Signal Processing and Polynomial Estimation: From One-Dimensional to Two-Dimensional Analysis

Lecture Notes

July 16, 2025

Abstract

These notes present a comprehensive treatment of signal processing techniques with particular emphasis on polynomial estimation methods for both one-dimensional and two-dimensional signals. We develop the theoretical framework for weighted least squares approximation, derive the fundamental algorithms for noise reduction, and extend the methodology to higher-dimensional cases. The exposition includes detailed mathematical derivations, algorithmic implementations, and practical applications in signal processing and image analysis.

Contents

1 Introduction and Motivation

The fundamental problem in signal processing involves extracting meaningful information from noisy observations. Consider a continuous signal $s(t)$ corrupted by additive noise $n(t)$, yielding the observed signal:

$$y(t) = s(t) + n(t) \tag{1}$$

The objective is to estimate $s(t)$ from the observations $y(t)$ while minimizing the impact of noise. This estimation problem becomes particularly challenging when dealing with discrete observations at finite sampling points.

Problem Statement Given a set of observations $\{y_i\}_{i=1}^{N}$ at points $\{t_i\}_{i=1}^{N}$, we seek to construct an estimator $\hat{s}(t)$ that approximates the underlying signal $s(t)$ with minimal estimation error.

The polynomial approximation approach provides a robust framework for this estimation problem by representing the signal as a linear combination of basis functions. This methodology extends naturally to higher-dimensional signals, such as images, where two-dimensional polynomial surfaces can effectively model smooth variations in pixel intensities.

1.1 Historical Context and Applications

Polynomial estimation techniques have found widespread applications in:

- **Image Processing**: Smoothing and noise reduction in digital images
- **Signal Reconstruction**: Interpolation and extrapolation of sampled signals
- **Data Analysis**: Trend estimation in time series data
- **Computer Vision**: Surface fitting for 3D reconstruction

2 One-Dimensional Signal Estimation

2.1 Fundamental Setup and Notation

Consider a one-dimensional signal observed at N discrete points. Let $\{x_i\}_{i=1}^N$ denote the sampling locations and $\{y_i\}_{i=1}^N$ the corresponding observed values. Without loss of generality, we can normalize the domain to the interval $[-1, 1]$.

Definition 2.1 (Normalized Domain). *For computational efficiency and numerical stability, we map the original domain* $[a, b]$ *to the normalized interval* $[-1, 1]$ *using the transformation:*

$$x_{norm} = \frac{2x - (a + b)}{b - a} \tag{2}$$

2.2 Uniform Distribution of Sampling Points

For optimal polynomial approximation, sampling points should be uniformly distributed across the domain. This ensures that the polynomial basis functions are well-conditioned and that the approximation error is evenly distributed.

Theorem 2.2 (Optimal Sampling Distribution). *For polynomial approximation of degree d, the optimal sampling points in* $[-1, 1]$ *are given by:*

$$x_i = -1 + \frac{2(i-1)}{N-1}, \quad i = 1, 2, \ldots, N \tag{3}$$

where $N > d$ to ensure an overdetermined system.

2.3 Weighted Least Squares Formulation

The core methodology employs weighted least squares to account for varying confidence levels in different observations. Let $\mathbf{w} = (w_1, w_2, \ldots, w_N)^T$ represent the weight vector, where $w_i > 0$ indicates the reliability of the i-th observation.

Definition 2.3 (Weight Matrix). *The weight matrix* $\mathbf{W} \in \mathbb{R}^{N \times N}$ *is defined as:*

$$\mathbf{W} = \text{diag}(w_1, w_2, \ldots, w_N) \tag{4}$$

The weighted least squares objective function takes the form:

$$J(\mathbf{c}) = \sum_{i=1}^{N} w_i \left(y_i - \sum_{j=0}^{d} c_j x_i^j \right)^2 \tag{5}$$

where $\mathbf{c} = (c_0, c_1, \ldots, c_d)^T$ represents the polynomial coefficients.

