Smoothing functional data with constraints

Yoan Miche

CIS, HUT

February 6, 2007

Previously. . .

Basis expansion on the functions basis $\phi_k(t)$, $k \in [\![1, K]\!]$ for the functional inputs $x_i(t)$, $i \in [\![1, N]\!]$:

- $x_i(t) \approx \sum_{k=1}^{K} c_k \phi_k(t)$
- Approximation:
  - computation made easier
  - dimension reduction

$\Rightarrow$ Determine the c_k using a “correct” criterion

Outline

1. Smoothing data by Least Squares
2. Constrain smoothing by roughness penalty
3. Constrained functions

The setting...

Fit the observations $y_j, j \in [\![1, n]\!]$ using model $y_i = x(t_j) + \epsilon_j$ with $x(t)$ defined by basis expansion

$$x(t) = \sum_{k}^{K} c_k \phi_k(t) = \mathbf{c}^T \phi$$

Define by $\mathbf{\Phi}$ the $n \times K$ matrix with values $\phi_k(t_j)$

$$\mathbf{\Phi} = \begin{pmatrix} \phi_0(t_1) & \phi_1(t_1) & \dots & \phi_K(t_1) \\ \phi_0(t_2) & \phi_1(t_2) & \dots & \phi_K(t_2) \\ \vdots & & \ddots & \vdots \\ \phi_0(t_n) & \phi_1(t_n) & \dots & \phi_K(t_n) \end{pmatrix}$$

Ordinary Least Squares fits

SMSSE Criterion

$$\mathrm{SMSSE}(\mathbf{y}|\mathbf{c}) = \sum_{j=1}^{n} \left[y_j - \sum_{k}^{K} c_k \phi_k(t) \right]^2 = ||\mathbf{y} - \mathbf{\Phi c}||^2$$

Minimizing SMSSE criterion

Leads to a vector of fitted values

$$\hat{\mathbf{y}} = \mathbf{\Phi} \hat{\mathbf{c}} = \mathbf{\Phi}(\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \mathbf{y}$$

Appropriate for residuals iid, zero mean and constant variance

Ordinary Least Squares fits

SMSSE Criterion

$$\mathrm{SMSSE}(\mathbf{y}|\mathbf{c}) = \sum_{j=1}^{n} \left[y_j - \sum_{k}^{K} c_k \phi_k(t) \right]^2 = ||\mathbf{y} - \mathbf{\Phi c}||^2$$

Minimizing SMSSE criterion

Leads to a vector of fitted values

$$\hat{\mathbf{y}} = \mathbf{\Phi}\hat{\mathbf{c}} = \mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{y}$$

Appropriate for residuals iid, zero mean and constant variance

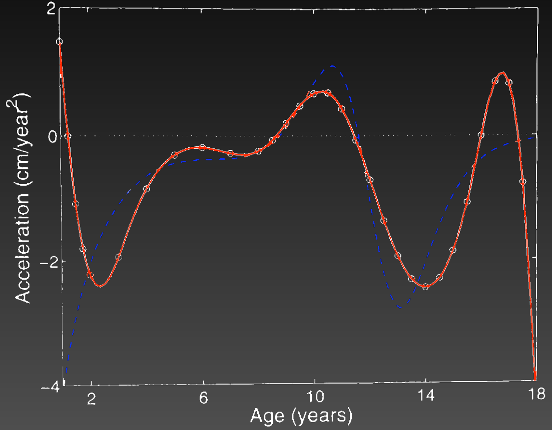
Least Squares fits in real case

Nonstationary/autocorrelated errors → differential weighting of residuals:

$$\text{SMSSE}(\mathbf{y}|\mathbf{c}) = (\mathbf{y} - \mathbf{\Phi c})^T \mathbf{W} (\mathbf{y} - \mathbf{\Phi c})$$

$\mathbf{W}$ symmetric positive definite. Best case, variance-covariance matrix $\mathbf{\Sigma}_e$ of residuals known, and $\mathbf{W} = \mathbf{\Sigma}_e^{-1}$

Least Squares fits in real case (2)



Estimate under for pubertal age. Boundaries get away → Needs improvements there

Choosing the number of Basis Functions K

Bias/Variance dilemma

- High K
  - High order of expansion$\longrightarrow$best fit to data
  - $\text{Bias}[\hat{x}(t)] = x(t) - E[\hat{x}(t)]$ is small
  - Fit of noise or wrong variations
- Low K
  - Miss of important aspects of estimated function
  - $\text{Var}[\hat{x}(t)] = E\left[(\hat{x}(t) - E(\hat{x}(t))^2\right]$ is small

Choosing the number of Basis Functions K (2)

