Polynomial Regression

Noted by Md. Inzamamul Lohani

Software Engineering, IIT, DU

November 23, 2026

Contents

1 The Big Picture & Motivation **3**
1.1 Why Do We Need Polynomial Regression? . . . 3
1.2 What Exactly is a Polynomial? . . . 4
1.3 The Crucial Distinction: Linear in Parameters, Nonlinear in Features . . . 5
1.4 Comparison Table: Linear vs. Polynomial vs. Non-Parametric Models . . . 5

2 Mathematical Formulation & Normal Equations **5**
2.1 The Univariate Polynomial Regression Model . . . 5
2.2 The Vandermonde Design Matrix Representation . . . 6
2.3 First-Principles Derivation of the OLS Estimator . . . 6

3 Model Fitting: A Complete Worked Numerical Example **7**
3.1 The Dataset and Problem Definition . . . 7
3.2 Step-by-Step Construction of the Design Matrix . . . 7
3.3 Computing the Gram Matrix $\mathbf{X}^T\mathbf{X}$ . . . 8
3.4 Computing the Moment Vector $\mathbf{X}^T\mathbf{y}$ . . . 8
3.5 Matrix Inversion and Coefficient Estimation . . . 8
3.6 Predictions, Residual Analysis, and RSS . . . 9

4 Multivariate Polynomials & Feature Transformation **10**
4.1 Multivariate Polynomial Expansion . . . 10
4.2 Interaction Terms: Modeling Synergistic Relationships . . . 10
4.3 The Combinatorial Explosion of Feature Space . . . 11
4.4 Data Leakage Prevention during Transformation . . . 11

5 Bias-Variance Trade-off, Overfitting & Regularization **12**
5.1 Underfitting vs. Optimal Fit vs. Overfitting . . . 12
5.2 The Bias-Variance Decomposition . . . 12
5.3 Regularization: Taming Complex Polynomials . . . 13
5.3.1 Ridge Regression (L_2 Penalty) with Polynomial Features . . . 13
5.3.2 Lasso Regression (L_1 Penalty) for Feature Selection . . . 13
5.4 The Absolute Necessity of Feature Scaling . . . 14

6 Revision Sheet & Self-Test **14**
6.1 The Essential Formulas . . . 15
6.2 Conceptual Checklist . . . 15
6.3 Self-Test Questions . . . 15
6.4 Short Answers . . . 15

1 The Big Picture & Motivation

1.1 Why Do We Need Polynomial Regression?

In standard Ordinary Least Squares (OLS) Linear Regression, we assume that the relationship between the independent predictor $x \in \mathbb{R}$ and the continuous response variable $y \in \mathbb{R}$ is linear:

$$y = \beta_0 + \beta_1 x + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2). \tag{1.1}$$

This formulation enforces a strict constraint: the rate of change of y with respect to x is constant ($\frac{\partial y}{\partial x} = \beta_1$).

However, real-world phenomena rarely adhere to strict straight lines. Consider familiar scenarios:

- **Economics & Utility:** Diminishing marginal returns, where additional investments yield progressively smaller gains.
- **Physics & Kinetics:** Braking distance of a vehicle, which grows quadratically with vehicle velocity ($d \propto v^2$).
- **Medicine & Pharmacology:** Dose-response curves, where drug efficacy increases, plateaus, and can eventually become toxic at high concentrations.
- **Education & Cognition:** Exam performance vs. hours spent studying, where initial hours provide substantial improvements, followed by saturation and burnout.

When we fit a straight line to inherently curved data, the model suffers from severe **systematic bias** (underfitting). The residuals display distinct non-random parabolic or cyclical patterns rather than white noise.

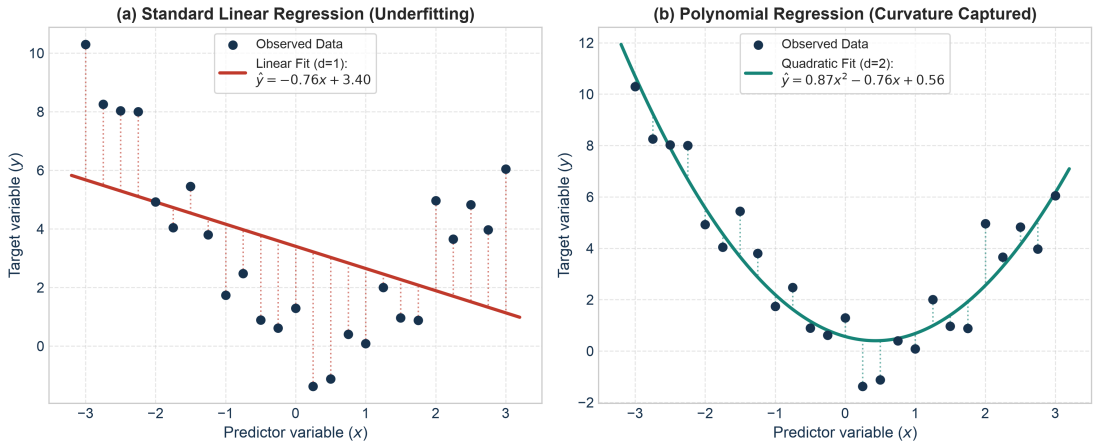

Figure 1: Comparison of Linear Regression vs. Polynomial Regression on nonlinear data. In (a), a linear model suffers from high bias, producing large systematic residuals. In (b), a degree-2 polynomial smoothly tracks the data's true underlying curvature.

