Uncertainty Quantification: Data2LD for ODEs

Dr. Michelle Carey
University College Dublin

What is functional data?

Functional data analysis (FDA) concerns the analysis of information on curves or functions.

The key assumption is smoothness:

$$y_{ij} = x_i(t_j) + \epsilon_{ij}$$

- y_{ij} are discrete observations from the function x_i, at the points t_j
- ϵ_{ij} is typically assumed to be normally distributed with mean 0 and variance σ^2

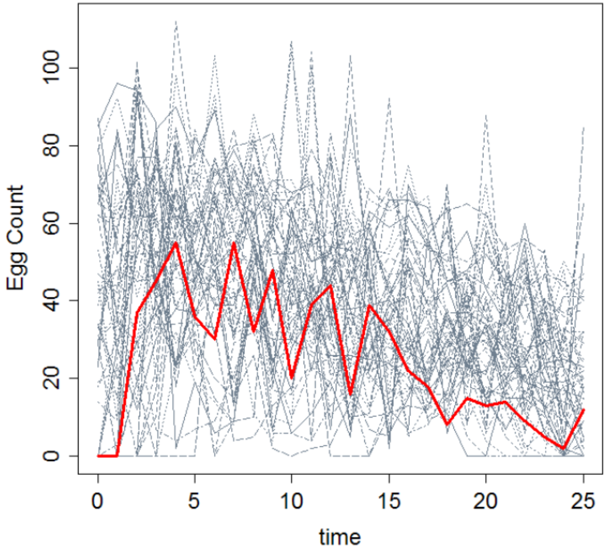
for $i = 1, \ldots, N$ and $j = 1, \ldots, n_i$

Medfly data

Records the number of eggs laid by Mediterranean Fruit Fly (Ceratitis capitata) over a 25 day period.

- Total of 50 flies.
- Assume egg count measurements relate to smooth process governing fertility.
- We would like to understand how fertility over the flies' lifetime influences lifespan.

Medfly data



From Discrete to Functional Data

Represent data recorded at discrete times as a continuous function in order to

- Allow the evaluation of a record at any time point (especially if the observation times are not the same across records).
- Evaluate the rates of change.
- Reduce noise.
- Allow registration onto a common time-scale.

From discrete to functional data

1. Representing data as a continuous function.

   - Basis-expansion methods:

$$x(t) = \sum_{k=1}^{K} \phi_k(t) c_k$$

   where

     - $\phi_k(t_j)$ are the basis functions evaluated at time t_j.
     - c_k are the corresponding coefficients.

From discrete to functional data

2. Reducing the noise in the measurements

   - Smoothing Penalties:

$$c = \operatorname{argmin} \sum_{i=1}^{N} \sum_{j=1}^{n_i} [y_{ij} - x_i(t_j)]^2 + \lambda \int [L(x_i(t))]^2 \, \mathrm{d}t$$

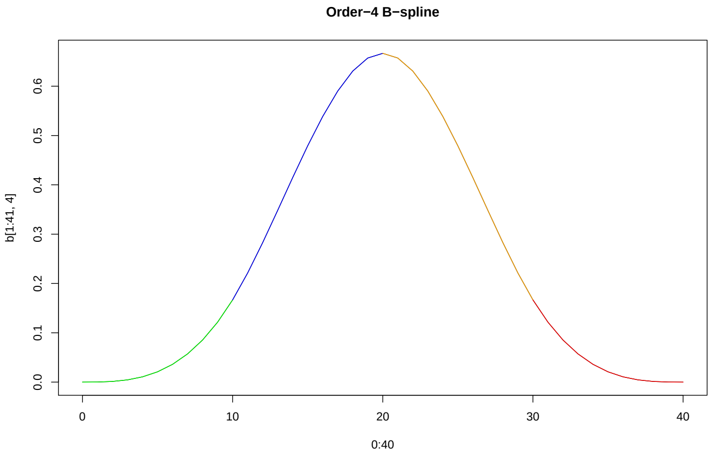
   - $[y_{ij} - x_i(t_j)]^2$: is a measure of x's fidelity to the data.
   - $L(x_i(t))$: measures the regularity of x_i.
   - λ is a "smoothing parameter" that controls the trade off between the fit to the data and the regualirty of x_i.

Splines

Spline functions are formed by joining polynomials of degree $m+1$ together at fixed points called knots.

In order to induce smoothness at the knots, the functions are constrained by the requirement that at each knot the derivatives up to $m-2$ must be equivalent for the adjoining functions.

Splines



Splines

Splines are defined by:

1. The order m (order $=$ degree$+1$) of the polynomial
2. The location of the knots.

How do I choose the order m of a spline?

The order m that we use depends on how many derivatives we will need to compute from the spline function.

A cubic spline is smooth in itself, but its first derivative will appear to change with noticeable abruptness at the breakpoints, and its second derivative will be a polygonal line.

If we want a smooth second derivative, we must use a spline with an order of at least six, so that its second derivative will be at least as smooth as a cubic spline.

How do I choose the knots?

The more knots, the more flexible the spline and we put less knots where we don't need much curvature.

There should be at least one observed value t_j within each subinterval.

The problem of deciding exactly where to position knots is often determined in one of two ways:

1. Make the knots equally spaced, paying attention to the requirement of having at least one observation in every subinterval.
2. Place a knot at every fixed number of observed values of y i.e. quantile placement.

Splines

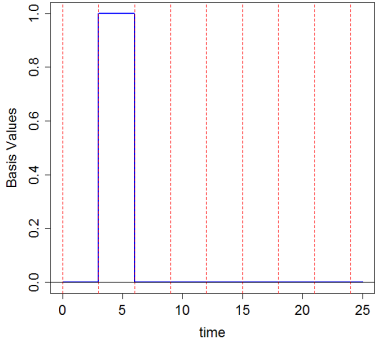
$$\begin{aligned} \mathbf{x} &= \sum_{k=1}^{K} \phi_k(\mathbf{t}) c_k \\ \mathbf{x} &= \mathbf{\Phi c} \end{aligned}$$

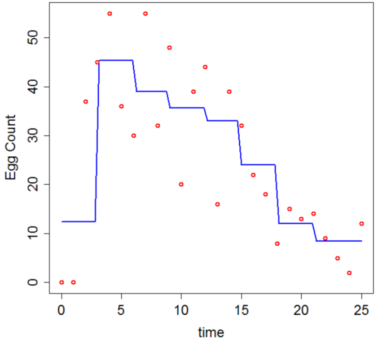
where

- $\mathbf{\Phi}$ is a $n \times K$ matrix with entries $\phi_k(t_j)$
- $\mathbf{c}$ is a $K \times 1$ vector with elements c_k

Splines

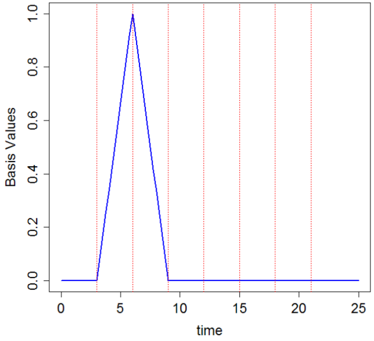
Splines of order 1: piecewise constant, discontinuous derivative.

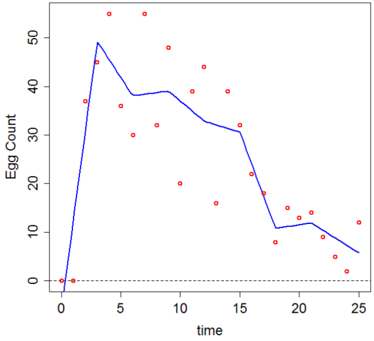




Splines

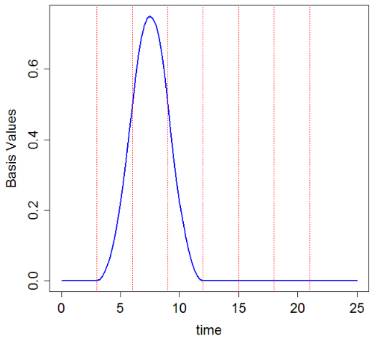
Splines of order 2: piecewise linear, continuous derivative.

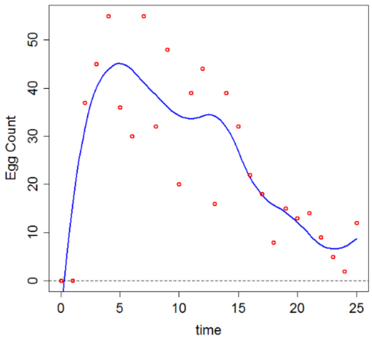




Splines

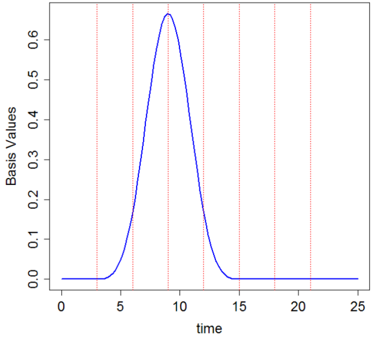
Splines of order 3: piecewise quadratic, continuous derivative.

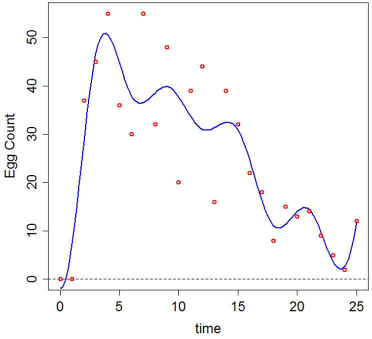




Splines

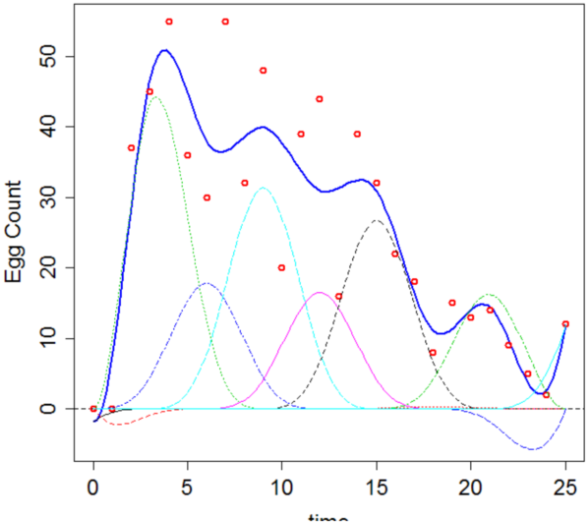
Splines of order 4: piecewise cubic, continuous 2nd derivative.





Splines

$\hat{\mathbf{f}}$ is the sum of the scaled basis functions.

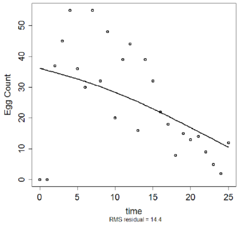


What do we mean by smoothness?

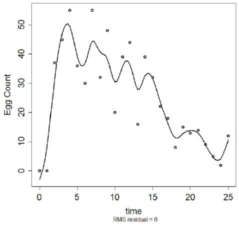
Some things are clearly smooth e.g. constants and straight lines.

What we really want to do is eliminate small "wiggles" in the data while retaining the right shape

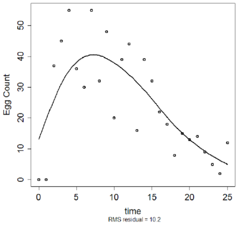
Too smooth



Too rough



Just right



Curvature

The derivative

- $\frac{\mathrm{d}x(t)}{\mathrm{d}t}$ provides the instantaneous slope of $x(t)$
- $\frac{\mathrm{d}^2x(t)}{\mathrm{d}t^2}$ is the curvature of $x(t)$

We measure the size of the curvature for all of x by

$$\int_{t_0}^{t_{max}} \left(\frac{\mathrm{d}^2x(t)}{\mathrm{d}t^2}\right)^2 \mathrm{d}t$$

Smoothing spline

$$c = \operatorname{argmin} \left\{ \sum_{j=1}^{n} (y_j - x(t_j))^2 + \lambda \int \left[\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} \right]^2 \mathrm{d}t \right\}$$

The trade off between the fit to the data and smoothness of the fit is controlled by the smoothing parameter λ.

- $\lambda \to 0$ leads to the least squares estimate of x.
- $\lambda \to \infty$ leads to a straight line estimate of x

Penalized least squares

$$\begin{aligned}
\mathrm{PENSSE}(\mathbf{c}) &= \sum_{j=1}^{n} (y_j - x(t_j))^2 + \lambda \int_{t_0}^{t_{max}} \left[\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} \right]^2 \mathrm{d}t \\
&= (\mathbf{y} - \mathbf{\Phi c})^T (\mathbf{y} - \mathbf{\Phi c}) + \lambda \int_{t_0}^{t_{max}} \left(\frac{\mathrm{d}^2 (\mathbf{\Phi c})}{\mathrm{d}t^2} \right)^2 \mathrm{d}t \\
&= (\mathbf{y} - \mathbf{\Phi c})^T (\mathbf{y} - \mathbf{\Phi c}) + \lambda \mathbf{c}' \left[\int_{t_0}^{t_{max}} \frac{\mathrm{d}^2 \mathbf{\Phi}}{\mathrm{d}t^2}^T \frac{\mathrm{d}^2 \mathbf{\Phi}}{\mathrm{d}t^2} \mathrm{d}t \right] \mathbf{c} \\
&= (\mathbf{y} - \mathbf{\Phi c})^T (\mathbf{y} - \mathbf{\Phi c}) + \lambda \mathbf{c}' \mathbf{R} \mathbf{c}
\end{aligned}$$

Penalized Least Squares

The penalized least squares estimate for **c** is

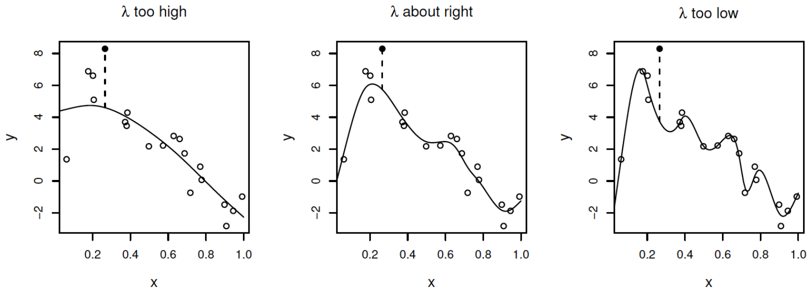
$$\hat{\mathbf{c}} = \left(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\mathbf{R}\right)^{-1}\boldsymbol{\Phi}^T\mathbf{y}$$

This is a linear smoother:

$$\hat{\mathbf{x}} = \boldsymbol{\Phi}\hat{\mathbf{c}} = \boldsymbol{\Phi}\left(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\mathbf{R}\right)^{-1}\boldsymbol{\Phi}^T\mathbf{y}$$

Choosing Smoothing Parameter

The fifth datum has been omitted from fitting and the continuous line shows a penalized regression spline fitted to the remaining data.



