Universidad Carlos III de Madrid
www.uc3m.es

TESIS DOCTORAL

Functional Linear Models

Autor:

Nicola Mingotti

Director/es:

Rosa Lillo and Juan Romo

Statistics Departments / University Carlos III of Madrid

Leganés, July 2015

Universidad
Carlos III de Madrid
www.uc3m.es

TESIS DOCTORAL

Functional Linear Models

Autor: *Nicola Mingotti*

Director/es: **Rosa Lillo and Juan Romo**

Firma del Tribunal Calificador:

Firma

Presidente: Francisco Javier Prieto Fernández

Vocal: Ana Justel Eusebio

Secretario: Eva Senra Díaz

Calificación:

Leganés, de de

Functional Linear Models

Nicola Mingotti

Statistics Department
University Carlos III of Madrid

Advisors: Professor Rosa Lillo and
Professor Juan Romo

July 15, 2015

Acknowledgments

I would like thank first of all my advisors Dr. Rosa Lillo and Dr. Juan Romo. If it was not for Rosa I think I would have left many years ago, when things were going very far from good enough. Thank to them I could see many different real world problems in Statistics and this was very motivating for me, who till 30 years old I saw only theorems, proofs and a good amount of computer code.

I would like to thank my family for all the support they gave me in these long years of study. It has for sure been tough for them to see me leave a good job to start an adventure of which they did not understand the purpose.

I would like to thank my girlfriend Anna for her love, patience and support. If it was not for her I would be still walking from one bar to another, there are too many bars in Madrid.

I would like to thank all of my friends, but especially Pippo, a real friend, somebody you can count on, always. Joanna, who shared with me all the study path in the last six years and a couple of friends who do not live in Madrid any more, but still I remember them with pleasure in long walks and more or less serious conversations, Daniele and Miguel.

Finally I would like to thank all of the many students who worked with me on large projects which I could not have closed alone, especially Melodia.

This thesis has been written with TeXmacs, a software which is almost as powerful as LaTeX, but far more enjoyable to work with. I am grateful to the developers of this program, as well as to all people working on the many GNU tools which are cornerstones of my work.

Abstract

This work aims at the exposition of two different results we have obtained in Functional Data Analysis. The first is a variable selection method in Functional Regression which is an adaptation of the well known Lasso technique. The second is a brand new Random Walk test for Functional Time Series. Being the results afferent to different areas of Functional Data Analysis, as well as of general Statistics, the introduction will be divided in three parts. Firstly we expose the fundamentals of Functional Data Analysis. Then we will recall some variable selection methods in ordinary Linear Regression. Finally we will review some basics of Time Series analysis and briefly review some existing Random Walk tests. These introductory sections will motivate our research putting it in a general framework. Since Functional Data Analysis can be seen as a data reduction method we will talk incidentally of Big Data and we will provide some comments on the current definition of it. All results of our research are supported by extensive computer simulations and in general, all of FDA is based on extensive computer deployment so some attention will be given to software and computation methods. The Lasso has been used in Functional Regression before this work, our contribution is twofold, we provide a reduction of Lasso in Functional Regression from a functional optimization problem to a numerical one via algebraic manipulations, no sampling is required. Then, we augment the Lasso with a *post hoc* analysis method which helps deciding which regressors have to be dropped, we called this augmented strategy *The Shaked Lasso*. About testing if a Functional Autoregressive Process can be considered a Random Walk, our proposed test, as far as we could establish, is the first one in literature.

Abstract in Spanish

En esta tesis se abordan dos problemas relacionados con el análisis de datos funcionales. El primero consiste en selección de variables en un problema de regresión con respuesta funcional adaptando la técnica conocida como Lasso. El segundo problema pretende abrir una línea nueva de investigación proponiendo un test para contrastar si una serie temporal funcional puede ser considerada como un camino aleatorio.

Como los resultados que se muestran en esta tesis están relacionados con áreas diferentes del análisis de datos funcionales y de la estadística en general, la Introducción está dividida en tres partes. En primer lugar, se exponen los fundamentos del análisis de datos funcionales. En la segunda sección se revisan algunos métodos de selección de variables en regresión lineal y por último, se recopilan brevemente las bases de series temporales así como los contrastes de hipótesis que se han utilizado en la literatura para contrastar caminos aleatorios. Estas secciones introductorias ayudan a motivar las aportaciones de la tesis encuadrándolas en su entorno de investigación. Además, ya que el análisis de datos funcionales se puede ver como un método de reducción de la dimensión de los datos, se incluirán algunos comentarios sobre Big Data y sus definiciones.

Todos los resultados de nuestra investigación están soportados por un extenso trabajo de simulación y, puesto que en los métodos estadísticos aplicados a datos funcionales es esencial la parte de computación, se ha prestado especial atención a todos los aspectos relacionados con el software y la modelización. El procedimiento Lasso de selección de variables se ha aplicado anteriormente en la literatura de regresión funcional pero no a los modelos que se analizan en la tesis. Las contribuciones en este aspecto son dos: por una parte se proporciona un método de selección de variables Lasso para un problema de regresión con respuesta funcional convirtiendo un problema de optimización funcional a un problema de optimización numérica vía manipulaciones algebraicas y sin necesidad de remuestreo. Después de ejecutar el problema de optimización, como segunda contribución se propone un análisis de las soluciones para decidir los regresores que deben ser eliminados. Este segundo análisis se ha denominado “The Shaked Lasso” porque se basa en alterar un parámetro del proceso de optimización para observar cómo se “mueven” las soluciones. Respecto al segundo capítulo de contribuciones de la tesis, se propone un contraste de hipótesis para testear si un proceso autoregresivo funcional se puede considerar como un camino aleatorio. Hasta lo que nosotros conocemos en la literatura en este campo, es el primer test de este tipo que se propone en la literatura.

Table of contents

Acknowledgments . . . 3

Abstract . . . 5

Abstract in Spanish . . . 7

1 Introduction . . . 11

1.1 Functional data . . . 11
  1.1.1 Modern data are often Big Data . . . 12
  1.1.2 Modern Big and Dirty Data . . . 13
  1.1.3 Meeting functional data . . . 17
  1.1.4 From data to functions. Smoothing. . . . 18
  1.1.5 The Functional Data Toolbox . . . 19
    1.1.5.1 Sample Statistics . . . 19
    1.1.5.2 Functional PCA . . . 21
  1.1.6 The Multiple Nature of Functional Data . . . 22
  1.1.7 Functional Data and Computers . . . 23
1.2 Linear Regression . . . 24
  1.2.1 Simple Least Squares . . . 26
    1.2.1.1 The Empirical Variable Selection . . . 27
    1.2.1.2 APM variable selection . . . 28
  1.2.2 Classic Multivariate Regression . . . 28
    1.2.2.1 Stepwise regression . . . 30
  1.2.3 Penalized Regression and Lasso . . . 31
  1.2.4 Other methods . . . 34
1.3 Time Series . . . 36
  1.3.1 Stationary and Autoregressive Processes . . . 36
  1.3.2 Random Walk tests, an Overview . . . 41
  1.3.3 Random walks, Applications and Curiosities . . . 43
    1.3.3.1 Etymology of “Random Walk” and the drunken man . . 43
    1.3.3.2 Random Walks in Economics . . . 45
1.4 Structure of the Thesis . . . 47

2 The Shaken Functional Lasso . . . 49

2.1 Introduction . . . 50
2.2 Methodology . . . 54
    2.2.0.3 First Example . . . 56
    2.2.0.4 Second Example . . . 62

2.3 Simulation Study 63
2.3.0.5 Simulations Diagram 64
2.3.0.6 Simulations Analysis 66
2.4 Case Study: Low Speed Car Accidents and Whiplash Injury . . . 67
2.5 Conclusions 73

3 A Random Walk Test for Functional Time Series 75

3.1 Introduction 75
3.2 Test Procedure on a Real Dataset 78
3.3 Simulation Methodology 80
3.3.1 Simulate and Test a FAR(1) process 81
3.4 Simulations Analysis and Results 82
3.4.1 Analysis 84
3.5 Applications 85
3.5.1 Electrical energy consumption in France 85
3.5.2 Bitcoin daily prices 87
3.6 Conclusions 89

4 Conclusions and Future Work 91

Bibliography 95

Appendix A 99

A.1 FAR(1) Simulations 99

Appendix B 107

B.1 Details about the Confidence Band Plot 107
B.2 Shaken Lasso Simulations 108

Chapter 1
Introduction

1.1 Functional data

What is Functional Data? Or citing [Cuevas, 2014] “Do really exist such things as functional data”. This is not a frivolous question because all of data, which are measurements of some kind, come in discrete form as sequences of numbers. On the other side, Mathematics trained us to work with objects whose empirical existence in quite questionable, consider complex numbers, do they exist? Does π exist? In the end, to the scientist it is irrelevant if these things empirically exist in strict sense, completely mirroring their mathematical definition. The important thing is that they work, in the sense that they provide a reasonable model, give insight or extend our degree of understanding and dominance on the world of Nature. To the mathematician, *conditio sine qua non* for the acceptance of a new object, is that it should interact well with other mathematical structures, so the question of existence is irrelevant, if I can define it and it does not rise contradictions it exists. Then, even if Functional Data, as complex numbers, they do not probably exist in a strict practical sense, the central question is, what can we make with them? Are they useful?

The first motivation for Functional Data is that data we are dealing with are often continuous in nature. Consider for example temperatures T at some time t_i in a certain location. It is apparent that $T(t)$ is a continuous function, at least for the range of variations we often consider of interest. Then, if we model temperature with a continuous function, not only we are sticking to the natural continuity of the observable, we are also simplifying the problem because assuming continuity we know that $T(t_i)$ carries information about all the $T(t_j)$ when t_j is near enough to t_i.

A second motivation for Functional Data is economy of thought and manipulation. Consider again the case of temperatures, if you watch at them minute by minute you will need to carry around thousands of values of which only a few are really relevant. If you model them as a function, fit at the desired level of precision, you will be able to move around an algebraic formula which is usually a couple of lines long. Moreover, seeing your data as formulas you can apply all the powerful tools of Calculus: differentiation, integrals, differential equations, which have been the heart of science since the time of Euler.

A clarification is in order. When we talk about Functional Data, we often imply the functions are continuous and possibly smooth enough. That is, they are in the domain of Calculus operations. Indeed, each sequence of $(t_i, T(t_i))$ defines a function[1.1] but, if there is no reasonably compact representation for those couples and perhaps also continuity is lost, then it is hard to imagine how seeing the data set as a function could improve the situation.

There are other data sets which we can not suppose continuous *tout court*, think of stock prices, they can jump, continuity is not really granted, and for sure differentiability is lost. Well, in this cases we know Functional Data does not fit the true nature of data set but, if we can approximate the data well enough with smooth functions, then we are still in business.

In this introduction we will illustrate briefly how to convert a data set into a function and incidentally, we will review some of the difficulties arising when working with modern large datasets. After that, we will review the fundamentals of classic regression theory and time series analysis. Our aim is to arrive at a compact description of the Lasso technique in multivariate regression and Random Walk test in autoregressive processes of order one.

1.1.1 Modern data are often Big Data

The development of Functional Data Analysis has been prompted by the large amount of data made available by modern automatic data collection tools and the revolutionary simplification in information exchange due to the Internet. Let's make an example, suppose we are interested in temperatures and, for example, we want to establish if temperatures are increasing with time, the classic highly debated *Global Warming* issue. Faced with such a problem, a scientist of the past would probably had to take measurements by himself, look at some thermometers, write down results and collect them for a considerable amount of time. Or, if more lucky, he should had to dig into thick and dusty volumes in a library, as Sir Ronald Fisher did in Rothamsted. A scientist of our days instead, can set up a cheap computer to take measurements of temperature every second automatically, in different parts of the World, and receive all the results in his office, for example by email. Then, he may put this data in a public repository on the Web so that other scientists could study them.

Figure 1.1 shows the very different kind of datasets the two scientists could have worked on. Part [a] shows monthly data temperatures in Oxford[1.2], taken in 1853. Each data point was elaborated by a person, probably someone used to manage data, and stored in a paper archive. Part [b] shows hourly data temperature in Oakland[1.3] (California) taken in 2013. Data

[1.1] Supposing t_i are all different.

[1.2] Data available at http://www.metoffice.gov.uk/

was written by a machine, and stored electronically in a way that a far user could access it easily and instantaneously, as we did.

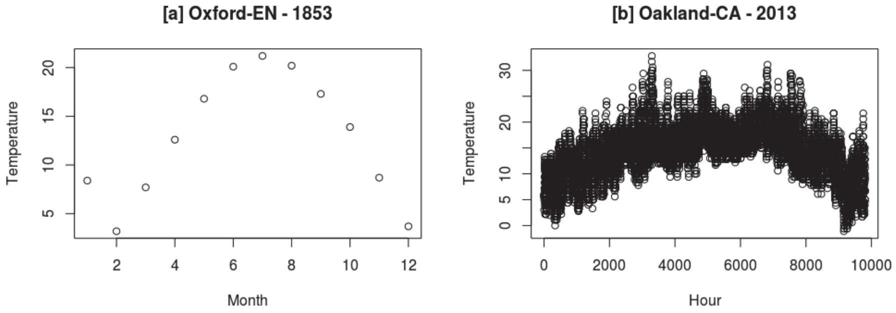


Figure 1.1. Data from the past and the present. Figure [a] shows monthly average temperatures in Oxford (England) in 1853. Figure [b] shows hourly temperatures in each day, for a year, in Oakland (California) in 2013.

Last figure speaks by itself, the amount of data we have in present days is overwhelming. But it is not only that. Observing closely part [b] we can see there is something wired. There are approximately 8760 hours in a year, so how is it possible that there are about ten thousands observations in that plot? The answer will come in next section.

1.1.2 Modern Big and Dirty Data

Even the simple name Big Data is subject of debate in these days, many people is trying to open new business opportunities out of it and that does not simplify the understanding of the real thing going on. According to Chapter 20 of [Topi and Tucker, 2014] Big Data can be characterized by a few items[1.4].

1. [Big Data] It is too large to fit into ordinary [consumer] hardware storage devices.

2. It is too large to be manipulated in reasonable time by generic database software.

3. It is often generated automatically by machines.

4. It has not been designed to be friendly, nor to be used by a specific purpose, often it has not been designed at all.

[1.3] Data available at `http://www.wunderground.com/`

[1.4] I make some modification to the original work phrasing.

5. A large part of it can be near to worthless. Interesting parts need to be dug out.

To be more direct, Big Data is not only about **quantity**, it is about **quality**. This kind of data has not be taken for a specific closed purpose in mind, and they are not filtered by the mind of a trained data professional. They come from automatic machine loggers, video cameras, microphones, Web logs, geo tagging and so on. Their format is the one in which the computer programmers was more comfortable in.

Statistics is the science of drawing significant conclusions from data. Data is supposed to be scarce, but of good quality. In Big Data the amount of information is so large that there is no interest in considering confidence intervals for whatever estimator but, the data can be tremendously biased. Let's consider an example, it is fashionable these days to do sentiment analysis about some brand or product. Well, there are so many tweets around the problem is not to establish if it is higher the proportion of satisfied or unsatisfied customers, that will be direct. The point is, who is talking about your product? Are these tweet representative of the population humor? Or better again, is the twitting people your target future customer? We see no way to answer these question if not to perform a well designed survey with a few but quality data and an attentive analysis. Massive data, is not better in principle than small and quality data, the case of the "*The Literary Digest*" forecasting *Landon* v.s. *Roosevelt* election[1.5] is paradigmatic in this sense. Another case, very recent, is the *Google* failure at forecasting flu[1.6].

Data in Fig.1.1[b] is in some sense an example of Big Data, not because of size, it is only one Megabyte, but we had to dig it from the Web and parse it, probably nobody has checked it till today and it does not correspond to its online description strictly. All temperatures are usually taken at minute 53 of every hour. But, there are some days in which the computer logged also at other minutes. We ignore the reason behind it but that is what we have. In conclusion, this dataset is rich, but scarcely reliable.

Further, in this work the reader will see analyzed datasets that were never studied in literature. Getting this data to work was not a negligible effort. These are the main difficulties we encountered in analyzing real modern datasets which can be present, in all or in part, in most of all Big Data sources. These nuisances are probably familiar to everybody who has worked with data but, if with a small dataset you can spot these problems by sight and correct them immediately by hand, when the dataset is large, you can not. You must instruct a computer to scan for problems and also, eventually, to correct them. For this reason, Big Data analysis requires competence in

[1.5] http://www.math.upenn.edu/~deturck/m170/wk4/lecture/case1.html

[1.6] http://www.forbes.com/sites/stevensalzberg/2014/03/23/why-google-flu-is-a-failure/

Statistics and in Computer Science. Here follows a brief list of difficulties related to the study of many large dataset.

- **Non Systematic**. In figure Fig.1.1.[b] we expected 8760 items but we had more. Data was taken at minute 53 of every hour, but sometimes also in other minutes. This data is non systematic, in the "documentation" it is said to be hourly temperature but sometimes it is not.

- **Non Tabular**. We expect data to come as nice tables, in *.csv* format or, in the industry mostly as Excel files or from a database. Well, this is not always the case, if you are going to study something new expect to fight to have it in a pretty tabular format. Data for *Oakland* temperatures for example was taken from many web pages, it had to be cleaned from residual HTML code and properly joined. There are also more extreme cases, a dataset about car accidents that will be studied in detail in Chapter 2, we did not have any number to crunch, only plots, on thousands of papers, documented in German language, like the ones in the next picture.

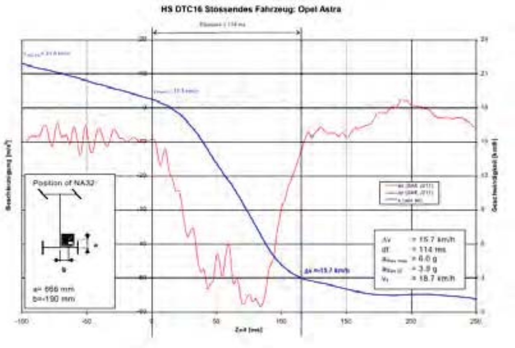


Figure 1.2. Velocity and acceleration plots of a car after a low speed crash with another vehicle.

We had to go back from the plot to a table of (x, y) coordinates, it was a long lengthy job to complete.

- **Inconsistent**. When more then one person work to fill the same dataset, if they are not precisely directed, there will be many inconsistencies. One for all, one employee will decide to encode empty data as "NC", another will write "unknown", and other will leave a white space.

- **Duplicate**. Data often comes with double entries. Data about Oakland temperatures contained only one double entry. Dataset about *Bitcoins* and stock prices, to be seen in Chapter 3, contained hundreds of double lines. If dataset was compiled by hands in general, expect doubles and if by a poorly programmed machine, expect thousands.

- **Typos**. People make mistakes. Sometimes instead of 1000.34 you will find 10034 or 100; 43. Instead of *blue* you will find *bleu*. Many times this mistakes will break your analysis path, you will find them and correct them. But there are cases in which finding the mistake is more difficult. Consider a car weight, the true value to put in would have been 1240 Kg but the operator mistakenly wrote 1420 Kg, it is a simple transposition, it is still a reasonable value, it will not break the analysis path but the error it introduced can be significant.

  If you saw this weight associated to a *Fiat Punto* maybe you could guess there is something wrong. But, in Big Data, you will **never** read all the data, so you will not find this kind of mistakes unless you teach your computer to recognize non plausible lines, which is not a trivial task.

- **Missing Data**. In another dataset about car accidents with 1400 car crashes and 250 variables we faced the condition expressed by the next picture. A little red square represents a missing data. We just ignore variable columns where missing data is more than 80%. What to do with columns where around 20% of measures are missing?

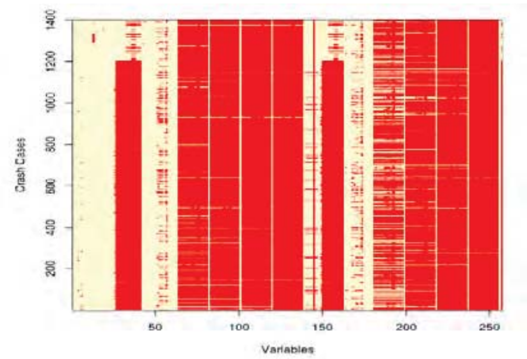


Figure 1.3. This is an illustrative version of a data table, every small red square is a missing data.

- **Internationalization**. This is a source of extreme frustration, characters that are not part of the English alphabet are encoded in different ways across operating systems and often, also across different programs on the same system. I warmly suggest to rewrite all the dataset in ASCII characters before starting the analysis. Remove every accent, every non English letter has to be replaced, write all numbers following the U.S. system that is, use dots to separate decimals, not commas. Finally, change every white character[1.7] in strings with a "-", we don't read white characters, but the computer does.

[1.7] Space, Tab, Newline etc.

- **Badly Documented**. Or not documented at all. This is unfortunately a cancer of our epoch, probably diffused by the modern computer Graphical User Interfaces and the slides culture. You don't have a manual for *Windows* or *OSX,* you just try to click stuff and prey you will get what you want. If you ask for a manual you will get a *Power Point* presentation. If the dataset is small you can reverse engineer most of the data sets, you can build your half invented manual with your personal understanding of all columns. Now imagine a dataset with some hundreds columns, you need a good motivation to guess a manual for that.

1.1.3 Meeting functional data

Functional Data is data that can be seen as functions. The last statement is trivial and almost meaningless. Every set of N measurements x_i taken at times t_i can be seen as a function $\hat{f}$, indeed $\hat{f}$ is defined simply saying that $\hat{f}(t_i) = x_i$ and undefined for every $t \notin \{t_1, ..., t_N\}$. This is not what we are looking for. It will be clear from the following examples that we are looking for a regular function, possibly a smooth function with just a few jumps. Indeed, it must be more profitable to work with $\hat{f}$ than with the sequence $\{(t_i,x_i)\}_i$.

Next figures show how the two dataset of Oxford and Oakland temperature can be seen as functions.

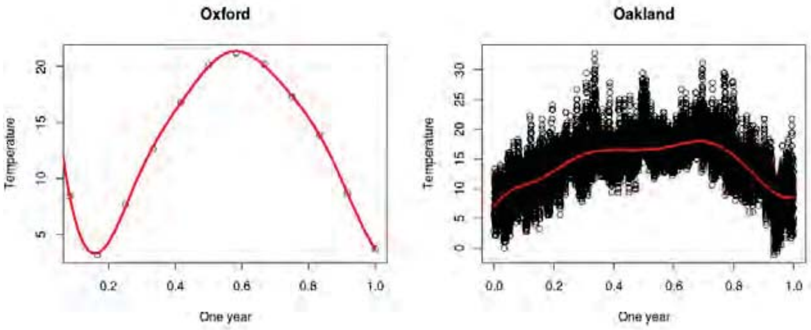


Figure 1.4. Oxford and Oakland yearly temperatures as functions.

It must be noted that the two transposition to function are, in a way, completely different. In the case of *Oxford*, our hypothesis of continuity of temperature function is giving us more data respect to what we really have, we are implicitly **augmenting the dataset**. In the case of *Oakland* instead, the continuity of temperature act as a **data reduction** restriction. Of the two, the most perilous is the first case. Indeed, if we cut information we know what we have removed, if we add data on the other side, all of our analysis is standing on our preliminary hypothesis.

1.1.4 From data to functions. Smoothing.

Data, once is cleaned and well prepared, always come in form of tables. In this section we see how to transform tables into functions.

One of the ways to express a set of points as a function is to choose a *function basis* ϕ_1, ϕ_2, ... and then compute a finite linear combination of $(\phi_i)_i$ such that the approximation to the original points is considered good enough.

More precisely, given a sequence of observations taken at time t_i and written as $y(t_i)$ we have to choose a basis functions $\{\phi_i\}_{i\in I}$, a natural number K and a set of coefficients $\{c_i\}_{i\in\{1,\ldots,K\}}$ such that,

$$\begin{cases} x(t) = \sum_{k=1}^{K} c_k\,\phi_k(t) \\ x(t_i) \approx y_i. \end{cases} \tag{1.1}$$

The most popular basis functions are *Fourier* and *BSplines*. The first is the best choice when data exhibits a periodic behavior. The second is more flexible and is to be preferred when there is not periodicity or when the function becomes more volatile in a certain parts of its domain.