The Mean Square Error Criterion

$$\text{MSE}\left[\hat{x}(t)\right] = E\left[\left(\hat{x}(t) - x(t)\right)^2\right]$$

or L^2 *loss function*: express clearly a quantity to be minimized

In practice

Hardly possible to minimize because requires knowledge of $x(t)$

$$\text{MSE}\left[\hat{x}(t)\right] = \text{Bias}^2\left[\hat{x}(t)\right] + \text{Var}\left[\hat{x}\right]$$

$\Longrightarrow$ Better tolerate some bias if we can have a sensible reduction of variance

Choosing the number of Basis Functions K (2)

The Mean Square Error Criterion

$$\text{MSE}\left[\hat{x}(t)\right] = E\left[\left(\hat{x}(t) - x(t)\right)^2\right]$$

or L^2 *loss function*: express clearly a quantity to be minimized

In practice

Hardly possible to minimize because requires knowledge of $x(t)$

$$\text{MSE}\left[\hat{x}(t)\right] = \text{Bias}^2\left[\hat{x}(t)\right] + \text{Var}\left[\hat{x}\right]$$

$\Longrightarrow$ Better tolerate some bias if we can have a sensible reduction of variance

Choosing the number of Basis Functions K (3)

Idea

- Have a good fit on the data: low residual sum of squares $\sum [y_j - x(t_j)]^2$
- But not too good to keep a low enough variance
- MSE: good way of expressing quality of estimate:
  - Sacrifice some bias $\Rightarrow$ lower variance
  - How to lower MSE? Roughness penalty

Choosing the number of Basis Functions K (3)

Idea

- Have a good fit on the data: low residual sum of squares $\sum [y_j - x(t_j)]^2$
- But not too good to keep a low enough variance
- MSE: good way of expressing quality of estimate:
  - Sacrifice some bias $\Rightarrow$ lower variance
  - How to lower MSE? Roughness penalty

Choosing the number of Basis Functions K (3)

Idea

- Have a good fit on the data: low residual sum of squares $\sum [y_j - x(t_j)]^2$
- But not too good to keep a low enough variance
- MSE: good way of expressing quality of estimate:
  - Sacrifice some bias $\Rightarrow$ lower variance
  - How to lower MSE? Roughness penalty

Defining the *roughness*

Curvature is the squared second derivative $\left[D^2 x(s)\right]^2$ of function $x(t)$

$$\mathrm{PEN}_m(x) = \int \left[D^m x(s)\right]^2 ds$$

m-th order in $\mathrm{PEN}_m(x)$: when derivatives data are the interest:

- Considering acceleration from position data:
- Requires 2-nd order derivative for acceleration
- And 4-th order derivative for acceleration curvature

Updated fitting criterion

From SSE to

$$\text{PENSSE}_\lambda(x|\mathbf{y}) = [\mathbf{y} - x(\mathbf{t})]^T \mathbf{W} [\mathbf{y} - x(\mathbf{t})]^2 + \lambda \text{PEN}_2(x)$$

λ smoothing parameter:

- λ small: fitted curve more variable (roughness penalty low)
- For $\lambda \to 0$: curve close to perfect interpolation of data (high variance)

Smoothing spline: a solution to PENSSE criterion is a piece-wise cubic spline with knots on the sample points.

Updated fitting criterion (2)

Previous expression of the data fitting vector $\hat{\mathbf{y}}$

$$\hat{\mathbf{y}} = \mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{W}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{W}\mathbf{y} = \mathbf{S}_\phi\mathbf{y}$$

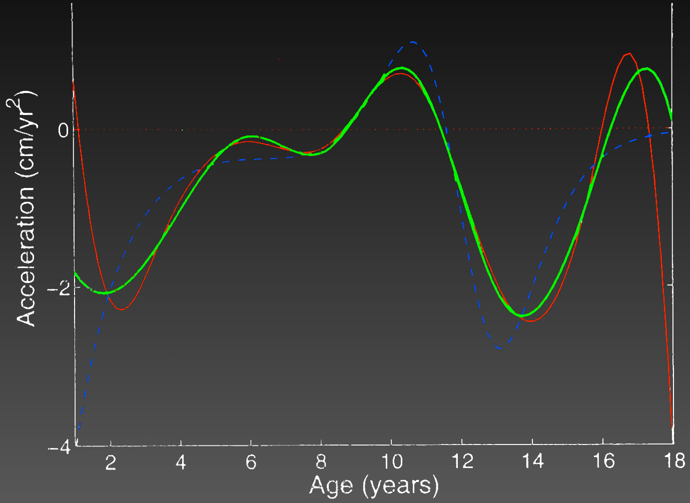
Now, with the new fitting criterion

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

with $\mathbf{R} = \int D^m\phi D^m\phi^T$

Form also useful to evaluate degrees of freedom of spline smooth
$\mathrm{df}(\lambda) = \mathrm{trace}(\mathbf{S}_{\phi,\lambda})$

Updated fitting criterion (3)



Choosing the smoothing parameter λ

Practically, solution of the linear system involving

$$\mathbf{M}(\lambda) = \mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi} + \lambda \mathbf{R}$$

poses computational limits because of derivative order used.

