NP - Final Prep

Chapter 0

Asymptotic Notation: Big O and Little o

Deterministic sequences

Let $\{a_n\}$ and $\{b_n\}$ be real sequences with $b_n > 0$.

- **Big** O:

$$a_n = O(b_n) \quad \iff \quad \exists\, C < \infty,\ \exists\, n_0 :\ |a_n| \le C\, b_n \text{ for all } n \ge n_0.$$

  Intuition: a_n is at most a constant multiple of b_n for large n.

- **Little** o:

$$a_n = o(b_n) \quad \iff \quad \frac{a_n}{b_n} \to 0 \quad \text{as } n \to \infty.$$

  Intuition: a_n is negligible compared to b_n asymptotically.

Basic algebra:

- If $a_n = o(b_n)$, then $a_n = O(b_n)$.

- If $a_n = O(b_n)$ and $c_n = O(d_n)$, then

$$a_n + c_n = O\big(b_n + d_n\big),$$

  and, in particular, if b_n and d_n are comparable,

$$a_n + c_n = O\big(\max\{b_n, d_n\}\big).$$

- If $a_n = O(b_n)$ and $c_n = O(d_n)$, then

$$a_n c_n = O(b_n d_n).$$

- If $a_n = o(b_n)$ and $c_n = O(d_n)$, then

$$a_n c_n = o(b_n d_n).$$

Stochastic versions (often used in this course)

For random quantities X_n, Y_n with $Y_n > 0$:

- **Big** O_p:

$$X_n = O_p(Y_n)$$

  means that the family $\{X_n/Y_n\}$ is **bounded in probability**, i.e. for every $\varepsilon > 0$ there exists C such that

$$\sup_n \Pr\Big(|X_n| > CY_n\Big) < \varepsilon.$$

- **Little** o_p:

$$X_n = o_p(Y_n)$$

  means

$$\frac{X_n}{Y_n} \xrightarrow{p} 0.$$

These are used, for example, to summarize the stochastic order of bias and variance terms of estimators.

Examples used in Chapters 1–2

1. **Bias–variance rate comparison in 1D regression/density problems**
   For many kernel estimators in 1D,

$$\text{Bias}(x) = O(h^2), \qquad \text{Var}(x) = O\left(\frac{1}{nh}\right).$$

   Choosing $h \asymp n^{-1/5}$ gives

$$\text{Bias}(x) = O\left(n^{-2/5}\right), \qquad \text{Var}(x) = O\left(n^{-4/5}\right),$$

   so

$$\text{Bias}^2(x) = O\left(n^{-4/5}\right),$$

   and bias^2 and variance are of the **same order** in the MSE.

2. **Higher-order remainder terms becoming negligible**
   In several derivations in Chapters 1–2, you see algebra of the form

$$O(h^4)\, O\left(\frac{1}{nh}\right) = o\left(\frac{1}{nh}\right),$$

   under bandwidth choices where $h \to 0$ and $nh \to \infty$ in such a way that h^4 is of smaller order than $1/(nh)$.
   This means the product of a higher-order bias term $O(h^4)$ with a variance-scale term $O(1/(nh))$ is **negligible** compared to the leading $1/(nh)$ term.

3. **Integrated criteria and rates in density estimation (MISE)**
   For kernel density estimators,

$$\text{MISE}(h) = \text{IBias}^2(h) + \text{IVar}(h),$$

where

$$\mathrm{IBias}^2(h) = O(h^4), \qquad \mathrm{IVar}(h) = O\left(\frac{1}{nh}\right).$$

With optimal bandwidth $h_{\text{opt}} \asymp n^{-1/5}$,

$$\mathrm{MISE}(h_{\text{opt}}) = O\left(n^{-4/5}\right),$$

and any term that is $o\big(n^{-4/5}\big)$ can be dropped in the asymptotic expansion.

Chapter 0.5

Table 1: Summary of Rates of Convergence for Common Estimators (Optimal Rate is MSE/MISE)

Estimator	Bias	Variance	Optimal bandwidth	Optimal rate
Parametric MLE	$O(n^{-1})$	$O(n^{-1})$	None	n^{-1}
Histogram (1D)	$O(h)$	$O((nh)^{-1})$	$h \asymp n^{-1/3}$	$n^{-2/3}$
KDE (1D)	$O(h^2)$	$O((nh)^{-1})$	$h \asymp n^{-1/5}$	$n^{-4/5}$
NW / Local Linear Regression (1D)	$O(h^2)$	$O((nh)^{-1})$	$h \asymp n^{-1/5}$	$n^{-4/5}$
Multivariate KDE / LPR (d-dim)	$O(h^2)$	$O((nh^d)^{-1})$	$h \asymp n^{-1/(4+d)}$	$n^{-4/(4+d)}$
Deconvolution (Ordinary Smooth)	$O(h^2)$	inflated by $\lvert\Psi_e(t)\rvert^{-2}$	Slower than $n^{-1/5}$	$n^{-a},\ a < 4/5$
Deconvolution (Supersmooth)	$O(h^2)$	$\exp(c/h^\beta)$	$h \asymp (\log n)^{-1/\beta}$	$(\log n)^{-a}$

Chapter 1

1. The Nonparametric Regression Problem

Setup:
We observe data of the form

$$Y_i = m(X_i) + \varepsilon_i,$$

with minimal assumptions on the regression function $m(\cdot)$.

Key ideas:

- Nonparametric regression avoids specifying a parametric family.
- Smoothness assumptions (derivatives of m) replace strong model assumptions.
- Goal is to recover $m(x)$ in a flexible way.

Oral questions to prepare for:

- What is the difference between parametric and nonparametric regression?
- Why are smoothness assumptions necessary?
- Why can't we simply interpolate the data?

Answers

- **Parametric vs nonparametric:**
  Parametric regression assumes a finite-dimensional form for $m(x)$ (e.g., linear, quadratic), so inference is about a small parameter vector. Nonparametric regression treats $m(\cdot)$ as an infinite-dimensional object and only assumes it is "smooth enough," letting the data determine its shape.
- **Why smoothness assumptions?**
  Without some regularity (e.g., bounded derivatives), there is no way to control the variability of the estimator or obtain convergence rates. Smoothness lets us approximate $m(x)$ locally by low-order polynomials and get explicit bias–variance expansions.
- **Why not interpolate?**
  Exact interpolation fits noise as well as signal, so the variance of the estimator is huge and the resulting function oscillates wildly. Smoothing trades off small bias for a large reduction in variance, which improves prediction and generalization.

---

2. Kernel Smoothers / Moving Average Estimators

General form:

$$\hat{m}(x) = \sum_{i=1}^{n} W_i(x) Y_i,$$

with weights determined by a kernel $K(\cdot)$ and bandwidth h.

Key ideas:

- Kernel determines *shape* of local averaging.
- Bandwidth controls *how much* smoothing is performed.
- Most kernels perform similarly; bandwidth dominates performance.

Oral questions to prepare for:

- What does the kernel do conceptually?
- Why is the bandwidth more important than the kernel choice?
- How do weight functions behave near and far from x?

Answers

- **Conceptual role of the kernel:**
  The kernel specifies how we weight observations as a function of their distance from x: nearby points get high weight, distant points get little or no weight. It defines the *shape* of the local neighborhood used to estimate $m(x)$.
- **Why bandwidth matters more:**
  For reasonable kernels (symmetric, integrating to 1, finite variance), asymptotic bias and variance differ only by kernel-dependent constants. The bandwidth h determines the *scale* of smoothing and thus dominates both the amount of bias and the variance.
- **Behavior of the weights:**
  As $|x_i - x|$ increases, the weight $W_i(x)$ decays according to the kernel (often roughly bell-shaped). Inside a window of width h, weights are non-negligible; outside that window they are essentially zero.

---

3. Bias–Variance Tradeoff

Key decomposition:

- Bias increases with larger bandwidth.
- Variance increases with smaller bandwidth.

For local constant estimators: - Bias $\sim m''(x)h^2$
- Variance $\sim 1/(nh)$

Oral questions to prepare for:

- Intuitively, why does curvature of m matter for bias?
- Why is variance inversely proportional to the number of points in the window?
- What happens when h is too small or too large?

Answers

- **Curvature and bias:**
  Locally, we approximate $m(x)$ by a constant (or low-order polynomial). If m is nearly linear or flat, this approximation is good; when m has high curvature (large $|m''(x)|$), the local constant fit systematically misses the true function, creating larger bias of order $h^2 m''(x)$.
- **Variance and number of points:**
  The effective number of observations contributing to $\hat{m}(x)$ is about nh in 1D. Averaging over more points reduces variability like $1/(nh)$, so smaller windows (smaller h) mean fewer effective observations and hence larger variance.
- **Extremes of h:**
  If h is too small, the estimator is very wiggly: negligible bias but huge variance (undersmoothing). If h is too large, the estimator is overly smooth: low variance but high bias because it washes out local features (oversmoothing).

4. Optimal Bandwidth and MSE Rates

Balancing squared bias and variance yields an optimal bandwidth and rate of convergence.