2.4 Matrix Formulation and Solution

2.4.1 Vandermonde Matrix Construction

The polynomial evaluation at all sampling points can be expressed in matrix form using the Vandermonde matrix:

$$\mathbf{V} = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^d \\ 1 & x_2 & x_2^2 & \cdots & x_2^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^d \end{pmatrix} \tag{6}$$

2.4.2 Normal Equations Derivation

The weighted least squares solution is obtained by minimizing the objective function (5). Taking the derivative with respect to $\mathbf{c}$ and setting it to zero:

$$\frac{\partial J}{\partial \mathbf{c}} = -2\mathbf{V}^T\mathbf{W}(\mathbf{y} - \mathbf{V}\mathbf{c}) = 0 \tag{7}$$

Rearranging terms:

$$\mathbf{V}^T\mathbf{W}\mathbf{V}\mathbf{c} = \mathbf{V}^T\mathbf{W}\mathbf{y} \tag{8}$$

This yields the normal equations:

$$\boxed{\mathbf{A}\mathbf{c} = \mathbf{b}} \tag{9}$$

where:

$$\mathbf{A} = \mathbf{V}^T\mathbf{W}\mathbf{V} \tag{10}$$

$$\mathbf{b} = \mathbf{V}^T\mathbf{W}\mathbf{y} \tag{11}$$

Theorem 2.4 (Existence and Uniqueness of Solution)**.** *If* $\mathbf{V}$ *has full column rank (i.e.,* $\operatorname{rank}(\mathbf{V}) = d + 1$*) and* $\mathbf{W}$ *is positive definite, then the system* (9) *has a unique solution:*

$$\mathbf{c} = (\mathbf{V}^T\mathbf{W}\mathbf{V})^{-1}\mathbf{V}^T\mathbf{W}\mathbf{y} \tag{12}$$

Proof. The matrix $\mathbf{A} = \mathbf{V}^T\mathbf{W}\mathbf{V}$ is positive definite since:

$$\mathbf{x}^T\mathbf{A}\mathbf{x} = \mathbf{x}^T\mathbf{V}^T\mathbf{W}\mathbf{V}\mathbf{x} \tag{13}$$

$$= (\mathbf{V}\mathbf{x})^T\mathbf{W}(\mathbf{V}\mathbf{x}) \tag{14}$$

$$= \sum_{i=1}^{N} w_i(\mathbf{V}\mathbf{x})_i^2 > 0 \tag{15}$$

for any $\mathbf{x} \neq \mathbf{0}$, provided $\mathbf{V}$ has full column rank and $w_i > 0$ for all i. □

2.5 Signal Reconstruction and Smoothing

2.5.1 Convolution Interpretation

The estimated signal at any point x can be expressed as:

$$\hat{s}(x) = \sum_{j=0}^{d} c_j x^j = \mathbf{v}(x)^T \mathbf{c} \tag{16}$$

where $\mathbf{v}(x) = (1, x, x^2, \ldots, x^d)^T$ is the polynomial basis vector.
Substituting the solution from Theorem 2.4:

$$\hat{s}(x) = \mathbf{v}(x)^T (\mathbf{V}^T \mathbf{W} \mathbf{V})^{-1} \mathbf{V}^T \mathbf{W} \mathbf{y} \tag{17}$$

This can be rewritten as a weighted combination of observations:

$$\hat{s}(x) = \sum_{i=1}^{N} h_i(x) y_i \tag{18}$$

where the smoothing weights are:

$$h_i(x) = \mathbf{v}(x)^T (\mathbf{V}^T \mathbf{W} \mathbf{V})^{-1} \mathbf{V}^T \mathbf{W} \mathbf{e}_i \tag{19}$$

2.5.2 Center Point Estimation

For practical applications, we often focus on estimating the signal at the center of the domain ($x = 0$). The center point estimate is:

$$\hat{s}(0) = \mathbf{v}(0)^T \mathbf{c} = c_0 \tag{20}$$

This corresponds to extracting the constant term from the polynomial fit, which represents the smoothed value at the center point.

