Quantitative Research Methods

Chapter 7: Moving Beyond Linearity

Prof. Dr. Christoph Weisser

HSBI

Summer Semester 2026

The course at a glance

	Lecture	Topic
	1	Ch. 1 + 2 — Introduction; what is statistical learning
	2	Ch. 2 — Accuracy, bias–variance, Bayes classifier, KNN
	3–4	Ch. 3 — Linear regression (estimation, inference, diagnostics)
	5–6	Ch. 4 — Classification (logistic, LDA/QDA, naive Bayes, ROC)
	7	Ch. 5 — Resampling (validation, CV, bootstrap)
	8	Ch. 6 — Model selection and regularisation (ridge, lasso)
▶	**9**	**Ch. 7 — Moving beyond linearity (splines, GAMs)**
	10	Ch. 8 — Tree-based methods (trees, forests, boosting)
	11	Ch. 10 — Deep learning (MLPs, CNNs, training)
	12	Ch. 13 — Multiple testing (FWER, FDR)

Twelve sessions of 180 minutes. ▶ marks today's chapter. Short exercises every ∼20 min; extended exercises every ∼45 min; each chapter has a companion lab notebook.

Contents

Optional and advanced material is collected in the **appendix** at the end of the deck.

Notation in this chapter

Symbol	Meaning
$b_j(x)$	Basis function: fixed transformation of x entering linearly
d	Degree of a polynomial (or of each spline piece)
$C_k(x)$	Indicator (dummy) for the k-th bin of a step function
ξ_k	Knot: breakpoint where adjacent spline pieces join
$(x-\xi)_+^3$	Truncated power term: 0 for $x \le \xi$, else $(x-\xi)^3$
K	Number of knots; cubic-spline df $= K+4$, natural-spline df $= K$
λ	Smoothing parameter of the penalised (smoothing-spline) fit
$\int g''(t)^2\,dt$	Roughness penalty: total curvature of the fit g
df_λ	Effective degrees of freedom, trace(S_λ), where $\hat{g} = S_\lambda y$
s	Span: fraction of nearest neighbours used by LOESS
$f_j(X_j)$	Smooth component function of predictor X_j in a GAM

Sources and references

Primary textbook

James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). *An Introduction to Statistical Learning, with Applications in Python.* Springer.

About these slides

Each method is introduced with motivation, intuition, formal definition, and worked examples. Slide titles are descriptive.

Learning objectives for this chapter

By the end of this chapter you will be able to:

- **Fit** polynomial and step-function regressions and discuss their respective failure modes.
- **Construct** regression splines, smoothing splines, and local regressions; **choose** smoothness by cross-validation.
- **Articulate** the boundary problem of cubic splines and the fix offered by natural splines.
- **Build** a generalised additive model (GAM) over multiple predictors and **interpret** its component functions.
- **Reason** about when nonlinear methods help and when they merely add variance.

Roadmap of this chapter

A ladder of flexibility

1. Polynomials.
2. Step functions.
3. Regression splines (cubic, natural).
4. Smoothing splines.
5. Local regression (LOESS).
6. Generalised additive models (GAMs).

All of these are linear regression with cleverly chosen basis functions.

Outline

Why move beyond linearity?

Linear regression is interpretable and stable, but the world is rarely linear. Whenever the relationship between X and Y bends, a linear model leaves bias on the table.

The strategy of this chapter

Replace the strict linear form $\beta_j X_j$ with smooth functions $f_j(X_j)$, while keeping the additive structure that makes linear regression so easy to interpret.

Three reasons nonlinearity matters

Concrete examples

- Wage data peaks in middle age and then drops — not linear.
- Sales have diminishing returns to advertising spend — not linear.
- Survival probability is bounded in $(0, 1)$ — not linear.

Takeaway

Every method in this chapter is just linear regression with a cleverly chosen *basis expansion* of the original predictors. The OLS machinery still applies.

Where this chapter is used in industry

Sector	Concrete application	Which decision it drives
Motor, health insurance	Claim frequency and severity as smooth functions of driver age, vehicle age, mileage	The tariff a policy is priced from
Energy, utilities	Electricity demand against temperature (heating below, cooling above a comfort band)	Day-ahead procurement and capacity planning
Banking, fixed income	Yield curves fitted to quoted bond yields (cubic splines, Nelson–Siegel–Svensson)	How the book is valued and hedged
Retail, travel	Price-response and promotion curves, saturating at both ends	Price points, promotion depth, marketing budget
Pharma, toxicology	Dose–response curves for efficacy and for safety endpoints	The dose carried into the next trial phase
Life insurance, pensions	Mortality and morbidity as smooth functions of age	Reserves, annuity and longevity pricing

In industry — The common thread

In every row the fitted *curve* is the deliverable: it gets plotted, argued about and signed off. That rules out a black box — but a straight line would be plainly wrong.

Industry case in depth: pricing a motor insurance policy

In industry — The model behind the tariff

Insurers price in two pieces — **frequency** (claims per policy-year, Poisson) and **severity** (cost per claim, gamma) — and increasingly fit each as an additive model with smooth terms:

$$\log \mathbb{E}[\text{claims}] = \beta_0 + f_1(\text{driver age}) + f_2(\text{vehicle age}) + f_3(\text{mileage}) + \text{region} + \text{bonus–malus}.$$

What the fitted age curve looks like

It is famously *non-monotone*: risk is high for the youngest drivers, falls through middle age, then rises again for the oldest. No linear term in age can represent that; a smooth can, and prints as one readable curve — the curve an underwriter shows a broker or a customer who asks why a premium moved.

Who signs it off — and one caveat

The curves become tariff tables reviewed by pricing and the actuarial function, so they must be smooth and defensible: that, not accuracy, is why a GAM beats a black box here. *Caveat:* the tariff must also price rarely observed cells, where a high-degree polynomial would swing wildly — and additivity means an age $\times$ vehicle-power interaction has to be added deliberately.

Outline

Polynomial regression: the simplest extension

Replace X with a basis of powers.

Model

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_d X^d + \varepsilon.$$

How to read this

Still a *linear* model in the coefficients — all OLS machinery applies (standard errors, F-tests, CV). We are simply augmenting the design matrix.

In industry — Retail and travel: price-response curves

Demand bends in price and saturates at both ends, so pricing teams fit a quadratic or cubic in (log) price and read the elasticity off the fitted curve. Miss the curvature and the recommended price lands on the flat part of the curve — margin given away for no extra volume.

Polynomial regression: the boundary problem

Beware of high degree

Degrees beyond 4 rarely improve fit but routinely produce wild oscillations near the boundaries of the data range. Splines are the safer alternative.

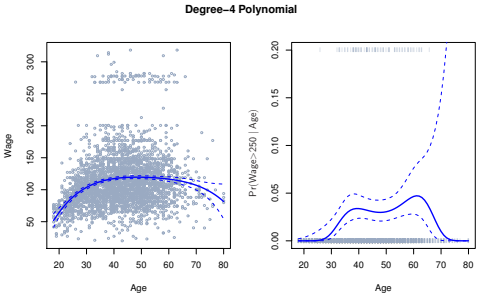
Why the wiggles?

A single global polynomial must use *one* formula everywhere. Each extra degree buys another turning point, and where data are sparse (the edges) those bends are barely constrained — so the curve swings and its confidence band balloons.

