Orthonormal bases of functions

April 24, 2018

Outline

Downloading R-package and first steps

Downloading R-package and first steps

- Statistical R-package available for free download here

Downloading R-package and first steps

- Statistical R-package available for free download here
- Available on any PC platform (Mac, Windows, Linux).

Downloading R-package and first steps

- Statistical R-package available for free download here
- Available on any PC platform (Mac, Windows, Linux).
- Worry free and fast downloading procedure (a couple of minutes).

Downloading R-package and first steps

- Statistical R-package available for free download here
- Available on any PC platform (Mac, Windows, Linux).
- Worry free and fast downloading procedure (a couple of minutes).
- We will be working in the command line window of R (most direct way of accessing R-package).

Downloading R-package and first steps

- Statistical R-package available for free download here
- Available on any PC platform (Mac, Windows, Linux).
- Worry free and fast downloading procedure (a couple of minutes).
- We will be working in the command line window of R (most direct way of accessing R-package).
- No experience is required – all of the code that will be needed will be provided on our webpage!

Downloading R-package and first steps

- Statistical R-package available for free download here
- Available on any PC platform (Mac, Windows, Linux).
- Worry free and fast downloading procedure (a couple of minutes).
- We will be working in the command line window of R (most direct way of accessing R-package).
- No experience is required – all of the code that will be needed will be provided on our webpage!
- There some so-called R front-ends (such *R Commander* or *R-Studio* or *Jupyter*) that ease writing more complex programming in R – while you can use and utilize them, I assume **only a very basic R installation** with the primitive *copy-and-paste-to-the-command-line* approach as a method of running the programs.

Getting access to ‘fda’ package

- After installing R, any package available on the r-project webpage can be easily downloaded and installed on individual computers.
- This also applies to ’fda’ package.
- The instructions for downloading and launching a package differ depending on platform.
- All are however straightforward and take no longer than a minute.
- In the package are all scripts that have been used in our main textbook.

```
system.file('scripts', package='fda')
# [1] "/Users/mats-ksp/Library/R/3.4/library/fda/scripts"
dir( "/Users/mats-ksp/Library/R/3.4/library/fda/scripts")
# [1] "afda-ch01.R"   "afda-ch02.R"   "afda-ch03.R"   "afda-ch04.R"   "afda-ch05.R"   "afda-
# [7] "afda-ch07.R"   "fda-ch01.R"    "fda-ch02.R"    "fda-ch03.R"    "fda-ch07.R"    "fda-c
#[13] "fda-ch13.R"    "fda-ch17.R"    "fdarm-ch01.R" "fdarm-ch02.R" "fdarm-ch03.R" "fdarm
#[19] "fdarm-ch05.R" "fdarm-ch06.R" "fdarm-ch07.R" "fdarm-ch08.R" "fdarm-ch09.R" "fdarm
#[25] "fdarm-ch11.R"
```

Example of a very simple R session with fda package

Example of a very simple R session with fda package

- This is based on the script `"afda-ch01.R"` availabe in the fda package.

Example of a very simple R session with fda package

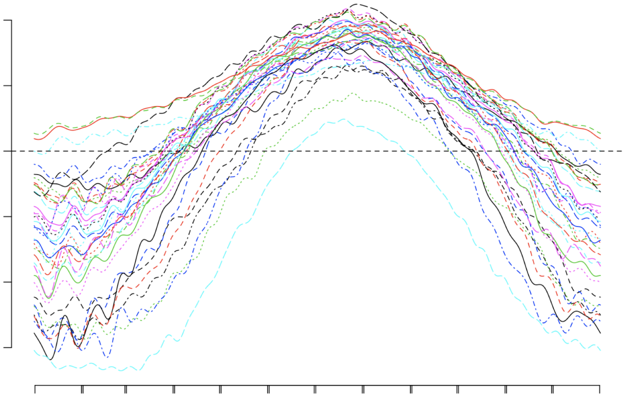
- This is based on the script "`afda-ch01.R`" availabe in the fda package.
- If the following lines of code do not create the error messages then the package is properly installed:

```
daybasis65 <- create.fourier.basis(rangeval=c(0, 365), nbasis=65)

#  -----------  set up the harmonic acceleration operator  ----------

harmaccelLfd365 <- vec2Lfd(c(0,(2*pi/365)^2,0), c(0, 365))

#  ---------  create fd objects for temp. and prec. ---------------

# First check the distribution
qqnorm(CanadianWeather$dailyAv[,,"Temperature.C"], datax=TRUE)
# Consistent with a strong annual cycle
# plus weaker normal noise

daytempfd <- with(CanadianWeather, smooth.basis(day.5,
                                                dailyAv[,,"Temperature.C"],
                                                daybasis65, fdnames=list("Day", "Statio
plot(daytempfd, axes=FALSE)
axisIntervals(1)
axis(2)
```

Smoothed Canadian weather

Smoothed Canadian weather

- Few lines of the code from the previous result should produce



Canadian average annual weather cycle

Canadian average annual weather cycle

- Description: Daily temperature and precipitation at 35 different locations in Canada averaged over 1960 to 1994.
- CanadianWeather is a list with the following components:
  - dailyAv: a three dimensional array c(365, 35, 3) summarizing data collected at 35 different weather stations in Canada on the following:
    [,,1] = [,, 'Temperature.C']: average daily temperature for each day of the year
    [,,2] = [,, 'Precipitation.mm']: average daily rainfall for each day of the year rounded to 0.1 mm.
    [,,3] = [,, 'log10precip']: base 10 logarithm of Precipitation.mm after first replacing 27 zeros by 0.05 mm.
  - place: Names of the 35 different weather stations in Canada whose data are summarized in 'dailyAv'. These names vary between 6 and 11 characters in length. By contrast, daily[["place"]] which are all 11 characters, with names having fewer characters being extended with trailing blanks.
  - province: names of the Canadian province containing each place
  - coordinates: a numeric matrix giving 'N.latitude' and 'W.longitude' for each place.
  - region: Which of 4 climate zones contain each place: Atlantic, Pacific, Continental, Arctic.
  - monthlyTemp: A matrix of dimensions (12, 35) giving the average temperature in degrees celcius for each month of the year.
  - monthlyPrecip: A matrix of dimensions (12, 35) giving the average daily precipitation in milimeters for each month of the year.
  - geogindex: Order the weather stations from East to West to North

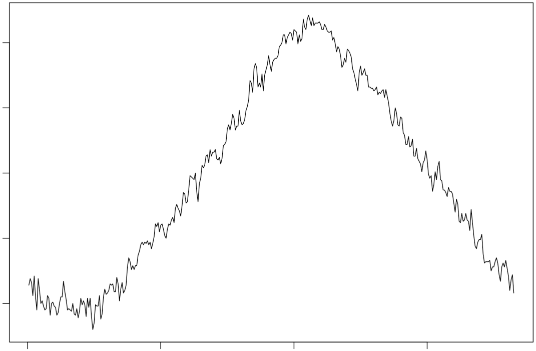
A sample from one station

A sample from one station

- Extracting temperature and plotting it for the first station

```
a=CanadianWeather$dailyAv[,,"Temperature.C"]
plot(a[,1],type="l")
```

- The result



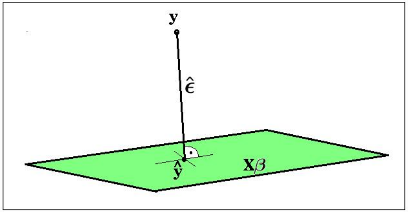
A vector or a function

- We have seen an example of data for which a functional data analysis will be applied.
- If one considers a single location, a sample is a 365 dimensional vector of averaged daily temperatures.
- The vector of this size can be easily analyzed using multivariate analysis.
- Why do we want to use functions?

Outline

Vectors and orthogonality – review

- A picture to have in mind



- Inner product.
- Orthogonality.
- Dimension.
- Orthogonal projections

Inner product

- Defintion.
- Properties.
- Geometric interpretation.

Orthogonality

- Defintion.
- Pythagorean Theorem.
- Geometric interpretation.

Dimension

- Definition of a linear basis.
- Orthogonal bases.
- Dimension.

Linear operations - matrices

- Linear transformation (linear operator).
- Representation as a matrix.

Linear regression – example of vector valued conceptualization

- We have response variables $\mathbf{Y} = (Y_1, \dots, Y_n)$ and a matrix of regressor (explanatory variables) $\mathbf{X} = [x_{ij}]_{i=1,j=1}^{n,r}$.
- The assumed model is

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon.$$

- Here β is an unknown vector of the parameters and ϵ is a random noise.
- The goal is to estimate β given the set of observations $(\mathbf{Y}, \mathbf{X})$.

