Topics on High-Dimensional Data Analytics

Functional Data Analysis

Kamran Paynabar, Ph.D.

Associate Professor

School of Industrial & Systems Engineering

Introduction to HD and Functional Data

Learning Objectives

- To understand the definition of high-dimensional data and Big Data.
- To explain the concepts of "curse of dimensionality" and "low-dimensional learning."
- To define Functional Data.

Big Data

The initial definition revolves around the three Vs:
Volume, Velocity, and Variety

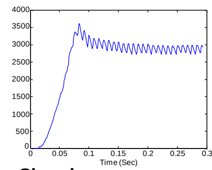
Volume: Large sample size, each sample could be high-dimensional. Use MapReduce, Hadoop, etc. when data are too large to be stored in one machine.

Velocity: Data is generated and collected very quickly. Increase computational efficiency.

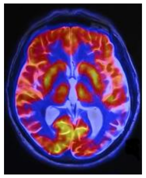
Variety: The data types you mention all take different shapes. How to deal with high-dimensional data, e.g., profiles, images, videos, etc.?

High-Dimensional Data

High-Dimensional data is defined as a data set with large number of attributes.



Signals
- ***100 KHz***



Images
- ***1M Pixels***

Videos
- ***Images Sequence***

Surveys
- ***Movies Ratings***

How to extract useful information from such massive datasets?

High-Dimensional Data vs. Big Data

	Small n	Large n
Small p	Traditional Statistics with limited samples	Classic large sample theory **Big Data Challenge**
Large p	HD Statistics and optimization High-Dimensional Data Challenge	Deep Learning and Deep Neural Networks

BD Analytics challenge: n is too large to be stored or processed on one machine.

- Solutions: big-data framework for data storage and computation (e.g., parallel computing, MapReduce, Hadoop, Spark, etc.)

High-Dimensional Data vs. Big Data

	Small n	Large n
Small p	Traditional Statistics with limited samples	Classic large sample theory Big Data Challenge
Large p	HD Statistics and optimization **High-Dimensional Data Challenge**	Deep Learning and Deep Neural Networks

HD Analytics challenge is mainly related to “curse of dimensionality”:

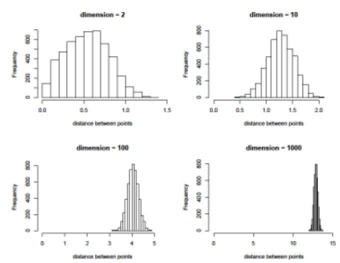
Computational issue: In some algorithm optimizations, we need $\left(\frac{!}{"}\right)^{\#}$ evaluations in order to obtain an solution within $ of the optimum.

Curse of Dimensionality

HD Analytics challenge is mainly related to "curse of dimensionality":

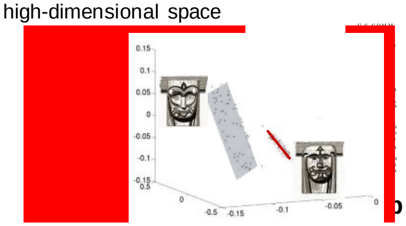
Model learning issue: As distance between observations increases with the dimensions, the sample size required for learning a model drastically increases.

– Solutions: Feature extraction and dimension reduction through low-dimensional learning .



Low-Dimensional Learning from High-Dimensional Data

- High-dimensional data usually have low dimensional structure
- Real data highly concentrated on low-dimensional, sparse, or degenerate structure in high-dimensional space



How can the LD structure be learned and exploited from HD data?

LD Learning Methods

Functional Data Analysis
- Splines
- Smoothing Splines
- Kernels

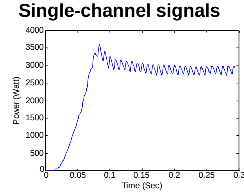
Tensor Analysis
- Multilinear Algebra
- Low Rank Tensor Decomposition

Rank Deficient Methods
- (Functional) Principal Component Analysis (FPCA)
- Robust PCA (RPCA)
- Matrix Completion

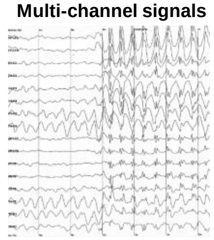
Functional Data

Definition: A fluctuating quantity or impulse whose variations represent information and is often represented as a function of time or space.

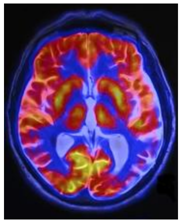
Single-channel signals



Multi-channel signals



Images



Point Cloud

Topics on High-Dimensional Data Analytics

Functional Data Analysis

Kamran Paynabar, Ph.D.
Associate Professor
School of Industrial & Systems Engineering

Review of Regression

Learning Objectives

- To review linear regression
- To understand geometric interpreation of linear regerssion
- To explain feature extraction using regression

Regression

- Observe a collection of i.i.d. training data

$$(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$$

Where x's are explanatory (independent) variables and y is the response (dependent) variable

- We want to build a function *f*(x) to model the relationship between x's and y

- An intuitive way of finding *f*(x) is by minimizing the following loss function

$$\min_{f(\boldsymbol{x})} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2$$

- We have to impose some constraints/structure on f(x), e.g.,

$$f(\boldsymbol{x}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

Regression – Least Square Estimates

! = #$ + &

! = [' (; ⋮ ; ' *]   # = [+((⋯ +(- ; ⋮ ⋱ ⋮ ; +*(⋯ +*-] $ = [/(; ⋮ ; /-]   & = [0(; ⋮ ; 0-]

We wish to find the vector of least squares estimators that minimizes:

1 = &'& = (! − #$)5(! − #$)

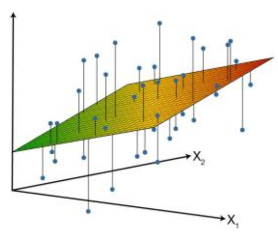
The resulting least squares estimate is

$̂ = (#5#)8(#5!

Example

Pull strength for a wire bond against
wire length and die height. (Montgomery and Runger 2006)

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon$$



Hastie. et al. 2009

$$\mathbf{X} = \begin{bmatrix} 1 & 2 & 50 \\ 1 & 8 & 110 \\ 1 & 11 & 120 \\ 1 & 10 & 550 \\ 1 & 8 & 295 \\ 1 & 4 & 200 \\ 1 & 2 & 375 \\ 1 & 2 & 52 \\ 1 & 9 & 100 \\ 1 & 8 & 300 \\ 1 & 4 & 412 \\ 1 & 11 & 400 \\ 1 & 12 & 500 \\ 1 & 2 & 360 \\ 1 & 4 & 205 \\ 1 & 4 & 400 \\ 1 & 20 & 600 \\ 1 & 1 & 585 \\ 1 & 10 & 540 \\ 1 & 15 & 250 \\ 1 & 15 & 290 \\ 1 & 16 & 510 \\ 1 & 17 & 590 \\ 1 & 6 & 100 \\ 1 & 5 & 400 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 9.95 \\ 24.45 \\ 31.75 \\ 35.00 \\ 25.02 \\ 16.86 \\ 14.38 \\ 9.60 \\ 24.35 \\ 27.50 \\ 17.08 \\ 37.00 \\ 41.95 \\ 11.66 \\ 21.65 \\ 17.89 \\ 69.00 \\ 10.30 \\ 34.93 \\ 46.59 \\ 44.88 \\ 54.12 \\ 56.63 \\ 22.13 \\ 21.15 \end{bmatrix}$$

Example cont.