In 1D:

$$h_{\text{opt}} \asymp n^{-1/5}, \quad \text{MSE}_{\text{opt}} \asymp n^{-4/5}.$$

Key ideas:

- Smoothing makes the problem fundamentally harder than parametric estimation.
- Optimal nonparametric rates are slower due to infinite-dimensional nature of m.
- Choosing h is the central practical challenge.

Oral questions:

- Why does balancing bias and variance yield $n^{-1/5}$?
- Why is the parametric rate n^{-1} unattainable?
- What makes bandwidth selection difficult in practice?

Answers

- **Deriving $n^{-1/5}$ (conceptually):**
  MSE behaves like
  $$\text{MSE}(h) \approx C_1 h^4 + C_2 \frac{1}{nh},$$
  where h^4 comes from squared bias and $1/(nh)$ from variance. Minimizing this in h balances these two terms and yields $h_{\text{opt}} \asymp n^{-1/5}$.

- **Why not n^{-1} rates?**
  Parametric estimators estimate finitely many parameters, so error shrinks at rate n^{-1} for MSE. Here we estimate an entire function, effectively infinitely many parameters, so information per "degree of freedom" is much smaller, leading to slower rates like $n^{-4/5}$.
- **Why bandwidth selection is hard:**
  The optimal h depends on unknown features of m (e.g., curvature) and noise level. In practice we approximate it via data-driven rules (CV, plug-in, etc.), but these methods can be unstable and sensitive to the sample and model assumptions.

5. Equivalent Kernel Interpretation

Concept:
Even non-kernel smoothers (e.g., local polynomial, spline) can be written as:

$$\hat{m}(x) = \sum_{i=1}^{n} W_i^{\text{eq}}(x) Y_i.$$

Key ideas:

- Equivalent kernels unify different smoothers under a single weighting interpretation.
- Equivalent kernels help analyze bias, variance, and boundary behavior.
- Local linear/regression techniques adjust weights dynamically.

Oral questions:

- What is an equivalent kernel and why is it useful?
- How do equivalent kernels differ between Nadaraya–Watson and local linear?
- How does the equivalent kernel reveal boundary correction?

Answers

- **What is an equivalent kernel?**
  It is the implicit weight function $W_i^{\text{eq}}(x)$ that any linear smoother induces at point x, written in a kernel-like form. It tells you how each observation Y_i is weighted in forming $\hat{m}(x)$.
- **NW vs local linear equivalent kernels:**
  The NW equivalent kernel is basically the rescaled original kernel—symmetric and centered at x. For local linear regression, the equivalent kernel is *tilted* in response to the local design; it can be asymmetric and even negative, especially near boundaries.
- **Boundary correction in the equivalent kernel:**
  At boundaries, local linear weights shift mass inward and adjust for the lack of data on one side. This is visible as an asymmetric equivalent kernel that effectively extrapolates a local line instead of just averaging.

6. Boundary Behavior

Key phenomenon:
Standard kernel smoothers suffer severe bias near boundaries.

Key ideas:

- At the boundary, the kernel loses symmetry and underweights boundary points.
- Local linear (and higher-order) methods correct this by fitting a slope.
- Equivalent kernel for local linear is asymmetric near boundaries and corrects bias.

Oral questions:

- Why do kernel estimators behave poorly at boundaries?
- How exactly does local linear regression mitigate boundary bias?
- Why is boundary correction important in practice?

Answers

- **Why poor boundary behavior?**
  Near a boundary, half of the kernel window falls outside the support where there is no data. A symmetric kernel still "expects" data on that side, so the average is pulled toward the interior, creating systematic bias.
- **How local linear fixes it:**
  Local linear regression fits a line rather than a constant using weighted least squares. This fit accounts for the slope of m near the boundary and uses asymmetric weights to extrapolate, which largely cancels the one-sided bias.
- **Why boundary correction matters:**
  Many practical quantities (e.g., endpoints of time series, extremes of covariate ranges) lie near boundaries. Ignoring boundary bias leads to misleading inferences precisely where data are scarce and uncertainty is high.

---

7. Asymptotic Distribution

For fixed x, under regularity conditions:

$$\sqrt{nh}\,(\hat{m}(x) - m(x) - \text{Bias}) \xrightarrow{d} N(0, \sigma^2 R(K)/f_X(x)).$$

Key ideas:

- Asymptotic normality characterizes uncertainty for large n.
- Bias generally dominates unless we undersmooth or correct it.

- Variance depends on kernel shape and density of design points.

Oral questions:

- What is the interpretation of asymptotic normality?
- Why must we undersmooth for valid confidence intervals?
- How does the sampling density $f_X(x)$ influence variance?

Answers

- **Interpretation of asymptotic normality:**
  After rescaling by $\sqrt{nh}$, the centered estimator behaves like a normal random variable for large n. This allows us to approximate the distribution of $\hat{m}(x)$ and construct confidence intervals.
- **Why undersmoothing for CIs?**
  The bias is of order h^2, while the standard error is of order $(nh)^{-1/2}$. With an MSE-optimal bandwidth, these are of the *same* order, so bias does not vanish relative to the standard error. Choosing a slightly smaller h (undersmoothing) makes the bias $o(\text{SE})$, so normal-based intervals become valid.
- **Role of $f_X(x)$:**
  The variance term is proportional to $1/f_X(x)$: when the design density is small, there are fewer nearby observations, so the variance of $\hat{m}(x)$ is larger. Regions with sparse X-values are intrinsically harder to estimate.

---

8. Curse of Dimensionality

For dimension d:

$$h_{\text{opt}} \asymp n^{-1/(4+d)}, \quad \text{MSE} \asymp n^{-4/(4+d)}.$$

Key ideas:

- Required sample size grows exponentially with dimension.
- Local neighborhoods become sparse in high dimension.
- Motivates additive models, projection methods, and dimension reduction.

Oral questions:

- Why does nonparametric regression degrade rapidly as d increases?
- What is the geometric intuition behind sparsity in high dimensions?
- How do practitioners circumvent the curse of dimensionality?

Answers

- **Why performance degrades with d:**
  As d increases, you need exponentially more data to fill the space well enough to estimate $m(x)$ locally. The optimal rates $n^{-4/(4+d)}$ get much slower with larger d, so nonparametric regression becomes statistically inefficient.

- **Geometric intuition for sparsity:**
  In high dimensions, most points are far from each other; even a large sample looks “thin” in the space. A ball of fixed radius around x contains a tiny fraction of the total volume, so very few points fall in any local neighborhood.

- **Practical strategies:**
  People use dimension reduction (PCA, sufficient dimension reduction), additive models, single-index models, or impose structure (e.g., sparsity, smoothness in selected directions) to reduce the effective dimensionality before applying nonparametric methods.

Chapter 2

1. Error Criteria for Density Estimation

Key concepts:

- Definitions of pointwise MSE and integrated MISE.
- MISE decomposes into integrated squared bias and integrated variance.
- L1 vs L2 loss: L2 is analytically convenient; L1 is more robust.

Oral questions:

- What is the difference between MSE and MISE?
- Why do we use integrated criteria for density estimation?
- Why are L2-based criteria easier for theoretical work?

Answers

- **Difference between MSE and MISE:**
  Pointwise MSE evaluates estimator accuracy at a single x. MISE integrates MSE over the whole domain and measures *global* performance. MISE is the density-estimation analogue of the regression IMSE criterion.
- **Why integrated criteria?**
  Density estimation aims to estimate the entire density function, not one value. MISE summarizes global error and works well analytically because expectation and integration commute.
- **Why L2-based criteria?**
  L2 loss leads to closed-form bias and variance expressions (e.g., using Parseval's identity) and makes asymptotic optimization tractable. L1 loss lacks smooth derivatives and is mathematically harder to analyze.

---

2. Histogram Estimators

Estimator:
Divide domain into bins of width h and estimate density via normalized bin counts.

Key ideas:

- Simple but discontinuous.
- Bias depends on bin alignment.
- Converges slower than kernel estimators.

Oral questions:

- Why are histograms not smooth?

- Why does bin alignment affect performance?

- What makes histograms less efficient than kernel estimators?

Answers

- **Lack of smoothness:**
  The histogram is piecewise constant: it jumps abruptly at bin boundaries. No local averaging occurs beyond assigning data to bins.

- **Bin alignment issues:**
  The position of bin edges relative to the data can drastically change the estimate, especially with small bins. Small shifts in the grid can move points into different bins, altering the estimate discontinuously.

- **Why less efficient:**
  Histograms waste information by treating all points in a bin identically and ignoring distances within bins. KDEs exploit smooth weighting and yield smaller bias and variance.

---

3. Kernel Density Estimators (KDE)

Estimator:

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right).$$

Key ideas:

- Smooth analogue of histograms.

- Bandwidth controls smoothness; kernel less important asymptotically.

- Undersmoothing vs oversmoothing tradeoff.

Oral questions:

- What does the kernel function do conceptually?

- Why are most kernels interchangeable asymptotically?

- How does bandwidth affect bias and variance?

Answers

- **Conceptual role of kernel:**
  It defines the *shape* of the local neighborhood around x. Points closer to x get more weight; distant points get little or none.