The approximation symbol "$\approx$" is very important. We are not much interested in a perfect fit to the y_i, every y_i can come with a measure error or also, more importantly, our model is not expected to provide a perfect match to any observation since it is impossible to take into account all variables affecting an empirical quantity. Every model is finite and, by its nature, every realistic prediction will come with an error.

How do we say that $x(t)$ is a good approximation? There is not a close absolute way to answer this question but there are a few general principles. The model should be parsimonious, that is it should use as few variables as possible. Then, in case of functional data where the simplifying power comes from regularity, the functions should be as smooth as possible. The eye of the modeler(s) and his knowledge of the nature of the data is the final judge in deciding how much smoothness is desirable, how much is appropriate and how much is an unacceptable oversimplification.

One way to determine the function $x(t)$ is by minimizing the sum of the squared distances between the function and the raw data, this is called **Least Squares Smoothing**.

$$\{\hat{c}_k\}_{k\in 1\ldots K} := \underset{\{c_k\}_{k\in 1\ldots K}}{\text{Argmin}} \sum_{i=1}^{N} (y_i - \sum_{k=1}^{K} c_k\,\phi_k(t_i))^2 \tag{1.2}$$

From the computational point of view this is a simple quadratic optimization problem without constraints and it is known to have a unique solution.

This problem can be also seen as a regression problem, so, once rewritten in an appropriate matricial form, the solution can be found by the classic expression $(X'\,X)^{-1}\,X'\,Y$, even if this writing is more fit to theoretical development than to direct calculations.

The smoothness of our approximating function $x(t)$ in this case is determined only by the type of basis functions we chose and the number K of basis functions.

Sometimes it is desirable to have a finer control over the smoothness of $x(t)$. In these cases, we can augment the previous expression (1.2) with a penalization term $\Pi(x(t))$ and a tuning parameter λ, the resulting technique is called **Roughness Penalty Smoothing**.

$$\begin{cases} x(t) & = \sum_{k=1}^{K} c_k\,\phi_k(t) \\ \hat{c} & = \underset{c\in R^K}{\text{Argmin}}\ \sum_{i=1}^{N}\,(y_i - x(t_i))^2 + \lambda\cdot\Pi(x(t)) \end{cases} \tag{1.3}$$

A common choice for the functional Π is $\Pi(x(t)) := \int_D\,(\ddot{x}(t))^2\,\mathrm{dt}$ where D is x domain. This last choice of Π penalizes the total curliness of $x(t)$, more we increase the value of λ, the more the resulting $x(t)$ will tend to have maxima and minima flattened out.

There are many other possible choices of Π, for example we could be interested in penalizing curliness only in a subset of the domain $S \subset E$. Or, we could set $\Pi(x(t)) := \int_D\,|\ddot{x}(t)|\,d\,t$. Or also penalize with two separate functionals Π_1 and Π_2, we have a lot of freedom here. However, the important point to emphasize is that whatever will be the choice of Π, we have to make sure the resulting optimization problem in (1.3) is solvable, possibly the solution should be unique, and be sure some algorithms have been invented to find it.

1.1.5 The Functional Data Toolbox

1.1.5.1 Sample Statistics

> *„Man muß jederzeit an Stelle von „Punkte, Geraden, Ebenen“ „Tische, Stühle, Bierseidel“ sagen können“*[1.8]

D.Hilbert

Once we transformed data into functions we want to perform our statistical analysis on the functional objects. The two cornerstones of Statistics are for the **mean** and the **variance**. In our new world where data are functions, how do we define them? Given the functions $x_1(t)...x_N(t)$, It turns out that the **sample mean** and the **sample variance** are defined in the usual way

$$\bar{X}(t) = \frac{1}{N}\sum_{i=1}^{N} x_i(t)\,, \tag{1.4}$$

$$\mathrm{Var}_X = \frac{1}{N-1}\sum_{i=1}^{N} (x_i(t) - \bar{x}(t))^2\,. \tag{1.5}$$

What about the **mean** and the **variance**? *Hic sunt dracones*. Whilst the definition the two sample statistics is trivial, the correspondent population statistics are not. The books that most helped in popularizing FDA, [Ramsay and Silverman, 2005] and [Ferraty and Vieu, 2006], do not even mention this problem. We will not deal with theoretical topics in this thesis so we just point the reader toward two books that seems the most appropriate to fill the gap, [Grenander, 1981] and [Ash and Gardner, 1975].

Leaving aside the problem of population statistics we move on defining the [sample] **covariance** as

$$\mathrm{cov}_X(t_1, t_2) = \frac{1}{N-1}\sum_{i=1}^{N} (x_i(t_1) - \bar{x}(t_1)) \cdot (x_i(t_2) - \bar{x}(t_2)). \tag{1.6}$$

We observe explicitly that this quantity corresponds to the **autocovariance** of Time Series analysis. There is a link between FDA and Time Series that will be stressed at the end of the introduction.

The [sample] **cross covariance** between the functional random variables X and Y is

$$\mathrm{cov}_{X,Y}(t_1, t_2) = \frac{1}{N-1}\sum_{i=1}^{N} (x_i(t_1) - \bar{x}(t_1)) \cdot (y_i(t_2) - \bar{y}(t_2)). \tag{1.7}$$

[1.8]. One must be to say “tables, chairs, beer-mugs” each time in place of “points, lines, planes”. An interesting discussion about this quote can be found at http://goo.gl/eBpahm.

From the formulas it is easy to see the intended meaning of the two definitions. The *covariance* between functional data is intended as the classical covariance between two points in time of the same functional random variable. The *cross covariance* instead is the classical covariance between two points in time of two different functional random variables. So, **a functional random variable** is considered as a collection of ideally infinite, but practically finite, univariate random variables, in conclusion, a functional random variable, according to the way we are manipulating it, **is a stochastic process**. A time series is also a stochastic process, but often required to be discrete and **stationary**, we will see more about this in Section 1.3.

1.1.5.2 Functional PCA

Functional Principal Component (**FPCA**) is a cornerstone of Functional Data Analysis as it is in multivariate Statistics. In the multivariate case, there are many equivalent ways to define the principal components of a data matrix X, where $x_{i,j} \in X$ are observations of the same variable in column j. We will use the next one that is totally opaque on the meaning of the operation but is very compact and permits an immediate extension to the Functional context.

PCA

If $V_{p,p}$ is the covariance (or correlation) matrix of the data matrix $X_{n,p}$ then there are at most p different eigenvalues/eigenvectors for V,

$$V\xi_j = \rho_j \xi_j \,.$$

The first q principal components are the first q couples (ρ_i, ξ_i) having the largest ρ_i.

We can now define FPCA from PCA. We suppose the data has been centered, this is done, for example, to simplify the next expression. From a previus definition we get the [empirical] [auto] covariance function is

$$v(s,t) = \frac{1}{N-1} \sum_{i=1}^{N} x_i(s)\, x_i(t)\,. \tag{1.8}$$

And we define the [empirical] covariance operator V as

$$V\xi = \int_D v(s,t)\, \xi(t)\, dt\,. \tag{1.9}$$

The first K [empirical] functional principal components will be the functions $\xi(t)$ such that

$$V\xi = \rho\,\xi\,. \tag{1.10}$$

Since such a relation will be generally satisfied for many ξ, then we say the first principal component is the one with largest ρ, the second the one associated to the second largest ρ, and so on.

A good and far more detailed description of the process is given in [Ramsay and Silverman, 2005, Ch.8]. Two ways to compute the solution of expression in Equation.1.10 are given in the same book at Section 8.4. Principal Components is a deep topic, an interesting book fully dedicated to it is [Jolliffe, 2002]. A recent overview of Functional PCA can be found in [Ferraty and Romain, 2011, ch.8, "Principal Component Analysis for Functional Data", *Peter Hall*].

1.1.6 The Multiple Nature of Functional Data

Functional Data is a recent research field, many things still have not been strictly codified. Suppose you have two datasets $(t_i,\, v(t_i))$ and $(t_i,\, w(t_i))$ which you want to consider as functional, that is, as two functions $v(t)$ and $w(t)$.

Now, there are at least three real representations for your ideal functions $v(t)$ and $w(t)$.

1. The *raw* representation: $(t_i, v(t_i))$, $(t_i, w(t_i))$.
2. the *function* representation: $\sum_{k=1}^{K} c_{v,k}\,\phi_k(t)$, $\sum_{k=1}^{K} c_{w,k}\,\phi_k(t)$.
3. The *Fourier* representation: $(c_{v,1}, \dots c_{v,K})$, $(c_{w,1}, \dots c_{w,K})$.

The choice of one over another is not indifferent. Consider you want to compute the distance between your two data functions. And suppose the distance you have in your mind is the distance L^2, $d(v,\, w) := \int_D\, (v(t) - w(t))^2\, d\,t$. How do you compute it?

The first approach you may think of is to use a Computer Algebra System (CAS), like for example *Mathematica*, *Maple* or *Maxima*, apply directly the representation (2) and then use the definition of L^2 distance. This way is for sure straight and correct but has a drawback, in general it is slow.

The second way is to use representation (3) and compute $\hat{d}(v,\, w) := \sum_{k=1}^{K}\, (c_{v,k}\, -\, c_{w,k})^2$. This is fast, it is easy but one must remind that it is true in general that $\hat{d}(u,\, v) = d(u,\, v)$ only if the basis $\{\phi_i(t)\}_i$ is orthonormal. If the basis is not orthonormal, like BSpline, *ad hoc* corrections to the formula are needed.

Somebody may want to compute the distance as $\hat{d}(u, v) = \sum_{i=1}^{N} (v(t_i) - w(t_i))^2$. That is, computing the distance directly on the observed values coming as data. This is the easiest possible way of doing things but it is not optimal because it sums also the noise coming with the data. The smoothing part is usually taken to transport the data to the world of functions, but also to separate the data from noise.

Finally, if the t_i grids for v and w do not correspond, or the basis $\{\phi_i(t)\}_i$ is not orthonormal, one can use first representation (2) then choose a new sampling grid $T_1 \ldots T_M$, compute two new sequences $(T_i, v(T_i))$, $(T_i, w(T_i))$, and finally say $\hat{d} = \sum_{i=1}^{M} (v(t_i) - w(t_i))^2$. The only trouble with this method is that the choice of new grid $(T_i)_i$ becomes just another nuisance parameter we have to guard against. In some way we should be confident that our results do not depend much on our arbitrary choices of analysis. A first choice here is the appropriate basis functions and the second one is the grid $(T_i)_i$.

1.1.7 Functional Data and Computers

Even if the theory of Statistics on the space of functions can be carried out with only pencil and paper, the applied part requires so many computations that the use of a computer is necessary. This is the main reason why it flourished only in the last decades.

There are two ways to compute with Functional Data, with a Computer Algebra System (CAS), like *Mathematica*, *Maple*, *Maxima* that permits to manipulate directly functional objects, as represented in point (2) of section 1.1.6. Or, we can use representation (1) and (3) and this can be done with basically every general purpose programming language like *R*, *Matlab* or *C*. In this thesis functions have beeen programmed in both ways, with *Mathematica* and *R*.

Using Mathematica it is easy to see what is going on, and there is a lot of freedom because each problem can be reduced quite easily to an optimization problem in its analytical form, the program solves it without asking too many hints and sometimes[1.9] solves it correctly. There are two drawbacks, everything has to be coded from the ground up so, for example, if you want to use a basis function then you have to know something about that basis function, not just its name. Secondly, computations are slow because they are made on abstract objects represented as trees.

[1.9] Unfortunately *Mathematica* (release 10) *Minimize* command, is very limited in power. It is often necessary to transport the optimization problem to Maple or Matlab to get a reasonable solution.

Using *R* there is a completely different perspective. The community of statisticians developed two packages to deal with functional data, *"fda"* documented in [Ramsay et al., 2009] and *"fda.usc"*, documented in [Febrero-Bande and Oviedo de la Fuente, 2012], I worked only with "fda" package. Many common techniques like smoothing, plotting, FPCA, simple regression etc. are implemented and ready to use as procedures. The user has little to do besides feed in the correct arguments. But, all the computation are opaque to the user and considering the current status of the documentation[1.10] the only way to understand what your computer is doing is to jump between the books [Ramsay et al., 2009] and [Ramsay and Silverman, 2005], read the *R fda help* and also read the code examples ..., a painful experience. Until what you want to do is implemented in a library and you can find an understandable example to copy, you will be very satisfied with *R*, if what you want to do is new, or unusual, you are going to have some bad days with R, the language is baroque, good for interactive computations but awkward for programming. There lacks a decent *hash table* data structure, object orientation is not available through "dot notation", nested lists are not natural, matrix operations need to be escaped and finally, the central data structure is the *Data Frame* which is good for representing a data table but inadequate for basically everything else (see [Ihaka and Lang, 2008] and [Ihaka, 2010] for further considerations).

1.2 Linear Regression

"All models are wrong but some are useful"

George Box

Regression is a method to find the relation between an output variable Y and some input variables X_i that is, to find a function f such that $Y = f(X_1, ..., X_p)$. Often the output variable is called *response variable* and the input variables can be named *regressors*, *predictor variables* or *features*, we will use these terms indifferently. When there is more than one predictor the problem is called more specifically *multivariate regression*. The variables Y and X_i can be qualitative or quantitative but in our discussion we will see only the case in which all the variables are quantitative and real numbers. The purpose of our introduction is to motivate a variable selection method, that is a way to select which X_i are most influential in the relation f and which ones can be discarded. Many books can be of help in filling the necessary large gaps we left in this introduction. In particular, [Wasserman, 2003] for a short overview, [Chatterjee and Hadi, 2006] and [Weisberg, 1985] for an

[1.10] Few complex mathematical software are documented in detail, the only exception known to me is Maple which has an excellent manual. Just give it a look here http://www.maplesoft.com/support/help/.

in depth treatment of regression in practice, for a theoretical point of view (there is no one single real data set) see [Seber and Lee, 2003] and finally, for a machine learning point of view [James et al., 2013] which contains R code for examples or the more advanced and deep [Hastie et al., 2009].

Our variables Y and X_i in practice always come as numerical tables and it is convenient to represent them in matricial form as

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ ... \\ y_n \end{pmatrix}, \qquad X = \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & ... & x_{1,p} \\ 1 & x_{2,1} & x_{2,2} & ... & x_{2,p} \\ ... & ... & ... & ... & ... \\ 1 & x_{n,1} & x_{n,2} & ... & x_{n,p} \end{pmatrix}, \tag{1.11}$$

where n is the number of observations and p is the number of regressors. The ones column is a handy way to insert also the intercept parameter β_0 in our computations.

The relation f between X_i and Y can be in principle any kind of function but we will focus only on one special family, the *linear functions*. Even if restricting the relation to linearity may seem a strong restriction, it has some important benefits.

1. **Analytically Easy**. Linear relations are the easiest to manipulate symbolically and they are very well understood theoretically.

2. **Understandable**. Proportionality is our way to express qualitatively all relations which are not cyclical. “More I eat, more I get fat”, “more I study, more I will learn’, “more a country is rich, more the mortality rate will be low”. All these statements are false in general, but locally, on small domains, these relations hold. Linearity is the first and simplest kind of relation we are able to perceive, perhaps, besides easiness, it is because often we can experience phenomena only on a small scale of variation.

3. **Computationally Efficient**. Linear computations can be reduced to matrix algebra which is extremely efficient from the computational point of view.

4. **Locally Universal**. Even if the real function f describing a phenomenon is not linear, if it is smooth then it is locally linear. So, in a first local approximation a linear relation is still valuable.

5. **Effective**. The linear models have proven to be at the heart of each discipline, from Physics, to Biology to Finance.

Assuming linearity[1.11], the relation$f$ can be written as

$$Y_i = \beta_0 + \beta_1 X_{i,1} + ... + \beta_p X_{i,p} \ , \qquad \text{for } i \text{ in } 1, ..., n. \tag{1.12}$$

The main task in linear regression is to find an estimation of β_0, β_1, ..., β_p such that the relation in (1.12) be fulfilled in the best way possible. We will denote such estimates es $\hat{\beta}_0$, $\hat{\beta}_1$, ..., $\hat{\beta}_p$ or with some other kind of hats over the betas if comparing different estimators. We will see in the next sections the most common ways to find the $\hat{\beta}_i$.

1.2.1 Simple Least Squares

This method is quite old and still the core of many modern methods. It was developed by the needs of astronomy and geodesy in the eighteeth century when open crucial problem where, for example, to determine the path to follow when sailing in open sea, and the orbits of planets. The first clean exposition of the method is contended between Legendre and Gauss. In 1801 Gauss was able to predict the position of the asteroid Ceres after it had disappeared from sight following his orbit around the Sun[1.12]. Stated as an optimization problem the method is written as

$$\hat{\beta}_0 \cdots \hat{\beta}_p := \underset{\beta_0 \cdots \beta_p}{\text{Argmin}} \left(\sum_{i=1}^{N} (y_i - (\beta_0 + \beta_1 x_{i,1} + \cdots + \beta_p x_{i,p}) \right)^2 \tag{1.13}$$

so, the quantities $\hat{\beta}_i$ are defined in such a way that the sum of squared errors in approximating Y with $X\beta$ is as small as possible. The minimized quantity is called **RSS** (Residual Sum of Squares). It must be observed that we can find the minimum of it with standard multivariable calculus ideas see [Adams, 2003, pg.255], this explains its early origin. It is also straightforward to prove that $\hat{\beta}_i$ can be computed with a direct matrix multiplication; in matrix form RSS is $\text{Argmin}_\beta \, \|Y - X\beta\|^2$. We will not enter into details but the following results are of remarkable importance.

1. The solution of the optimization problem can be computed as

$$\hat{\beta} = (X'X)^{-1} X'Y \,. \tag{1.14}$$

2. Saying $\hat{Y} := X\hat{\beta}$, the least square approximation of Y, we can state that $\hat{Y}$ is the projection of Y on the linear space generated by the columns of X, that is the projection on the regressors world.

[1.11] Linearity is more general than the single expression I'm giving here, see for example [Seber and Lee, 2003, pg.4-5].

[1.12] See the interesting material at this Web page, http://www.math.rutgers.edu/~cherlin/History/Papers1999/weiss.html

3. A measure of fit of our linear model to the data can be obtained by the *coefficient of determination* (R^2) which is defined simply as $R^2 := \mathrm{Cor}\big(Y, \hat{Y}\big)^2$ and has a clear geometric interpretation. Moreover R^2 can be written as

$$R^2 = 1 - \frac{\sum_i \big(Y_i - \hat{Y}_i\big)^2}{\sum_i \,(Y_i - \bar{Y})^2} = 1 - \frac{\mathrm{RSS}}{\sum_i \,(Y_i - \bar{Y})^2} \;, \tag{1.15}$$

   which can be interpreted in another important way, we can see R^2 as a function of variability of the "predictions" respect to the variability of the data to predict, see [Seber and Lee, 2003, pg.111] or [Chatterjee and Hadi, 2006, pg.61].

4. R^2 increases always with the number of variables so it is not a good candidate to compare models with different numbers of regressors. In the practice of multivariate regression, the *adjusted*-R^2 is the most popular substitute for R^2; we will see it in the next section because it depends on additional hypotheses. Another possibility is to replace the R^2 with the estimated residual variance S^2

$$S^2 := \frac{\mathrm{RSS}}{n - p_*} \;,$$

   which though is not as practical as R^2 because it is unbounded from the top. In the last equation, p_* is used instead of p to remind that also the constant has to be counted so, if we have p regressors, as in our design matrix (1.11) then $p_* = p + 1$, see [Seber and Lee, 2003, pg.400].

How to do variable selection in this scenario? There are at least two ways we can consider but of course variants of both are possible.

1.2.1.1 The Empirical Variable Selection

Method.1 - Empirical

1. Standardize column Y and all regressors X_i.

2. Perform a Least Squares and find the estimated values $\hat{\beta}_i$. Also, take note of the fit of your model when there are all variable by S^2, cross validation or others methods.

3. From all $\hat{\beta}_i$, with $i \geqslant 1$, consider equal to zero the ones that are considerably smaller than the others. Drop from the matrix X the column corresponding to the small valued beta-hats.

4. Perform another Least Squares on Y and the new X and compare the fit of the new model with the one obtained at step (2).

This method is simply a truncation, as we truncate decimal digits in a number because we consider them too small to be interesting, here we truncate variables which gave small contribute, in magnitude, to the explanation of Y. It is for sure appealing to scientists because they know the in-field meaning and value of the variables they are cutting down. Since we are comparing magnitudes of objects potentially on very different scales the standardization is fundamental.

1.2.1.2 APM variable selection

Method.2 - APM

1. Having available p data columns, the number of all possible models we can build by inclusion/exclusion of some regressors are

$$\sum_{i=0}^{p} \binom{p}{i} = 2^p .$$

2. For each of the 2^p models we compute a measure of fit by S^2, cross-validation or other methods and select the model providing the best fit.

APM means *All Possible Models*. It is a simple solution but very impractical, with only 20 variables we should check approximately one million models, see [Seber and Lee, 2003, pg.392].

1.2.2 Classic Multivariate Regression

If we augment our hypothesis about our model, we can strengthen a lot our methodology for model fitting and variable selection. Indeed, basically all classical regression analysis is not based on (1.12) but on

$$Y_i = \beta_0 + \beta_1 X_{i,1} + \ldots + \beta_p X_{i,p} + \varepsilon_i \tag{1.16}$$

where ε_i are usually assumed to be independent identically distributed random variables, with $\varepsilon_i \sim N(0, \sigma^2)$.

Requiring ε_i to be i.i.d. $E(\varepsilon_i) = 0$ and *homoskedastic* (with equal variance $\mathrm{Var}(\varepsilon_i) = \sigma^2$), we can draw this results:

1. $\hat{\beta}$ is an unbiased estimator for β, $E\big(\hat{\beta}\big) = \beta$ and its variance is $\mathrm{Var}\big(\hat{\beta}\big) = \sigma^2 (X'X)^{-1}$.

2. Between all unbiased estimators of β which are a linear function of the data, the least square estimator $\hat{\beta}$ is the one with minimum variance. $\hat{\beta}$ is said to be the BLUE estimator for β (Best Linear Unbiased Estimator), see [Seber and Lee, 2003, pg.42].

3. An unbiased estimator for σ is $\hat{\sigma} := \frac{\text{RSS}}{n-p-1}$.

If we add the hypothesis of errors **normality** $\varepsilon_i \sim N(0, \sigma^2)$, we can get more results which are important for hypothesis testing. In the following, we will use this notation for compactness $C := (X'X)^{-1}$ and $c_{\text{ij}} := C_{i,j}$.

4. The distribution of the vector $\hat{\beta}$ is multivariate normal, $\hat{\beta} \sim N(\beta, \sigma^2 C)$. The marginals $\hat{\beta}_i$ are Normal, $\hat{\beta}_i \sim N(\beta_i, \sigma^2 c_{\text{ii}})$.

5. We know the distribution of the variance estimator, $\hat{\sigma}^2 \sim \chi^2_{n-p-1}$.

6. $\hat{\beta}$ and $\hat{\sigma}^2$ are independent random variables.

7. If it is true that $E(\hat{\beta}_j) = \beta_j$ the random variable U defined as

$$U := \frac{\hat{\beta}_j - \beta_j}{s.e.(\hat{\beta}_j)}$$

is distributed as a Student t distribution, $U \sim t_{n-p-1}$.

8. If it is true that there is a linear relation between the parameters β_j expressed in matrix notation as $\text{AX} = \beta$ where $A \in M_{q,p}$ then the random variable W defined as

$$W := \frac{(\text{RSS}_H - \text{RSS})/q}{\text{RSS}/(n-p)}$$

is distributed as F, $W \sim F_{q,n-p}$. RSS_H is simply the RSS computed under the null hypothesis H: $\text{AX} = c$ which is easily achieved computing $\dot{\hat{\beta}}$ as

$$\dot{\hat{\beta}} := \begin{cases} \underset{\beta}{\text{Argmin}} & \|Y - X\beta\|^2 \\ s.t. & \text{AX} = c \end{cases}$$

and then $\mathrm{RSS}_H := \left\| Y - X\dot{\hat{\beta}} \right\|^2$. A matrix version of this computation con be found in [Seber and Lee, 2003, sec.4.3].