Core Motivation of Polynomial Regression

Polynomial Regression expands the hypothesis class of linear models to fit nonlinear curves by adding powers of original predictors $(x^2, x^3, \ldots, x^d)$ as new input features, while keeping the estimation problem entirely linear with respect to the unknown coefficients $\boldsymbol{\beta}$.

1.2 What Exactly is a Polynomial?

A polynomial in a single real variable x is a mathematical expression composed of variables and coefficients involving only the operations of addition, subtraction, multiplication, and non-negative integer exponents:

$$P(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d = \sum_{j=0}^{d} \beta_j x^j. \tag{1.2}$$

The highest power $d \in \mathbb{N}_0$ with a non-zero coefficient $\beta_d \neq 0$ is called the **degree** of the polynomial.

- **Degree 0 (Constant):** $y = \beta_0$ (horizontal flat line, 0 turning points).
- **Degree 1 (Linear):** $y = \beta_0 + \beta_1 x$ (sloped straight line, 0 turning points).
- **Degree 2 (Quadratic):** $y = \beta_0 + \beta_1 x + \beta_2 x^2$ (parabola, exactly 1 turning point / vertex).
- **Degree 3 (Cubic):** $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$ (S-shaped curve, inflection point, up to 2 turning points).
- **Degree 4 (Quartic):** $y = \beta_0 + \beta_1 x + \cdots + \beta_4 x^4$ (W or M-shaped curve, up to 3 turning points).

Geometric Intuition of Degree & Turning Points

A polynomial of degree d can have at most $d - 1$ turning points (local extrema where the first derivative $\frac{dP}{dx} = 0$). Increasing the degree allows the function to bend more frequently, enabling it to fit complex curves.

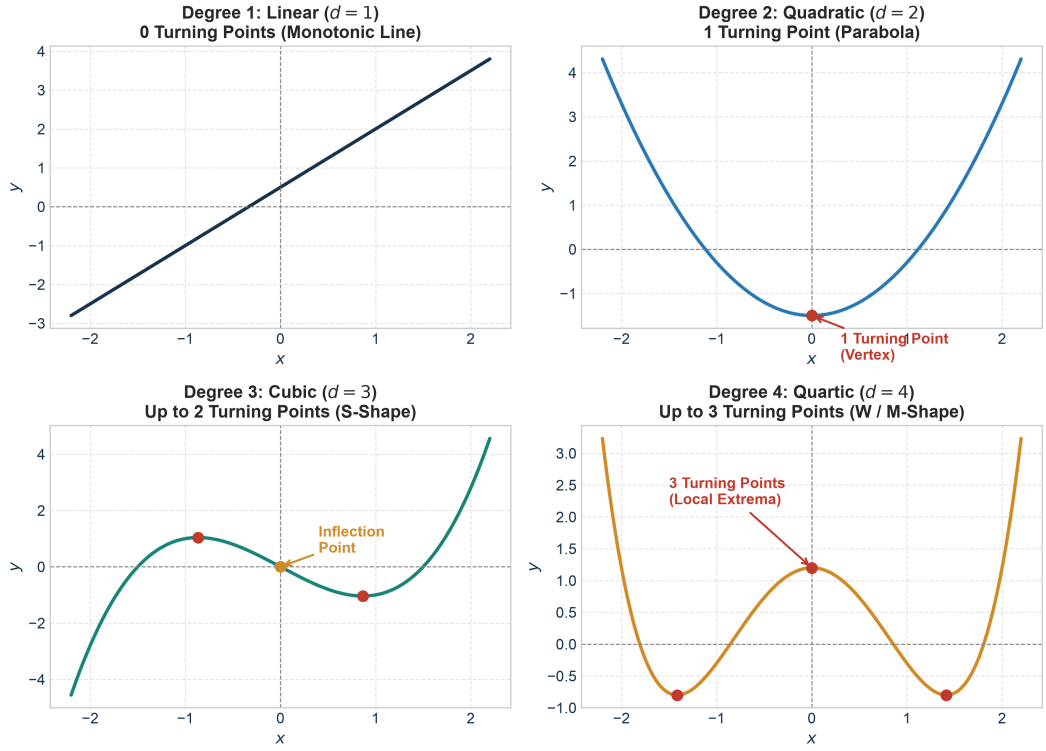

Figure 2: Geometric behavior of polynomial functions across degrees $d \in \{1, 2, 3, 4\}$. Notice that degree d introduces up to $d - 1$ local extrema and inflection points.

1.3 The Crucial Distinction: Linear in Parameters, Nonlinear in Features

A common point of confusion for beginners is why Polynomial Regression is classified as a *linear* regression technique when its graph is clearly a curve.

The Fundamental Definition of Linearity in Machine Learning

A regression model is defined as **linear** if it is linear in its **parameters (coefficients)** $\boldsymbol{\beta}$, regardless of how nonlinear the transformations of the input features $\mathbf{x}$ may be!

Consider the quadratic model:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon. \tag{1.3}$$

Let us define new synthesized variables:

$$z_1 = x, \quad z_2 = x^2. \tag{1.4}$$

Rewriting the equation with these transformed features yields:

$$y = \beta_0 + \beta_1 z_1 + \beta_2 z_2 + \varepsilon. \tag{1.5}$$

This is a standard multiple linear regression model in two variables z_1 and z_2. Because no coefficient is multiplied by another coefficient, raised to a power, or nested inside a nonlinear function (such as $e^{\beta_1 x}$ or $\sin(\beta_1 x)$), the loss function remains quadratic and convex with respect to $\boldsymbol{\beta}$. Consequently, all analytical properties and algorithms of linear regression—including the OLS closed-form normal equations—apply directly.

