Nonparametrics and Local Methods

Charlie Murry & Richard L. Sweeney

based on slides by Chris Conlon

Empirical Methods
Spring 2020

Section outline

Why nonparametrics?

Not always just interested in the mean of a (conditional) distribution.

- sometimes just interested in the distribution
- sometimes this is the first stage and we want to integrate
- sometimes want to do something semiparametric

In this section, we are interested in estimating the **density** $f(x)$ under minimal assumptions.

Examples from my own research

1. Estimation involves matching whole distribution (not just mean/variance).
   - Ciliberto, Tamer, and Murry – Estimating static full-info games.
   - Histogram
2. First step to recover distributions as input into structural estimation.
   - Gaurab, Murry, and Williams – Dynamic price discrimination in airline industry.
   - Kernel and NN.
3. Distribution is outcome of estimation.
   - Murry and Schurter – Estimate w.t.p. of used car purchasers. [like an auction]
   - semi-parametric regression.

Let's start with the histogram

One of the more successful and popular uses of nonparametric methods is estimating the density or distribution function $f(x)$ or $F(x)$.

$$\hat{f}_{HIST}(x_0) = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathbf{1}(x_0 - h < x_i < x_0 + h)}{2h}$$

- Divide the dataset into bins, count up fraction of observations in each bins.
- $2h$ is the length of a bin.

Kernel Estimation

Let's rewrite the histogram estimator

$$\hat{f}_{HIST}(x_0) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x_i - x_0}{h}\right)$$

Where $K(z) = \frac{1}{2} \cdot \mathbf{1}(|z| < 1)$

We can think of more general forms of $K(z; h)$.

Density estimator interpretation

- for each observation, there is probability mass 1 to spread around
- use the function $K(\cdot)$ and smoothing parameter h to choose how to allocate this mass
- then, for any given x_0, sum over these functions that spread out mass, and normalize by dividing by N

Smooth Kernels

We call $K(\cdot)$ a Kernel function and h the bandwidth. We usually assume

1. (i) $K(z)$ is symmetric about 0 and continuous.
2. (ii) $\int K(z)dz = 1$, $\int zK(z)dz = 0$, $\int |K(z)|dz < \infty$.
3. (iii) Either (a) $K(z) = 0$ if $|z| \geq z_0$ for some z_0 or (b) $|z|K(z) \to 0$ as $|z| \to \infty$.
4. (iv) $\int z^K(z)dz = \kappa$ where κ is a constant.

Smooth Kernels

We call $K(\cdot)$ a Kernel function and h the bandwidth. We usually assume

1. $K(z)$ is symmetric about 0 and continuous.
2. $\int K(z)dz = 1$, $\int zK(z)dz = 0$, $\int |K(z)|dz < \infty$.
3. Either (a) $K(z) = 0$ if $|z| \geq z_0$ for some z_0 or (b) $|z|K(z) \to 0$ as $|z| \to \infty$.
4. $\int z^K(z)dz = \kappa$ where κ is a constant.

Usually we choose a smooth, symmetric K. But a common nonsmooth choice: $K(x) = (|x| < 1/2)$ gives the *histogram* estimate.

Some Common Kernels

Table 9.1. *Kernel Functions: Commonly Used Examples*[a]

Kernel	**Kernel Function $K(z)$**	δ
Uniform (or box or rectangular)	$\frac{1}{2} \times \mathbf{1}(\|z\| < 1)$	1.3510
Triangular (or triangle)	$(1 - \|z\|) \times \mathbf{1}(\|z\| < 1)$	–
Epanechnikov (or quadratic)	$\frac{3}{4}(1 - z^2) \times \mathbf{1}(\|z\| < 1)$	1.7188
Quartic (or biweight)	$\frac{15}{16}(1 - z^2)^2 \times \mathbf{1}(\|z\| < 1)$	2.0362
Triweight	$\frac{35}{32}(1 - z^2)^3 \times \mathbf{1}(\|z\| < 1)$	2.3122
Tricubic	$\frac{70}{81}(1 - \|z\|^3)^3 \times \mathbf{1}(\|z\| < 1)$	–
Gaussian (or normal)	$(2\pi)^{-1/2} \exp(-z^2/2)$	0.7764
Fourth-order Gaussian	$\frac{1}{2}(3 - z)^2(2\pi)^{-1/2} \exp(-z^2/2)$	–
Fourth-order quartic	$\frac{15}{32}(3 - 10z^2 + 7z^4) \times \mathbf{1}(\|z\| < 1)$	–