9. It can now be defined the most popular measure of fit in multivariate liner regression, the *adjusted-R^2*

$$R^2_{\text{adj}} := 1 - (1 - R^2)\frac{n}{n - p - 1} \tag{1.17}$$

which takes into account that R^2 always increases adding regressors and rescales it appropriately, see [Seber and Lee, 2003, sec.4.4].

10. We have available now also other measures of fit for our model which take into account the increasing number of regressors.

    a) **Mallows** C_p.

    b) **AIC**. Akaike Information Criterion.

    c) **BIC**. Bayesian Information Criterion.

    They all depend on the likelihood or on $\hat{\sigma}^2$. They will not be used in what follows; their definition, in a few words, can be found in [Wasserman, 2003, sec.13.6].

Now that we have set by hypothesis the distribution of errors ε_i and assumed that they are Normal with zero mean and fixed variance our toolbox is much better. Indeed a *t-test*, from point (7), permits us to test directly if one single β_j is zero. The *F-test*, from point (8), permits us to test if a whole block of $\{\beta_j\}_{j \in J}$ is zero without doing a multiple testing on each single β_j. Then, results at points (10) and (9) let us compare the fit of models with different number of variables without using *cross validation*.

1.2.2.1 Stepwise regression

Method.3 - Forward Stepwise Regression

1. We start with a model with only β_0.
2. We add to the model one single regressor, the one which increases most the scores respect to R^2_{adj} (or other fit criterion as BIC, AIC etc.).
3. We repeat step (2) until adding a new regressor does not improve the score. At that point we have selected one possible best regressors set according to the chosen fit (score) criterion.

Method.4 - Backward Stepwise Regression

1. We start with the full model, all regressors are included.
2. We remove one single regressor, the one which improves most the R^2_{adj} (or BIC, AIC, etc.).
3. We repeat step (2), removing each time one regressor until removing every possible regressor does not improve R^2_{adj}. At that point, we have selected one possible best regressors set according to the chosen fit criterion.

We must observe that each stepwise regression is a big improvement in computation cost respect to APM; indeed, in the worst scenario, the algorithm requires $p!$ simple steps where APM required always 2^p simple steps. On the other side though, stepwise regression offers a *greedy solution*, which steps through local optimal models but does not guarantees to arrive at the best possible regressors set.

1.2.3 Penalized Regression and Lasso

Lasso as defined in [Tibshirani, 1996], also known as *basis pursuit* in signal processing [Chen et al., 1998], is one of the ways to do penalized regression. It is the method we will extend to functional regression and it developed from *Ridge Regression*, which will be described briefly in the next section. The idea is to penalize the β_j magnitudes in the optimization phase in such a way that some of them will be shrunken automatically to zero. The estimators are defined by

$$\tilde{\boldsymbol{\beta}} := \underset{\boldsymbol{\beta}}{\text{ArgMin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|. \tag{1.18}$$

Or equivalently, this can be seen as a *constraint optimization problem* which, by Lagrange multipliers method, can be written as

$$\tilde{\boldsymbol{\beta}} := \begin{cases} \underset{\boldsymbol{\beta}}{\text{Argmin}} & \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j} \right)^2 \\ s.t. & \sum_{j=1}^{p} |\beta_j| \leqslant \mu. \end{cases} \tag{1.19}$$

It is clear that, if $\mu \to +\infty$ (or $\lambda \to 0^+$) then Lasso estimated $\tilde{\boldsymbol{\beta}}$ corresponds to Least Squares $\hat{\boldsymbol{\beta}}$. When μ decreases under a certain threshold value that depends on the problem there will be not enough freedom for β_i to obtain the values L.S. would assign them; when this happens Lasso will sacrifice those β_j which are less influential on RSS.

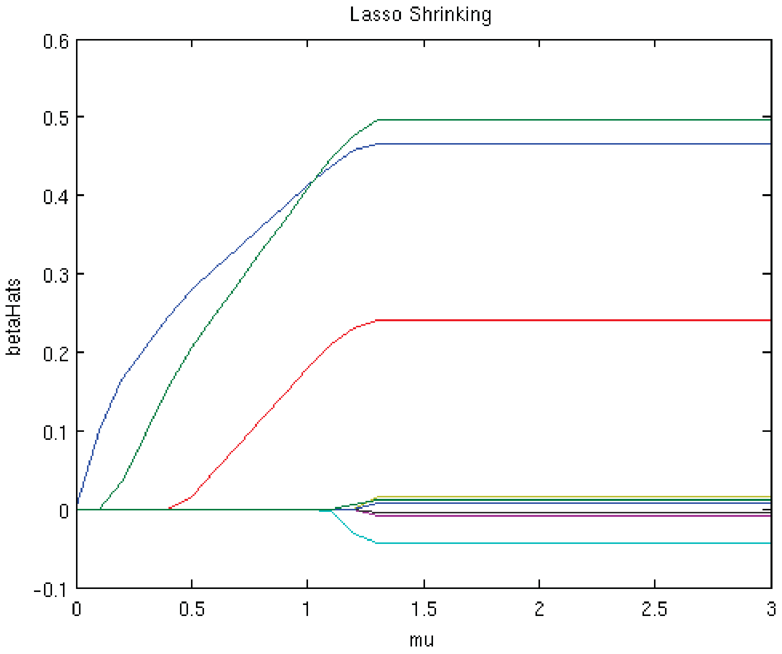
Since the method penalizes the magnitude of all β_j together, they need to be comparable, on the same scale, so it is necessary to standardize all regressors before starting the optimization process. To have a consistent view of the performance of Lasso across different problems we suggest to standardize also the response variable Y. In order to make more clear the usefulness of standardization let's consider an example. Suppose your regressors are *GDP* (Gross Domestic Product) and the *Gini coefficient* of some countries and the response variable is *Male expected life at birth*. The order of magnitude of *GDP* can be around 10^{15}, *Gini* about 10^1 and the *expected life* about 10^1. Then, to have a reasonable linear influence on *life expectancy* the β_{GDP} will be a tiny number and the β_{Gini} will be on the order of 10^0. Consequently, in choosing μ, its smallest variation will influence β_{GDP} but it will be irrelevant to β_{Gini} since it is usually defined with two decimal digits. If you standardize also Y, then you are sure that across all problems $\beta_i \approx 0.01$ is a small number and $\beta_i \approx 1$ is a large one.

The choice of μ is done by *cross validation*, once the value $\tilde{\mu}$ giving the minimum *prediction error* is found, thus the Lasso estimators are completely defined by

$$\hat{\boldsymbol{\beta}}_{\text{lasso}} := \begin{cases} \underset{\boldsymbol{\beta}}{\text{ArgMin}} & \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j}\right)^2 \\ s.t. & \sum_{j=1}^{p} |\beta_j| \leqslant \tilde{\mu}. \end{cases}$$

It must be observed that Lasso provides a quasi-continuous shrinkage of the parameters, the amount of continuity is determined by number of parameters μ_i we try in *cross validation*, a (positive) side effect of this shrinkage is that some parameters go to zero, the special geometry of the l^1 constraint makes, in some sense, zero an attractor for small coefficients. In *forward/backward stepwise selection*, for example, this is not true, a variable is either *in* or *out* of the model. Another important difference between Lasso and stepwise selection is that computation cost is not so much strongly growing with the number of regressors p. On the other side, the choice of the grid values for μ in order to find a minimum in the cross validation error can be problematic and the computational cost grows linearly with the number of grid points for μ.

Figure 1.5 shows how parameters are shrunken by *Lasso* and *Ridge Regression* on decreasing the value of μ. Lasso corresponds to the figure on the top. Starting from the right side of the plot the value of μ is very large so the estimated parameters $\hat{\beta}_i$ are not affected by it and they keep their regular Least Square values. But when μ goes small and they can't keep their original value a choice must be made on how to shrink. We see that *Ridge* compress the parameters but can't put them straight to zero. *Lasso* instead tends to go directly to zero.



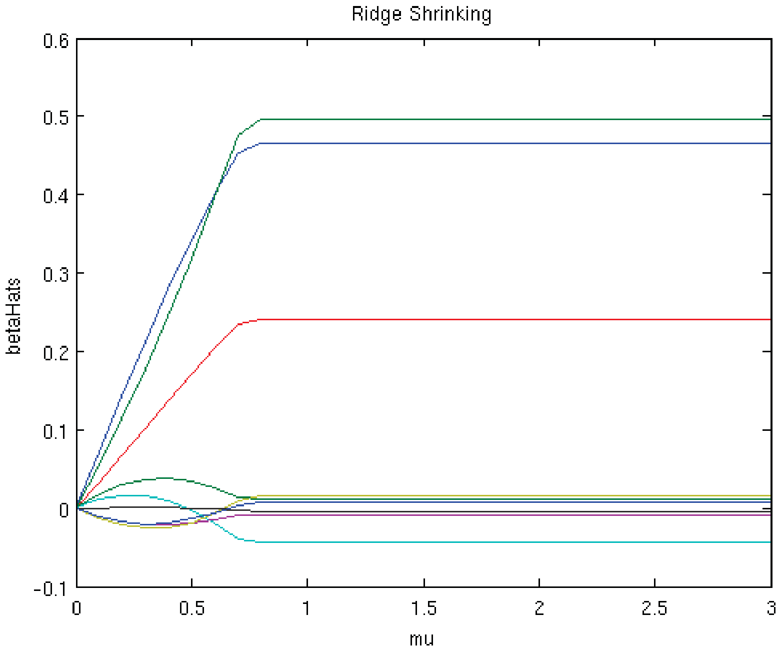


Figure 1.5. The figure on the top represents the parameter shrinkage performed with Lasso, the one on the bottom the shrinkage performed with Ridge. The only difference is that in the first plot the constraint is $\sum |\beta_i| \leqslant \mu$, in the second $\sqrt{\sum \beta_i^2} \leqslant \mu$.

Why does Lasso work? Figure 1.6 gives an explanation. It is a representation of a constraint regression with two regressors where one is fake. The green area is the constraint, on the left there is Lasso, l^1 constraint $|\beta_1| + |\beta_2| \leqslant \mu$ and on the right *Ridge*, l^2 constraint $\beta_1^2 + \beta_2^2 \leqslant \mu^2$. The red ellipses are level curves of the RSS, $\sum_i (y_i - \beta_0 - \beta_1 x_{i,1} - \beta_2 x_{i,2})^2$. The constrained minimization has a solution when the red curves touch the green area. It is clear that with an l^1 constraint there is a good probability that the intersection will happen on a corner of the green area, but a corner is exactly a point in which one of the betas is zero. In higher dimensions this is more difficult to visualize but, under some regularity assumptions, it has been found that Lasso finds the correct model with high probability, see [Hastie et al., 2009, sec.3.8.5] for a brief discussion and a list of references.

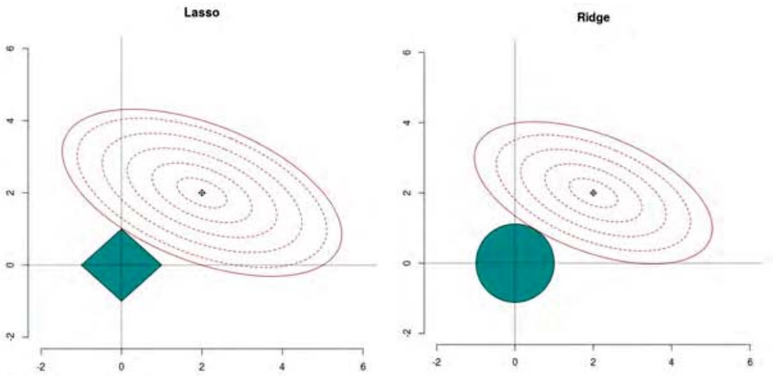


Figure 1.6. On the left a pictorial representation of Lasso constraint and solution, on the right Ridge.

1.2.4 Other methods

In many situations the regressors are highly correlated, so it makes sense to transform them in a convenient way, *PCR*[1.13] and *PLS*[1.14] follow this path. It must be reminded though that in this way we are not any more selecting on the original variables set.

[1.13] Principal Component Regression.

[1.14] Partial Last Squares.

PCR. *Principal Component Regression*. It is a simple Least Squares regression applied after a Principal Component. First, we standardize the regressors columns X_i, then we apply a principal component analysis and find a set of new (columns) regressors $\{Z_i\}_{i=1\cdots M}$, with $M \leqslant p$. Finally we apply regression on the Z_i. It is apparent that least squares can only select at most M variables. If $M = p$, PCR reduces to Least Squares.

PLS. [Wold, 1975] *Partial Least Squares*. It is a method more involved than PCR but still based on iterative linear transformations and orthogonalizations of the original columns-regressors X_i. The main difference respect to PCR is that also the response variable Y enters into the transformation process. As in PCR there is a maximum number of directions M to choose, if $M = p$, that is if not choice is made and same number of variable as is the original dataset is kept, then PLS solution is equivalent to Least Squares.

LAR. [Efron et al., 2004] *Least Angle Regression*. It is a method which gives results similar to Lasso but its computation is totally different, iterative and similar to stepwise regression, a brief introduction can be found in [Hastie et al., 2009, pg.73].

Ridge Regression. [Hoerl and Kennard, 1970] This is a penalization method which precedes Lasso, the only difference is that the penalization on β_i is in norm L^2 instead of L^1. Its associated optimization problem is

$$\hat{\beta}_j^{\text{ridge}} := \operatorname*{ArgMin}_{\boldsymbol{\beta}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \,. \tag{1.20}$$

Respect to Lasso the advantage of Ridge is that estimators are a linear functions of Y and can be written as

$$\hat{\beta}^{\text{ridge}} = (X'X + \lambda I)^{-1} X'Y \,. \tag{1.21}$$

The disadvantage of the norm L^2 is that small $\hat{\beta}_j^{\text{ridge}}$ do not tend to go to zero as directly as $\hat{\beta}_j^{\text{lasso}}$ do.

Elastic Networks. [Zou and Hastie, 2005] It is a combination of Ridge and Lasso, its nature is apparent from the penalization term in which the parameter α permits to tune L^1 and L^2 penalty

$$\hat{\beta}^{\text{elast}} := \operatorname*{ArgMin}_{\boldsymbol{\beta}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^{p} \left(\alpha |\beta| + (1-\alpha) \beta_j^2 \right) . \tag{1.22}$$

The Dantzig Selector. [Candes and Tao, 2007]. Instead of minimizing the RSS, it minimizes the maximum absolute value of its gradient, the penalization term is equal to Lasso

$$\hat{\beta}^{\text{dan}} := \begin{cases} \underset{\boldsymbol{\beta}}{\text{ArgMin}} & \|X'(Y - X\boldsymbol{\beta})\|_\infty \\ s.t. & \|\boldsymbol{\beta}\|_1 \leqslant t. \end{cases} \tag{1.23}$$

The good thing about this method is that it is a *linear programming* optimization problem, for that it was named in memory of *George Dantzig*. Unfortunately this method performs equally or worst than Lasso in predictions and gives extremely erratic betas on changing the values of constraint term t, see [Meinshausen et al., 2007].

After this review of the methods and problems related to variable selection in multivariate regression we change completely topic in the next section and give a short reminder of the basics of time series analysis.

1.3 Time Series

Our aim in this section is to give a minimal overview of linear processes, especially the AR(1) process and *Random Walks*. This material will make clear what the problem is and why it was interesting to translate it to the Functional Data context.

1.3.1 Stationary and Autoregressive Processes

A time series is a stochastic process $\{x_t\}$[1.15] where t is thought as a moment in time and, for the classical theory as developed exposed in [Box and Jenkins, 1970] or [Fuller, 1976], t is an integer number. Time series has its own terminology, we must introduce some amount of it to talk even of the most basic. In what follow I will adapt the material from [Shumway et al., 2000].

Definition 1.1. *The* ***autocovariance function*** *is defined as covariance at two temporal points.*

$$\gamma(s,t) = \text{cov}(x_s, x_t) = E\left((x_s - \mu_s)\,(x_t - \mu_t)\right). \tag{1.24}$$

[1.15] Here $\{x_t\}$ is a family of random variables indexed by t, it could be written as $\{X_t\}$ to emphasize its stochastic nature but the first way of writing is more popular in time series literature.

Definition 1.2. *The* ***autocorrelation function*** *is defined as*

$$\rho(s,t) = \frac{\gamma(s,t)}{\sqrt{\gamma(s,s)\,\gamma(t,t)}}\,. \tag{1.25}$$

Now let's take into account a fundamental point of time series. Often, we observe a time series only once, there are no repetitions. Think of the GDP of a country or of the inflation index for a currency, there are no repetitions, they will happen only once in history. Therefore, we should ask ourselves what sense to give to the just defined autocovariance structure. If we can observe x_s and x_t only once, simply, the autocovariance function makes no sense at all.

So, to work around this challenging condition we make a powerful assumption, we suppose that the series of x_t is extremely redundant, in particular, if we observe a block $(x_{t_1}$, ..., $x_{t_k})$ then we move to the future (or to the past) and observe $(x_{t_1+Q}$, ..., $x_{t_k+Q})$ these two blocks should be ruled by same distribution. This assumption will be called **stationarity** and it is the central idea in the classic theory of time series analysis.

Definition 1.3. *A time series is* ***strictly stationary*** *if*

$$P\left(x_{t_1} \leq c_1, ..., x_{t_k} \leq c_k\right) = P\left(x_{t_1+h} \leq c_1, ..., x_{t_k+h} \leq c_k\right) \tag{1.26}$$

for all time shifts $h \in \mathbb{Z}$.

Strict stationarity is a great simplification but still, too difficult to check to be useful in practice. So the theory developed in a more straightforward direction, in some sense, limiting the ideas appearing in strict stationarity to the the first two moments, expectation and variance.

Definition 1.4. *A stochastic process with finite variance is a* ***weakly stationary*** *time series if*

1. *The mean* μ_t *is finite and constant,* $\mu_{t_1} = \mu_{t_2}$ *for all choices of* t_1, t_2.

2. *The autocovariance* $\gamma(s,\ t)$ *depends on* s *and* t *only through their difference* $|t - s|$. *Or equivalently, if there exists a function* g *such that* $\gamma(s,t) = g(|t-s|)$.

Result 1.5. *In a weakly stationary process the* $\operatorname{Var}(x_t)$ *is constant in time.*

Definition 1.6. *A stochastic process x_t is said to be a **Gaussian process** if, for every n and for every collection of points $\{t_1, t_2, \dots t_n\}$ the random vector $\boldsymbol{x} = (x_{t_1}, \dots, x_{t_n})$ follows a Multivariate Normal distribution.*

Result 1.7. *If a Gaussian process is weakly stationary then it is also strictly stationary.*

Definition 1.8. *From now until the end of the section we will call a weakly stationary process as **stationary** process.*

Definition 1.9. *A process $\{x_t\}$ is called **white noise** if*

1. *x_t and x_s are uncorrelated, for all t and s, $t \neq s$.*

2. *$E(x_t) = 0$ for all t.*

3. *$\mathrm{Var}(x_t) = \sigma^2$ for all t.*

We will denote the white noise process with $\{w_t\}$.

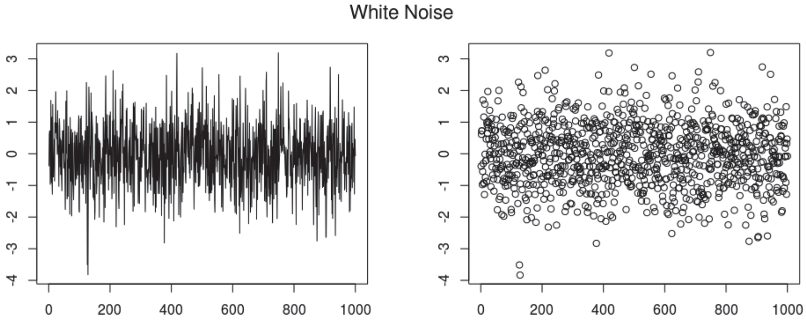


Figure 1.7. Representation of a white noise process. On the right only points (t, w_t) are drawn. On the left, each point (t, w_t) is joined to is successor $(t + 1, w_{t+1})$ by a straight line. This is the cheapest way to build a continuous process from discrete observations.

Result 1.10. *A white noise process $\{w_t\}$ is a stationary process.*

Definition 1.11. *A process is called a **random walk** if*

$$x_{n+1} = x_n + w_n \tag{1.27}$$

If we set $x_0 = p$ *and* $\mathrm{Var}(w_t) = \sigma^2$ *then* $E(x_n) = p$ *for all* n , $\mathrm{Var}(x_n) = n\,\sigma^2$ *and* $\gamma(s,t) = \min(s,t) \cdot \sigma^2$*. So a random walk is not stationary.*

Result 1.12. *The random walk process is not stationary.*

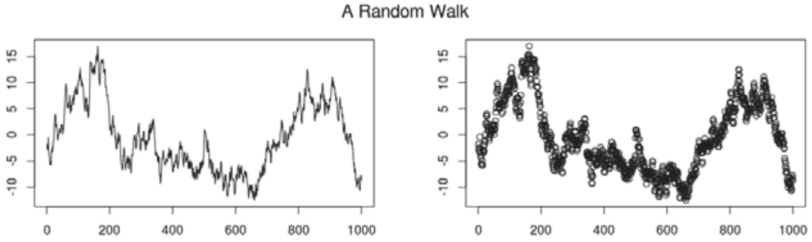


Figure 1.8. On the right a realization of a Random Walk process. On the left the process is made continuous joining every successive points with a line, see comments on Figure 1.7.

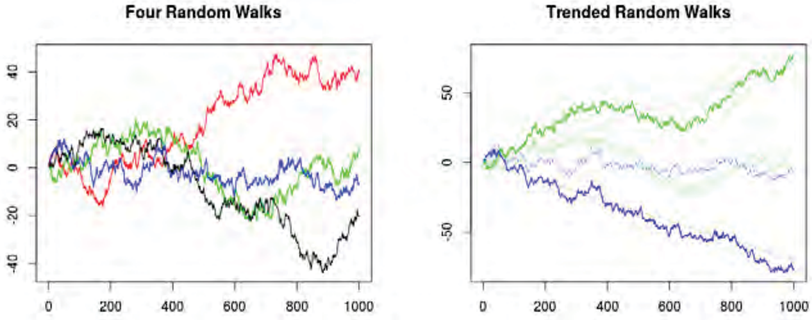


Figure 1.9. On the left, 4 realizations of the same Random Walk process where $w_i \sim N(0,\ 1)$ for $i = 1...1000$. On the right, two Random Walk realizations as in the left figure and their trended associate: $x_{i+1} = k * i + x_i + w_i$ where $k = \{-0.07, 0.07\}$.

Definition 1.13. *A process is called* ***autoregressive*** *of order one, and denoted as AR(1) if*

$$x_t = \phi\, x_{t-1} + w_t\,. \tag{1.28}$$

Result 1.14. *An AR(1) process is stationary if* $|\phi| < 1$*. A random walk is a special case of an AR(1) process.*

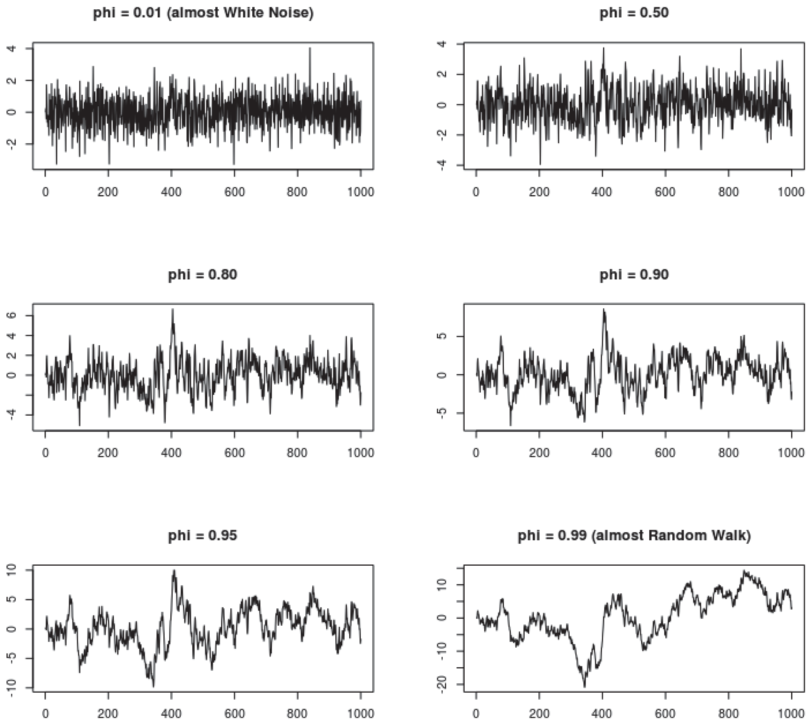


Figure 1.10. An AR(1) process ($x_{t+1} = \phi\, x_t + w_t$) changing the value of ϕ, $\phi \in [0,1]$. When ϕ goes toward zero the process tends to become a White Noise, when ϕ goes toward one the process tends to become a Random Walk.

