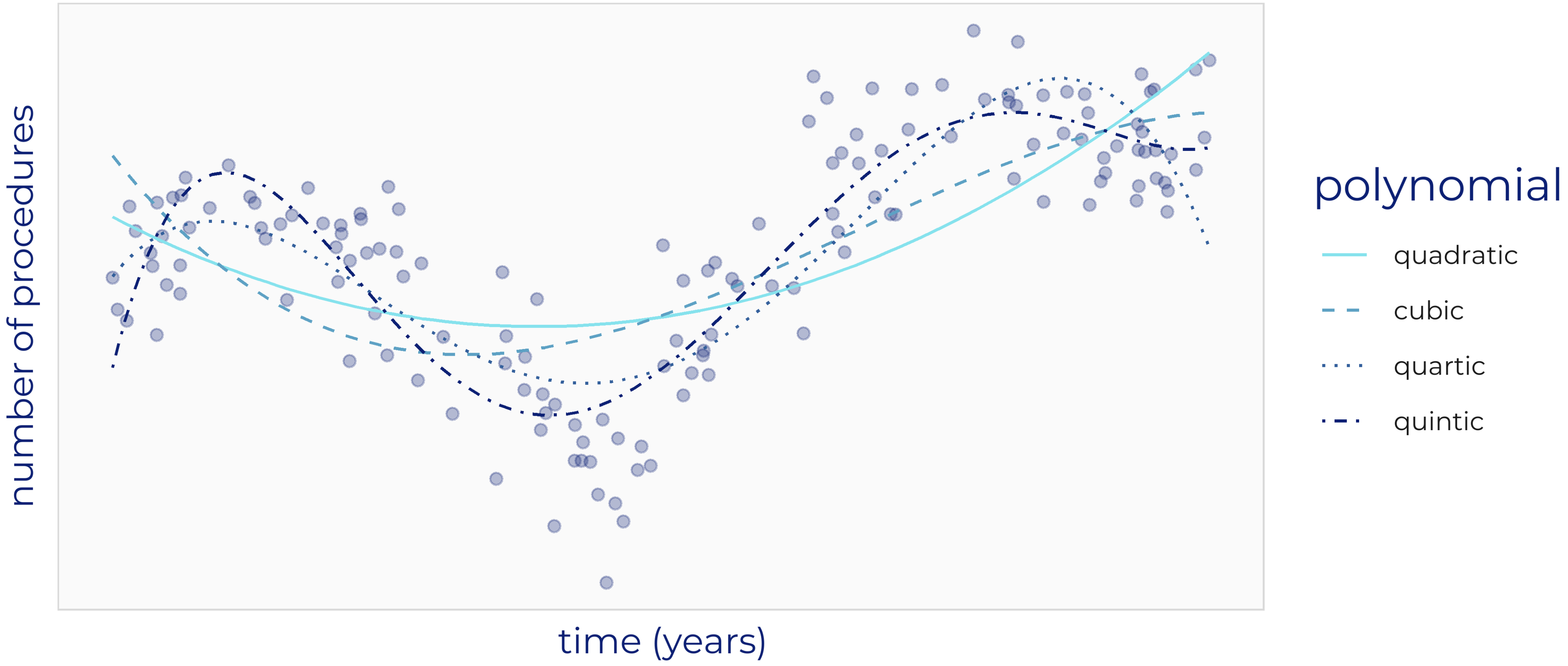
Complex Non-linear Forms: Polynomials

Polynomials are very flexible:



Complex Non-linear Forms: Polynomials

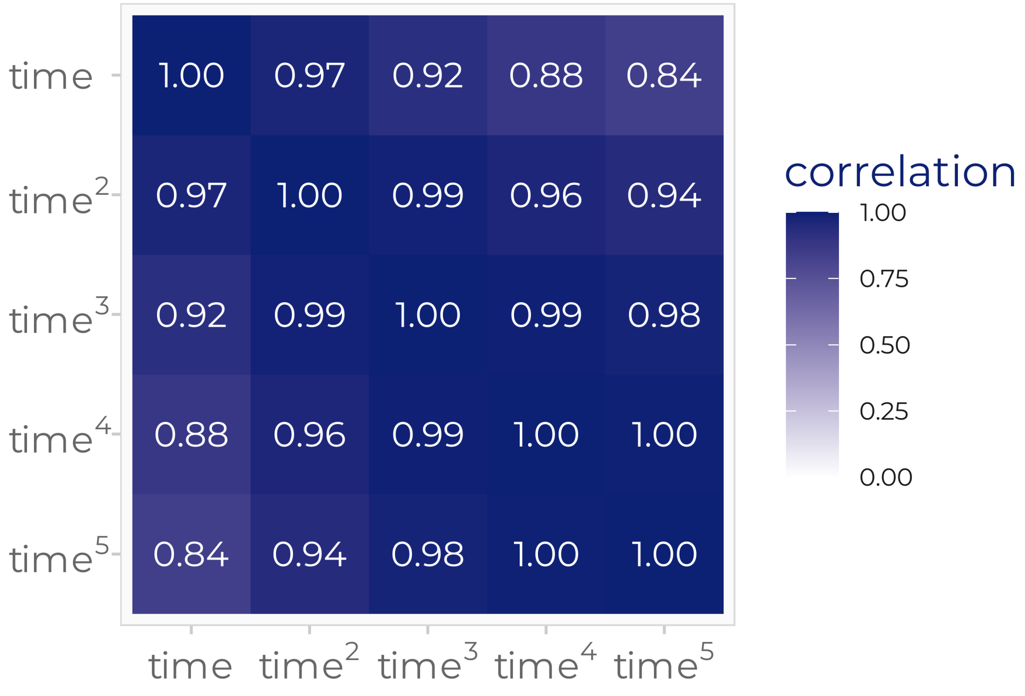
Polynomials are very flexible:



Complex Non-linear Forms: Polynomials

Problem:
Polynomial terms of the same variable are often highly correlated.

⇨ **Multicollinearity!**



Variance Inflation Factor:

	model			
term	**quadratic**	**cubic**	**quartic**	**quintic**
time	16	112	470	1656
$time^2$	16	690	8692	69807
$time^3$		278	18441	412699
$time^4$			4127	447096
$time^5$				62893

Orthogonal Polynomials

In R:
Instead of

```
lm(nr_proc ~ time + I(time^2) + I(time^3) + I(time^4), data = example_data)
```

we can use

```
lm(nr_proc ~ poly(time, degree = 4), data = example_data)
```

to fit **orthogonal polynomials**.

Orthogonal Polynomials

In R:
Instead of

```
lm(nr_proc ~ time + I(time^2) + I(time^3) + I(time^4), data = example_data)
```

we can use

```
lm(nr_proc ~ poly(time, degree = 4), data = example_data)
```

to fit **orthogonal polynomials**.

Variance Inflation Factor:

	VIF
poly(time, degree = 4)1	1
poly(time, degree = 4)2	1
poly(time, degree = 4)3	1
poly(time, degree = 4)4	1

Orthogonal Polynomials

Remember:
Orthogonal polynomials do not have the same values as standard polynomials (but contain the same information).
⇨ The design matrices differ.