- **Why kernels are interchangeable:**
  As long as kernels satisfy standard regularity (symmetric, finite second moment), bias and variance differ only by constants. Bandwidth determines the rate; kernel choice only affects constants.

- **Bandwidth effect:**
  Larger h → more smoothing → larger bias, smaller variance.
  Smaller h → less smoothing → small bias, large variance.

Additional note:
Although symmetry of the kernel is not strictly required for KDE, using a symmetric kernel greatly simplifies theoretical derivations. In particular, symmetry ensures

$$\int u\,K(u)\,du = 0,$$

so odd-order terms vanish in Taylor expansions. This is why the leading bias term depends only on $f''(x)$ and the second moment of the kernel:

$$\text{Bias}(x) \approx \tfrac{1}{2}h^2 f''(x)\,\mu_2(K).$$

---

4. Bias and Variance of KDE

Leading terms (1D): - Bias: $\frac{1}{2}h^2 f''(x)\mu_2(K)$.
- Variance: $\frac{1}{nh}R(K)f(x)$.

Key ideas:

- Bias depends on curvature of f.
- Variance depends on number of nearby observations.
- Tradeoff creates optimal bandwidth.

Oral questions:

- Intuition behind the h^2 bias term?
- Why does variance scale like $1/(nh)$?
- How do curvature and roughness affect optimal bandwidth?

Answers

- **Why h^2 bias:**
  The kernel is effectively fitting a constant locally. Taylor expansion of $f(x-u)$ shows the first non-vanishing term contributing to bias is proportional to $f''(x)h^2$.
- **Variance scaling:**
  Effective sample size $\approx nh$ in 1D. More points in the local window means less randomness, giving $1/(nh)$ behavior.
- **Effect of curvature:**
  If f is highly curved, bias grows faster for fixed h, so optimal h becomes smaller. Rougher densities require more undersmoothing.

---

5. Boundary Bias and Boundary Corrections

Issue:
Standard kernels place weight outside the support, causing downward bias.

Key ideas:

- Severe underestimation near boundaries.
- Boundary kernels or reflection methods help.
- Transformation methods map the domain to an unbounded space.

Oral questions:

- Why is KDE biased downward near boundaries?
- How do boundary kernels fix the problem?
- When would transformation methods be used instead?

Answers

- **Downward bias explanation:**
  A symmetric kernel centered near a boundary puts significant weight on nonexistent data outside the support. The estimator averages in "phantom observations," lowering the estimate.
- **Boundary kernel fix:**
  Modify or truncate the kernel near the boundary so that all weight stays inside the support. This restores appropriate weighting and reduces bias.
- **Transformation methods:**
  When the domain is naturally bounded (e.g., proportions, positive variables), transformations like log or logit map $[0, \infty)$ or $[0, 1]$ to $\mathbb{R}$ where standard kernels can be applied.

Additional boundary correction technique – pseudodata:
Another approach for reducing boundary bias is to augment the observed data with **pseudodata**, often via reflection.
For example, near a lower boundary a, an observation X_i close to a may be supplemented with a reflected pseudovalue $2a - X_i$.
This restores symmetry in the kernel window without modifying the kernel itself and substantially reduces boundary bias.

---

6. Bandwidth Selection Methods

Methods:

- Rule-of-thumb (normal reference).

- Oversmoothing rule.
- LSCV and biased CV.
- Plug-in estimators.
- Smoothed cross-validation.

Key ideas:

- Bandwidth dominates estimator behavior.
- LSCV tends to undersmooth.
- Plug-in methods estimate unknown curvature terms.

Oral questions:

- Why is bandwidth selection central?
- Why does LSCV undersmooth?
- What is the idea behind plug-in methods?

Answers

- **Central role of bandwidth:**
  It controls the bias–variance tradeoff entirely. Almost all practical quality differences between KDE fits arise from bandwidth choice, not kernel choice.
- **Why LSCV undersmooths:**
  LSCV tries to minimize integrated squared error, but it is extremely noisy and tends to chase spurious bumps in the data, pushing h smaller than optimal.
- **Plug-in idea:**
  Start from the asymptotically optimal bandwidth formula, which involves unknown functionals like $\int (f''(x))^2 dx$. Estimate these quantities using pilot estimates, then "plug in" to yield a data-driven h.

---

7. Kernel Density Estimation in Practice

Practical considerations:

- Sensitive to sample size, skewness, multimodality.
- Requires visualization and diagnostics.
- Stability of bandwidth choice important.

Oral questions:

- How would you compare multiple KDE fits?
- When is KDE unreliable?
- Why might practitioners prefer plug-in methods?

Answers

- **Comparing fits:**
  Overlay KDEs with different bandwidths, examine persistence of modes, and inspect derivative estimates. Stable features across bandwidths are likely real.
- **When KDE is unreliable:**
  Small samples, heavy tails, or highly irregular densities. In high dimensions KDE becomes nearly unusable.
- **Why prefer plug-in:**
  Plug-in selectors are smoother, more stable, and less variable than cross-validation, making them preferable in many applied settings.

---

8. Multivariate KDE and Curse of Dimensionality

Estimator:

$$\hat{f}_h(x) = \frac{1}{nh^d} \sum K\left(\frac{x - X_i}{h}\right).$$

Key ideas:

- Ball volume shrinks rapidly in high d.
- Rates deteriorate: $n^{-4/(4+d)}$.
- Bandwidth becomes a matrix (or scalar for isotropic case).

Oral questions:

- Why does KDE perform poorly in high dimensions?
- What is the intuition for the h^d scaling?
- How do practitioners mitigate the curse of dimensionality?

Answers

- **Poor performance in high d:**
  Local neighborhoods contain very few points unless h is huge, which creates overwhelming variance. To compensate, h must be large, but that inflates bias—creating an impossible tradeoff.
- **Intuition for h^d:**
  The "window volume" is h^d. In higher dimensions, this volume grows or shrinks exponentially with h. Therefore, the effective sample size is nh^d, which becomes tiny when d is large.

- **Mitigation strategies:**
  Dimension reduction (PCA, sliced inverse regression), additive models, projection pursuit, variable selection, or restricting KDE to low-dimensional summaries.

Chapter 3

1. The Multivariate Density Estimation Problem

Setup:
We observe $X_i \in \mathbb{R}^d$ with density $f(x)$.
Goal: estimate $f(x)$ without a parametric model.

Key ideas:

- All 1D concepts (bias, variance, smoothing, bandwidth) extend to d-dimensions.
- Dimensionality drastically alters convergence rates.
- Geometry plays a central role in bandwidth selection.

Oral questions:

- What changes when we move from 1D to d-dimensions?
- Why is density estimation fundamentally harder in higher dimensions?
- What does the curse of dimensionality mean in this context?

Answers

- **What changes from 1D to d:**
  In 1D, neighborhoods are intervals; in d-D they are balls/ellipsoids whose volume scales like h^d. All the familiar bias–variance formulas now depend on the *dimension* via this h^d volume factor and via multivariate derivatives (gradient, Hessian).
- **Why it is harder in higher dimensions:**
  To get enough points in a local neighborhood, the window must be large in volume, which inflates bias. If we shrink the window to reduce bias, the number of observations inside it collapses. This tradeoff worsens as d increases.
- **Curse of dimensionality in this context:**
  The optimal rates of convergence become much slower with d; the sample size required to maintain a given level of accuracy grows (roughly) exponentially in d. Nonparametric methods become practically infeasible beyond low dimensions.

---

2. Multivariate Histograms

Estimator:
Divide space into d-dimensional bins; estimate probability mass in each bin.

Key ideas:

- Simple but scale-dependent and poorly adapted to geometry.

- Discontinuous and sensitive to bin placement.
- Suffers from exponential bin growth as d increases.

Oral questions:

- Why are multivariate histograms rarely used in practice?
- How does dimension affect computational feasibility?
- Why do histograms distort geometric structure in high dimension?

Answers

- **Rarely used in practice:**
  The number of bins grows like $(1/h)^d$. Even with moderate d, you need an enormous sample size to avoid most bins being empty or nearly empty, so the estimator is extremely noisy.
- **Dimensionality and feasibility:**
  Memory and computation both grow with the number of bins, which explodes as d increases. Keeping a fine grid in 5–10 dimensions is essentially impossible.
- **Geometric distortion:**
  Axis-aligned bins ignore the covariance structure and correlations in the data. Distances and angles are not respected, so the histogram partitions the space in a way that is unrelated to the intrinsic geometry of the distribution.

---

3. Multivariate Kernel Density Estimator (KDE)

Estimator:

$$\hat{f}_H(x) = \frac{1}{n}\sum_{i=1}^{n} K_H(x - X_i),$$

where

$$K_H(u) = |H|^{-1/2} K(H^{-1/2}u),$$

and H is the **bandwidth matrix**.

Key ideas:

- Requires choosing shape + scale of smoothing region.
- Kernel shape less important than bandwidth matrix.
- Ensures generalization of 1D properties.

Oral questions:

- Why do we need a bandwidth *matrix* and not a scalar?
- Why is kernel choice still less important than bandwidth selection?
- What geometric role does H play?