Definition 1.15. *A process is called* ***moving average*** *of order q and denoted as MA(q) if*

$$x_t = \theta_1\, w_{t-1} + \theta_2\, w_{t-2} + \ldots + \theta_q\, w_{t-q}$$

Result 1.16. *An MA(q) process is always stationary, for every choice of $\theta_1, \ldots, \theta_q$.*

Observation 1.17. *If in a process realization we can see a* ***trend*** *then it is improbable it will be stationary indeed its mean seems to be non constant.*

Observation 1.18. *If in a process we can see* ***seasonality*** *it means that we can find a period T such that $x_t \approx x_{t+T}$ and $x_s \approx x_{s+T}$ where $x_t \neq x_s$. It is improbable that a process with seasonality will be stationary because it seems not to have a fixed mean.*

Observation 1.19. *If in a process we can see there is not constant* ***variability*** *then it is improbable it will be stationary because the variance of the process should be constant in time.*

Observation 1.20. *The last three observations use the word "improbable" because sometimes randomness can deceive us and build structures that we think are deterministic. For example, a random walk has no deterministic trend but, if by chance the realizations of its white noise come out in large majority to be larger than zero we may see a deterministic trend that is only outward. We will see an example of this in a next section.*

1.3.2 Random Walk tests, an Overview

Since we are interested in Random Walks, we are considering only tests on AR(1) processes, in the general context of ARIMA the same topic is named **unit root** test.

Now our AR(1) process is such that

$$x_t = \phi\, x_{t-1} + \varepsilon_t \quad , \tag{1.29}$$

where are ε_t are i.i.d. random variables distributed as $N(0, \sigma^2)$.

Given n observations[1.16] $x_1, \dots x_n$, the least squares estimator of ϕ is

$$\hat{\phi} = (\sum_{t=1}^{n} x_{t-1}^2)^{-1} \sum_{t=1}^{n} x_t\, x_{t-1}\,. \tag{1.30}$$

Method 1, [Box and Jenkins, 1970]. Under this method, we assume the model is a random walk, that is we suppose $\phi = 1$ (H_0: $\phi = 1$) and we compute the residuals as $e_t = x_t - x_{t-1}$. Then compute the Box-Pierce statistic Q_K

$$Q_K = n \sum_{k=1}^{K} r_k^2$$
$$r_k = (\sum_{t=1}^{n} e_{t-1}^2)^{-1} \sum_{t=k+1}^{n} e_t\, e_{t-k}\,.$$

Under the null hypothesis H_0: $\phi = 1$, Q_K is approximately distributed as χ^2 with K degrees of freedom.

[1.16] If necessary, consider $x_0 = 0$.

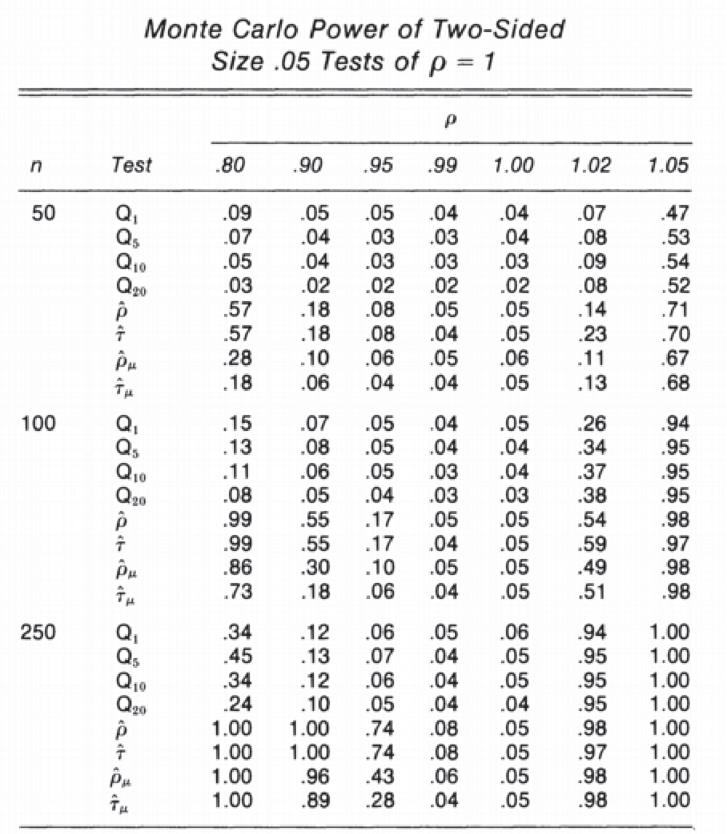
Method 2, [Dickey and Fuller, 1979]. This is probably the most cited Random Walk test. Given that the likelihood ratio statistic for H_0: $\phi = 1$ is a function of

$$\hat{\tau} = (\hat{\phi} - 1)\, S_e^{-1} (\sum_{t=2}^{n} x_{t-1}^2)^{1/2} \tag{1.31}$$

where

$$S_e^2 = (n-2)^{-1} \sum_{t=2}^{n} (x_t - \hat{\phi}\, x_{t-1})^2, \tag{1.32}$$

the authors established the limiting distribution of $\hat{\phi}_n$ and $\hat{\tau}_n$ under the assumption than $|\phi| = 1$. By Montecarlo simulation they show their test is more powerful than Box-Jenkins. I include below the same table appearing in their article.



Monte Carlo Power of Two-Sided Size .05 Tests of $\rho = 1$

		ρ						
n	Test	.80	.90	.95	.99	1.00	1.02	1.05
50	Q_1	.09	.05	.05	.04	.04	.07	.47
	Q_5	.07	.04	.03	.03	.04	.08	.53
	Q_{10}	.05	.04	.03	.03	.03	.09	.54
	Q_{20}	.03	.02	.02	.02	.02	.08	.52
	$\hat{\rho}$	.57	.18	.08	.05	.05	.14	.71
	$\hat{\tau}$	.57	.18	.08	.04	.05	.23	.70
	$\hat{\rho}_\mu$	.28	.10	.06	.05	.06	.11	.67
	$\hat{\tau}_\mu$	.18	.06	.04	.04	.05	.13	.68
100	Q_1	.15	.07	.05	.04	.05	.26	.94
	Q_5	.13	.08	.05	.04	.04	.34	.95
	Q_{10}	.11	.06	.05	.03	.04	.37	.95
	Q_{20}	.08	.05	.04	.03	.03	.38	.95
	$\hat{\rho}$	.99	.55	.17	.05	.05	.54	.98
	$\hat{\tau}$	.99	.55	.17	.04	.05	.59	.97
	$\hat{\rho}_\mu$	.86	.30	.10	.05	.05	.49	.98
	$\hat{\tau}_\mu$	.73	.18	.06	.04	.05	.51	.98
250	Q_1	.34	.12	.06	.05	.06	.94	1.00
	Q_5	.45	.13	.07	.04	.05	.95	1.00
	Q_{10}	.34	.12	.06	.04	.05	.95	1.00
	Q_{20}	.24	.10	.05	.04	.04	.95	1.00
	$\hat{\rho}$	1.00	1.00	.74	.08	.05	.98	1.00
	$\hat{\tau}$	1.00	1.00	.74	.08	.05	.97	1.00
	$\hat{\rho}_\mu$	1.00	.96	.43	.06	.05	.98	1.00
	$\hat{\tau}_\mu$	1.00	.89	.28	.04	.05	.98	1.00

Table 1.1. Shows the power the Dickey-Fuller proposed test with statistics ρ and τ against Q_K. The statistics ρ_μ and τ_μ refer to a version of the test which accepts as input a series with trend. The letter ρ is what we are calling ϕ in this section.

Method 3, [Dickey et al., 1984]. A seasonal time series can't be a stationary process because the mean is not constant. To overcome this issue the authors adapt the statistics $\hat{\phi}$ and $\hat{\tau}$ to the equation

$$x_t = \phi x_{t-d} + \varepsilon_t \tag{1.33}$$

where d is the lag corresponding to the seasonality of interest.

Method 4, [Phillips and Perron, 1988]. Their work relaxes the hypothesis of ε_t error being Normal and i.i.d. Moreover, their test manages also the case of random walks with drift and trend.

Method 5, [Ferretti and Romo, 1996]. The authors provide a Bootstrap based test for unit root. Through Montecarlo simulation they conclude the test is better than the alternatives in small samples. The test is not bound to Normality of ε_i.

In the last decades the research moved to many different directions, some of them are listed below.

1. Test the presence of unit root in more and more rich ARIMA models.
2. Test unit roots in presence of shocks.
3. Test in unit roots in panel data.
4. Consider the case of ε_i with fat tails.

1.3.3 Random walks, Applications and Curiosities

It may appear that a *Random Walk* is a very narrow and special topic with a catchy name but far from being of general interest. This is not the case. In the present section we report the curious historical birth of the expression "random walk" and some intriguing application of Random Walks to Economics. The interested reader could see [Feller, 1957, ch.3] for a different but still simple view on the subject from a different perspective and more examples of application ranging from *The Ballot Theorem* to the *Arcsine Law*.

1.3.3.1 Etymology of "Random Walk" and the drunken man

We owe credit for the introduction of the term *random walk* to Karl Pearson [Hughes, 1996]. He wrote a letter the the Journal *Nature* which appeared on the issue 27 July 1905 saying:

The problem of the random walk

> Can any of you readers refer me to a work wherein I should find a solution of the following problem, or failing the knowledge of any existing solution provide me with an original one? I should be extremely grateful for aid in the matter.
>
> A man starts from a point O and walks l yards in a straight line; he then turns through any angle whatever and walks in another l yards in a second straight line. He repeats this process n times. I require the probability that after n of these stretches he is at a distance between r and $r + \delta\, r$ from his starting point O.
>
> Karl Pearson
> The Gables, East IIsley, Berks

One week later, on 3 August 1905, on Nature was published the reply by Lord Rayleigh:

> This problem, proposed by Prof. Karl Pearson in the current number of NATURE, is the same as that of the composition of n iso-periodic vibrations of unit amplitude and of phases distributed at random, considered in *Phil. Mag. x, p.73, 1880; xlvii, p 246, 1899; ('Scientific Papers', i.,p. 491, iv., p.370).*
>
> If n be very great, the probability sought is
>
> $$\frac{2}{n}\, e^{-r^2/n}\, r\, d\, r.$$
>
> Probably methods similar to those employed in the papers referred to would avail for the development of an approximate expression applicable when n is only moderately great.
>
> RAYLEIGH
> Terling Place, July 29.

Then there is the reply of Pearson, which I will report only in part:

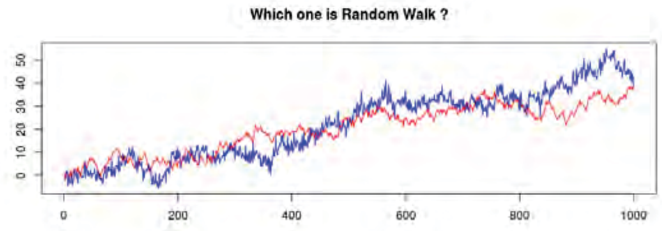
> ... I ought have known it, but my reading of late years has drifted into other channels, and one does not expect to find the first stage in biometric problem provided in a memoir on sound. From the purely mathematical point of view it would still be very interesting to have a solution for n comparatively small.

> … The lesson of Lord Rayleigh's solution is that in open country the most probable place of finding a drunken man who is at all capable of keeping on his feet is somewhere near his starting point.
>
> KARL PEARSON

Pearson's interest in the problem of Random Walk arose from an attempt to model random migrations, with the particular case of mosquitoes invading cleared jungle regions.

1.3.3.2 Random Walks in Economics

Can we distinguish something that grows because of randomness from something that grows because there is a fixed law behind it? Look at next picture, one series is a Random Walk, the other is an AR(1) process added to a deterministic positive trend. In one the growth is "true", caused by a discernible force, in the other is just an illusion, the whim of a stochastic process.



The red line is a Random Walk, the blue is an AR(1) with drift, formally $x_t = 0.035 + 0.8 * x_{t-1} + e_t$. The e_i were i.i.d from a $N(0, 1)$ for both the series. These data were simulated. I crafted the two processes to look similar. But what happens in real life, when we have a time series realization and we don't know its generating mechanism? Can we distinguish a trend from randomness?

Economists strive to build models to predict our GDP growth, unemployment level, demand in the house market, cost of electrical energy and a lot of other parameters that directly influence our day by day life. All of these quantities are apparent time series. Economists also know from the beginning that their model will never be totally precise, there are too many factors to take into account that they can not control. But, it is largely supposed that in our econometric models there are two different parts, a static, deterministic part that we can understand and partially control, and also a random part which is out of our reach.

The article [Nelson and Plosser, 1982] shocked this assumption. The authors analyzed 14 historic econometric series[1.17] from the beginning of 1900 to 1970 and compared them with the Random Walk. Their conclusion was that, it could not be rejected that these series were Random Walk. This fact is of the utmost importance because if a series grows because of a random walk instead of being a random fluctuation around a deterministic trend, it means it can change direction at any time, and we have no power to influence it because it is totally randomness driven.

A contemporary article by [Meese and Rogoff, 1983] raised another very interesting issue. Analyzing U.S. dollar exchange rates[1.18] the authors found that in out of the sample predictions, a Random Walk with drift performs as well as the most reliable econometric model. The two authors were both at the *Federal Reserve Board* when they wrote the paper so, they were comparing Random Walk not to some obscure model with nice analytical properties but with the one in use by U.S. government to make predictions. Indeed, the paper was a revision of a previous one they submitted in 1981 to the International Monetary Fund.

So let's conclude our introduction with a provocation, if you were an investor and you could see the following plots about two popular NASDAQ companies. Would you buy their shares looking at the plots? Do you think they are Random Walks? Beware, if they are Random Walks, your investment is as grounded as flipping a coin.

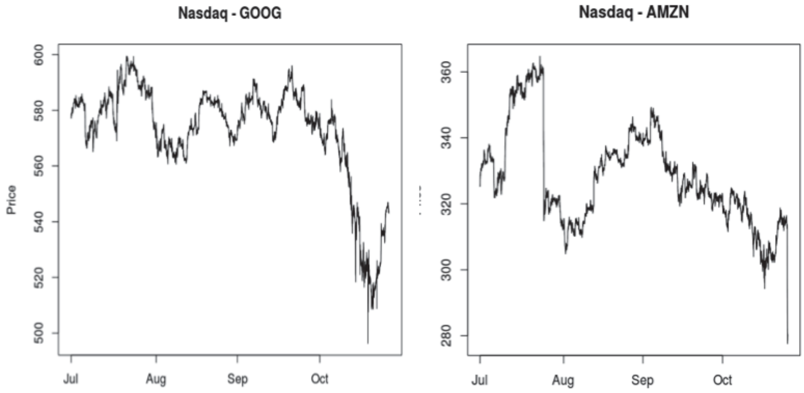


Figure 1.11. NASDAQ stock prices for Google and Amazon, from 1-July to 23-October 2014. Data were bought from eoddata.com.

[1.17] Real GNP, Nominal GNP, Real per capita GNP, Industrial production, Employment, Unemployment rate, GNP deflator, Consumer prices, Wages, Real Wages, Money stock, Velocity, Bond yield, Common stock prices.

[1.18] dollar/pound, dollar/mark, dollar/yen and "trade-weighted-dollar".

1.4 Structure of the Thesis

The purpose of this introduction was to make accessible and to motivate the material in the next chapters. What we have seen until now applies to classical discrete dataset, from here on we radically change perspective, data will be no more merely a set of numbers but it will be a set of functions. It is of course all in the hypotheses, data are still coming as discrete tables, but we aggregate them properly and see them as functions.

In Chapter 2 we will extend the Lasso to Functional Regression, indeed in the case we study the response variable is no more a number but a function. Our way to solve the resulting functional problem does not depend on sampling, data for us remain functions, till the solution of the problem. After that, we will improve the Lasso to take care of a special condition arising in Functional Regression which can not be easily appreciated in the multivariate case. We called the resulting *post hoc* technique the Shaked Lasso.

In Chapter 3 we will introduce a Random Walk test for Functional Autoregressive Processes of order One. We saw in a classical autoregressive processes of order one the current state is represented by a number x_t, the nearest future state by another number x_{t+1} and they are related by Equation 1.28. In the functional case the current and the nearest future state are both functions and, as far as we could establish, our work is the first one to test the Random Walk hypothesis under this condition.

In Chapter 4 we will stress some conclusions of our research and point out some possible directions for future work. Finally, in the Appendix can be found the detailed results of our simulations.

Chapter 2

The Shaken Functional Lasso

2.1 Introduction

In this chapter we describe a variable selection method for functional regression that we call **The Shaken Functional Lasso** (for short **Shaken Lasso** or also **SLasso**). The development of the method was motivated by the following problem we had at hand. *Given a set of car accidents in which one car is standing still and the other runs it over, we know the speed of the two cars just before and after the impact and their respective length, weight, height and width. We want to establish which factors are important in modeling the speed of the struck car just after the impact*. This practical problem and its implications will be discussed in detail in Section.2.4. In the following we will give an overview of the variable selection problem in Functional Regression, then we will introduce our computation methodology for the Functional Lasso and our *post hoc* decision strategy to improve the Lasso which we call the Shaked Lasso. An extensive simulation study to support the method validity presented in Section.2.3.

We have seen in Section 1.2 the fundamentals of classic Linear Regression and many ways to perform variable selection. In the functional case, there are three different types of linear regression, depending on where the functional objects are located: in the response, in the regressor(s) or in both terms.

$$
\begin{array}{lll}
1) & Y_i(t) = \int \psi(t,s)\, X_i(s)\, \mathrm{ds} + \varepsilon_i(t) & \text{(Fully Functional Model)} \\
2) & Y_i = \int \psi(s)\, X_i(s)\, \mathrm{ds} + \varepsilon_i & \text{(Scalar Response Model)} \\
3) & Y_i(t) = \psi(t)\, x_i + \varepsilon_i(t) & \text{(Functional Response Model)}
\end{array}
$$

An introduction to the three kind functional regression is available in [Ramsay and Silverman, 2005, ch.12-16], a recent compact review can be found in [Horváth and Kokoszka, 2012, ch.8] and in [Ferraty and Romain, 2011, ch.1-2]. The three kinds of regression rise different problems and require different treatment. We will consider here only the case of Functional Response and Scalar Regressors.

The case of Scalar Regressors and Functional Response is, in some sense, more troublesome than others because it requires to control a function by a set of scalar valued regressors that are lower dimensional, less informative, objects. The error terms ϵ_i are in this context functions $\epsilon_i(t)$ and therefore we can not apply familiar distributional properties. Finally, the results of our models will be functions. To understand if they are interesting we have to compare them with other output functions, that is, we have to compare plots.

We are going to study models with several variables, of type

$$Y_i(t) = \beta_0(t) + \beta_1(t)\, X_{i,1} + \cdots + \beta_J(t)\, X_{i,J} + \varepsilon_i(t) \qquad i = 1 \ldots I\,. \tag{2.1}$$

Our objective is to establish which of the $\beta_i(t)$ are useful and which can be considered null. Equivalently, we want to establish which regressor X_i influence $Y_i(t)$ and which ones do not. We will call an influential X_i as *active/useful regressor* and its associated $\beta_i(t)$ an *active* (or *useful*, or *non null*) coefficient. We would like to discard as many regressors as possible because a smaller model is easier to interpret than a large one and also, because reducing the number of variables in general reduces the variance of the estimator and prevents overfitting.

Estimating the parameters $\beta_j(t)$ can be done in principle by Least Squares,

$$\hat{\beta}_0 \ldots \hat{\beta}_J := \underset{\beta_0(t)\ldots\beta_j(t)}{\text{Argmin}} \sum_{i=1}^{I} \left\| Y_i(t) - \beta_0(t) - \sum_{j=1}^{J} \beta_j(t)\, X_{i,j} \right\|_2 , \tag{2.2}$$

$\beta_i(t)$ are constrained to some functions space we will describe later. After estimation, we want to establish which $\hat{\beta}_j(t)$ to consider zero. Looking back the methods we have seen in 1.2 we face now some difficulties which make many of them not directly available.

1. **Empirical Variable Selection**. It is more difficult now to establish if a parameter β_j is "small" because here β_j is a function $\beta_j(t)$. Thus, it can be small for a large part of the domain, and then not so small in the remaining part.

2. **APM**. It is still available if coupled with cross validation as a measure of performance of each single model. The problem is that it is very inefficient in general.

3. **Classical Methods**. Based on *Student* t or F distribution are not available because they require ε_i to be Normal, in our case not only ε_i is non Normal, it is a function.

4. **Forward/Backward Stepwise**. Being based usually on R^2_{adj}, BIC, or AIC to compare each model to the one at the next step they can not be used directly without assuming normality, or other distributional properties of ε_i.

5. **Based on Principal Components**. These methods are in principle available because we can use FPCA as defined in 1.1.5.2, but they have a drawback: they mix all the regressors making the interpretation difficult.

As we have see in the previous list, the fact that ε_i are functions is the major cause of difficulties but there is a way to cut off the problem of the functional error and it is to consider our Functional Regression pointwise as an (infinite) sequence of scalar regressions, let's call this strategy the **the pointwise loophole**. In such a way, provided the $\varepsilon_j(t)$ be realization of Gaussian process, we can assume Normality for $\varepsilon_j(t_i)$ at all times t_i. This method is followed in [Ramsay and Silverman, 2005, ch.12-13] but, as the authors recognize in Section.13.5.3, it should be handled with care. The method is indeed powerful and has the great benefit of reducing the problem to known methodologies but, on the other side, each test is made independently on every time point t_i and this contrast with our idea that data are coming from smooth functions, which implies the $\{\varepsilon_j(t_i)\}_i$ can not be independent. Under the shadow cast by this contradiction, we tried to look for a different solution and, more in general, to work always on functions. After we smooth a dataset, we work with the function basis coefficients, we don't go back to point by point arguments.

A solution based on Confidence Intervals. The first idea we tried was to establish a 90% (or 95%) confidence band for each $\beta_i(t)$ and then, if the confidence band contains always zero (the X-axis) we should consider the parameter $\beta_i(t)$ zero. To get a confidence band, first of all we replicate the estimation of $\beta_i(t)$ via Bootstrap and then, we consider as 90% confidence interval the MinMax band of the 90% deepest curves in the set. The depth function we use is the one defined in [López-Pintado and Romo, 2009]. A result of the process can be seen in Fig.2.1, the part in green is the MinMax band of the 90% deepest curves. Details on the model and the plot can be found in Appendix B.1.

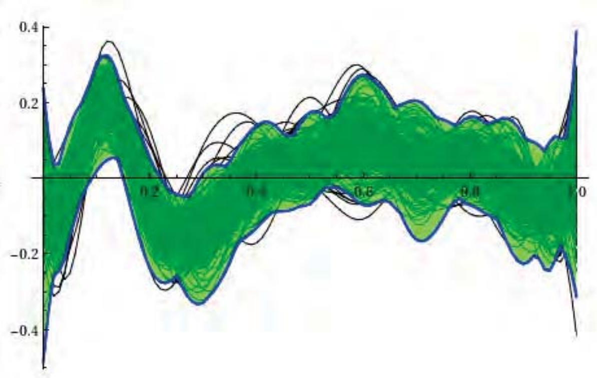


Figure 2.1. A 90% confidence band for a parameter β_i.