Least squares

- In the least square approach, one estimates β by minimizing the Eucledean distance between the observations and the linear model
$$\|\mathbf{y} - \mathbf{X}\beta\|^2.$$
- In other words, we want to find a point $\hat{\beta}$ such that $\mathbf{X}\hat{\beta}$ is the closet point to the observations $\mathbf{y}$ among all $\mathbf{X}\beta$'s.
- It follows from the principles of linear algebra that $\mathbf{X}\hat{\beta}$ is the orthogonal projection $\hat{\mathbf{y}}$ of the observation vector $\mathbf{y}$ to the space of spanned by the columns of the design matrix $\mathbf{X}$. This projection can be expressed as the matrix $\mathbf{P} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$, so that
$$\hat{\mathbf{y}} = \mathbf{P}\mathbf{y}$$
$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$

Outline

Why functions?

- Since our data are vectors (always!), why to bother about functions?

Why functions?

- Since our data are vectors (always!), why to bother about functions?
- Because:
  - The vector based analysis is invariant on the permutations of coordinates of vectors. Think about the analysis of the Canadian weather data after randomly permuting coordinates.
  - The dimension reduction: high dimensional data maybe represented using few functions (one function has infinitely many coordinates so the size does not matter).
  - Certain problems can be only expressed in the functional terms: increase of the sampling frequency of the data, search for an efficient way of representing certain structural features in the data – selection of the basis and its optimality (think images).
  - Possibility of using operations that are specific for functions: derivatives, integrals.
  - Certain physical phenomena are conveniently described through functions and their derivatives: response to a signal.

Why functions?

- Since our data are vectors (always!), why to bother about functions?
- Because:
  - The vector based analysis is invariant on the permutations of coordinates of vectors. Think about the analysis of the Canadian weather data after randomly permuting coordinates.
  - The dimension reduction: high dimensional data maybe represented using few functions (one function has infinitely many coordinates so the size does not matter).
  - Certain problems can be only expressed in the functional terms: increase of the sampling frequency of the data, search for an efficient way of representing certain structural features in the data – selection of the basis and its optimality (think images).
  - Possibility of using operations that are specific for functions: derivatives, integrals.
  - Certain physical phenomena are conveniently described through functions and their derivatives: response to a signal.
  - Finally,

Why functions?

- Since our data are vectors (always!), why to bother about functions?
- Because:
  - The vector based analysis is invariant on the permutations of coordinates of vectors. Think about the analysis of the Canadian weather data after randomly permuting coordinates.
  - The dimension reduction: high dimensional data maybe represented using few functions (one function has infinitely many coordinates so the size does not matter).
  - Certain problems can be only expressed in the functional terms: increase of the sampling frequency of the data, search for an efficient way of representing certain structural features in the data – selection of the basis and its optimality (think images).
  - Possibility of using operations that are specific for functions: derivatives, integrals.
  - Certain physical phenomena are conveniently described through functions and their derivatives: response to a signal.
  - Finally, why not? The theory is not that much different.

Conceptualization - Hilbert space

- Hilbert space: A collection of vectors with all properties the same as in Euclidean space except finite dimensionality.
- Functions constitute a Hilbert space (there are also other Hilbert spaces).
- A Hilbert space with a countable a basis $\{e_k, k \in Z\}$) with inner product $\langle \cdot, \cdot \rangle$ which generates norm $\| \cdot \|$

Samples in functional spaces

- An important example is the Hilbert space $L^2 = L^2([0,1])$, which is the set of measurable real-valued functions x defined on [0,1] satisfying $\int_0^1 x(t)^2 dt < \infty$.
- Observations are random functions i.e. random variables defined on some common probability space (Ω, A, P) with values in L^2.
- We say that X is integrable if $E\|X\| < \infty$ and square integrable if $E\|X\|^2 < \infty$.

Outline

Fourier bases

- There are several versions of Fourier bases but they all express in terms of sines and cosines.
- The most standard is

$$\{\sqrt{2}\sin(2\pi n x);\ n \in \mathbb{N}\} \cup \{\sqrt{2}\cos(2\pi n x);\ n \in \mathbb{N}\} \cup \{1\}.$$

Fourier bases

- There are several versions of Fourier bases but they all express in terms of sines and cosines.
- The most standard is

$$\{\sqrt{2}\sin(2\pi nx);\ n \in \mathbb{N}\} \cup \{\sqrt{2}\cos(2\pi nx);\ n \in \mathbb{N}\} \cup \{1\}.$$

- Two alternatives are the sine basis

$$\{\sqrt{2}\sin(\pi nx);\ n \in \mathbb{N}\}$$

and the cosine basis

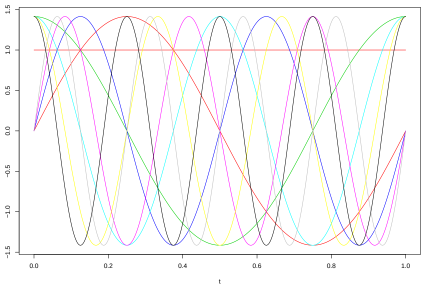
$$\{\sqrt{2}\cos(\pi nx);\ n \in \mathbb{N}\} \cup \{1\}.$$

Fourier bases in R - a direct approach

```
#Fourier basis
nb=45


n=2002    #size of the equidistant one dimensional grid
h=1/n #increment size
t=matrix(seq(h,1,by=h),ncol=1) #grid

Fbase=matrix(0,ncol=n,nrow=nb) #matrix of basis values

Sn=matrix(2*(1:floor(nb/2)),ncol=1)
Cn=matrix(2*(1:ceiling(nb/2))-1,ncol=1)

Fbase[Sn,]=sqrt(2)*sin(pi*Sn%*%t(t)) #sin in basis
Fbase[Cn,]=sqrt(2)*cos(pi*(Cn-1)%*%t(t)) #cos in basis
Fbase[1,]=Fbase[1,]/sqrt(2)



plot(t,Fbase[2,],type="l", col=2,ylab="")
lines(t,Fbase[1,],type="l", col="red")
for(i in 3:9)
{
lines(t,Fbase[i,],type="l", col=i)
}
```

Elements of basis and the fit to a function

The first nine elements of the basis are shown in Figure (left).



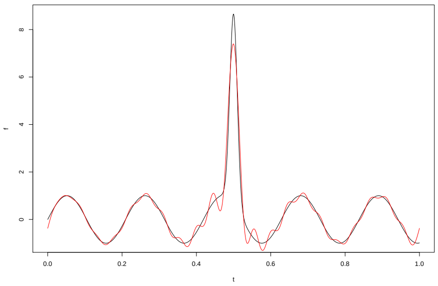


Figure: Fourier basis – *(left)*. A function (in black) and its approximation by an expansion in the Fourier basis (in red) – *(right)*.

The code and the theory – least squares

- One important benefit of using these basis is availability of effective computational algorithm, the Fast Fourier Transform (FFT), for evaluation of the basis expansion coefficients.
- However, in our considerations, we will not benefit greatly from this algorithm so we rather compute the coefficient in a more direct way as shown in the next code.

```
#Test function and its plot
f=t(8*exp(-5000*(t-1/2)^2)+sin(30*t))
plot(t,f,type="l")
#Computation of coefficients

Cf=f%*%t(Fbase)*h

#Computation of expansion (projections)
Pf=Cf%*%Fbase

#Plot of the projection
lines(t,Pf,type="l",col='red')
```

- We can see from the code that it is just the regular least square method applied to the functions.

More on the least squares in the expansion of a function

- O-U basis
- Coefficients of expansion.
- Pythagorean theorem and the optimality of the reminder.

Ploynomial bases

- It follows from the Stone-Weirstrass theorem that every continuous function defined on $[0, 1]$ can be uniformly approximated as closely as desired by a polynomial function.
- Thus at the first sight, it seems reasonable to consider polynomial bases.
- An obvious normalized while not orthonormal basis is the following sequence of monomials

$$\{\sqrt{2n+1}\, x^n; n \in \mathbb{N}_0\}, \tag{1}$$

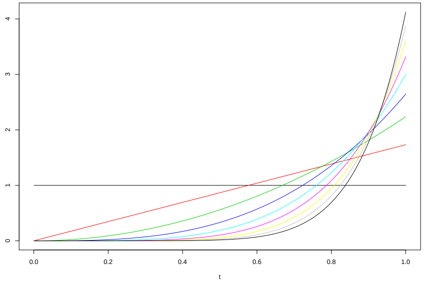
where $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$.