Orthogonal:

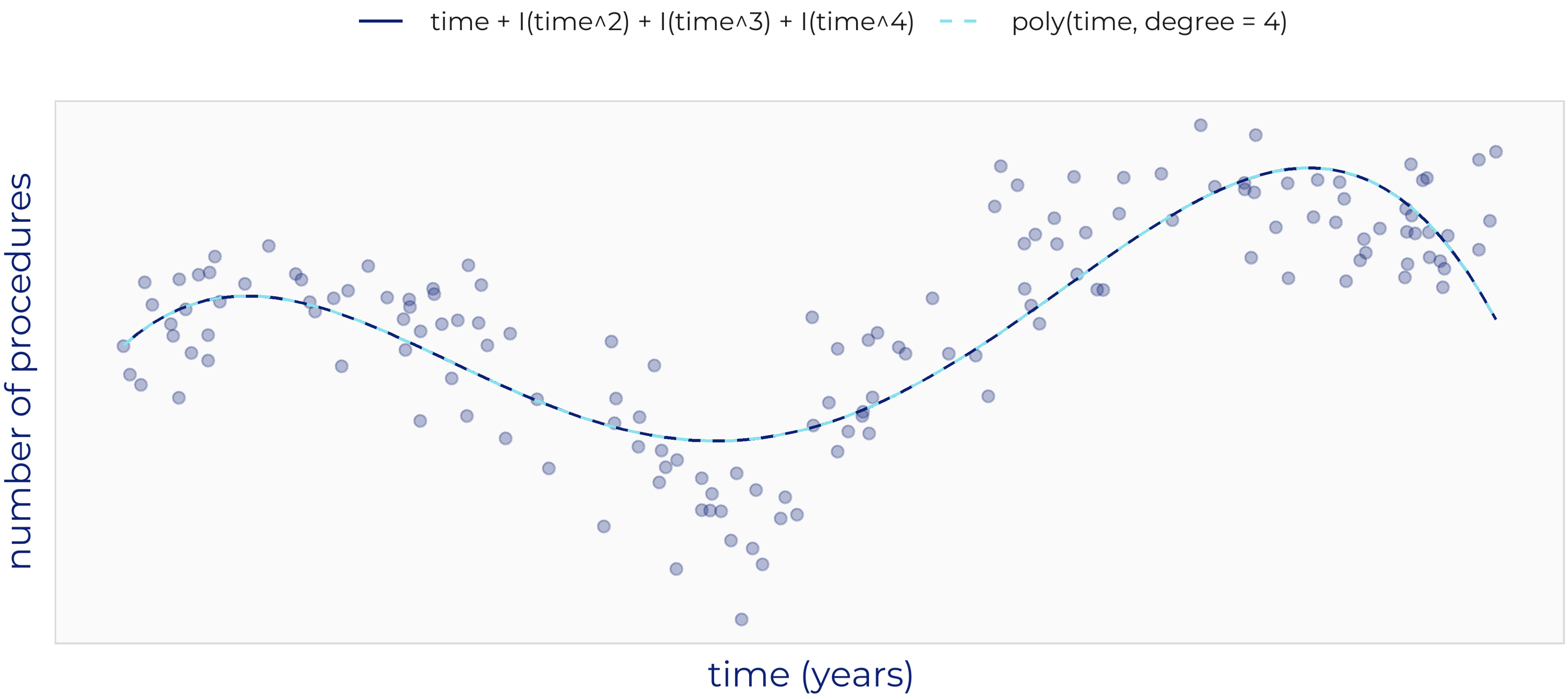
	β
(Intercept)	4.31
poly(time, degree = 4)1	6.97
poly(time, degree = 4)2	7.65
poly(time, degree = 4)3	-3.50
poly(time, degree = 4)4	-6.39

Standard:

	β
(Intercept)	3.82
time	0.87
I(time^2)	-0.29
I(time^3)	0.03
I(time^4)	0.00

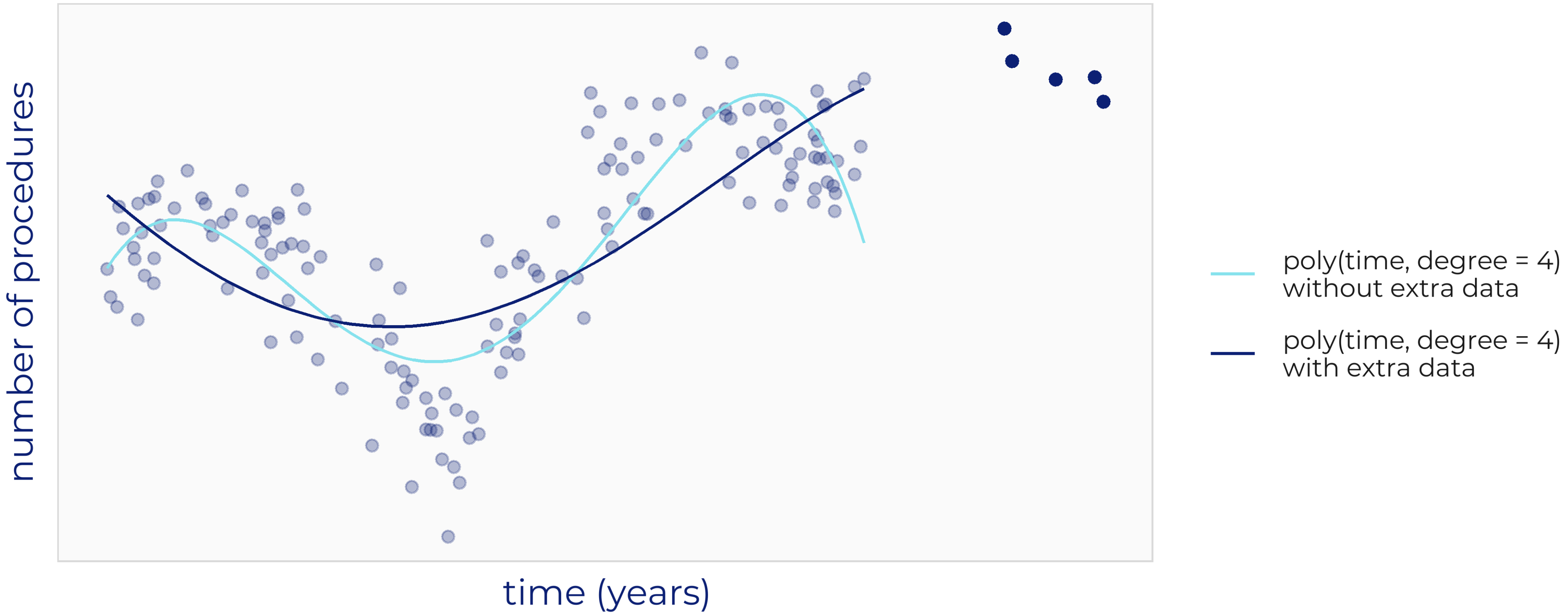
⇨ The regression coefficients are not identical, but the fitted values are.