Answers

- **Need for a matrix:**
  In multiple dimensions, we must decide not only “how big” the neighborhood is but also its *shape* and *orientation*. A scalar h only rescales all directions equally; a matrix H encodes anisotropy and rotation.
- **Kernel vs bandwidth:**
  As in 1D, as long as K is a reasonable multivariate kernel (symmetric, integrable, etc.), the detailed shape affects only constants. The bandwidth matrix H determines the effective local neighborhood and thus dominates bias and variance.
- **Geometric role of H:**
  The level sets of $(u^\top H^{-1} u)$ are ellipsoids. H selects the ellipsoidal neighborhood around x: its eigenvalues control spread in each principal direction, and eigenvectors control orientation.

---

4. Bandwidth Matrices: Full, Diagonal, and Scalar

Types: - Scalar bandwidth: $H = h^2 I$ (isotropic smoothing).
- **Diagonal bandwidth**: scales differ per coordinate.
- **Full matrix**: allows rotation + anisotropic smoothing.

Key ideas:

- Full H captures correlations between dimensions.
- Diagonal H handles different scales but not orientation.
- Scalar h easiest but often insufficient.

Oral questions:

- Why may scalar bandwidth be inappropriate in correlated data?
- When is a diagonal bandwidth acceptable?
- How does the full matrix reflect data geometry?

Answers

- **Scalar bandwidth issues:**
  If variables have very different scales or strong correlations, a spherical neighborhood is either too wide in some directions or too narrow in others. It fails to adapt to the covariance structure.
- **When diagonal is okay:**
  When variables are approximately uncorrelated but on different scales, a diagonal H can rescale each coordinate appropriately without needing rotation.
- **Full matrix and geometry:**
  A full H can align its principal axes with the main directions of variation in the data (e.g., via the covariance matrix). This allows elongated ellipsoids that follow the geometry of f, reducing bias for a fixed level of variance.

---

5. Bias and Variance in Multivariate KDE

Leading behavior: - Bias increases with the smoothness and eigenvalues of H.
- Variance behaves as:

$$\mathrm{Var}(\hat{f}_H(x)) \sim \frac{1}{n|H|^{1/2}}.$$

Key ideas:

- Bias depends on curvature of f in multiple directions.
- Variance depends on hyper-volume of the smoothing region.
- Larger d dramatically increases variance for fixed n.

Oral questions:

- Why does variance scale with $|H|^{-1/2}$?
- How does increasing dimension affect bias?
- How does curvature generalize to Hessians in higher dimensions?

Answers

- **Variance and $|H|^{-1/2}$:**
  The effective local volume is $|H|^{1/2}$. The expected number of points in this region is about $n|H|^{1/2}$, so variance behaves like the inverse of that count, giving $1/(n|H|^{1/2})$.
- **Effect of increasing dimension on bias:**
  Bias involves combinations of second (and higher) derivatives of f along each coordinate or principal direction. As d grows, there are more directions in which curvature can contribute, and controlling all these contributions is harder.
- **Curvature as Hessians:**
  In multivariate settings, curvature is described by the Hessian matrix $D^2 f(x)$. The bias involves traces or quadratic forms in H and the Hessian, e.g. terms like $\mathrm{tr}(HD^2 f(x))$.

---

6. Curse of Dimensionality

Rates:

$$\mathrm{MISE} \asymp n^{-4/(4+d)}, \qquad h_{\mathrm{opt}} \asymp n^{-1/(4+d)}.$$

Key ideas:

- Required sample size grows exponentially in dimension.
- Local neighborhoods become sparse (“distance concentration”).

- Practical KDE typically limited to $d \leq 4$.

Oral questions:

- Why does KDE deteriorate so quickly with increasing d?
- What is the geometric intuition for sparse neighborhoods?

Answers

- **Why KDE deteriorates:**
  The effective sample size in a local region is nh^d. To keep variance small, we need nh^d large; to keep bias small, we need h small. Balancing these gives the rate $n^{-4/(4+d)}$, which gets worse rapidly with d.
- **Geometric intuition for sparsity:**
  In high dimensions, most of the volume of a ball is near its surface, and distances between points tend to concentrate. For any fixed radius, the fraction of space covered by a neighborhood is tiny, so very few data points fall close to any given x. This makes local averaging extremely noisy unless n is enormous.

Chapter 4

1. The Nonparametric Regression Model

Model:

$$Y_i = m(x_i) + \varepsilon_i,$$

where $m(\cdot)$ is an unknown smooth regression function.

Key ideas:

- We model the **mean function** $m(x)$ without assuming parametric structure.
- Noise may be homoscedastic or heteroscedastic.
- Goal: estimate $m(x)$ at arbitrary points.

Oral questions:

- How is nonparametric regression different from kernel density estimation?
- Why do we not assume a specific functional form for $m(\cdot)$?
- What role does smoothness play?

Answers

- **Difference from KDE:**
  KDE estimates the density of X; regression estimates the conditional mean of Y given X. Regression involves an *input–output* relationship and must account for noise ε_i, whereas KDE is noise-free and purely about smoothing the empirical distribution of X.
- **Why no functional form:**
  Parametric forms risk misspecification; nonparametric regression lets the data determine the shape of $m(x)$. This flexibility avoids bias from incorrect model assumptions.
- **Role of smoothness:**
  Smoothness assumptions (e.g., bounded $m''(x)$) give control of local approximation error and make it possible to derive bias/variance expansions and convergence rates. Without smoothness, meaningful estimation of $m(x)$ is impossible.

---

2. Nadaraya–Watson (NW) Kernel Regression

Estimator:

$$\hat{m}_h(x) = \frac{\sum_{i=1}^n K_h(x - x_i)\, Y_i}{\sum_{i=1}^n K_h(x - x_i)}.$$

Key ideas:

- Local constant approximation.

- Simple and intuitive; severe boundary bias.
- Weighted average of nearby Y_i.

Oral questions:

- Why is NW called a "local constant" estimator?
- Why does NW suffer from significant boundary bias?
- Why is the denominator necessary?

Answers

- **Local constant:**
  NW arises from minimizing a *locally weighted* least squares criterion where the fitted function is restricted to be constant in a neighborhood around x. That is, it approximates $m(x)$ by a constant β_0 locally.
- **Boundary bias:**
  At boundaries, the symmetric kernel extends outside the support of X. The estimator then averages only interior points, pulling the estimate toward the center of the data distribution and causing systematic bias.
- **Denominator's role:**
  It normalizes the kernel weights so that they sum to 1, ensuring $\hat{m}(x)$ is a convex combination of the observed Y_i and that the estimator behaves properly when the density of design points varies.

Optimization interpretation:
The Nadaraya–Watson estimator is the minimizer of the locally weighted least squares objective

$$\min_{\beta_0} \sum_{i=1}^{n} K_h(x_i - x)\,(Y_i - \beta_0)^2.$$

Thus NW is a **local constant regression estimator**, obtained by minimizing the local MSE at the evaluation point x.

---

3. Local Polynomial Regression (LPR)

Estimator:
Fit a polynomial of degree p near x via weighted least squares:

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} K_h(x_i - x) \left[Y_i - \sum_{j=0}^{p} \beta_j (x_i - x)^j \right]^2.$$

Key ideas:

- $p = 0$: NW estimator.

- $p = 1$: local linear (most widely used).
- $p \geq 2$: lower bias but less stable; introduces negative weights.

Oral questions:

- Why do we center the polynomial at the target x?
- Why is local linear the "default" choice?
- Why do higher-order LPRs introduce negative weights?

Answers

- **Centering the polynomial:**
  Centering at x makes the fitted intercept $\hat{\beta}_0$ directly estimate $m(x)$. It also ensures the Taylor expansion aligns with the point of interest and simplifies bias analysis.
- **Why local linear is default:**
  It corrects boundary bias automatically, reduces the leading bias term relative to NW, and remains numerically stable. It offers the optimal balance between bias reduction and variance inflation.
- **Negative weights in higher-order LPR:**
  Fitting higher-degree polynomials locally requires the estimator to *oscillate* to match curvature, which forces some weights to be negative. This can cause instability and spurious wiggles in the fitted function.

Additional theoretical insight on degree selection:
Local polynomial regression exhibits a useful odd–even phenomenon.
When estimating $m(x)$ or its derivatives, the leading variance term depends primarily on the **even part** of the equivalent kernel, whereas the bias depends on the polynomial degree p.

As a result:

- For certain tasks—especially estimating **odd-order derivatives**—increasing the polynomial degree by 2 (e.g., $p = 1 \to 3$)
  reduces the leading bias term without increasing the variance order.
- Similarly, for **even-order derivative estimation**, increasing p from one even value to the next even value reduces bias while leaving the variance order unchanged.

This explains why odd degrees often perform best for odd-derivative estimation and even degrees for even-derivative estimation: the additional polynomial terms eliminate lower-order bias terms but do not alter the first-order variance structure.

---

4. Equivalent Kernel Representation

Representation:

$$\hat{m}(x) = \sum_{i=1}^{n} W_i^{\mathrm{eq}}(x)\, Y_i.$$

Key ideas:

- Reveals how smoothers weight observations.
- Local linear equivalent kernel is asymmetric near boundaries → bias correction.
- Unifies many regression smoothers.