Ploynomial bases

- The elements of this basis are shown in Figure and computed using the following code.

```
#Plotting monomial basis
nb=45
n=2002
h=1/n
t=matrix(seq(h,1,by=h),ncol=1) #grid

n=dim(t)[1]

i=0
Mono=matrix(1,ncol=nb,nrow=n)
Mono[,1]=sqrt(2*i+1)*matrix(1,ncol=1,nrow=n)

for(i in 1:(nb-1))
{
  Mono[,i+1]=(((2*i+1)^(1/(2*i)))*t)^i
}
Max=max(Mono[,1:15])
Min=min(Mono[,1:15])
plot(t,Mono[,1],type='l',ylim=c(Min,Max),ylab='')
for(i in 2:15)
{
  lines(t,Mono[,i],type='l',col=i)
}
```

Graphs



- The monomial basis
- How to make these polynomials orthogonal (orthonormal)?

Gram-Schmidt orthonormalization

- For a sequence of linearly independent elements $h_n \in \mathcal{H}$, $n \in \mathbb{N}_0$, $\mathcal{H}$ is some Hilbert space, the Gram-Schmidt orthonormalization of h_n defines an orthonormalized vectors e_n, $n \in \mathbb{N}_0$, through the following recurrence

$$\begin{aligned} e_0 &= h_0/\|h_0\| \\ e_{n+1} &= \alpha_n e_n + \beta_{n+1} h_{n+1}, n \in \mathbb{N}_0, \end{aligned}$$

where

$$\alpha_n = -\beta_{n+1}\langle e_n, h_{n+1}\rangle, \qquad \beta_{n+1}^2 = \frac{1}{\|h_{n+1}\|^2 - |\langle e_n, h_{n+1}\rangle|^2}.$$

Orthonormalization of polynomials

- It is clear that the sequence of the polynomials obtained through this procedure has increasing order.
- The coefficients of these polynomials can be obtained algebraically through the following simple recurrent linear procedure.
- Let h_k, $k = 1, \ldots, n$, be represented as a sequence of numerical vectors $\mathbf{a}_k$, $k = 1, \ldots, n$, in a certain basis (not necessarily orthonormal and its form is irrelevant for the procedure).
- Moreover let $\mathbf{H} = \left[\langle h_i, h_j \rangle\right]_{i,j=0}^{n}$.
- Define $\mathbf{b}_0 = \mathbf{a}_0 / \| h_0 \| = \mathbf{a}_0 / \sqrt{h_{00}}$, which is the representation of the first vector of the G-S orthonormal basis.
- The second vector can be obtained by defining first

$$\tilde{\mathbf{a}}_k = \mathbf{a}_k - \frac{h_{k0}}{h_{00}} \mathbf{a}_0, \; k = 1, \ldots, n$$
$$\tilde{\mathbf{H}} = \left[h_{ij} - \frac{h_{i0} h_{0j}}{h_{00}} \right]_{i,j=0}^{n}$$

  and then taking $\mathbf{b}_1 = \tilde{\mathbf{a}}_1 / \sqrt{\tilde{h}_{11}}$.
- Moreover, to find the next vector $\mathbf{b}_2$, one applies the same procedure (but by one dimension smaller) to $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{H}}$ in place of $\mathbf{a}$ and $\mathbf{H}$.

Implementation in R

- This approach is implemented in the following routine.

```
#Implementation of Gram–Schmidt orthonormalization
gso=function(A,H)
{
  nb=dim(H)[1]
  k=dim(A)[1]
  HH=H
  AA=A
  B=A #the k x nb, k>=nb matrix output with columns being orthonormalized columns of A
  i=1
  B[,1]=AA[,1, drop=F]/HH[1,1] #the first output vector normalized
  while(nb>1){
    H1=HH[1,,drop=F]                # row 1 x nb
    AA=AA-(H1%x%AA[,1, drop=F])/HH[1,1]           # k x nb
    HH=HH-t(H1)%*%H1/HH[1,1]
    HH=HH[2:nb,2:nb, drop=F]
    AA=AA[,2:nb,drop=F]
    i=i+1
    B[,i]=AA[,1,drop=F]/sqrt(HH[1,1]) #subsequent column to the output
    nb=nb-1
    }
B
}
```

Back to polynomials

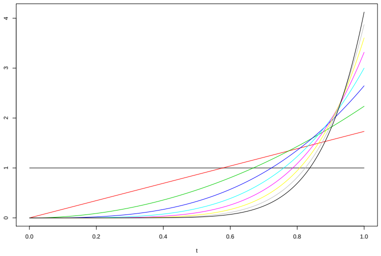
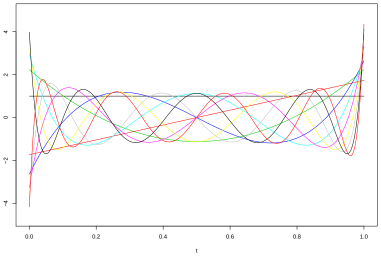
- We note that for the monomial basis the matrix of the inner products is given by

$$\mathbf{H} = \left[\frac{\sqrt{2i+1}\sqrt{2j+1}}{i+j+1} \right]$$

- Thus the following program computes the orthonormal basis.

```
#G-S orthonormalization
nb=12
A=diag(nb)

I=t(rep(1,nb))
S=0:(nb-1)
H=1/(I%x%S+1/2+t(I%x%S+1/2))
S=(sqrt(2*S+1))%*%t(sqrt(2*S+1))
H=H*S

B=gso(A,H)

#Plotting orthonormal polynomial basis

PBase=matrix(1,ncol=nb,nrow=n)
PBase=Mono[,1:nb]%*%B
Max=max(PBase[,1:nb])
Min=min(PBase[,1:nb])
plot(t,PBase[,1],type='l',ylim=c(Min,Max),ylab='')
for(i in 2:10)
{
  lines(t,PBase[,i],type='l',col=i)
}
```

Basis vs Orthornormal basis

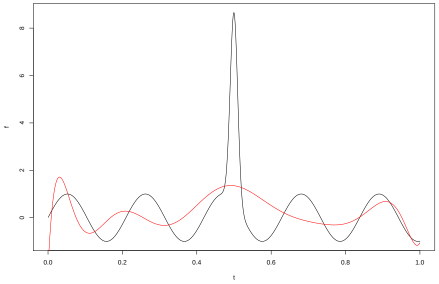
-  

Instability and approximation

- The resulting orthonormal basis was illustrated in Figure.
- Unfortunately, the choice of initial monomials is not wise due to the numerical instability of the orthonormalization of so similarly looking functions.
- Due to this the routine crashes after 13 iterations.
- The routine for computing the expansion of a function is similar as before.

```
#The test function
plot(t,f,type="l")

#Computation of coefficients

Cf=f%*%t(PBase)*h

#Computation of expansion (projections)
Pf=Cf%*%PBase

#Plot of the projection
lines(t,Pf,type="l",col='red')
```

Performance



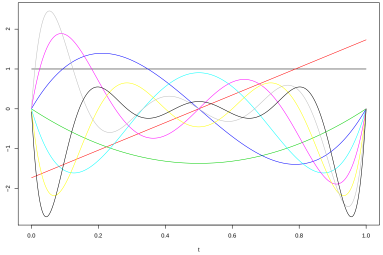
- The approximation using 13 elements of the orthonormalized polynomial basis – *(left)*. We see that 13 polynomials perform poorly.

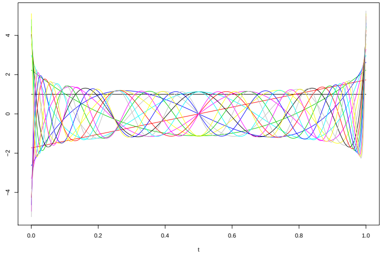
Stabilizing

- To have more fair comparison with the Fourier expansion we need more elements of the orthonormal basis and for this one has to start with a better polynomial basis.
- In fact, it will be more numerically stable to start with the following functions

$$
\begin{aligned}
h_0(x) &= 1,\\
h_1(x) &= x - 1/2,\\
h_n(x) &= \frac{\sqrt{2}}{2e}\prod_{i=0}^{n-1}\left(2\sqrt{e\,n}\left(x-\frac{i}{n-1}\right)\right), \quad n \geq 2,
\end{aligned}
$$

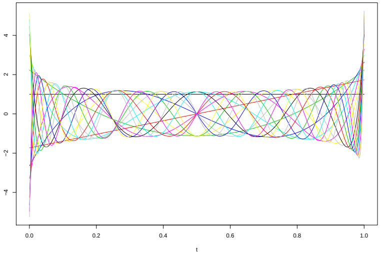
as their provide more variability as shown in the plot, where h_n are normalized before being drawn. A numerically stable initial polynomial basis – *(left)* and the orthonormalized output from the G-S procedure *(right)*.

