Functional Data Analysis (Lecture 2)

Zoltán Szabó

October 11, 2016

Reminder

Last time:

- Smoothing by least squares, kernel smoothing:

$$\hat{x}(t) = \langle \hat{\mathbf{c}}, \phi(t) \rangle, \; J(\mathbf{c}) = (\mathbf{y} - \mathbf{\Phi c})^T \mathbf{W} (\mathbf{y} - \mathbf{\Phi c}) \to \min_{\mathbf{c} \in \mathbb{R}^B},$$

$$\hat{x}(t) = \sum_{j=1}^{n} S_j(t) y_j, \; S_j(t) \leftarrow K, h.$$

- Regularization parameters:
  - $B = dim(\phi)$ and h. Choice: a few heuristics came up.

Reminder

Last time:

- Smoothing by least squares, kernel smoothing:

$$\hat{x}(t) = \langle \hat{\mathbf{c}}, \phi(t) \rangle, \; J(\mathbf{c}) = (\mathbf{y} - \mathbf{\Phi c})^T \mathbf{W} (\mathbf{y} - \mathbf{\Phi c}) \to \min_{\mathbf{c} \in \mathbb{R}^B},$$

$$\hat{x}(t) = \sum_{j=1}^{n} S_j(t) y_j, \; S_j(t) \leftarrow K, h.$$

- Regularization parameters:
  - $B = dim(\phi)$ and h. Choice: a few heuristics came up.

Today:

smoothing with roughness penalty (regularization).

Smoothing with roughness penalty

- Meaning of "smooth": explicitly expressed.
- Wide applicability.
- In practice: often better results (derivatives).

Roughness measures

Let D denote derivative. Curvature of x at t: $[D^2x(t)]^2$; zero for lines.

- $PEN_2(x) := \int [D^2x(t)]^2 dt \leftarrow$ roughness of x.

Roughness measures

Let D denote derivative. Curvature of x at t: $[D^2x(t)]^2$; zero for lines.

- $PEN_2(x) := \int [D^2x(t)]^2 dt \leftarrow$ roughness of x.
- $PEN_M(x) := \int [D^M x(t)]^2 dt \leftarrow$ roughness of $D^{M-2}x$.

Roughness measures

Let D denote derivative. Curvature of x at t: $[D^2x(t)]^2$; zero for lines.

- $PEN_2(x) := \int [D^2x(t)]^2 dt \leftarrow$ roughness of x.
- $PEN_M(x) := \int [D^Mx(t)]^2 dt \leftarrow$ roughness of $D^{M-2}x$.
- Harmonic acceleration operator: $Lx = D^3x + \omega^2 Dx$, ω: period $= \frac{2\pi}{\omega}$

$$Lx = 0 \Leftrightarrow x(t) = c_1 + c_2 \sin(\omega t) + c_3 \cos(\omega t).$$

Roughness measures

Let D denote derivative. Curvature of x at t: $[D^2x(t)]^2$; zero for lines.

- $PEN_2(x) := \int [D^2x(t)]^2 dt \leftarrow$ roughness of x.
- $PEN_M(x) := \int [D^M x(t)]^2 dt \leftarrow$ roughness of $D^{M-2}x$.
- Harmonic acceleration operator: $Lx = D^3x + \omega^2 Dx$, ω: period $= \frac{2\pi}{\omega}$

$$Lx = 0 \Leftrightarrow x(t) = c_1 + c_2 \sin(\omega t) + c_3 \cos(\omega t).$$

- More generally: linear differential operator

$$Lx = \sum_{j=0}^{M} \beta_j D^j x \rightarrow PEN_L(x) = \|Lx\|^2 = \int (Lx)^2(t) dt.$$

Smoothing by roughness penalty

- Idea: combine least squares with roughness penalty.

Smoothing by roughness penalty

- Idea: combine least squares with roughness penalty.
- Objective, $x(\mathbf{t}) := [x(t_1); \ldots; x(t_n)]$, $\lambda > 0$:

$$J(x) = \underbrace{[\mathbf{y} - x(\mathbf{t})]^T \mathbf{W} [\mathbf{y} - x(\mathbf{t})]}_{\text{least squares}} + \lambda \underbrace{\|Lx\|^2}_{\text{roughness of x}} \to \min_x$$

  - $\lambda \to 0$: interpolation, $x(t_j) \approx y_j$.
  - $\lambda \to \infty$: $Lx \approx 0$.