(Montgomery and Runger 2006)

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 2 & 8 & \cdots & 5 \\ 50 & 110 & \cdots & 400 \end{bmatrix} \begin{bmatrix} 1 & 2 & 50 \\ 1 & 8 & 110 \\ \vdots & \vdots & \vdots \\ 1 & 5 & 400 \end{bmatrix} = \begin{bmatrix} 25 & 206 & 8{,}294 \\ 206 & 2{,}396 & 77{,}177 \\ 8{,}294 & 77{,}177 & 3{,}531{,}848 \end{bmatrix}$$

$$\mathbf{X}'\mathbf{y} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 2 & 8 & \cdots & 5 \\ 50 & 110 & \cdots & 400 \end{bmatrix} \begin{bmatrix} 9.95 \\ 24.45 \\ \vdots \\ 21.15 \end{bmatrix} = \begin{bmatrix} 725.82 \\ 8{,}008.47 \\ 274{,}816.71 \end{bmatrix}$$

"! = (%&%)0%&*

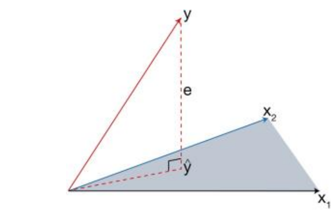
$$\begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix} = \begin{bmatrix} 25 & 206 & 8{,}294 \\ 206 & 2{,}396 & 77{,}177 \\ 8{,}294 & 77{,}177 & 3{,}531{,}848 \end{bmatrix}^{-1} \begin{bmatrix} 725.82 \\ 8{,}008.37 \\ 274{,}811.31 \end{bmatrix}$$

$$= \begin{bmatrix} 0.214653 & -0.007491 & -0.000340 \\ -0.007491 & 0.001671 & -0.000019 \\ -0.000340 & -0.000019 & +0.0000015 \end{bmatrix} \begin{bmatrix} 725.82 \\ 8{,}008.47 \\ 274{,}811.31 \end{bmatrix}$$

$$= \begin{bmatrix} 2.26379143 \\ 2.74426964 \\ 0.01252781 \end{bmatrix}$$

$$\hat{y} = 2.26379 + 2.74427x_1 + 0.01253x_2$$

Geometric interpretation



Hastie. et al. 2009

&%= (((* () , # (- & = . &

Projection Matrix (a.k.a. Hat matrix)

The outcome vector ! is orthogonally projected onto the hyperplane spanned by the input vectors "# and "$. The Projection %represents the vector of predictions obtained by the least square method

Properties of OLS Estimators

Unbiased estimators:

!($#) = ! [(()() * + (),]
= ! [(()() * + () (($ + /)]
= $

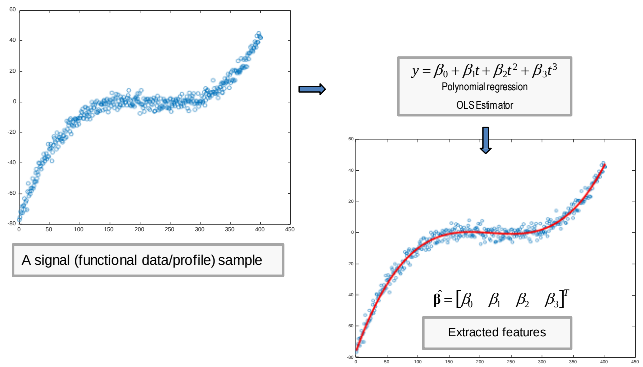
Covariance Matrix:

cov($#) = 3 4 (()() * +

35 4 = 6 6 ! / 7 − 9

According to the Gauss-Markov Theorem, among all unbiased linear estimates, the least square estimate (LSE) has the minimum variance and it is unique.

Feature Extraction Using Regression



Reference

- Montgomery, D. C., Runger, G., (2013), Applied Statistics and Probability for Engineers. 6th Edition. Wiley, NY, USA.
- Hastie, T., Tibshirani, R., and Friedman, J., (2009) The Elements of Statistical Learning. Springer Series in Statistics Springer New York Inc., New York, NY, USA.

Topics on High-Dimensional Data Analytics

Functional Data Analysis

Kamran Paynabar, Ph.D.

Associate Professor

School of Industrial & Systems Engineering

Splines

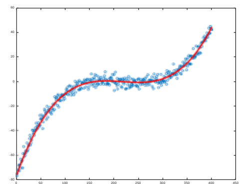
Learning Objectives

- To discuss local vs. global polynomial regression
- To explain splines and piecewise polynomial regression
- To recognize splines basis and truncated power basis

Polynomial Vs. Nonlinear Regression

mth-order polynomial regression

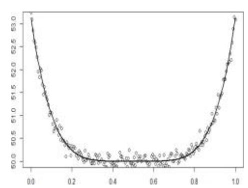
$$y=\beta_0+\beta_1x+\beta_2x^2+\beta_3x^3+!+\beta_mx^m+\varepsilon$$



Nonlinear Regression:

Often requires domain knowledge or first principles for finding the underlying nonlinear function

$$y=\begin{cases} a_1(x-c)^{b_1}+d+\varepsilon & x>c \\ a_2(-x+c)^{b_2}+d+\varepsilon & x\le c \end{cases}$$

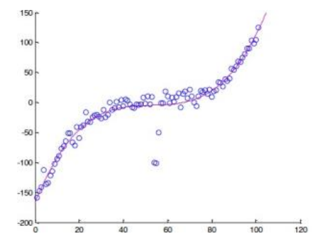


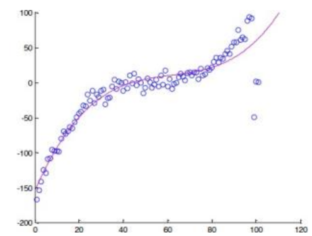
Polynomial Regression

mth-order polynomial $y=\beta_0+\beta_1x+\beta_2x^2+\beta_3x^3+!+\beta_mx^m$

Disadvantages of Polynomial Regression:

- Remote part of the function is very sensitive to outliers
- Less flexibility due to global functional structure

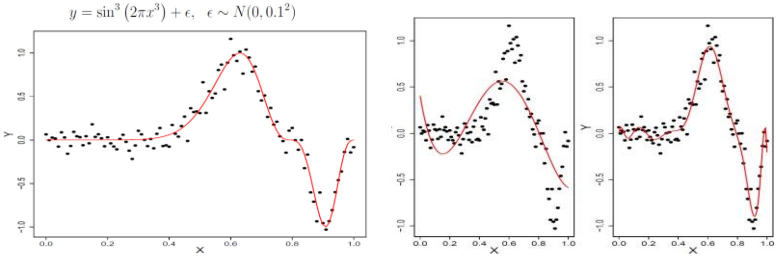




Polynomial Regression

Disadvantages of Polynomial Regression:
- Remote part of the function is very sensitive to outliers
- Less flexibility due to global functional structure

$y = \sin^3(2\pi x^3) + \epsilon, \quad \epsilon \sim N(0, 0.1^2)$



Example from Ji Zhou, 2011

Estimated using polynomials

Splines

- Linear combination of Piecewise polynomial functions under continuity assumption
- Partition the domain of x into continuous intervals and fit polynomials in each interval separately
- Provides flexibility and local fitting

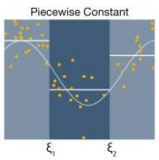
Suppose $x \in [a,b]$. Partition the x domain using the following points (a.k.a. knots).

$$a < \xi_1 < \xi_2 < \cdots < \xi_K < b \quad \xi_0 = a, \xi_{K+1} = b$$

Fit a polynomial in each interval under the continuity conditions and integrate them by

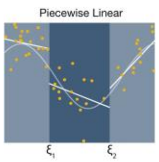
$$f(X) = \sum_{m=1}^{K} \beta_m h_m(X)$$