Proposed rule of thumb

$10||\mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi}|| < ||\lambda \mathbf{R}|| < 10^{10}||\mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi}||$

Choosing the smoothing parameter λ

Practically, solution of the linear system involving

$$\mathbf{M}(\lambda) = \mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi} + \lambda \mathbf{R}$$

poses computational limits because of derivative order used.

Proposed rule of thumb

$10||\mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi}|| < ||\lambda \mathbf{R}|| < 10^{10}||\mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi}||$

Cross-Validation and Generalized Cross-Validation

Cross-Validation is widely known: Take a subset of the whole data and make it the *validation* set. The rest is for the actual training: the *training set*.

Problems

- Can be computationally intensive (Leave-One-Out)
- Minimizing CV may under-smooth the data: may favor fitting noise or high freq (to be ignored for smoothing)

Cross-Validation and Generalized Cross-Validation

Cross-Validation is widely known: Take a subset of the whole data and make it the *validation* set. The rest is for the actual training: the *training set*.

Problems

- Can be computationally intensive (Leave-One-Out)
- Minimizing CV may under-smooth the data: may favor fitting noise or high freq (to be ignored for smoothing)

Cross-Validation and Generalized Cross-Validation

Introduction of the Generalized cross-validation criterion (GCV)

$$\mathrm{GCV}(\lambda) = \frac{n^{-1}\mathrm{SSE}}{\left[n^{-1}\mathrm{trace}(\mathbf{I} - \mathbf{S}_{\Phi,\lambda})\right]^2}$$

Cross-Validation and Generalized Cross-Validation

Minimization of GCV criterion

Find λ by grid-search or numerical optimization algorithm on

$$\mathrm{GCV}(\lambda) = \frac{n\,\mathrm{trace}(\mathbf{Y}^T[\mathbf{I} - \mathbf{S}_{\Phi,\lambda}]^{-2}\mathbf{Y})}{(\mathrm{trace}[\mathbf{I} - \mathbf{S}_{\boldsymbol{\Phi},\boldsymbol{\lambda}}])^2}$$

with $\mathbf{Y}$ the $n \times N$ data matrix, $\boldsymbol{\Phi}$ the $n \times K$ matrix of basis functions values

This can be made “easy” by some tricks to invert the $\mathbf{M}(\lambda)$ matrix.

Cross-Validation and Generalized Cross-Validation

The book presents an application of all this to a bi-resolution analysis: Two sets of basis functions.

Well detailled and interesting to see things in "action"

Why constrained functions?

Up to now:

- Smooth functions "constrained" with penalty
- Only thing required was: smoothness

What about constraints?

Need for being positive, monotone, represent a pdf or such: How to manage?
Book details four cases, we go through 2

Why constrained functions?

Up to now:

- Smooth functions "constrained" with penalty
- Only thing required was: smoothness

What about constraints?

Need for being positive, monotone, represent a pdf or such: How to manage?
Book details four cases, we go through 2

1. Fitting positive functions

Can be defined by an exponential (base does not matter)

$$x(t) = e^{W(t)}$$

with $W(t)$ an unconstrained function, that can thus be expanded to basis functions by

$$W(t) = \sum_k c_k \phi_k(t)$$

1. Fitting positive functions (2)

Not forgetting the roughness: defined as roughness of its logarithm, $W(t)$:

$$\text{PENSSE}_\lambda(W|\mathbf{y}) = \left(\mathbf{y} - e^{W(\mathbf{t})}\right)^T \mathbf{W} \left(\mathbf{y} - e^{W(\mathbf{t})}\right)^2 + \lambda \int [D^2 W(t)]^2 dt$$

Minimization of PENSSE criterion has now to be done numerically, by iterative decreases of initial estimate of $W(t)$

Convergence is fast even with values for $W(t)$ that differ greatly from final value.

2. Fitting monotone functions (quickly)

Again, express the condition by

$$Dx(t) = e^{W(t)}$$

thus

$$x(t) = C + \int_{t_0}^{t} e^{W(u)} du$$

and same ideas then...

Rest is skipped, please refer to the book for details

Conclusions

Some parts skipped in this presentation:

- Performance assessment;
- Confidence intervals estimation (functional probes,. . .);
- Localized least squares (kernel smoothing);
- Other things I forgot. . . =)

Some a bit heavy in math. sense, some not detailled in the book but may seem useful anyway.

You can look at confidence intervals estimation parts (4.6, 5.5).