Takeaway

A useful tool for capturing mild curvature, especially with degree 2 or 3. Beyond that, switch to splines.

Wage data: degree-4 polynomial



Takeaway

Left: the degree-4 fit with its pointwise band; right: the fitted Pr(wage $>$ 250 | age). The curve captures the rise-then-plateau, but bends and widens at the sparse upper edge.

Source: Figure 7.1 — ISLP, James et al. (2023).

Step functions: piecewise constant

An alternative when the relationship is naturally piecewise constant.

Model

Bin X into $K+1$ ranges and fit a constant in each:

$$Y = \beta_0 + \beta_1 1\{c_1 \le X < c_2\} + \cdots + \beta_K 1\{c_K \le X\} + \varepsilon.$$

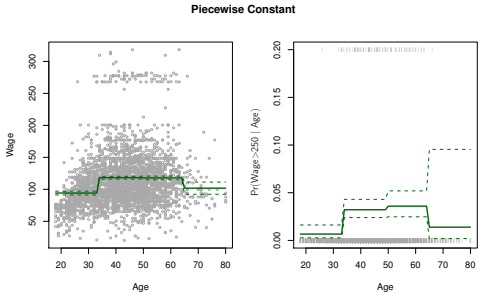
Pros and cons

Easy to interpret ("income bracket"), but discontinuous at the breakpoints — often visually ugly. Useful when natural categories exist (age groups, salary tiers).

In industry — Banking: scorecard bands

Credit scorecards deliberately bin continuous inputs — age, income, months since last arrears — into bands and award points per band: a step function by choice, because it is auditable and easy to monitor. The price is a visible jump for two applicants either side of a cut-off.

Step function fit on Wage



Takeaway

Piecewise-constant: flat within each band, jumping at every cutpoint. The level shifts are captured, but the discontinuities are an artefact of binning.

Source: Figure 7.2 — ISLP, James et al. (2023).

Exercise 7.1 — Evaluating a fitted cubic [Math]

Exercise

Consider a cubic polynomial regression of `wage` on age using the raw basis $\{1, x, x^2, x^3\}$.

1. Write the design-matrix row for a single observation with $x = 2$.
2. The fitted model is $\hat{f}(x) = 50 + 4x - 0.5x^2 + 0.1x^3$. Predict the response at $x = 2$ and $x = 4$.
3. Degree three sounds nonlinear. Explain why this is still a *linear* model.

Solution — Exercise 7.1 (1/2)

Solution

(a) Design-matrix row. Polynomial regression is ordinary OLS on a design matrix whose columns are powers of x, so each observation contributes the row of its basis values. With basis $\{1, x, x^2, x^3\}$ and $x = 2$:

$$\big[\,1,\ 2,\ 2^2,\ 2^3\,\big] = \big[\,1,\ 2,\ 4,\ 8\,\big].$$

For n observations the full design matrix stacks such rows: $X_{i\cdot} = [1,\ x_i,\ x_i^2,\ x_i^3]$.

(b) Predict by substituting term by term into $\hat{f}(x) = 50 + 4x - 0.5x^2 + 0.1x^3$:

$$\hat{f}(2) = 50 + 4(2) - 0.5(4) + 0.1(8) = 50 + 8 - 2 + 0.8 = 56.8,$$

$$\hat{f}(4) = 50 + 4(4) - 0.5(16) + 0.1(64) = 50 + 16 - 8 + 6.4 = 64.4.$$

Interpretation: between $x = 2$ and $x = 4$ the prediction rises by $64.4 - 56.8 = 7.6$; the curve still climbs here, but the negative x^2 term is already damping the growth.

Solution — Exercise 7.1 (2/2)

Solution

(c) Why this is still a linear model. "Linear" refers to the *coefficients*, not to the shape of the fitted curve. Writing

$$\hat{f}(x) = \beta_0 \cdot 1 + \beta_1 \cdot x + \beta_2 \cdot x^2 + \beta_3 \cdot x^3,$$

we see $\hat{f}$ is a linear combination of *known, pre-computed* regressors $1, x, x^2, x^3$. Hence all of the OLS machinery applies unchanged: closed-form $\hat{\beta}$, standard errors, t- and F-tests, cross-validation. The nonlinearity lives in x, not in the parameters.

Interpretation: this is the basis-function idea behind the whole chapter — keep the linear-regression machinery, change only the columns of the design matrix.

Common mistake

Predicting $\hat{f}(4)$ by doubling $\hat{f}(2)$. Polynomial predictions do not scale linearly in x; every power term must be substituted separately.

Exercise 7.2 — Step functions and dummy coding [Concept]

Exercise

We fit a piecewise-constant (step-function) regression of `wage` on `age` using cutpoints at 35 and 50, giving three bins: $(-\infty, 35)$, $[35, 50)$, $[50, \infty)$.

1. How many dummy variables do we create, and why is one bin omitted?
2. Write the fitted model with an intercept.
3. With $\hat{\beta}_0 = 40$, $\hat{\beta}_1 = 15$ (bin $[35, 50)$), $\hat{\beta}_2 = 25$ (bin $[50, \infty)$), predict `wage` for ages 30, 40, and 60.
4. State one limitation of this approach.

Solution — Exercise 7.2 (1/2)

Solution

(a) Two dummies for three bins. Define $C_1(x) = 1[35 \leq x < 50]$ and $C_2(x) = 1[x \geq 50]$. The first bin $(-\infty, 35)$ is the *reference category*, absorbed into the intercept. *Why omit one bin?* With an intercept, the column of ones equals the sum of all three bin dummies, so including all three would make the design matrix perfectly collinear — the "dummy-variable trap". In general: m bins $\Rightarrow m - 1$ dummies plus an intercept.

(b) Fitted model.

$$y = \beta_0 + \beta_1 C_1(x) + \beta_2 C_2(x) + \varepsilon.$$

Reading the coefficients: β_0 is the mean wage in the reference bin; β_1 and β_2 are the mean *differences* of bins 2 and 3 relative to that reference — not relative to each other.

Solution — Exercise 7.2 (2/2)

Solution

(c) Predictions. Locate each age's bin, switch on its dummy, and add:

- Age $30 < 35$: reference bin, $C_1 = C_2 = 0 \Rightarrow \hat{y} = 40$.
- Age $40 \in [35, 50)$: $C_1 = 1, C_2 = 0 \Rightarrow \hat{y} = 40 + 15 = 55$.
- Age $60 \geq 50$: $C_1 = 0, C_2 = 1 \Rightarrow \hat{y} = 40 + 25 = 65$.

Interpretation: predicted wage steps from 40 to 55 at age 35 and to 65 at age 50 — everyone inside a bin gets the same prediction.

(d) Limitation. The fit is constant within each bin and jumps at the cutpoints: it ignores any trend inside a bin, and the cutpoints are chosen arbitrarily. Unless the true relationship really has breakpoints, this misses smooth structure.

Common mistake

Reading $\beta_2 = 25$ as the jump from bin 2 to bin 3. Both coefficients compare to the *reference* bin; the bin-2-to-bin-3 step is $\beta_2 - \beta_1 = 10$.

Outline

Basis functions: a unifying view

Polynomials and step functions are special cases of:

Basis-function regression

$$Y = \beta_0 + \sum_{j=1}^{K} \beta_j b_j(X) + \varepsilon,$$

for fixed basis functions $b_1, \ldots, b_K$.