This approach, even if pictorially impressive, was opening more questions than providing answers. Indeed, as it can be seen from Fig.2.1, it can happen that the 90% confidence band will envelop the X-axis without containing it fully. We speculated about considering the parameter null if it contains the X-axis 90% percent of the time. That idea was not further investigated because, if the peak at $t \approx 0.1$ becomes high enough, we should discard the hypothesis of null $\beta_i(t)$ in any case. In conclusion, we thought to abandon the method in favor of something more robust and possibly giving a direct answer to a simple question: "Can we discard the $\beta_i(t)$ parameter?".

At this stage Lasso came into play. It was chosen because its working does not depend upon the distribution of ε_i. Starting from its definition in multivariate regression

$$\bar{\beta}_0^{(\lambda)} \dots \bar{\beta}_J^{(\lambda)} = \underset{\beta_0 \dots \beta_J}{\text{Argmin}} \sum_{i=1}^{I} \left(Y_i - \sum_{j=0}^{J} \beta_j X_{i,j} \right)^2 + \lambda \cdot \sum_{j=1}^{J} |\beta_j|, \tag{2.3}$$

we adapt it to our case of functional regression and the optimization problem becomes

$$\{\bar{\beta}_j^{(\lambda)}(t)\}_j = \underset{\beta_0(t) \dots \beta_J(t)}{\text{Argmin}} \sum_{i=1}^{I} \int_a^b \left(Y_i(t) - \sum_{j=0}^{J} \beta_j(t) X_{i,j} \right)^2 dt + \lambda \cdot \sum_{j=1}^{J} || \beta_j(t)||_1 \tag{2.4}$$

Then, as usual, changing λ we find the value $\bar{\lambda}$ that minimizes the cross validation error and finally call $\bar{\beta}_j$ the Lasso estimators for β_j ($\bar{\beta}_j \overset{d}{=} \bar{\beta}_j^{(\bar{\lambda})}$ for all j).

The $\{\beta_0(t), ..., \beta_J(t)\}$ live in some function space to be chosen and for us it will be the linear combinations of BSpline basis function. The 1-norm is defined, as usual, as $|| \beta_j(t)||_1 \overset{d}{=} \int_a^b |\beta_j(t)|$ dt. The last addend in (2.4) involves the integration of absolute values, therefore obtaining a direct analytic solution of the optimization problem is not trivial. In Section.2.2 we will see how to replace the penalization term $\sum_{j=1}^{J} || \beta_j(t)||_1$ with $\sum_{j,k} |b_{j,k}|$ where $b_{j,k}$ are the coordinates of $\beta_j(t)$ respect to some functional BSpline basis functions $\mathcal{B}$. All of the problem will be reduced to an optimization on scalar values by algebraic transformations, without sampling, this is the first original contribution of our work.

The Lasso, or L_1 regularization, has been already approached in Functional Regression during the last years, even if not performed in the same way we are doing here. In [Matsui and Konishi, 2011] it is applied for variable selection to a functional regression with functional predictors and scalar response using Gaussian basis functions. In [Hong and Lian, 2011], the Lasso method is applied to a functional regression with functional predictors and functional response. In this case β_j are scalars and functions are sampled on arbitrary grids to reduce the problem to a numerical solvable one. In [Zhao et al., 2012] the response is scalar, there is only one functional variable to estimate, the basis is Wavelet and Lasso is used to set to zero as many coefficients as possible in the Wavelet basis.

Lasso, in general, performs well even in Functional Regression but it is not immune to the situation illustrated in Fig.2.1. That is, it happens frequently that it shrinks $\beta_i(t)$ to zero almost everywhere, except for some isolated parts of the domain. In these cases we do what we call "**shake the Lasso**". Basically, we move a bit the parameter λ from its optimal[2.1] value $\bar{\lambda}$ and observe the change in shape of the estimated $\beta_i(t)$. Under this variation, parameters associated to active/inactive regressors tend to have a characteristic behaviour which makes them recognizable. We will see in Section.2.3.0.6 that this behaviour permits us to solve systematically many of the uncertain cases left open by Functional Lasso. This is the second original contribution of our work and will be called in the next section *Rule 2*. This name emphasizes the fact that it can be applied after the Functional Lasso.

2.2 Methodology

Problem (2.4) is a functional problem that could be difficult or very tedious to solve analytically. To be able to compute numerically the $\bar{\beta}_j^{(\lambda)}(t)$ we choose a basis function $\mathcal{B} = \{\phi_0(t), ..., \phi_K(t)\}$ and express all functions in (2.4) as a linear combination of the basis. For example, the response variables become $Y_i(t) = \sum_{k=0}^{K} a_{i,k}\ \phi_k(t)$ and beta parameters become $\beta_j(t) = \sum_{k=0}^{K} b_{j,k}\ \phi_k(t)$. It has to be stressed that coefficients $a_{i,k}$ are known real numbers because $Y_i(t)$ are known functions. On the contrary, $b_{j,k}$ are unknown reals since $\beta_j(t)$ are unknown functional parameters to be estimated. We will estimate the values $b_{j,k}$ solving the following optimization problem and denote the estimates as $\bar{b}_{j,k}^{(\lambda)}$.

$$\begin{aligned}\{\bar{b}_{j,k}^{(\lambda)}\}_{j,k} = \underset{b_{j,k}}{\text{Argmin}} \ & \sum_{i=1}^{I} \int_a^b \left(\sum_{k=0}^{K} a_{i,k}\, \phi_k(t) - \sum_{j=0}^{J} (\sum_{k=0}^{K} b_{j,k}\, \phi_k(t))\, X_{i,j} \right)^2 dt + \\ & + \lambda \cdot \sum_{j=1}^{J} \int_a^b \left| \sum_{k=0}^{K} b_{j,k}\, \phi_k(t) \right| dt.\end{aligned} \tag{2.5}$$

The first part of the optimization function, the sum of integrals of a square, reduces algebraically to a quadratic form on variables $b_{j,k}$ but the remaining part can not be easily simplified without further informations. To overcome this difficulty we resort to a BSpline property, citing DeBoor *"B-Spline coefficients model the function they represent."*, see [De Boor, 2001], Example IX.2. The property is illustrated by an example in Fig. 2.2. If we suppose we are using a cubic spline with knots $\{t_0, t_1, ...\ t_n\}$ on the domain $[a, b]$

[2.1] Optimal value of λ, in the sense that it minimizes the cross validation error.

where[2.2] $t_0 = t_1 = t_2 = t_3 = 0$ and $t_n = t_{n-1} = t_{n-2} = t_{n-3} = 1$. For all other t_i we set $\Delta := t_{i+1} - t_i$ then, for each j

$$\begin{aligned}\int_a^b \left|\sum_{k=0}^{K} b_{j,k}\,\phi_k(t)\right| dt &= \|\,\beta_j(t)\|_1 = \int_a^b |\beta_j(t)|\,dt \\ &\approx \sum_{i=3}^{n-4} |\beta_j(t_i)| \cdot \Delta && \text{(Riemann Integral)} \\ &\approx \Delta \cdot \sum_{k=1}^{K-2} |b_{j,k}| \leq \Delta \cdot \sum_k |b_{j,k}| && \text{(BSplines Property).}\end{aligned} \tag{2.6}$$

The Δ value can be removed because it would only rescale λ (see Eq. 2.5). We finally get the objective function:

$$\{\bar{b}^{(\lambda)}_{j,k}\} = \underset{b_{j,k}}{\text{Argmin}} \left(\text{Quadratic}(b_{j,k}) + \lambda \cdot \sum_{j,k} |b_{j,k}|\right). \tag{2.7}$$

Fixing $\lambda \geqslant 0$ we can easily compute (2.7) because it is now a numerical convex optimization problem for which there are specialized solvers as CVX [Grant and Boyd, 2011, Grant and Boyd, 2008]. The problem being convex ensures solutions $\bar{b}^{(\lambda)}_{j,k}$ to be unique[2.3].

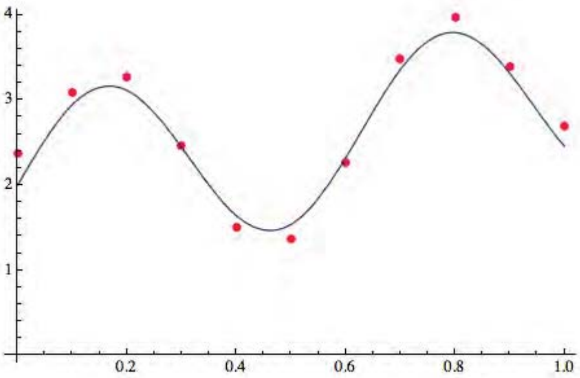


Figure 2.2. A function $f(x) = 2 + x + \text{Sin}(10 \cdot x)$ in [0, 1] has been plotted in blue, it has been sampled in 101 points $x_i = i \cdot 0.01$ for $i = 0, ..., 100$ and finally fitted by an order 3 BSplines with 17 knots $\mathcal{K} = \{0, 0, 0, 0, 0.1, 0.2, ..., 0.8, 0.9, 1, 1, 1, 1\}$. The initial and final repeating values are needed for $f(x)$ non periodicity. The BSpline functions basis is composed of 13 elements $\{\beta_0(t), ..., \beta_{12}(t)\}$, in that basis $f(x)$ is represented as $\sum_{i=0}^{12} b_i\,\beta_i(t)$. The red points have coordinates $\{(0.1 \cdot (i-1), b_i)\}_{i=1...11}$ and lay all tight around the function graph. We could include extreme basis coefficients by averaging.

[2.2] These cumbersome conditions are necessary to get a proper behaviour from the BSpline on the domain borders $\{0, 1\}$.

[2.3] Actually, strict convexity would ensure uniqueness of the point of minimum.

Till now we have shown how to get a numerical solution to the Functional Lasso when basis functions are BSpline. Our proposed method for variable selection is composed of two steps, the first one is basically the Lasso, the second is a graphical criterion that might be applied when Lasso fails to give a secure answer. Being the criterion of graphic nature it is best introduced by examples than by abstract statements. In this section we will show two examples. The first one illustrates and motivates all of the test practice. The second example explores what happens in the very special case when no variable is null.

2.2.0.3 First Example

A data set is made of 30 functional observations built as follows:

s1. Generate 6 regressors as 6 vectors of 30 random values: $X_{i,j} \sim N(0, \sqrt{5})$, $i = 1 \dots 30, j = 1 \dots 6$.[2.4]

s2. Define three functions $\beta_j(t), t \in [0, 1], j = 0, 1, 2$ as

$$\begin{cases} \beta_0 = 30\, t\, (1-t)^{3/2} \\ \beta_1 = 10\, (t-0.6)^2 + 1 \\ \beta_2 = S\, i\, n\, (4\, \pi\, t) - 2. \end{cases} \tag{2.8}$$

s3. Define 30 error functions $\epsilon_i(t)$ in $t \in [0, 1]$ by generating 101 random points P_k and then joining them continuously with a linear interpolation. $P_k := (x_k, y_k)$, $x_k := \frac{1}{100}\, k$, $y_k \sim \text{Normal}(0, \sigma = 0.8)$ for $k = 0, 1, \dots, 100$. The value $\sigma = 0.8$ is arbitrary but appears to be reasonably sized (see Figure 2.4).

s4. Generate 30 functional response variables as

$$y_i(t) = \beta_0(t) + \beta_1(t)\, X_{i,1} + \beta_2(t)\, X_{i,2} + \epsilon_i(t) \qquad i = 1, \dots, 30. \tag{2.9}$$

s5. Get the discrete representation of the response variables as: $Y_{i,j} = y_i(t_j)$, $t_j = \frac{1}{100}\, j$, $j = 0,\ 1,\ \dots,\ 100$. From this point on, consider cleared the variables $y_i(t)$, we need them to denote other objects. For a representation of $y_i(t)$ and Y_i see Figure 2.3.

[2.4] Normal distributions will be always written as $N(\mu, \sigma)$ in this chapter.

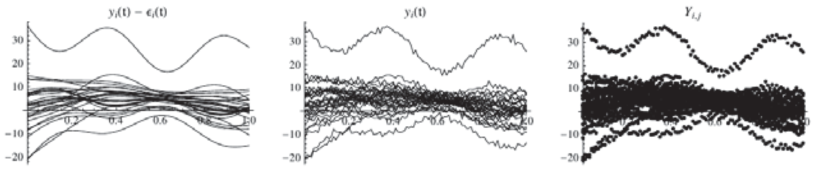


Figure 2.3. Generation of an artificial data set $Y_{i,\,j}$ by known functional parameters $\{\beta_0(t), \beta_1(t), \beta_2(t)\}$. The first plot, on the left, represents the set $f_i(t) = \beta_0(t) + \beta_1(t)\, X_{i,1} + \beta_2(t)\, X_{i,2}$, the second displays $y_i(t) = f_i(t) + \epsilon_i(t)$ and the third one finally all $Y_{i,\,j}$. It should not surprise too much that there is a line far from the others, it happened that some of the $X_{i,\,j}$ was large, it is $X_{12,2} \approx -7.6$, it lays between $3\,\sigma$ and $4\,\sigma$ from the mean$(X_{i,\,j}) = 0$. The probability of obtaining always values smaller than 3σ for all the 180 $X_{i,\,j}$ is approximately 60%, we were on the other 40% side.

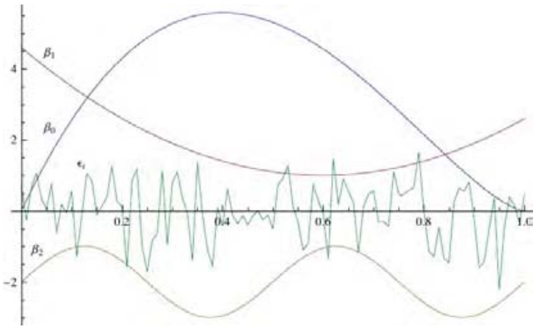


Figure 2.4. Initial parameters $\{\beta_j(t)\}_{j=0,1,2}$ and a realization of the error function $\epsilon_i(t)$.

At this point we have a data set $(Y_{i,\,j},\ X_{i,1},\ X_{i,2},\ X_{i,3},\ X_{i,4},\ X_{i,5},\ X_{i,6})$ where $i = 1, \ldots, 30$, $j = 0, \ldots, 100$. We forget now we know how this data has been generated. To improve readability we will use a vector notation to denote the discrete response variables. Y_i will be a vector of 101 elements whose k-th element is $Y_{i,k}$. We want to explain the response variable $\mathbf{Y}_i$ by mean of a functional linear regression model with functional response and scalar covariates $\{X_{i,1}, X_{i,2}, X_{i,3}, X_{i,4}, X_{i,5}, X_{i,6}\}$. A well performing method will recognize that useful regressors are only $\{X_{i,1}, X_{i,2}\}$ and will find the estimated parameters $\tilde{\beta}_0(t), \tilde{\beta}_1(t), \tilde{\beta}_2(t)$[2.5] to be close to the original parameters $\beta_0(t), \beta_1(t), \beta_2(t)$.

As required to apply our functional regression we transform the response variables $\mathbf{Y}_i$ into functions. One way to do this is to fit each $\mathbf{Y}_i$ with a function in some predefined functions space. The choice of the functions space

[2.5] $\tilde{\beta}_j(t)$ denotes the estimation of a parameter $\beta_j(t)$ by some method left to determine.

is in part arbitrary, [Ramsay and Silverman, 2005, Ch.3] presents some classical basis functions and some rules of thumb to choose between them. In this case we choose order three BSpline basis functions with ten equally spaced internal knots more three equal knots at points 0 and 1, beginning and end of functions domain. The equal values at the ends are required to reduce smoothness at the domain borders.

$$\mathcal{K} = \{0, 0, 0, 0, 0.1, 0.2, ..., 0.8, 0.9, 1, 1, 1, 1\} \tag{2.10}$$

The knots sequence has been chosen by trading off simplicity and effectiveness. Other ways to place the knots are surely possible. A cross validation could be used to determine, in some sense, the optimal number of knots, but the amount of smoothness required for each case study remains largely dependent on the eye of the modeler [Faraway, 1997]. Using the knots sequence $\mathcal{K}$ we obtain a 13 element cubic BSpline basis functions $\mathcal{B} = \{\phi_0(t), \phi_1(t), \ldots\ \phi_{12}(t)\}$, we compute them with *Mathematica 8.0 BSplineBasis* built-in command. An introduction to symbolic BSplines manipulation with *Mathematica* can be found in [Iglesias et al., 2007]. The analysis proceeds as follows:

s6. We standardize[2.6] all variables. Standardized regressors will be denoted with XS_j and computed naturally as:

$$\mathrm{XS}_j := \mathrm{Standardize}((X_{1,j}, X_{2,j}, \ldots X_{30,j})), \quad \text{for } j \in \{1, ..., 6\}, \tag{2.11}$$

response variables $\mathbf{Y}_i$ are standardized all together as:

$$\mathrm{Standardize}((Y_{1,0}, ..., Y_{1,100}, Y_{2,0}, ..., Y_{2,100}, ..., Y_{30,0}, ..., Y_{30,100})), \tag{2.12}$$

their standardized versions are denoted YS_i.

s7. Fit each YS_i to a function $y_i(t)$ in the function space determined by $\mathcal{B}$ minimizing the squared error.

s8. Set XS_0 to be a length 30, vector of ones and and define a linear model $\mathcal{M}_i(t)$ to explain each of $y_i(t)$ as $\mathcal{M}_i(t) := \sum_{j=0}^{6} \beta_j(t) \cdot \mathrm{XS}_{i,j}$, $\mathrm{XS}_{i,j}$ is the i-th element in vector XS_j.

s9. A solution in this context is an estimation of the $\beta_j(t)$ parameters giving a best fit to the data. We find here two kinds of solutions and compare them, the common least squares solution and the new functional lasso

[2.6] Variables standardization is required also in the unfunctional Lasso. Since Lasso drops parameters looking at their magnitudes, to make a meaningful comparison between parameters laying potentially on very different scales it is necessary to standardize them all.

solution, each of them will be denoted respectively as LS and FL. Parameters that are LS solution will be denoted as $\hat{\beta}_j(t)$, FL solution will be denoted as $\bar{\beta}_j(t)$.

$$\begin{aligned}\{\hat{\beta}_j\}_{j=0,\dots,6} &:= \underset{\{\beta_j\}_{j=0\dots6}}{\text{Argmin}} \sum_{i=1}^{30} \int_0^1 (y_i(t) - \mathcal{M}_i(t))^2 \\ \{\bar{\beta}_j^{(\lambda)}\}_{j=0,\dots,6} &:= \underset{\{\beta_j\}_{j=0\dots6}}{\text{Argmin}} \sum_{i=1}^{30} \int_0^1 (y_i(t) - \mathcal{M}_i(t))^2 + \lambda \cdot \sum_{j=1\dots6} ||\,\beta_j(t)||_1\end{aligned} \tag{2.13}$$

s10. Observe explicitly that $\hat{\beta}_j = \bar{\beta}_j^{(0)}$. We are going to reduce the sum of integrals to a quadratic form in $b_{j,k}$ by means of *Mathematica* computer algebra capabilities. Reduce $\sum\,||\,\beta_j(t)||_1$ to $\sum\,|b_{j,k}|$ as illustrated in the previous section and solve the resulting unconstrained convex optimization problem

$$\underset{b_{j,\,k}}{\text{Argmin}}\,\left(\text{Quadratic}(b_{j,k}) + \lambda \sum |b_{j,k}|\right)$$

by *Matlab CVX* package. The part that takes more time is the algebraic reduction of integrals, more or less half an hour with a mid-range laptop, the optimization part is faster and takes around a minute.

s11. We compute $\bar{\beta}_j^{(\lambda)}$ for many values of λ and look for the value $\bar{\lambda}$ that minimizes the five-out cross validation error. Once found, we define the FL parameters as $\bar{\beta}_j(t) := \bar{\beta}_j^{(\bar{\lambda})}$ for $j = 0, \dots, 6$.

The estimated parameters computed by least squares ($\hat{\beta}_j(t)$) and by the Functional Lasso ($\bar{\beta}_j(t)$) can be seen in Figure 2.5 and Figure 2.6. In Figure 2.5 there is a direct comparison between $\hat{\beta}_j(t)$ (in dashed red stroke) and $\bar{\beta}_j(t)$ (in full black) for each j. It can be seen that Lasso shrinks all spurious parameters $\{\beta_3(t), \dots, \beta_6(t)\}$ to zero while ordinary least squares keep them fluctuating around the x-axis without annihilating them. It is exactly the same thing that happens in ordinary multiple regression. The difference here is that, instead of scalars, whole functional parameters are set to zero. It is much easier to decide which is a useless regressor using Lasso solution. The shape of estimated $\{\beta_0, \beta_1, \beta_2\}$ are similar to their original values for both methods, only at domain borders there is a little discrepancy. The cross validation is minimal for the Lasso solutions, we shrunk the parameters but we actually improved the performance of the model. We must stress that parameter selection in this case has been very easy since some of them have been completely shut down to zero. In general, it will not always be so clear, therefore we set a formal rule to decide if a parameter has to be dropped applying only the Functional Lasso.

Rule 1. Rule of magnitude. We consider a regressor variable X_j spurious, or not effective, if its associated functional parameter is too small in magnitude: $\max_{t\in[0,1]} |\beta_j(t)| \leq 0.01$. This rule is a conservative extension to the functional context of the one implicitly used in [Tibshirani, 1996]. It makes sense only when variables are standardized or transformed to lay near zero. Parameter 0.01 is arbitrary.

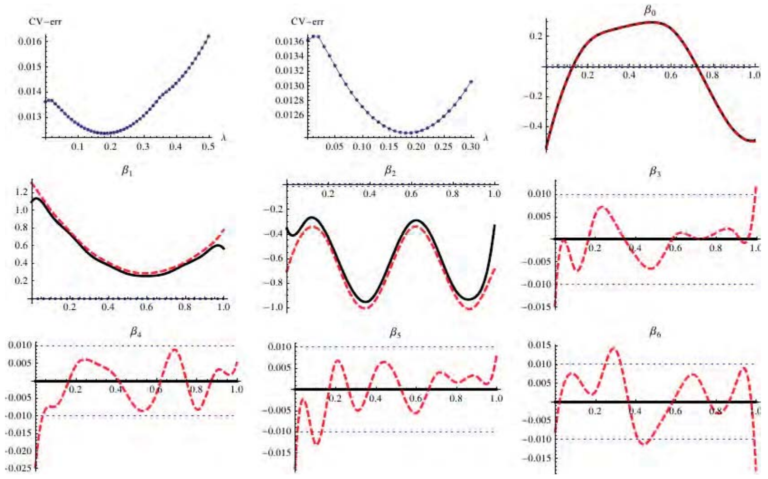


Figure 2.5. The first two plots starting from the top left corner show the cross validation error as a function of λ, it can be seen it reaches a minimum at $\lambda \approx 0.18$. The remaining plots compare the least square versus the Lasso estimation of all parameters $\beta_j(t)$. Least squares estimations are drawn in dashed red lines, Lasso in thick black. It is evident how lasso shrunk to zero all the spurious parameters $\beta_3(t), \ldots, \beta_6(t)$.

In Figure 2.6 the parameter shrinkage process is shown. We change the penalization term λ in the interval [0, 0.02] and observe how parameters do change. It is manifest that spurious parameters change a lot. On the contrary, effective parameters remain almost unvaried. In this case we know in advance which parameter should be dropped but, in general, this could be a useful explorative technique to decide if a parameter has to be retained or dropped. If, increasing λ, some parameters change far more than others then these parameters are likely spurious parameters.

Rule 2. Rule of inertia. When the Functional Lasso shrinks a set of parameters near to zero but not enough to make them satisfy Rule 1, then we consider them *candidate null*. To establish if these parameters are to be considered null we increase the penalization term λ. If on increasing λ the candidate null parameters are the only ones to change shape and to further shrink toward zero then we conclude they are definitely null.