Kernel Comparison

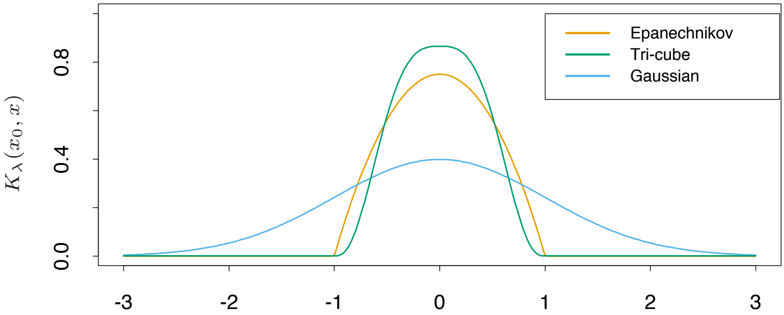

FIGURE 6.2. *A comparison of three popular kernels for local smoothing. Each has been calibrated to integrate to 1. The tri-cube kernel is compact and has two continuous derivatives at the boundary of its support, while the Epanechnikov kernel has none. The Gaussian kernel is continuously differentiable, but has infinite support.*

Mean and Variance of $\hat{f}(x_0)$

Assume that the derivative of $f(x)$ exists and is bounded, and $\int zK(z)dz = 0$

Then the estimator has **bias**

$$b(x_0) = E\left[\hat{f}(x_0)\right] - f(x_0) = \frac{1}{2}h^2 f''(x_0)\int z^2 K(z)dx$$

The **variance** of the estimator is

$$V\left[\hat{f}(x_0)\right] = \frac{1}{Nh}f(x_0)\int K(z)^2 dz \left\{+o(\frac{1}{Nh})\right\}$$

So, unsurprisingly, the bias is *increasing* in h, and the variance is *decreasing* in h.

How to Choose h

- We want both bias and variance to be as small as possible, as usual.
- In parametric estimation, it is not a problem: they both go to zero as sample size increases.
- In nonparametric estimation reducing h reduces bias, but increases variance; how are we to make his trade off?
- Note that how we set h is going to be much more important than the choice of $K(\cdot)$

Mean Integrated Square Error

- Start with the *local* performance at x_0

$$MSE\left[\hat{f}(x_0)\right] = E\left[\left(\hat{f}(x_0) - f(x_0)\right)^2\right]$$

- Calculate the *integrated* (as opposed to expected) squared error

$$\int \left(\hat{f}(x) - f(x)\right)^2 dx = \qquad \int \text{bias}^2\left(\hat{f}(x)\right) + \text{var}\left(\hat{f}(x)\right) dx$$

- Simple approximate expression (symmetric order 2 kernels):

$$(\text{bias})^2 + \text{variance} = Ah^4 + B/nh$$

with $A = \int \left(f''(x)\right)^2 \left(\int u^2 K\right)^2 /4$ and $B = f(x) \int K^2$

Optimal bandwidth

- The AMISE is

$$Ah^4 + B/nh$$

- Minimize by taking the FOC

$$h_n^* = \left(\frac{B}{4An}\right)^{1/5}$$

- bias and standard error are *both* in $n^{-2/5}$
- and the AMISE is $n^{-4/5}$—**not** $1/n$ as it is in parametric models.
- But: A and B both depend on K (known) and $f(y)$ (unknown), and especially "wiggliness" $\int (f'')^2$ (unknown, not easily estimated). Where do we go from here?

Optimal bandwidth

Can be shown that the optimal bandwidth is

$$h* = \delta \left(\int f''(x_0)^2 dx_0 \right)^{-0.2} N^{-0.2}$$

where δ depends on the kernel used (Silverman 1986) [these δ's are given in the kernel table]

Note the "optimal" kernel is Epanechnikov, although the difference is small.

Silverman's Rule of Thumb

- If f is normal with variance σ^2 (may not be a very appropriate benchmark!), the optimal bandwidth is
$$h_n^* = 1.06\sigma n^{-1/5}$$
- In practice, typically use **Silverman's plug-in estimate**:
$$h_n^* = 0.9 * \min(s, IQ/1.34) * n^{-1/5}$$
where IQ=interquartile distance
- Investigate changing it by a reasonable multiple.

Silverman's Rule of Thumb

- If f is normal with variance σ^2 (may not be a very appropriate benchmark!), the optimal bandwidth is
$$h_n^* = 1.06\sigma n^{-1/5}$$
- In practice, typically use **Silverman's plug-in estimate**:
$$h_n^* = 0.9 * \min(s, IQ/1.34) * n^{-1/5}$$
where IQ=interquartile distance
- Investigate changing it by a reasonable multiple.