Oral questions:

- What is the equivalent kernel and why is it valuable?
- How does it show boundary correction?
- Why does NW's equivalent kernel remain symmetric (and why is that bad)?

Answers

- **What is it / why valuable:**
  The equivalent kernel expresses any linear smoother as a weighted average of the data. This makes comparison straightforward and allows theoretical results (bias, variance, boundary behavior) to be analyzed through weight shapes.
- **How it shows boundary correction:**
  For local linear regression, the equivalent kernel becomes asymmetric near boundaries, allocating more weight to interior points to offset missing data on the exterior side. This "tilting" cancels the leading boundary bias term.
- **Why NW remains symmetric:**
  NW uses the raw kernel weights, which do not adapt to boundaries. Because the kernel allocates weight outside the support where no data exist, the estimator is pulled inward, causing persistent boundary bias.

---

5. Bias and Variance of Kernel Regression

Leading behavior (local linear):

- Bias: $\sim \frac{1}{2}h^2 m''(x)\mu_2(K)$.
- Variance: $\sim \sigma^2(x)\,\frac{R(K)}{nhf_X(x)}$.

Key ideas:

- Bias depends on curvature of $m(x)$.
- Variance depends on local sample size and design density.
- Design density matters because fewer nearby points → higher variance.

Oral questions:

- Why is curvature central to bias?
- Why does design density affect variance?
- How do these compare to KDE scaling?

Answers

- **Curvature and bias:**

  Local polynomial regression approximates $m(x)$ with a Taylor expansion. If $m''(x)$ is large, the local linear fit cannot capture local curvature and thus incurs greater bias.

- **Design density and variance:**

  Variance depends on how many design points fall near x. If $f_X(x)$ is small, few points are weighted heavily, leading to large variance even with large n.

- **Comparison to KDE:**

  The variance term $\sim 1/(nhf_X(x))$ parallels KDE exactly: regression smoothers behave like KDE applied to residual variation. Bias structures are similar but relate to $m''(x)$ rather than $f''(x)$.

---

6. Boundary Behavior

Issue:
NW estimator has significant bias at boundaries.

Key ideas:

- Kernel symmetry breaks at the boundary.
- Local linear regression corrects by fitting a slope.
- Equivalent kernel becomes asymmetric.

Oral questions:

- Why is NW biased at boundaries?
- How does local linear regression fix it?
- What does the equivalent kernel look like near boundaries?

Answers

- **NW boundary bias:**
  The kernel allocates weight outside the domain; with no data there, the estimator averages only interior points, pushing the estimate toward the interior mean.
- **Local linear fix:**
  By fitting a line locally, the estimator extrapolates to the boundary rather than averaging. This cancels the leading bias term.

- **Equivalent kernel:**
  At boundaries, it is asymmetric, with heavier weighting of interior points and lighter weighting near the boundary, effectively tilting the smoothing window.

---

7. Bandwidth Selection for Regression

Methods: - CV, GCV, plug-in, rules of thumb.

Key ideas:

- Bandwidth dominates the smoother's behavior.
- CV tends to undersmooth due to noisy risk estimates.
- Plug-in methods use biased-but-stable derivative estimates.

Oral questions:

- Why is bandwidth selection harder in regression?
- Why does CV choose too-small bandwidths?
- What assumptions underlie plug-in methods?

Answers

- **Harder than density estimation:**
  Regression involves estimating a conditional expectation with noise in Y. Variation in ε adds randomness to the CV criterion, making the risk surface noisier and harder to optimize.
- **Why CV undersmooths:**
  CV focuses on prediction error. It over-penalizes bias and under-penalizes variance, leading to small h that chase noise rather than structure.
- **Assumptions behind plug-in:**
  Plug-in estimators rely on approximating unknown quantities such as $m''(x)$, $\sigma^2(x)$, or $f_X(x)$ using preliminary pilot smoothers. These assume local smoothness and consistent pilot estimates.

---

8. Local Likelihood and Other Extensions

Concepts:

- Local likelihood fits exponential-family models locally.
- Handles heteroscedasticity and non-Gaussian responses.
- Uses kernel weighting but models the distribution, not just the mean.

Oral questions:

- Why might local likelihood be preferable to local polynomials?
- How does local likelihood generalize NW and LPR?
- What advantages does it offer for non-Gaussian data?

Answers

- **Why preferable:**
  Local likelihood respects distributional structure (e.g., Poisson counts, binomial proportions). This often reduces bias and improves interpretability compared to polynomial fits on the mean scale alone.
- **Generalization:**
  NW and LPR are special cases of local likelihood under Gaussian assumptions with identity link. Replacing Gaussian log-likelihood with another exponential-family likelihood yields generalized smoothers.
- **Advantages for non-Gaussian data:**
  It handles variance–mean relationships naturally (e.g., Poisson: variance = mean). It avoids transforming the response and often produces more stable and statistically efficient estimators.

Chapter 5

1. Historical Background and Motivation

Key historical points (slides pg. 2–4): - Origins in **statistical learning theory** (Vapnik & Chervonenkis, 1960s–1990s).
- Built around **structural risk minimization** and **VC dimension** concepts.
- First SVM classifiers: mid-1990s.
- Modern SVMs: hinge loss + kernels + convex optimization.

Motivation:

- Find a classifier with strong generalization.
- Avoid overfitting by maximizing geometric margin.
- A rare example of a model derived from rigorous theory first, applications second.

Oral questions:

- What is the basic historical motivation behind SVMs?
- How does SVM relate to VC theory?
- Why is maximizing margin theoretically appealing?

Answers

- **Basic historical motivation:**
  SVMs were built to provide classifiers with *provable* generalization guarantees, grounded in statistical learning theory. Instead of just minimizing training error, they explicitly control model complexity via the margin.
- **Relation to VC theory:**
  VC theory gives bounds on generalization error in terms of empirical error + a complexity term depending on VC dimension. SVMs embody **structural risk minimization** (SRM): choose the function (hyperplane) that achieves a good tradeoff between these two, with the margin acting as a complexity proxy.
- **Why margin is appealing:**
  Larger margins correspond to simpler classifiers in VC theory. Intuitively, a large margin means the classifier is robust to perturbations of the data, which usually translates into better performance on unseen samples.

---

2. Linear Separating Hyperplanes and Geometric Margin

Hyperplane:

$$w^\top x + b = 0.$$

Key ideas (slides pg. 6–9):

- Margin = distance between hyperplane and closest points.
- Max-margin classifier chooses hyperplane with **maximum geometric margin**.
- Only a few points (support vectors) determine the solution.

Oral questions:

- What is the geometric margin?
- Why do only the closest points matter?
- What does a large margin imply about generalization?

Answers

- **Geometric margin:**
  For a labeled point (x_i, y_i), the geometric margin is
  $$\gamma_i = \frac{y_i(w^\top x_i + b)}{\|w\|},$$
  the signed distance to the hyperplane. The overall margin is the minimum γ_i over all training points.
- **Why only closest points matter:**
  Maximizing the minimum distance depends only on the points that achieve that minimum—the ones lying on or closest to the margin boundaries. These are exactly the *support vectors*; moving other points slightly does not affect the optimal hyperplane.
- **Implication of large margin:**
  A large margin makes the classifier robust: small perturbations in the inputs (or sampling variability) are unlikely to change the predicted labels. This is precisely the kind of robustness VC theory associates with low generalization error.

---

3. Hard-Margin SVM (Linearly Separable Case)

Optimization problem:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{s.t. } y_i(w^\top x_i + b) \geq 1.$$

Key ideas (slides pg. 10–13):

- Equivalent to maximizing the margin $1/\|w\|$.
- Convex, quadratic programming problem.
- Support vectors lie exactly on the margin boundaries.

Oral questions:

- What does the hard-margin objective represent geometrically?
- Why is the optimization convex?
- What happens if the data are perfectly separable but barely so?

Answers

- **Geometric interpretation of objective:**
  Minimizing $\frac{1}{2}\|w\|^2$ subject to constraints ensures the margin
  $$\gamma = \frac{1}{\|w\|}$$
  is maximized. So the optimization explicitly finds the separating hyperplane with the largest geometric margin.
- **Convexity:**
  The objective $\frac{1}{2}\|w\|^2$ is strictly convex and the constraints are linear. This makes the problem a convex quadratic program with a unique global optimum—no local minima issues.
- **Barely separable data:**
  If the data are just barely separable, the maximum margin can be extremely small. The hard-margin SVM will still separate perfectly, but the resulting classifier can be very sensitive to noise and may generalize poorly—motivating the soft-margin extension.

---

4. Soft-Margin SVM (Nonseparable Case)

Slack variables:

$$y_i(w^\top x_i + b) \geq 1 - \xi_i, \qquad \xi_i \geq 0.$$

Optimization:

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i.$$

Key ideas (slides pg. 14–17):

- Balances margin size and classification error.
- Parameter C controls penalty on misclassification.
- When C is large → low bias, high variance; when small → larger margin, more errors.

Oral questions:

- Why introduce slack variables?
- What is the role of the tuning parameter C?
- How does soft margin prevent overfitting?