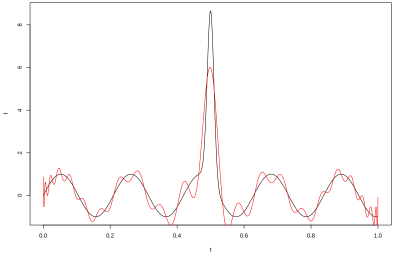




Performance with more basis functions

- The orthonormal polynomial is again attempted using the previous G-S routine and this time 45 elements is obtained and 16 of their elements are shown in Figure *(left)*.
- We observe that the polynomial expansion still does not perform as well as the trigonometric expansion.





16 elements of the orthonormalized polynomial basis – *(left)*. An approximation of the test function with all 45 elements– *(right)*.

Conclusions

- An important lesson from the polynomial basis example is that a choice of the basis may greatly affect the performance of numerical algorithms, even if in the mathematical terms this should not create a problem.
- In fact we have seen that the unique polynomial basis on interval $[0, 1]$ could not be effectively computed if the initial choice of the polynomials was not screened for the variability (the case of the monomials).
- The numerical problem disappeared by considering 'less linearly dependent' initial basis.
- From the mathematical point of view the two bases were equivalent but for computational purposes one performed poorly and the other was quite reasonable.
- Approximation using monomial polynomials is not numerically effective but through a more suitable initial choice of the linear basis to be orthogonalized we arrived to better numerical stability of the Gram-Schmidt orthonormalization.

Analytical solution to the problem of polynomial ON basis

- The good news is that for the case of the orthonormal basis of polynomials on an interval we have an analytical solution to the problem and thus the problem of stability discussed in the previous slides can be avoided.
- It can be derived analytically also using the Gram-Schmidt orthogonalization process to the basis $1, x, x^2, \dots$.
- It will be convenient to present the solution on interval $[-1, 1]$ rather than on $[0, 1]$ but the following transformation leads from the $[-1, 1]$ case to the case of $[0, 1]$:

$$\tilde{f}(y) = \frac{\sqrt{2}}{2} f\left(\frac{y+1}{2}\right).$$

Legendre polynomials

- The elements of the resulting basis for $[-1, 1]$ are known as the Legendre polynomial and their recurrent definition is given next.

Definition

The Legendre polynomials, denoted by $L_k(x)$, are the orthogonal polynomials in $L^2(-1, 1)$. The three-term recurrence relation for the Legendre polynomials reads

$$\begin{aligned} L_0(x) &= 1, \\ L_1(x) &= x, \\ (n+1)L_{n+1}(x) &= (2n+1)xL_n(x) - nL_{n-1}(x), \; n \geq 1. \end{aligned}$$

Theorem

For $n, m \geq 0$ and $x \in [-1, 1]$ we have

$$\int_{-1}^{1} L_m(x)L_n(x)\,dx = \frac{2}{2n+1}\delta_{mn},$$

and an explicit form of the Legendre polynomials as

$$L_n(x) = \sum_{k=0}^{n}(-1)^k \binom{n}{k}^2 \left(\frac{1+x}{2}\right)^{n-k} \left(\frac{1-x}{2}\right)^k.$$

The first six

- The theorem above shows that the Legendre polynomials are orthogonal in $L^2(-1, 1)$.
- The first six Legendre polynomials are:

$$
\begin{aligned}
L_0(x) &= 1, \\
L_1(x) &= x, \\
L_2(x) &= \frac{1}{2}(3x^2 - 1), \\
L_3(x) &= \frac{1}{2}(5x^3 - 3x), \\
L_4(x) &= \frac{1}{8}(35x^4 - 30x^2 + 3), \\
L_5(x) &= \frac{1}{8}(63x^5 - 70x^3 + 15x).
\end{aligned}
$$

- The above polynomials are not normalized but from Theorem it follows that $\tilde{L}_n = \sqrt{n + 1/2}\, L_n$ are normalized.

The bases over the whole real line

- While restricting to an interval and considering functions only on an interval is often practically valid, it may happen that restricting to a specific interval will appear artificial. One could question why this interval is chosen not other.
- Here one has two possible ways of treating such functions within the framework of Hilbert spaces:
  - either limit oneself to the functions that are square integrable over the entire line
  - or consider a weighted inner product that decrease effects of the functional tails.
- Here we only announce an example of the second approach, which leads to an interesting little theory.

The weighted inner product and Gaussian random variable

- By L_2^Φ, we denote the Hilbert space of square integrable functions on $\mathbb{R}$ with respect the Gaussian measure $\Phi(dx) = \phi(x)dx$, where $\phi(x) = e^{-x^2/2}/\sqrt{2\pi}$ so the inner product of $f, g \in L_2^\phi$ is given by

$$\langle f, g \rangle_\phi = \int_{\mathbb{R}} f(x)g(x)\phi(x)\, dx.$$

- Let Z be a standard normal variable, then for $f \in L_2^\Phi$ the random variable $f(Z)$ is has finite second moment and the set of all random variables of such form constitutes a closed subspace of $\mathbb{L}_2$ that is isomorphic to L_2^Φ, where isomorphism is given by

$$\mathbb{E}(f(Z)g(Z)) = \langle f, g \rangle_\phi.$$

Bases of polynomials – Hermite polynomials

- Since all moments of a standard normal random variable are finite, all polynomials are in $\mathbb{L}_2^\Phi$. Moreover, they constitute a dense linear subspace of L_2^Φ. Recall the formula for the moments of the standard normal distribution
$$\mathbb{E}Z^{2k} = \frac{(2k)!}{2^k k!}.$$
Consequently, $M_k(x) = \sqrt{2^k k!/(2k)!}z^k$'s form normalized basis in L_2^Φ.

Definition

The Hermite polynomials H_k, $k \in \mathbb{N}_0$ are the orthonormal elements in L_2^Φ obtained by the Gram-Schmidt orthonormalization procedure applied to M_k, $k \in \mathbb{N}_0$.

Bases of polynomials – Hermite polynomials

- Since the Gram-Schmidt orthonormalization produces H_N as a linear combination of M_k, $k = 0, 1, \ldots, N$ thus H_N is a polynomial of order N and it is orthogonal to M_k, $k = 0, 1, \ldots, N-1$.
- The first six Hermite polynomials:

$$\begin{aligned}
H_0(x) &= 1, \\
H_1(x) &= x, \\
H_2(x) &= (x^2 - 1)/\sqrt{2}, \\
H_3(x) &= (x^3 - 3x)/\sqrt{6}, \\
H_4(x) &= (x^4 - 6x^2 + 3)/\sqrt{24}, \\
H_5(x) &= (x^5 - 10x^3 + 15x)/\sqrt{120}.
\end{aligned}$$

- Notice that H_{2k} are having only even powers of x while H_{2k+1} are having only odd numbered ones, it follows, for example from the next slide.

Bases of polynomials – Hermite polynomials

- Let $P_n(x) = H_n(x)/\alpha_n$, where α_n is the coefficient of H_n at x^n. The polynomials P_n's are referred to as the un-normalized Hermite polynomials. We note the following convenient representation of P_n and H_n.