This tends to work pretty well. But can we do better?

Why not search for optimal h in our data?

- Know we want to minimize MISE.
- One option is to find the h that minimizes it *in sample*
  - Loop through increments of h
  - Calculate MISE
- Example: Old Faithful R data
  - Waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.
  - See R code in this folder.

Cross-validation

- General concept in the whole of nonparametrics: choose h to minimize a criterion $CV(h)$ that approximates

$$AMISE(h) = \int E(\hat{f}_n(x) - f(x))^2 dx.$$

- Usually programmed in metrics software. *If you can do it, do it on a subsample, and rescale.*
- CV tries to measure what the expected out of sample (OOS or EPE) prediction error of a new never seen before dataset.
- The main consideration is to prevent overfitting.
  - In sample fit is always going to be maximized by the most complicated model.
  - OOS fit might be a different story.
  - ie 1-NN might do really well in-sample, but with a new sample might perform badly.

Sample Splitting/Holdout Method and CV

Cross Validation is actually a more complicated version of sample splitting that is one of the organizing principles in machine learning literature.

Training Set This is where you estimate parameter values.

Validation Set This is where you choose a model- a bandwidth h or tuning parameter λ by computing the error.

Test Set You are only allowed to look at this after you have chosen a model. Only Test Once: compute the error again on fresh data.

- Conventional approach is to allocate 50-80% to training and 10-20% to Validation and Test.
- Sometimes we don't have enough data to do this reliably.

Sample Splitting/Holdout Method

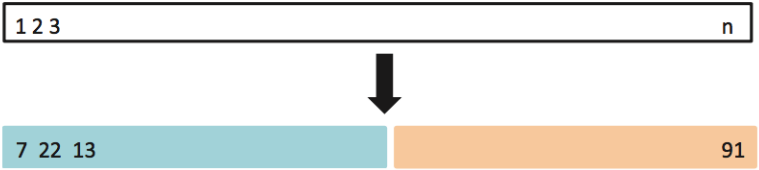

FIGURE 5.1. *A schematic display of the validation set approach. A set of n observations are randomly split into a training set (shown in blue, containing observations 7, 22, and 13, among others) and a validation set (shown in beige, and containing observation 91, among others). The statistical learning method is fit on the training set, and its performance is evaluated on the validation set.*

Challenge with Sample Splitting

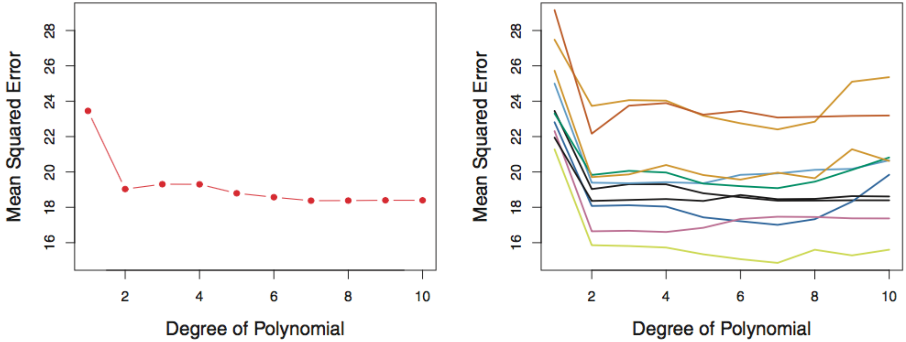

FIGURE 5.2. *The validation set approach was used on the* Auto *data set in order to estimate the test error that results from predicting* mpg *using polynomial functions of* horsepower. Left: *Validation error estimates for a single split into training and validation data sets.* Right: *The validation method was repeated ten times, each time using a different random split of the observations into a training set and a validation set. This illustrates the variability in the estimated test MSE*

k-fold Cross Validation

- Break the dataset into k equally sized "folds" (at random).
- Withhold $i = 1$ fold
  - Estimate the model parameters $\hat{\theta}^{(-i)}$ on the remaining $k-1$ folds
  - Predict $\hat{y}^{(-i)}$ using $\hat{\theta}^{(-i)}$ estimates for the ith fold (withheld data).
  - Compute $MSE_i = \frac{1}{k \cdot N} \sum_j (y_j^{(-i)} - \hat{y}_j^{(-i)})^2$.
  - Repeat for $i = 1, \ldots, k$.
- Construct $\widehat{MSE}_{k,CV} = \frac{1}{k} \sum_i MSE_i$