Orthogonal Polynomials

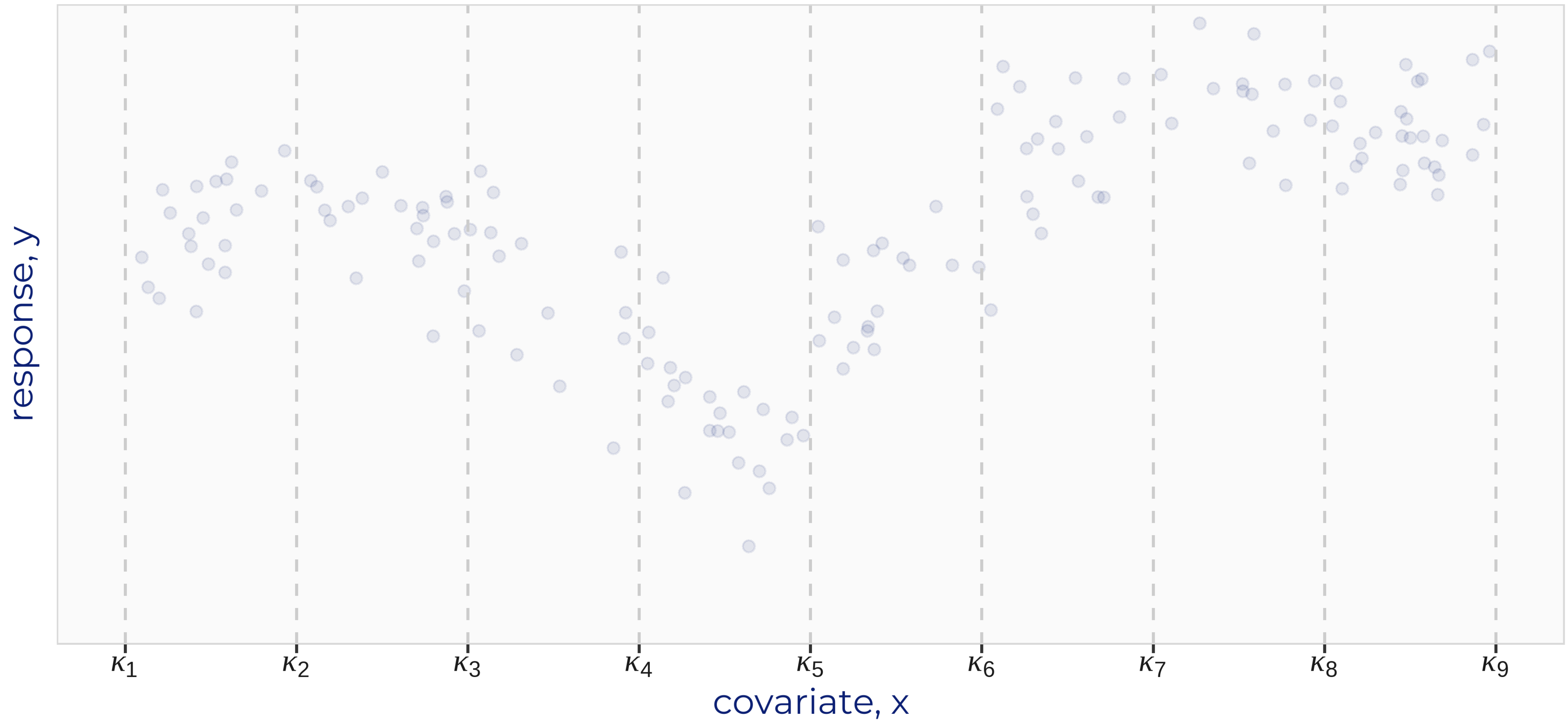


Polynomials

Drawback: Polynomials are defined over the whole range of the covariate.
⇨ Local changes have global impact.

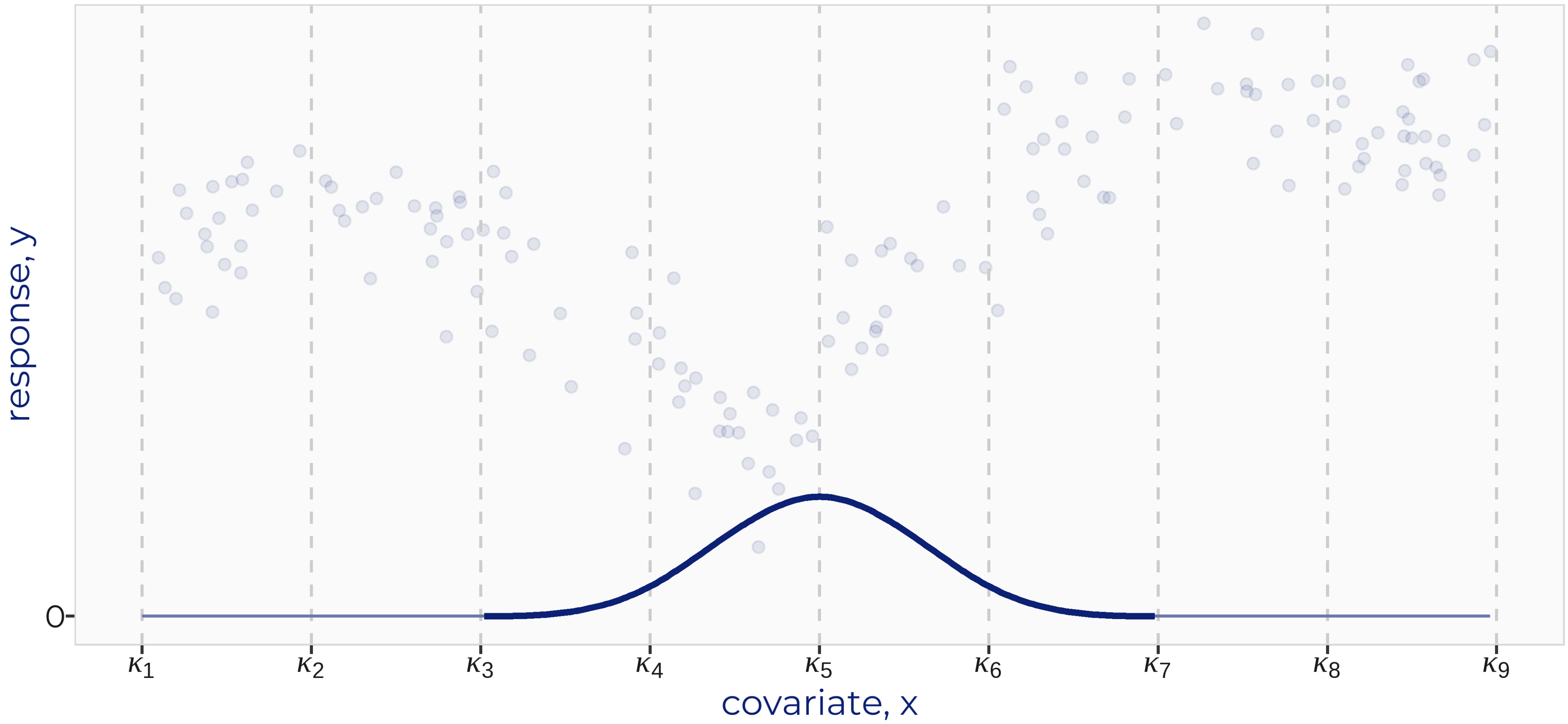


Splines



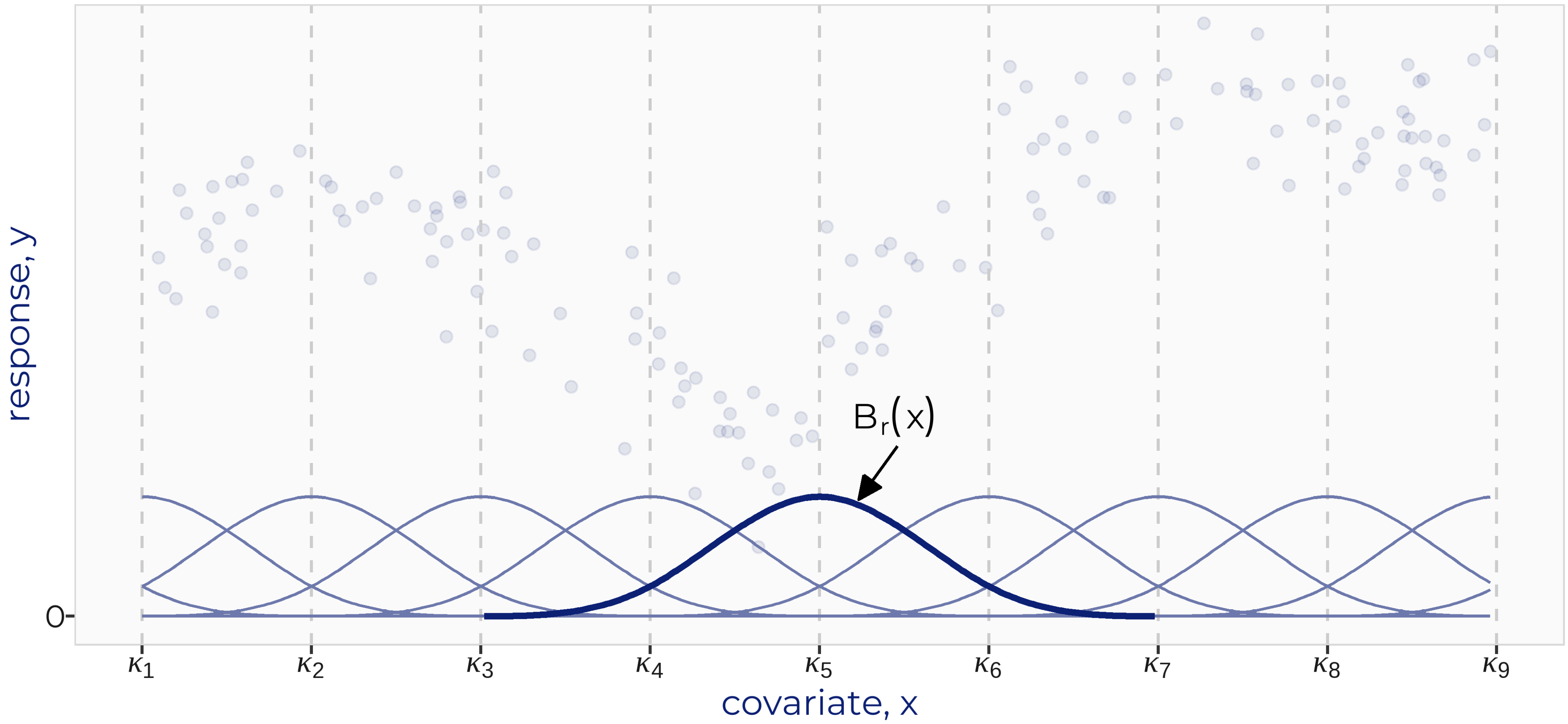
response, y
κ_1 κ_2 κ_3 κ_4 κ_5 κ_6 κ_7 κ_8 κ_9
covariate, x