Splines – Simple Example



$$h_1(X) = I(X < \xi_1), \quad h_2(X) = I(\xi_1 \le X < \xi_2), \quad h_3(X) = I(\xi_2 \le X).$$

$$f(X) = \sum_{m=1}^{3} \beta_m h_m(X) \xrightarrow{\text{LSE}} \hat{\beta}_m = Y_m$$

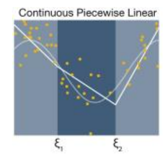


$$h_{m+3} = h_m(X)X, \; m = 1, \ldots, 3.$$

$$f(X) = \sum_{m=1}^{6} \beta_m h_m(X)$$

Image taken from: Hastie. et al. 2014

Splines – Simple Example

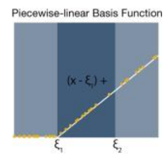


$$f(X) = \sum_{m=1}^{6} \beta_m h_m(X)$$

Impose continuity constraint for each knot:

$$f(\xi_1^-) = f(\xi_1^+) \Longrightarrow \beta_1 + \xi_1\beta_4 = \beta_2 + \xi_1\beta_5$$

Total number of free parameters (degrees of freedom) is 6-2=4



Alternatively, one could incorporate the constraints into the basis functions:

$$h_1(X) = 1, \quad h_2(X) = X, \quad h_3(X) = (X - \xi_1)_+, \quad h_4(X) = (X - \xi_2)_+,$$

This basis is known as truncated power basis

Image taken from: Hastie. et al. 2014

Splines with Higher Order of Continuity

Cubic Polynomials

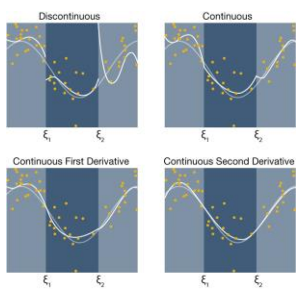


Image taken from: Hastie. et al. 2014

Continuity constraints for smoothness:

$$\begin{aligned} f(\xi_j^-) &= f(\xi_j^+) \\ f'(\xi_j^-) &= f'(\xi_j^+) \\ f''(\xi_j^-) &= f''(\xi_j^+),\ j = 1, \ldots, K \end{aligned}$$

$$\Downarrow$$

$$h_1(X) = 1, \quad h_3(X) = X^2, \quad h_5(X) = (X - \xi_1)_+^3,$$
$$h_2(X) = X, \quad h_4(X) = X^3, \quad h_6(X) = (X - \xi_2)_+^3.$$

splines df is calculated by
(# of regions)(# of parameters in each region) –
(# of knots)(# of constraints per knot)

Order-M Splines

Piecewise polynomials of order M-1, continuous derivatives up to order M-2

- M=1 piecewise-constant splines
- M=2 linear splines
- M=3 quadratic splines
- M=4 cubic splines

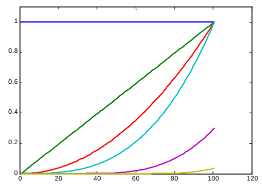
Truncated power basis functions:

$$\begin{aligned} h_j(X) &= X^{j-1},\ j = 1, \ldots, M, \\ h_{M+\ell}(X) &= (X - \xi_\ell)_+^{M-1},\ \ell = 1, \ldots, K. \end{aligned}$$

- Total degrees of freedom is K+M
- Cubic spline is the lowest order spline for which the knot discontinuity is not visible to human eyes
- Knots selection: a simple method is to use x quantiles. However, the choice of knots is a variable/model selection problem.

Estimation

$$h_j(X) = X^{j-1},\ j = 1, \dots, M,$$
$$h_{M+\ell}(X) = (X - \xi_\ell)_+^{M-1},\ \ell = 1, \dots, K.$$
$$f(X) = \sum_{m=1}^{K} \beta_m h_m(X)$$



Least square method can be used to estimate the coefficients

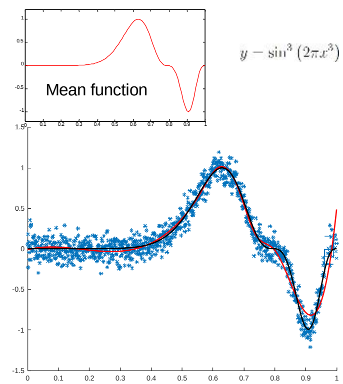
$$\mathbf{H} = [\mathbf{h}_1(\mathbf{x}) \quad \mathbf{h}_2(\mathbf{x}) \quad \mathbf{h}_3(\mathbf{x}) \quad \mathbf{h}_4(\mathbf{x}) \quad \mathbf{h}_5(\mathbf{x}) \quad \mathbf{h}_6(\mathbf{x})] \Rightarrow \hat{\boldsymbol{\beta}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$$

Linear Smoother: $\hat{\mathbf{y}} = \mathbf{H}\hat{\boldsymbol{\beta}} = \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y} = \mathbf{S}\mathbf{y}$

Degrees of Freedom: $df = \operatorname{trace} \mathbf{S}$

- Truncated power basis functions are simple and algebraically appealing.
- Not efficient for computation and ill-posed and numerically unstable. $\det(\mathbf{H}^T\mathbf{H})^{-1} = 1.3639\text{e-}06$

Example



%Data Generation
- X=[0:0.001:1];
- Y_true=sin(2*pi()*X.^3).^3;
- Y=sin(2*pi()*X.^3).^3+normrnd(0,0.1,1,length(X));

%Define knots and basis
- k = [0.5:0.596:6/7];
- h1=ones(1,length(X));h2=X;h3=X.^2;h4=X.^3;
- h5=(X-k(1)).^3;h5(h5<=0)=0;
- h6=(X-k(2)).^3;h6(h6<=0)=0;
- h7=(X-k(3)).^3;h7(h7<=0)=0;
- h8=(X-k(4)).^3;h8(h8<=0)=0;
- h9=(X-k(5)).^3;h9(h9<=0)=0;
- h10=(X-k(6)).^3;h10(h10<=0)=0;
- H=[h1' h2' h3' h4' h5' h6' h7' h8' h9' h10']

%Least square estimates
- B=(H'*H)\H'*Y'
- scatter(X,Y,'.'); hold on
- plot(X,H*B,'r')
- plot(X,Y_true,'k')

Topics on High-Dimensional Data Analytics

Functional Data Analysis

Kamran Paynabar, Ph.D.

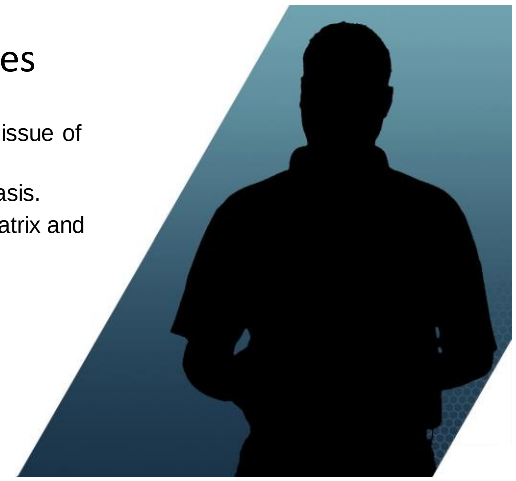
Associate Professor

School of Industrial & Systems Engineering

Bsplines

Learning Objectives

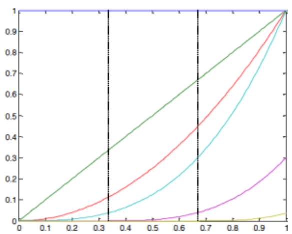
- To discuss computational issue of splines.
- To understand B-spline basis.
- To define the smoother matrix and degrees of freedom.