Choosing Smoothing Parameter

- When λ is too high, the spline fits the data poorly and does no better with the missing point.
- For the intermediate λ the spline is fitting the underlying signal quite well, but smoothing through the noise: as a result the missing datum is reasonably well predicted.
- When λ is too low, the spline fits the noise as well as the signal and the extra variability that this induces causes it to predict the missing datum rather poorly.

Generalized Cross Validation

GCV minimizes the prediction error of the fit.

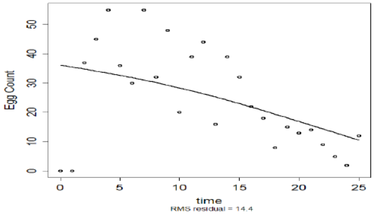
$$\mathrm{GCV}(\lambda) = \frac{(\mathbf{y} - \mathbf{\Phi}\hat{\mathbf{c}})^T(\mathbf{y} - \mathbf{\Phi}\hat{\mathbf{c}})}{[n - \mathrm{df}]^2}$$

where

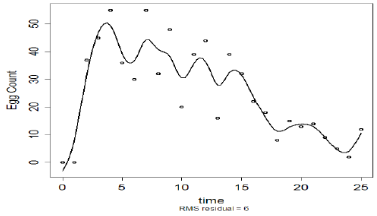
- $\hat{\mathbf{c}} = \left(\mathbf{\Phi}^T\mathbf{\Phi} + \lambda\mathbf{R}\right)^{-1}\mathbf{\Phi}^T\mathbf{y}$
- $\mathbf{S} = \mathbf{\Phi}\left(\mathbf{\Phi}^T\mathbf{\Phi} + \lambda\mathbf{R}\right)^{-1}\mathbf{\Phi}^T$
- $\mathrm{df} = \mathrm{trace}(\mathbf{S})$

Generalized Cross Validation

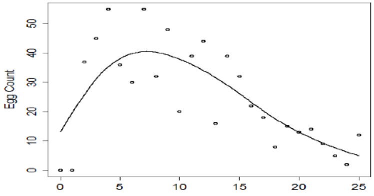
Smooth



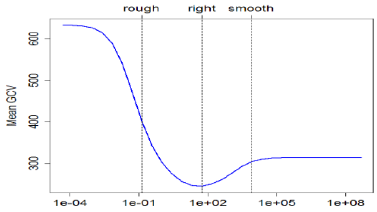
Rough



Right



GCV



Smoothing in R/Matlab

The fda package provides utilities based on basis expansions and smoothing penalties.

fda works by defining objects that can be manipulated with pre-defined functions.

In particular

- **basis** objects define basis systems that can be used
- **fd** objects store functional data objects
- **fdPar** objects collect fd, plus a smoothing parameter

Functional Data (fd) Objects

$$x(t_j) = \sum_{k=1}^{K} \phi_k(t_j) c_k$$

Stores functional data: a list with elements

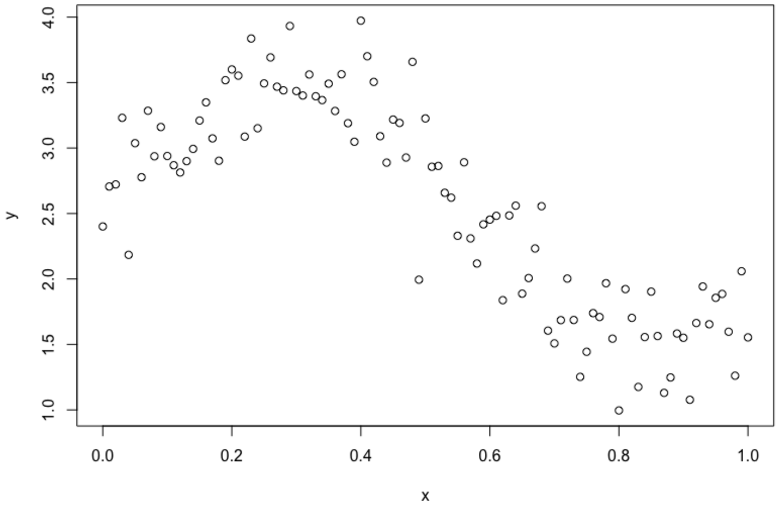
- **coefs** array of coeffcients c_k for $k = 1, \ldots, K$
- **basis** basis object containing the basis functions $\phi_k(t_j)$

```
fdobj = fd(coefs,basis)
```

Example

```
# S1: Generate x on a linear grid and simulate y

m = 101
x = seq(0, 1, length = m)
set.seed(555)
y = sin(5.5 * x) + 2.5 + rnorm(m) * 0.3

# Scatterplot to check

plot(x, y)
```

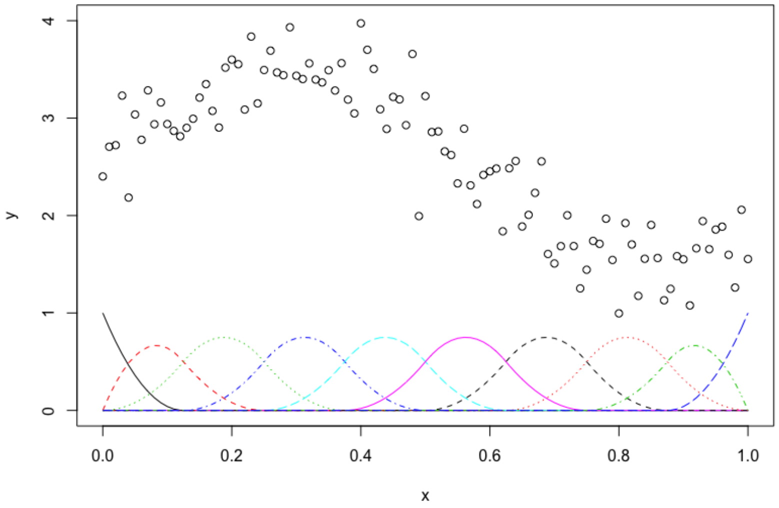
Example



Example

```
nbasis = 10

norder = 3

rng    = c(0,1)

knots  = seq(rng(1), rng(2), length=nbasis-norder+2)

basis = create.bspline.basis(rng,nbasis,norder,knots)
```

Example



fdPar Objects

fdPar is a utility for imposing smoothing. It collects

- **fdobj** an fd (or a basis) object.
- **Lfdobj** a Lfd object
- **lambda** a smoothing parameter

```
fdParobj = fdPar(basis,2,0.01)
```

Smoothing Functions

Main smoothing function is **smooth.basis**

- **x** x-axis values
- **y** y-axis values
- **fdParobj** an fdPar object

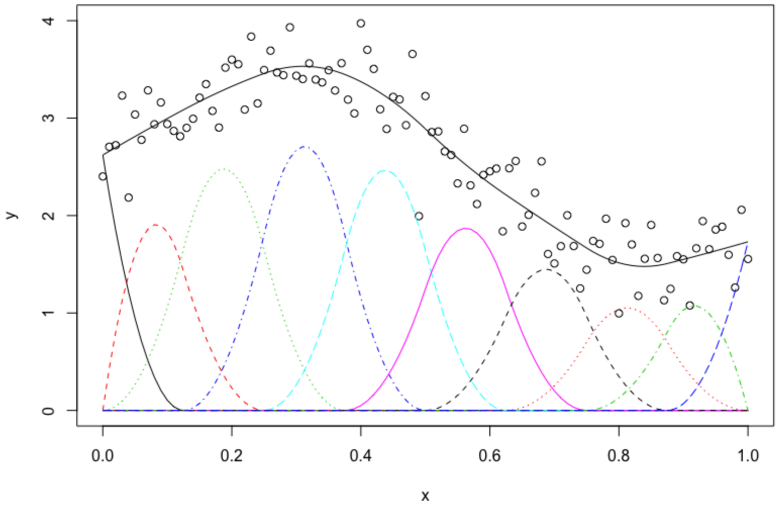
```
fit = smooth.basis(x,y,fdParobj)
```

Smoothing Functions

The smoothing function **smooth.basis** returns an fd object along with

- **df** equivalent degrees of freedom
- **SSE** total sum of squared errors
- **gcv** GCV for each smooth

Example



More General Smoothing Penalties

- The scientist often has prior knowledge informing them that the function has a power-law, exponential or sinusoidal behaviour and they have theoretical grounds for requiring that the functional estimation problem incorporates this information.
- Penalizing departure from a linear function is analogous to requiring that
  $$\frac{\mathrm{d}^2\hat{x}}{\mathrm{d}t^2} = 0$$
  The penalty we have been using to control the curvature of $\hat{\mathbf{x}}$
- More generally we can penalize departure from a specified class of parametric models for the data by requiring that $\hat{\mathbf{x}}$ satisfies a differential equation.

Differential Equations

An ordinary differential equation (ODE):

An ODE is a mathematical equation that relates some univariate function with its derivatives.

A first-order ODE is a relationship of the form

$$F(t, x, \frac{\mathrm{d}x}{\mathrm{d}t}) = 0$$

where $F(\cdot, \cdot, \cdot)$ is some function of three variables.

A second-order ODE is a relationship of the form

$$F(t, x, \frac{\mathrm{d}x}{\mathrm{d}t}, \frac{\mathrm{d}^2x}{\mathrm{d}t^2}) = 0$$

where $F(\cdot, \cdot, \cdot, \cdot)$ is some function of four variables.

First-order ODE:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \beta x,$$

has the solution

$$x(t) = C \exp^{\beta t},$$

where C is any constant. This equation has infinitely many solutions depending on the coefficient C.

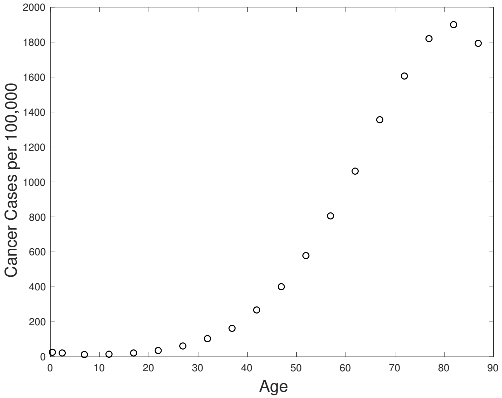
This solution exhibits:

- exponential decay when $\beta < 0$
- exponential growth when $\beta > 0$

Exponential functions play key roles in modelling many natural phenomena.

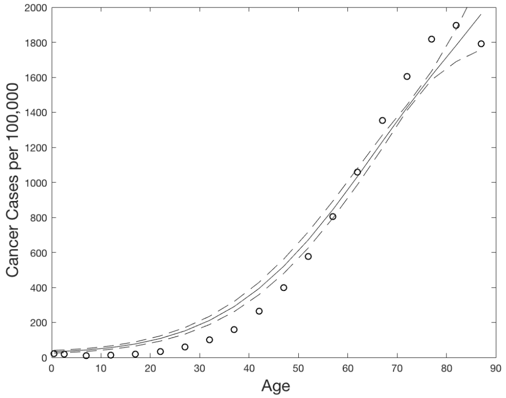
Incidence of cancer vrs age for American females in 2001

The chances of getting cancer increase as you age. In fact, by looking at data compiled by the National Cancer Institute you can readily see that incidence of cancer increases dramatically between the ages of 35 and 80.



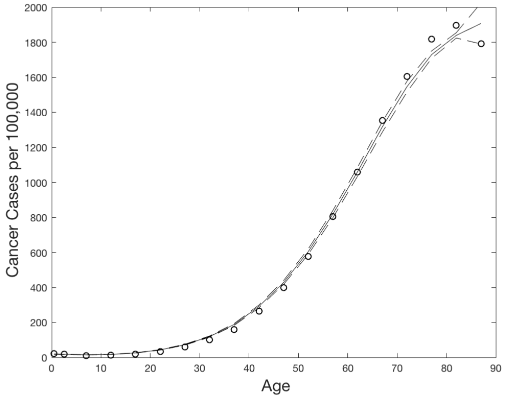
Incidence of cancer vrs age for American females in 2001

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = 0.02x$$



Incidence of cancer vrs age for American females in 2001

$$\mathrm{PENSSE}(\mathbf{c}) = \sum_{j=1}^{n} (y_j - x(t_j))^2 + \lambda \int \left[\frac{\mathrm{d}x(t)}{\mathrm{d}t} - 0.02x(t)\right]^2 \mathrm{d}t$$



First-order ordinary-differential equation:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \beta(t)x(t)$$

which has the solution

$$x(t) = C\exp(\int -\beta(t)\,\mathrm{d}t)$$

where C is any constant.

This equation has infinitely many solutions depending on the coefficient C.