3 Extension to Two-Dimensional Signals

3.1 Problem Formulation in 2D

Consider a two-dimensional signal $s(u, v)$ observed at discrete grid points (u_i, v_j) with observations y_{ij}. The extension to 2D requires:

- Bivariate polynomial basis functions
- Two-dimensional weight matrices
- Grid-based sampling strategies

Definition 3.1 (2D Signal Domain)**.** *Let* $\Omega = [-1, 1] \times [-1, 1]$ *denote the normalized 2D domain. The observed signal is given by:*

$$y_{ij} = s(u_i, v_j) + n_{ij} \tag{21}$$

where n_{ij} *represents additive noise.*

3.2 Bivariate Polynomial Basis

For polynomial degree d, the bivariate polynomial basis consists of all monomials $u^k v^l$ with $k + l \leq d$:

$$\mathcal{B}_d = \{u^k v^l : k, l \geq 0, k + l \leq d\} \tag{22}$$

The total number of basis functions is:

$$M = \binom{d+2}{2} = \frac{(d+1)(d+2)}{2} \tag{23}$$

3.2.1 Ordering of Basis Functions

A systematic ordering of the basis functions is essential for matrix construction. We adopt the graded lexicographic ordering:

Example 3.2 (Basis Functions for $d = 2$)**.** *For quadratic polynomials ($d = 2$), the basis functions are:*

$$\phi_1(u, v) = 1 \tag{24}$$
$$\phi_2(u, v) = u \tag{25}$$
$$\phi_3(u, v) = v \tag{26}$$
$$\phi_4(u, v) = u^2 \tag{27}$$
$$\phi_5(u, v) = uv \tag{28}$$
$$\phi_6(u, v) = v^2 \tag{29}$$

3.3 Matrix Construction in 2D

3.3.1 Design Matrix Assembly

Let (u_i, v_j) denote the grid points for $i = 1, \ldots, N_u$ and $j = 1, \ldots, N_v$. The design matrix $\mathbf{X} \in \mathbb{R}^{N \times M}$ (where $N = N_u \times N_v$) has rows corresponding to grid points and columns corresponding to basis functions:

$$\mathbf{X}_{(i-1)N_v+j,k} = \phi_k(u_i, v_j) \tag{30}$$

3.3.2 Vectorization of Observations

The 2D observation matrix $\mathbf{Y} \in \mathbb{R}^{N_u \times N_v}$ is vectorized as:

$$\mathbf{y} = \text{vec}(\mathbf{Y}) = \begin{pmatrix} y_{11} \\ y_{21} \\ \vdots \\ y_{N_u 1} \\ y_{12} \\ \vdots \\ y_{N_u N_v} \end{pmatrix} \tag{31}$$

3.4 2D Weighted Least Squares

The 2D weighted least squares problem becomes:

$$\min_{\mathbf{c}} \|\mathbf{W}^{1/2}(\mathbf{y} - \mathbf{X}\mathbf{c})\|_2^2 \tag{32}$$

where $\mathbf{W} \in \mathbb{R}^{N \times N}$ is the weight matrix.

Theorem 3.3 (2D Normal Equations)**.** *The solution to the 2D weighted least squares problem is given by:*

$$\mathbf{c} = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{y} \tag{33}$$

provided that $\mathbf{X}$ *has full column rank.*

3.5 Surface Reconstruction

The estimated 2D signal is reconstructed as:

$$\hat{s}(u, v) = \sum_{k=1}^{M} c_k \phi_k(u, v) \tag{34}$$

For center point estimation, we evaluate:

$$\hat{s}(0, 0) = c_1 \tag{35}$$

which corresponds to the constant term in the polynomial expansion.

4 Numerical Considerations and Implementation

4.1 Conditioning and Stability

The numerical stability of polynomial fitting depends critically on the conditioning of the normal equations matrix $\mathbf{A} = \mathbf{V}^T\mathbf{W}\mathbf{V}$.

Definition 4.1 (Condition Number)**.** *The condition number of a matrix* $\mathbf{A}$ *is defined as:*

$$\kappa(\mathbf{A}) = \frac{\sigma_{\max}(\mathbf{A})}{\sigma_{\min}(\mathbf{A})} \tag{36}$$

where $\sigma_{\max}$ *and* $\sigma_{\min}$ *are the largest and smallest singular values, respectively.*

Remark 4.2 (Conditioning Issues)**.** *For high-degree polynomials, the condition number grows exponentially, leading to numerical instability. This motivates the use of:*

- *Orthogonal polynomials (Chebyshev, Legendre)*
- *Regularization techniques*
- *Domain normalization*

4.2 Alternative Solution Methods

4.2.1 QR Decomposition

For improved numerical stability, the normal equations can be solved using QR decomposition:

$$\mathbf{W}^{1/2}\mathbf{V} = \mathbf{Q}\mathbf{R} \tag{37}$$
$$\mathbf{R}\mathbf{c} = \mathbf{Q}^T\mathbf{W}^{1/2}\mathbf{y} \tag{38}$$

4.2.2 Singular Value Decomposition

For rank-deficient or ill-conditioned systems, SVD provides a robust solution:

$$\mathbf{V} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{39}$$

The pseudoinverse solution is:

$$\mathbf{c} = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T\mathbf{W}\mathbf{y} \tag{40}$$

where $\mathbf{\Sigma}^\dagger$ is the pseudoinverse of $\mathbf{\Sigma}$.