How to read this

Read it as: each $b_j(\cdot)$ is a *fixed* transformation chosen in advance, and β_j is its ordinary regression weight — so once the b_j are set, this is plain OLS. Special cases:

- Polynomials: $b_j(X) = X^j$.
- Step functions: $b_j(X) = 1\{c_j \le X < c_{j+1}\}$.
- Splines: piecewise polynomials joined smoothly at chosen knots.

Regression splines: idea

Splines combine the flexibility of polynomials with the locality of step functions.

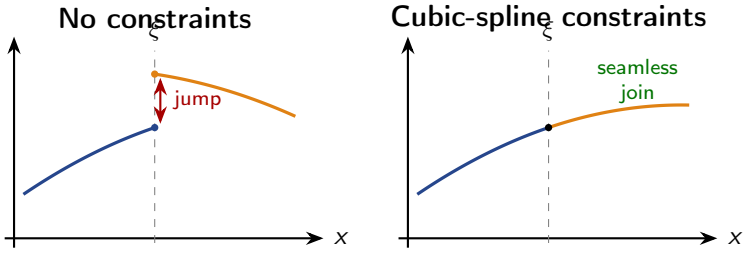
Construction

Split the range of X at K *knots* $\xi_1 < \cdots < \xi_K$. Fit a polynomial of degree d on each piece, joined at the knots so the function and its first $d-1$ derivatives are continuous.

Practical defaults

Cubic splines ($d = 3$) are the standard choice — they are smooth visually. Total degrees of freedom: $K + d + 1$.

Why the continuity constraints? Broken vs. smooth joins



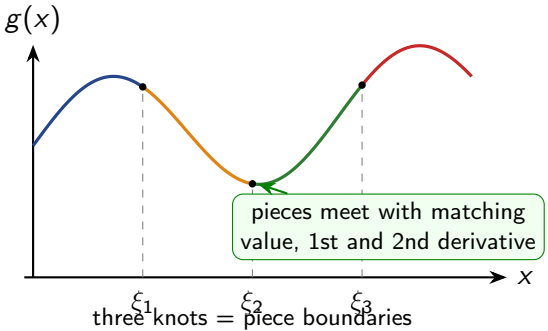
two pieces fit *separately*

equal value, slope & curvature

Takeaway

A regression spline is *not* one global polynomial: it is separate low-degree pieces. All the work is in the join — forcing equal value, slope and curvature at each knot makes the seam invisible while each piece stays flexible.

How a cubic spline is built: continuity at the knots



Takeaway

Each region is a free cubic, yet at every knot the pieces share value, slope and curvature — so the joins are invisible.

The natural-spline boundary fix

Boundary problem

Cubic splines flare beyond extreme knots. The CIs at the boundaries become unreliably wide.

Why the edges misbehave

Past the outermost knots the fit is still a free cubic, but almost no data pin it down there. A cubic can curve sharply, so a small change in a boundary point swings the whole tail — inflating variance exactly where evidence is thinnest.

Natural cubic splines

Constrain the spline to be *linear* past the boundary knots. The resulting fits are stable at the edges with negligible bias inside.

Choosing knots and degrees of freedom

Two practical decisions

- **How many knots?** Choose by cross-validation. Modest df (4–8) usually suffices.
- **Where?** Equal quantiles of X usually work well. With small n, place knots manually at obvious change-points.

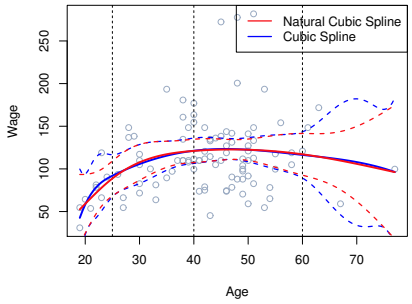
Why cubic and not quintic?

Cubic is the lowest degree that gives visually smooth fits (matches second derivative). Higher degrees rarely improve the fit but add parameters and instability.

In industry — Fixed income: how many knots in a yield curve?

A desk fitting a spline to quoted bond yields faces exactly this decision: more knots reproduce every quote but can imply implausible forward rates; fewer knots give a smooth curve that misprices some bonds. The chosen curve then discounts the whole book — so a smoothness choice is a valuation choice.

Cubic vs. natural spline on Wage

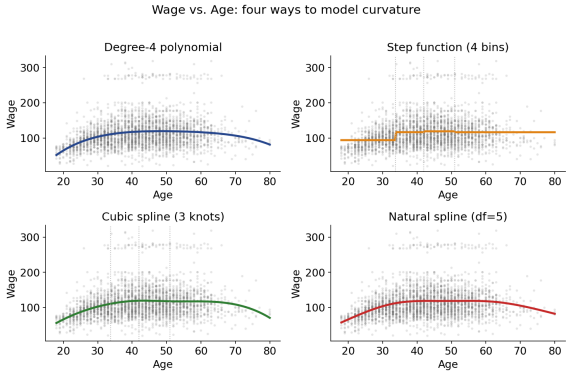


Takeaway

Same data, two fits: the cubic spline's error bands blow up at the edges while the natural spline's stay controlled — boundary behaviour is the deciding difference.

Source: Figure 7.4 — ISLP, James et al. (2023).

Wage vs. age: four models side by side



Takeaway

All four recover the same rise–plateau–fall shape but differ at the edges: the polynomial dips, the step function is blocky, the natural spline stays smooth and stable.

Exercise 7.3 — Degrees of freedom of a cubic spline [Math]

Exercise

Consider a cubic regression spline with K interior knots, fitted so that the function and its first two derivatives are continuous at each knot.

1. If we fit a *separate* cubic in each of the $K + 1$ regions, how many free parameters is that?
2. How many constraints do the continuity conditions impose in total?
3. Show that the net degrees of freedom equal $K + 4$.
4. How many basis functions does the truncated-power basis use (including the intercept)? Confirm consistency.

Solution — Exercise 7.3 (1/2)

Solution

(a) Free parameters. K knots split the line into $K+1$ regions. Each region carries its own cubic $a_j + b_j x + c_j x^2 + d_j x^3$, i.e. 4 coefficients, so fitting them separately costs

$$4 \times (K+1) = 4K + 4 \text{ parameters.}$$

(b) Constraints. At each interior knot we require the two adjacent pieces to agree in *value*, *slope*, and *curvature* (g, g', g'' continuous): 3 constraints per knot, hence

$$3 \times K = 3K \text{ constraints in total.}$$

Why stop at the second derivative? Matching g''' as well would force adjacent cubics to be identical — the "spline" would collapse to one global cubic. Continuity up to g'' is exactly what the eye perceives as smooth.

Solution — Exercise 7.3 (2/2)

Solution

(c) Net degrees of freedom. Each independent linear constraint removes one free parameter:

$$\underbrace{(4K+4)}_{\text{free params}} - \underbrace{3K}_{\text{constraints}} = K+4.$$

Sanity check: $K=0$ gives 4 — a single cubic, as it must. Each additional knot buys exactly *one* extra degree of freedom.