Smoothing by roughness penalty

- Idea: combine least squares with roughness penalty.
- Objective, $x(\mathbf{t}) := [x(t_1); \ldots; x(t_n)]$, $\lambda > 0$:

$$J(x) = \underbrace{[\mathbf{y} - x(\mathbf{t})]^T \mathbf{W}[\mathbf{y} - x(\mathbf{t})]}_{\text{least squares}} + \lambda \underbrace{\|Lx\|^2}_{\text{roughness of x}} \to \min_x$$

  - $\lambda \to 0$: interpolation, $x(t_j) \approx y_j$.
  - $\lambda \to \infty$: $Lx \approx 0$.

We got a variational problem ($\min_x$). Solution=?

Good news for $L = D^2$

'Carl de Boor: A Practical Guide to Splines, 2002': The minimum of

$$J(x) = [\mathbf{y} - x(\mathbf{t})]^T \mathbf{W} [\mathbf{y} - x(\mathbf{t})] + \lambda PEN_2(x) \to \min_x,$$

is a cubic spline with knots at t_j-s.

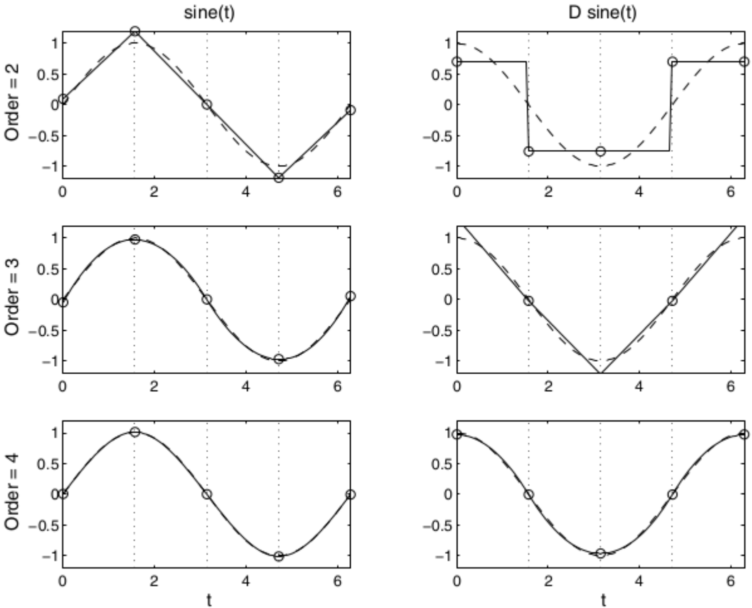
Now

We will

1. shortly review splines, B-spline basis, then
2. continue with the general case: PEN_L.

Splines

Spline: example



Spline: properties

- Divide the interval to L parts, with endpoints:

$$\tau_0, \tau_1, \ldots, \tau_{L-1}, \tau_L \leftarrow L + 1 \text{ points.}$$

- A spline is a polynomial of degree m on each interval, its
- $\leqslant m - 2$-order derivatives join up smoothly at the breakpoints.

Spline: properties

- Divide the interval to L parts, with endpoints:

$$\tau_0, \tau_1, \ldots, \tau_{L-1}, \tau_L \leftarrow L + 1 \text{ points.}$$

- A spline is a polynomial of degree m on each interval, its
- $\leqslant m - 2$-order derivatives join up smoothly at the breakpoints.

Example:

- order 4 cubic spline $\Rightarrow$ the 2nd derivative is a polygonal line.

Spline: degree of freedom

- Order 2 spline (=polygonal line) in the demo:

$$\underbrace{\color{blue}{2}}_{\color{blue}{\text{line}}} \times \underbrace{\color{red}{4}}_{\color{red}{\#\text{ of intervals}}} - \underbrace{3}_{\text{continuity constraints}} = 5.$$

Spline: degree of freedom

- Order 2 spline (=polygonal line) in the demo:

$$\underbrace{2}_{\text{line}} \times \underbrace{4}_{\#\text{ of intervals}} - \underbrace{3}_{\text{continuity constraints}} = 5.$$

- More generally:

$$\underbrace{m}_{\text{degree}} \times \underbrace{L}_{\#\text{ of intervals}} - \underbrace{(m-1)}_{D^0 s, \ldots, D^{m-2} s} \times \underbrace{(L-1)}_{L \text{ interval} \Rightarrow L-1 \text{ internal point}} =$$