1.4 Comparison Table: Linear vs. Polynomial vs. Non-Parametric Models

Property	Simple Linear Regression	Polynomial Regression	Non-Parametric (e.g., Splines)
Hypothesis Class	Straight lines ($d = 1$)	Global polynomials of degree d	Piecewise polynomials / kernels
Linear in Parameters?	Yes	Yes	Varies by model
Optimization	Closed-form OLS	Closed-form OLS	Convex / constrained quadratic
Feature Space	$\mathbb{R}^1 \to \mathbb{R}^1$	$\mathbb{R}^1 \to \mathbb{R}^d$ (powers)	Infinite or localized basis
Boundary Behavior	Stable extrapolation	Highly unstable (Runge's effect)	Controlled boundary constraints
Computational Cost	$O(n)$	$O(nd^2 + d^3)$	Moderate to high

2 Mathematical Formulation & Normal Equations

2.1 The Univariate Polynomial Regression Model

Given a dataset of n paired observations $\{(x_i, y_i)\}_{i=1}^n$, a univariate polynomial regression model of degree d expresses the target y_i as:

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d + \varepsilon_i, \quad i = 1, \ldots, n, \tag{2.1}$$

where:

- β_0 is the intercept parameter (expected value of y when $x = 0$).
- $\beta_1, \beta_2, \ldots, \beta_d$ are the polynomial regression coefficients.

- ε_i is the unobserved random error term satisfying Gauss-Markov assumptions: $\mathrm{E}[\varepsilon_i] = 0$, $\mathrm{Var}(\varepsilon_i) = \sigma^2$, and $\mathrm{Cov}(\varepsilon_i, \varepsilon_k) = 0$ for $i \neq k$.

2.2 The Vandermonde Design Matrix Representation

To solve for the optimal coefficients simultaneously, we assemble all n equations into compact matrix-vector notation:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{2.2}$$

where:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^{n\times 1}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_d \end{bmatrix} \in \mathbb{R}^{(d+1)\times 1}, \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \in \mathbb{R}^{n\times 1}. \tag{2.3}$$

The matrix $\mathbf{X} \in \mathbb{R}^{n\times(d+1)}$ is the celebrated **Vandermonde Design Matrix**:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^d \\ 1 & x_2 & x_2^2 & \dots & x_2^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^d \end{bmatrix}. \tag{2.4}$$

Each row corresponds to an individual sample whose original scalar value x_i has been mapped into a $(d+1)$-dimensional feature vector:

$$\phi(x_i) = \begin{bmatrix} 1 & x_i & x_i^2 & \dots & x_i^d \end{bmatrix} \in \mathbb{R}^{1\times(d+1)}. \tag{2.5}$$

2.3 First-Principles Derivation of the OLS Estimator

We seek the optimal coefficient vector $\widehat{\boldsymbol{\beta}}$ that minimizes the Residual Sum of Squares (RSS):

$$\mathrm{RSS}(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{d} \beta_j x_i^j \right)^2. \tag{2.6}$$

In vector notation, the residual vector is $\mathbf{e} = \mathbf{y} - \mathbf{X}\boldsymbol{\beta}$, so the objective function is:

$$\begin{aligned} J(\boldsymbol{\beta}) &= \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \\ &= (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \\ &= \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\boldsymbol{\beta} - \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{y} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}. \end{aligned} \tag{2.7}$$

Because $\mathbf{y}^T\mathbf{X}\boldsymbol{\beta}$ is a scalar, it equals its transpose $\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{y}$. Hence:

$$J(\boldsymbol{\beta}) = \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}. \tag{2.8}$$

To find the minimizer, we compute the gradient with respect to $\boldsymbol{\beta}$:

$$\nabla_\beta J(\boldsymbol{\beta}) = \frac{\partial}{\partial \boldsymbol{\beta}} \left[\mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} \right] = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}. \tag{2.9}$$

Setting the gradient vector equal to zero:

$$-2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \mathbf{0} \implies \boxed{\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^T\mathbf{y}.} \tag{2.10}$$

Equation (2.10) is the system of **Normal Equations**. Provided that the columns of $\mathbf{X}$ are linearly independent (which requires at least $d+1$ distinct x_i values and $n \geq d+1$), the Gram matrix $\mathbf{X}^T\mathbf{X}$ is positive-definite and invertible. Multiplying by $(\mathbf{X}^T\mathbf{X})^{-1}$:

$$\boxed{\widehat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.} \tag{2.11}$$

Geometric Projection in Polynomial Space

The vector of fitted values $\hat{\mathbf{y}} = \mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}$ is the orthogonal projection of the observed data vector $\mathbf{y} \in \mathbb{R}^n$ onto the column space $\text{col}(\mathbf{X}) \subset \mathbb{R}^n$, where $\mathbf{H}$ is the hat matrix. The residual vector $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}}$ is strictly orthogonal to every polynomial basis column: $\mathbf{X}^T\mathbf{e} = \mathbf{0}$.

3 Model Fitting: A Complete Worked Numerical Example

To master the matrix mechanics of polynomial regression, we now work through a complete, unrounded numerical calculation from scratch.

3.1 The Dataset and Problem Definition

Suppose an educational researcher investigates the relationship between the number of hours students study (X) and their resulting examination score (Y). We collect $n = 4$ observations:

Student index (i)	1	2	3	4
Study Hours (x_i)	0	1	2	3
Exam Score (y_i)	2	5	6	12

We wish to fit a degree-2 quadratic polynomial:

$$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x^2. \tag{3.1}$$

3.2 Step-by-Step Construction of the Design Matrix

Our feature space requires three columns: the intercept ($x^0 = 1$), the linear term ($x^1 = x$), and the quadratic term (x^2).