Incidence of cancer vrs age for American females in 2001

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \beta(t)x(t)$$

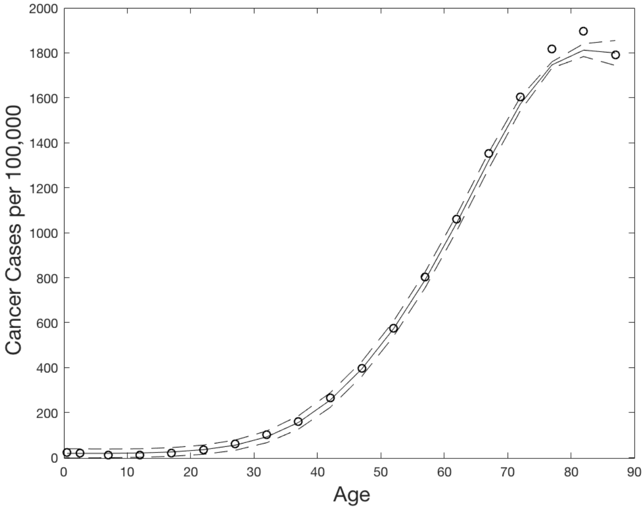
We represent $\beta(t)$ by a linear expansion

$$\beta(t) \approx \sum_l b_l \psi_l(t),$$

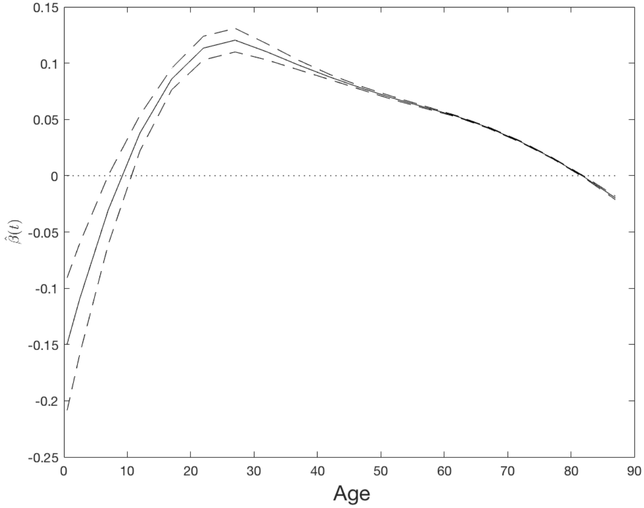
where

- $\psi_l(t)$ are spline basis functions
- b_l are the corresponding coefficients.

Incidence of cancer vrs age for American females in 2001



Incidence of cancer vrs age for American females in 2001



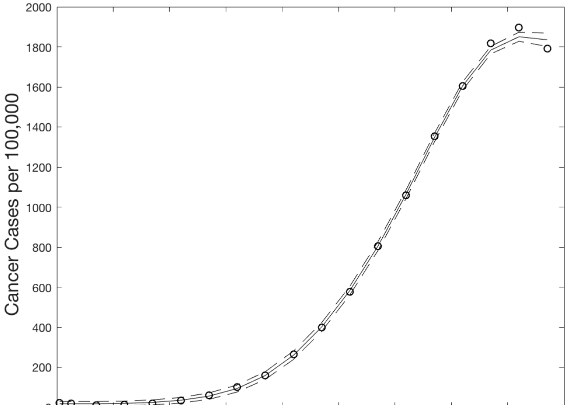
Incidence of cancer vrs age for American females in 2001

As females age from

- 0 to 10 the decay rate of the incidences of cancer decreases.
- 10 to 27 the growth rate of the incidences of cancer increases.
- 27 to 82 the growth rate of the incidences of cancer decreases.
- 82 to 87 the decay rate of the incidences of cancer increases.

Incidence of cancer vrs age for American females in 2001

$$\mathrm{PENSSE}(\mathbf{c}) = \sum_{j=1}^{n} (y_j - x(t_j))^2 + \lambda \int \left[\frac{\mathrm{d}x(t)}{\mathrm{d}t} - \hat{\beta}(t)x(t)\right]^2 \mathrm{d}t$$



Forced first-order ordinary-differential equation

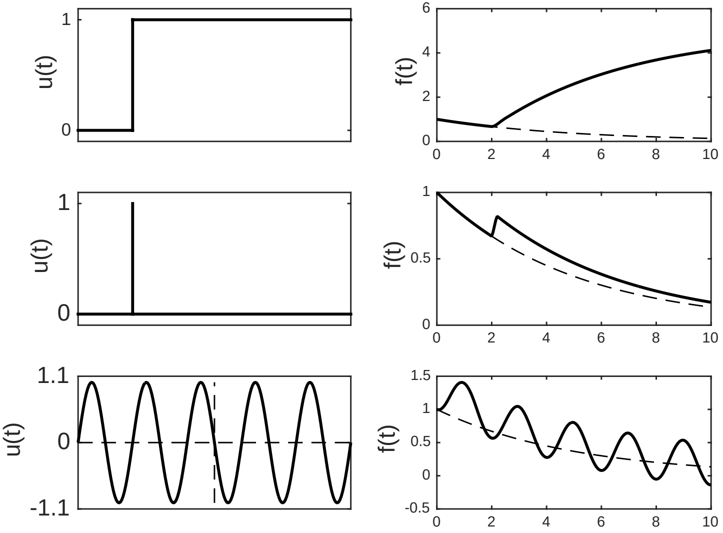
$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = -\beta x(t) + \alpha u(t)$$

where α is a nonzero constant modulating the impact of an external explanatory variable $u(t)$ on $\frac{\mathrm{d}x(t)}{\mathrm{d}t}$.

$$x(t) = C\exp(-\beta t) + \alpha \int_{s_0}^{s_{max}} \exp(-\beta(t-s))u(s)\mathrm{d}s$$

The output x is a buffered version of u in the sense that u's sharp variation will show up in x's variation as being spread out over time.

Forced first-order ordinary-differential equation



Temperature in Montreal

Much of the solar radiation reaching the earth is absorbed by the land and sea and then gradually released back into the atmosphere.

Hence, atmospheric heat transfer is a buffered process which, to a first order, can be modelled by,

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \beta(t)x(t),$$

where $\beta(t)$ is a negative time-varying parameter.

Temperature in Montreal

We model the 5-day averages of the temperature in Montreal as a first order differential equation with two explanatory variables:

- α_0 represents the annual mean temperature
- $\alpha_2 \cos(2\pi(t + 192)/365)$ which is a crude approximation of the seasonal variation in the ground-level heat absorption.

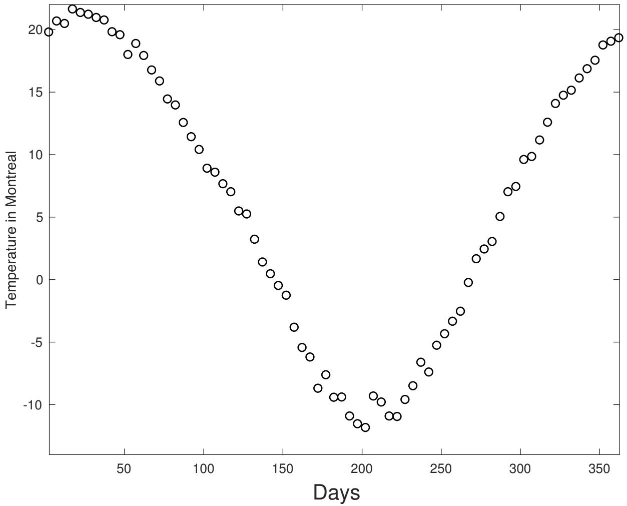
Temperature in Montreal

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \beta(t)x + \alpha_1 + \alpha_2 \cos\left(\frac{2\pi(t+192)}{365}\right)$$

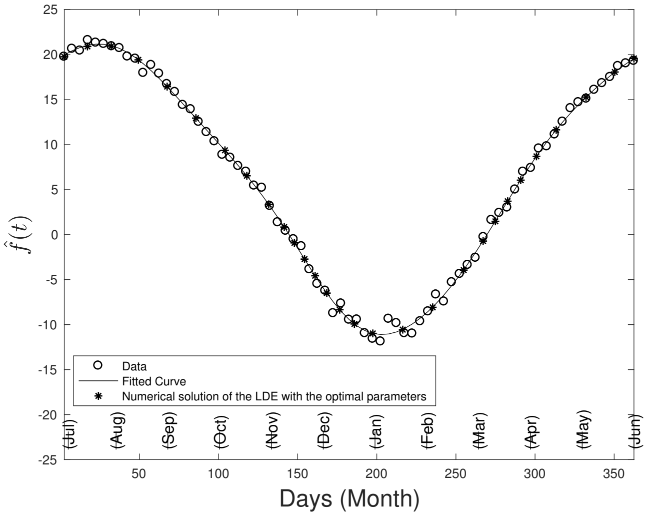
The translation $(t + 192)$ places the minimum of the cosine at winter solstice and positions winter in the center of our displays.

To capture the natural periodicity in the time-varying parameter $\beta(t)$ we represent it by a linear expansion $\beta(t) \approx \sum_l b_l \psi_l(t)$, where $\psi_l(t)$ are Fourier basis functions and b_l are the corresponding coefficients.

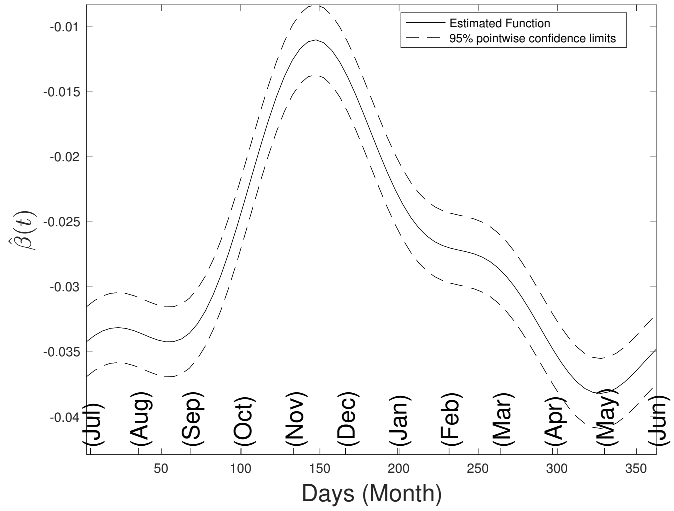
Temperature in Montreal



Temperature in Montreal



Temperature in Montreal



Temperature in Montreal

The estimated parameters of the F with 95% confidence intervals are $\hat{\alpha_1} = 0.23 \pm 0.01$ and $\hat{\alpha_2} = 0.56 \pm 0.01$.

This indicates that the annual mean temperature in Montreal is just above zero deg. C and that the solar radiation has a significant impact on the temperature.

As expected $\beta(t|\hat{\boldsymbol{\theta}})$ is negative. The value $\tau(t|\hat{\boldsymbol{\theta}}) = \frac{1}{\beta(t)}$ is the length of time for a buffered change in temperature to attain about $\frac{2}{3}$ of its value and we see that this corresponds to about 30 days in the spring/summer months and 100 days in the autumn/winter.

Second-order ordinary-differential equation:

The second law states that the acceleration of an object is dependent upon two variables - the net force acting upon the object and the mass of the object.

$$\begin{aligned}\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} &= \text{damping force} + \text{restoring force} \\ &= \beta_1 \frac{\mathrm{d}x(t)}{\mathrm{d}t} + \beta_0 x(t)\end{aligned}$$

Second-order ordinary-differential equation:

$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} = -\beta_1 \frac{\mathrm{d}x(t)}{\mathrm{d}t} - \beta_0 x(t)$$

which has the solution

$$x(t) = C_1 \exp(\lambda_1 t) + C_2 \exp(\lambda_2 t)$$

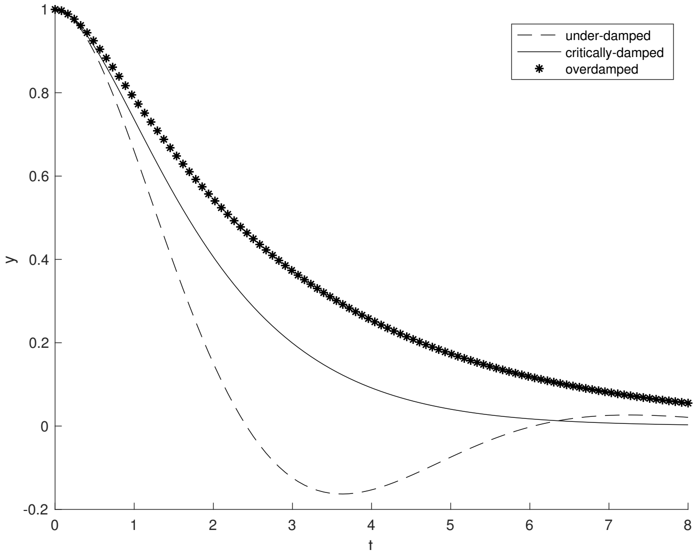
where C_1 and C_2 are arbitrary constants.

- $\lambda_1 = \frac{-\beta_1 + \sqrt{\beta_1^2 - 4\beta_0}}{2}$
- $\lambda_2 = \frac{-\beta_1 - \sqrt{\beta_1^2 - 4\beta_0}}{2}$

Second-order ordinary-differential equation:

- Overdamping $\beta_1^2 - 4\beta_0 > 0$: oscillations do not occur as there is a strong damping force.
- Critical damping $\beta_1^2 - 4\beta_0 = 0$: the damping is just sufficient to suppress the oscillations.
- Underdamping $\beta_1^2 - 4\beta_0 < 0$: there are oscillations that are damped by the factor $\exp(-\frac{\beta_1}{2\beta_0}t)$ that is the oscillations decay to 0 as t increases.