Computational Issue of Splines

- Truncated power basis functions are simple and algebraically appealing.
- Not efficient for computation and ill-posed and numerically unstable.

$$h_1(X) = 1, \quad h_3(X) = X^2, \quad h_5(X) = (X - \xi_1)^3_+,$$
$$h_2(X) = X, \quad h_4(X) = X^3, \quad h_6(X) = (X - \xi_2)^3_+.$$



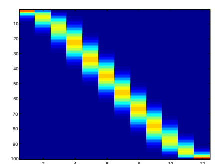
Cubic truncated power basis functions

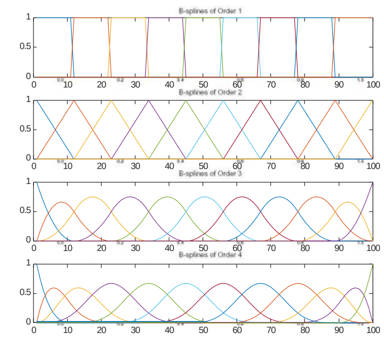
$$\det(\mathbf{H}^T\mathbf{H}) = 1.3639\text{e-}06$$

Bsplines

Alternative basis vectors for piecewise polynomials that are computationally more efficient (deBoor 1978)

- Each basis function has a local support, i.e., it is nonzero over at most M (spline order) consecutive intervals
- The basis matrix is banded





Bspline Basis

Let $B_{j,m}(x)$ be the j^{th} B-spline basis function of order m ($m \le M$) for the knot sequence τ

$$a < \xi_1 < \xi_2 < \cdots < \xi_K < b$$

Define the augmented knots sequence τ:

$\tau_1 \le \tau_2 \le \ldots \le \tau_M \le \xi_0$ e.g. $\tau_1 = \,! = \tau_M = \xi_0$

$\tau_{M+j} = \xi_j, j = 1, \ldots, K$

$\xi_{K+1} \le \tau_{M+K+1} \le \tau_{M+K+2} \le \ldots \le \tau_{2M+K}$ e.g. $\xi_{K+1} = \tau_{M+K+1} = \,! = \tau_{2M+K}$

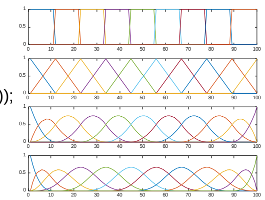
For j= 1,…, $2M+K-1$, $$B_{j,1}(x) = \begin{cases} 1 & \text{if } \tau_j \le x < \tau_{j+1} \\ 0 & \text{otherwise} \end{cases}$$

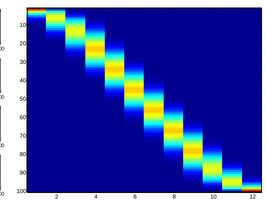
For j= 1,…, $2M+K-m$, $$B_{j,m}(x) = \frac{x - \tau_j}{\tau_{j+m-1} - \tau_j} B_{j,m-1}(x) + \frac{\tau_{j+m} - x}{\tau_{j+m} - \tau_{j+1}} B_{j+1,m-1}(x)$$

Bspline Basis in Matlab

```
n = 100
for sd = 1:4
    subplot(4,1,sd)
    knots = [ones(1,sd-1)...
linspace(1,n,10) n * ones(1,sd-1)];
    nKnots = length(knots) - sd;
    kspline = spmak(knots,eye(nKnots));
    B=spval(kspline,1:n)';
    plot(B)
end
```

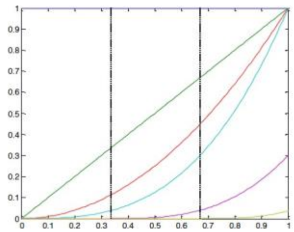
Show the matrix B,
low bandwidth matrix: imagesc(B)





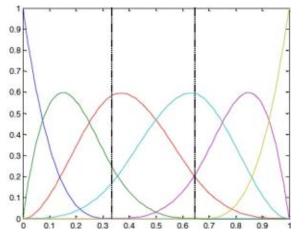
Generate bspline basis using R: bs(x, df, knots, intercept)

Example



Cubic truncated power basis functions

$\det(\mathbf{H}^T\mathbf{H}) = 1.3639\mathrm{e}-06$



Cubic Bspline basis functions

$\det(\mathbf{B}^T\mathbf{B}) = 1.4119\mathrm{e}+04$

Smoother Matrix

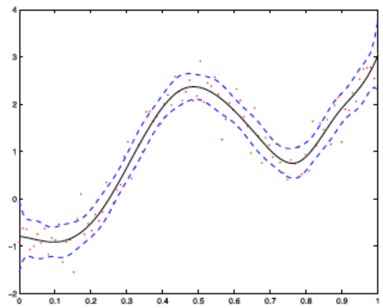
Consider a regression Spline basis B

$$\hat{\mathbf{f}} = \mathbf{B}\left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{y} = \mathbf{H}\mathbf{y}$$

- H is the smoother matrix (a.k.a. projection matrix)
- H is idempotent
- H is symmetric
- Degrees of freedom: trace (H)

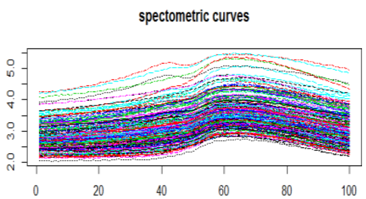
Example - MATLAB

```
% Generate data:
n = 100; D = linspace(0,1,n); sigma = 0.3;
fun = @(x) 2.5 * x - sin(10 * x) - exp(-10 * x);
y = fun(D) + randn(1,n)*sigma; y = y';
% Generate B-spline basis:
sd=4;
knots = [ones(1,3) linspace(1,n,10) n * ones(1,3)];
nKnots = length(knots) - sd;
kspline = spmak(knots,eye(nKnots));
B=spval(kspline,1:n)';
% Least Square Estimation:
yhat = B/(B'*B)*B'*y;
K= trace(B/(B'*B)*B')
sigma2 = 1/(n-K)*(y-yhat)'*(y-yhat);
yn = yhat-3*sqrt(diag(sigma2*B/(B'*B)*B'));
yp = yhat+3*sqrt(diag(sigma2*B/(B'*B)*B'));
plot(D,y,'r.',D,yn,'b--',D,yp,'b--',D,yhat,'k-')
```



Example: Fat content prediction

- A beef distributor wants to know the fat content of meat from spectrometric curves, which correspond to the absorbance measured at 100 wavelengths.
- She obtains the spectrometric curves for 215 pieces of finely chopped meat, (functional predictors).
- Additionally, through a time consuming chemical processing, she estimates the fat content of each piece (response).
- She wants us to build a model to predict the fat content of a new piece using the spectrometric curve.



The original dataset can be found at http://lib.stat.cmu.edu/datasets/tecator.

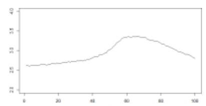
Example: Fat content

- We split the dataset into train dataset, 195 curves, and test dataset, 20 curves.
- Regular approach: We builds a linear regression using the 100 measurements from the spectrometer as predictors.
- Functional approach: We use B-spline to model each curve and extract features.
- The estimated B-spline coefficients are used as predictive features that can be used in building the fat regression model.
- The mean square errors of the predictions for the test dataset are:

!"#$%&'(= 27.02

!"#$.%&/01' = 14.25

Spectrometric Curves



B-Spline Coefficients

2.61	2.64	2.63	2.75	2.74
3.41	3.35	3.07	2.89	2.81

Fat Content

12.5

Reference

- Hastie, T., Tibshirani, R., and Friedman, J., (2009) The Elements of Statistical Learning. Springer Series in Statistics Springer New York Inc., New York, NY, USA.