Sample i	x_i	y_i	Basis: $[1, x_i, x_i^2]$
1	0	2	$[1, 0, 0]$
2	1	5	$[1, 1, 1]$
3	2	6	$[1, 2, 4]$
4	3	12	$[1, 3, 9]$

Assembling the matrices:

$$\mathbf{X} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{bmatrix} \in \mathbb{R}^{4\times 3}, \quad \mathbf{y} = \begin{bmatrix} 2 \\ 5 \\ 6 \\ 12 \end{bmatrix} \in \mathbb{R}^{4\times 1}. \tag{3.2}$$

3.3 Computing the Gram Matrix $\mathbf{X}^T\mathbf{X}$

The transpose $\mathbf{X}^T \in \mathbb{R}^{3\times 4}$ is:

$$\mathbf{X}^T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 4 & 9 \end{bmatrix}. \tag{3.3}$$

Multiplying $\mathbf{X}^T$ by $\mathbf{X}$:

$$\mathbf{X}^T\mathbf{X} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 4 & 9 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{bmatrix}. \tag{3.4}$$

Evaluating each element individually:

$$\begin{aligned}
[\mathbf{X}^T\mathbf{X}]_{11} &= 1(1) + 1(1) + 1(1) + 1(1) = 4, \\
[\mathbf{X}^T\mathbf{X}]_{12} &= 1(0) + 1(1) + 1(2) + 1(3) = 6, \\
[\mathbf{X}^T\mathbf{X}]_{13} &= 1(0) + 1(1) + 1(4) + 1(9) = 14, \\
[\mathbf{X}^T\mathbf{X}]_{22} &= 0(0) + 1(1) + 2(2) + 3(3) = 14, \\
[\mathbf{X}^T\mathbf{X}]_{23} &= 0(0) + 1(1) + 2(4) + 3(9) = 36, \\
[\mathbf{X}^T\mathbf{X}]_{33} &= 0(0) + 1(1) + 4(4) + 9(9) = 1 + 16 + 81 = 98.
\end{aligned}$$

By symmetry, $[\mathbf{X}^T\mathbf{X}]_{21} = 6$, $[\mathbf{X}^T\mathbf{X}]_{31} = 14$, and $[\mathbf{X}^T\mathbf{X}]_{32} = 36$. Thus:

$$\boxed{\mathbf{X}^T\mathbf{X} = \begin{bmatrix} 4 & 6 & 14 \\ 6 & 14 & 36 \\ 14 & 36 & 98 \end{bmatrix}}. \tag{3.5}$$

3.4 Computing the Moment Vector $\mathbf{X}^T\mathbf{y}$

Multiplying $\mathbf{X}^T$ by the target vector $\mathbf{y}$:

$$\mathbf{X}^T\mathbf{y} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 4 & 9 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 6 \\ 12 \end{bmatrix} = \begin{bmatrix} 1(2) + 1(5) + 1(6) + 1(12) \\ 0(2) + 1(5) + 2(6) + 3(12) \\ 0(2) + 1(5) + 4(6) + 9(12) \end{bmatrix} = \boxed{\begin{bmatrix} 25 \\ 53 \\ 137 \end{bmatrix}}. \tag{3.6}$$

3.5 Matrix Inversion and Coefficient Estimation

We invert the 3×3 matrix $\mathbf{X}^T\mathbf{X}$. First, we compute the determinant:

$$\begin{aligned}
\det(\mathbf{X}^T\mathbf{X}) &= 4\left(14 \cdot 98 - 36^2\right) - 6\left(6 \cdot 98 - 36 \cdot 14\right) + 14\left(6 \cdot 36 - 14 \cdot 14\right) \\
&= 4(1372 - 1296) - 6(588 - 504) + 14(216 - 196) \\
&= 4(76) - 6(84) + 14(20) = 304 - 504 + 280 = 80.
\end{aligned} \tag{3.7}$$

Computing the adjugate matrix and dividing by $\det = 80$ yields the exact inverse:

$$(\mathbf{X}^T\mathbf{X})^{-1} = \frac{1}{80}\begin{bmatrix} 76 & -84 & 20 \\ -84 & 196 & -60 \\ 20 & -60 & 20 \end{bmatrix} = \begin{bmatrix} \frac{19}{20} & -\frac{21}{20} & \frac{1}{4} \\ -\frac{21}{20} & \frac{49}{20} & -\frac{3}{4} \\ \frac{1}{4} & -\frac{3}{4} & \frac{1}{4} \end{bmatrix}. \tag{3.8}$$

Now we calculate the coefficient vector $\widehat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$:

$$\widehat{\boldsymbol{\beta}} = \begin{bmatrix} \frac{19}{20} & -\frac{21}{20} & \frac{1}{4} \\ -\frac{21}{20} & \frac{49}{20} & -\frac{3}{4} \\ \frac{1}{4} & -\frac{3}{4} & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 25 \\ 53 \\ 137 \end{bmatrix}. \tag{3.9}$$

Evaluating each entry explicitly:

$$\hat{\beta}_0 = \frac{19(25) - 21(53) + 5(137)}{20} = \frac{475 - 1113 + 685}{20} = \frac{47}{20} = \mathbf{2.35},$$
$$\hat{\beta}_1 = \frac{-21(25) + 49(53) - 15(137)}{20} = \frac{-525 + 2597 - 2055}{20} = \frac{17}{20} = \mathbf{0.85},$$
$$\hat{\beta}_2 = \frac{1(25) - 3(53) + 1(137)}{4} = \frac{25 - 159 + 137}{4} = \frac{3}{4} = \mathbf{0.75}.$$