B-Splines



response, y
0
κ_1
κ_2
κ_3
κ_4
κ_5
κ_6
κ_7
κ_8
κ_9
covariate, x

B-Splines



response, y
0
$B_r(x)$
κ_1
κ_2
κ_3
κ_4
κ_5
κ_6
κ_7
κ_8
κ_9
covariate, x

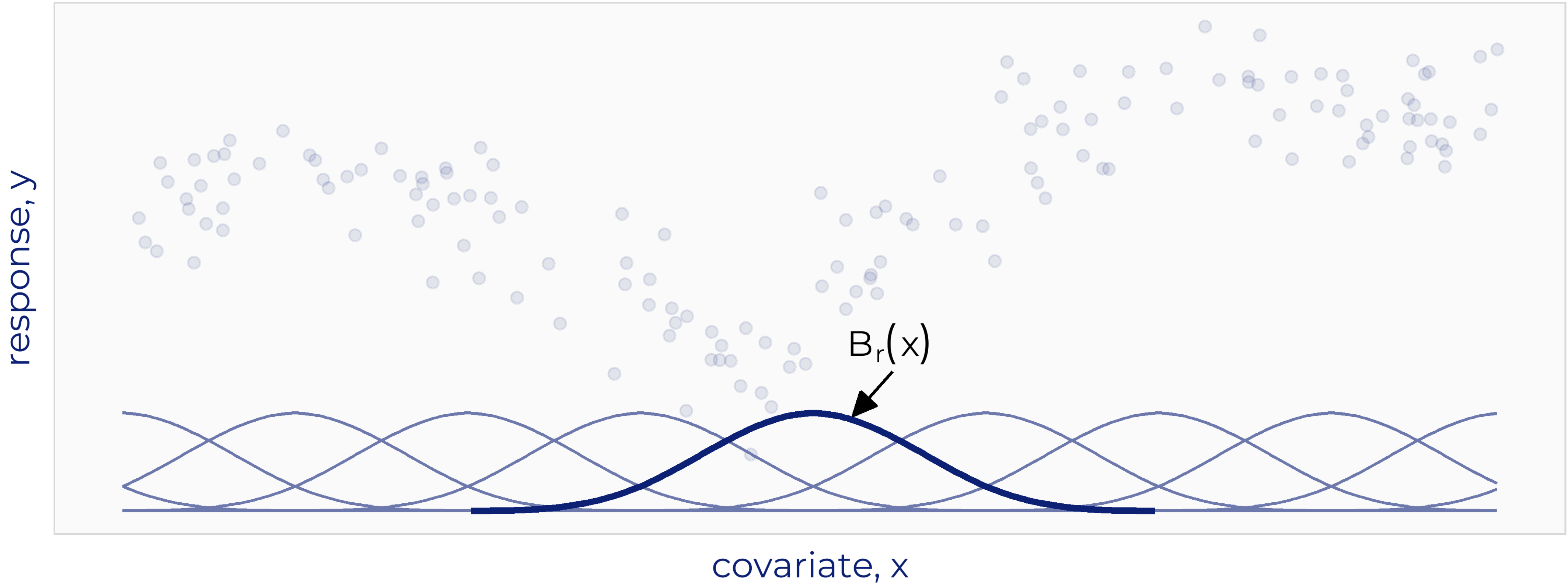
B-Splines

Splines fit into the framework of linear models:

$$
\begin{aligned}
y_i &= \beta_0 + f(x_i)^\top \boldsymbol{\beta} + \varepsilon_i \\
&= \beta_0 + \underbrace{\beta_1 B_1(x_i) + \beta_2 B_2(x_i) + \beta_3 B_3(x_i) + \ldots}_{f(x_i)^\top \boldsymbol{\beta}} + \varepsilon_i \\
&= \beta_0 + \underbrace{\sum_{r=1}^{d} \beta_r B_r(x_i)}_{f(x_i)^\top \boldsymbol{\beta}} + \varepsilon_i
\end{aligned}
$$