The result synthesised in the last *Rule of inertia* has been observed during experimentation with different linear models and error function realizations. In this chapter it can be seen applied on the real data set, see Fig. 2.17, and in about half of the systematic simulations described in Section 2.3, The Rule of Inertia, is the distinguishing feature of the Shaken Functional Lasso.

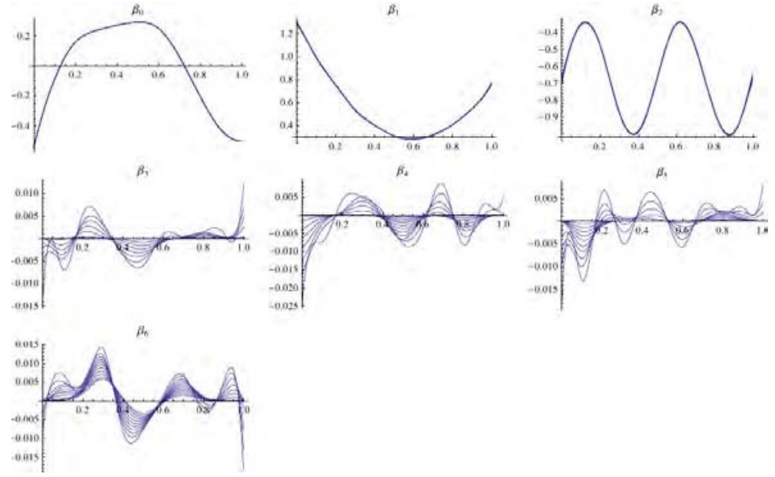


Figure 2.6. Effect of the penalization term λ on the size and shape of functional Lasso parameters $\bar{\beta}_j(t)$. Here λ takes values in the arithmetic sequence from 0 to 0.02 with 0.002 step. As we can see the effective parameters $\{\beta_0, \beta_1, \beta_2\}$ are far less sensitive to λ changes respect to spurious ones $\{\beta_3, \beta_4, \beta_5, \beta_6\}$. This can be considered a valuable explorative tool when it is unsure if a parameter should be discarded looking only at its magnitude.

Rule 2 is phenomenological, it has been discovered by experiments and by now we don't have a thoretical support for it. In any case, let's consider what we have. Rule 2 runs after Rule 1, so after the best parameters ($\bar{\lambda}$, $\{\bar{\beta}_i\}_i$) have been determined according to L^1 penalty and the cross validation score. Cross validation takes into account only predictive power, not model parsimony. Increasing a bit the value λ respect to $\bar{\lambda}$ we are willing to sacrifice small quantum of predictive power in favour of model simplicity. If increasing λ all the change is on shrinking some parameters β_i which were already very small at $\bar{\lambda}$ and only locally outside the zero band $[-0.01, 0.01]$, then, we conclude those parameter should be definitely set to zero. Indeed, if all the increase in penalization discharge only on a few small parameters we conclude they were locally non null not to fit a global behaviour of the data, but a transient one. We have a strong evidence that those localized humps were filled to fit some noise.

2.2.0.4 Second Example

What happens if there are not spurious regressors? Does Lasso try to drop the useful ones? The answer is no. In case all parameters are useful, Lasso selects all of them and reduces to the LS solution. An example can be given using the same method of the previous simulation with a few changes.

- Define four functions $\beta_j(t), t \in [0,1],\ j = 0, 1, 2, 3,$ as:

$$\begin{cases} \beta_0 = 30\,t\,(1-t)^{3/2} \\ \beta_1 = 10\,(t-0.6)^2 + 1 \\ \beta_2 = \mathrm{Sin}\,(4\,\pi\,t) - 2 \\ \beta_3 = \mathrm{Cos}\,(4\,\pi\,t + 0.5) + 1. \end{cases} \tag{2.14}$$

- Generate 30 functional response variables depending on three regressors,

$$y_i(t) = \beta_0(t) + \beta_1(t)\,X_{i,1} + \beta_2(t)\,X_{i,2} + \beta_3(t)\,X_{i,3} + \varepsilon_i(t) \quad i = 1...30 \tag{2.15}$$

- Reduce functions $y_i(t)$ to numerical observations $\mathbf{Y}_i$ by sampling, then standardize $\mathbf{Y}_i$ and $X_{i,j}$ regressors.

- Apply the Functional Lasso technique to the data set $(\mathbf{Y}_i, X_{i,1}, X_{i,2}, X_{i,3})$ for $i = 1, \ldots, 30$. Observe this time we have exactly the same regressors we used in the model. If the technique performs well it has to recognize that all regressors are useful and rebuild the parameters $\beta_0(t), \ldots, \beta_3(t)$ as best as possible.

The cross validation error is represented in Figure 2.7. It is monotonically increasing as λ increases and the minimum is at $\lambda = 0$. Then, the solution reduces to least squares. $\beta_j(t)$ shapes are correctly estimated, as can be seen in Figure 2.8. Their differences in scale are a consequence of standardization. This result has occurred repeatedly in our experiments, so we conjecture the result holds in general and spell it as a rule.

Rule 3. Reduction to LS. In case there will be no regressors to drop Lasso method will choose, as best λ, the value $\bar{\lambda}=0$ and FL solution will reduce to the LS solution.

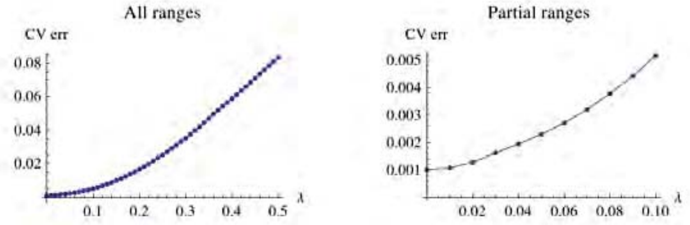


Figure 2.7. Cross validation error for the data set with 3 active regressors. The error is monotone increasing and has minimum in $\lambda=0$.

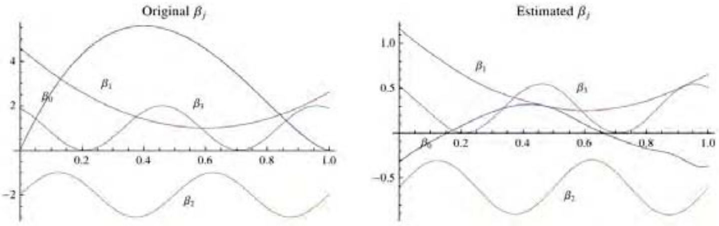


Figure 2.8. On the left there are original $\beta_j(t)$ parameters. On the right their estimation $\hat{\beta}_j(t)$ on standardized data.

2.3 Simulation Study

To consolidate our belief that the Shaken Functional Lasso is performing well in selecting variables we carried out a set of more than forty simulations. Each simulation ends with a graphical report similar to Fig.2.5 which needs to be scrutinized by the statistician. Sometimes it is immediate to understand which regressors to select, one needs to apply only **Rule 1**. In such cases the regular Functional Lasso solves the problem. In other occasions the choice is more complex because Rule 1 is violated in some subset of the domain. In such cases, **Rule 2** is helpful in forming a conclusive decision, in other words here we apply the Shaken Lasso. Indeed, Rule 1 looks only at the magnitude of the estimated $\hat{\beta}_j(t)$ while Rule 2 takes into account the fluctuations of $\hat{\beta}_j(t)$ on changing the value of the penalization term λ.

In Functional Regression, there are many details that could in principle influence the final estimated $\hat{\beta}_j$. The basis functions, the number of basis functions, for BSplines, the placement of knots, the stochastic nature of the error term and its variance, the number of regressors, the number of active regressors, the number of observations, the cross validation scheme, ... A choice must be made. We choose to maintain the simulation scheme as in Section 2.2.0.3, from there a few characteristic values are modified, in particular: the error nature, the error variance, the number of active regressors and the values of $\beta_j(t)$. The following list details which changes are made respect to the procedure decribed previously at points **s1-s11**.

m1. (change respect to previous point **s1**) The errors are not anymore fixed White Noise on Normal(0, 0.08). We keep the White Noise structure but the independent variable can be N(0, $\sigma = 0.5$), N(0,1), N(0,2), N(0,3), Student t_6 or t_4. With this we will see if increasing the noise and the frequency of extreme values affect the variable selection procedure.

m2. (change to **s1**) The number of total regressors entering the regression is kept fix to 6 but, the number of good/active regressors can be one of $\{0, 2, 5, 6\}$. With this variations we establish if the method performs well where (1) there is not a good variable to select, it is all noise (2) some variables are good, some are not, the most frequent scenario (3) only one variable can fit noise (4) all variables are good, in principle there is nothing to shrink.

m3. (change in **s2**) When the good regressors are 2, they can be the ones defined in **s2** or also these

$$\begin{cases} \beta_0 = \mathrm{Sin}(2\,\pi\,t) \\ \beta_1 = 3\,t + \mathrm{Sin}(4\,\pi\,t + 1) \\ \beta_2 = \mathrm{Sin}(4\,\pi\,t) - 2 \end{cases} .$$

m4. When there are 5 good regressors then $\{\beta_0,\ \beta_1,\ \beta_2\}$ are defined as in **s1**, the others are

$$\begin{cases} \beta_3 = \mathrm{Sin}(2\,\pi\,t) \\ \beta_4 = 3\,t + \mathrm{Sin}(4\,\pi\,t + 1) \\ \beta_5 = 10\,|t - 0.5| - 3 \end{cases} .$$

m5. When there are 6 good regressors then all betas are equal to the one defined in **m4** and more, $\beta_6 = 4\,\mathrm{Sinc}(8\,\pi\,(t - 0.5))$.

2.3.0.5 Simulations Diagram

All simulations are placed in the Appendix with a table summarizing the steps peformed for the analysis, an useful overview of all of them can be seen here in Figure 2.9. Each simulation has an identifier code that is an integer number between one and sixty, there are in total 41 simulations. The leaves of the three in Figure 2.9 are the simulations codes, the names in the middle of the trees describe the characteristic of each simulation. The first level of the tree contains the codes “GR:0/6”, “GR:2/6”, etc. they mean that on a total of 6 regressors, zero are good/active, 2 are active, etc. The following level describes the error type which is always White Noise but can be made with Normal$(0, \sigma)$ or Student t_ν i.i.d random variables. Then, if necessary the value of σ is specified. Finally, the code “$\beta 2$” means that instead of using $\beta_j(t)$ as defined in Section 2.2.0.3 point **s2**, we are using the ones defined in Section 2.3 point **m3**. If two experiments are build with the same charateristic values then they differ in the random seed. For example experiments 14 and 18 differ only in the random seed which, as it can be seen in the page of each experiment report in the Appendix, is 123 for case 14, and 1231 for case 18.

For example, in simulation coded “22” there are 0 active regressors and the error is White Noise with $N(0, \sigma = 0.5)$. Simulation number “19” is made with 2 active regressors, the error is White Noise with $N(0, \sigma = 2)$ and the beta set is the one defined in **m3**.

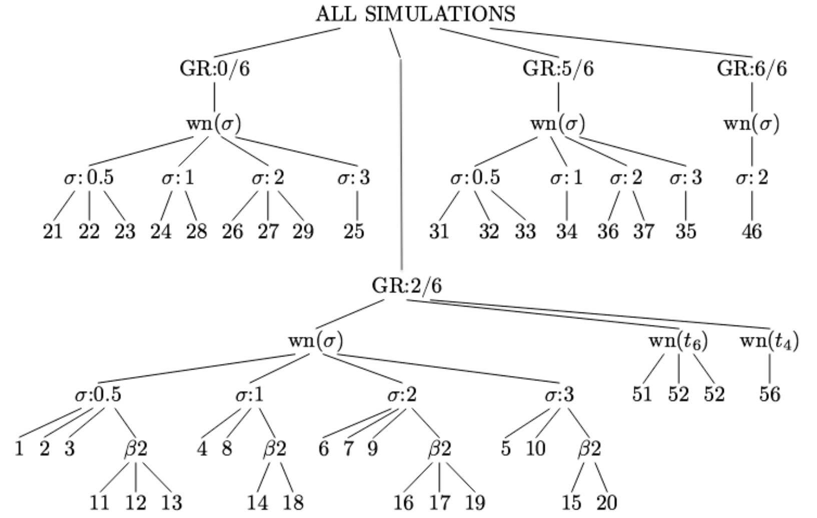


Figure 2.9. Diagram of all the simulations. “GR:x/6” is a shortcut for x-active regressors on a total of 6 regressors. “wn(σ)” stands for Gaussian white noise $N(0, \sigma)$, with σ to be specified in the lower level of the tree. “wn(t_ν)” stands for White Noise with iid Student t_ν random variables. “$\beta 2$” means the true values of $\beta_j(t)$ are the one defined in Section 2.3 at point **m3**.

2.3.0.6 Simulations Analysis

The simulation produced four kinds of output. They are marked in different color in Figure.2.10. In green there are all cases in which the regular Functional Lasso was enough to decide if all regressors are active or not. In another way, we can say these cases can be decided using only Rule 1. For example, **case 2**, $\hat{\beta}_3$, ..., $\hat{\beta}_6$ as shrunk fully to zero, and $\hat{\beta}_1$, $\hat{\beta}_2$ are always outside the zero barrier $[-0.01, 0.01]$, in all the domain, we conclude $\boldsymbol{X}_3, ...,$ $\boldsymbol{X}_6$ are non active without difficulty. **Case 1** is a bit more difficult, this time $\hat{\beta}_3, ...,$ $\hat{\beta}_6$ are not complete zeros as before, but they are inside the zero barrier in all the domain so, by Rule 1, they are null.

In yellow are marked all cases in which Rule 1 was not enough, some parameter $\hat{\beta}_j$ was always near to zero, but escaping the $[-0.01, 0.01]$ barrier in some part of the domain. In these cases we shake the Lasso and apply Rule 2, we move the parameter λ and observe what happens to the estimated $\hat{\beta}_j$. If the parameters $\hat{\beta}_j$ that we are doubting to set to zero fluctuate visibly more than the others then we conclude they are inactive parameters which we can definitely set to zero. For example, **case 20**, here $\hat{\beta}_3$, ..., $\hat{\beta}_6$ are inside $[-0.01,\ 0.01]$ most of the time but not always, we suspect they are associated to inactive regressors. To prove it, we shake the λ and see that only these parameters fluctuate and they tend to enter $[-0.01, 0.01]$ barrier so, we conclude they are indeed associated to inactive regressors and they are simply fitting noise. Observe that, in each case study, on the top of the page it is specified how λ was pushed, in this case we moved from 0.05 to 0.08 by 0.01 steps.

In orange are marked two cases in which it is quite hard to doubt Lasso found a good solution, the only remark here is that the cross validation error is non smooth. In these occasions is better to change the cross validation train/test set ratio in order to obtain continuity. It was not done because we wanted to keep the analysis procedure uniform over all the experiments.

Finally, in red there is only one element, **case 35**, we suspect $\hat{\beta}_6$ could be zero and we apply Rule 2. We see that increasing λ, $\hat{\beta}_6$ tends to enter the $[-0.01, 0.01]$ barrier but there is a problem, also $\hat{\beta}_3$, $\hat{\beta}_2$, $\hat{\beta}_5$, are starting to fluctuate and these are very far from being null parameters. We could say that the relative variation of $\hat{\beta}_6$ is far stronger than in the other parameters, in any case, this experiment result is less clean than the others and can not be considered solved after Rule 2.

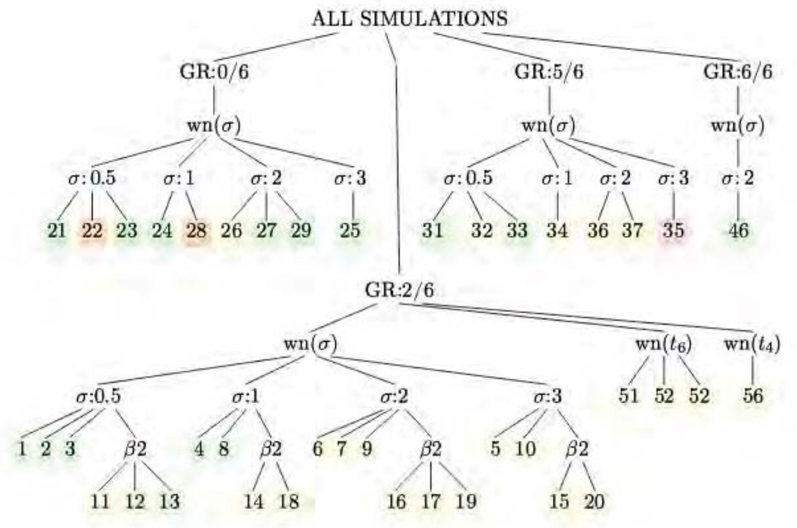


Figure 2.10. In green the cases that could be solved applying only Rule 1, that is simply Lasso. In yellow the cases that to be decided needed the use of Rule 2, that is the Shaked Lasso, in orange, cases with non smooth cross validation error, in red finally a more difficult case.

Observing Figure 2.10 we see that the Shaked Lasso helped resolving more then half of all the experiments which were closed with some degree of ambiguity by the Lasso.

2.4 Case Study: Low Speed Car Accidents and Whiplash Injury

In this section we are going to use the Shaken Functional Lasso to model the velocity of an impacted car in a low speed accident. This study is part of a larger project for the understanding and control of whiplash injury risk.

Whiplash injury is very common, its incidence is about 4 per 1000 persons. It happens when sudden acceleration-deceleration forces are applied to the neck and the upper trunk. The term “whiplash” was introduced in 1928. Before, the injury was referred to as “railway spine” since the most frequent cause of it were train accidents. Nowadays, the most frequent cause are car accidents. Victims are usually sitting in a car standing still when another car hits it in the back. Whiplash injuries are usually not life threatening but they are common, expensive and can give long term consequences. It has been estimated that the U.S. annual economic cost related to whiplash is $3.9 billion, including medical care, sick leave and lost work productivity. Taking into account also litigation costs the number rises to $29 billion [Eck and Hodges, 2001].

Whiplash risk is correlated with the impacted car speed variation and its average acceleration [Kraft et al., 2011]. In the following we will try to predict the impacted car speed function $v(t)$. Half of the data base we are using is publicly available, in raw form, at *AGU*[2.7] *(Arbeitsgruppe für Unfallmechanik)*. The other half comes from proprietary *AXA*[2.8] documentation. *AGU* data contains high frequency speed and acceleration measurements for a set of more than a hundred car accidents. For each car in each crash we extracted some car characteristics from AXA documentation resources as car weight, length, etc. A considerable amount of work was needed to get the speed functions since they were originally just pictures in *pdf* files.

From all the car accidents we selected a set of 25 that are particularly homogeneous. In each selected accident there are two cars, A and B. Car B is initially standing still. Car A is initially traveling at some known constant low[2.9] speed until it hits car B in its back. Cars A and B are perfectly aligned: from the top view their symmetry axes lay on the same line. Car B does not have the rear hook. For each car we have available the following variables: initial speed (vi), weight (wei) , length (len), width (wid), height (hei), and we know the speed, as a function of time, of car B after it has been hit ($v^{(B)}(t)$). Our aim is to model $v^{(B)}(t)$ for the first 0.2 seconds after the impact.

The problem can be seen as functional linear regression. The response variable $v^{(B)}(t)$ is functional and $\{vi, wei, len, wid, hei\}$ are scalar regressors. Instead of using directly these regressors, mechanic considerations suggest we use their standardized differences {AviS, ΔweiS, ΔlenS, ΔwidS, ΔheiS}. For example, ΔweiS is the standardized vector of differences in weight between car B and A, ΔlenS is the standardized vector of length differences and so on for all other variables. The only exception is AviS, since Bvi is always zero, we only standardized car A speeds. Correlations between regressors are shown in Table 2.1.

[2.7] http://www.agu.ch

[2.8] http://www.axa.es

[2.9] Low speed here means a speed inferior to 30 Km/h.

	AviS	ΔweiS	ΔheiS	ΔwidS	ΔlenS
AviS	1.00	0.61	0.16	0.61	0.44
ΔweiS	0.61	1.00	0.16	0.87	0.81
ΔheiS	0.16	0.16	1.00	0.05	-0.23
ΔwidS	0.61	0.87	0.05	1.00	0.84
ΔlenS	0.44	0.81	-0.23	0.84	1.00

Table 2.1. Regressor correlations for the car speeds problem.

Each response variables $v_i^{(B)}(t)$ is originally represented as a set of $(x,\ y)$ coordinates of varying length. We rescale the x coordinate to $[0,1]$ interval and standardize respect to y. This means we standardize a curve speed value respect to all 25 curve speed values. Then, we approximate each curve points with a BSpline function minimizing the least square error. In Figure 2.11, plate (a) are represented the original car accelerations. In plate (b) car velocities. Finally, in plate (c) the standardized BSplines smoothed velocities we will use in our functional regression. The BSpline basis is the same used in the simulated data example, order 3 with equally spaced knots sequence: $\mathcal{K} = \{0, 0, 0, 0, 0.1, 0.2, ..., 0.8, 0.9, 1, 1, 1, 1\}$. The basis is chosen for its simplicity.

Model 0. Using the procedure illustrated in the previous section we find the best parameter estimation $\hat{\beta}_0(t), ..., \hat{\beta}_5(t)$ for the linear model

$$\begin{aligned} v_i(t) \;=\; & \beta_0(t) + \beta_1(t)\,\text{AviS} + \beta_2(t)\,\Delta\text{weiS} + \beta_3(t)\,\Delta\text{lenS} + \\ & \beta_4(t)\,\Delta\text{widS} + \beta_5(t)\,\Delta\text{heiS} + \varepsilon_i(t)\,. \end{aligned} \tag{2.16}$$

Compare two solutions, the ordinary one given by least squares (LS) with the one provided by functional Lasso (FL). All solutions are obtained working with an 11-out cross validation sample, almost half of the set. The first 11 curves of the set are left out and considered test set, see Figure 2.12 for an illustration. The first FL solution we obtain is not practically useful but interesting. Looking at Figure 2.13 we see that there is a minimum in the cross validation error ($\lambda \approx 0.82$) but globally that minimum gives a very small gain respect to larger values of λ. So, the prediction error is not notably small compared to the one of a trivial model containing only $\beta_0(t)$. In Figure 2.14 we can see parameters estimated by LS in dashed red and FL in solid green. LS provided three large and fluctuating estimates, and two small ones (ΔheiS and AviS). On the contrary, FL sets to zero all parameters except ΔwidS. FL solution is already better respect to LS because it is more compact, only one regressor has been selected and the cross validation error is smaller. This solution is not very informative because the cross validation error is very near to the one at $\lambda \to \infty$ and last, but not least, the only variable selected is ΔwidS, this clashes with our physical intuition.

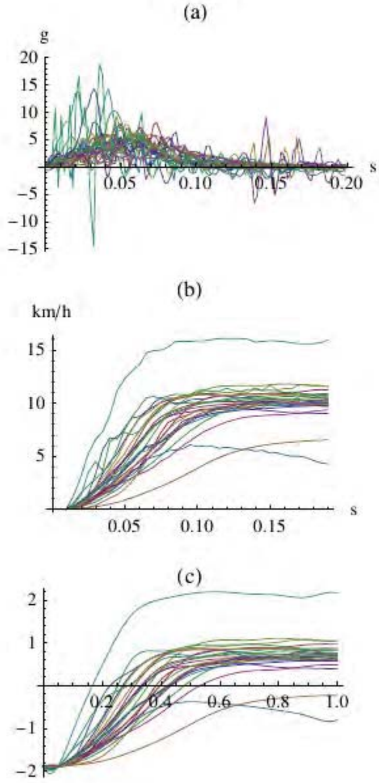


Figure 2.11. Acceleration and speed curves for car B, the struck car. Plate (a) for accelerations, (b) for velocities and (c) for standardized and smoothed velocities.

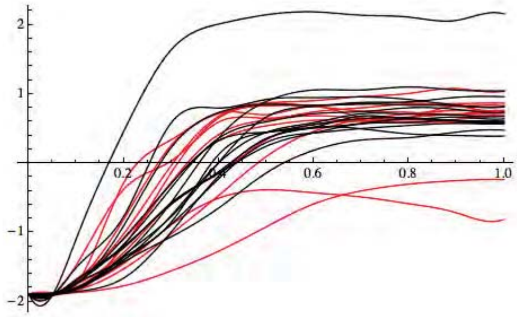


Figure 2.12. Cross validation "in" and "out" of the sample curves. Black curves are used to estimate model parameters, red curves as the cross validation test set. The outlier is "in", it is in the train set, it is used to estimate parameters.