Answers

- **Why slack variables:**
  In nonseparable data, some points must be on the wrong side of the margin (or even misclassified). Slack variables ξ_i quantify margin violations and allow the optimization to trade off hard margin constraints against classification errors.

- **Role of C:**
  C controls how harshly we penalize violations. Large C pushes the solution toward fewer training errors (even at the cost of small margin). Small C tolerates more misclassification to gain a larger margin and simpler classifier.

- **Preventing overfitting:**
  By penalizing norm size and violations jointly, the soft-margin SVM discourages very complex boundaries that perfectly fit noisy labels. The margin regularization term $\frac{1}{2}\|w\|^2$ acts as a complexity control.

---

5. The Dual Problem and Support Vectors

Dual formulation (slides pg. 19–21):

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^\top x_j$$

subject to

$$0 \leq \alpha_i \leq C, \qquad \sum_i \alpha_i y_i = 0.$$

Key ideas:

- Dual depends only on inner products $x_i^\top x_j$.
- Support vectors have $\alpha_i > 0$.
- Enables kernels.

Oral questions:

- Why is the dual formulation essential for kernels?
- What determines which points become support vectors?
- How does the dual reflect margin maximization?

Answers

- **Why dual is essential for kernels:**
  In the dual, the data only appear as inner products $x_i^\top x_j$. This allows us to replace these inner products with kernel evaluations $k(x_i, x_j)$, effectively working in an implicit feature space without ever computing $\phi(x)$.

- **Which points become support vectors:**
  Points with $\alpha_i > 0$ affect the decision boundary. Typically these are the margin points and misclassified points. Observations far inside the correct side of the margin have $\alpha_i = 0$ and do not enter the final classifier.
- **Dual and margin maximization:**
  The dual objective subtracts a quadratic form in α with matrix $y_i y_j x_i^\top x_j$, which encodes the squared norm $\|w\|^2$. Maximizing this dual subject to constraints is equivalent (via KKT conditions) to minimizing $\|w\|^2$ in the primal, i.e., maximizing the margin.

---

6. Kernel Trick and Nonlinear SVMs

Kernel idea (slides pg. 23–28): Replace inner product with kernel:

$$k(x_i, x_j) = \phi(x_i)^\top \phi(x_j).$$

Common kernels:

- Polynomial
- Gaussian RBF
- Sigmoid / neural-tangent style

Key ideas:

- Allows linear machinery to operate in high-dimensional feature spaces.
- Decision boundary becomes nonlinear in input space.
- Avoids explicit computation of $\phi(x)$.

Oral questions:

- What is a kernel, conceptually?
- Why does SVM rely only on inner products?
- What geometric transformation is induced by an RBF kernel?

Answers

- **Conceptual definition of kernel:**
  A kernel is a function $k(x, x')$ that computes the inner product of two points in some (possibly infinite-dimensional) feature space: $k(x, x') = \langle \phi(x), \phi(x') \rangle$. It's an implicit similarity measure.
- **Why SVM uses inner products:**
  The SVM dual formulation and prediction rule can be written entirely in terms of dot products: both in optimization and classification, only terms like $x_i^\top x_j$ and $w^\top x$ appear. This structure is what enables the kernel trick.

- **Geometric effect of RBF kernel:**
  The RBF kernel corresponds to mapping points into an infinite-dimensional feature space where similarity decays with Euclidean distance. In input space, this yields highly flexible, locally adaptive decision boundaries that can curve around clusters.

---

7. VC Dimension, Structural Risk Minimization, and Generalization

Theoretical foundation (slides pg. 31–36): - SVM margin relates to VC bounds: large margin → smaller effective VC dimension.
- SRM balances empirical error + complexity.

Key ideas: - Margin is a complexity control knob.
- SVMs can generalize well even in very high or infinite-dimensional spaces.
- Kernels expand feature space; margin prevents overfitting.

Oral questions: - How does margin relate to effective VC dimension?
- Why can SVMs generalize in infinite-dimensional spaces?
- How is SRM reflected in soft-margin SVM?

Answers

- **Margin and effective VC dimension:**
  For linear classifiers, VC dimension grows with $\|w\|$ and the radius of the data. A large margin (small $\|w\|$) effectively reduces the class's capacity, tightening generalization bounds.
- **Generalizing in infinite dimensions:**
  Even if feature space is infinite-dimensional (e.g., RBF kernel), the *effective* complexity is governed by the margin and the norms of the functions considered. As long as the margin is not too small, VC bounds can still guarantee good generalization.
- **SRM in soft-margin SVM:**
  The objective $\frac{1}{2}\|w\|^2 + C\sum_i \xi_i$ is exactly an SRM tradeoff: the first term is a complexity penalty, the second is empirical error. Varying C moves along a structural hierarchy of classifiers with different complexity levels.

---

8. Practical Behavior, Interpretation, and Limitations

Practical insights (slides pg. 38–42):

- Sparse solution: only support vectors matter.
- Scaling of covariates is essential.
- Choice of kernel + tuning parameters (C, bandwidth) crucial.

Limitations:

- Harder to interpret than linear/logistic models.

- Training can be expensive for very large n.
- Sensitive to hyperparameter choice.

Oral questions:

- Why are SVMs sparse?
- What makes SVMs sensitive to scaling?
- When might SVMs perform poorly relative to, say, random forests or boosting?

Answers

- **Why sparse:**
  In the dual, only points with $\alpha_i > 0$ contribute to the classifier. Many points end up with $\alpha_i = 0$, so predictions depend only on a subset of training data—the support vectors—making the representation sparse.
- **Sensitivity to scaling:**
  Because SVMs use distances and dot products, features with larger scales dominate these computations. Without standardization, the margin and kernel similarity can be distorted, leading to poor boundaries.
- **When SVMs may underperform:**
  - When classes have very complex structure that is better captured by ensembles like random forests or gradient boosting.
  - When the dataset is extremely large, making kernel SVM training computationally expensive.
  - When interpretability is paramount, and logistic/linear models or trees are preferred.

Chapter 6

1. Why Resampling Methods? Motivation and Goals

Purpose:
Approximate variability, bias, and distribution of estimators when analytic methods are difficult or impossible.

Key ideas:

- Provide data-driven approximations to sampling distributions.
- Useful when asymptotics are poor or classical formulas unavailable.
- Particularly important for complex statistics (ratios, medians, nonlinear estimators).

Oral questions:

- Why do we need resampling methods?
- In what situations do analytic variance formulas fail?
- When is resampling preferable to asymptotics?

Answers

- **Why we need resampling methods:**
  Many modern estimators have no closed-form variance or rely on asymptotic approximations that perform poorly in finite samples. Resampling allows the *data itself* to approximate the sampling distribution.
- **When analytic formulas fail:**
  Analytic variance formulas can be unavailable or inaccurate for:
  - statistics that are ratios or nonlinear functionals;
  - estimators that depend on tuning parameters;
  - nonsmooth estimators (medians, percentiles, maxima);
  - complicated dependency structures.
- **When resampling is preferable:**
  When sample sizes are moderate and asymptotic approximations are unreliable, bootstrap/jackknife give more accurate inference. Resampling is also preferred when the statistic has no tractable distribution.

---

2. The Jackknife: Leave-One-Out Recomputing

Definition:
For estimator T_n, jackknife replicates:

$$T_{(i)} = T\big(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n\big).$$

Key ideas: - Based on *systematic* deletion (one observation at a time).
- Works best for smooth, approximately linear statistics.
- Useful for bias estimation in many classical estimators.

Oral questions: - Why does the jackknife work well for smooth estimators?
- Why does it struggle with nonsmooth statistics like medians?
- What is the conceptual meaning of "leave-one-out"?

Answers

- **Why jackknife works for smooth estimators:**
  If a statistic is approximately linear in the data (in the influence-function sense), then deleting one observation produces a predictable, small change. The jackknife exploits this near-linearity.
- **Why it fails for nonsmooth estimators:**
  For medians or maxima, deleting a single observation can dramatically change the estimator. This violates the smoothness assumption and leads to unstable or meaningless jackknife replicates.
- **Conceptual meaning of LOO:**
  The jackknife approximates the effect of a single data point on an estimator by *removing that point* and recomputing. It's a first-order sensitivity or influence assessment.

---

3. Jackknife Bias Estimation

Jackknife estimate of the bias:

$$\widehat{\text{bias}}_{\text{jack}}(T_n) = (n-1)(\bar{T}_{(\cdot)} - T_n),$$

where

$$\bar{T}_{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n} T_{(i)}.$$

Key ideas:

- Jackknife mimics first-order Taylor approximation.
- Often successfully removes $1/n$ bias terms.
- Not effective for nondifferentiable estimators.

Oral questions:

- Why does the jackknife improve bias for linearizable estimators?
- Why does it fail for extreme statistics (max, quantiles)?
- How does the factor $n-1$ arise in the bias formula?

Answers

- **Why jackknife improves bias:**
  For smooth estimators, the first-order Taylor expansion gives a clear expression for the bias. The jackknife cancels out the dominant $1/n$ term through averaging of leave-one-out replicates.
- **Why it fails for extreme statistics:**
  Extreme-value estimators change discontinuously when a single observation is removed. The Taylor expansion breaks down, so the jackknife cannot capture or correct bias reliably.
- **Where the factor $n-1$ comes from:**
  It arises from comparing the full-sample estimator T_n with the average of the $n-1$-sample estimators $T_{(i)}$. It's the correction needed so that the jackknife matches the first-order bias term.