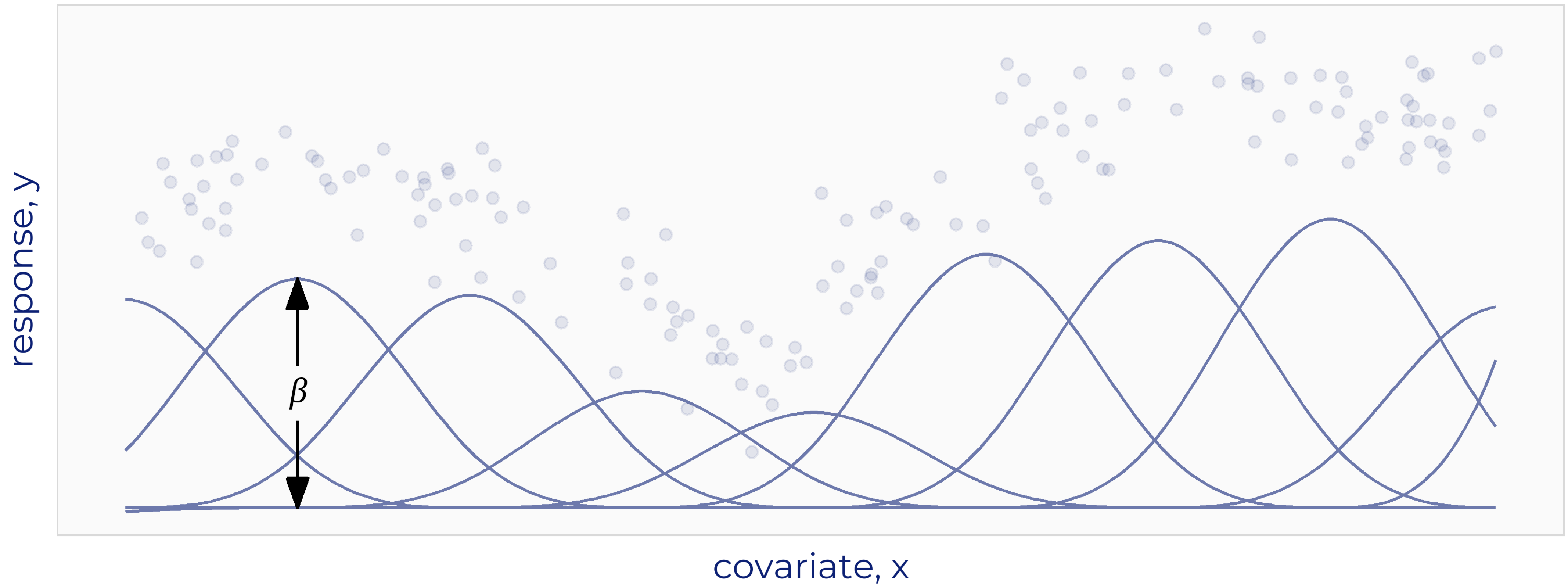
B-Splines

$$y_i = \beta_0 + \sum_{r=1}^{d} \beta_r \, B_r(x_i) + \varepsilon_i$$



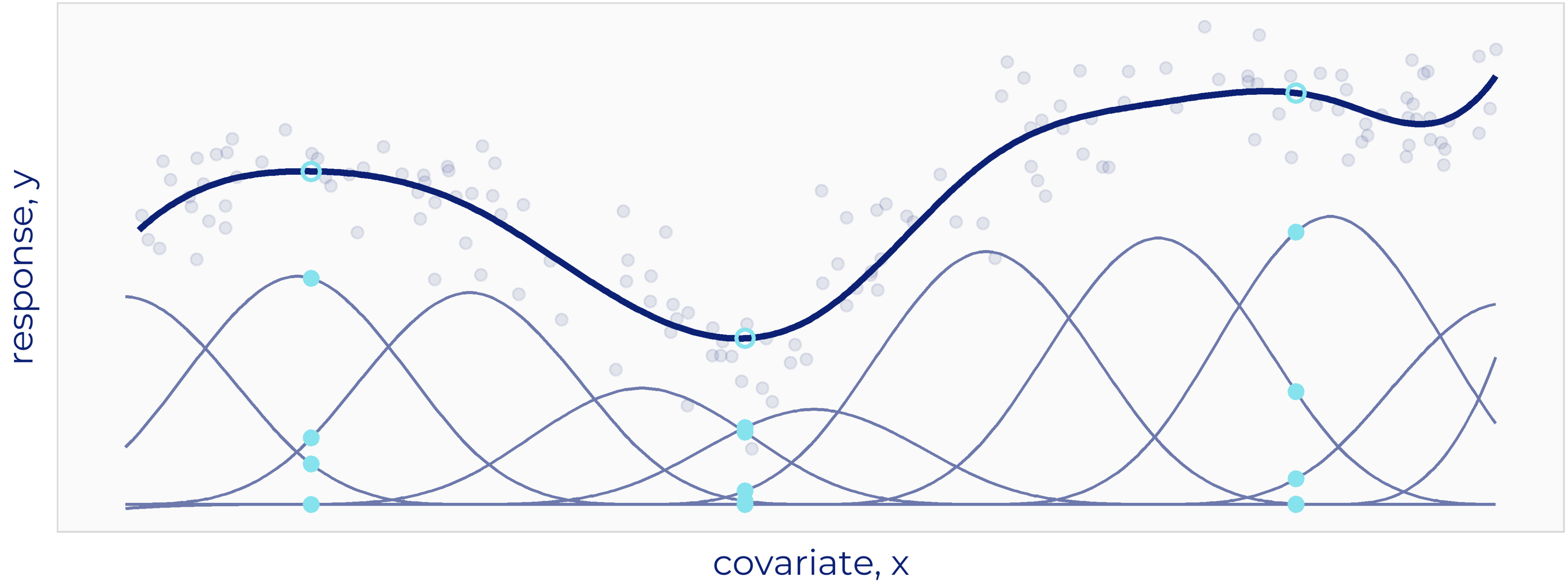
B-Splines

$$y_i = \beta_0 + \sum_{r=1}^{d} \beta_r B_r(x_i) + \varepsilon_i$$



B-Splines

$$y_i = \beta_0 + \sum_{r=1}^{d} \beta_r B_r(x_i) + \varepsilon_i$$



B-Splines

A **B-Spline** is a linear combination of a set of **basis functions**.

These **basis functions** are defined so that they are

- **polynomial functions** inside a given interval, and
- zero outside that interval,
- and connected so that form a (smooth) line.

The intervals are defined by a set of **knots**.

The polynomial function have a certain **degree** (i.e., constant, linear, quadratic, . . .)

B-Splines in R

The package **splines** provides the functions

- `bs()`: B-splines
- `ns()`: natural cubic (B-)splines

B-Splines in R

The package **splines** provides the functions

- `bs()`: B-splines
- `ns()`: natural cubic (B-)splines

Arguments

- `x`: the (name of the) covariate
- `df`: the number of degrees of freedom
- `degree`: degree of the polynomial (only for `bs()`)
- `knots`: position of the inner knots
- `Boundary.knots`: position of the boundary knots

B-Splines in R

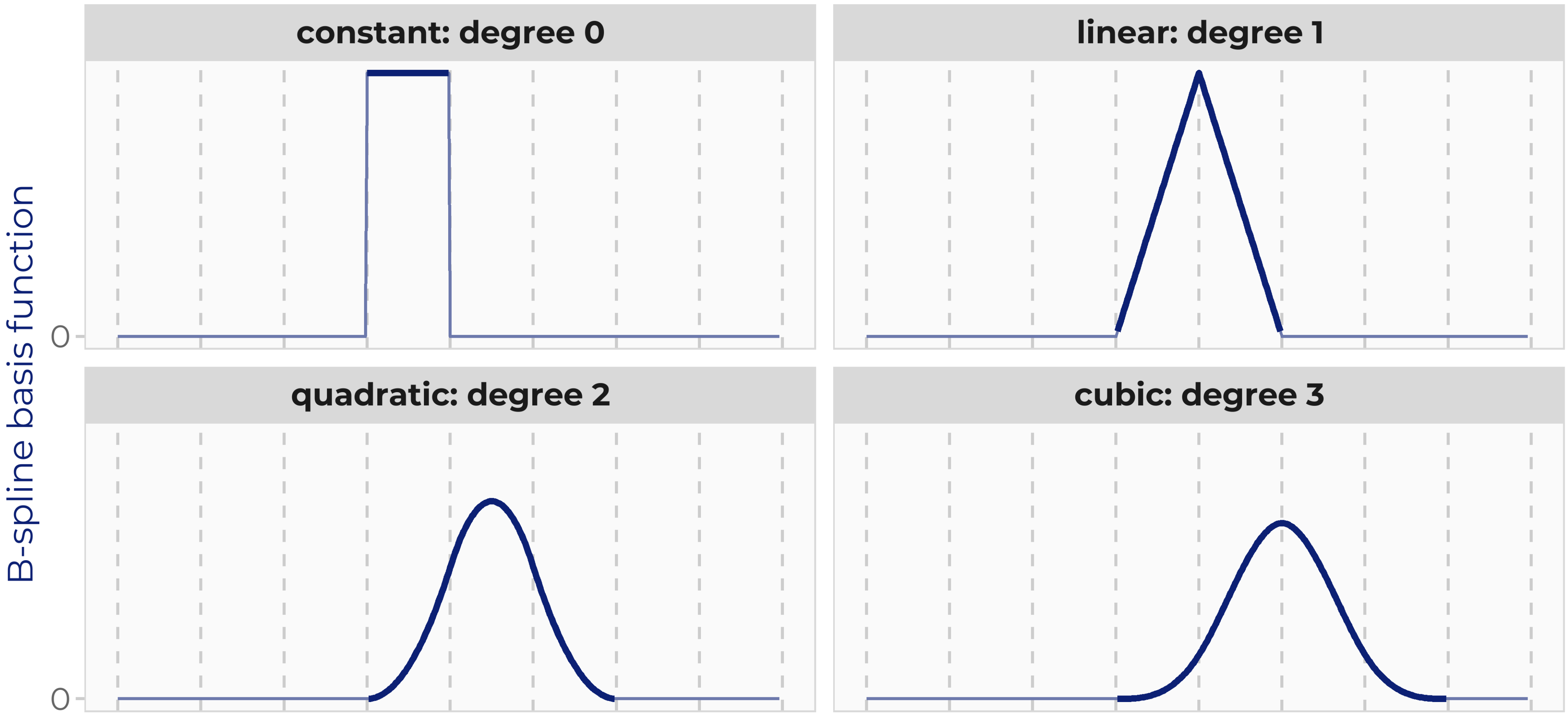
For example:

```
lm(height ~ ns(age, df = 3) + sex + kcal_sd, data = child)
```