Second-order ordinary-differential equation:



Instantaneous forward curve

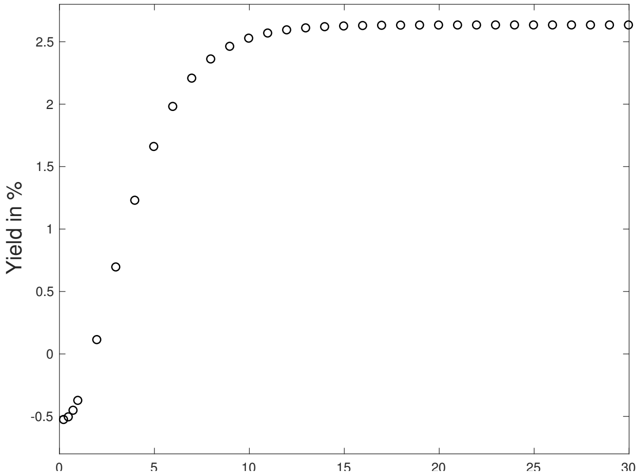
A yield curve (which can also be known as the term structure of interest rates) represents the relationship between market remuneration (interest) rates and the remaining time to maturity of debt securities.

Typical yield curve shapes are generated by the solution to a second order differential equation with constant coefficients.

- β_0 represents the level of the yield curve and indicates long-run inflation expectations.
- β_1 represents the slope of the yield curve and provides information on the current state of monetary policy (growth/recession)

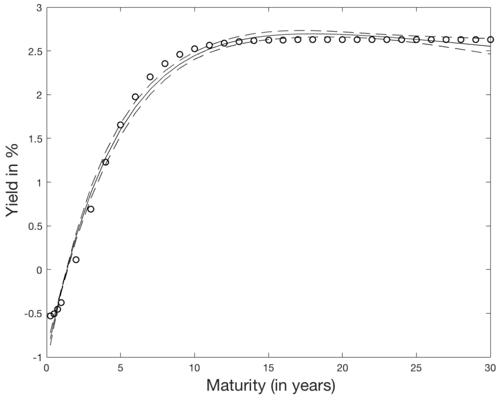
Instantaneous forward curve

The euro area yield curve shows AAA-rated euro area central government bonds



Instantaneous forward curve

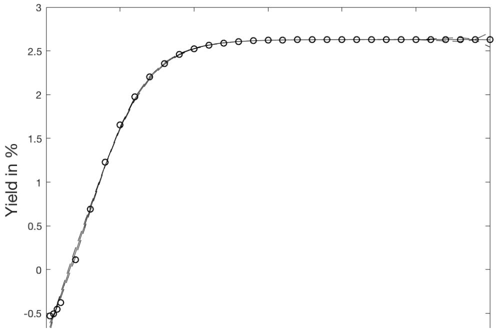
$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} = 0.215\frac{\mathrm{d}x(t)}{\mathrm{d}t} + 0.001x(t)$$



Instantaneous forward curve

$$\mathrm{PENSSE}(\mathbf{c}) = \sum_{j=1}^{n} (y_j - x(t_j))^2 + \dots$$

$$\lambda \int_{t_0}^{t_{max}} \left[\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} - 0.215 \frac{\mathrm{d}x(t)}{\mathrm{d}t} - 0.001 x(t) \right]^2 \mathrm{d}t$$



Second-order ordinary-differential equation:

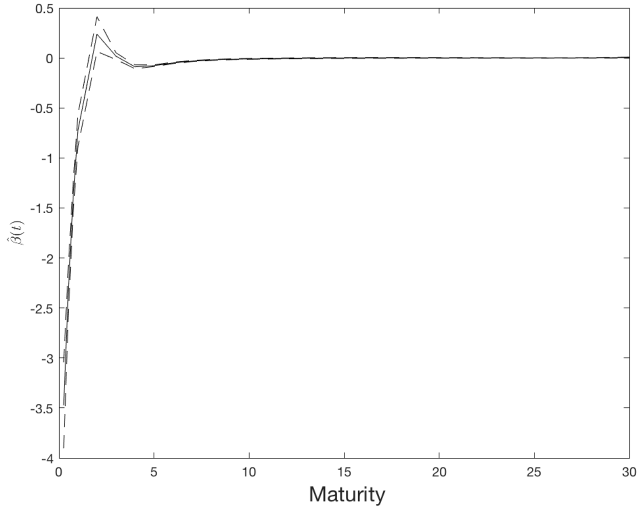
$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} = \beta(t)x(t)$$

The time-varying parameter $\beta(t)$ is represented by a linear expansion

$$\beta(t) \approx \sum_{l} b_l \psi_l(t),$$

where $\psi_l(t)$ are spline basis functions and b_l are the corresponding coefficients.

Instantaneous forward curve

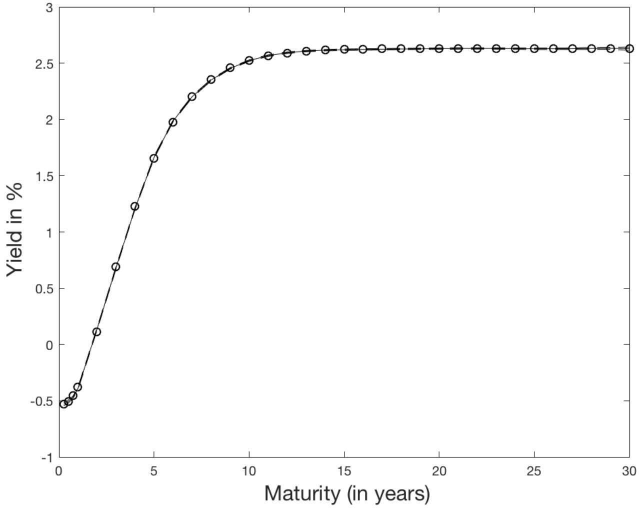


Instantaneous forward curve

The restoring force

- is negative for the 3 mth to the 1yr yield.
- is positive for the 2 and 3 year yield.

Instantaneous forward curve



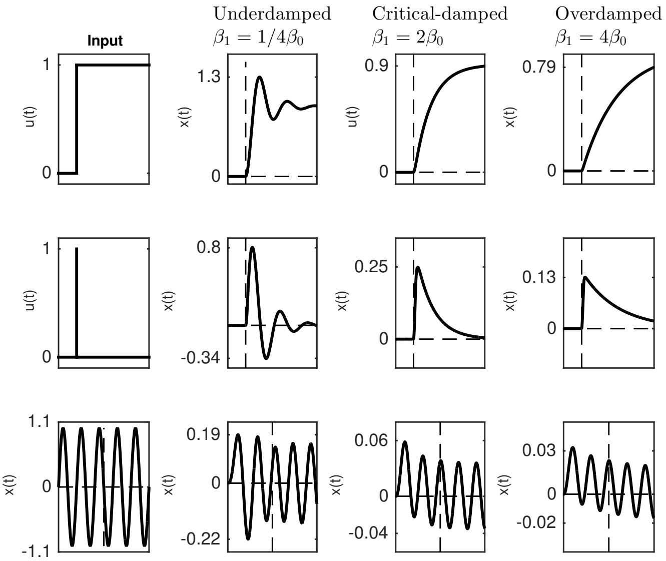
Forced second-order ordinary-differential equation:

Suppose that, in addition to the restoring force and the damping force, the trajectory is affected by an external force $u(t)$.

Then Newton's Second Law states

$$\begin{aligned}\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} &= \text{damping force} + \text{restoring force} + \text{external force} \\ &= \beta_1 \frac{\mathrm{d}x(t)}{\mathrm{d}t} + \beta_0 x(t) + \alpha_0 u(t)\end{aligned}$$

Forced second-order ordinary-differential equation:

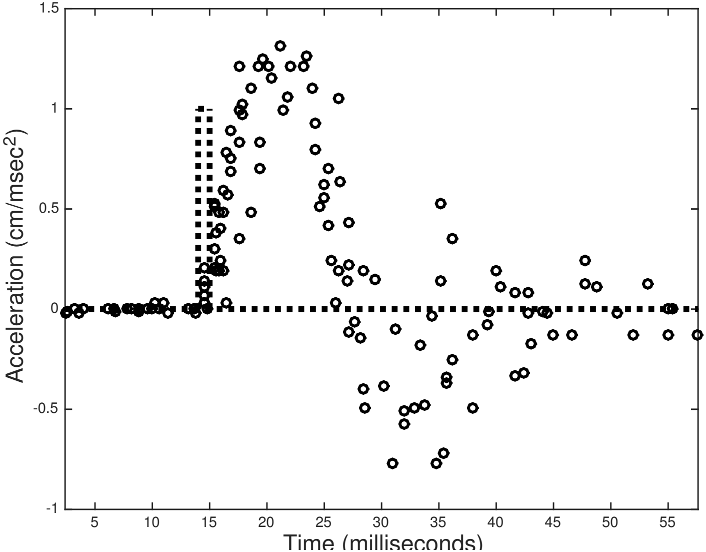


TBI example

Traumatic brain injury (TBI), contributes to just under a third (30.5%) of all injury-related deaths in the United States.

Schmidt et al. (1981) at the University of Heidelberg placed an accelerometer inside the skull of a cadaver in order to record the acceleration of the brain tissue before and after a series of five blows to the cranium

TBI example



1.5
1
0.5
0
-0.5
-1
Acceleration (cm/msec^2)
5
10
15
20
25
30
35
40
45
50
55
Time (milliseconds)

TBI example

Mechanical principles imply that the acceleration $x(t)$ can be modelled by a second order differential equation with a point impulse $u(t)$ representing the blow to the cranium.

The three parameters β_0, β_1 and α_0 in the second order buffer equation

$$\frac{\mathrm{d}^2x(t)}{\mathrm{d}t^2} = \beta_0 x(t) + \beta_1 \frac{\mathrm{d}x(t)}{\mathrm{d}t} + \alpha_0 u(t),$$

convey

- the period of the oscillation,
- the change in its amplitude (as $t \to \infty$ the oscillations decay exponentially to zero)
- the size of the impact from the point impulse, respectively.

TBI example

Setting up the knot structure for this problem requires some care because the box function is discontinuous, and the additive nature of the forcing implies a discontinuity in

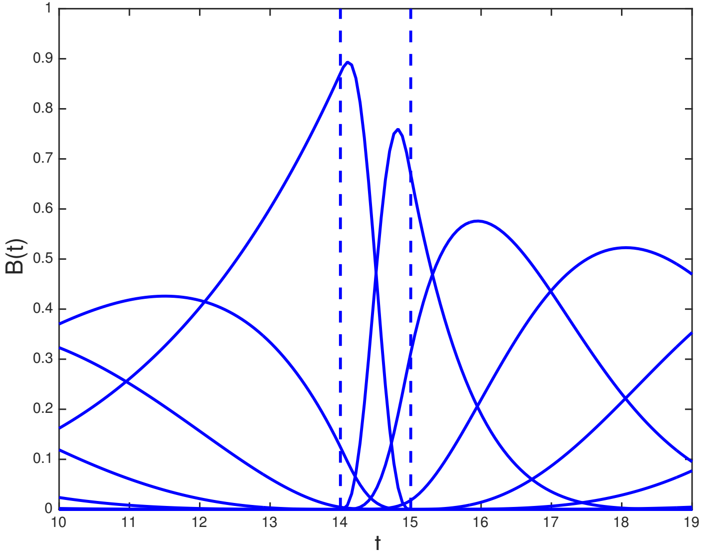
$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2}$$

.

We used order six B-spline functions, which by their nature have discontinuous fourth derivatives if all knots are singletons.

To achieve curvature discontinuity at the impact point and at that point plus one, we placed three knots at these locations.

TBI example



TBI example

$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} = -0.05x(t) - 0.15\frac{\mathrm{d}x(t)}{\mathrm{d}t} + 0.39u(t),$$

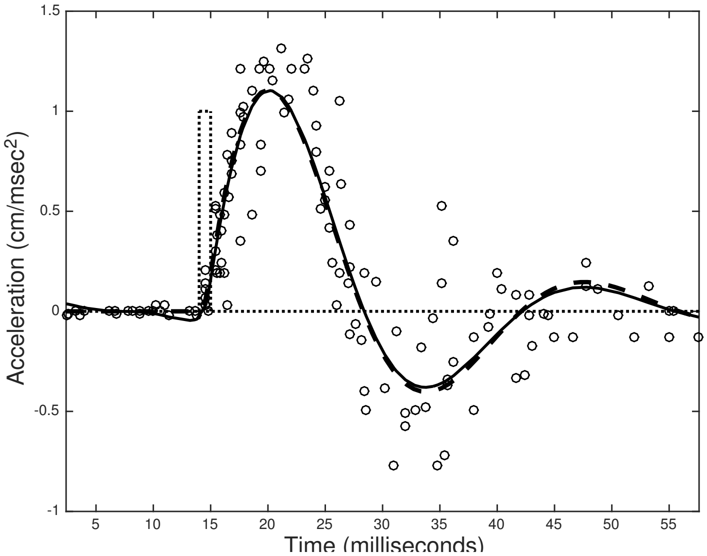
where $u(t)$ is one for $14 \leq t \leq 15$ and zero otherwise.

The final parameter estimates with 95% confidence intervals are:

- $\hat{\beta}_0 = -0.056 \pm 0.002$,
- $\hat{\beta}_1 = -0.150 \pm 0.018$
- $\hat{\alpha}_0 = 0.395 \pm 0.032$.

This is an under-damped process; after the blow to the cranium the acceleration will oscillate with a decreasing amplitude that will quickly decay to zero.