k-fold Cross Validation

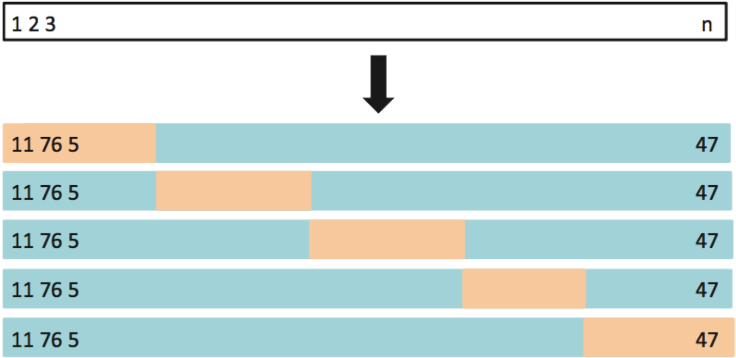

FIGURE 5.5. *A schematic display of 5-fold CV. A set of n observations is randomly split into five non-overlapping groups. Each of these fifths acts as a validation set (shown in beige), and the remainder as a training set (shown in blue). The test error is estimated by averaging the five resulting MSE estimates.*

Leave One Out Cross Validation (LOOCV)

Same as k-fold but with $k = N$.

- Withhold a single observation i
- Estimate $\hat{\theta}_{(-i)}$.
- Predict $\hat{y}_i$ using $\hat{\theta}^{(-i)}$ estimates
- Compute $MSE_i = \frac{1}{N}\sum_j (y_i - \hat{y}_i(\hat{\theta}^{(-i)}))^2$.

Note: this requires estimating the model N times which can be costly.

LOOCV vs k-fold CV

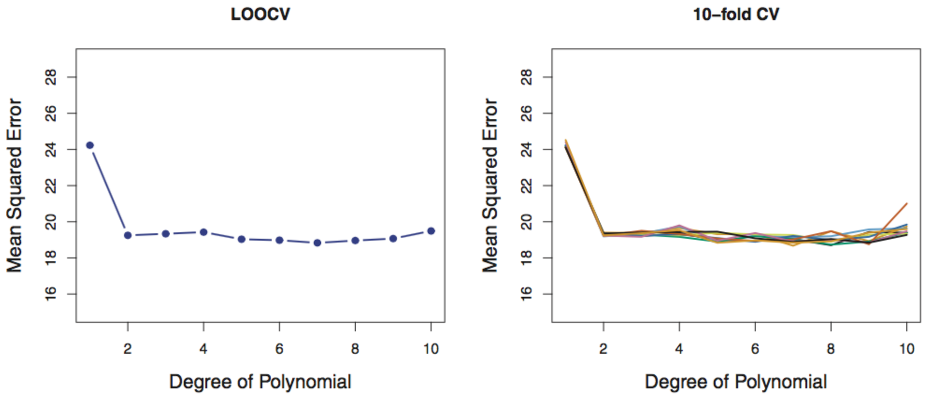

FIGURE 5.4. *Cross-validation was used on the* **Auto** *data set in order to estimate the test error that results from predicting* **mpg** *using polynomial functions of* **horsepower**. Left: *The LOOCV error curve.* Right: 10*-fold CV was run nine separate times, each with a different random split of the data into ten parts. The*

Cross Validation

- Main advantage of cross validation is that we use all of the data in both estimation and in validation.
  - For our purposes validation is mostly about choosing the right bandwidth or tuning parameter.
- We have much lower variance in our estimate of the OOS mean squared error.
  - Hopefully our bandwidth choice doesn't depend on randomness of splitting sample.

Test Data

- In Statistics/Machine learning there is a tradition to withhold 10% of the data as Test Data.
- This is completely new data that was not used in the CV procedure.
- The idea is to report the results using this test data because it most accurately simulates true OOS performance.
- We don't do much of this in economics.
  (Should we do more?)

Local Bandwidths

Local Bandwidths

If you only care about $f(y)$ at some given point, then

$$A = f''(y)^2 \left(\int u^2 K \right)^2 /4 \text{ and } B = f(y) \int K^2.$$

Local Bandwidths

If you only care about $f(y)$ at some given point, then

$$A = f''(y)^2 \left(\int u^2 K \right)^2 /4 \text{ and } B = f(y) \int K^2.$$

So in a low-density region, worry about variance and take h larger. In a curvy region, worry about bias and take h small.