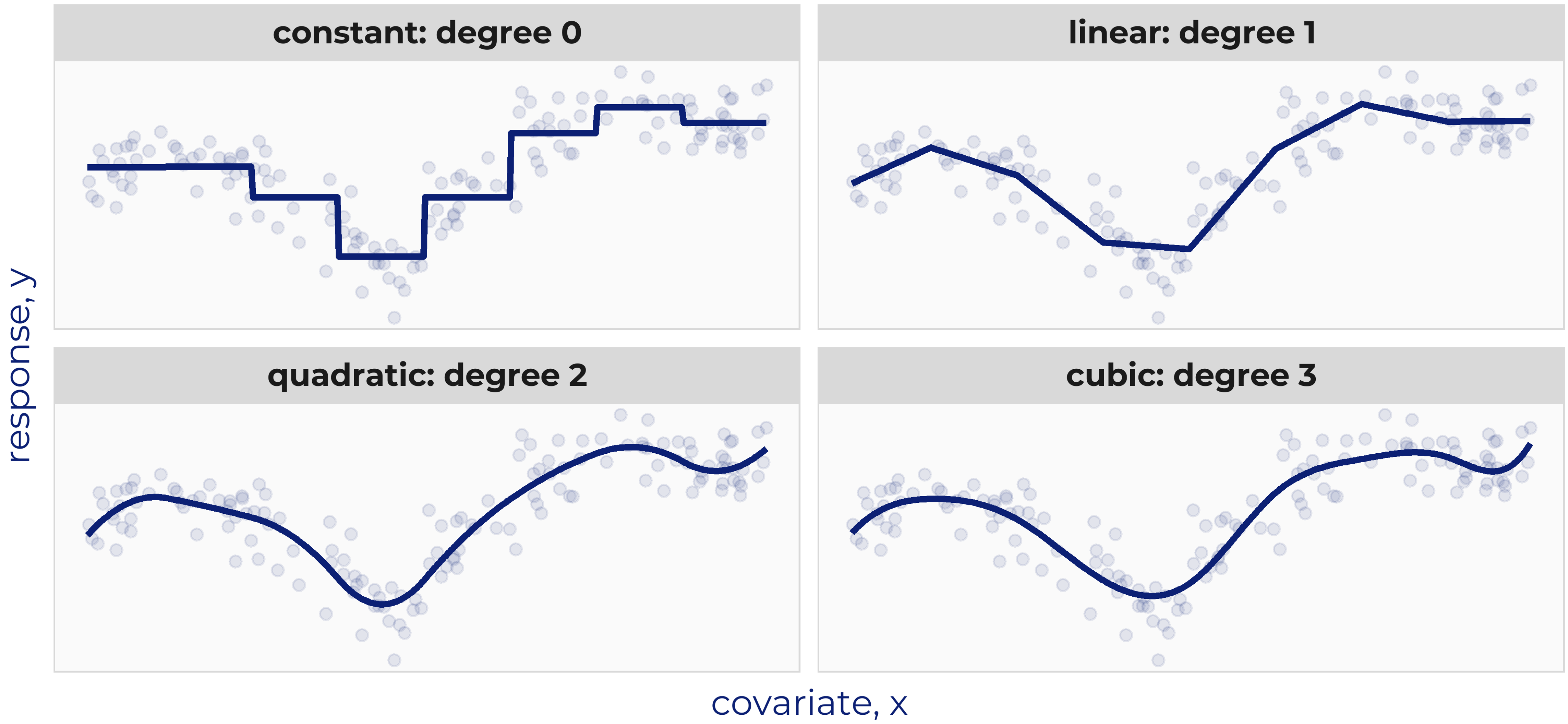
```
##
## Call:
## lm(formula = height ~ ns(age, df = 3) + sex + kcal_sd, data = child)
##
## Coefficients:
##       (Intercept)  ns(age, df = 3)1  ns(age, df = 3)2  ns(age, df = 3)3
##          0.269137          0.554308          1.037329          0.467639
##           sexgirl           kcal_sd
##          0.007267          0.000595
```

Regression coefficients associated with the spline do not have a clinically meaningful interpretation.

B-Splines: Degree



B-Splines: Degree



Splines: Knots & Degrees of Freedom

B-splines are defined based on two **boundary knots** and a set of **inner knots**.

Splines: Knots & Degrees of Freedom

B-splines are defined based on two **boundary knots** and a set of **inner knots**.

(Cubic) **B-splines** and **natural cubic splines** differ in how they are defined at/outside the boundary knots.

Splines: Knots & Degrees of Freedom

B-splines are defined based on two **boundary knots** and a set of **inner knots**.

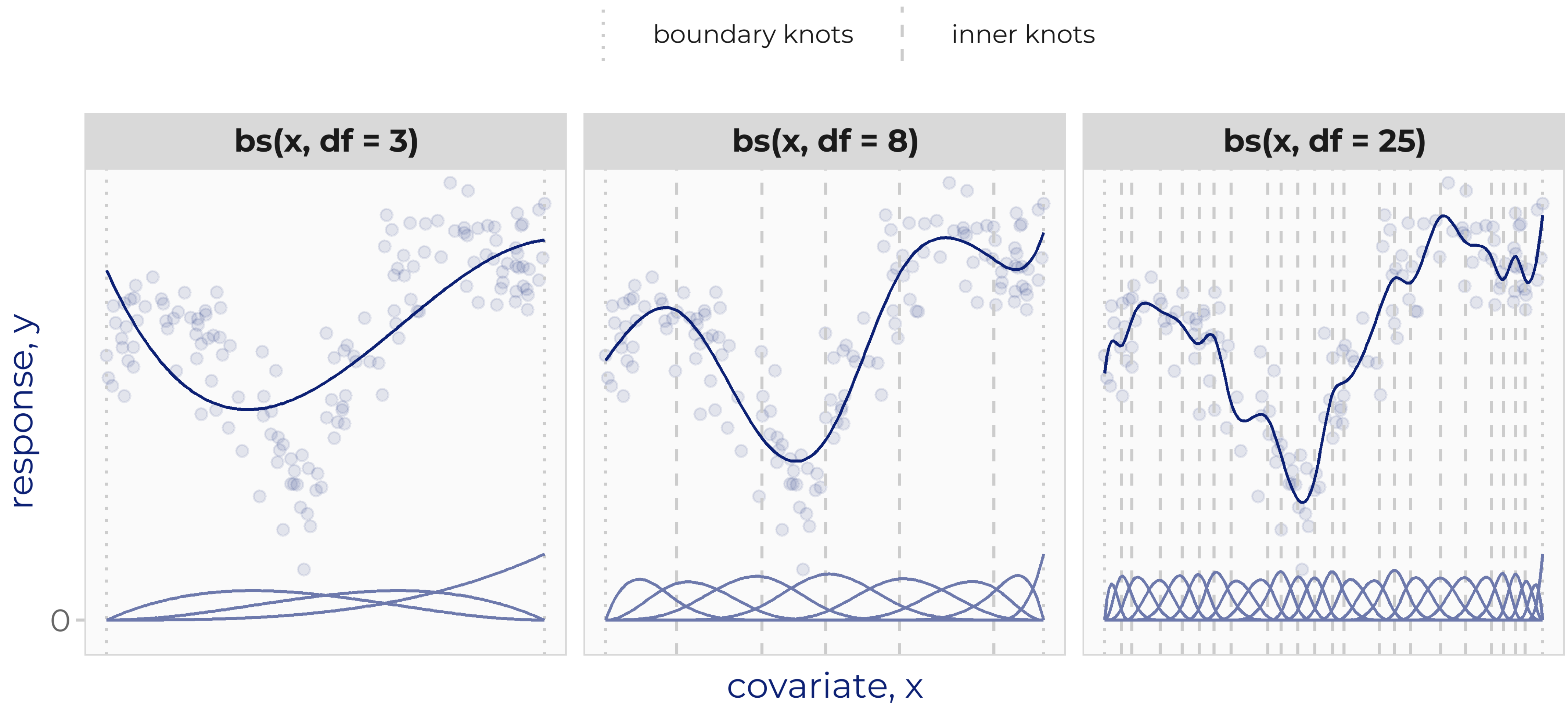
(Cubic) **B-splines** and **natural cubic splines** differ in how they are defined at/outside the boundary knots.