TBI example



1.5
1
0.5
0
-0.5
-1
Acceleration (cm/msec^2)
5
10
15
20
25
30
35
40
45
50
55
Time (milliseconds)

Dynamical Systems

A system of linear differential equations is a set of linear equations relating a group of functions to their derivatives.

$$\begin{array}{rcl}
\frac{\mathrm{d}x_1(t)}{\mathrm{d}t} & = & \beta_{1,0} + \beta_{1,1}x_1(t) + \ldots + \beta_{1,n}x_n(t) \\
\frac{\mathrm{d}x_2(t)}{\mathrm{d}t} & = & \beta_{2,0} + \beta_{2,1}x_1(t) + \ldots + \beta_{2,n}x_n(t) \\
\vdots & = & \vdots \ + \ \vdots \quad + \ldots + \ \vdots \\
\frac{\mathrm{d}x_n(t)}{\mathrm{d}t} & = & \beta_{n,0} + \beta_{n,1}x_1(t) + \ldots + \beta_{n,n}x_n(t)
\end{array}$$

Dynamical Systems

$$\begin{aligned}
\frac{\mathrm{d}x_1(t)}{\mathrm{d}t} &= \beta_{1,1}x_1(t) + \beta_{1,2}x_2(t) + \beta_{1,3}x_3(t) \\
\frac{\mathrm{d}x_2(t)}{\mathrm{d}t} &= \beta_{2,1}x_1(t) + \beta_{2,2}x_2(t) + \beta_{2,3}x_3(t) \\
\frac{\mathrm{d}x_3(t)}{\mathrm{d}t} &= \beta_{3,1}x_1(t) + \beta_{3,2}x_2(t) + \beta_{3,3}x_3(t)
\end{aligned}$$

Dynamical Systems

$$\begin{tikzcd}
 & x_1 \arrow[loop above, "\beta_{1,1}"] \arrow[dl, bend right, "\beta_{1,2}"'] \arrow[dr, bend left, "\beta_{1,3}"] & \\
x_2 \arrow[ur, "\beta_{2,1}"'] \arrow[loop left, "\beta_{2,2}"] \arrow[rr, bend right, "\beta_{2,3}"'] & & x_3 \arrow[ul, "\beta_{1,3}"] \arrow[ll, "\beta_{3,2}"'] \arrow[loop right, "\beta_{3,3}"]
\end{tikzcd}$$

Measles

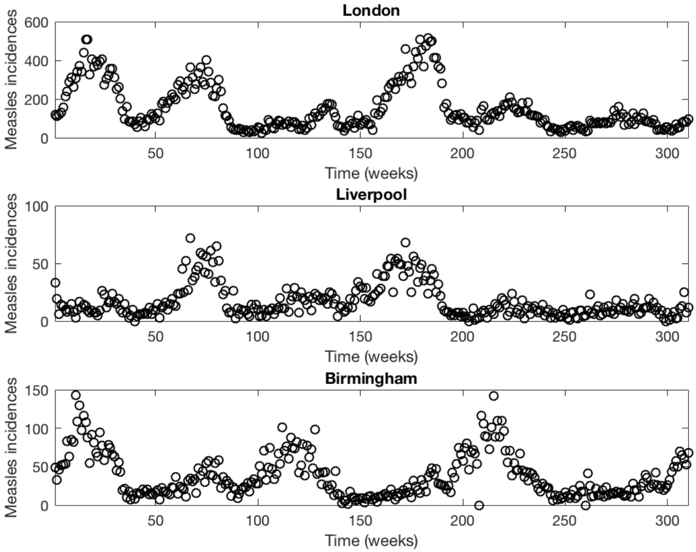
We will exam weekly measles incidences for London, Liverpool and Birmingham from 1982 to 1987.

Measles is a highly contagious infectious disease caused by the measles virus.

No other vaccine-preventable disease causes as many deaths.

In 1980, 2.6 million people died of it but by 2014, global vaccination programs had reduced the number of deaths from measles to 73,000.

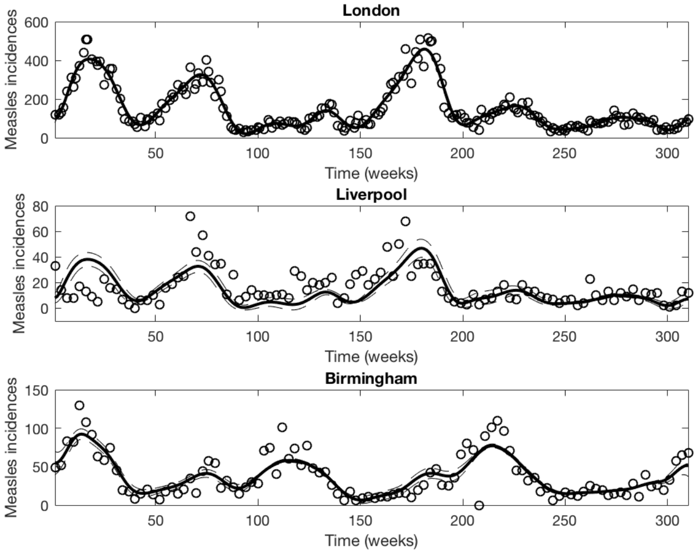
Measles


London
Liverpool
Birmingham
Measles incidences
Time (weeks)
0
200
400
600
50
100
150
0
50
100
50
100
150
200
250
300

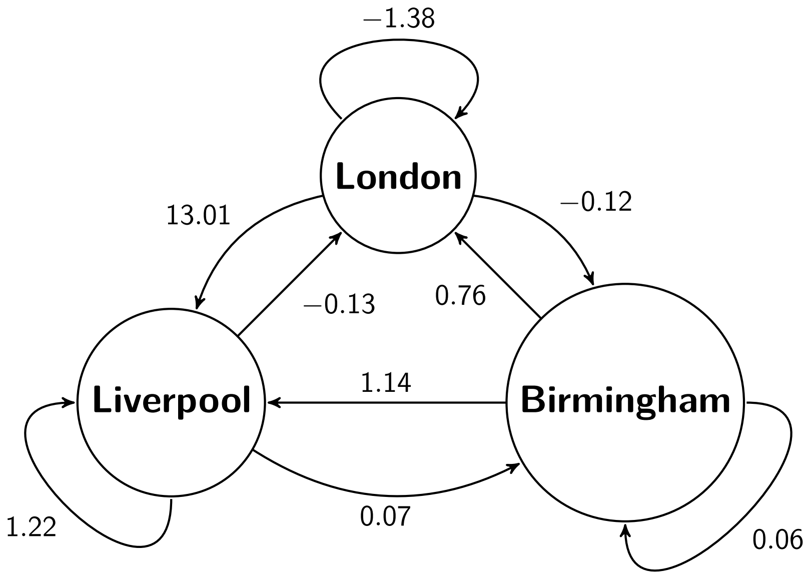
Measles

$$\begin{aligned}
\frac{\mathrm{d}London(t)}{\mathrm{d}t} &= \beta_{1,1} London(t) + \beta_{1,2} Liverpool(t) \\
&\quad + \beta_{1,3} Birmingham(t) \\
\frac{\mathrm{d}Liverpool(t)}{\mathrm{d}t} &= \beta_{2,1} London(t) + \beta_{2,2} Liverpool(t) \\
&\quad + \beta_{2,3} Birmingham(t) \\
\frac{\mathrm{d}Birmingham(t)}{\mathrm{d}t} &= \beta_{3,1} London(t) + \beta_{3,2} Liverpool(t) \\
&\quad + \beta_{3,3} Birmingham(t)
\end{aligned}$$

Measles


London
Liverpool
Birmingham
Measles incidences
Time (weeks)
600
400
200
0
80
60
40
20
150
100
50
250
300

Measles

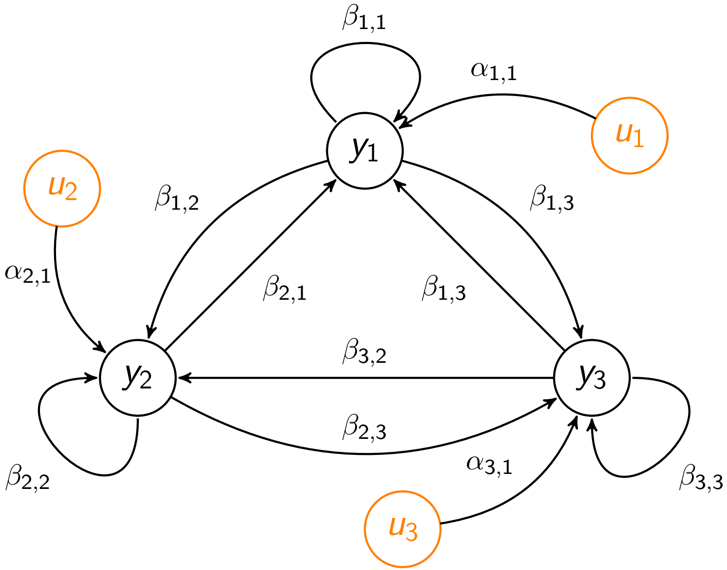


−1.38
London
13.01
−0.12
−0.13
0.76
Liverpool
1.14
Birmingham
1.22
0.07
0.06

Forced Dynamical Systems

$$\begin{aligned}
\frac{\mathrm{d}x(t)}{\mathrm{d}t} &= \beta_{1,1}x_1(t) + \beta_{1,2}x_2(t) + \beta_{1,3}x_3(t) + \alpha_{1,1}u_1(t) \\
\frac{\mathrm{d}x_2(t)}{\mathrm{d}t} &= \beta_{2,1}x_1(t) + \beta_{2,2}x_2(t) + \beta_{2,3}x_3(t) + \alpha_{2,1}u_2(t) \\
\frac{\mathrm{d}x_3(t)}{\mathrm{d}t} &= \beta_{3,1}x_1(t) + \beta_{3,2}x_2(t) + \beta_{3,3}x_3(t) + \alpha_{3,1}u_3(t)
\end{aligned}$$

Forced Dynamical Systems

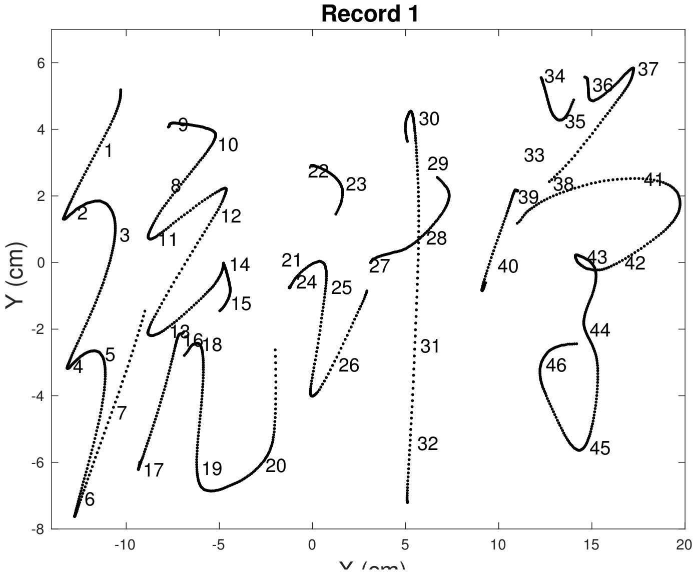


Handwriting

The hand is a complex bio-mechanical system that exhibits various kinds of harmonic or oscillatory behaviour, subject to forces of elastic and viscous components of the muscles, and must work in the context of gravitational and inertial forces.

The motor control centre in the cortex synchronizes the many muscles involved by transmitting neural activations in bundles at approximately equi-spaced time intervals.

Handwriting



Handwriting

The muscles and the bones in the arm, wrist and fingers that the neural activations control behave very much like lightly damped springs, and the activation changes their spring constants so as to determine the desired spatial extent and curvature of the pen movements over each of these relatively constant periods.

Handwriting

Thus it would be reasonable to assume that the X(t), Y(t) and Z(t) movement could be modeled with a second order harmonic dynamical system

$$\begin{aligned} D^2X(y) &= \beta_{0,X}X(t) + \alpha_X u_1(t) \\ D^2Y(t) &= \beta_{0,Y}Y(t) + \alpha_Y u_2(t) \\ D^2Z(t) &= \beta_{0,Z}Z(t) + \alpha_Z u_3(t), \end{aligned}$$

Handwriting

where

- $\beta_{0,X}$, $\beta_{0,Y}$ and $\beta_{0,Z}$ convey the natural period of oscillation of the X, Y and Z positions respectively
- the unknown forcing functions $\alpha_X u_1(t), \alpha_Y u_2(t)$ and $\alpha_Z u_3(t)$ represent the neural activations.

Handwriting

The data are well represented by 197 order six B-splines with equally-spaced knots.

Since spline functions are best estimated with inter-knot intervals of the order of one, we used as our time unit a hundredth of a second, or a centi-second.

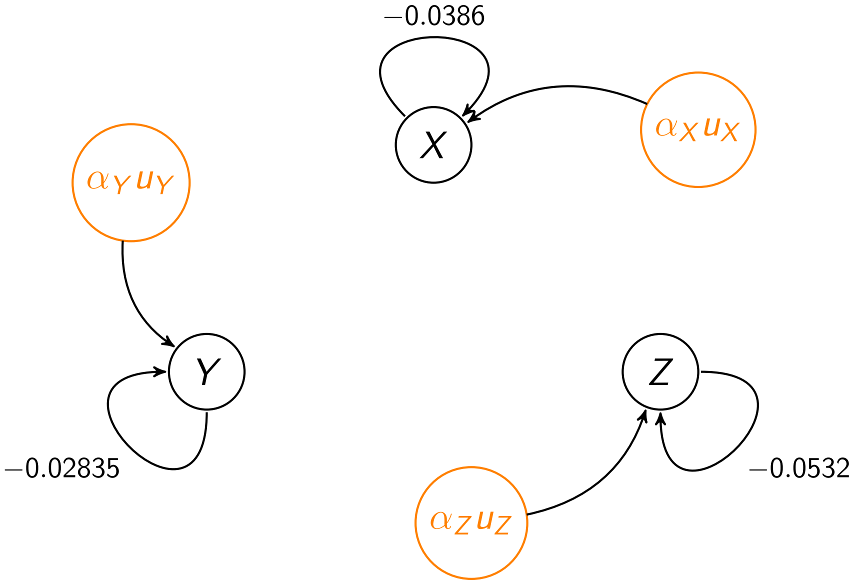
We represent the forcing functions by order 1 splines, corresponding to step functions, with knots at 48 equally spaced locations for each coordinate.