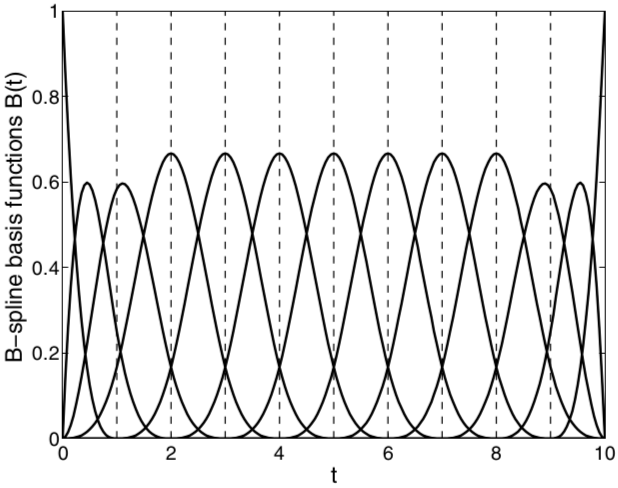
$$= m + (L-1)$$

$$= \text{order} + \text{number of internal points.}$$

Basis for splines

Multiple basis systems for splines. B-spline basis: let

- order 4 ($= m$), 9 equally space internal points ($L = 10$),
- $\xrightarrow{\text{formula}}$ degree of freedom $= m + L - 1 = 13$.



B-spline basis: properties

- Compact support: $\leqslant 4$ (or m) subintervals $\Rightarrow$ efficient computation.

B-spline basis: properties

- Compact support: $\leqslant 4$ (or m) subintervals $\Rightarrow$ efficient computation.
- Nested subspaces: for
  - new breakpoint or increased m.

B-spline basis: properties

- Compact support: $\leqslant 4$ (or m) subintervals $\Rightarrow$ efficient computation.
- Nested subspaces: for
  - new breakpoint or increased m.
- $\exists$: data-driven approaches for $\boldsymbol{\tau}$ choice, but expensive.
  - Cubic theorem: automatic $\boldsymbol{\tau}$.

Back to PEN_L-regularized problems

Back to PEN_L

- Recall the objective:

$$J(x) = [\mathbf{y} - x(\mathbf{t})]^T \mathbf{W} [\mathbf{y} - x(\mathbf{t})] + \lambda \|Lx\|^2 \to \min_x,$$
$$x(t) = \mathbf{c}^T \phi(t).$$

- Idea: rewrite $\|Lx\|^2$ to quadratic form in $\mathbf{c}$ $\Rightarrow$ ridge regression.

Rewrite PEN_L

$$PEN_L(x) = \int (Lx)^2(t)dt$$

Rewrite PEN_L

$$PEN_L(x) = \int (Lx)^2(t)dt \overset{(i)}{=} \int (L\mathbf{c}^T\phi)^2(t)dt$$

using the definition of x

Rewrite PEN_L

$$PEN_L(x) = \int (Lx)^2(t)dt \stackrel{(i)}{=} \int (L\mathbf{c}^T\phi)^2(t)dt \stackrel{(ii)}{=} \int (\mathbf{c}^T L\phi)^2(t)dt$$

using the definition of x, linearity of L

Rewrite PEN_L

$$PEN_L(x) = \int (Lx)^2(t)dt \overset{(i)}{=} \int (L\mathbf{c}^T\phi)^2(t)dt \overset{(ii)}{=} \int (\mathbf{c}^T L\phi)^2(t)dt$$
$$\overset{(iii)}{=} \int \mathbf{c}^T (L\phi)(t)(L\phi)^T(t)\mathbf{c}dt$$

using the definition of x, linearity of L, $(\mathbf{c}^T\mathbf{d})^2 = (\mathbf{c}^T\mathbf{d})(\mathbf{d}^T\mathbf{c})$.

Rewrite PEN_L

$$PEN_L(x) = \int (Lx)^2(t)dt \overset{(i)}{=} \int (L\mathbf{c}^T\phi)^2(t)dt \overset{(ii)}{=} \int (\mathbf{c}^T L\phi)^2(t)dt$$

$$\overset{(iii)}{=} \int \mathbf{c}^T(L\phi)(t)(L\phi)^T(t)\mathbf{c}dt = \mathbf{c}^T \underbrace{\left[\int (L\phi)(t)(L\phi)^T(t)\right]}_{=:\mathbf{R}=\left[R_{ij}\right]=\left[\int(L\phi_i)(t)(L\phi_j)(t)dt\right]} \mathbf{c},$$

using the definition of x, linearity of L, $(\mathbf{c}^T\mathbf{d})^2 = (\mathbf{c}^T\mathbf{d})(\mathbf{d}^T\mathbf{c})$.