---

4. Jackknife Variance Estimation

Estimate:

$$\widehat{\mathrm{Var}}_{\mathrm{jack}}(T_n) = \frac{n-1}{n}\sum_{i=1}^{n}(T_{(i)} - \bar{T}_{(\cdot)})^2.$$

Key ideas:

- Works well for smooth estimators.
- Rescaling corrects for reduced sample size.
- Fails when estimator is unstable under deletion.

Oral questions:

- Why is the prefactor $(n-1)/n$ necessary?
- When does jackknife variance fail?
- Why does jackknifing the sample maximum give nonsense?

Answers

- **Why the prefactor is needed:**
  Each jackknife replicate uses only $n-1$ observations, so its variability underestimates the true full-sample variance. Multiplying by $(n-1)/n$ adjusts the scale to match the true sampling variability.
- **When jackknife variance fails:**
  When the estimator is not smooth—such as maxima, medians, or discontinuous functionals—small changes in data cause large changes in estimates. The leave-one-out variability becomes erratic.
- **Why it fails for the sample maximum:**
  Removing the largest observation usually changes the max drastically, producing enormous apparent variance. This does not correspond to real sampling variability and is therefore meaningless.

---

5. Bootstrap: Sampling With Replacement

Basic bootstrap estimator:
Draw bootstrap samples $X_1^*, \dots, X_n^*$ from

$$\hat{F}_n = \frac{1}{n} \sum \delta_{X_i}.$$

Compute bootstrap replicates:

$$T_n^* = T(X_1^*, \dots, X_n^*).$$

Key ideas:

- Resamples mimic draws from true distribution F.
- Works for nonsmooth, irregular statistics.
- Provides distributional approximation for T_n.

Oral questions:

- Why does the bootstrap use sampling *with replacement*?
- What distribution is the bootstrap approximating?
- Why is the bootstrap more flexible than the jackknife?

Answers

- **Why sampling with replacement:**
  Sampling with replacement recreates the idea of drawing new samples from the true distribution. The empirical distribution $\hat{F}_n$ plays the role of the unknown F.
- **Which distribution is approximated:**
  The bootstrap approximates the sampling distribution of T_n under the true—but unknown—distribution F, by replacing F with $\hat{F}_n$.
- **Why more flexible than jackknife:**
  Unlike the jackknife, which uses only n deterministic leave-one-out datasets, the bootstrap can explore a large space of possible resamples and handles nonsmooth or discontinuous estimators well.

---

6. Bootstrap Variance and Bias Estimation

Variance:

$$\widehat{\mathrm{Var}}_{\mathrm{boot}}(T_n) = \frac{1}{B} \sum_{b=1}^{B} (T_n^{*(b)} - \bar{T}_n^*)^2.$$

Bias:

$$\widehat{\mathrm{bias}}_{\mathrm{boot}}(T_n) = \bar{T}_n^* - T_n.$$

Key ideas:

- Variance estimated from bootstrap replicates.
- Bias estimation can be noisy.
- Larger B improves stability.

Oral questions:

- Why does bootstrap variance improve on jackknife variance?
- When is bootstrap bias correction unstable?
- How large must B be for reliable estimates?

Answers

- **Why bootstrap variance is better:**
  Bootstrap uses many resamples that better approximate the estimator's sampling distribution. It captures nonlinearities and irregularity that the jackknife, based on only n replicates, cannot.
- **When bias correction becomes unstable:**
  If the estimator is noisy or the bootstrap distribution is skewed or heavy-tailed, $\bar{T}_n^*$ may vary substantially, making bias correction unreliable.
- **How large must B be:**
  For variance estimation, $B \approx 500$–2000 is usually adequate. For confidence intervals—especially BCa—larger B (e.g., 2000–10,000) is recommended.

---

7. Bootstrap Confidence Intervals

Methods:

- Normal approximation.
- Percentile interval.
- BC and BCa intervals.

Key ideas:

- Percentile and BCa intervals avoid explicit standard errors.
- BCa adjusts for both bias and skewness.
- Often better small-sample coverage than asymptotic methods.

Oral questions:

- Why do percentile intervals sometimes outperform normal-based intervals?
- What advantages does BCa offer?
- Why is the bootstrap especially good for skewed distributions?

Answers

- **Why percentile intervals can outperform normal intervals:**
  Normal intervals assume symmetry and approximate normality, which often fails for nonlinear estimators. Percentile intervals use the empirical bootstrap distribution directly.
- **Advantages of BCa:**
  BCa corrects both for *median bias* and *skewness* in the bootstrap distribution. It automatically adapts to the estimator's nonlinearity, leading to higher coverage accuracy.
- **Why helpful for skewed distributions:**
  The bootstrap "sees" the skewness in the data and reproduces it in the resamples, giving intervals that appropriately reflect asymmetry rather than forcing symmetry via $\pm z\hat{\sigma}$.

---

8. Jackknife vs. Bootstrap: Strengths and Limitations

Jackknife strengths: - Fast, deterministic, simple.
- Excellent for linear, smooth statistics.

Bootstrap strengths: - Great for irregular or nonsmooth estimators.
- More accurate CI construction.

Limitations:

- Jackknife fails for extremes, quantiles.
- Bootstrap can struggle with dependence or tiny samples.
- Both require adaptations for time series or clustering.

Oral questions:

- When is jackknife preferred over bootstrap?
- When does bootstrap fail?
- How do both methods relate to estimating sampling distributions?

Answers

- **When jackknife is preferred:**
  When computational speed matters and the estimator is smooth (e.g., means, regression coefficients). Jackknife is faster and stable in such cases.
- **When bootstrap fails:**
  - In small samples where $\hat{F}_n$ poorly approximates F.
  - With highly dependent data (unless block bootstrap is used).
  - When the statistic is extremely sensitive to resampling (e.g., unstable model selection procedures).

- **How both relate to sampling distributions:**
  Both aim to approximate the sampling distribution of T_n. Jackknife uses deterministic deletions; bootstrap uses stochastic resampling. Both replace mathematical derivation with empirical approximation.

Chapter 7

1. What Is Deconvolution and Why Do We Need It?

Definition:
Deconvolution is the process of recovering a target function f when we only observe its convolution with a known distribution G:

$$z = f * G = \int f(x-y)\, dG(y).$$

Key ideas:

- Convolution "blurs" or smooths the truth.
- Deconvolution attempts to reverse this blurring.
- Arises when data are contaminated by measurement error.

Applications (from slides):

- Density estimation with contaminated data.
- Nonparametric regression with errors-in-variables.
- Image deblurring, signal processing, econometrics.

Oral questions:

- Intuitively, what is convolution and why is deconvolution hard?
- Why do measurement errors turn the problem into a convolution problem?
- Why can't we simply invert the convolution directly?

Answers

- **Intuition for convolution / difficulty of deconvolution:**
  Convolution averages or "smears" the true signal with the noise distribution. Information is lost at high frequencies. Deconvolution is trying to *undo* this averaging, which is inherently unstable once information has been blurred away.
- **Why measurement error → convolution:**
  If $Y = X + \varepsilon$ with independent X and ε, then the density of Y is the convolution of the densities of X and ε. So observing Y instead of X automatically turns the problem into recovering f_X from a convolved density.
- **Why not directly invert the convolution:**
  Formally, you can divide characteristic functions and invert the Fourier transform, but in practice this division amplifies noise where the error characteristic function is small. The inversion is an ill-posed problem: tiny perturbations in the data generate huge changes in the estimate.

2. Measurement Error Model and Fourier Transform Framework

Contamination model (slides, pg. 10–14):

$$Y = X + \varepsilon,$$

where we observe Y but want the density of X.

Key ideas: - X and ε independent.
- The characteristic function (Fourier transform) converts convolution into multiplication:

$$\Psi_Y(t) = \Psi_X(t)\Psi_\varepsilon(t).$$

- Thus

$$\Psi_X(t) = \frac{\Psi_Y(t)}{\Psi_\varepsilon(t)}.$$

Oral questions:

- Why is the Fourier transform central to deconvolution?
- Why must the noise distribution G be known or estimable?
- What happens if $\Psi_\varepsilon(t)$ is close to zero?

Answers

- **Why Fourier transform is central:**
  Convolution in the time/space domain becomes *multiplication* in the frequency domain. This is what makes deconvolution algebraically simple: we can write $\Psi_X = \Psi_Y / \Psi_\varepsilon$.
- **Need to know the noise distribution:**
  To divide by $\Psi_\varepsilon(t)$, we need to know it (or estimate it). Without Ψ_ε, there is no way to separate the contributions of X and ε from Ψ_Y.
- **When $\Psi_\varepsilon(t)$ is near zero:**
  Division by very small values explodes any estimation noise in $\hat{\Psi}_Y(t)$. This leads to extremely unstable estimates and is the core reason deconvolution is harder than direct density estimation.