The **degrees of freedom** (df; number of associated regression coefficients) depend on the degree of the spline and number of inner knots:

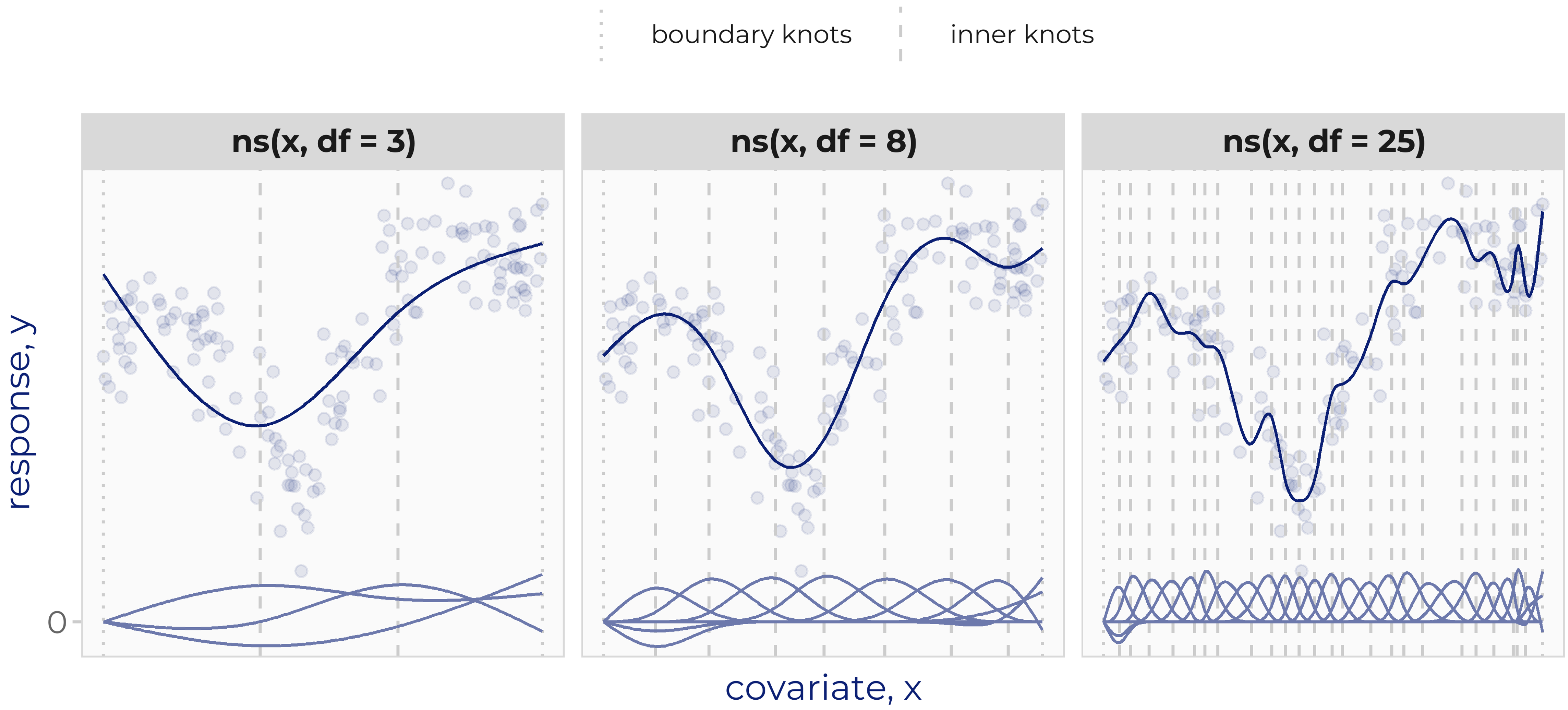
- **B-splines**: df = # inner knots + degree, (i.e., df $\geq$ degree)
- **natural cubic splines**: df = # inner knots + 1, (i.e., df $\geq$ 1)

⇨ The number of (inner) knots / degrees of freedom control the flexibility.