(d) Basis count. The truncated-power basis is

$$1,\ x,\ x^2,\ x^3,\ (x-\xi_1)_+^3, \ldots, (x-\xi_K)_+^3,$$

which is $4+K$ functions — one regression coefficient each, exactly matching the $K+4$ df from (c). Each truncated term adds one knot while preserving continuity of g, g', g'', so the two counting arguments agree.

Common mistake: counting 4 constraints per knot (including g''') or forgetting the intercept among the basis functions — both break the $K+4$ tally.

Exercise 7.4 — Natural vs. cubic splines [Concept]

Exercise

1. What extra constraints turn a cubic spline into a *natural* cubic spline?
2. A cubic spline with K knots has $K + 4$ df. How many df does a natural cubic spline with K knots have?
3. Why do natural splines behave better near the boundaries of the data?
4. What does this imply for the width of pointwise confidence bands at the extremes of x?

Solution — Exercise 7.4 (1/2)

Solution

(a) Extra constraints. A natural spline is required to be *linear* beyond the two boundary knots. A function is linear exactly when its second and third derivatives vanish, so the requirement translates into

$$g''(\xi) = 0 \quad \text{and} \quad g'''(\xi) = 0 \quad \text{at and beyond each boundary knot,}$$

i.e. 2 extra constraints at each of the 2 boundaries: 4 additional constraints in total.

(b) Degrees of freedom. Start from the cubic-spline count and subtract one df per constraint:

$$(K + 4) - 4 = K.$$

Interpretation: a natural spline with K knots is exactly as complex as a linear model with K coefficients — its flexibility budget is spent entirely in the interior, where the data live.

Solution — Exercise 7.4 (2/2)

Solution

(c) Why natural splines behave at the boundaries. Past the outer knots an unrestricted cubic spline is still a free cubic, but almost no data pin it down there: a handful of extreme observations steer four polynomial coefficients, so small data changes swing the whole tail — high variance exactly where evidence is thinnest. Forcing linear tails removes those unsupported cubic wiggles.

(d) Consequence for confidence bands. The pointwise band width tracks the variance of $\hat{g}(x)$. Because fewer parameters are spent at the extremes and the tail shape is constrained to be linear, natural-spline bands are substantially *narrower* at the boundaries than cubic-spline bands, at the price of a negligible bias increase in the interior.

Common mistake

Thinking the natural spline is linear *between* the boundary knots too — inside the boundary knots it is every bit as flexible a cubic spline; only the two tails are straightened.

Outline

Smoothing splines: let the data choose

Instead of choosing knots manually, place a knot at *every* data point and let a penalty control smoothness.

Penalised RSS

$$\min_g \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int g''(t)^2 \, dt.$$

In industry — Utilities: the temperature-response curve

Electricity load against temperature is a U shape — heating below the comfort band, cooling above it. Utilities fit that response as a smooth curve, add calendar effects, and buy power a day ahead against it. An over-smoothed peak understates demand on the hottest day — exactly when power costs most.

Reading the smoothing-spline objective

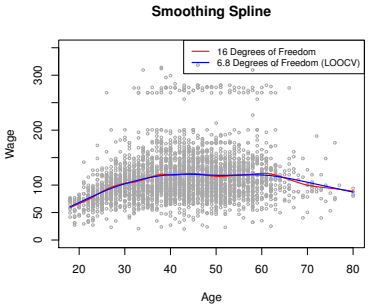
How to read this

Term	Role
$\sum(y_i - g(x_i))^2$	Training fit
$\int g''(t)^2 dt$	Penalty for "wiggliness" (curvature)
λ	Smoothness tuning parameter

Three observations

- The minimiser is a natural cubic spline with knots at every x_i.
- $\lambda = 0$: interpolation. $\lambda \to \infty$: linear fit.
- "Effective degrees of freedom" parameterise the smoothness.

Smoothing spline on Wage

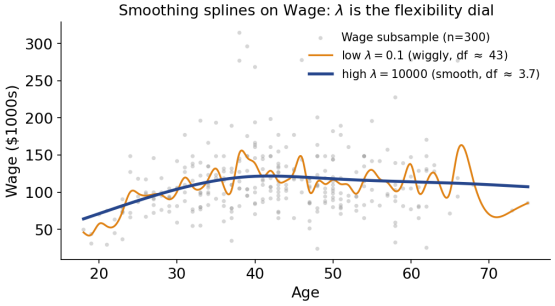


Takeaway

Two ways to fix smoothness: a preset 16 df vs. the ≈ 6.8 effective df chosen by CV. More effective df means a wigglier fit; CV finds the sweet spot.

Source: Figure 7.8 — ISLP, James et al. (2023).

Smoothing spline flexibility: λ is the dial



Takeaway

Same Wage data, two penalties: $\lambda = 10^4$ gives a calm trend (≈ 3.7 effective df) while $\lambda = 0.1$ (≈ 43 df) chases noise — λ is the flexibility dial, tuned by CV/LOOCV.

Local regression (LOESS): a different philosophy

Instead of one global function, fit a small model around each point.

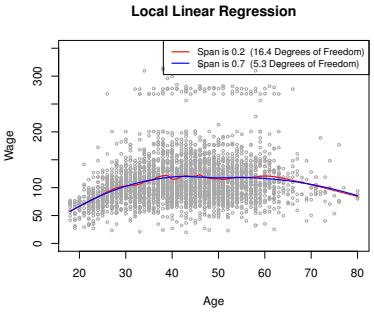
Recipe at each query point x_0

1. Find the fraction s ("span") of points closest to x_0.
2. Weight them by a kernel that decays with distance.
3. Fit a low-degree polynomial by weighted least squares.
4. Evaluate at x_0; repeat over the grid of x_0.

How to read this

Span s controls smoothness — the analogue of λ. LOESS is excellent for one or two predictors and dies fast as p grows.

LOESS on Wage



Takeaway

Each curve uses a different span: 0.2 tracks local wiggles, 0.7 is smoother. Span plays the role of λ, trading bias against variance.

Source: Figure 7.10 — ISLP, James et al. (2023).

Mini-check: spline choices

Quick self-assessment

1. Why prefer a natural cubic spline over a regular cubic spline at the boundaries?
2. How is a smoothing spline's λ related to its effective degrees of freedom?
3. In LOESS, what happens to bias as the span shrinks?

Answers

1. Natural splines are constrained linear past the boundary knots, avoiding wild boundary oscillation. 2. Larger $\lambda \Rightarrow$ smaller df (more shrinkage, smoother fit). 3. Bias decreases (the local fit is more local), but variance increases.

Exercise 7.5 — Smoothing splines and λ [Concept]

Exercise

A smoothing spline chooses g to minimise

$$\sum_{i=1}^{n}(y_i - g(x_i))^2 \; + \; \lambda \int g''(t)^2 \, dt.$$

1. Interpret the penalty term. What does it measure?
2. Describe the fit as $\lambda \to 0$ and as $\lambda \to \infty$.
3. Define the effective degrees of freedom df_λ and give its range.
4. How is λ chosen in practice?

Solution — Exercise 7.5 (1/2)

Solution

(a) The penalty measures curvature. $g''(t)$ is the rate of change of the slope at t; squaring removes the sign and integrating adds curvature up over the whole range, so $\int g''(t)^2\,dt$ is the total *roughness* of g. A straight line has $g'' \equiv 0$ and pays no penalty; a wiggly curve pays a lot. The penalty therefore discourages wiggliness.