4.3 Computational Complexity

Theorem 4.3 (Complexity Analysis)**.** *The computational complexity of the polynomial fitting algorithm is:*

- *Matrix assembly:* $O(N \cdot M)$
- *Normal equations formation:* $O(N \cdot M^2)$
- *System solution:* $O(M^3)$

where N *is the number of data points and* M *is the number of basis functions.*

5 Applications and Examples

5.1 Image Denoising Application

Consider a noisy image corrupted by additive Gaussian noise. The polynomial fitting approach can be applied locally to each pixel neighborhood for effective denoising.

Example 5.1 (Local Polynomial Denoising)**.** *For each pixel* (i, j)*, consider a* $(2k + 1) \times (2k + 1)$ *neighborhood. Apply 2D polynomial fitting of degree* d *to estimate the clean pixel value:*

$$\hat{I}(i, j) = \textit{polyfit2d}(\{I(i + r, j + s) : -k \leq r, s \leq k\}, d) \tag{41}$$

5.2 Signal Interpolation

Polynomial fitting provides a natural framework for signal interpolation between sample points.

---

1. Fit polynomial of degree d to observed data points
2. Evaluate polynomial at desired interpolation points
3. Assess interpolation quality using cross-validation

5.3 Trend Analysis

In time series analysis, polynomial fitting reveals underlying trends by removing high-frequency noise components.

$$\text{Trend}(t) = \sum_{k=0}^{d} c_k t^k \tag{42}$$

The residual signal $\text{Residual}(t) = y(t) - \text{Trend}(t)$ contains the detrended information.

6 Advanced Topics and Extensions

6.1 Adaptive Polynomial Degree Selection

The choice of polynomial degree d significantly impacts the bias-variance tradeoff. Several criteria can guide this selection:

6.1.1 Cross-Validation

K-fold cross-validation provides an empirical method for degree selection:

$$\mathrm{CV}(d) = \frac{1}{K}\sum_{k=1}^{K} \|\mathbf{y}^{(k)} - \mathbf{X}^{(k)}\hat{\mathbf{c}}^{(-k)}\|_2^2 \tag{43}$$

where $\hat{\mathbf{c}}^{(-k)}$ is the coefficient vector estimated without the k-th fold.

6.1.2 Information Criteria

The Akaike Information Criterion (AIC) balances model fit and complexity:

$$\mathrm{AIC}(d) = N\log(\mathrm{RSS}(d)) + 2M \tag{44}$$

where $\mathrm{RSS}(d)$ is the residual sum of squares and M is the number of parameters.

6.2 Regularization Techniques

6.2.1 Ridge Regression

Ridge regression adds an L_2 penalty to prevent overfitting:

$$J_{\mathrm{ridge}}(\mathbf{c}) = \|\mathbf{W}^{1/2}(\mathbf{y} - \mathbf{X}\mathbf{c})\|_2^2 + \lambda\|\mathbf{c}\|_2^2 \tag{45}$$

The solution becomes:

$$\mathbf{c}_{\mathrm{ridge}} = (\mathbf{X}^T\mathbf{W}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{y} \tag{46}$$

6.2.2 LASSO Regression

LASSO promotes sparsity through L_1 regularization:

$$J_{\mathrm{lasso}}(\mathbf{c}) = \|\mathbf{W}^{1/2}(\mathbf{y} - \mathbf{X}\mathbf{c})\|_2^2 + \lambda\|\mathbf{c}\|_1 \tag{47}$$

6.3 Multivariate Extensions

The methodology extends naturally to higher dimensions:

$$s(x_1, x_2, \ldots, x_p) = \sum_{|\alpha|\le d} c_\alpha \prod_{i=1}^{p} x_i^{\alpha_i} \tag{48}$$

where $\alpha = (\alpha_1, \ldots, \alpha_p)$ is a multi-index and $|\alpha| = \sum_{i=1}^{p}\alpha_i$.