Theorem

The Hermite polynomials are of the form

$$\begin{aligned} P_n(x) &= (-1)^n e^{x^2/2} \frac{d^n}{dx^n} \left(e^{-x^2/2} \right) \\ H_n(x) &= \frac{(-1)^n}{\sqrt{n!}} e^{x^2/2} \frac{d^n}{dx^n} \left(e^{-x^2/2} \right) \end{aligned} \tag{2}$$

- We have also an explicit form the Hermit polynomials as given in

$$P_n(x) = n! \sum_{k=0}^{[n/2]} \frac{(-1)^k}{2^{2k}} \frac{x^{n-2k}}{(n-2k)!k!}.$$

Locality and Smoothness or Wavelets and Splines

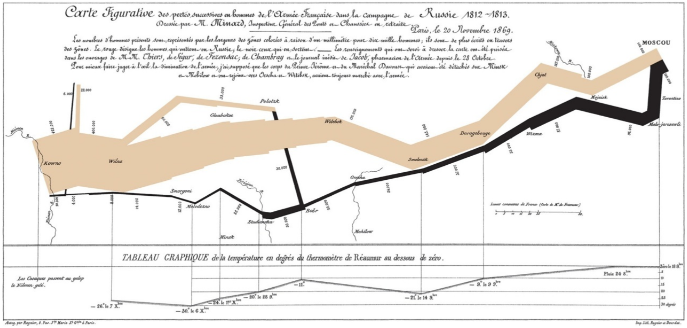
May 2, 2018

Motto

“A picture is worth a thousand words”

Motto

“A picture is worth a thousand words”

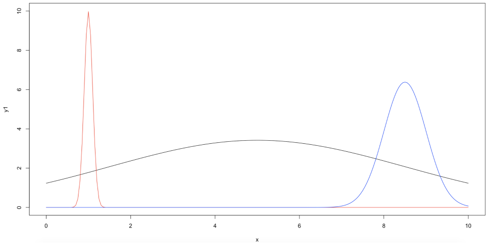


Outline

Overview wavelet bases

- The main idea behind the so called wavelet functions is to represent the local wave like behavior in a signal.
- Inspiration came from physical phenomena but mathematical foundations goes back to Alfréd Haar in 1909.
- The main features of a single wavelet:
    - **a location** – place on horizontal axis (time) where a wavelike disturbance occur
    - **a scale** – how big is the disturbance
    - **a resolution** – how spread is the disturbance around its location, representation of detail
- In a simple approach one could utilize Gaussian curve

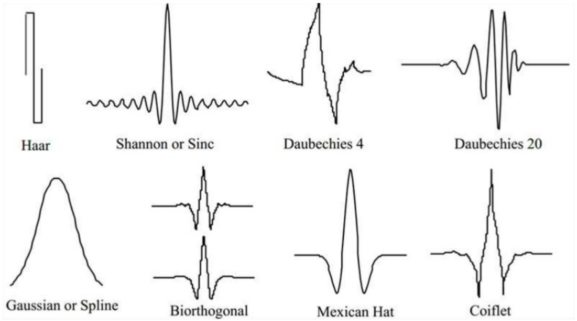
$$f(x; A, m, s) = Ae^{(x-m)^2/s^2}$$

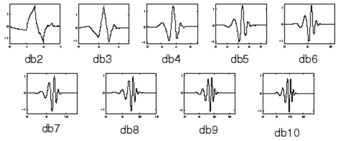


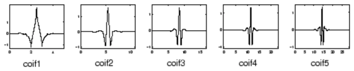
Orthogonalized wavelets

- The only problem with the Gaussian curves that they are not orthogonal
- Can one have curves that have locality and resolution and at the same time to be orthogonal?
- Yes, one can and these are wavelets.
- They are many orthonormal systems with these properties.
- They are generated by the so called **mother wavelets** to which then increasing resolution and scale and locations are added.
- Wavelets in higher dimensions also exist.
- Here we show pictures of them but we will focus from now on only on one system, the oldest and the easiest to understand – Haar wavelets.

Pictures with wavelets







Piecewise constant basis - Haar functions

- Haar functions are the simplest and for most purposes very effective wavelets.
- Define a father wavelet

$$\phi(x) = 1, \;\; x \in [0, 1]$$

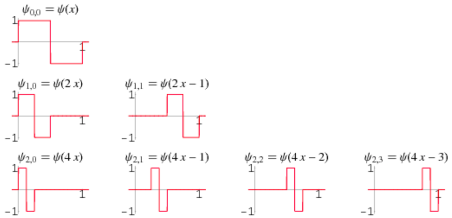
and a mother wavelet

$$\psi(x) = \begin{cases} 1: & 0 \leq x < 1/2; \\ -1: & 1/2 < x \leq 1; \\ 0: & \text{otherwise} \end{cases}$$

and children (orthogonal but not normalized)

$$\psi_{jk}(x) = \psi(2^j x - k)$$

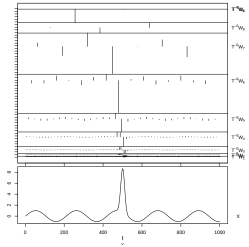
for j a nonnegative integer and $0 \leq k \leq 2^j - 1$.



Discrete Wavelet transform – DWT

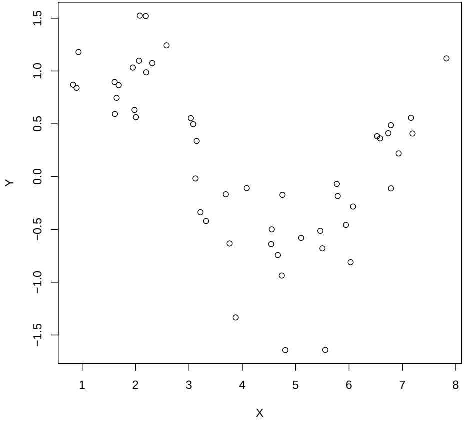
- Similarly as for the Fourier basis one have FFT for fast computations of the coefficients of decomposition of a signal, there is also a fast algorithm for computing wavelet coefficients.
- It is called the discrete Wavelet transform and is implemented in packages such as `wavelets` in R.

```
install.packages("wavelets")
#Test function and its plot
f=t(8*exp(-5000*(t-1/2)^2)+sin(30*t))
plot(t,f,type="l")


WD=dwt(f,filter="haar")
plot(WD)
```



Outline

A picture



Try to sketch a denoised relation between X and Y.

Noisy sine function

Noisy sine function

- Let us consider the following non-linear regression model

$$Y = f(X) + \epsilon$$

where X is an explanatory variable, ϵ is a noisy error and Y is an outcome variable (aka response or dependent variable).

Noisy sine function

- Let us consider the following non-linear regression model

$$Y = f(X) + \epsilon$$

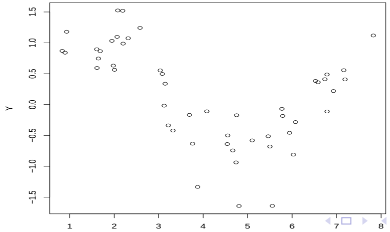
  where X is an explanatory variable, ϵ is a noisy error and Y is an outcome variable (aka response or dependent variable).
- The model is non-linear when $f(X)$ is not a linear function of X. Consider for example $f(X) = \sin(X)$.

Noisy sine function

- Let us consider the following non-linear regression model

$$Y = f(X) + \epsilon$$

  where X is an explanatory variable, ϵ is a noisy error and Y is an outcome variable (aka response or dependent variable).
- The model is non-linear when $f(X)$ is not a linear function of X. Consider for example $f(X) = \sin(X)$.
- A sample from such a model

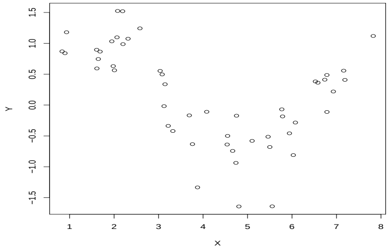


Outline

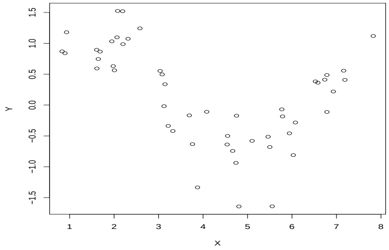
Noisy Sine R-code

```
#Non-linear regression

X=runif(50,0.5,8)
e=rnorm(50,0,0.35)
Y=sin(X)+e

pdf("NoisySine.pdf") #Save a graph to a file
plot(X,Y)
dev.off()            #Closes the graph file
```

How to (re-)discover a non-linear relation

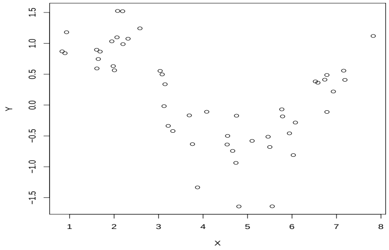


How to (re-)discover a non-linear relation



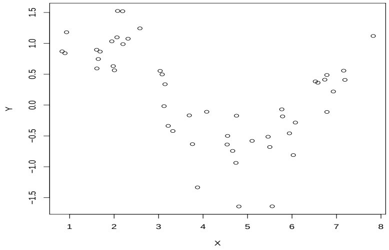
- We are now interested to recover from the above data the relation that stands behind them?