(b) The two extremes. As $\lambda \to 0$ curvature becomes free, so the minimiser drives the RSS term to 0: g *interpolates* the data — extremely wiggly, low bias, high variance. As $\lambda \to \infty$ any curvature is infinitely expensive, forcing $g'' = 0$ everywhere, so g collapses to the ordinary least-squares *line*.

Interpretation: λ sweeps a continuous bridge between interpolation and linear regression — it is the bias–variance dial of the smoothing spline.

Solution — Exercise 7.5 (2/2)

Solution

(c) Effective degrees of freedom. For fixed λ the fit is linear in y: $\hat{g} = S_\lambda\, y$ for a smoother matrix S_λ. In OLS the analogous "hat" matrix has trace equal to the number of parameters, which motivates the definition

$$\mathrm{df}_\lambda = \mathrm{trace}(S_\lambda),$$

ranging from n (at $\lambda = 0$, interpolation) down to 2 (at $\lambda \to \infty$, slope + intercept of the linear fit).

(d) Choosing λ. By cross-validation. LOOCV is essentially free because each leave-one-out residual follows from the *single* full fit via the leverage $\{S_\lambda\}_{ii}$:

$$\mathrm{RSS}_{\mathrm{cv}}(\lambda) = \sum_{i=1}^{n} \left(\frac{y_i - \hat{g}_\lambda(x_i)}{1 - \{S_\lambda\}_{ii}} \right)^2 .$$

Common mistake

Picking λ by minimising *training* RSS — that always selects $\lambda = 0$ (interpolation). Only out-of-sample criteria can trade fit against roughness.

Outline

From one predictor to many: GAMs

Everything we have done so far is for one predictor at a time. Generalised additive models lift the techniques into the multivariate setting.

The GAM model

$$Y = \beta_0 + \sum_{j=1}^{p} f_j(X_j) + \varepsilon,$$

where each f_j is a smooth function (spline or LOESS).

Symbol by symbol

β_0 is the overall level; each f_j is a smooth, data-driven curve for predictor X_j; summing them keeps the model additive. Setting $f_j(X_j) = \beta_j X_j$ recovers ordinary linear regression — the GAM simply lets each term bend.

Why GAMs are so useful

Three reasons

1. Generalises linear regression: $f_j(X_j) = \beta_j X_j$ is the linear special case.
2. Avoids the curse of dimensionality via the additive structure.
3. Each f_j can be plotted separately — excellent interpretability.

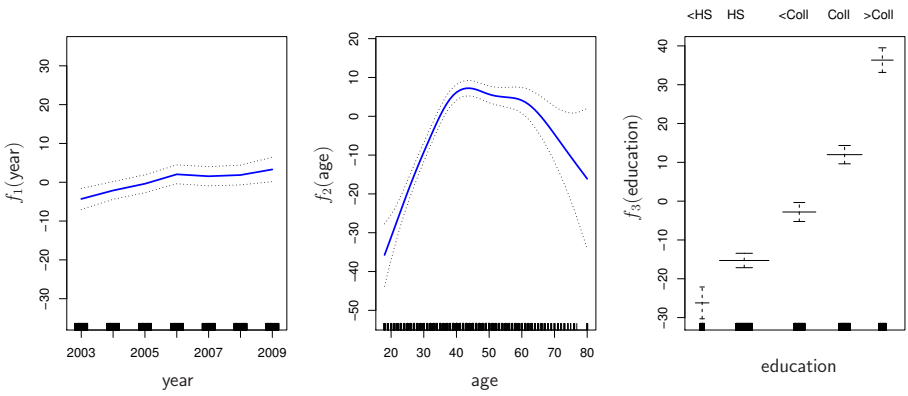
Limitation

By construction, GAMs miss interactions. Add tensor-product terms $f_{jk}(X_j, X_k)$ explicitly, or move to trees (Ch. 8).

In industry — Insurance and utilities: why they reach for GAMs

Insurers, utilities and banks need a model that bends *and* can be defended one predictor at a time: each f_j is a picture an actuary, a regulator or a model-risk reviewer can hold against domain knowledge.

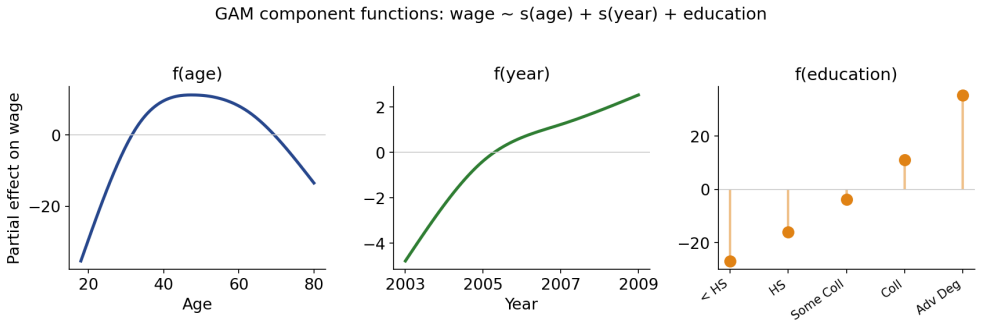
GAM fit on Wage



Each panel: estimated f_j for a different predictor, with pointwise confidence bands.

Source: Figure 7.12 — ISLP, James et al. (2023).

GAM component functions fitted on Wage



Takeaway

Reading each partial effect: wage rises with age then falls after ∼60; it drifts up gently with year; and it increases monotonically with education. The additive structure lets us plot and interpret one predictor at a time.

GAMs for classification

Same idea inside a GLM:

Logistic GAM

$$\log \frac{p(X)}{1-p(X)} = \beta_0 + \sum_{j=1}^{p} f_j(X_j).$$

Reading the model

The left side is the *log-odds* of the positive class; each smooth f_j shifts those log-odds up or down as X_j changes. It is an additive logistic model — interpret every f_j on its own, just as in a linear GLM.

Combines logistic regression's interpretability with smooth per-predictor effects.
Implementations: `pyGAM`, `statsmodels.gam`.

Extended Exercise 7.2 — A GAM for wage [Integrative]

Extended exercise (15 min)

Fit a generalised additive model

$$\texttt{wage} \approx \beta_0 + s(\texttt{age}) + \texttt{year} + f(\texttt{education}),$$

with a smooth spline in age, a (near-)linear term in year, and education as a factor.

1. Fit it (e.g. pyGAM) and state how a GAM is estimated.
2. Interpret each partial effect.
3. Which term is genuinely non-linear? Discuss the effective degrees of freedom (EDF) of each term.
4. State the advantages of the additive form and its key limitation. Give an interaction it would miss and how to add it.

Solution — Extended Exercise 7.2 (1/3)

Solution

(a) Fit & algorithm. Encode `education` to integer codes, then fit a spline in age, a linear term in year and a factor in education:

```
import pandas as pd
from pygam import LinearGAM, s, l, f

W = pd.read_csv('../ALL CSV FILES - 2nd Edition/Wage.csv') # load Wage
W['edu'] = W['education'].astype('category').cat.codes # factor -> codes
X = W[['age', 'year', 'edu']].values # predictor matrix
y = W['wage'].values # response
gam = LinearGAM(s(0) + l(1) + f(2)).fit(X, y) # s=spline, l=linear, f=factor
gam.summary() # effective DoF and p-value per term
```

A GAM is fit by *backfitting*: cycle through the terms, repeatedly re-estimating each f_j on the partial residual $y - \beta_0 - \sum_{k \neq j} f_k(X_k)$ until the fit stabilises — each smooth is refitted to whatever the other terms leave unexplained.