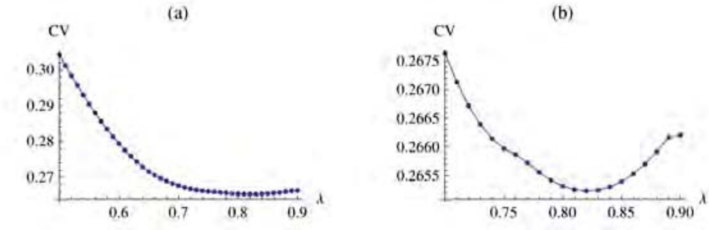


Figure 2.13. Parameters of car speed problem with var $\Delta\, w\, i\, d\, S$ estimated by LS and FL. FL (green curves) annihilates all regressors estimated by LS (red curves) excepted $\Delta\, w\, i\, d\, S$ that is severely shrunk.

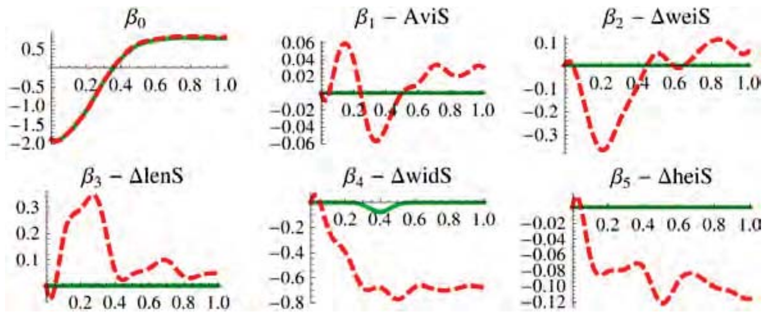


Figure 2.14. Parameters of car speed problem with var $\Delta\, w\, i\, d\, S$ estimated by LS and FL. FL (green curves) annihilates all regressors estimated by LS (red curves) excepted $\Delta\, w\, i\, d\, S$ that is severely shrunk.

Model 1. Looking at Table 2.1 it is easy to see what happened, ΔwidS is highly correlated with ΔweiS and ΔlenS. The weight, an expected dominant variable in every dynamics problem has been shaded by another, linearly correlated but much humbler. We prefer *weight* to be in our model respect to *width*, so we annihilate ΔwidS setting a constraint in the optimization phase: $\sum_{k=0}^{12} |b_{5,k}| \leq 10^{-7}$. Then, we estimate again the LS and FL parameters. From Figure 2.15 we observe at first how the cross validation error minimum is now one order of magnitude deeper. The CV minimum at $\lambda \approx 0.50$ is better than LS ($\lambda = 0$) and also better than the trivial model ($\lambda \to \infty$). Next, observing Figure 2.16 we see that FL has dropped two variables, { ΔheiS, ΔlenS } and shrunk the other two, {AviS, ΔweiS }. We can conclude that FL solution is better than LS because it is simpler (it has fewer regressors) and has stronger predictive power (smaller cross validation error). We may suspect also AviS should be considered zero because it is zero in most of the domain, to check this assumption we apply Rule 2, we Shake the Lasso solution. The results can be seen in Fig.2.17, which shows all the history of the parameters when moving λ from zero to $\lambda \approx 0.5$. It is evident how $\bar{\beta}_3$ and $\bar{\beta}_5$ change shape very fast and shrink to zero immediately where instead $\bar\beta_1$ and $\bar\beta_2$ oppose resistance to the shrinking and tend to maintain their shape, indeed the gray lines are so near they seem a continuous surface. In conclusion, we exclude that β_1 and β_2 could be zero.

Figure 2.15. Cross validation error for car accident problem, functional lasso model without $\Delta\, wid\, S$ regressor.

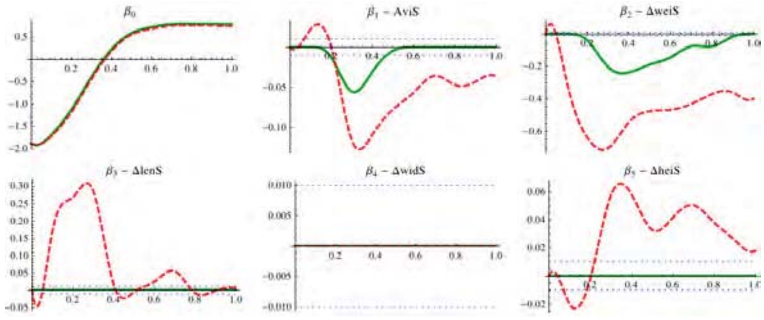


Figure 2.16. Estimated parameters for car accident problem without regressor ΔwidS. Red dashed curves are parameters estimated with least squares, green ones are estimated with functional lasso.

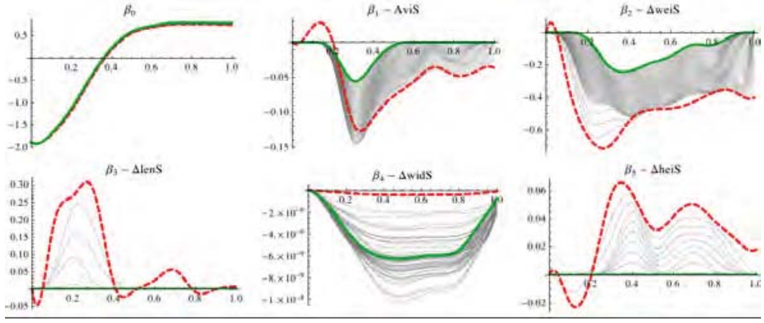


Figure 2.17. Parameters shape and size changes increasing λ from 0 to 0.5 for the car accident problem without variable width. Color codes as used in the previous plots.

We can conclude that only two variables can't be discarded if we take into account all the data we have accumulated, the speed and weight of the two cars. This is in agreement with the classical mechanics notion of momentum and the law of conservation of mechanical energy. The estimated parameters in Figure 2.16 display also that car A velocity has an influence in time only between 0.2 and 0.4 (standardized time units) while Δwei is influential for a much larger time period. The sign of AviS should not cause confusion, if that variable would have been standardized as all others we would have had std(Bvi − Avi) = std(−Avi) = −AviS so the function would have appeared reversed and ultimately, more appealing to intuition. The large outlying value is for a crash where car A was 650 Kg heavier than car B. Consider an average a car in Spain weights approximately 1250 Kg, a figure computed on 1200 whiplash accident cases closed by AXA in 2011. The two small outliers have different characterizations. The one in which $v_B(t)$ decays in the right part of the plot is for two cars with very similar weight (Δwei$=21\,Kg$), the other is a case in which car B is far heavier than car A (588 Kg more). This confirms (or implies) the parameters analysis in Figure 2.16, ΔweiS has a durable impact, in time, on $v_B(t)$.

2.5 Conclusions

In this chapter we have presented a new method for variable selection in functional regression with functional response and scalar regressors. The method is an extension *post hoc* to the well known Lasso technique to the functional case. We applied the SFL (Shaken Functional Lasso) to artificial datasets as well as to a new real dataset. The results are very promising. On the artificial data sets the SFL procedure closed all doubts about which regressors should be dropped, FL alone gave a clean answer only to about 37% of the cases. The functional parameters estimated with FL have often isolated non null humps that prevent to discard the parameter at first sight, SFL helps in deciding which humps are fitting real data, and which are filled with noise. Two phenomenological rules are defined to help a general process of variable selection, *Rule of magnitude*, which is basically the standard Lasso decision formula and *Rule of inertia*, that is the Shaken Lasso step.

As a real data set benchmark we studied low speed car accidents, a frequent whiplash injury cause. We related the speed function of the struck car to the initial difference in speed between the two cars, their weight, their height, their width and length. Studying a set of 25 accidents we can conclude that the only two variables can be considered significant, the weight difference and the pre-impact speed difference. The weight difference has a more durable and more incisive effect respect to the speed difference in determining the stroke car speed function. The preponderance of the mass respect to the speed can be maybe explained by the fact that we are observing only low speed car accidents.

The choice of BSpline basis gives many benefits. In functional data analysis we suppose our data consists of noisy samples from some underlying, inaccessible functions. With BSplines we can roughly control these functions variability, in their domain, while defining the knots sequence. They allow the description of non periodic functions. BSpline coefficients approximate the fitted function values and this permitted us to approximate each $||\beta_j(t)||_1$ with $\sum_k b_{j,k}$ that is, to solve a Lasso on functional objects by a Lasso on scalars without sampling. Lastly, BSpline basis functions are not orthogonal, this at first seems a negative characteristic but it is what gives our estimators interpretability. Indeed, if many coefficients of a functional object go to zero, then they tend to pull to zero all their neighbors, that is all other coefficients of the same object.

Chapter 3

A Random Walk Test for Functional Time Series

3.1 Introduction

The transposition of Time Series techniques to the Functional Data context is motivated by the same arguments as FDA in general: continuous nature of the data, dimension reduction and, it is hoped, improvement in the predictive power of the model. Anyway, in Time Series, FDA can bring some notable simplification in the models, we will show it with an example. Suppose you are going to model the daily electricity consumption in a county, the classic way to do it is to join a deterministic trend with a more or less complex stochastic model, for example an ARIMA. One of the main difficulties is that there are many important periodic factors to take into account occurring in a year, for example all weekends but also special occasions as Christmas, Easter and summer holidays. Every one of these special events has to be separately taken into account and this make the model complex and loaded with exceptions. One of the *desiderata* in using a Functional Model is that it could be able to cope automatically with many of the recurring periodic events. Indeed, if we model electricity consumption as a function that cover one full year, the year-2 will take into account at least what happened in the previous year, year-1, where all of the holidays and weekends were already present so, in principle, we would not need to adjust for many of the seasonalities. One exception to this scheme are moving holidays, like Easter, that would need still manual correction.

The last ten years have seen a lot of advances in Functional Time Series and Functional Autoregressive processes, both in theory and in applications. For example, [Horváth et al., 2010] proposed a method to check if the model can be considered constant in time, [Battaglia, 2005] proposed a method to identify outliers, [Kokoszka and Reimherr, 2013] a method to establish the order of an autoregressive process. In applications, [Damon and Guillas, 2002] used functional autoregressive process for Ozone forecasting, [Besse et al., 2000] to forecast ocean temperatures, [Guillas et al., 2011] to forecast the seabed evolution and maintain navigability channels. As of today, [Bosq, 2000] is the *de facto* reference for the theoretical aspects of Functional Autoregressive Processes while the recent [Horváth and Kokoszka, 2012] collects many recent results and is directed to the researcher as well as to the practitioner in FDA.

In this chapter, we present a Random Walk test for Functional Autoregressive Processes. As always in Functional Data, the first impulse is to start by looking at what was done in the past to solve the unfunctional problem. We have seen in Section.1.3.2 some methods to test the Random Walk hypothesis, but all of them, in one way or another, try to estimate the value ρ in $X_{n+1} = \rho\, X_n + \varepsilon_{n+1}$. We choose not to follow this direct approach in the functional context because here the ruling equation is $X_{n+1}(t) = \Psi\, X_n(t) + \varepsilon_{n+1}(t)$ and the available estimator for Ψ converges very slowly and in an unexpected[3.1] way, see [Horváth and Kokoszka, 2012, pg.240], [Kokoszka and Zhang, 2010].

In the functional context, the model corresponding to the AR(1) seen in sec.1.3.1 is Functional Autoregressive Process of order one, denoted for short as FAR(1) and defined by

$$X_{n+1}(t) = \Psi(X_n)(t) + \epsilon_{n+1}(t), \tag{3.1}$$

where X_i have mean zero. In the general setting of [Bosq, 2000], $X_i(t)$ are functions in an Hilbert space, Ψ is a bounded linear operator from H to H and $\epsilon_n(t)$ are H-white noise. In this work we will restrict our attention to the framework most used in applications, as in [Horváth and Kokoszka, 2012]. H will be $L^2[0,1]$, the space of functions in $[0,1]$ which are square integrable according to Lebesgue. The scalar product is the common $<f, g> := \int_0^1 f\,g$. $\epsilon_i(t)$ are i.i.d. with $E(\epsilon_i(t)) = 0$ and $E\,(||\,\epsilon_i(t)||^2) < \infty$ for all i. For Ψ, we consider only integral operators of type

$$\Psi(f)(t) = \int_0^1 \psi(t, s)\, f(s)\, d\, s, \tag{3.2}$$

which are always linear and bounded if the kernel function $\psi(t,\, s)$ is continuous. It might be useful to consider this operator as an integral average of f respect to a set of functions ψ_t. These operators are not too restrictive on what we can express, on the contrary, they are quite general. Indeed,

[3.1] The estimator depends on principal components but increasing the number of principal components reduces the performance of the estimator.

quoting from [Gohberg et al., 1990, ch.7], they are like an *universal model* for Hilbert-Schmidt[3.2] operators because for each operator $A\colon H \rightarrow H$ there exists a unitary operator U such that $U\,A\,U^{-1}$ is an integral operator as Eq.3.2.

We want to check if the dataset $X_1, ..., X_n$, which we suppose was generated by a process as Eq.3.1, can be considered a Random Walk. The idea is to compare the covariance of the original data set $\{X_i\}_i$ with the covariance of the same dataset resampled under the null hypothesis that Ψ is the Identity operator.

In details, we start with a functional data set $X_1(t)$, ..., $X_n(t)$ and compute its first $p \leq n - 1$ *Empirical Functional Principal Components*. The EFPC are eigenfunctions and eigenvalues of the empirical covariance operator and they will be denoted with with $(\hat{\xi}_i,\ \hat{\lambda}_i)_{i=1..p}$, see [Ramsay and Silverman, 2005, ch.8]. It is known that, under mild conditions, when n goes to infinity, $\hat{\lambda}_i$ converges to λ_i, the eigenvalues of the populational covariance operator, see [Horváth and Kokoszka, 2012, pg.31], [Bosq, 2000, sec.4.2], [Dauxois et al., 1982].

The Schmidt Norm of an Hilbert-Schmidt operator A can be computed as

$$||\,A||_S^2 = \sum_{j=1}^{\infty} ||\,A\,\phi_j||^2\,, \tag{3.3}$$

where $\phi_1, \phi_2, ...$ is an orthonormal basis, see [Gohberg et al., 1990, pg.141-143]. In our case A is the covariance operator K_Ψ and choosing as orthonormal basis its eigenfunctions given by PCA: $\xi_1, \xi_2, ...$, we get

$$||\,K_\Psi||_S^2 = \sum_{j=1}^{\infty} ||\,K_\Psi\,\xi_j||^2 = \sum_{j=1}^{\infty} ||\,\lambda_i\,\xi_j||^2 = \sum_{i=1}^{\infty} \lambda_i^2. \tag{3.4}$$

In applications we will use $\hat{\lambda}_i$ as estimator for λ_i so we will actually compute $\widehat{||\,K_\Psi||}_S^2$ but, using the aforementioned result of convergence and the continuity of the norm, for large values of n, $\widehat{||\,K_\Psi||}_S^2$ converges to $||\,K_\Psi||_S^2$.

Under the null hypothesis that the FAR(1) process is a Random Walk we can estimate its innovations as

$$\hat{\epsilon}_{n+1}(t) := X_{n+1}(t) - X_n(t). \tag{3.5}$$

[3.2] An operator A on an Hilbert space is an Hilbert-Schmidt operator if it is linear, continuous and such that $\sum_{i=1}^{\infty} ||\,A\,\phi_i||^2 < \infty$ for an orthonormal basis $\phi_1, \phi_2, \ldots$. There are also other equivalent definitions, see [Gohberg et al., 1990], pg.140.

Resampling $\hat{\epsilon}_n(t)$ and applying again the null hypothesis H_0 we can compute B Bootstrap copies of the observations set $\{X_i(t)\}_i$ as

$$X_{n+1}^{b,*}(t) := X_n^{b,*}(t) + \epsilon_{n+1}^{b,*}(t), \qquad \text{for } b \text{ in } 1 \ldots B. \tag{3.6}$$

For each set of resampled observations $\{X_i^{b,*}(t)\}_i$ we compute the estimated Schmidt norm of its covariance $(\widehat{||K_{Id}||}_S^{b,*})^2$ through principal components as done before. Then, build its empirical distribution $\mathcal{E}$. Finally, we compute $\widehat{||K_\Psi||}_S^2$ for the original data set and reject the null hypothesis that Ψ is Id if $\mathcal{E}(\widehat{||K_\Psi||}_S^2)$ is smaller than some predefined threshold α.

From the Bootstrap structure and the statistic involved it is apparent the nature of the test is

$$\begin{cases} H_0\colon \Psi = Id \\ H_1\colon ||\Psi|| < 1. \end{cases} \tag{3.7}$$

Indeed, if Ψ has norm one then there will be no reduction on the impact of all old innovations $\epsilon_{j \leq i+1}$ on X_i, so X_i could grow very fast and consequently also the covariance matrix and its eigenvalues, just as in the case of $\Psi = Id$. This condition makes Ψ and Id indistinguishable to our test.

Further in this chapter, Section 2 contains a detailed description of how to apply the test in practice. In Section 3 the Montecarlo simulation scheme, Section 4 a brief guide on how to read the simulation results and an analysis of them. Section 5 shows two applications to real datasets, we check for Random Walk the series of yearly electricity consumption in France and diary prices of Bitcoin. It also clarifies the procedure given in section 2 and can be red just after it by the reader most interested in the test deployment. In Section 6 there are the conclusions and finally in the Appendix are collected the simulation results.

3.2 Test Procedure on a Real Dataset

Suppose our dataset comes as a matrix $M_{n,m}$ where data to be considered functions are the columns $c_1, \ldots, c_m$. We take for granted here that data are already aligned and equispaced, techniques to make a dataset in this form are discussed in [Ramsay and Silverman, 2005].

Before starting the Random Walk test, by our understanding of the real phenomenon underlying the data, we check if there is a structure that can be considered deterministic and remove it applying a proper transformation. Bosq suggests to remove trend but not seasonality, if the seasonality component can be modeled through a FAR(1), see [Bosq, 2000, pg.152, 240].

Perhaps, to appreciate the importance of the previous step, the following imaginary experiment can be of help. Suppose, you are studying the daily phone calls functions in the Italian mobile telephone network. It is well known to all Italian people, that on New Year's Eve night the mobile network collapses, and it becomes difficult to send the important "Happy new year!" messages. This will happen every year, it is a deterministic structure and you need to remove it because the FAR(1) model has no way to get it straight since it looks only at what happened on the night before, that is on the not so special night of December 30. On the contrary, suppose now you are studying the yearly phone calls functions. In this case you should not, in principle, adjust for New Year's Eve because the previous year also contains it.

To apply the Random Walk test the following steps must be performed.

1. Smooth the dataset fitting each column $c_1 \ldots c_m$ to a base of your preference. We used BSpline and Fourier. The number of basis functions is largely to be selected depending on the nature of the problem but a good starting point could be $\sqrt{n}$. At the end of this step we will have the set of functional observations $X_1(t), \ldots, X_m(t)$.

2. Choose a number of principal components you want to use for your analysis. We suggest to start with $p = 3$ and adjust it *ex post* if necessary. There are classical ways of selecting the number of principal components based for example on the *scree plot* or on the *explained variance* but, according to our simulations, the power of the test is concentrated only on the first eigenvalue. Using $p > 1$ is instrumental in showing if the sequence of eigenvalues displays a continuous decay, which is characteristic of ordinary FAR(1) processes. Or a sudden extreme drop after the first eigenvalue, which is a characteristic of a Random Walk. A graphical representation of this phenomenon is presented in the Applications section.

3. Find the Empirical Functional Principal Components of your dataset and their associated eigenvalues $\hat{\lambda}_1 \ldots \hat{\lambda}_p$, then compute the estimated (squared) Schmidt Norm $\widehat{||K_\Psi||}_S^2 \leftarrow \sum_{i=1}^{p} \hat{\lambda}_i^2$. It is important to observe that we compute the EFPC always on the centered dataset, in this case it is $\{\tilde{X}_i | \tilde{X}_i = X_i - \bar{X} \text{ for } i = 1 \ldots r\}$. If you use *R* with *fda* package for your computations, this can be achieved automatically setting the parameter `centerfns=TRUE` in function `pca.fd` .

4. Resample the dataset under the null hypothesis H_0 that $\Psi = \mathrm{Id}$. First

compute the estimated innovations,

$$\hat{\epsilon}_{i+1}(t) \leftarrow X_{i+1}(t) - X_i(t) \qquad \text{for } i=2\dots m$$

Center the $\hat{\epsilon}_{i+1}(t)$ subtracting their common mean and find the resampled observations,

$$\begin{cases} X_1^{b,*}(t) \leftarrow \text{Mean}(X_1(t),...,X_m(t)) \\ X_{i+1}^{b,*}(t) \leftarrow X_i^{b,*}(t) + \hat{\epsilon}_{i+1}^{b,*}(t) & \text{for } i=2\dots m \end{cases}$$

Compute the associated Schmidt norm $(\widehat{||K_\Psi||}_s^{b,*})^2$ as in step (3). Some observations about the choice of the mean as first bootstrapped observations will be given at the end of the procedure.

5. Repeat step (4) for the necessary amount of iterations until you get B estimations of $(\widehat{||K_\Psi||}_s^{b,*})^2$. We usually start with $B=200$. Find the empirical distributions $\mathcal{E}$ of all the $(\widehat{||K_\Psi||}_s^{b,*})^2$. Set p-value$\leftarrow$ $\mathcal{E}(\widehat{||K_\Psi||}_s^2)$. Traditionally reject H_0 if p-value <0.05, or some other threshold of your choice.

6. Toggle the number of basis functions, principal components and Bootstrap replications to make sure results are stable.

The initial bootstrapped observation $X_1^{*,b}$ is a free variable in this problem because we can estimate only $m-1$ innovations. We choose to set it to $\bar{X}$ following this heuristic. If we want the Bootstrap to replicate something we have to give it a chance to do it right, the best chance to start seems to be the middle of the original dataset. Other possibilities are reasonable but were not tested, for example one could start the simulation picking randomly one X_i, or the median. A graphical comparison of the initial dataset and some Bootstrap replications can give some hints about the suitability of the selected starting point criterion.

3.3 Simulation Methodology

In this section we will describe in detail the simulation procedure. It has been implemented in the programming language *R* and it uses the external package *fda* which is presented in [Ramsay et al., 2009]. Steps in which *fda* function library is central will be denoted with *(fda)*.

There are a number of global numerical parameters that control the simulation, they are summarized in the next table for ease of reference.

n.obs	Number of functional observations. $X_1(t) \dots X_{\boldsymbol{n.obs}}(t)$.
n.pt	Number of raw (x, y) coordinates describing each function $X_i(t)$.
n.basis	Number of basis functions used to represent each $X_i(t)$.
n.pc	Number of functional principal components to consider.
n.boot	Number of bootstrap replications.
sd	Raw indicator of error function magnitude. It appears in the simulation of *Brownian Motion* and *Brownian Bridge*.

3.3.1 Simulate and Test a FAR(1) process

1. Define an operator Ψ of type (3.2) providing a kernel function $\psi(s,t)$.

2. Multiply the kernel function by a constant C to be able to see how the test performs with kernels that have same algebraic structure but different size.