Higher-Order Kernels

- K of order r iff $\int x^j K(x)dx = 0$ for $j < r$ and $\int x^r K(x)dx \neq 0$. Try $r > 2$?
- The beauty of it: bias in h^r if f is at least C^r...so AMISE can be reduced to $n^{-r/(2r+1)}$, almost $\sqrt{n}$-consistent if r is large.
- But gives wiggly (and sometimes negative) estimates $\rightarrow$ leave them to theorists.

Back to the CDF

Since now we have estimated the density with

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right),$$

Back to the CDF

Since now we have estimated the density with

$$\hat{f}_n(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right),$$

a natural idea is to integrate; let $\mathcal{K}(x) = \int_{-\infty}^{x} K(t)dt$, try

$$\hat{F}_n(y) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{K}\left(\frac{y - y_i}{h}\right)$$

as a reasonable estimator of the cdf in y.

Back to the CDF cont...

Reasonable estimator of the cdf in y?

$$\hat{F}_n(y) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{K}\left(\frac{y - y_i}{h}\right)$$

Very reasonable indeed:

- when $n \longrightarrow \infty$ and h goes to zero (at rate $n^{-1/3} \ldots$) it is consistent at rate $\sqrt{n}$
- it is nicely smooth
- by construction it accords well with the density estimator
- $\ldots$ it is a much better choice than the empirical cdf.

Example application: Auctions

- Why auctions?
- Great introduction to structural approach. Arguably the most successful application.
- Auctions are an example of a game with assymetric information: participants know the primitives of the game, but do not know their rivals exact valuations.
- By imposing rationality/ profit maximization, we can recover the distribution of values.
- Can then run counterfatuals
- Example: Asker (AER 2008) - stamp cartel

For more detail, check out Chris Conlon's slides in this folder, or John Asker's PhD lecture notes.

Setup

Let's consider the first price sealed bid (FPSB) auction

- bidders have **private information** (or type) scalar rv X_i with realization x_i
- signals are informative: $dE[U|x]/dx > 0$
- given their signal, they make a bid b_i
- if its the highest bid, recieve utility $[U|x_i, x_{-i}] - b_i$
- else get 0
- note that if we assume values are **independent**, we get $E[U|x_i, x_{-i}] = E[U|x_i]$

This introduces a tradeoff in first price auctions: Increasing the bid increases the probability of winning; but reduces your net utility from the object.

How to bid?

Denote the equilibrium bid as B_i, with realizations b_i

Perfect Bayes Nash equilibrium:

$$max_{\tilde{b}} \left(E\left[U_i | X_i = x_i\right] - b, max_{j \in N_{-i}} B_j \leq \tilde{b} \right)$$

$$Pr\left(max_{j \in N_{-i}} B_j \leq \tilde{b} | X_i = x_i \right)$$

See Athey and Haile or Krishna for an accessible derivation.

Can find bid as a function of primitives

$$v(x_i, \mathbf{x_i}; N) = b_i + \frac{G_{M_i|B_i}(b_i|b_i; N)}{g_{M_i|B_i}(b_i|b_i; N)}$$

where $G_{M_i|B_i}$ and $g_{M_i|B_i}$ are the CDF and PDF of the max bids given b_i and N

- we are typically interested in the LHS
- RHS is stuff we can observe or compute
- nice linear structure makes this easy to work with
- Guerre, Perrigne and Vuong (2000): can leverage assumption of equilibrium best response and invertibility of b to recover v

Estimation strategy I

[Assumption: FPSB with symmetric IPV]

1. Leverage independence assumption:

$$G_{M_i|B_i}(m_i|b_i;N) = G_{M_I|n}$$
$$= Pr(max_{j\neq i}B_j \leq m_i|n)$$

2. Value equation becomes

$$u = b + \frac{G_B\left(b|n\right)}{(n-1)g_B\left(b|n\right)}$$

where G_B and g_B are now the marginal distribution of equilibrium bids and the densities in n bidder auctions

Estimation strategy II

3. can estimate G and g using kernels
4. now have

$$\hat{u} = b + \frac{\hat{G}_B(b|n)}{(n-1)\hat{g}_B(b|n)}$$

5. finally, can recover the distribution of values with another kernel

$$\hat{f}(u) = \frac{1}{T_n h_f} \sum_{T=1} \frac{1}{n_t} \sum_{i=1}^{n} K\left(\frac{u_i - \hat{u}_{it}}{h_f}\right)$$

Algoritm

- for each b_o, estimate $\hat{G}_B(b_0|n)$ and $\hat{g}_B(b_0|n)$ using **all the data**
- infer $\hat{u}(b_0)$
- estimate $\hat{f}$
- plot bids, adjust bandwidth etc
- run counterfactuals