Solution — Extended Exercise 7.2 (2/3)

Solution

(b) Partial effects (each holds the others fixed):

- `age`: a pronounced *hump* — wage climbs steeply to about age 40, plateaus through the 50s, then declines after $\sim$60.
- `year`: a gentle, near-linear *increase* — slow wage growth over the sample window.
- `education`: a monotone *staircase* — each higher band pays more, "Advanced Degree" far above "<HS Grad".

(c) Non-linearity & EDF. Only age is genuinely non-linear: its smooth spends many effective degrees of freedom (≈ 15 at pyGAM's default penalty, dropping to ≈ 10 once λ is tuned by `gam.gridsearch`). `year` enters with EDF ≈ 1 (essentially a straight line); `education` as a factor contributes df $=$ (levels -1) $= 4$. The EDF is how a GAM quantifies "how wiggly" each term is.

Common mistake

Reading a partial-effect curve as an absolute wage level. Each panel is centred around zero; only its *shape* and within-panel differences are meaningful, with all other predictors held fixed.

Solution — Extended Exercise 7.2 (3/3)

Solution

(d) Advantages of additivity.

- Each term is estimated and *plotted separately*, so effects are directly interpretable.
- The additive structure sidesteps the curse of dimensionality (no p-dimensional smoother).
- It strictly generalises linear regression: a linear term is the special case $f_j(X_j) = \beta_j X_j$.

Key limitation — no interactions. By construction the shape of $s(\texttt{age})$ is the *same* at every education level. If the wage–age profile were steeper for college graduates than for high-school graduates (an $\texttt{age} \times \texttt{education}$ interaction), the additive GAM could not represent it. *Fix:* add a bivariate/tensor term $f(\texttt{age}, \texttt{education})$, fit separate age-smooths per group, or move to trees / boosting (Ch. 8), which capture interactions automatically.

Outline

Python lab — run it live in the notebook

Companion notebook: `chapter_07_lab.ipynb`

The code in this section is meant to be **demonstrated live**. Open the companion Jupyter notebook and run it cell by cell, section by section:

- §1, *Polynomial regression*, and §2, *Step functions*;
- §3, *Regression splines*, and §4, *LOESS*;
- §5, *GAM*.

Data loads via the `ISLP` package or the bundled CSV files; §6, *Exercises*, closes the notebook.

Polynomial regression

```python
import statsmodels.api as sm
from ISLP import load_data
from ISLP.models import poly, ModelSpec as MS

Wage = load_data("Wage") # load the Wage data
design = MS([poly("age", degree=4)]).fit_transform(Wage) # degree-4 basis
res = sm.OLS(Wage["wage"], design).fit() # OLS on expanded design
print(res.summary()) # coefficient table
```

Regression splines

```
from ISLP.models import bs, ns

design = MS([bs("age", df=6)]).fit_transform(Wage) # cubic spline, 6 df
design2 = MS([ns("age", df=4)]).fit_transform(Wage) # natural spline, 4 df
res = sm.OLS(Wage["wage"], design2).fit() # OLS on the spline basis
print(res.summary()) # per-basis coefficients
```

Local regression with statsmodels

```
import statsmodels.nonparametric.api as nps

l = nps.lowess(endog=Wage["wage"], exog=Wage["age"], # local regression
               frac=0.2, return_sorted=True) # span: 20% nearest points
import matplotlib.pyplot as plt
plt.scatter(Wage["age"], Wage["wage"], s=4, alpha=0.3) # raw data
plt.plot(l[:, 0], l[:, 1], color="orange") # fitted LOESS curve
plt.show()
```

A simple GAM with pyGAM

```
from pygam import LinearGAM, s, f

Wage["education"] = Wage["education"].astype("category").cat.codes # encode factor
X = Wage[["year", "age", "education"]] # three predictors
y = Wage["wage"] # response
gam = LinearGAM(s(0) + s(1) + f(2)).fit(X, y) # smooth year, age; factor edu
gam.summary() # EDF and p-values per term
```

Exercise 7.6 — Natural spline / GAM on Wage [Python]

Exercise

Using `Wage.csv`:

1. Fit `wage` on a natural cubic spline of `age` with 5 df via OLS.
2. Fit a GAM `wage` $\sim s(\texttt{age}) + s(\texttt{year})$.
3. Describe the estimated effect of `age`.

Solution — Exercise 7.6 (1/3)

Solution

(a) Natural cubic spline via patsy's `cr()` basis — `cr()` expands age into 5 natural-spline columns (linear tails built in), so the fit is still ordinary OLS:

```
import pandas as pd
import statsmodels.formula.api as smf

Wage = pd.read_csv('../ALL CSV FILES - 2nd Edition/Wage.csv') # load
fit = smf.ols('wage ~cr(age, df=5)', data=Wage).fit() # natural spline
print(fit.summary()) # coefficient table
```

What the output shows: 6 coefficients (intercept + 5 basis columns), $R^2 \approx 0.087$, and an overall F-test with $p \approx 10^{-56}$ — the spline terms are jointly highly significant, even though age alone explains only a modest share of wage variance.

Solution — Exercise 7.6 (2/3)

Solution

(b) A GAM with smooth terms via `pyGAM` — each `s()` is a penalised spline whose wiggliness is measured in effective degrees of freedom (EDF):

```
import numpy as np
from pygam import LinearGAM, s

X = Wage[['age', 'year']].values # predictor matrix
y = Wage['wage'].values # response
gam = LinearGAM(s(0) + s(1)).fit(X, y) # smooth terms in age and year
gam.summary() # EDF per smooth term
```

What the output shows: with pyGAM's default penalty the age smooth uses ≈ 15 EDF and `year` ≈ 6 — both flexible; tuning λ by `gam.gridsearch(X, y)` shrinks the smooths toward simpler shapes. Both terms are reported significant.

Solution — Exercise 7.6 (3/3)

Solution

(c) *Interpretation.* The fitted effect of age is clearly nonlinear: `wage` rises steeply from the late teens to about age 40, is roughly flat (a plateau) between 40 and 60, then declines gently after 60. A straight line would miss this rise-then-plateau-then-fall shape, which is why a spline / GAM is preferable. The `year` effect is mildly increasing (slow wage growth over time).

How to read the fit: plot `gam.partial_dependence()` per term rather than staring at coefficients — the spline basis coefficients have no individual meaning; only the fitted curve does.

Common mistake

Interpreting single spline-basis coefficients (or their t-tests) as "the effect of age". Judge the term by its fitted curve and a *joint* test of all its basis columns.

Extended Exercise 7.3 — Three fits for wage vs. age [Python]

Extended exercise (15 min)

Using `Wage.csv`, model `wage` as a function of age three ways and compare:

1. a **polynomial**, choosing the degree by nested ANOVA F-tests (or 5-fold CV);
2. a **step function** (bin age into several ranges);
3. a **natural cubic spline** with about 5 df.

Overlay the three fitted curves on a scatterplot, report the chosen complexity, and interpret where the fits agree and where they differ.