Handwriting

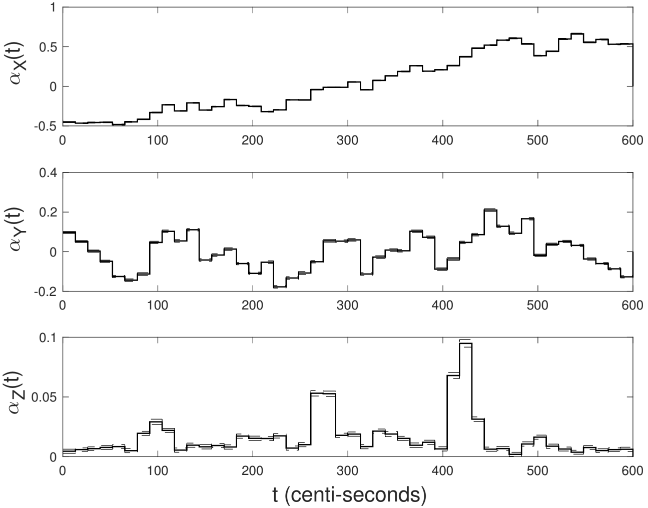
The spring constant estimates are:

- $\beta_{0X} = -0.0386 \pm 0.0002$,
- $\beta_{0Y} = -0.02835 \pm 0.0002$
- $\beta_{0Z} = -0.0532 \pm 0.0007$.

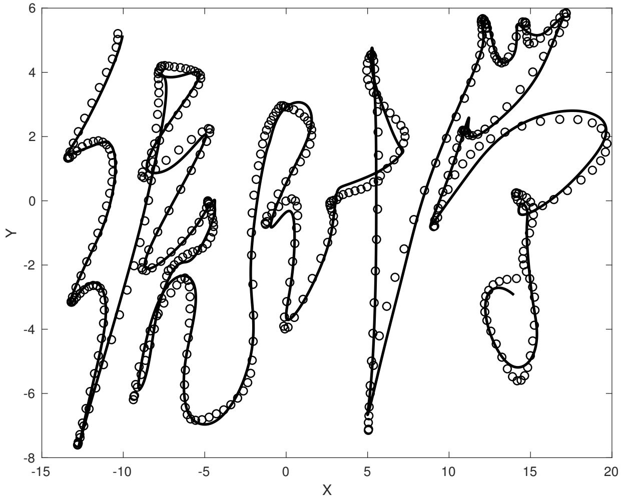
Dynamical Systems



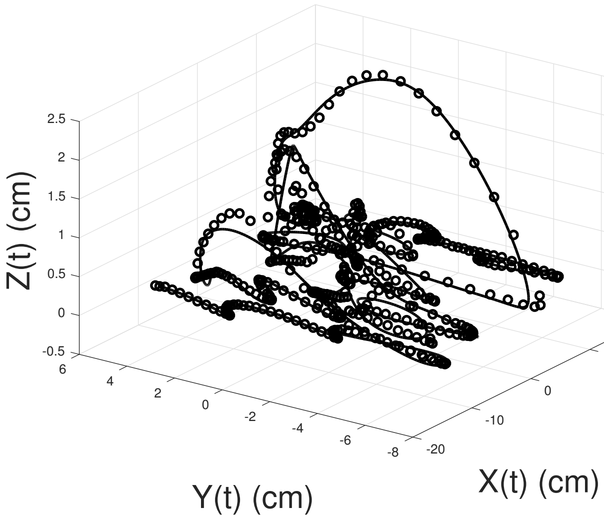
Handwriting



Handwriting



Handwriting



Parameter Estimation

Data2LD is an approach for estimating parameters defining a linear differential equations from:

- partially observed data
- possibly, observed over only a subset of the variables,
- distributed over the domain of the linear differential equation.

Two types of parameters

1. **Nuisance parameters: c** are required to fit the data

$$\hat{x}(t) = \sum_{k=1}^{K} c_k \phi_k(t) = \mathbf{\Phi c}$$

   - these are large in number, their number tends to depend on how much data is available.
   - they often define localized effects on the fit, and their values are seldom of direct interest.

Two types of parameters

The formulation of a single differential equation of order p is

$$\frac{\mathrm{d}^p x(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_r(t|\boldsymbol{\theta}) \frac{\mathrm{d}^r x(t)}{\mathrm{d}t^r} - \sum_{q=1}^{Q} \alpha_q(t|\boldsymbol{\theta}) u_q(t) = 0.$$

The coefficient functions:

- β_r and α_q may be non-constant functions over t, that depend on the parameters $\boldsymbol{\theta}$, but must not depend on the values of x.
- the variables $\alpha_q(t)u_q(t)$, $q = 1, \ldots, Q$ are the *forcing functions*

Two types of parameters

The coefficient functions are typically estimated by linear basis function expansions

$$\beta_r(t) \approx \sum_{j=1}^{J_r} b_{jr}\psi_{jr}(t) \quad \text{and} \quad \alpha_q(t) \approx \sum_{\ell=1}^{L_q} a_{\ell q}\xi_{\ell q}(t),$$

where

- $\psi_{jr}(t)$ and $\xi_{\ell q}(t)$ are known basis function
- the parameters of the differential equation are collectively referred to as the vector

$$\boldsymbol{\theta} = [b_{10}, \ldots, b_{J_{(p-1)}(p-1)}, a_{11}, \ldots, a_{L_Q Q}].$$

Two types of parameters

2. **Structural parameters:** $\boldsymbol{\theta}$ are the parameters of the differential equation
   - these are small in number,
   - they have interpretive importance.

Two types of parameters

The parameter cascade:

- Defines nuisance parameters as functions $\mathbf{c}(\boldsymbol{\theta})$ of structural $\boldsymbol{\theta}$.

These functional relationships are defined by choosing two different optimization criteria:

1. An inner optimization for the nuisance parameters
2. An outer optimization for the structural parameters

Inner optimization: nuisance parameters

Let

$$L(x(t)|\boldsymbol{\theta}, u_q(x)) = \frac{\mathrm{d}^p x(t)}{\mathrm{d}t^p} = \sum_{r=0}^{p-1} \beta_r(t|\boldsymbol{\theta}) \frac{\mathrm{d}^r x(t)}{\mathrm{d}t^r} + \sum_{q=1}^{Q} \alpha_q(t|\boldsymbol{\theta}) u_q(t).$$

where

- β_r and α_q may be non-constant functions over t dependant on the parameters $\boldsymbol{\theta}$
- the variables $\alpha_q(t)u_q(t)$, $q = 1, \ldots, Q$ are the *forcing functions*

Inner optimization: nuisance parameters

The lowest level or inner criterion is defined as

$$\begin{aligned}\mathrm{PENSSE}(\mathbf{c}|\boldsymbol{\theta},\rho) &= \frac{(1-\rho)}{n}(\mathbf{y}-\mathbf{\Phi c})^T(\mathbf{y}-\mathbf{\Phi c}) + \\ & \frac{\rho}{t_{max}-t_0}\int_{t_0}^{t_{max}} \left[L(\mathbf{\Phi c}|\boldsymbol{\theta},u_q(x))\right]^2 \,\mathrm{d}t\end{aligned}$$

where

- $\mathbf{c}$ are the nuisance parameters approximating x.
- $L(\mathbf{\Phi c}|\boldsymbol{\theta},u_q(x))$ is the differential equation evaluated at $\mathbf{x}=\mathbf{\Phi c}$.
- $\boldsymbol{\theta}$ are the structural parameters
- ρ is the complexity parameter defining the trade-off between fidelity to the data and adherence to the differential equation.

Inner optimization: nuisance parameters

The complexity parameter ρ is in the half-closed interval $[0, 1)$.

If $\rho = 0$ then the corresponding estimated function evaluated at the observations $\mathbf{t}$, $\hat{\mathbf{x}}$, is the least squares approximation of the data and hence does not depend on the differential equation.

As $\rho \to 1$, $\hat{\mathbf{x}}$ tends to the particular solution of the differential equation with the parameters $\boldsymbol{\theta}$ that best approximates the data.

Inner optimization: nuisance parameters

$$\begin{aligned}\mathrm{PENSSE}(\mathbf{c}|\boldsymbol{\theta},\rho) &= \frac{(1-\rho)}{n}\|\mathbf{y}-\boldsymbol{\Phi}\mathbf{c}\|^2 + \\ &\quad \frac{\rho}{(t_{max}-t_0)}\int_{t_0}^{t_{max}}\left[\mathbf{c}'\frac{\mathrm{d}^p\boldsymbol{\phi}(t)}{\mathrm{d}t^p} - \right. \\ &\quad \left.\sum_{r=0}^{p-1}\beta_r(t|\boldsymbol{\theta})\mathbf{c}'\frac{\mathrm{d}^r\boldsymbol{\phi}(t)}{\mathrm{d}t^r} - \sum_{q=1}^{Q}\alpha_q(t|\boldsymbol{\theta})u_q(t)\right]^2 \mathrm{d}t,\end{aligned}$$

Inner optimization: nuisance parameters

Let $\mathbf{R}(\boldsymbol{\theta})$ be the $K \times K$ matrix

$$\mathbf{R}(\boldsymbol{\theta}) = \int_{t_0}^{t_{max}} \left[\frac{\mathrm{d}^p \boldsymbol{\phi}(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_r(t|\boldsymbol{\theta}) \frac{\mathrm{d}^r \boldsymbol{\phi}(t)}{\mathrm{d}t^r} \right] \times \left[\frac{\mathrm{d}^p \boldsymbol{\phi}(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_r(t|\boldsymbol{\theta}) \frac{\mathrm{d}^r \boldsymbol{\phi}(t)}{\mathrm{d}t^r} \right]' \mathrm{d}t,$$

Inner optimization: nuisance parameters

Let $\mathbf{S}(\boldsymbol{\theta})$ be a $K \times 1$ vector

$$\begin{aligned}\mathbf{S}(\boldsymbol{\theta}) &= \int_{t_0}^{t_{max}} \left[\frac{\mathrm{d}^p \boldsymbol{\phi}(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_r(t|\boldsymbol{\theta}) \frac{\mathrm{d}^r \boldsymbol{\phi}(t)}{\mathrm{d}t^r}\right] \times \\ &\quad \left[-\sum_{q=1}^{Q} \alpha_q(t|\boldsymbol{\theta}) u_q(t)\right]' \mathrm{d}t.\end{aligned}$$

Inner optimization: nuisance parameters

The coefficient values that minimise $\mathrm{PENSSE}(\mathbf{c}|\boldsymbol{\theta}, \rho)$ are given analytically by

$$\begin{aligned}\hat{\mathbf{c}}(\boldsymbol{\theta}, \rho) &= \left[\frac{(1-\rho)}{n}\boldsymbol{\Phi}'\boldsymbol{\Phi} + \frac{\rho}{t_{max} - t_0}\mathbf{R}(\boldsymbol{\theta})\right]^{-1} \times \\ &\quad \left[\frac{(1-\rho)}{n}\boldsymbol{\Phi}'\mathbf{y} - \frac{\rho}{t_{max} - t_0}\mathbf{S}(\boldsymbol{\theta})\right].\end{aligned}$$

Outer optimization: structural parameters

To complete the parameter cascading strategy, we specify an outer fitting criterion to estimate $\boldsymbol{\theta}$,

$$\begin{aligned} H(\boldsymbol{\theta}|\rho) &= \|\mathbf{y} - \boldsymbol{\Phi}\hat{\mathbf{c}}(\boldsymbol{\theta};\rho)\|^2, \\ &= \|\mathbf{y} - \boldsymbol{\Phi}\left[\frac{(1-\rho)}{n}\boldsymbol{\Phi}'\boldsymbol{\Phi} + \frac{\rho}{t_{max}-t_0}\mathbf{R}(\boldsymbol{\theta})\right]^{-1} \\ &\quad \left[\frac{(1-\rho)}{n}\boldsymbol{\Phi}'\mathbf{y} - \frac{\rho}{t_{max}-t_0}\mathbf{S}(\boldsymbol{\theta})\right]\|^2, \end{aligned}$$

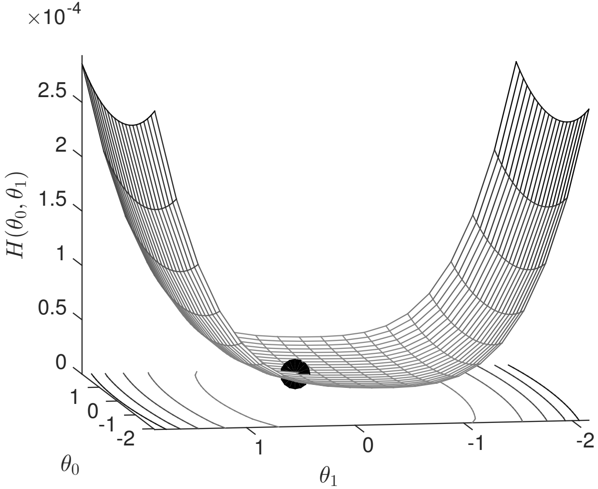
Outer optimization: structural parameters

That is, the parameter estimation problem has been reduced to a nonlinear regression problem.

Numerical optimization methods such as Gauss-Newton are used to minimize $H(\boldsymbol{\theta}|\rho)$ with respect to $\boldsymbol{\theta}$ for fixed ρ. The regulating

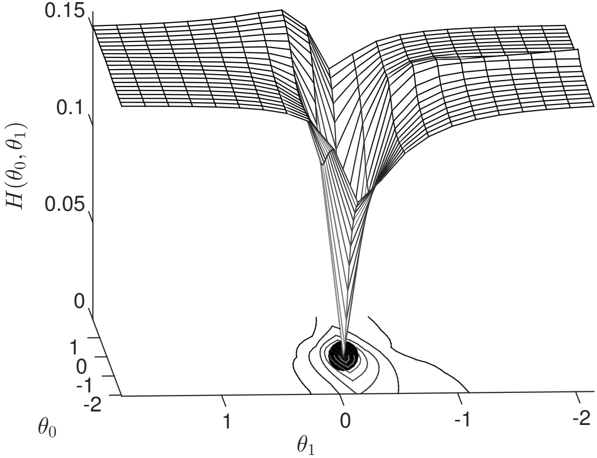
parameter ρ controls the complexity of the surface $H(\boldsymbol{\theta}|\rho)$.

The iteration of ρ



$H(\boldsymbol{\theta}|\rho)$ for the TBI problem, for $\rho = 0.01$.