Hence:

$$\boxed{\widehat{\boldsymbol{\beta}} = \begin{bmatrix} 2.35 \\ 0.85 \\ 0.75 \end{bmatrix}.} \tag{3.10}$$

The optimal fitted polynomial regression equation is:

$$\boxed{\hat{y} = 2.35 + 0.85x + 0.75x^2.} \tag{3.11}$$

3.6 Predictions, Residual Analysis, and RSS

We now calculate the fitted predictions $\hat{y}_i$ and residuals $e_i = y_i - \hat{y}_i$:

- $x_1 = 0 \implies \hat{y}_1 = 2.35 + 0.85(0) + 0.75(0)^2 = 2.35 \implies e_1 = 2 - 2.35 = \mathbf{-0.35}$.
- $x_2 = 1 \implies \hat{y}_2 = 2.35 + 0.85(1) + 0.75(1)^2 = 3.95 \implies e_2 = 5 - 3.95 = \mathbf{+1.05}$.
- $x_3 = 2 \implies \hat{y}_3 = 2.35 + 0.85(2) + 0.75(2)^2 = 7.05 \implies e_3 = 6 - 7.05 = \mathbf{-1.05}$.
- $x_4 = 3 \implies \hat{y}_4 = 2.35 + 0.85(3) + 0.75(3)^2 = 11.65 \implies e_4 = 12 - 11.65 = \mathbf{+0.35}$.

x_i	Actual y_i	Predicted $\hat{y}_i$	Residual $e_i = y_i - \hat{y}_i$	Squared Residual e_i^2
0	2	2.35	-0.35	0.1225
1	5	3.95	$+1.05$	1.1025
2	6	7.05	-1.05	1.1025
3	12	11.65	$+0.35$	0.1225
Sum of Residuals $\sum e_i = \mathbf{0.00}$			**RSS $= \sum e_i^2 = \mathbf{2.4500}$**	

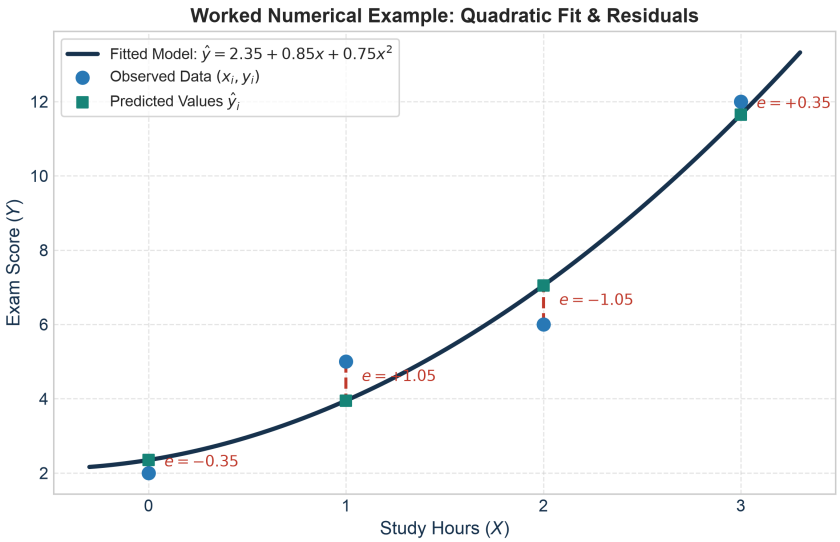

Figure 3: Fitted quadratic model $\hat{y} = 2.35 + 0.85x + 0.75x^2$ for the 4-point worked dataset, illustrating observed points, predicted points, and the four calculated residuals.

Verification of OLS Normal Equations

Notice two critical mathematical properties of OLS confirmed by this calculation:

1. **Zero sum of residuals:** $\sum_{i=1}^{4} e_i = -0.35 + 1.05 - 1.05 + 0.35 = 0$.
2. **Orthogonality to features:** $\mathbf{X}^T\mathbf{e} = \mathbf{0}$. We can compute this directly:

$$\mathbf{X}^T\mathbf{e} = \begin{bmatrix} \sum e_i \\ \sum x_i e_i \\ \sum x_i^2 e_i \end{bmatrix} = \begin{bmatrix} -0.35 + 1.05 - 1.05 + 0.35 \\ 0(-0.35) + 1(1.05) + 2(-1.05) + 3(0.35) \\ 0(-0.35) + 1(1.05) + 4(-1.05) + 9(0.35) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

4 Multivariate Polynomials & Feature Transformation

4.1 Multivariate Polynomial Expansion

In practical applications, real-world targets depend on multiple predictor variables. For example, predicting residential real estate prices y using both interior living area (X_1) and the number of bedrooms (X_2).

A standard linear model assumes independent, additive linear contributions:

$$\hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2. \tag{4.1}$$

If we expand this to a degree-2 polynomial without interactions, we obtain:

$$\hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2. \tag{4.2}$$

4.2 Interaction Terms: Modeling Synergistic Relationships

Pure power terms (X_1^2, X_2^2) capture curvature along individual variable axes. However, they assume that the effect of X_1 on y is entirely independent of the value of X_2. In reality, an additional 500 square feet of living space has a vastly different market valuation in a 1-bedroom apartment than in a 5-bedroom home.