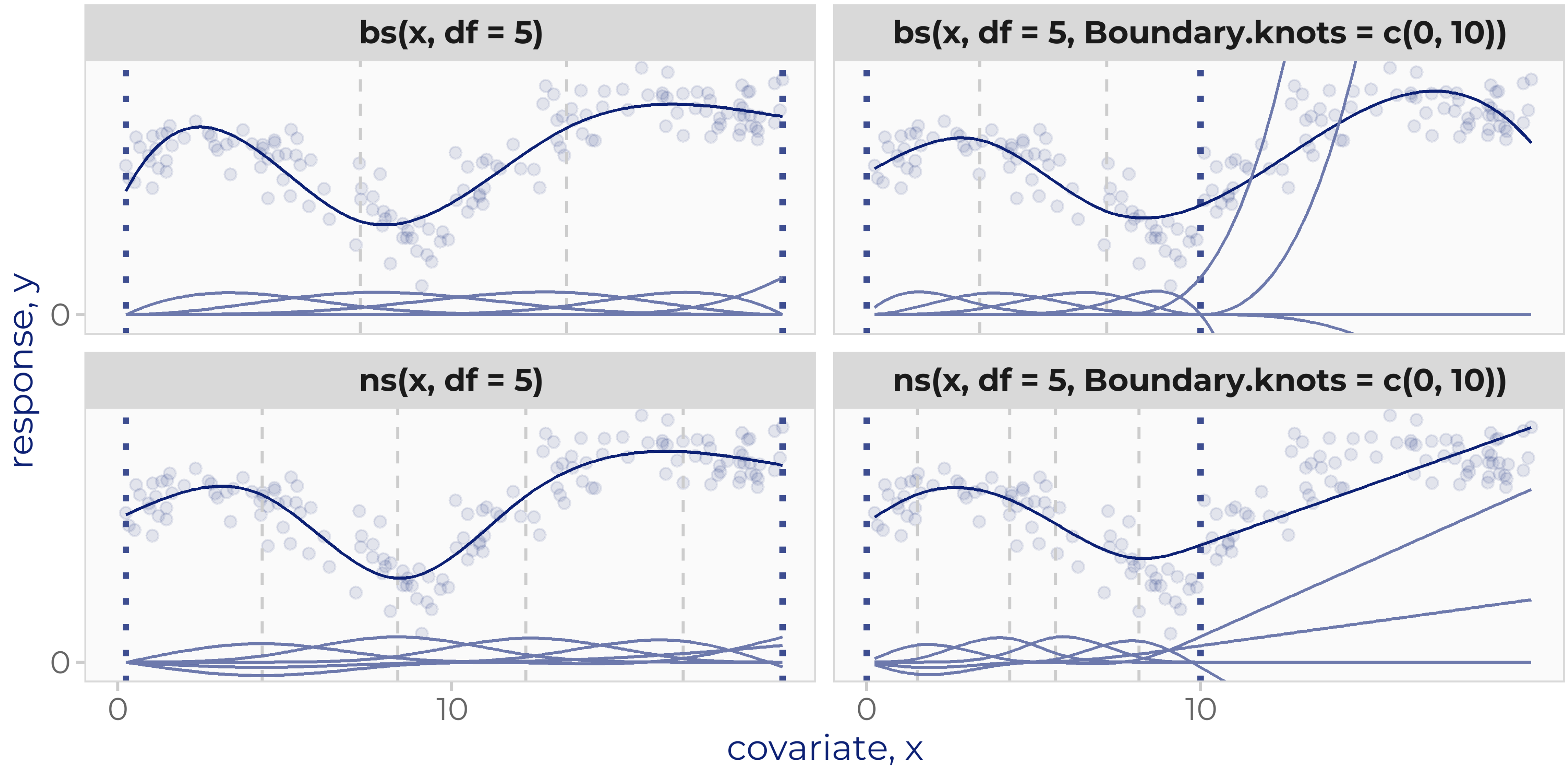
Knots/df with `bs()`



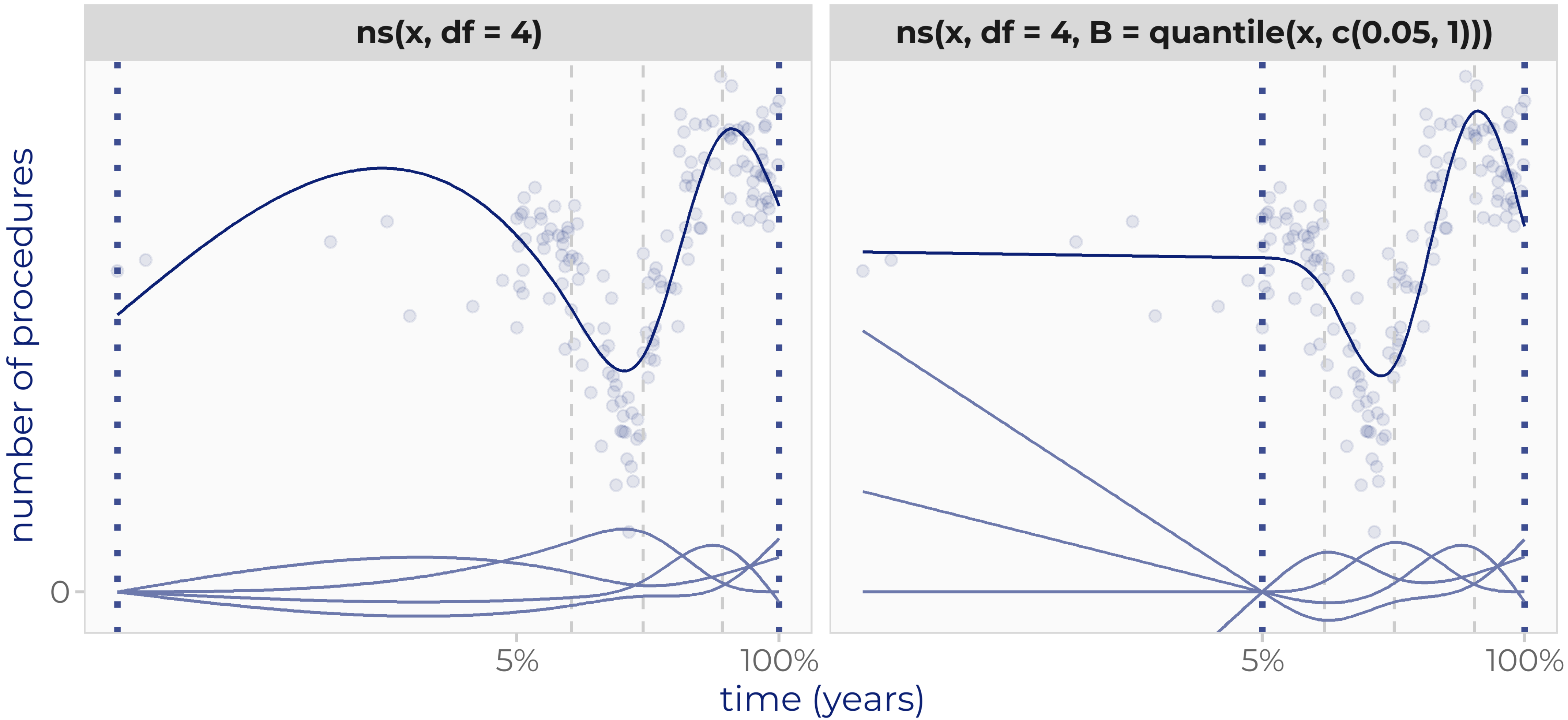
Knots/df with `ns()`



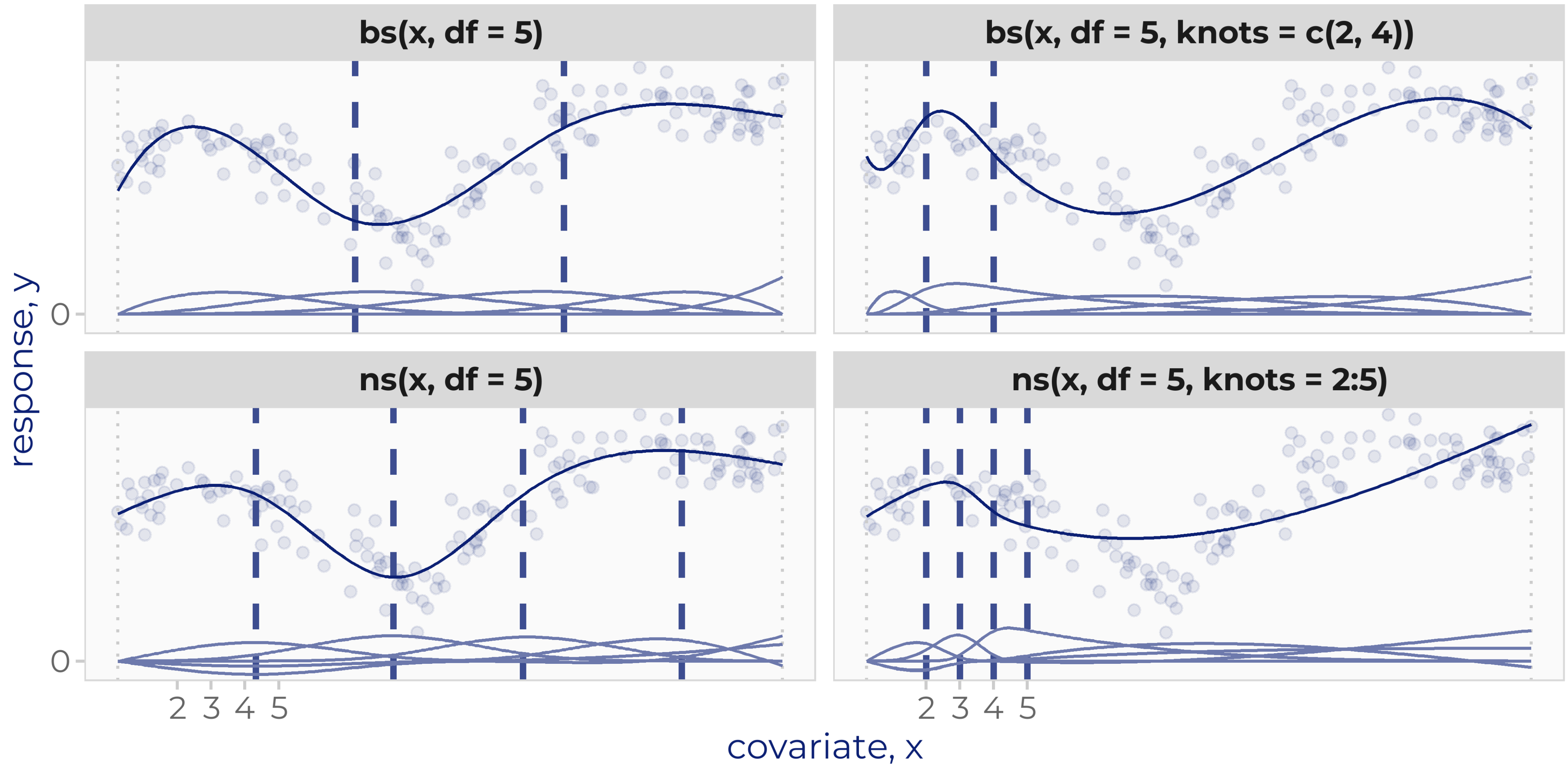
Boundary Knots



Boundary Knots for Skewed Data/Outliers



Placement of Knots



Summary

Non-linear effects can be included in the linear model in multiple ways, for example using **transformations**, **polynomials** or **(B-)splines**.

Transformations

- requires a known, simple structure
- interpretation with regards to 1 unit change in $f(x)$

Polynomials

- more flexible than (simple) transformations
- flexibility controlled by degree of polynomial
- coefficients of the separate terms need to be interpreted jointly
  ⇨ usually too complex for direct interpretation
  ⇨ effect plots

Summary

(B-)**Splines**

- more flexible than (simple) transformations
- specified locally ⇨ more stable than polynomials
- most common: natural cubic (B-)splines
- no direct interpretation of the coefficients
  ⇨ effect plots
- flexibility controlled via degrees of freedom