The iteration of ρ



$H(\boldsymbol{\theta}|\rho)$ for the TBI problem, for $\rho = 0.99$.

The iteration of ρ

As ρ reduces the topology of $H(\boldsymbol{\theta}|\rho)$ tends to a quadratic surface for which the global optimum can be located with only a few iterations.

As ρ approaches one, the topology of $H(\boldsymbol{\theta}|\rho)$ tends to become more complex, with local sharp curved ridges. This leads to a higher probability of local optima in the numerical optimisation.

SUMT

1. Specify an initial trade-off parameter on a logit scale $\rho^{(0)} = \frac{\exp(\gamma)}{1+\exp(\gamma)}$

2. Obtain $\boldsymbol{\theta}^{(u)}$ by minimizing $H(\boldsymbol{\theta}|\rho)$

3. Update the trade-off parameter $\rho^{(u)} = \frac{\exp(\gamma+1)}{1+\exp(\gamma+1)}$.

5. Repeat steps 2 and 3 until ρ reaches ρ_{max}

SUMT

The matrix $\mathbf{R}$ approaches singularity as ρ gets closer to one since it is easy to show that $\mathbf{Rc} = 0$ when $\hat{\mathbf{x}} = \boldsymbol{\Phi}\hat{\mathbf{c}}$ is an exact solution of the differential equation.

Tracing the condition number of $\mathbf{R}$ is therefore one technique for terminating the increasing sequence of ρ values.

A graphical strategy is plot both $\frac{\mathrm{d}^p \mathbf{x}(t)}{\mathrm{d}t^p}$ and the right side of the equation in order to monitor when they are sufficiently close.

Parameter estimation

Most dynamic models for observed systems are only partially founded upon basic scientific principles, usually because a complete model would be too complex to identify given the amount of data that are available.

The simple second order equation used for the head impact data, for example, ignores the wide variation in the mechanical properties of the brain tissue being deformed, as well as the bone material transmitting the shock to the soft tissue inside the skull.

Generalized Cross Validation

To determine if the dynamic model is well specified calculate the $\mathrm{GCV}(\rho)$ for all values of ρ, $\rho^{(0)}, \ldots, \rho_{max}$

$$\mathrm{GCV}(\rho) = \frac{(\mathbf{y} - \mathbf{\Phi}\hat{\mathbf{c}}(\boldsymbol{\theta}; \rho))^T(\mathbf{y} - \mathbf{\Phi}\hat{\mathbf{c}}(\boldsymbol{\theta}; \rho))}{[n - \mathrm{df}]^2}$$

where

$$\hat{\mathbf{c}}(\boldsymbol{\theta}; \rho) = \left[\frac{(1-\rho)}{n}\mathbf{\Phi}'\mathbf{\Phi} + \frac{\rho}{t_{max} - t_0}\mathbf{R}(\boldsymbol{\theta})\right]^{-1} \times$$

$$\left[\frac{(1-\rho)}{n}\mathbf{\Phi}'\mathbf{y} - \frac{\rho}{t_{max} - t_0}\mathbf{S}(\boldsymbol{\theta})\right].$$

$$\mathrm{df} = \mathrm{trace}(\mathbf{\Phi}\left[(1-\rho)/n\mathbf{\Phi}'\mathbf{\Phi} + \rho/(t_{max} - t_0)\mathbf{R}(\boldsymbol{\theta})\right]^{-1}\mathbf{\Phi}')$$

Generalized Cross Validation

If the data is well approximated by the solution of the differential equation then $\mathrm{GCV}(\rho_{max})$ provides the minimum GCV value across all values of ρ, $\rho^{(0)}, \ldots, \rho_{max}$

If the data is not well approximated by the solution of the differential equation,

- Fix $\theta^* = \boldsymbol{\theta}^{u_{max}}$
- Calculate the GCV for $\mathrm{PENSSE}(\mathbf{c}|\theta^*, \rho_i)$ for $i = 0, \ldots, max$.
- Select the estimate of **c** that minimizes GCV.

Statistical inference for $\boldsymbol{\theta}$

The conditional sampling variance of the estimated parameters of the ODE can be approximated by using the delta method:

$$\mathrm{Var}[\hat{\boldsymbol{\theta}}|\rho] \approx \hat{\sigma}_y^2 \left(\frac{\mathrm{d}\hat{\boldsymbol{\theta}}}{\mathrm{d}\mathbf{y}}\right)\left(\frac{\mathrm{d}\hat{\boldsymbol{\theta}}}{\mathrm{d}\mathbf{y}}\right)'.$$

The total derivative of the estimated parameters of the ODE with respect to data **y** is given by,

$$\frac{\mathrm{d}\hat{\boldsymbol{\theta}}}{\mathrm{d}\mathbf{y}} = -\left[\frac{\partial^2 H}{\partial\hat{\boldsymbol{\theta}}^2}\right]^{-1}\left[\frac{\partial^2 H}{\partial\hat{\boldsymbol{\theta}}\partial\mathbf{y}}\right],$$

Statistical inference for $\boldsymbol{\theta}$

The second partial derivative of H with respect to $\hat{\boldsymbol{\theta}}$ is

$$\frac{\partial^2 H}{\partial \hat{\boldsymbol{\theta}}^2} = 2\left(\boldsymbol{\Phi}\frac{\partial \hat{\mathbf{c}}}{\partial \hat{\boldsymbol{\theta}}}\right)'\left(\boldsymbol{\Phi}\frac{\partial \hat{\mathbf{c}}}{\partial \hat{\boldsymbol{\theta}}}\right) - 2(\mathbf{y} - \boldsymbol{\Phi}\hat{\mathbf{c}})'\boldsymbol{\Phi}\frac{\partial^2 \hat{\mathbf{c}}}{\partial \hat{\boldsymbol{\theta}}^2},$$

and the second partial derivative of H with respect to $\hat{\boldsymbol{\theta}}$ and $\mathbf{y}$ is

$$\frac{\partial^2 H}{\partial \hat{\boldsymbol{\theta}} \partial \mathbf{y}} = -2\boldsymbol{\Phi}\frac{\partial \hat{\mathbf{c}}}{\partial \hat{\boldsymbol{\theta}}}.$$

Statistical inference for $\boldsymbol{\theta}$

The partial derivative of $\hat{\mathbf{c}}$ with respect to $\boldsymbol{\theta}$ is

$$\frac{\partial \hat{\mathbf{c}}}{\partial \hat{\boldsymbol{\theta}}} = -\frac{\rho}{t_{max} - t_0} \left[\frac{(1-\rho)}{n} \mathbf{\Phi}' \mathbf{\Phi} + \frac{\rho}{t_{max} - t_0} \mathbf{R}(\boldsymbol{\theta}) \right]^{-1} \frac{\mathrm{d}\mathbf{R}(\boldsymbol{\theta})}{\mathrm{d}\boldsymbol{\theta}} \hat{\mathbf{c}} -$$

$$\frac{\rho}{t_{max} - t_0} \left[\frac{(1-\rho)}{n} \mathbf{\Phi}' \mathbf{\Phi} + \frac{\rho}{t_{max} - t_0} \mathbf{R}(\boldsymbol{\theta}) \right]^{-1} \frac{\mathrm{d}\mathbf{S}(\boldsymbol{\theta})}{\mathrm{d}\boldsymbol{\theta}}$$

Statistical inference for $\hat{x}(t)$

The point-wise sampling variance of $\hat{x}(t)$ is estimated by

$$\mathrm{Var}[\hat{x}(t)|\rho] \approx \boldsymbol{\Phi}' \left(\frac{\mathrm{d}\hat{\boldsymbol{c}}'}{\mathrm{d}\boldsymbol{\theta}} \right) \mathrm{Var}[\hat{\boldsymbol{\theta}}|\rho] \left(\frac{\mathrm{d}\hat{\boldsymbol{c}}}{\mathrm{d}\boldsymbol{\theta}} \right) \boldsymbol{\Phi}.$$

Data2LD package

Data2LD package

The Data2LD package provides utilities for statistical inference on the parameters and solutions of linear differential equations.

Data

Let's assume we have data

$$[x_{i,j}, y_{i,j}]$$

for $i = 1, \dots, N$ and $j = 1, \dots, n_i$. where

- N denotes the number of variables
- n_i denotes the number of observation on the i^{th} variable.

YCell/YLIST

YLIST

- A list of length N.
- Each list contains in turn a struct object with fields:
  - **argvals** a vector of length n_i containing the observation times
  - **y** a matrix with n_i rows and n_{REP} columns containing the observations.
  - n_{REP} the number of replicates (must be the same for all variables).
- If a list is empty, that variable is assumed to be not observed.

Motorcycle impact data

133 observations on the variable Acceleration.

```
Time    Acceleration
 ----    ------------

 2.4          0
 2.6       -1.3
 3.2       -2.7
 3.6          0
   4       -2.7
 6.2       -2.7
 6.6       -2.7
```

Motorcycle impact data

```
#  time in milliseconds
HeadImpactTime <- HeadImpact[,2]

#  acceleration in millimeters/millisecond^2
HeadImpact     <- HeadImpact[,3]

# Define range time for estimated model
HeadImpactRng  <- c(0,60)

#  Define the List array containing the data
yList1 = list(argvals=HeadImpactTime, y=HeadImpact)
yList <- vector("list",1)
yList[[1]] <- yList1
```

BASISLIST

BASISLIST

- A list array of length N.
- Each member contains a basis object representing the basis function expansion for each x_i.

Motorcycle impact data

```
## Set up the basis system

motorng   = c(0,60)
knots = c(0,impact,impact,impact,
impact+1,impact+1,seq(impact+1,60,len=11))
norder    = 6
delta     = 1
nbasis    = 21
motobasis = create.bspline.basis(motorng,nbasis,
                                   norder,knots)

basisList = vector("list",1)
basisList[[1]] = motobasis
```

Differential Equations

The formulation of a single differential equation of order p is

$$\frac{\mathrm{d}^p x_i(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_{i,r}(t) \frac{\mathrm{d}^r x_i(t)}{\mathrm{d}t^r} - \sum_{q=1}^{Q} \alpha_{i,q}(t) u_{i,q}(t).$$

The coefficient functions:

- $\beta_{i,r}$ and $\alpha_{i,q}$ may be non-constant functions over t, often via the values of other known functions, but must not depend on the values of x_i.
- the variables $\alpha_{i,q}(t)u_{i,q}(t)$, $q = 1, \ldots, Q$ are the *forcing functions*

COEFCELL

A cell array of length n_{coef}. Each cell contains a list object with members:

- **parvec** a vector of initial parameters usually 0.01
- **estimate** a binary vector indicating whether the parameter should be estimated (1) or held fixed (0)
- **coeftype** Is the parameter $\beta_r(t)$ ‘beta’ or $\alpha_q(t)$ ‘alpha’

COEFCELL

- **fun** a functional basis, fd, or fdPar object, or a struct object for a general function with fields:
    - **fd** a function handle for evaluating a function
    - **Dfd** a function handle for evaluating the partial derivative of the function with respect to its parameters.
    - **more** object providing additional information for evaluating the function.

Motorcycle impact data

Mechanical principles imply that the acceleration y can be modelled by a second order differential equation with a point impulse u representing the blow to the cranium.

The three parameters β_0, β_1 and α_0 in the second order buffer equation

$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} = \beta_0 x(t) + \beta_1 \frac{\mathrm{d}x(t)}{\mathrm{d}t} + \alpha_0 u(t),$$

convey

- the period of the oscillation,
- the change in its amplitude (as $t \to \infty$ the oscillations decay exponentially to zero)
- the size of the impact from the point impulse, respectively.

Motorcycle impact data

```
# Set up the constant basis
conbasis <- create.constant.basis(HeadImpactRng)

# Define the three constant coefficient functions
coef1 <- make.coef(fun=conbasis, parvec=0.01,
estimate=TRUE, coeftype="beta")

coef2 <- make.coef(fun=conbasis, parvec=0.01,
estimate=TRUE, coeftype="beta")

coef3 <- make.coef(fun=conbasis, parvec=0.01,
estimate=TRUE, coeftype="alpha")
```

Motorcycle impact data

```
coefList  <- vector("list",3)
coefList[[1]] <- coef1
coefList[[2]] <- coef2
coefList[[3]] <- coef3

#  check coefficient list

coefCheckList <- coefCheck(coefList)
coefList      <- coefCheckList$coefList
ntheta        <- coefCheckList$ntheta

print(paste("ntheta = ",ntheta))
```

Differential Equations

The terms in the differential equation

$$\frac{\mathrm{d}^p x_i(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_{i,r}(t) \frac{\mathrm{d}^r x_i(t)}{\mathrm{d}t^r} - \sum_{q=1}^{Q} \alpha_{i,q}(t) u_{i,q}(t).$$

XLIST

XList a list whose length is the number of homogeneous terms:

- **variable** the index of the variable *i*
- **derivative** the derivative of the variable
- **ncoef** the position of the corresponding coefficient in coefList
- **factor** a scalar multiplier (def. 1)
- **estimate** indicating whether the variable should be estimated (1) or held fixed (0)

XLIST

$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} = \beta_0 x(t) + \beta_1 \frac{\mathrm{d}x(t)}{\mathrm{d}t} + \alpha_0 u(t),$$

Motorcycle impact data

```
# Define the two terms in the homogeneous
# part of the equation

Xterm0 <- make.Xterm(variable=1, derivative=0,
ncoef=1, factor=-1, estimate=1)

Xterm1 <- make.Xterm(variable=1, derivative=1,
ncoef=2, factor=-1, estimate=1)

# Set up the XList vector of length two
XList <- vector("list",2)
XList[[1]] <- Xterm0
XList[[2]] <- Xterm1
```

Differential Equations

The forcing functions

$$\frac{\mathrm{d}^p x_i(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_{i,r}(t) \frac{\mathrm{d}^r x_i(t)}{\mathrm{d}t^r} - \sum_{q=1}^{Q} \alpha_{q,i}(t) u_{q,i}(t).$$

FLIST

FList a list whose length is the number of forcing terms

- **Ufd** an fd object for the forcing function
- **ncoef** the position of the corresponding coefficient in coefList
- **factor** a scalar multiplier (def. 1)

FLIST

$$\frac{\mathrm{d}^2 y}{\mathrm{d}t^2} = \beta_0 y + \beta_1 \frac{\mathrm{d}y}{\mathrm{d}t} + \alpha_0 u(t),$$

Motorcycle impact data

```
# Define a unit pulse function located at times 14-15
Pulsebasis <- create.bspline.basis(HeadImpactRng, ...
3, 1, c(0,14,15,60))

Pulsefd    <- fd(matrix(c(0,1,0),3,1),Pulsebasis)

# Define the forcing term
Fterm      <- make.Fterm(ncoef=3, Ufd=Pulsefd, factor=1)

# Set up the forcing term list of length one
FList      <- vector("list",1)
FList[[1]] <- Fterm
```

MODELLIST

A list array of length N. Each list contains a struct object with members:

- **XList** list of homogeneous terms
- **FList** list of forcing terms
- **order** the highest order of derivative p

Motorcycle impact data

```
#  Define the single differential equation in the model
HeadImpactVariable <- make.variable(order=2,
                          XList=XList, FList=FList)

#  Define list of length one containing
#  the equation definition
HeadImpactList=vector("list",1)
HeadImpactList[[1]] <- HeadImpactVariable

#  check the object for internal consistency
HeadImpactList <- modelCheck(HeadImpactList, coefList)
```

Data2LD

Approximates the data in argument YLIST by one or smooth functions x_i, $i = 1, \ldots, N$.