To capture this interdependency, we introduce the **interaction term** X_1X_2:

$$\boxed{\hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2.} \tag{4.3}$$

Interpreting Interaction Terms via Partial Derivatives

Taking the partial derivative of $\hat{y}$ in Equation (4.3) with respect to X_1:

$$\frac{\partial \hat{y}}{\partial X_1} = \beta_1 + 2\beta_3 X_1 + \beta_5 X_2. \tag{4.4}$$

The marginal rate of change of $\hat{y}$ with respect to X_1 now explicitly depends on the level of X_2! When $\beta_5 \neq 0$, the variables interact synergistically.

4.3 The Combinatorial Explosion of Feature Space

How many total features are generated when applying a degree-d polynomial expansion to p original continuous variables?

Feature Dimension Formula

For p original input variables and a maximum total polynomial degree d, the total number of terms (including the constant intercept 1 and all cross-product interaction terms $\prod_{k=1}^{p} X_k^{d_k}$ with $\sum d_k \leq d$) is given by the combination with repetition formula:

$$D = \binom{p+d}{d} = \frac{(p+d)!}{p!\,d!}. \tag{4.5}$$

Let us examine how quickly this dimension grows:

Original Features (p)	Degree $d=1$	Degree $d=2$	Degree $d=3$	Degree $d=4$	Degree $d=5$
1	2	3	4	5	6
2	3	6	10	15	21
5	6	21	56	126	252
10	11	66	286	1,001	3,003
20	21	231	1,771	10,626	53,130

The Curse of Dimensionality in Polynomial Expansion

With only $p = 10$ input features and degree $d = 5$, the model expands into 3,003 columns. If the training sample size $n < 3{,}003$, the Gram matrix $\mathbf{X}^T\mathbf{X}$ is rank-deficient and singular, making ordinary least squares impossible without regularization.

4.4 Data Leakage Prevention during Transformation

When performing train-test splits, feature transformations must be managed strictly to prevent **data leakage**:

1. **Step 1: Partition first.** Always split raw data into training ($\mathcal{D}_{\text{train}}$) and test ($\mathcal{D}_{\text{test}}$) sets before fitting any transformations.
2. **Step 2: Fit on train only.** Compute transformation parameters (such as mean and variance during feature standardization) exclusively on $\mathcal{D}_{\text{train}}$.

3. **Step 3: Transform test deterministically.** Apply the fitted mapping to $\mathcal{D}_{\text{test}}$ without re-estimating parameters.

5 Bias-Variance Trade-off, Overfitting & Regularization

5.1 Underfitting vs. Optimal Fit vs. Overfitting

The choice of polynomial degree d acts as the direct complexity dial of the model:

- **Underfitting (High Bias, Low Variance, e.g., $d = 1$):** The hypothesis space is overly restrictive. The model cannot capture the true underlying non-linear manifold, resulting in high training error and high generalization error.
- **Optimal Fit (Balanced Bias and Variance, e.g., $d = 3$):** The model captures the true functional signal while smoothing over random noise fluctuations.
- **Overfitting (Low Bias, High Variance, e.g., $d = 12$):** With d excessively large, the model has enough degrees of freedom to interpolate training points exactly (RSS $\to 0$), capturing sample-specific noise.

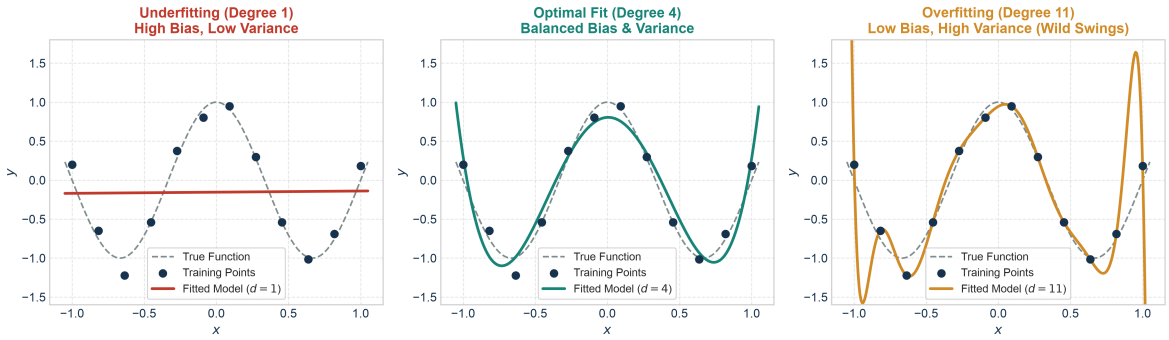

Figure 4: Comparison of polynomial fits on noisy data ($n = 12$). Underfitting ($d = 1$) fails to capture curvature; optimal fit ($d = 4$) mirrors the true underlying cosine signal; overfitting ($d = 11$) passes through every point but oscillates violently between samples.

Runge's Phenomenon and Boundary Oscillations

High-degree polynomial interpolation on equally spaced points is notorious for **Runge's Phenomenon**: the fitted curve develops violent, high-frequency oscillations near the boundaries of the input interval. Predicting outside or even near the edges of the observed training range with high-degree polynomials leads to catastrophic extrapolation errors.