Using the quadratic form of PEN_L

the objective becomes

$$J(\mathbf{c}) = (\mathbf{y} - \mathbf{\Phi c})^T \mathbf{W} (\mathbf{y} - \mathbf{\Phi c}) + \lambda \mathbf{c}^T \mathbf{R} \mathbf{c} \rightarrow \min_{\mathbf{c} \in \mathbb{R}^B}.$$

Ridge solution (J is quadratic in $\mathbf{c}$):

$$\hat{\mathbf{c}} = (\mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi} + \lambda \mathbf{R})^{-1} \mathbf{\Phi}^T \mathbf{W} \mathbf{y},$$
$$\hat{\mathbf{y}} = \mathbf{\Phi} \hat{\mathbf{c}} =: \mathbf{S}_\lambda \mathbf{y}.$$

Using the quadratic form of PEN_L

the objective becomes

$$J(\mathbf{c}) = (\mathbf{y} - \mathbf{\Phi c})^T \mathbf{W}(\mathbf{y} - \mathbf{\Phi c}) + \lambda \mathbf{c}^T \mathbf{R c} \to \min_{\mathbf{c} \in \mathbb{R}^B}.$$

Ridge solution (J is quadratic in $\mathbf{c}$):

$$\hat{\mathbf{c}} = (\mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi} + \lambda \mathbf{R})^{-1} \mathbf{\Phi}^T \mathbf{W} \mathbf{y},$$
$$\hat{\mathbf{y}} = \mathbf{\Phi} \hat{\mathbf{c}} =: \mathbf{S}_\lambda \mathbf{y}.$$

Degree of freedom (will be useful in λ-choice):

$$df(\lambda) = \mathrm{Tr}(\mathbf{S}_\lambda).$$

Two questions

1. Can we compute $\mathbf{R} = \int (L\phi)(t)(L\phi)^T(t)dt$?
2. How can one choose λ?

Two questions

1. Can we compute $\mathbf{R} = \int (L\phi)(t)(L\phi)^T(t)dt$?
   - $L = D^m$, traditional basis systems (B-spline, Fourier): ✓
   - General case: quadrature rules.
2. How can one choose λ?

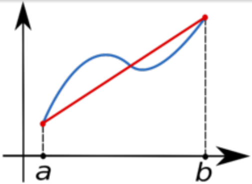
Two questions

1. Can we compute $\mathbf{R} = \int (L\phi)(t)(L\phi)^T(t)dt$?
   - $L = D^m$, traditional basis systems (B-spline, Fourier): ✓
   - General case: quadrature rules.
2. How can one choose λ?
   - cross-validation,
   - generalized cross-validation.

Two simple quadrature rules

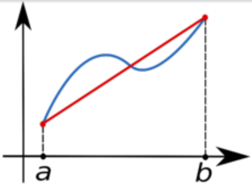
Trapezoid rule

Idea: $\int_a^b f(x)dx \approx (b-a)\frac{f(a)+f(b)}{2}$.



Trapezoid rule

Idea: $\int_a^b f(x)dx \approx (b-a)\frac{f(a)+f(b)}{2}$.

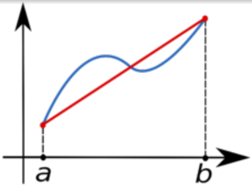


- For uniform grid: $a = x_1 < \ldots < x_{n+1} = b$:

$$\int_a^b f(x)dx \approx \frac{h}{2}\sum_{k=1}^{n}[f(x_k) + f(x_{k+1})]$$
$$= \frac{b-a}{2N}\left[f(x_1) + 2\sum_{k=2}^{n} f(x_k) + f(x_{n+1})\right].$$

Trapezoid rule

Idea: $\int_a^b f(x)dx \approx (b-a)\frac{f(a)+f(b)}{2}$.