How to (re-)discover a non-linear relation



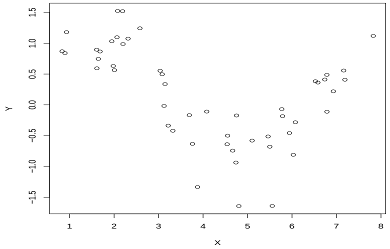
- We are now interested to recover from the above data the relation that stands behind them?
- In practice we do not know that there is any specific function (in this case *sine function*) involved.

How to (re-)discover a non-linear relation



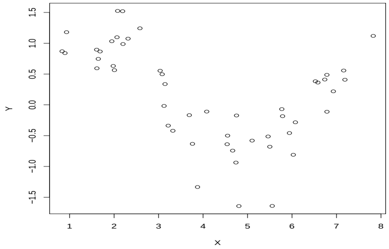
- We are now interested to recover from the above data the relation that stands behind them?
- In practice we do not know that there is any specific function (in this case *sine function*) involved.
- We clearly see that the relation is non-linear.

How to (re-)discover a non-linear relation



- We are now interested to recover from the above data the relation that stands behind them?
- In practice we do not know that there is any specific function (in this case *sine function*) involved.
- We clearly see that the relation is non-linear.
- We want a standardized and automatized approach.

How to (re-)discover a non-linear relation



- We are now interested to recover from the above data the relation that stands behind them?
- In practice we do not know that there is any specific function (in this case *sine function*) involved.
- We clearly see that the relation is non-linear.
- We want a standardized and automatized approach.
- **Any ideas?**

Piecewise constant

Piecewise constant

- We first divide the domain into disjoint regions marked by the **knot points** $\xi_0 < \xi_1 < \cdots < \xi_n < \xi_{n+1}$.

Piecewise constant

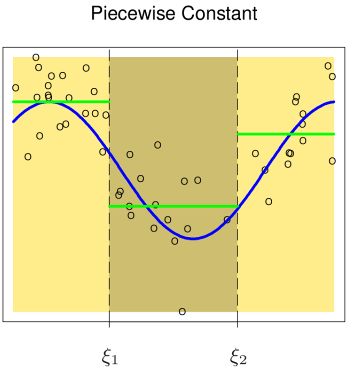
- We first divide the domain into disjoint regions marked by the **knot points** $\xi_0 < \xi_1 < \cdots < \xi_n < \xi_{n+1}$.
- ξ_0 the begining of the x-interval and ξ_{n+1} its end

Piecewise constant

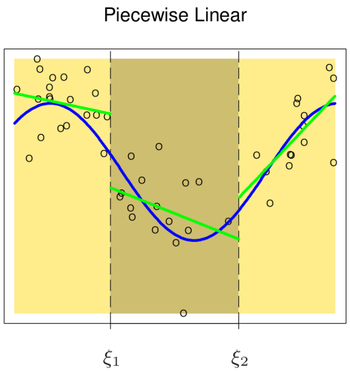
- We first divide the domain into disjoint regions marked by the **knot points** $\xi_0 < \xi_1 < \cdots < \xi_n < \xi_{n+1}$.
- ξ_0 the begining of the x-interval and ξ_{n+1} its end
- On each interval we can fit independently.

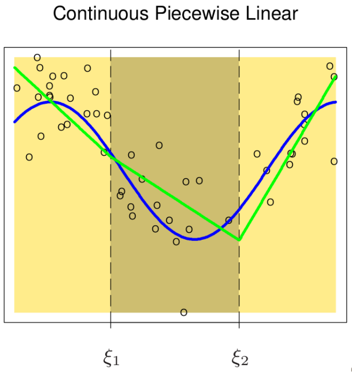
Piecewise constant

- We first divide the domain into disjoint regions marked by the **knot points** $\xi_0 < \xi_1 < \cdots < \xi_n < \xi_{n+1}$.
- ξ_0 the begining of the x-interval and ξ_{n+1} its end
- On each interval we can fit independently.
- For example by constant functions

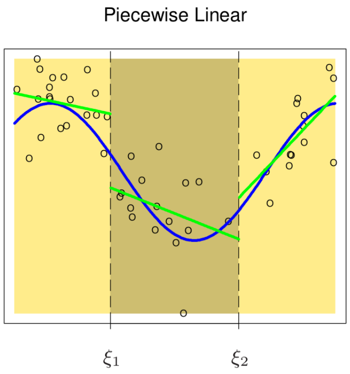


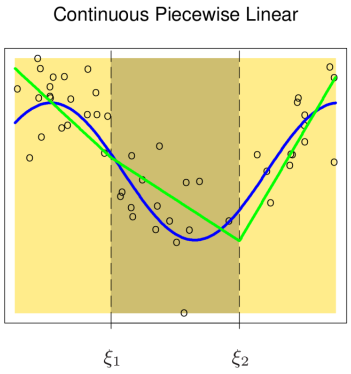
Piecewise linear





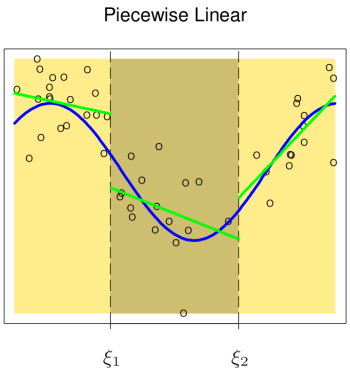
Piecewise linear

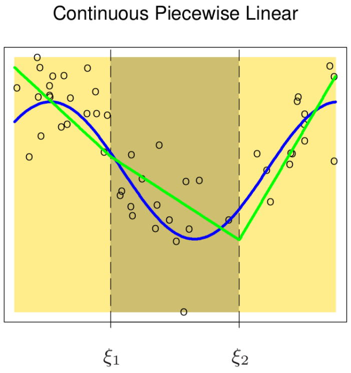




- Where the difference between the two pictures lies?

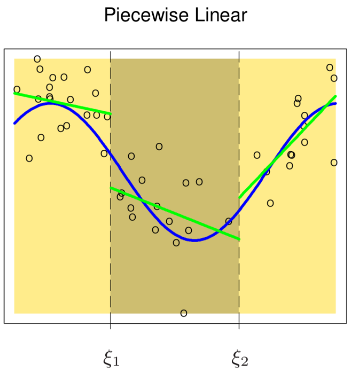
Piecewise linear

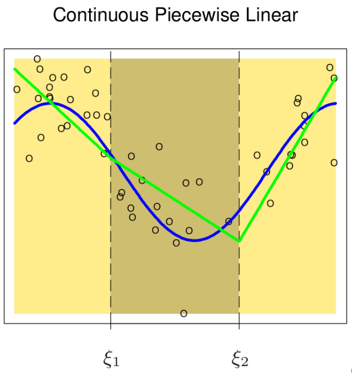




- Where the difference between the two pictures lies?
- The second is continuous – a linear **spline**.

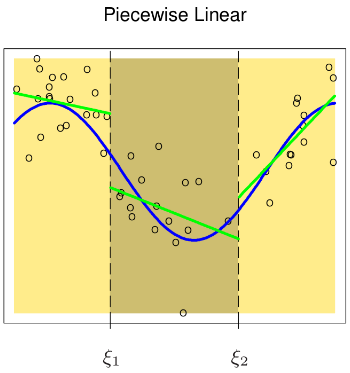
Piecewise linear

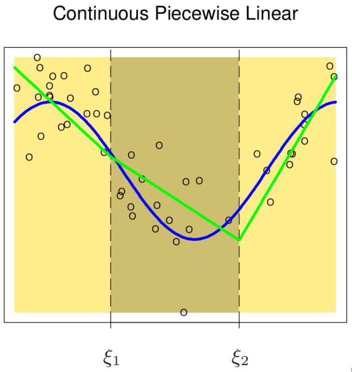




- Where the difference between the two pictures lies?
- The second is continuous – a linear **spline**.
- Fit is **no longer independent** between regions.

Piecewise linear





- Where the difference between the two pictures lies?
- The second is continuous – a linear **spline**.
- Fit is **no longer independent** between regions.
- How to do it?

Analysis of the problem

Analysis of the problem

- How many parameters there are in the problem?

Analysis of the problem

- How many parameters there are in the problem?
- 3-intercepts $+$ 3-slopes $-$ 2-knots $=$ 4 (we subtract knots because each knot sets one equation to fulfill the continuity assumption.

Analysis of the problem

- How many parameters there are in the problem?
- 3-intercepts $+$ 3-slopes $-$ 2-knots $=$ 4 (we subtract knots because each knot sets one equation to fulfill the continuity assumption.
- The problem should be fitted with four parameters.