5.2 The Bias-Variance Decomposition

Formally, let the true data-generating process be $y = f(x) + \varepsilon$ with $\mathrm{E}[\varepsilon] = 0$ and $\mathrm{Var}(\varepsilon) = \sigma^2$. For a model $\hat{f}(x)$ trained on dataset $\mathcal{D}$, the expected squared prediction error at a query point x_0 decomposes into three orthogonal components:

$$\mathrm{E}_{\mathcal{D}}\left[\left(y - \hat{f}(x_0)\right)^2\right] = \underbrace{\left(f(x_0) - \mathrm{E}[\hat{f}(x_0)]\right)^2}_{\text{Bias}^2} + \underbrace{\mathrm{E}\left[\left(\hat{f}(x_0) - \mathrm{E}[\hat{f}(x_0)]\right)^2\right]}_{\text{Variance}} + \underbrace{\sigma^2}_{\text{Irreducible Noise}}. \quad (5.1)$$

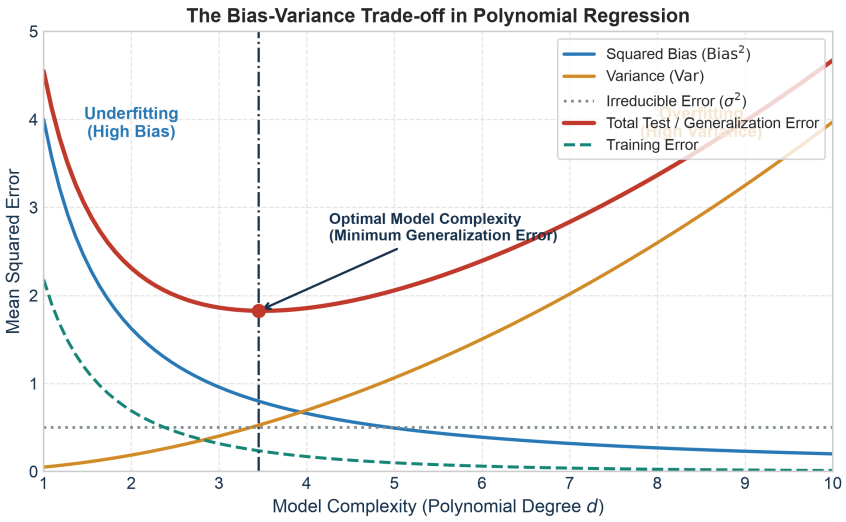

Figure 5: The Bias-Variance Trade-off curve as a function of polynomial degree d. Training error monotonically decreases, but test error forms a U-shaped curve, identifying the optimal model complexity.

As degree d increases:

- Bias^2 decreases monotonically because higher-order polynomials subsume lower-order polynomials.
- Variance increases rapidly because coefficient estimates become hypersensitive to minor shifts in training samples.
- The total generalization error achieves a minimum at the sweet spot where the marginal decrease in squared bias equals the marginal increase in variance.

5.3 Regularization: Taming Complex Polynomials

Rather than manually searching for a single scalar degree d, an elegant modern approach is to allow a flexible polynomial degree (e.g., $d = 8$ or 10) while constraining coefficient magnitudes via **regularization**.

5.3.1 Ridge Regression (L_2 Penalty) with Polynomial Features

Ridge regression modifies the OLS objective by adding an L_2 shrinkage penalty:

$$J_{\text{Ridge}}(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \phi(x_i)\boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{d} \beta_j^2, \quad \lambda \geq 0. \tag{5.2}$$

Notice that the intercept β_0 is unpenalized. In matrix form:

$$\boxed{\widehat{\boldsymbol{\beta}}_{\text{Ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{P})^{-1}\mathbf{X}^T\mathbf{y},} \tag{5.3}$$

where $\mathbf{P} = \text{diag}(0, 1, 1, \ldots, 1) \in \mathbb{R}^{(d+1)\times(d+1)}$. The penalty adds λ to the diagonal of the feature Gram matrix, ensuring invertibility and shrinking high-frequency oscillations.

5.3.2 Lasso Regression (L_1 Penalty) for Feature Selection

Lasso replaces the squared penalty with an L_1 absolute value norm:

$$J_{\text{Lasso}}(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \phi(x_i)\boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{d} |\beta_j| \,. \tag{5.4}$$

Due to the geometry of the L_1 ball, Lasso forces unneeded higher-order coefficients to become **exactly zero**, automatically performing polynomial degree and interaction pruning.

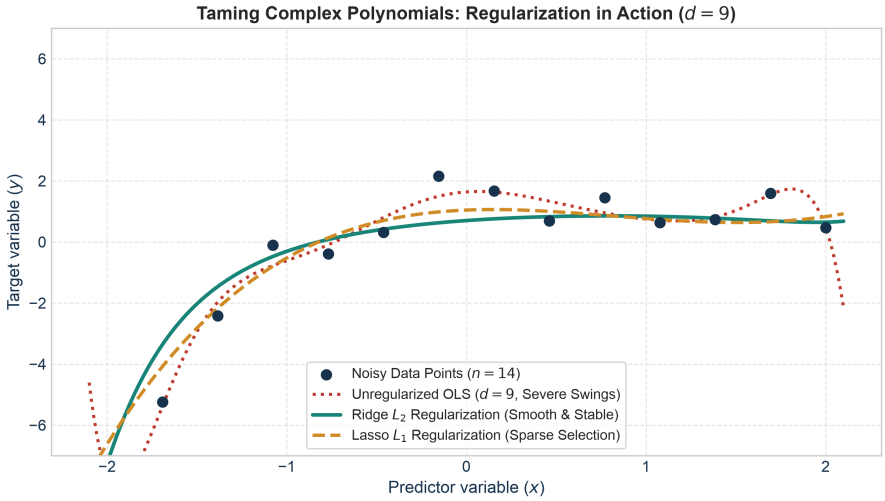

Figure 6: Comparison of high-degree ($d = 9$) polynomial fits with and without regularization. Unregularized OLS swings violently; Ridge (L_2) produces a smooth, stable curve; Lasso (L_1) prunes non-essential higher powers.