---

3. Naive Deconvolution Estimator and Its Problems

Naive estimator from slides (pg. 16–17):

$$\hat{\Psi}_X(t) = \frac{1}{n}\sum_{j=1}^{n} e^{itY_j} \,/\, \Psi_\varepsilon(t),$$

$$\hat{f}_{\text{naive}}(x) = \frac{1}{2\pi}\int e^{-itx}\hat{\Psi}_X(t)\,dt.$$

Key problems:

- Division by $\Psi_\varepsilon(t)$ amplifies noise dramatically.
- For many noise distributions, $|\Psi_\varepsilon(t)| \to 0$ as $|t|$ grows.
- Naive estimator is unstable and highly variable.

Oral questions:

- Why is naive deconvolution unstable?
- What does the behavior of $\Psi_\varepsilon(t)$ imply about the difficulty?
- Why does the naive estimator require regularization?

Answers

- **Why unstable:**
  The empirical characteristic function $\hat{\Psi}_Y(t)$ is noisy, especially at large $|t|$. Dividing by a small $|\Psi_\varepsilon(t)|$ magnifies this noise, producing wild oscillations in $\hat{\Psi}_X(t)$ and hence in $\hat{f}$.
- **Implication of $|\Psi_\varepsilon(t)| \to 0$:**
  The faster $|\Psi_\varepsilon(t)|$ decays, the more difficult the inverse problem. Rapid decay means high frequencies of X are heavily damped, so any attempt to reconstruct them is extremely ill-conditioned.
- **Need for regularization:**
  We must deliberately *suppress* or down-weight high-frequency components where the division is unstable. Regularization trades some bias for a huge reduction in variance.

---

4. Kernel Density Deconvolution Estimator

Regularized estimator (slides pg. 18–22): Use an ordinary kernel estimate of the contaminated density:

$$\hat{z}(x) = \frac{1}{nh}\sum_{j=1}^{n} K\left(\frac{x - Y_j}{h}\right),$$

with Fourier transform $Z(t) = \hat{\Psi}_Y(t)\, K(th)$.

Then define:

$$\hat{\Psi}_X(t) = \hat{\Psi}_Y(t)\frac{K(th)}{\Psi_\varepsilon(t)},$$

$$\hat{f}(x) = \frac{1}{2\pi}\int e^{-itx}\,\frac{\hat{\Psi}_Y(t)K(th)}{\Psi_\varepsilon(t)}\,dt.$$

Key ideas:

- Kernel provides **regularization** (smooth cutoff in Fourier domain).
- Deconvolution reduces to weighting empirical characteristic function.
- Avoids explosive variance of the naive estimator.

Oral questions:

- How does the kernel act as a regularizer?
- Why is smoothing done in *frequency space*?
- What determines an appropriate bandwidth in deconvolution?

Answers

- **Kernel as regularizer:**
  In frequency space, $K(th)$ decays as $|t|$ grows, effectively *damping* high-frequency components. This prevents us from dividing by tiny $\Psi_\varepsilon(t)$ values with full force, thus controlling variance.
- **Why smoothing in frequency domain:**
  The ill-posedness is a frequency-domain phenomenon (small $\Psi_\varepsilon(t)$ at high $|t|$). Regularizing directly in that domain is natural and leads to clean expressions for bias and variance.
- **Bandwidth choice:**
  The bandwidth h governs the tradeoff between bias and variance in deconvolution just as in KDE—but the variance term now carries an extra penalty from $|\Psi_\varepsilon(t)|^{-2}$. Optimal h must balance kernel smoothing *and* the decay of Ψ_ε.

---

5. Ordinary-Smooth vs Supersmooth Errors

Definitions: - Ordinary smooth noise: $|\Psi_\varepsilon(t)| \sim C|t|^{-\alpha}$.
- **Supersmooth** noise: $|\Psi_\varepsilon(t)| \sim \exp(-c|t|^\beta)$.

Key ideas:

- Supersmooth errors (e.g., Gaussian) make problem dramatically harder.
- Convergence rates depend on how fast $\Psi_\varepsilon(t)$ decays.
- Bandwidth selection differs drastically between the two cases.

Oral questions:

- Why does supersmooth noise make deconvolution much more difficult?
- What is the role of the decay rate of $\Psi_\varepsilon(t)$?
- How do rates differ between ordinary and supersmooth cases?

Answers

- **Why supersmooth is harder:**
  Exponential decay in $|\Psi_\varepsilon|$ means high-frequency components are *almost completely annihilated.* Attempting to recover them requires multiplying by exponentially large factors, which is hopelessly unstable.

- **Role of decay rate:**
  The rate at which $|\Psi_\varepsilon(t)|$ decays determines how aggressively we must regularize and how fast our estimation error can decrease. Slower (polynomial) decay is much more forgiving than exponential decay.
- **Difference in rates:**
  Ordinary smooth errors give algebraic rates (e.g., n^{-a} for some $a > 0$), whereas supersmooth errors often yield logarithmic or stretched-exponential rates (much slower). So Gaussian measurement error can be *asymptotically brutal*.

---

6. Deconvolution in Nonparametric Regression (Errors-in-Variables)

Model (slides pg. 24–26): - Classical errors-in-variables:

$$W = X + \delta, \qquad Y = m(X) + \varepsilon.$$

- Goal: estimate $m(x)$ even though covariates are contaminated.

Key ideas:

- Kernel regression must be adapted via Fourier inversion.
- Deconvolution kernel appears in numerator and denominator.
- Fan & Truong (1993) integral representation.

Oral questions:

- Why can't we use ordinary kernel regression when covariates have measurement error?
- How does deconvolution modify the weights in NW estimator?
- What assumptions are required about the noise distribution?

Answers

- **Why ordinary kernel regression fails:**
  If we regress Y directly on W, we estimate $E[Y \mid W = w]$, not $m(x)$. Measurement error in X induces attenuation and bias; ordinary kernels smear the regression function, giving a biased estimate of m.
- **How weights are modified:**
  In the deconvolution regression estimator, the kernel in the NW numerator and denominator is replaced by a *deconvolution kernel* whose Fourier transform includes division by $\Psi_\delta(t)$. This re-weights observations to undo the contamination.
- **Required assumptions on noise:**
  Typically:
  - known (or estimable) $\Psi_\delta(t)$;
  - independence of X and δ;

  - some smoothness / decay conditions on m and the noise;
  - often known error variance or distributional form.

---

7. Unknown Noise Distribution: Practical Solutions

From supplemental slides (pg. 29–31):

- Noise distribution g may be unknown.
- Solutions:
  - **Validation data** (gold-standard measurements).
  - **Replicated measurements** of X.
  - **Additional experiment** to estimate noise distribution.

Key ideas:

- Deconvolution requires *some* knowledge of noise.
- Replicates identify $\Psi_\varepsilon(t)$ via repeated measurement structure.

Oral questions:

- Why is knowledge of the noise distribution essential?
- How can replicated measurements be used to estimate noise?
- When is deconvolution impossible?

Answers

- **Why noise knowledge is essential:**
  Deconvolution requires dividing by $\Psi_\varepsilon(t)$. Without at least an estimate of this function, you cannot separate signal and noise in the frequency domain.
- **Using replicates:**
  If you have repeated measurements $W_1 = X + \delta_1$, $W_2 = X + \delta_2$ with independent errors, their difference removes X: $W_1 - W_2 = \delta_1 - \delta_2$. This isolates the noise and allows estimation of its distribution and characteristic function.
- **When deconvolution is impossible:**
  If there is no information about the noise distribution (no structural assumptions, no replicates, no validation data), then many different pairs (f_X, f_ε) can produce the same observed distribution of Y. The problem is not identifiable.

---

8. Practical Behavior, Rates, and Interpretation

Practical issues:

- Deconvolution estimators are highly sensitive to noise.
- Bandwidth selection is more delicate than in standard KDE.
- Supersmooth noise → extremely slow rates.

Key theoretical insight:

- Rates are limited by decay of $\Psi_\varepsilon(t)$.
- High noise variance can make the problem nearly non-estimable.

Oral questions:

- Why are deconvolution estimators much noisier than standard KDE?
- How do we interpret regression fits with contaminated covariates?
- What distinguishes well-posed vs ill-posed inverse problems?

Answers

- **Why noisier than standard KDE:**
  Standard KDE involves *multiplying* by a kernel in frequency space, which damps noise. Deconvolution requires *dividing* by $\Psi_\varepsilon(t)$, which amplifies noise exactly where $|\Psi_\varepsilon(t)|$ is small. Even after regularization, variance is much larger than in conventional smoothing.
- **Interpreting regression with contaminated covariates:**
  The deconvolution estimator attempts to reconstruct $m(x)$ as if we had clean X. In practice, curves will be rougher and more variable. Confidence bands need to reflect the additional uncertainty arising from measurement error.
- **Well-posed vs ill-posed inverse problems:**
  - **Well-posed:** small perturbations in data cause small changes in the solution (stable).
  - **Ill-posed:** small data changes can cause huge solution changes. Deconvolution, especially with supersmooth errors, is classically ill-posed: the inverse map from observed to latent density is extremely unstable without regularization.