Topics in High-Dimensional Analytics

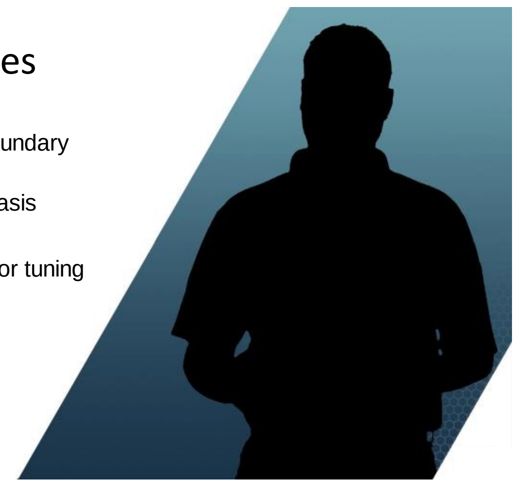
Functional Data Analysis

Kamran Paynabar, Ph.D.
Associate Professor
School of Industrial & Systems Engineering

Smoothing splines

Learning Objectives

- Discuss B-spline basis boundary issue
- Introduce natural spline basis
- Define smoothing splines
- Discuss cross-validation for tuning penalty parameter.

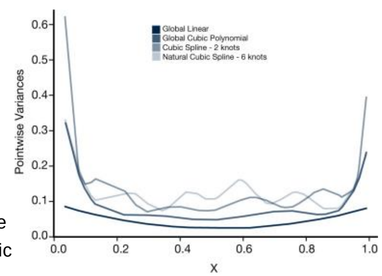


Boundary Effects on Splines

Consider the following setting with the fixed training data

$$\begin{aligned} y_i &= f(x_i) + \epsilon_i \\ \epsilon_i &\sim \mathrm{iid}(0, \sigma^2) \\ \mathrm{Var}(\hat{f}(x)) &= \boldsymbol{h}(x)^{\mathsf{T}}(\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H})^{-1}\boldsymbol{h}(x)\sigma^2 \end{aligned}$$

- Behavior of splines tends to be sporadic near the boundaries, and extrapolation can be problematic



From Hastie. et al. 2009

Natural Cubic Splines

Natural Cubic Splines

- Additional constraints are added to make the function linear beyond the boundary knots
- Assuming the function is linear near the boundaries (where there is less information) is often reasonable
- Cubic spline; linear on $(-\infty, \xi_1]$ and $[\xi_K, \infty)$
- Prediction variance decreases
- The price is the bias near the boundaries
- Degrees of freedom is K, the number of knots
- Each of these basis functions has zero second and third derivative in the linear region

$$N_1(x) = 1,\ N_2(x) = x,\ N_{k+2}(x) = d_k(x) - d_{K-1}(x) \qquad d_k(x) = \frac{(x-\xi_k)_+^3 - (x-\xi_K)_+^3}{\xi_K - \xi_k},\ k = 1, \ldots, K-2.$$

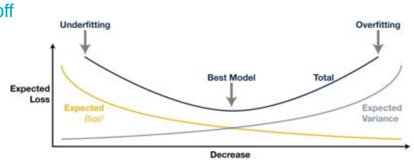
B = ns(x, df, intercept)

Smoothing Splines

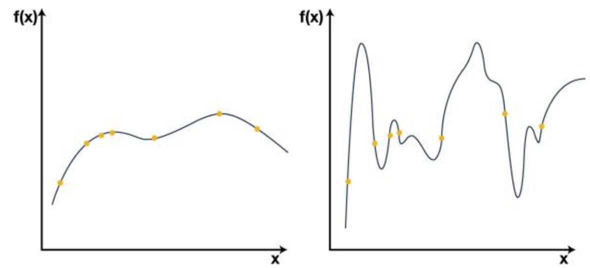
$$\min_f \frac{1}{n}\sum_{i=1}^{n}[y_i - f(x_i)]^2 + \lambda \int_a^b [f''(x)]^2 dx$$

- First term measures the closeness of the model to the data (related to bias).
- Second term penalizes curvature of the function (related to variance).
- Avoid knot selection. Select as many knots as the observations number.

- λ is smoothing parameter controlling trade off between bias and variance.
- λ = 0 interpolate the data (overfitting)
- λ = ∞ linear least-square regression



Example of Overfitting



True

Estimated curve with a large number of knots

Smoothing Splines

Penalized residual sum of squares

$$\min_f \frac{1}{n}\sum_{i=1}^{n}[y_i - f(x_i)]^2 + \lambda \int_a^b [f''(x)]^2 dx$$

It can be shown that the minimizer is a natural cubic spline: $\hat{f}(x) = \sum_{j=1}^{n} \theta_j N_j(x)$

Where N_j's are a set of natural cubic spline basis with knots at each of unique x_i's

Matrix form

$$\mathrm{RSS}(\boldsymbol{\theta}, \lambda) = (\boldsymbol{y} - \boldsymbol{N}\boldsymbol{\theta})^\intercal(\boldsymbol{y} - \boldsymbol{N}\boldsymbol{\theta}) + \lambda\boldsymbol{\theta}^\intercal\boldsymbol{\Omega}\boldsymbol{\theta}$$

$$\begin{aligned}\{\boldsymbol{N}\}_{ij} &= N_j(x_i) \\ \Omega_{jj'} &= \int N_j''(x)N_{j'}''(x)dx\end{aligned}$$

Solution

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{N}^\intercal\boldsymbol{N} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{N}^\intercal\boldsymbol{y}$$

Smoother Matrix

Smoothing spline estimator is a linear smoother

$$\begin{aligned}\hat{\boldsymbol{f}} &= \boldsymbol{N}(\boldsymbol{N}^\intercal\boldsymbol{N} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{N}^\intercal\boldsymbol{y} \\ &= \boldsymbol{S}_\lambda\boldsymbol{y}\end{aligned}$$

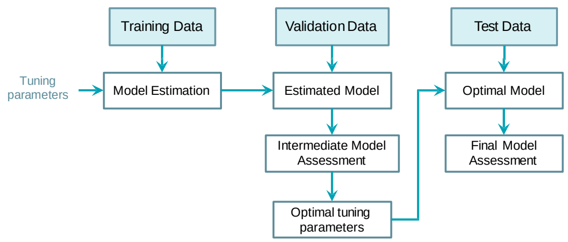
- S_λ is the smoother matrix
- S_λ is NOT idempotent
- S_λ is symmetric
- S_λ is positive definite

$$\boldsymbol{a}^\intercal\boldsymbol{\Omega}\boldsymbol{a} = \int [\sum_{j=1}^{n} a_j N_j''(x)]^2 dx \geq 0$$

- Degrees of freedom: trace (S_λ)

Choice of Tuning Parameter

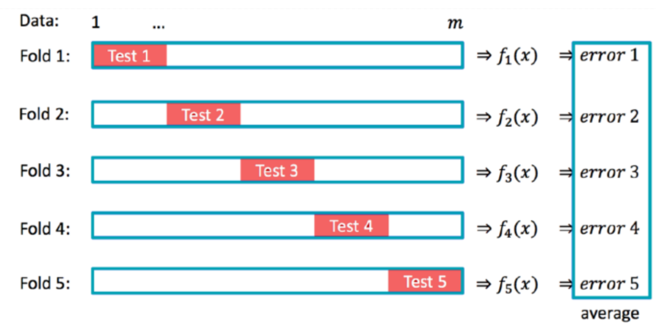
Collect 3 independent data sets for training, validation and test



- If an independent validation dataset is not affordable, the K-fold cross validation (CV) or leave-one-out CV can be used.