This approximation is defined by a set of linear differential equations defined by a set of parameters some of which require estimation from the data.

Data2LD

The approximation minimizes the sum of squared residuals, expressed as:

$$H(\theta) = \sum_{i}^{N} \sum_{j}^{n_i} \sum_{\kappa}^{d} [y_{ij\kappa} - x_i(t_j)]^2$$

where:

$i = 1, \ldots, N$ indexes equations in a dynamical system.

$j = 1, \ldots, n_i$ indexes times of observation of a variable.

$\kappa = 1, \ldots, d$ indexes replications of observations.

Data2LD

$$x_i(t_j) = \sum_{k}^{K_i} c_{ik}(\theta|\rho)\phi_{ik}(t_j)$$

where

- $\phi_{ik}(t_j)$ is the k^{th} function in a system of K_i basis functions used to approximate the i^{th} variable.
- The number of K_i basis functions and the type of basis function system can vary from one variable to another.
- The coefficients $c_{ik}(\theta|\rho)$ defining the smoothing functions are functions of the unknown parameters θ
- The smoothing parameter ρ is a value in the interval $[0, 1)$.

Data2LD

The coefficient functions $\hat{c}_{ik}(\theta|\rho)$ minimize

$$PENSSE(c_{ik}|\theta) = \frac{(1-\rho)}{n_i} \sum_{j}^{n_i} [y_{ij} - \sum_{k}^{K} c_{ik}\phi_{ik}(t_j)]^2 +$$

$$\frac{\rho}{t_{max} - t_0} \int_{t_0}^{t_{max}} [L(x_i(t)|\theta)]^2 dt$$

where

- $L(x_i(t)|\theta)$ contains the linear differential equation for the i^{th} variable
- As the smoothing parameter ρ increases toward its upper limit of 1, the second term in $PENSSE(c_{ik}|\theta)$ is emphasized and the fit to the data is less emphasized, so that x_i is required to approach a solution to its respective equation.

Data2LD

$$L(x(t)|\theta) = \frac{\mathrm{d}^p x_i(t)}{\mathrm{d}t^p} - \sum_{r=0}^{p-1} \beta_{ir}(t) \frac{\mathrm{d}^r x_i(t)}{\mathrm{d}t^r} - \sum_{q=1}^{Q} \alpha_{q,i}(t) u_{qi}(t).$$

The ODE parameter estimation problem is the estimation of the coefficient functions:

- $\beta_{ir}(t)$
- $\alpha_{qi}(t)$

Data2LD

A coefficient function need not depend linearly on the parameters defining it, but the special case where these are defined by linear expansions

$$\beta_r(t) \approx \sum_{j=1}^{J_r} b_{jr}\psi_{jr}(t) \quad \text{and} \quad \alpha_q(t) \approx \sum_{\ell=1}^{L_q} a_{\ell q}\xi_{\ell q}(t),$$

where $\psi_{jr}(t)$ and $\xi_{\ell q}(t)$ are known basis functions.

The parameters of the differential equation will be collectively referred to as the vector

$$\boldsymbol{\theta} = [b_{10}, \dots, b_{J_{(p-1)}(p-1)}, a_{11}, \dots, a_{L_Q Q}].$$

Data2LD

```
Data2LD <- function(yList, XbasisList, modelList,
           coefList, rhoVec = 0.5*rep(1,nvar),
           loadTensor = FALSE)
```

Input Arguments:

- **YLIST** A list array containing the data.
- **BASISLIST** A list array containing the basis function expansion for x_i
- **MODELLIST** A list array containing the specification for the terms in the differential equation.

Data2LD

- **COEFCELL** A list array containing the specification for the parameters of the differential equation.
- **RHOVEC** A vector containing the complexity parameters
- **LoadTensor** If nonzero, attempt to load the lists BtensorList, BAtensorList and AtensorList.

Data2LD

Output Arguments:

- **MSE** A vector of length N containing the error sum of squares for each variable divided by the number of replications N and by the number of observations n_i.
- **DMSE** A vector of length n_θ that is the gradient of H with respect to θ.
- **D2MSE** A square symmetric matrix of order n_θ that contains the expected values of the second partial derivatives of H with respect to θ

Data2LD

Output Arguments:

- **XFDLIST** A list of length N containing functional parameter objects of class fdPar for the estimated functions $x_i(t)$.
- **DF** An equivalent degrees of freedom value.
- **GCV** The generalized cross-validation measure.
- **ISE** The sum across variables of the integrated squared value of the differential operator.

Data2LD

Output Arguments:

- **RMAT** A square symmetric matrix of order equal to the sum of the coefficients in the basis function expansions of the variables.
- **SMAT** Either a vector of length equal to the order of RMAT if there is only one replication, or a matrix with number of columns equal to the number of replications NREP.
- **Y2CMAP** This matrix is the linear map from the data to the combined coefficients.

Motorcycle impact data

```
#  sequence of values of rho
gammavec <- c(0:8)
rhoVec   <- exp(gammavec)/(1+exp(gammavec))

#Define the complexity
rhoi <- rhoVec[1]

#Data2LD
DataListResult <- Data2LD(yList, basisList,
             HeadImpactList, coefList, rhoi)
```

Data2LD

```
Data2LD.opt <- function(yList, XbasisList,
    modelList, coefList, rhoMat,
    convcrit=1e-6, iterlim=20, dbglev=1,
    parMap=diag(rep(1,npar)))
```

- **CONVCRIT** The convergence criteria. The first criterion is applied to the function change and the second is applied to the gradient norm.
- **ITERLIM** Maximum number of iterations allowed.

Data2LD

- **DBGLEV** An integer controlling the amount of output per iteration. Defaults to 1, which prints summary results for each iteration.

- **PARMAP** A rectangular matrix with number of rows equal to the number of parameters to be estimated and number of columns equal to the number of parameters less the number of linear constraints on the estimated parameters.

Data2LD

```
# Optimization of the criterion

#  algorithm constants
dbglev   <-  1    #  debugging level
iterlim  <- 50    #  maximum number of iterations
convrg  <- c(1e-8, 1e-4)  #  convergence criterion

#  sequence of values of rho
gammavec <- c(0:8)
rhoVec   <- exp(gammavec)/(1+exp(gammavec))
nrho     <- length(rhoVec)
```

Data2LD

```
#  Matrices to hold results
dfesave <- matrix(0,nrho,1)
gcvsave <- matrix(0,nrho,1)
MSEsave <- matrix(0,nrho,1)
ISEsave <- matrix(0,nrho,1)
thesave <- matrix(0,nrho,ntheta)

#  Initialize coefficient list
coefList.opt <- coefList
```

Data2LD

```
#  Loop through rho values
for (irho in 1:nrho) {
    rhoi <- rhoVec[irho]
    print(paste(" -----  rhoVeci <- ", round(rhoi,4),
                " ------------------"))
    Data2LDResult <- Data2LD.Opt(yList, XbasisList,
                          HeadImpactList, coefList.opt,
                          rhoi, convrg, iterlim, dbglev)
    theta.opti     <- Data2LDResult$theta
```

Data2LD

```
        coefList.opti  <- modelVec2List(theta.opti, coefList)
        DataListResult <- Data2LD(yList, XbasisList,
                          HeadImpactList, coefList.opti, rhoi)

        thesave[irho,] <- theta.opti
        dfesave[irho]  <- DataListResult$df
        gcvsave[irho]  <- DataListResult$gcv
        MSEsave[irho]  <- DataListResult$MSE
        ISEsave[irho]  <- DataListResult$ISE
        coefList.opt   <- coefList.opti
    }
```

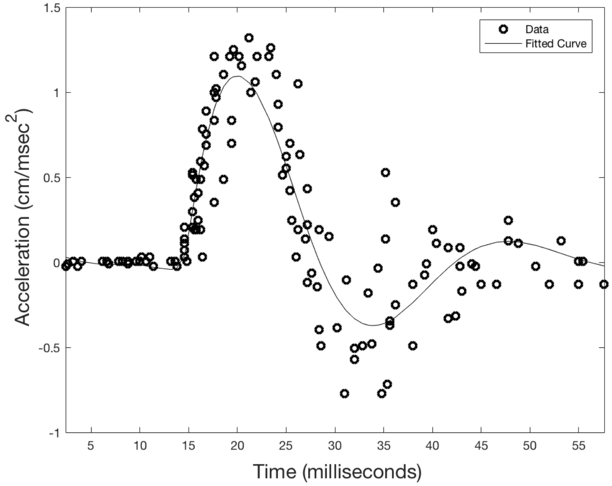
Data2LD

```
# display the optimal parameter values
print("  rho   Stiffness   Damping   Forcing")
print(round(cbind(rhoVec,thesave),4))

# display degrees of freedom and gcv values
print(" rho    df      gcv")
print(round(cbind(rhoVec, dfesave, gcvsave),4))
```

Data2LD

```
# plot fit to the data
delta  <- 1
impact <- 14

plot(HeadImpactTimefine, HeadImpactfine, type="l",
     xlim=c(0,60), ylim=c(-1,1.5),
     xlab="Time (msec)", ylab="Acceleration (cm/msec^2)")
points(HeadImpactTime, HeadImpact, pch="o")
lines(c(impact,      impact),       c(0,1), lty=2)
lines(c(impact+delta,impact+delta), c(0,1), lty=2)
lines(c(impact+delta,impact+delta), c(0,1), lty=2)
lines(c(impact,      impact+delta), c(1,1), lty=2)
lines(c(0,60), c(0,0), type="l", lty=3)
```

Data2LD



Data
Fitted Curve
1.5
1
0.5
0
-0.5
-1
5
10
15
20
25
30
35
40
45
50
55
Time (milliseconds)
Acceleration (cm/msec^2)

Data2LD

```
# compute standard error and confidence
# limits for forcing

stderr  <- sqrt(diag( Data2LDResult$Var.theta))
theta   <- thesave[irho,]

thetaUP <- theta + 2*stderr
thetaDN <- theta - 2*stderr
```

Data2LD

```
#  display parameters along with confidence limits
#  and standard error

print("rate constant beta0, confidence limits
and std. error")

print(round(c(theta[1], thetaDN[1], thetaUP[1],
stderr[1]),4))

print("rate constant beta1, confidence limits
and std. error")

print(round(c(theta[2], thetaDN[2], thetaUP[2],
 stderr[2]),4))
```

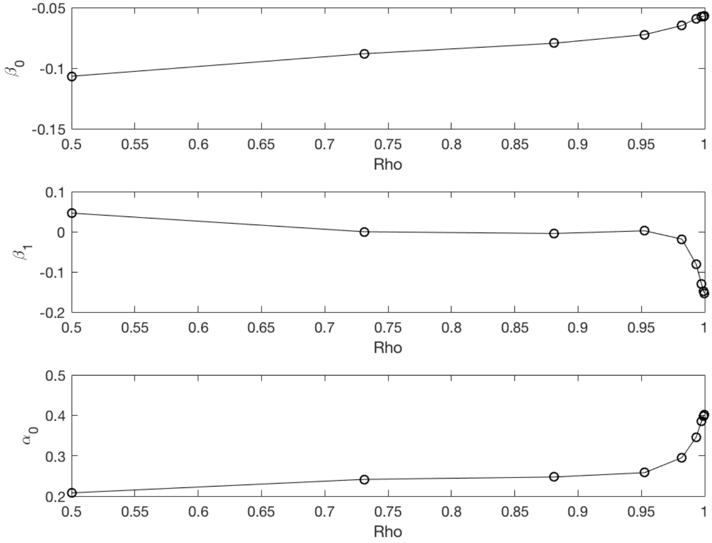
Data2LD

```
#  display parameters along with confidence limits
#  and standard error

print("rate constant alpha, confidence limits
and std. error")

print(round(c(theta[3], thetaDN[3], thetaUP[3],
stderr[3]),4))
```

Data2LD

```
#  plot the evolution of the parameters over
#  the values of rho

par(mfrow=c(1,1))

matplot(rhoVec, thesave, type="b", pch="o",
     xlab="\rho", ylab="parameter")
```

Data2LD



-0.05
-0.1
-0.15
β_0
0.5
0.55
0.6
0.65
0.7
0.75
0.8
0.85
0.9
0.95
1
Rho
0.1
0
-0.1
-0.2
β_1
0.5
0.55
0.6
0.65
0.7
0.75
0.8
0.85
0.9
0.95
1
Rho
0.5
0.4
0.3
0.2
α_0
0.5
0.55
0.6
0.65
0.7
0.75
0.8
0.85
0.9
0.95
1
Rho