5.4 The Absolute Necessity of Feature Scaling

When we compute powers of a feature (e.g., $x, x^2, x^3, \ldots, x^d$), the numerical scales diverge exponentially: If $x \in [1, 100]$, then $x^2 \in [1, 10{,}000]$ and $x^4 \in [1, 10^8]$.

- In unregularized OLS, severe numerical roundoff and condition numbers $\kappa(\mathbf{X}^T\mathbf{X}) \gg 10^{15}$ cause total loss of arithmetic precision.
- In regularized regression (Ridge and Lasso), the uniform penalty $\lambda \sum \beta_j^2$ disproportionately crushes coefficients of higher-power terms simply because their unscaled features are huge.

Therefore, **standardization (z-score scaling) after polynomial expansion is mandatory**:

$$z_j = \frac{x^j - \mu_j}{\sigma_j}. \tag{5.5}$$

6 Revision Sheet & Self-Test

6.1 The Essential Formulas

$$\begin{aligned}
\text{Univariate Model:} &\quad y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d + \varepsilon \\
\text{Vandermonde Matrix:} &\quad \mathbf{X} = \begin{bmatrix} 1 & x_i & x_i^2 & \dots & x_i^d \end{bmatrix}_{i=1}^{n} \in \mathbb{R}^{n \times (d+1)} \\
\text{Matrix OLS Formulation:} &\quad \mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \\
\text{Normal Equations:} &\quad \mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^T\mathbf{y} \\
\text{OLS Solution:} &\quad \widehat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\
\text{Bivariate Model with Interaction:} &\quad \hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 \\
\text{Total Features Formula:} &\quad D = \binom{p+d}{d} = \frac{(p+d)!}{p!\,d!} \\
\text{Ridge Regularized Solution:} &\quad \widehat{\boldsymbol{\beta}}_{\text{Ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{P})^{-1}\mathbf{X}^T\mathbf{y} \\
\text{Bias-Variance Decomposition:} &\quad \mathrm{E}[(y - \hat{f})^2] = \mathrm{Bias}^2 + \mathrm{Var}(\hat{f}) + \sigma^2
\end{aligned}$$

6.2 Conceptual Checklist

- **Why Linear:** Polynomial regression is linear in parameters $\boldsymbol{\beta}$, allowing ordinary convex quadratic optimization via standard OLS normal equations.
- **Curvature vs. Degree:** A polynomial of degree d can exhibit at most $d-1$ turning points.
- **Interaction Terms:** Cross-product terms X_1X_2 allow the marginal sensitivity $\frac{\partial \hat{y}}{\partial X_1}$ to depend on the magnitude of X_2.
- **Extrapolation Danger:** High-degree polynomials suffer from Runge's phenomenon at boundaries; never use them for long-range forecasting.
- **Feature Scaling:** High powers create extreme disparities in column magnitude, mandating z-score scaling when regularizing.

6.3 Self-Test Questions

1. Explain why the model $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$ is a linear regression model, whereas $y = \beta_0 e^{\beta_1 x} + \varepsilon$ is not.
2. If an input dataset has $p = 4$ features and we apply a polynomial expansion of degree $d = 3$, how many total features (including the intercept) are generated?
3. What happens to the condition number of the Gram matrix $\mathbf{X}^T\mathbf{X}$ as polynomial degree d increases on unscaled features, and how does this affect numerical stability?
4. Why does an unregularized polynomial model of degree $d = n - 1$ always achieve RSS $= 0$ on a dataset with n distinct points, and why is this problematic for generalization?
5. How do Ridge and Lasso regularization differ when applied to a high-degree polynomial feature matrix?

6.4 Short Answers

1. The first model is linear with respect to its unknown parameters $\beta_0, \beta_1, \beta_2$. The partial derivatives $\frac{\partial \hat{y}}{\partial \beta_j}$ do not depend on any β. In contrast, in the exponential model, $\frac{\partial \hat{y}}{\partial \beta_1} = \beta_0 x e^{\beta_1 x}$, which depends nonlinearly on β_1 and β_0.
2. Using the formula $D = \binom{p+d}{d} = \binom{4+3}{3} = \binom{7}{3} = \frac{7\times6\times5}{3\times2\times1} = \mathbf{35}$ total features.

3. The condition number $\kappa(\mathbf{X}^T\mathbf{X})$ explodes rapidly toward infinity because columns x^k and x^{k+1} are highly collinear and their magnitudes differ by orders of magnitude. This makes the matrix ill-conditioned, causing extreme numerical errors during matrix inversion.

4. By the Fundamental Theorem of Algebra, any n distinct points with unique x_i can be uniquely interpolated by an $(n-1)$-th degree polynomial. While RSS $= 0$ on training data, the function oscillates wildly between points (overfitting / high variance), leading to massive test errors.

5. Ridge (L_2) shrinks all polynomial coefficients toward zero smoothly, dampening high-frequency oscillations while keeping all powers active. Lasso (L_1) shrinks coefficients and sets non-informative powers to exact zeros, performing automatic polynomial term selection.

Final Mental Model

Polynomial regression is simply multiple linear regression performed on a nonlinearly transformed basis. Treat the polynomial degree d as a hyperparameter controlling capacity: keep d modest, always standardize features, and combine with Ridge or Lasso regularization to harvest curvature without succumbing to runaway variance.