7 Conclusion and Future Directions

7.1 Summary of Key Results

This exposition has developed a comprehensive framework for polynomial-based signal estimation in both one and two dimensions. The key contributions include:

- Rigorous derivation of weighted least squares formulations
- Extension to multidimensional signals with bivariate polynomials
- Numerical stability considerations and alternative solution methods
- Practical applications in signal processing and image analysis

7.2 Theoretical Insights

The polynomial approximation approach provides several theoretical advantages:

1. **Universality**: Polynomials can approximate any continuous function on a compact domain (Weierstrass theorem)
2. **Linearity**: The estimation problem reduces to linear algebra
3. **Interpretability**: Polynomial coefficients have clear geometric meaning
4. **Efficiency**: Fast algorithms exist for evaluation and fitting

7.3 Future Research Directions

Several avenues merit further investigation:

- **Adaptive Methods**: Locally adaptive polynomial degrees based on signal characteristics
- **Robust Estimation**: Techniques for handling outliers and non-Gaussian noise
- **Sparse Representations**: Combining polynomial fitting with sparsity constraints
- **Real-time Processing**: Efficient algorithms for streaming data applications

7.4 Practical Recommendations

For practitioners implementing these methods:

1. Always normalize the domain to $[-1, 1]$ for numerical stability
2. Use cross-validation for polynomial degree selection
3. Consider regularization for high-degree polynomials
4. Implement QR decomposition for improved numerical accuracy
5. Validate results using synthetic data with known ground truth

A Mathematical Proofs

A.1 Proof of Theorem 2.2

The optimal sampling points for polynomial interpolation are those that minimize the maximum interpolation error. For the interval $[-1, 1]$, these are the Chebyshev points:

$$x_k = \cos\left(\frac{(2k-1)\pi}{2n}\right), \quad k = 1, 2, \ldots, n \tag{49}$$

However, for least squares fitting with uniform weights, equally spaced points provide optimal coverage of the domain.

B Computational Algorithms

B.1 Algorithm: 1D Polynomial Fitting

```
function polyfit1d(x, y, w, degree)
    Input: x (sample points), y (observations), w (weights), degree
    Output: coefficients c

    n = length(x)
    m = degree + 1

    // Construct Vandermonde matrix
    V = zeros(n, m)
    for i = 1:n
        for j = 1:m
            V[i,j] = x[i]^(j-1)
        end
    end

    // Form weighted normal equations
    W = diag(w)
    A = V' * W * V
    b = V' * W * y

    // Solve system
    c = A \ b

    return c
end
```

B.2 Algorithm: 2D Polynomial Fitting

```
function polyfit2d(u, v, z, w, degree)
```

```
    Input: u, v (grid coordinates), z (observations), w (weights), degree
    Output: coefficients c

    // Vectorize grid
    [U, V] = meshgrid(u, v)
    u_vec = U(:)
    v_vec = V(:)
    z_vec = z(:)
    w_vec = w(:)

    // Construct design matrix
    m = (degree+1)*(degree+2)/2
    X = zeros(length(u_vec), m)

    col = 1
    for total_degree = 0:degree
        for u_power = 0:total_degree
            v_power = total_degree - u_power
            X[:, col] = (u_vec.^u_power) .* (v_vec.^v_power)
            col = col + 1
        end
    end

    // Solve weighted least squares
    W = diag(w_vec)
    A = X' * W * X
    b = X' * W * z_vec
    c = A \ b

    return c
end
```

C Numerical Examples

C.1 Example: 1D Signal Denoising

Consider the test signal:

$$s(t) = \sin(2\pi t) + 0.5\cos(4\pi t), \quad t \in [0, 1] \tag{50}$$

corrupted by Gaussian noise with $\sigma = 0.1$. Polynomial fitting with degree $d = 6$ and uniform weights yields effective denoising with mean squared error reduction of approximately 80%.

C.2 Example: 2D Image Smoothing

For a 256×256 noisy image, local polynomial fitting with 5×5 neighborhoods and degree $d = 2$ provides:

- PSNR improvement: 15.3 dB to 22.7 dB
- Computation time: 0.8 seconds (MATLAB implementation)
- Edge preservation: Good for moderate noise levels