K-fold Cross-Validation (CV)

- 5-fold cross-validation (blank: training; red: test)



Choice of Tuning Parameter

Model Selection Criteria

- Akaike information criterion (AIC) $-2\log(L) + 2k$

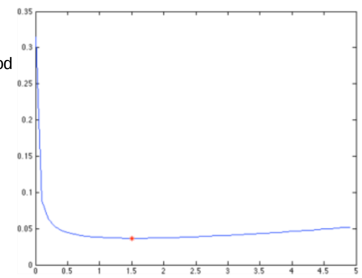
where k is the # of estimated parameters and L is Likelihood function

- Bayesian information criterion (BIC)

$-2\log(L) + k\log(n)$ where n is the sample size

- Generalized Cross-Validation (GCV)

$\hat{\lambda} = \arg\min_{\lambda} GCV(\lambda) = \arg\min_{\lambda} \frac{\|Y - \hat{Y}\|^2 / n}{(1 - n^{-1}\,tr(\hat{S}(\lambda)))^2}$



Example - Over-fitting

Generate 40 knots for fitting 100 data samples

- Generate data:

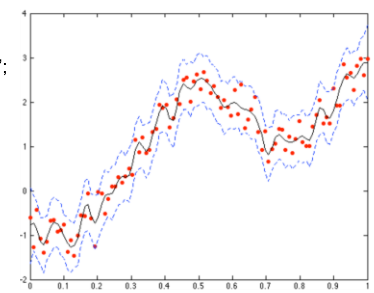
```
fun = @(x) 2.5 * x - sin(10 * x) - exp(-10 * x);
n = 100; D = linspace(0,1,n); k =40;
sigma = 0.3; y = fun(D) + randn(1,n)*sigma; y = y';
```

- Generate B-spline basis:

```
knots = [ones(1,2) linspace(1,n,k) n * ones(1,2)];
nKnots = length(knots) - 3;
kspline = spmak(knots,eye(nKnots));
B=spval(kspline,1:n)';
```

- Least Square Estimation:

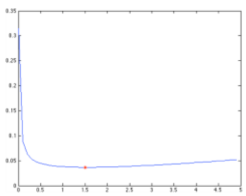
```
yhat = B/(B'*B)*B'*y;
sigma2 = 1/(n-K)*(y-yhat)'*(y-yhat);
yn = yhat-3*sqrt(diag(sigma2*B/(B'*B)*B'));
yp = yhat+3*sqrt(diag(sigma2*B/(B'*B)*B'));
plot(D,y,'r.',D,yn,'b--',D,yp,'b--',D,yhat,'k-')
```

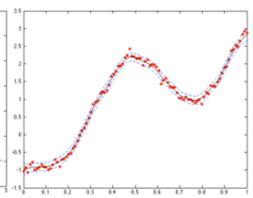


Example - Avoid Over-fitting by Smoothing Penalty

```
% B is defined in the previous slide
D1 = (B(2:n,:)-B(1:(n-1),:))/(1/(n-1));
D2 = (B1(2:(n-1),:)-B1(1:(n-2),:))/(1/(n-1));
%Different lambda selection
alllambda = 0.001:0.005:5;
n =length(alllambda);
RSS = zeros(n,1);
df = zeros(n,1);
for i = 1:n
   S = B/(B'*B+alllambda(i)*D2'*D2)*B';
   yhat = S*y;
   RSS(i) = sum((yhat-y).^2);
   df(i) = trace(S);
end
%GCV criterion
GCV = (RSS/n)./(1-df/n).^2;
plot(GCV)
```

$$\hat{\lambda} = \arg\min \ \mathrm{GCV}(\lambda) = \arg\min \ \frac{\|Y-\hat{Y}\|^2/n}{(1-n^{-1}\,\mathrm{tr}(\hat{S}(\lambda)))^2}$$





Reference

- Hastie, T., Tibshirani, R., and Friedman, J., (2009) The Elements of Statistical Learning. Springer Series in Statistics Springer New York Inc., New York, NY, USA.

Topics on High-Dimensional Data Analytics

Functional Data Analysis

Kamran Paynabar, Ph.D.
Associate Professor
School of Industrial & Systems Engineering

Kernel Smoothers

GTx

Learning Objectives

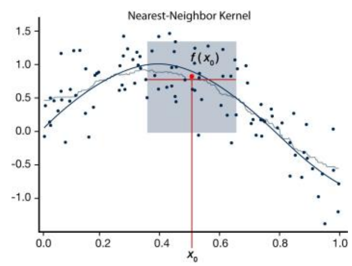
- To define kernel functions
- To understand KNN regression, weighted kernel regression, and linear and polynomial kernel regression.

K-Nearest Neighbor (KNN)

KNN Average $\hat{f}(x_0) = \sum_{i=1}^{n} w(x_0, x_i) y_i$

where $\sum_{i=1}^{n} w(x_0, x_i) = \begin{cases} \frac{1}{K} & \text{if } x_i \in N_k(x_0) \\ 0 & \end{cases}$

- Simple average of the *k* nearest observations to x_0 (local averaging)
- Equal weights are assigned to all neighbors
- The fitted function is in form of a step function (non-smooth function)



From Hastie. et al. 2009

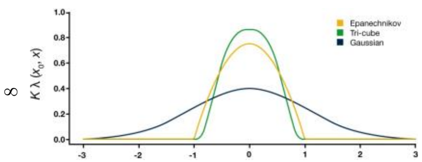
Kernel Function

Any non-negative real-valued integrable function that satisfies the following conditions:

1. $\int_{-\infty}^{\infty} K(u)du = 1$
2. K is an even function; $K(-u) = K(u)$
3. It has finite second moment; $\int_{-\infty}^{\infty} u^2 K(u)du < \infty$

Examples of Kernel functions

- Symmetric Beta family kernel
  - Uniform kernel (d=0) $K(u,d) = \frac{(1-u^2)^d}{2^{2d+1}B(d+1,d+1)} I(|u|<1)$
  - Epanechnikov kernel (d=1)
  - Bi/Tri-Weight (d=2,3)
- Tri-cube kernel $K(u) = (1-|u|^3)^3 I(|u|<1)$
- Gaussian kernel $K(u) = 1/\sqrt{2\pi} \exp(-u^2)$



From Hastie. et al. 2009

Kernel Smoother Regression

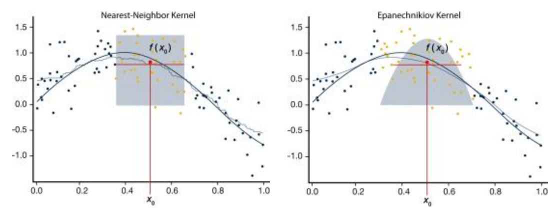
Kernel Regression

- Is weighted local averaging that fits a simple model separately at each query point x_0
- More weights are assigned to closer observations.
- Localization is defined by the weighting function.
- For any point

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{n} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{n} K_\lambda(x_0, x_i)} \quad \text{where} \quad K_\lambda(x_0, x_i) = K\left(|x_0 - x_i| / \lambda\right)$$

- K is a kernel function.
- λ is so-called "bandwidth" or "window width" that defines the width of neighborhood.
- Kernel regression requires little training; all calculations get done at the evaluation time.