- For uniform grid: $a = x_1 < \ldots < x_{n+1} = b$:

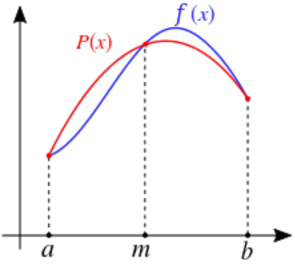
$$\int_a^b f(x)dx \approx \frac{h}{2}\sum_{k=1}^{n}[f(x_k) + f(x_{k+1})]$$
$$= \frac{b-a}{2N}\left[f(x_1) + 2\sum_{k=2}^{n} f(x_k) + f(x_{n+1})\right].$$

- Generally:

$$\int_a^b f(x)dx \approx \frac{1}{2}\sum_{k=1}^{n}(x_{k+1} - x_k)\left[f(x_{k+1}) + f(x_k)\right].$$

Simpson's rule

$\int_a^b f(x)dx \approx \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right].$



1. Replace f with a parabola interpolating at $a, m = \frac{a+b}{2}, b$:

$$P(x) = f(a)\frac{(x-m)(x-b)}{(a-m)(a-b)} + f(m)\frac{(x-a)(x-b)}{(m-a)(m-b)} + f(b)\frac{(x-a)(x-m)}{(b-a)(b-m)}$$

2. Approximation: $\int_a^b P(x)dx$.

(Generalized) cross-validation

Cross-validation

- Idea: in iteration

$$\text{data} = \underbrace{\text{training data}}_{\text{estimate model}} \cup \underbrace{\text{validation data}}_{\text{goodness of } \lambda}.$$

Cross-validation

- Idea: in iteration

$$\text{data} = \underbrace{\text{training data}}_{\text{estimate model}} \cup \underbrace{\text{validation data}}_{\text{goodness of } \lambda}.$$

- Extreme: leave-one-out cross-validation.

Cross-validation

- Idea: in iteration

$$\text{data} = \underbrace{\text{training data}}_{\text{estimate model}} \cup \underbrace{\text{validation data}}_{\text{goodness of } \lambda}.$$

- Extreme: leave-one-out cross-validation.
- Typically: $\log(\lambda)$ is scanned.

Cross-validation

- Idea: in iteration

$$\text{data} = \underbrace{\text{training data}}_{\text{estimate model}} \cup \underbrace{\text{validation data}}_{\text{goodness of } \lambda}.$$

- Extreme: leave-one-out cross-validation.
- Typically: $\log(\lambda)$ is scanned.
- Drawbacks:
  1. can be computationally expensive.
  2. prone to undersmoothing.

Generalized cross-validation (GCV)

- Motivation:
  1. avoid re-smoothing n times,
  2. less tendency to undersmooth.

Generalized cross-validation (GCV)

- Motivation:
  1. avoid re-smoothing n times,
  2. less tendency to undersmooth.
- Goodness of λ:

$$SSE(\lambda) = \sum_{j=1}^{n} [y_j - \hat{y}_j(\lambda)]^2, \ df(\lambda) = \text{Tr}(\mathbf{S}_\lambda),$$

$$GCV(\lambda) = \frac{n^{-1} SSE(\lambda)}{\left[n^{-1} \text{Tr}(\mathbf{I} - \mathbf{S}_\lambda)\right]^2} = \left(\frac{n}{n - df(\lambda)}\right)\left(\frac{SSE(\lambda)}{n - df(\lambda)}\right) \to \min_{\lambda > 0}.$$

$GCV(\lambda)$ is small: if $SSE(\lambda)$ and $df(\lambda)$ is so.

Bi-resolution analysis

- Two basis systems:
  1. $\{\phi_k\}$: capture large-scale features (smooth),
  2. $\{\psi_j\}$: for local features.

  Penalize on $Im(\{\psi_j\})$ only: $PEN_L(x_R)$.

Bi-resolution analysis

- Two basis systems:
  1. $\{\phi_k\}$: capture large-scale features (smooth),
  2. $\{\psi_j\}$: for local features.

  Penalize on $Im(\{\psi_j\})$ only: $PEN_L(x_R)$.
- Model, objective (ridge regression):

$$x = \sum_{k=1}^{B_1} c_j\phi_k + \sum_{j=1}^{B_2} d_j\psi_j =: x_S + x_R,$$

$$J(\mathbf{c},\mathbf{d}) = \|\mathbf{y} - \mathbf{\Phi}\mathbf{c} - \mathbf{\Psi}\mathbf{d}\|^2 + \lambda\mathbf{c}^T\mathbf{R}\mathbf{c} \to \min_{\mathbf{c},\mathbf{d}},$$

$$R_{ij} = \int L\psi_i(t)L\psi_j(t)dt.$$

Summary

PEN_L-regularized least squares:

- For $L = D^2$: solution $=$ cubic splines.
- Ridge regression.
- **R**: analytical formula/quadrature rules.
- λ-choice: (generalized) cross-validation.

We covered Chapter 5 from [1].