From now on we assume the knots locations are decided for and not changing.

Making non-linear linear

Making non-linear linear

- What is the minimal number of vectors needed to express linearly any vector in 4 dimensions?

Making non-linear linear

- What is the minimal number of vectors needed to express linearly any vector in 4 dimensions? **4**

Making non-linear linear

- What is the minimal number of vectors needed to express linearly any vector in 4 dimensions? **4**
- Such vectors are (linearly) independent (none is linearly expressed by the remaining ones)

Making non-linear linear

- What is the minimal number of vectors needed to express linearly any vector in 4 dimensions? **4**
- Such vectors are (linearly) independent (none is linearly expressed by the remaining ones)
- Find 4 piecewise linear continuous functions that are ‘independent’, say $h_1(X)$, $h_2(X)$, $h_3(X)$, $h_4(X)$.

Making non-linear linear

- What is the minimal number of vectors needed to express linearly any vector in 4 dimensions? **4**
- Such vectors are (linearly) independent (none is linearly expressed by the remaining ones)
- Find 4 piecewise linear continuous functions that are 'independent', say $h_1(X)$, $h_2(X)$, $h_3(X)$, $h_4(X)$.
- Then any function piecewise linear with the given knots can be written linearly by them

$$f(X) = \beta_1 h_1(X) + \beta_2 h_2(X) + \beta_3 h_3(X) + \beta_4 h_4(X) = \sum_{j=1}^{4} \beta_j h_j(X).$$

Making non-linear linear

- What is the minimal number of vectors needed to express linearly any vector in 4 dimensions? **4**
- Such vectors are (linearly) independent (none is linearly expressed by the remaining ones)
- Find 4 piecewise linear continuous functions that are 'independent', say $h_1(X)$, $h_2(X)$, $h_3(X)$, $h_4(X)$.
- Then any function piecewise linear with the given knots can be written linearly by them

$$f(X) = \beta_1 h_1(X) + \beta_2 h_2(X) + \beta_3 h_3(X) + \beta_4 h_4(X) = \sum_{j=1}^{4} \beta_j h_j(X).$$

- $f(X)$ is continuous in X because each of $h_j(X)$ is.

Making non-linear linear

- What is the minimal number of vectors needed to express linearly any vector in 4 dimensions? **4**
- Such vectors are (linearly) independent (none is linearly expressed by the remaining ones)
- Find 4 piecewise linear continuous functions that are 'independent', say $h_1(X)$, $h_2(X)$, $h_3(X)$, $h_4(X)$.
- Then any function piecewise linear with the given knots can be written linearly by them

$$f(X) = \beta_1 h_1(X) + \beta_2 h_2(X) + \beta_3 h_3(X) + \beta_4 h_4(X) = \sum_{j=1}^{4} \beta_j h_j(X).$$

- $f(X)$ is continuous in X because each of $h_j(X)$ is.
- There are four parameters, so that any continuous piecewise linear function should be fitted by proper choice of β_j's.

Basis functions

Basis functions

- There many choices for h_j, $j = 1, \ldots, 4$.

Basis functions

- There many choices for h_j, $j = 1, \ldots, 4$.
- The following is a natural one

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

where t_+ is a positive part of a real number t.

Basis functions

- There many choices for h_j, $j = 1, \dots, 4$.
- The following is a natural one
  $$h_1(X) = 1,\ h_2(X) = X,\ h_3(X) = (X - \xi_1)_+,\ h_4(X) = (X - \xi_2)_+,$$
  where t_+ is a positive part of a real number t.
- The model for the data
  $$Y_i = \beta_1 h_{i1} + \cdots + \beta_r h_{ir} + \varepsilon_i,$$
  $i = 1, 2, \dots, n$, where $h_{ij} = h_j(X_i)$.

Basis functions

- There many choices for h_j, $j = 1, \dots, 4$.
- The following is a natural one

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

  where t_+ is a positive part of a real number t.
- The model for the data

  $$Y_i = \beta_1 h_{i1} + \cdots + \beta_r h_{ir} + \varepsilon_i,$$

  $i = 1, 2, \dots, n$, where $h_{ij} = h_j(X_i)$.
- The model in the matrix notation

  $$\mathbf{Y} = \mathbf{H}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

  where $\mathbf{H}$ is the matrix of h_{ij}'s.

Basis functions

- There many choices for h_j, $j = 1, \dots, 4$.
- The following is a natural one

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

  where t_+ is a positive part of a real number t.
- The model for the data

  $$Y_i = \beta_1 h_{i1} + \cdots + \beta_r h_{ir} + \varepsilon_i,$$

  $i = 1, 2, \dots, n$, where $h_{ij} = h_j(X_i)$.
- The model in the matrix notation

  $$\mathbf{Y} = \mathbf{H}\beta + \varepsilon,$$

  where **H** is the matrix of h_{ij}'s.
- Fitting problem is solved by fitting the **linear regression problem** (the least squares method).

Extension to smoother version – cubic splines

Extension to smoother version – cubic splines

- The piecewise linear splines have discontinuous derivative at knots.

Extension to smoother version – cubic splines

- The piecewise linear splines have discontinuous derivative at knots. **Why?**

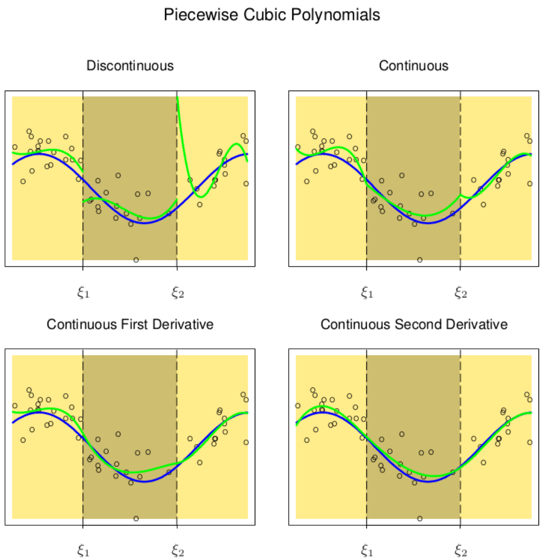
Extension to smoother version – cubic splines

- The piecewise linear splines have discontinuous derivative at knots. **Why?**
- We can increase the order of **smoothness at the knots** by increasing the degree of polynomial that is fitted in each region and then imposing the **continuity constraints at each knot**.

Extension to smoother version – cubic splines

- The piecewise linear splines have discontinuous derivative at knots. **Why?**
- We can increase the order of **smoothness at the knots** by increasing the degree of polynomial that is fitted in each region and then imposing the **continuity constraints at each knot**.
- The cubic splines are quite popular for this purpose.

Illustration – cubic splines



Basis

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is:

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is: **4**

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is: **4**
  - Number of knots is **4** so we have **3** polynomials (we count the right and the left point of the abscissa's range)

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is: **4**
  - Number of knots is **4** so we have **3** polynomials (we count the right and the left point of the abscissa's range)
  - The number of knots where the smoothness constraints are imposed:

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is: **4**
  - Number of knots is **4** so we have **3** polynomials (we count the right and the left point of the abscissa's range)
  - The number of knots where the smoothness constraints are imposed: **2**

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is: **4**
  - Number of knots is **4** so we have **3** polynomials (we count the right and the left point of the abscissa's range)
  - The number of knots where the smoothness constraints are imposed: **2**
  - The number of constraints at a knot to have smooth second derivative:

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is: **4**
  - Number of knots is **4** so we have **3** polynomials (we count the right and the left point of the abscissa's range)
  - The number of knots where the smoothness constraints are imposed: **2**
  - The number of constraints at a knot to have smooth second derivative: **3** (the equations for continuity of the functions and their two derivatives)

  Number of the parameters:

$$3 * 4 - 2 * 3 = 6$$

Basis

- Let us count the number of parameters needed.
  - Number of parameter of a cubic polynomial is: **4**
  - Number of knots is **4** so we have **3** polynomials (we count the right and the left point of the abscissa's range)
  - The number of knots where the smoothness constraints are imposed: **2**
  - The number of constraints at a knot to have smooth second derivative: **3** (the equations for continuity of the functions and their two derivatives)

  Number of the parameters:

$$3 * 4 - 2 * 3 = 6$$

- Example of the basis

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

Outline

Another Basis – B-splines

Another Basis – B-splines

- There are convenient splines that can be defined recursively.

Another Basis – B-splines

- There are convenient splines that can be defined recursively.
- They are called B-splines.