Example - Kernel Smoother Regression



$$K_\lambda(x_0, x_i) = K\left(|x_0 - x_i| / \lambda\right)$$

$$K(u, d) = \frac{3}{4}(1 - u^2) I(|u| < 1)$$

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{n} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{n} K_\lambda(x_0, x_i)}$$

From Hastie. et al. 2009

Choice of λ

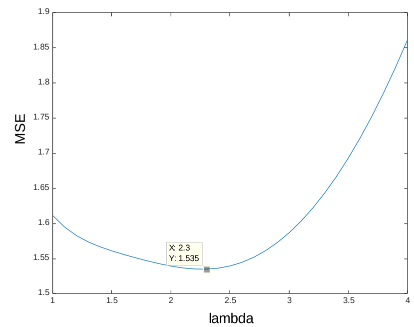
- λ defines the width of neighborhood.
- Only points within $[x_0 - \lambda, x_0 + \lambda]$ receive positive weights
- Smaller λ: rougher estimate, larger bias, smaller variance
- Larger λ: smoother estimate, smaller bias, larger variance

The following criteria can be used for determining of λ:

- Leave-one-out cross validation
- K-fold cross validation
- Generalized cross validation

Example – RBF Kernel

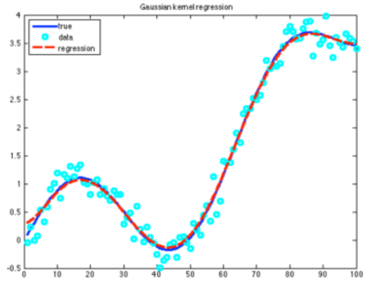
```
% Data Genereation
x=[0:100];
y=[sin(x/10)+(x/50).^2+0.1*normrnd(0,1,1,101)]';
kerf=@(z)exp(-z.*z/2)/sqrt(2*pi);
% leave-one-out CV
h1=[1:0.1:4];
for j=1:length(h1); h=h1(j) ;
   for i=1:length(y)
        X1=x;Y1=y;X1(i)=[];Y1(i)=[];
       z=kerf((x(i)-X1)/h); yke=sum(z.*Y1')/sum(z);
       er(i)=y(i)-yke;
   end
   mse(j)=sum(er.^2);
end
  plot(h1,mse); h=h1(find(mse==min(mse)));
```



Example – RBF Kernel

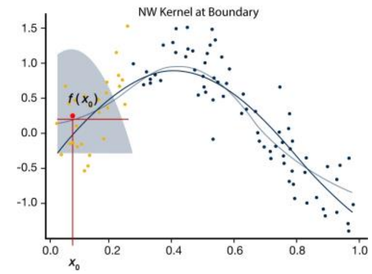
```
% Interpolation for N values
N=1000;
xall = linspace(min(x),max(x),N);

f = zeros(1,N);
for k=1:N
   z=kerf((xall(k)-x)/h);
   f(k)=sum(z.*y')/sum(z);
end
```



Drawbacks of Local Averaging

The local averaging can be biased on the boundaries of the domain due to the asymmetry of the kernel in that region.



From Hastie. et al. 2009

Local Linear Regression

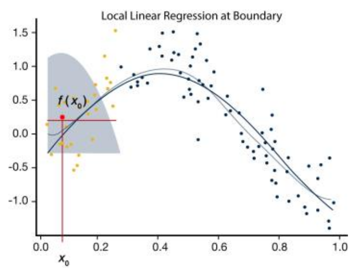
Locally weighted linear regression model is estimated by

$$\underset{\beta_0(x_0),\beta_1(x_0)}{\arg\min} \sum_{i=1}^{n} K_\lambda(x_0, x_i)\left[y_i - \beta_0(x_0) - \beta_1(x_0)x_i\right]^2$$

The estimate of the function at x_0 is then

$$\hat{f}(x_0) = \hat{\beta}_0(x_0) + \hat{\beta}_1(x_0)x_0$$

Local linear regression corrects the bias on the boundaries



From Hastie. et al. 2009

Local Polynomial Regression

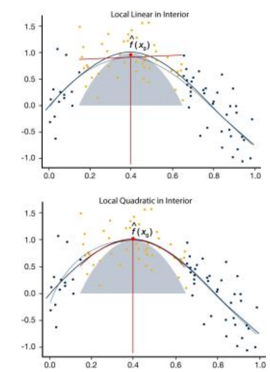
Locally weighted polynomial regression model is estimated by

$$\underset{\beta_0(x_0),\beta_1(x_0)}{\arg\min} \sum_{i=1}^{n} K_\lambda(x_0, x_i)\left[y_i - \beta_0(x_0) - \sum_{j=1}^{p}\beta_j(x_0)x_i^j\right]^2$$

The estimate of the function at x_0 is then

$$\hat{f}(x_0) = \hat{\beta}_0(x_0) + \sum_{j=1}^{p}\hat{\beta}_j(x_0)x_0^j$$

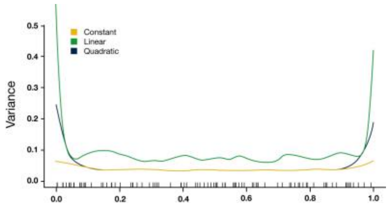
Local polynomial regression corrects the bias in curvature regions



From Hastie. et al. 2009

Local Polynomial Regression

- Higher-order polynomials result in lower of the bias, higher variance.
- Local linear fits can help reduce linear bias on the boundaries.
- Local quadratic fits are effective for reducing bias due to curvature in interior region, but not in boundary regions (increase the variance)



From Hastie. et al. 2009

Reference

- Hastie, T., Tibshirani, R., and Friedman, J., (2009) The Elements of Statistical Learning. Springer Series in Statistics Springer New York Inc., New York, NY, USA.

Topics on High-Dimensional Data Analytics

Functional Data Analysis

Kamran Paynabar, Ph.D.

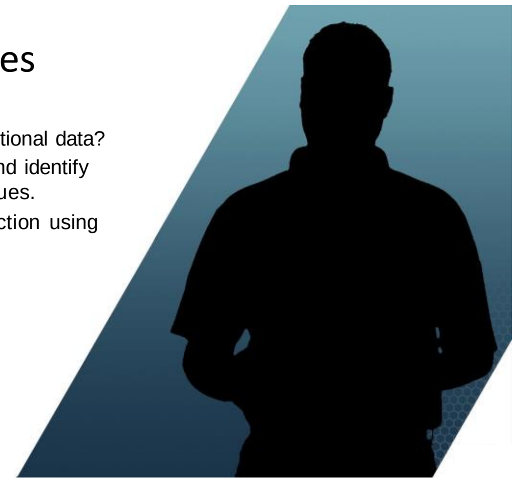
Associate Professor

School of Industrial & Systems Engineering

Functional Principal Component

Learning Objectives

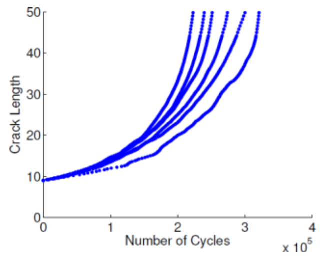
- How to perform PCA on functional data?
- To understand KL theorem and identify eigen-function and eigen-values.
- To demonstrate feature extraction using FPCA.