Run this live: *Extended Exercise 7.3 — Three fits for wage vs. age*

Under *Lecture exercises — worked Python solutions* in `chapter_07_lab.ipynb`: the nested ANOVA, the three fits and the overlay plot, ready to run.

Solution — Extended Exercise 7.3 (1/3)

Solution

Choose the polynomial degree by nested ANOVA.

```
import numpy as np, pandas as pd
import statsmodels.formula.api as smf
from statsmodels.stats.anova import anova_lm

W = pd.read_csv('../ALL CSV FILES - 2nd Edition/Wage.csv') # load Wage
terms = lambda d: '+'.join(f'np.power(age,{k})' for k in range(1, d+1)) # power basis
models = [smf.ols('wage ~'+terms(d), data=W).fit() for d in range(1, 6)] # degrees 1..5
print(anova_lm(*models)[['df_diff', 'F', 'Pr(>F)']].round(4)) # nested F-tests
```

Each row of `anova_lm` tests degree d against $d+1$ — valid because the models are *nested*. The F-tests are highly significant up to degree 3 ($2 \to 3$: $p \approx 0.002$), *borderline* for $3 \to 4$ ($p \approx 0.05$) and clearly insignificant for $4 \to 5$ ($p \approx 0.37$). 5-fold CV of test MSE is essentially flat over degrees 3–4. **Chosen degree: 3 (cubic).**

Solution — Extended Exercise 7.3 (2/3)

Solution

Step function, natural spline, and the overlay.

```
import matplotlib.pyplot as plt
poly = smf.ols('wage ~'+terms(3), data=W).fit() # cubic polynomial
W['bin'] = pd.cut(W['age'], [17,25,35,45,55,65,81]) # six age bins
step = smf.ols('wage ~C(bin)', data=W).fit() # step function
ns = smf.ols('wage ~cr(age, df=5)', data=W).fit() # natural spline

g = pd.DataFrame({'age': np.arange(18, 81)}) # prediction grid
g['bin'] = pd.cut(g['age'], [17,25,35,45,55,65,81]) # bin grid too
plt.scatter(W['age'], W['wage'], s=4, alpha=.15, color='grey') # data
for m, lab in [(poly,'poly (d=3)'), (step,'step'), (ns,'natural spline')]:
    plt.plot(g['age'], m.predict(g), label=lab) # overlay each fit
plt.legend(); plt.xlabel('age'); plt.ylabel('wage'); plt.show()
```

Why the grid carries both columns: `predict` re-applies each formula, so `g` must contain `age` (for `poly`, `ns`) *and* the matching `bin` labels (for `step`), built with the *same* cutpoints as in training.

Solution — Extended Exercise 7.3 (3/3)

Solution

Reading the overlay.

- *Where they agree:* in the data-dense interior (ages ∼25–70) all three fits trace the same rise–plateau–fall — `wage` climbs steeply to about age 40, is roughly flat through the 50s, then eases down.
- *Where they differ:* at the sparse boundaries. The polynomial can bend sharply at the extremes; the step function is blocky and *discontinuous* at the cut-points; the natural cubic spline stays smooth and, being linear beyond its outer knots, is the most stable at the edges.

Chosen complexity: a cubic polynomial (degree 3–4), a 6-bin step function, and a natural spline with ∼5 df — all three explain a similar share of variance ($R^2 \approx 0.08$–0.09), but the natural spline is preferred for its boundary stability.

Common mistake

Picking the degree that maximises training R^2 — it rises mechanically with every added term; only the nested F-tests or CV can say when the gain is real.

Outline

Chapter 7 in one slide

What we accomplished

A toolbox for adding flexibility to a regression while keeping the additive, interpretable structure of a linear model.

- Polynomials, step functions, regression splines, smoothing splines, LOESS.
- GAMs as the natural multivariate extension.
- Cross-validation chooses the smoothness in every method.

Key formulas at a glance

Polynomial $y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d + \varepsilon_i$.

Step function $y_i = \beta_0 + \sum_{k=1}^{K} \beta_k C_k(x_i) + \varepsilon_i$, with bins C_k.

Cubic spline truncated-power basis $1, x, x^2, x^3, (x-\xi_k)_+^3$; K knots $\Rightarrow$ $\mathrm{df} = K + 4$.

Natural spline linear beyond the boundary knots; $\mathrm{df} = K$.

Smoothing spline $\min_g \ \sum_i \big(y_i - g(x_i)\big)^2 + \lambda \int g''(t)^2 \, dt$.

Effective df $\mathrm{df}_\lambda = \mathrm{trace}(S_\lambda)$, where $\hat{g} = S_\lambda y$; larger $\lambda \Rightarrow$ *smaller* df_λ: from n (at $\lambda = 0$, interpolation) down to 2 (as $\lambda \to \infty$, a straight line).

GAM $y_i = \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + \varepsilon_i$.

Backfitting re-estimate each f_j in turn on the partial residual $y - \beta_0 - \sum_{k \neq j} f_k(X_k)$, cycling through the terms until convergence.

Vocabulary checklist

Term	Meaning
Basis function	Fixed transformation $b_j(X)$ entering linearly
Knot	Breakpoint where pieces of a spline meet
Cubic / natural spline	Degree-3 piecewise polynomial / linear at boundaries
Smoothing spline	Penalised second-derivative fit
Effective degrees of freedom	Trace of the smoother matrix
LOESS / local regression	Weighted local polynomial
Span	Fraction of nearest neighbours used in LOESS
GAM	$Y = \beta_0 + \sum_j f_j(X_j) + \varepsilon$
Backfitting	Iterative algorithm for fitting GAMs

Methods comparison

Method	Strength	Weakness
Polynomial	Easy, captures curvature	Boundary oscillation
Step function	Interpretable	Discontinuous
Cubic spline	Smooth, controllable df	Boundary blow-up
Natural spline	Stable boundaries	Tiny extra bias inside
Smoothing spline	Auto-fitting smoothness	Black-box df
LOESS	Locally adaptive	Curse of dimensionality
GAM	Additive, interpretable	Misses interactions unless added

Decision rules of thumb

- For a single X and modest curvature: polynomial degree 2–4.
- For a single X with multiple regimes: regression splines with a handful of knots placed at quantiles.
- For boundary stability: *natural* cubic splines.
- For multiple predictors with little interaction: a GAM.
- Choose smoothness (df, λ, span) by cross-validation.
- For visible interactions: add tensor-product terms or move to trees (Ch. 8).

Common pitfalls

Don't

1. Use polynomials of degree > 4 on real data.
2. Place knots arbitrarily without checking residuals.
3. Forget that LOESS fails fast as p grows.
4. Compare GAM df with linear-regression df one-for-one.
5. Trust pointwise CIs of smoothing splines for inference.
6. Ignore extrapolation — splines outside the data range are unreliable.

Connections to the rest of the course

- Ch. 3 is the special case where every f_j is linear.
- Ch. 5 chooses df / λ / span by CV.
- Ch. 6 ridge / lasso can be applied to spline basis expansions.
- Ch. 8 trees are the nonparametric alternative when interactions dominate.
- Ch. 10 deep nets generalise GAMs by learning the basis.

Self-check questions

Each question names the slide that answers it — a wrong answer tells you where to read.