Another Basis – B-splines

- There are convenient splines that can be defined recursively.
- They are called B-splines.
- We consider only the special case of cubic splines (see the textbooks for more general discussion).

Another Basis – B-splines

- There are convenient splines that can be defined recursively.
- They are called B-splines.
- We consider only the special case of cubic splines (see the textbooks for more general discussion).
- Assume $\xi_1, \ldots, \xi_K$ internal knots and two endpoints ξ_0 and ξ_{K+1}.

Another Basis – B-splines

- There are convenient splines that can be defined recursively.
- They are called B-splines.
- We consider only the special case of cubic splines (see the textbooks for more general discussion).
- Assume $\xi_1, \dots, \xi_K$ internal knots and two endpoints ξ_0 and ξ_{K+1}.
- Add three *artificial* knots that are equal to ξ_0 and similarly additional three knots that are equal to ξ_{K+1} for the total of $K + 8$ knots that from now on are denoted by τ_i, $i = 1, \dots, K + 8$.

Another Basis – B-splines

- There are convenient splines that can be defined recursively.
- They are called B-splines.
- We consider only the special case of cubic splines (see the textbooks for more general discussion).
- Assume $\xi_1, \dots, \xi_K$ internal knots and two endpoints ξ_0 and ξ_{K+1}.
- Add three *artificial* knots that are equal to ξ_0 and similarly additional three knots that are equal to ξ_{K+1} for the total of $K+8$ knots that from now on are denoted by τ_i, $i = 1, \dots, K+8$.
- Define recursively functions $B_{i,m}$ that are splines of the $(m-1)$th order of smoothness (0 smoothness is discontinuity at the knots), $i = 1, \dots, K+8$, $m = 1, \dots, 4$

Recursion

Recursion

- For the knots τ_i, $i = 1, \dots, K + 8$ we define $B_{i,m}$, $i = 1, \dots, K + 8$, $m = 1, \dots, 4$

Recursion

- For the knots τ_i, $i = 1, \dots, K+8$ we define $B_{i,m}$, $i = 1, \dots, K+8$, $m = 1, \dots, 4$
- The piecewise constant (0-smooth), $i = 1, \dots, K+7$,

$$B_{i,1}(x) = \begin{cases} 1 & \text{if } \tau_i \leq x < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

Recursion

- For the knots τ_i, $i = 1, \dots, K+8$ we define $B_{i,m}$, $i = 1, \dots, K+8$, $m = 1, \dots, 4$
- The piecewise constant (0-smooth), $i = 1, \dots, K+7$,

$$B_{i,1}(x) = \begin{cases} 1 & \text{if } \tau_i \leq x < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

- Higher $(m-1)$ order of smoothness , $i = 1, \dots, K+8-m$,

$$B_{i,m}(x) = \frac{x - \tau_i}{\tau_{i+m-1} - \tau_i} B_{i,m-1}(x) + \frac{\tau_{i+m} - x}{\tau_{i+m} - \tau_{i+1}} B_{i+1,m-1}(x).$$

Recursion

- For the knots τ_i, $i = 1, \dots, K+8$ we define $B_{i,m}$, $i = 1, \dots, K+8$, $m = 1, \dots, 4$
- The piecewise constant (0-smooth), $i = 1, \dots, K+7$,

$$B_{i,1}(x) = \begin{cases} 1 & \text{if } \tau_i \leq x < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

- Higher $(m-1)$ order of smoothness , $i = 1, \dots, K+8-m$,

$$B_{i,m}(x) = \frac{x - \tau_i}{\tau_{i+m-1} - \tau_i} B_{i,m-1}(x) + \frac{\tau_{i+m} - x}{\tau_{i+m} - \tau_{i+1}} B_{i+1,m-1}(x).$$

- $B_{i,4}$ are cubic order splines that constitutes basis for all cubic splines.

Illustration – evenly distributed knots

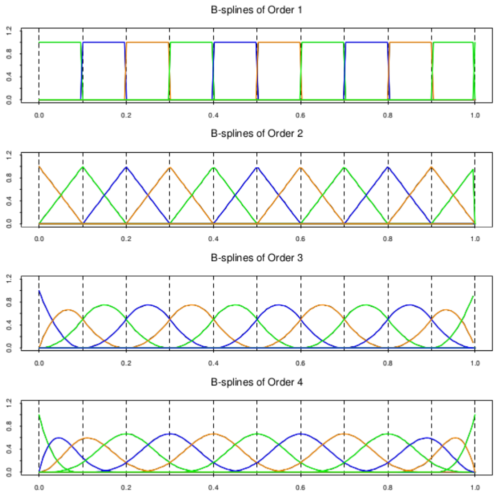
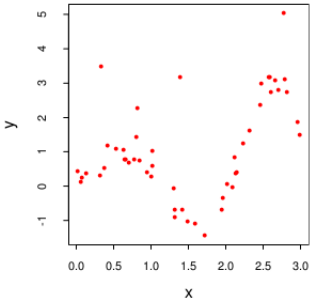
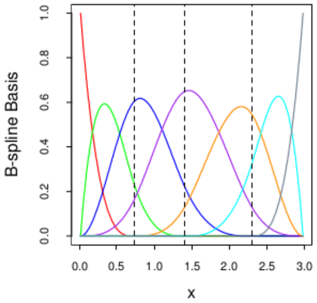


Illustration – non-evenly distributed knots

Illustration – non-evenly distributed knots

- Another data set and B-spline basis





Outline

Splines without knot selection

The regression problem with one predictor

$$y = \alpha + f(x) + \epsilon.$$

- The maximal set of knots: a knot is located at each abscissa location in the data.
- Clearly, without additional restrictions this leads to overfitting and non-identifiability.

Splines without knot selection

The regression problem with one predictor

$$y = \alpha + f(x) + \epsilon.$$

- The maximal set of knots: a knot is located at each abscissa location in the data.
- Clearly, without additional restrictions this leads to overfitting and non-identifiability. **Why?**
- These issues are taken care of since irregularity is penalized.
- Outside the range of predictors it is estimated by a linear function (smoothing on the boundaries).

Penalty for being non-smooth

- Minimize the penalized residual sum of squares

$$PRSS(f,\lambda) = \sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \int f''(t)^2 dt$$

- $\lambda = 0$: any fit that interpolates data exactly.
- $\lambda = \infty$: the least square fit (second derivative is zero)
- We fit by the cubic splines with knots set at all the values of x's and the solution has the form

$$f(x) = \sum_{j=1}^{N+4} \gamma_j B_j(x), \tag{1}$$

where γ_j's have to be found.

B-spline basis

- The splines $B_j(x)$, $j = 1, \dots, N+4$, are used in the smoothing splines, where the initial x_i, $i = 1, \dots, N$ are augmented by 2 end points defining the range of interest for the total of $N+2$ knots.
- We have seen that if there is N internal points, then there have to be $N+4$ of the third order splines in order for them to constitute basis.
- One can compute explicitly the coefficients of the following matrix
$$\boldsymbol{\Omega}_B = \left[\int B_i''(t) B_j''(t)\, dt \right]$$

Solution

- The solution has the following explicit form

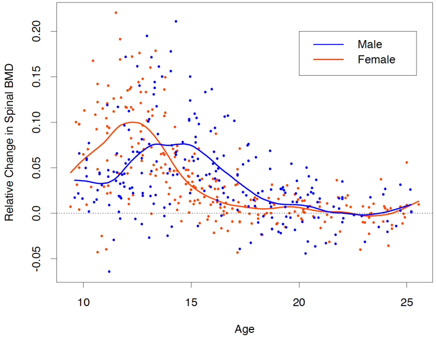
$$\hat{\gamma} = \left(\mathbf{B}^T\mathbf{B} + \lambda\mathbf{\Omega}_B\right)^{-1}\mathbf{B}^T\mathbf{y},$$

  where

$$\mathbf{\Omega}_B = \left[\int B_i''(t)B_j''(t)\,dt\right]$$

- To see this substitute to the PRSS – it becomes a regular least squares problem that is solved by $\hat{\gamma}$.
- Further details in Discussion Sesssion 2.

Example – bone mineral density



The response is the relative change in bone mineral density mea- sured at the spine in adolescents, as a function of age. A separate smoothing spline was fit to the males and females, with $\lambda = 0.00022$. It can be argued that this choice of λ corresponds to about 12 degrees of freedom (the number of parameters in a comparable standard spline fit of the solution). See the textbook for the discussion of transformation from the degrees of freedom to λ and vice versa.