Signal Functional Form

!" ($) = , ($) + -"($)

- !"($): observed signals, &= 1, … , +
- , ($): continuous functional mean
- -"($): realizations from a stochastic process with mean function 0 and covariance function . ($,$′). It includes both random noise and signal-to-signal variations



Karhunen–Loeve Theorem

Using Karhunen–Loeve Theorem !(#) can be written as

!&(#) = : ξ&'/'(#)
';<
=

Where ξ&' are zero-mean and uncorrelated coefficients, i.e., ((ξ&') = 0 & ([(ξ&')'] = λ' and /'(#) are eigen-functions of the covariance function 0(#,#′) = cov(!(#), !(#′)) i.e.,

0(#,#′) = : >'/'(#)/'(#′)
';<
=

λ6 ≥ λ8 ≥ ⋯ are ordered eigen-values. The eigen-functions can be obtained by solving:

? 0(#,#′)/'(#)B# = >'/'(#′)
@
A

Functional PCA

The variance of ξ"# quickly decays with $. Therefore, only a few ξ"#, also known as FPC-scores, would be enough to accurately approximate the noise function. That is,

&"(') ≅) -"#.#(') (with "," above and "#*+" below the ")")

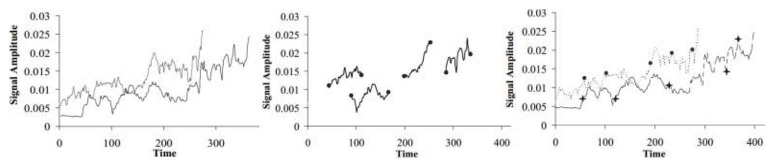
Signals decomposition is given by

1"(') = 3(') + &"(')

≅ 3(') + ∑#* + -"#. #(')

Model Estimation

- Complete signals: sampled regularly
- Incomplete signals: sampled irregularly, sparse, fragmented



Estimation of Mean Function

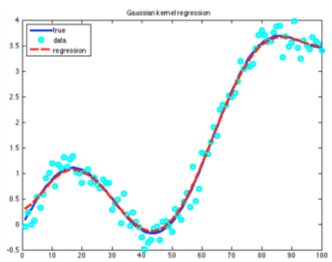
Historical signals !" (#$)

- %= 1, … , *, is the signal index
- += 1, … , , ", is the observation index in each signal
- !" (#$) ≅ . (#$) + ∑ 5"1 61 (#"$)

We can estimate mean function .^(#) using local linear regression by minimizing

min @ @ D ((#$ − #)/h) {!"(#$) − GH − (# − #$)G3}

– Solution: .^(#) = GH,J



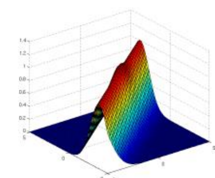
Estimation of Covariance Function

First, we use estimated mean functions to estimate the raw covariance function "!(#, #%):

'&9(#9>, #90) = (F9(#9>) − G(#9>)) (F9(#90) − G(#90))

To estimate the covariance surface &'(#, #%) we use local quadratic regression

min 8 8 B ((#9> − #)/h, (#90 − #')/h) {'&9(#9>, #90) −)* −); (# − #9>) −)E(#% − #90)}E



Solution: &'(#, #%) =)*(#, #')

Solve the estimated covariance function

/·0 (#) is estimated by discretizing the estimated covariance function "!(#, #%)

Computing FPC-Scores

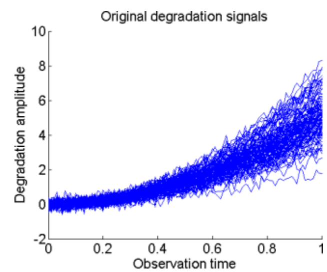
Computing eigen-function $\phi_k(t_j)$ by solving $\int_0^T \hat{C}(t, t')\phi_k(t)dt = \lambda_k \phi_k(t')$

- $\int_0^T \phi_k(t)\phi_m(t)dt = \begin{cases}1, & k = m\\ 0, & k \neq m\end{cases}$
- solved by discretizing the estimated covariance function $\hat{C}(t_j, t_{j'})$

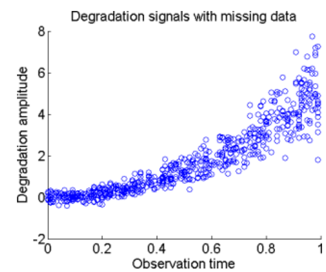
Computing FPC-scores ξ_{ik} $\quad \xi_{ik} = \int_0^T (S_i(t) - \hat{\mu}(t))\phi_k(t)dt$

- Numerical integration where $t_0 = 0$ $\quad \hat{\xi}_{ik} = \sum_{j=1}^{m} \left(S_i(t_j) - \hat{\mu}(t_j)\right)\hat{\phi}_k(t_j)(t_j - t_{j-1})$

FPCA Example

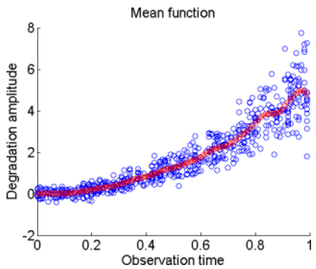


Original signals

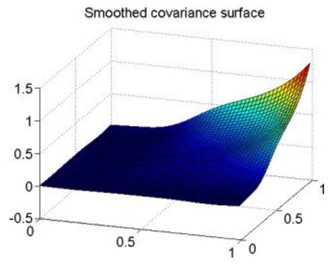


Signals with missing data (6 observations for each signal)

FPCA Example

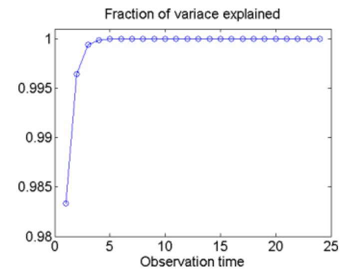


Mean function

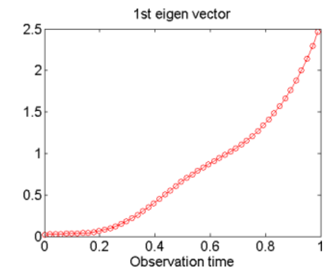


Smoothed covariance function

FPCA Example



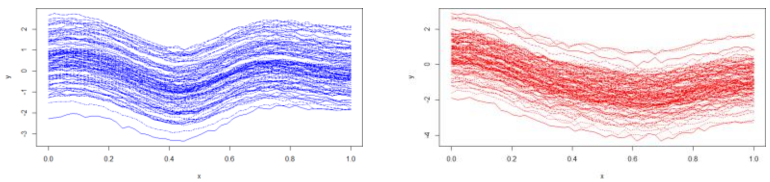
Fraction of variance explained
(1st eigen value explained more than 98% of the total variation)



1st eigen function

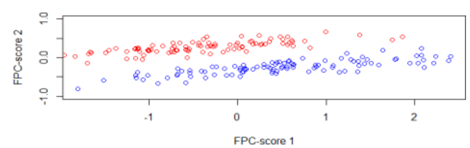
Example: Functional Data

- In a press machine the load profiles are measured during the forging process. The goal is to predict the quality of produced product based on the load profiles.
- There are 200 profiles along with their quality labels. 100 non-defective and 100 defective parts.



- For a new curve, we want to decide if it belongs to class 1 or to class 2.
  - Option 1: B-spline coefficients
  - Option 2: Functional principal components

Example: Functional Data Classification



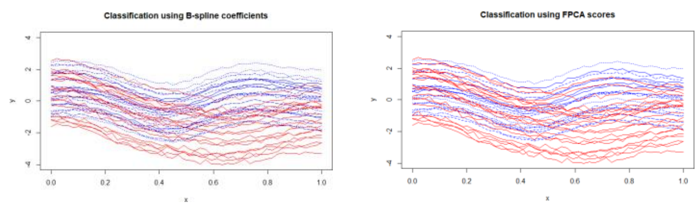
Step 1: Extract features from the functional data

- B-spline coefficients
- Functional principal components

Note: Each curve has 50 time observations, using B-splines we reduced the curve dimension from 50 to 10, and from 50 to 2 using FPCA scores.

Step 2: Train a classifier (e.g., Random Forest, SVM, etc.) using the extracted features.

Example: Functional Data Classification



Step 3. Predict the class for 40 new observations.

- Using 10 B-splines, all the curves were correctly classified.
- Using the scores of the first two principal components, 2 curves of class 1 were classified in class 2.