1. Why is a cubic spline preferred over a quintic spline? (p. 31)
2. Argue intuitively why smoothing splines have a unique minimiser. (p. 42)
3. How does the smoothing-spline penalty λ relate to df? (p. 44)
4. Sketch a GAM with two predictors. Where would you add an interaction? (p. 53)
5. For LOESS, what happens to bias and variance as the span shrinks? (p. 47)
6. In the Wage data, would you expect a linear or a spline term in age to win? Why? (p. 33)

Seven things to remember

1. Polynomials = linear regression with X^j basis.
2. Step functions = piecewise constant.
3. Cubic splines = piecewise cubics joined smoothly.
4. Natural splines stabilise the boundaries.
5. Smoothing splines: penalised second derivative.
6. LOESS: weighted local polynomials.
7. GAMs: $\sum_j f_j(X_j)$ — flexible *and* interpretable.

What's next

Chapter 8: *Tree-Based Methods*.

- Decision trees for regression and classification.
- Bagging, random forests, boosting, BART.
- The dominant family of methods for tabular data.

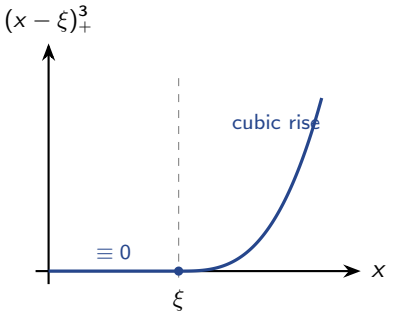
Appendix — what is in here, and why

Nothing in the appendix is needed to follow the chapter: each slide is a formal derivation, a longer worked exercise, or a side topic. Read it when you want the full story, or when a later chapter sends you back here.

Contents of the appendix

- **How the spline basis is built** — the truncated-power term $(x-\xi)_+^3$ and the constraint counting that turns loose cubic pieces into one smooth curve. The main deck keeps the picture ("Why the continuity constraints?") and the degrees-of-freedom rule, which is what you are asked to apply.
- **Regression splines by hand (Extended Exercise 7.1)** — evaluating a spline basis and its fit with pencil and paper. Exercise 7.3 covers the same counting argument in five minutes.

The building block: one truncated-power term per knot



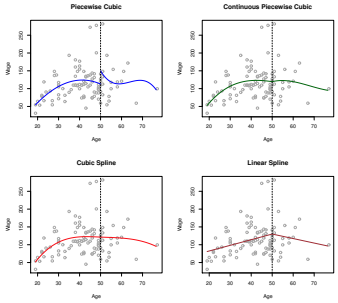
Reading $(x-\xi)_+^3$

The "+" is the *positive part*: the term is 0 for $x \le \xi$ and equals $(x-\xi)^3$ afterwards. Its value, slope and curvature are all 0 at ξ, so switching it on adds a bend *only to the right* of the knot without breaking C^2-smoothness.

Takeaway

Full basis for K knots: $1,\ x,\ x^2,\ x^3,\ (x-\xi_1)_+^3, \ldots, (x-\xi_K)_+^3$ — that is $K+4$ functions, exactly the $K+4$ degrees of freedom of a cubic spline.

Adding constraints turns loose pieces into a spline



Takeaway

Top-left: unconstrained pieces jump. Adding continuity of value, then slope, then curvature (bottom-left) removes the kinks — the cubic spline is smoothest.

Source: Figure 7.3 — ISLP, James et al. (2023).

Extended Exercise 7.1 — Regression splines by hand [Math]

Extended exercise (15 min)

Build a cubic regression spline for `wage` on `age` with $K = 3$ interior knots $\xi_1 = 25 < \xi_2 = 40 < \xi_3 = 60$.

1. Write the truncated-power basis and state the total degrees of freedom.
2. The fitted function is

$$\hat{g}(x) = 40 + 0.5\,x + 0.0004\,(x-25)_+^3 - 0.0002\,(x-40)_+^3 + 0.0003\,(x-60)_+^3.$$

   Evaluate $\hat{g}(50)$.
3. Show that $\hat{g}, \hat{g}', \hat{g}''$ are continuous at $\xi_2 = 40$ but $\hat{g}'''$ jumps.
4. How many degrees of freedom does a *natural* cubic spline with the same three knots use? Explain the difference.

Solution — Extended Exercise 7.1 (1/3)

Solution

(a) Truncated-power basis. With $K = 3$ knots the cubic-spline basis is

$$1,\ x,\ x^2,\ x^3,\ (x-\xi_1)_+^3,\ (x-\xi_2)_+^3,\ (x-\xi_3)_+^3,$$

so a fitted spline is $\hat{g}(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \sum_{k=1}^{3} \beta_{3+k}(x-\xi_k)_+^3$.

Degrees of freedom. Four base terms plus one per knot: $4 + K = 7$. Equivalently, a free cubic in each of the $K+1 = 4$ regions has $4 \cdot 4 = 16$ parameters, minus 3 continuity constraints (g, g', g'') at each of the 3 knots: $16 - 9 = 7$.

Why this basis works: each $(x-\xi_k)_+^3$ has value, slope and curvature 0 at its knot, so it bends the fit only to the *right* of ξ_k without breaking smoothness — plain OLS on these 7 columns automatically yields a C^2 piecewise cubic.

Solution — Extended Exercise 7.1 (2/3)

Solution

(b) Evaluate $\hat{g}(50)$. Here the quadratic and cubic *base* coefficients happen to be zero, so only the linear base term and the *active* truncated terms contribute. At $x = 50$ the knots 25 and 40 are exceeded but 60 is not:

$$(50-25)^3 = 25^3 = 15625, \quad (50-40)^3 = 10^3 = 1000, \quad (50-60)_+^3 = 0.$$

Therefore

$$\begin{aligned}\hat{g}(50) &= 40 + 0.5(50) + 0.0004(15625) - 0.0002(1000) + 0.0003(0) \\ &= 40 + 25 + 6.25 - 0.20 + 0 = 71.05.\end{aligned}$$

Only the basis functions "switched on" at $x = 50$ enter the sum — this locality is the point of the truncated basis.

Common mistake

Evaluating the third term as $(50-60)^3 = (-10)^3 = -1000$ instead of 0. The "+" truncation zeroes the term for $x \leq 60$; forgetting it would give the wrong value $71.05 - 0.30 = 70.75$.

Solution — Extended Exercise 7.1 (3/3)

Solution

(c) Continuity at $\xi_2 = 40$. Only the term $t(x) = (x-40)_+^3$ changes behaviour there. With $t' = 3(x-40)_+^2$ and $t'' = 6(x-40)_+$, its one-sided limits at $x = 40$ are

	t	t'	t''
$x \to 40^-$	0	0	0
$x \to 40^+$	0	0	0

so $\hat{g}, \hat{g}', \hat{g}''$ are continuous. But $t''' = 6 \cdot 1\{x > 40\}$ jumps from 0 to 6, hence $\hat{g}'''$ is discontinuous — the signature of a cubic spline (continuous curvature, kinked third derivative).

(d) Natural cubic spline. It is forced *linear* beyond the two boundary knots, imposing 2 extra constraints at each end ($g'' = g''' = 0$), i.e. 4 fewer df: $7 - 4 = 3$. A natural spline spends its budget in the interior and is far more stable at the edges.