3. Compute constant $\hat{C}_{Id}$ such that the following relation[3.3] is satisfied

$$|| \, \Psi ||_S^2 = \int\int_U \hat{C}_{Id}^2 \, \psi^2(t,s) \, \mathrm{dt} \, \mathrm{ds} = 1. \tag{3.8}$$

4. Set the random number generator to a fixed value to provide reproducible results.

5. Define a vector of constants that will multiply C_{Id} and vary the kernel size.

$$\mathrm{kset} \leftarrow (0.5, 0.7, 0.8, 0.9, 0.95, 0.99, 1.0) \tag{3.9}$$

6. For each value K in kset repeat what follows 100 times and store the final result.

   a. Create an initial FAR(1) data set with $X_0(t) = f_0(t)$ and $X_{i+1}(t) = \Psi(X_i(t)) + \epsilon_{i+1}(t)$. f_0 is set initially to $f_0(t) = (x^2 + 1) + S\,i\,n\,(8\,\pi\,x)$. There are no special reasons to choose this function, but error functions parameters were selected to have a reasonable order of magnitude compared to it, the precise *signal to noise ratio* in general depends on applications. The error functions $\epsilon_i(t)$ can be independent trajectories of *Brownian Bridge* (BB) or *Brownian Motion* (BM). They are generated from a cumulative sum of 100 Normal independent random variables with $\mu = 0$ and σ to be set to **sd**. A detailed discussion can be found in [Iacus, 2009, sec.1.6, 1.8]. We use ten different kernel functions $\psi(s,\, t)$, some of

[3.3] It holds that, $\|\Psi\|_S^2 = \int \int_U \psi^2(s,t) \, \mathrm{ds}\mathrm{dt}$, see [Gohberg et al., 1990, pg.143].

them appear in literature like the the *Gaussian*, *Wiener* and *Parabolic kernels* in [Gabrys et al., 2010].

b. (fda) Following a common rule of thumb we create a BSpline or Fourier functions basis with number of elements $\boldsymbol{n.basis} \leftarrow \lfloor \sqrt{\mathbf{n.pt}} \rfloor$ and fit all the $X_i(t)$ to the new functions space.

c. (fda) Perform a principal component analysis on the functional data set, extract the eigenvalues $\hat{\lambda}_1 \ldots \hat{\lambda}_p$ corresponding to the first **n.pc** empirical principal components and set $\widehat{||K_\Psi||}_S^2 \leftarrow \sum_{i=1}^{\boldsymbol{n.pc}} \hat{\lambda}_i^2$.

d. Compute the estimated residuals $\hat{\epsilon}_i(t)$ under H_0 and center them subtracting their common mean,

$$\hat{\epsilon}_{i+1}(t) \leftarrow X_{i+1}(t) - X_i(t). \tag{3.10}$$

e. Create **n.boot** copies of the data data set $\{X_i(t)\}_i$, each time resampling the residuals with the following rule

$$X_{i+1}^{b,*}(t) \leftarrow X_i^{b,*}(t) + \hat{\epsilon}_{i+1}^{b,*}(t). \tag{3.11}$$

The parameter b is the bootstrap index and varies in $1, \ldots, \mathbf{n.boot}$.

f. (fda) For each family of bootstrapped observations compute the first **n.pc** empirical principal components, with their respective eigenvalues and set $\left(\widehat{\|K_{\mathrm{Id}}\|}_S^{b,*}\right)^2 \leftarrow \sum_{i=1}^{p} (\hat{\lambda}_i^{b,*})^2$.

g. Build the empirical distribution function of the $\left(\widehat{\|K_{\mathrm{Id}}\|}_S^{b,*}\right)^2$ values and name it $\mathcal{E}$.

h. The p-value associated to the current experiment will be $\mathcal{E}(\widehat{||K_\Psi||}_S^2)$. We reject H_0 that is, we reject that Ψ is the Identity operator if p-value<0.05.

7. In the results tables we read the rejection rate for each K, the number of experiments in which H_0 was rejected divided by 100.

3.4 Simulations Analysis and Results

All experiments results are in tabular format and all tables can be found in the Appendix. The table following this paragraph (Table 4.1) is a reference to all of the simulation results. In the first column it tells the *code* of the experiment, a unique identifier by which it is possible to find the appropriate table in the Appendix. In the second, it tells if Ψ was a constant, if not, it reports the formula for its kernel $\psi(s,t)$. "N" tells the number of Montecarlo replications used to estimate the power of the test and *base* specifies if we are using the BSpline or Fourier basis. Columns **n.boot**, **n.pc** and **sd** have the same meaning previously defined.

code	Ψ	N	**n.boot**	**n.pc**	base	**sd**
b1	constant	100	200	1	S	0.5
b2	constant	100	200	1	S	0.05
b4	constant	100	1000	1	S	0.5
b5	constant	100	1000	1	S	0.05
b7	constant	100	200	3	S	0.5
b8	constant	100	200	3	S	0.05
b10	constant	100	200	1	F	0.5
b11	constant	100	200	1	F	0.05
k1b1	$e^{-(s^2+t^2)}$	100	200	1	S	0.5
k1b2	$e^{-(s^2+t^2)}$	100	200	1	S	0.05
k1b7	$e^{-(s^2+t^2)}$	100	200	3	S	0.5
k2b1	$e^{(s^2+t^2)}$	100	200	1	S	0.5
k2b2	$e^{(s^2+t^2)}$	100	200	1	S	0.05
k3b1	$min(s,t)$	100	200	1	S	0.5
k3b2	$min(s,t)$	100	200	1	S	0.05
k4b1	$(t-\frac{1}{2})^2+(s-\frac{1}{2})^2$	100	200	1	S	0.5
k4b2	$(t-\frac{1}{2})^2+(s-\frac{1}{2})^2$	100	200	1	S	0.05
k5b1	$(t+\frac{1}{2})^2+(s+\frac{1}{2})^2$	100	200	1	S	0.5
k5b2	$(t+\frac{1}{2})^2+(s+\frac{1}{2})^2$	100	200	1	S	0.05
k6b1	$Sin(2\pi t+s)$	100	200	1	S	0.5
k7b1	$Sin(2\pi s+t)$	100	200	1	S	0.5
k8b1	$Sin(2\pi s)\,Sin(2\pi t)$	100	200	1	S	0.5
k9b1	$\lvert Sin(2\pi s)\,Sin(2\pi t)\rvert$	100	200	1	S	0.5
k10b1	$Sin(8\pi s)\,Sin(8\pi t)$	100	200	1	S	0.5

Table 3.1. Experiments reference.

Suppose we are interested in simulations with a *Gaussian Kernel*, so choosing, for example, the experiment *k1b2*, here is what we will find in the Appendix. In the first column it is stated if error is of type *Brownian Motion* or *Brownian Bridge*. In the second the sample size, that is the number of simulated functional observations X_1 ... X_n entering the test. The third column tells C_{id} was multiplied by 0.5, and the remaining tell C_{id} was multiplied by $0.7, 0.8$, etc.

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	0.98	0.86	0.75	0.64
	100	1	1	1	1	0.9	0.64
	200	1	1	1	1	1	0.75
	500	1	1	1	1	1	0.87
bb	50	1	0.96	0.9	0.64	0.36	0.2
	100	1	1	1	0.92	0.68	0.33
	200	1	1	1	1	0.95	0.36
	500	1	1	1	1	1	0.67

Table 3.2. Tabular output corresponding to experiment **k1b2**.

It will be noticed that in this table there is not $K = 1.00$ multiplying C_{Id}, the last value is 0.99. This was made on purpose to stress that when $||\Psi|| = 1$ ($||\Psi||_S$ is the best estimator of $||\Psi||$ we have) our test has no chances to determine if $\Psi = Id$.

3.4.1 Analysis

There are very different kinds of operators in the simulation from which different results are expected. Experiments with code **b1**, **b2**, ..., **b11** all have a constant operator and are a special case to see how the algorithm performs when H_0 is true, when $\Psi = \text{Id}$. Operator kernels in experiments **k1X**, **k2X**, **k3X**, **k4X**, **k9X** are all positive and symmetric. The remaining kernels in experiments **k6X**, **k7X**, **k8X**, **k10X** are not positive on the whole domain $[0,1] \times [0,1]$.

We observe that the power of the test is much higher when there are non positive kernels. In all corresponding tables the power of the test is very high, even with a small sample size. This fact is easy to explain intuitively, a fixed sign kernel is, generally speaking, more similar to the identity operator than a non fixed sign operator.

All experiments with constant operator and positive definite kernels display a similar triangular structure in the *power* distribution. Power of the test increases increasing the sample size, and decreases when the kernel Schmidt norm goes near to one. The performance increasing with the sample size is what we expect from every statistical test, in general, more information is available, more the decision task is simplified. The identity operator has uniform[3.4] norm one. The Schmidt norm is an estimator from the top of the uniform norm. When the Schmidt norm of the kernel is forced by mul-

[3.4] The uniform norm for an operator A is defined as $||\,A||_L := \sup_{||x|| \leq 1} ||\,A\,x||$, it is proved that $||\,A||_L \leq ||\,A||_S$, that is, the Schmidt norm dominates the uniform norm, see [Bosq, 2000] pg.34-35 tiplicative constant to go near one then the operator is forced to become, from the point of view of the norm, similar to the identity. Consequently, becoming the operator more similar to Id, it becomes more difficult for the test to discriminate between them.

There is an operator that is positive definite but gets *power* much higher then the other ones. It is the *Wiener* kernel, **k3X**. Its power is as high as the one associated to non-positive kernels. Probably the reason for this anomaly is that the kernel $\psi(s,t)$ of the Wiener process is just the covariance of Brownian Motion, which is the model for our innovations.

Changing the error term from Brownian Bridge to Brownian motion had little influence on the power. Also changing the size of the error, from $\mathrm{sd} = 0.5$ to $\mathrm{sd} = 0.05$ had not visible impact.

Comparing **k1b7-k1b1** and **b1-b8**, we do not see an important change in performance changing the number of principal components. Comparing **b1-b10** and **b2-b11**, we do not observe important changes in power passing to Fourier basis. Comparing **b1-b4** and **b2-b5** again display not important changes, so also increasing the number of bootstrap resamples does not seem to increase much the power of the test.

3.5 Applications

3.5.1 Electrical energy consumption in France

We analyze France electrical energy consumption from the beginning of 1996 to the end of 2012. The dataset is available from *RTE France* one the Web[3.5]. As stated by the provider, data covers power consumption in metropolitan France area, except Corsica. It includes losses on the network but it does not take into account power withdrawn by hydroelectric installations. Paper by [Cho et al., 2013] uses our same dataset and provides supplementary informations about energy consumption and factors influencing it.

Electricity consumption is observed every 30 minutes. Following [Cho et al., 2013] we study the series of weekly average consumption. We apply a logarithm transformation to cope with the apparent increase in variance and remove the trend which was estimated with a LOESS using *R* default parameters. The original dataset and its appearance after each

[3.5] `http://clients.rte-france.com`

transformation is shown in Fig.3.1

After the transformations we are working with a 52x16 matrix, on columns we have years, on rows weeks. Each column of the matrix is transformed (smoothed) into a functional object in $[0,1]$ using a BSpline basis with $\lfloor\sqrt{52}\rfloor$ basis functions, knots are distributed equidistantly between zero and one, included. Applying our test for unit root with **n.pc** $\leftarrow$3, and **n.boot** $\leftarrow$200 we get p-value zero. The estimated $\widehat{||K_\Psi||}^2_S$ is $8.0 \cdot 10^{-7}$ and the quartiles for $\left(\widehat{\|K_{\mathrm{Id}}\|}^{b,*}_S\right)^2$ are $1.7 \cdot 10^{-6}$, $1.6 \cdot 10^{-5}$, $2.7 \cdot 10^{-5}$, $5.3 \cdot 10^{-5}$, $1.09 \cdot 10^{-3}$. Changing the number of principal components, doubling the number of basis functions and doubling the bootstrap replications did not affect the result significantly. The two plots in Fig.3.2 show that, besides the Schmidt norm, there is a significant difference in the structure of the estimated eigenvalues of K_Ψ and K_{Id}. The first ones decrease smoothly in magnitudes. For the random walk instead the first eigenvalues dominates all the others.

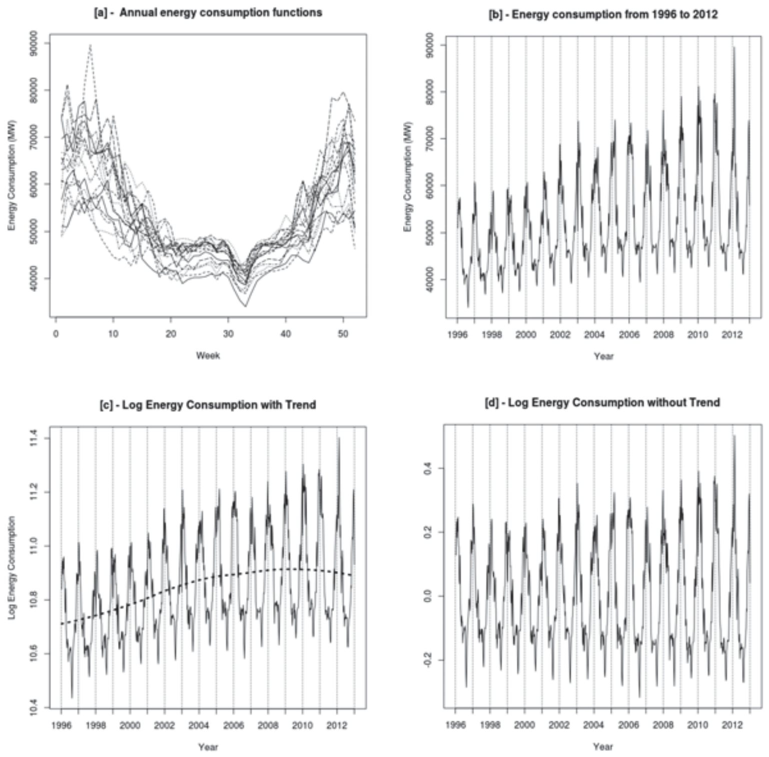


Figure 3.1. These plots represent electrical power consumption in France. [a] All yearly consumption's are plotted together. [b] Historic plot of energy consumption, from 1996 to 2012. [c] Log is applied to previous plot with LOESS trend function superimposed. [d] The trend is removed from the previous.

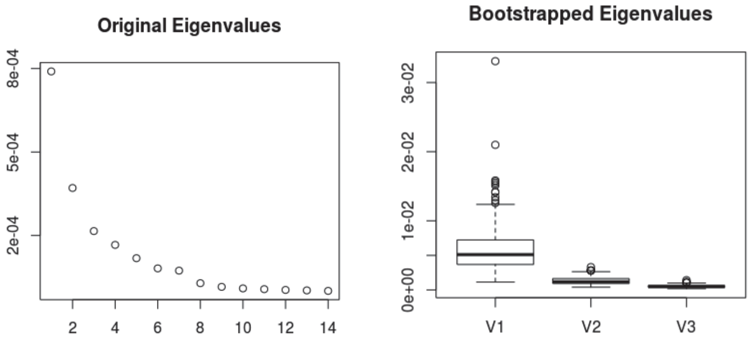


Figure 3.2. The two plots show the estimated eigenvalues of K_Ψ on the right, and the bootstrapped estimated eigenvalues of K_{Id} on the left.

3.5.2 Bitcoin daily prices

Bitcoin is a virtual currency introduced by [Nakamoto, 2008]. Bitcoins are traded twenty-four hours per day, all days of the year, prices are known to have large variability and suffered a burst on beginning of 2014 after a period of explosive growth. In the recent paper by [Kristoufek, 2013], it was shown that the series of average daily prices between 1-May-2011 and 30-June-2013 is non stationary. The paper provides some introductory information about the Bitcoin currency to which the interested reader may refer.

We follow [Kristoufek, 2013] using the same dataset[3.6] and the same temporal window for our investigations. But, instead of using daily average prices, we will consider the much more detailed series of daily prices, where each day is seen as a functional observation $X_i(t)$.

The Bitcoin prices we have are the ones processed at *Mt.Gox*, once the largest currency trading center. They are available for free on the Internet. The number of trades in *Mt.Gox* during the considered time period is extremely variable. For six days there were no transactions at all, these days were removed from our analysis. Excluding zeros, the minimum number of transactions per day was 373, the maximum 66293, the other deciles: 2042, 2722, 3448, 4255, 5000, 5994, 7322, 9483, 13736.

The series of trade prices was passed to logarithm and detrended. Fig.3.3 illustrates the dataset at each step. Then, the dataset was divided in day blocks. For each daily data, a linear interpolating function with domain in [0, 1] was built. Each function was sampled in 2000 equidistant points

[3.6] File `mtgoxUSD.csv` at `http://api.bitcoincharts.com/v1/csv/`

from zero to one creating a matrix of 2000x786 elements where each column corresponds to a day. The data matrix was converted to a set of functional observations respect to a BSpline basis with $\lfloor\sqrt{2000}\rfloor$ basis function, knots were equispaced between zero and one. We applied our test for unit root setting **n.pc** ←3 and **n.boot** ←200. The resulting p-value was 0.63, the estimated $\widehat{||K_\Psi||}_S^2$ is 0.43 and the quartiles for $\left(\widehat{\|K_{\mathrm{Id}}\|}_S^{b,*}\right)^2$ are 0.011, 0.14, 0.30, 0.62, 14.29. The null hypothesis can not be rejected. Moreover, comparing the structure of eigenvalues in Fig.3.4 we see they are similar, the first eigenvalue dominates all the others and also, the magnitude of the first eigenvalue for K_Ψ is approximately the median of the bootstrapped eigenvalues of K_{Id}. Increasing the number of principal components and doubling the number of basis functions did not affect the p-value significantly.

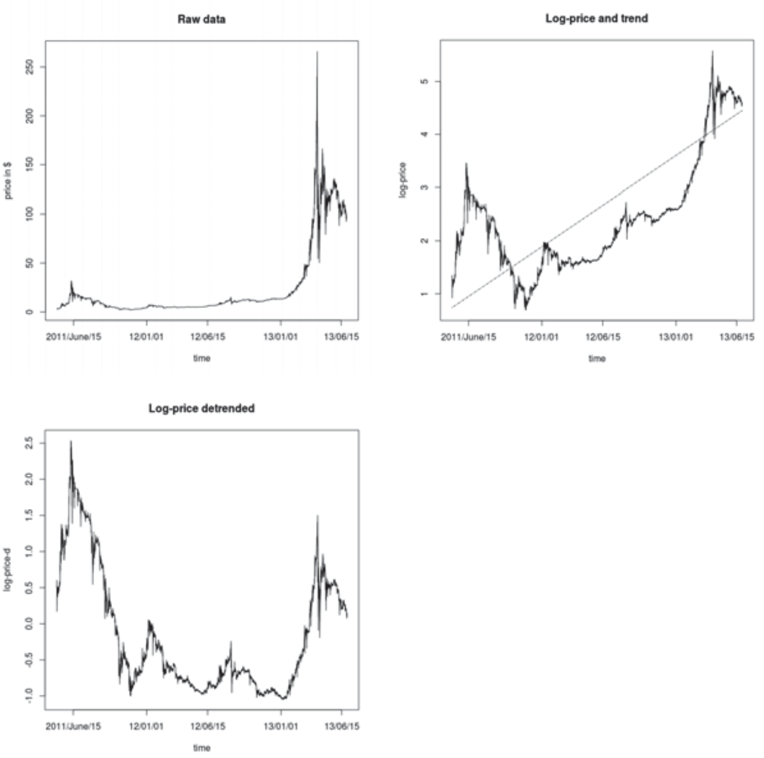


Figure 3.3. The original dataset is represented in the top left pane, Bitcoin trading price in U.S. dollars. On the top-right, the series after applying the logarithm with a dashed line superimposed for the trend. On the bottom, the the data after logarithm transform and removal of the trend.

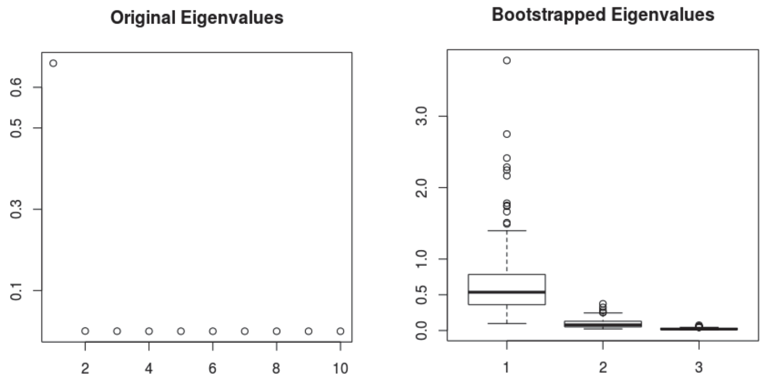


Figure 3.4. Plot of K_Ψ eigenvalues and the boxplots of bootstrapped K_{Id} eigenvalues.

3.6 Conclusions

Using $\widehat{||K_\Psi||}_S^2$ as test statistic associated with our Bootstrap scheme has given very good results in the numerical simulations. Comparing our power results to the ones previously found for AR(1) in [Ferretti and Romo, 1996] we see they are surprisingly high. The comparison is not completely correct because we are working on a different framework but there is no other Random Walk test on FAR(1) against which we could compare.

It has emerged from our investigations that a Random Walk tends to have a first large eigenvalue that dominates all the following ones. Indeed, this feature can be of help in deciding if to reject the null hypothesis in cases in which the p-value would be near to the rejection region. Or also, in cases in which the p-value alone would give a result in sharp contrast with our intuition about the problem.

Applying the test to the two real data sets has given the results we were expecting from visual inspection. Yearly innovations in France electrical energy consumption can not be considered a Random Walk. On the other side, it is not rejected that Bitcoin daily prices could be a random walk. It was necessary to apply the common tools of time series analysis before entering the test: remove the trend and adjust for variance. Indeed, a FAR(1) process defined according to equations (3.1) and (3.2) has not enough analytical freedom to cope with this conditions which are not of local nature, but global external factors.

As all simulations in functional data, there are many parameters that could be tuned: the choice of the basis, the number of basis functions, the number points in the real data set and their relative distance, the errors, their dependence structure, the data smoothing, the empirical distribution smoothing and so on. We tried to stick to the most widespread choices, the most common errors types and operators, and the most popular initial analysis setups. There are large possibilities for further experimentation and, of course, for a desirable theoretical development in support of the numerical evidence.

Chapter 4
Conclusions and Future Work

In this work we have presented two different methods for the analysis of Functional Data. This first one is variable selection method in Functional Regression, the second is Random Walk test for Functional Autoregressive Processes of order one.

The problem of variable selection in Regression is, and will be, probably for long time a central topic in Statistics. In this work we adapted the celebrated Lasso technique to Functional Regression with functional response and scalar regressors. The computations are reduced to a numerical optimization problem without sampling on functional objects, only by algebraic transformation on the BSpline representation of the functional datum. After the Functional Lasso is applied, it may not be clear if some of regressors have to be dropped. In that case we run a second analysis phase that we call "shake the Lasso", which is based on varying the penalization term λ respect to the optimal value $\bar{\lambda}$ and observe the elasticity of estimated parameters $\beta_i(t)$. The Shaken Lasso is graphical technique, we ran it over more that forty simulated problems and we see it gives consistently good results and improves the decision in almost 40% of the simple Functional Lasso outputs.

The second result we have presented is a Random Walk test for Functional Autoregressive processes of order one. As far as we know, this is the first test of this kind, there is nothing in literature about testing the Random Walk condition for a Functional Autoregressive Process. Our test is based on the Bootstrap, comparing the covariance structure of the data with the one bootstrapped under the null hypothesis. An extensive simulation set has shown the method to be reliable and powerful.

Leaving aside the desirable theoretical grounding for our results and the many possible simulations we could add to the ones we presented, there are many directions in which we could push the research forward in both the Shaken Functional Lasso and the Random Walk test for Functional Autoregressive processes. It is first of all of some interest to establish if we can "shake" with profit also the multivariate Lasso. That is, to establish if the same method we use to check if a parameter is null in the functional case can be used in the unfunctional case. We did some experiments about it but we still are at a preliminary stage.

The Lasso is based on penalized Least Squares solutions, and Least Squares solutions are sensitive to outliers. Is there away to make a Robust Functional Lasso? We have done some preliminary experiments in this sense but still we have not reached a firm method to robustify the Functional Lasso.

About applications, we remember that the car accident dataset we used in Chapter 2 was containing only 25 observations some of which were visibly outliers. The amount of information we had was at the minimum limit to reach some interesting conclusions, taking into account also that a part of the data had to be used for cross validation. It would be interesting to have more observations to perform a confirmatory study on our preliminary results. We think it will be much easier in the future to have the kind of data we used because insurance companies in U.S. and Europe are moving toward having *black box* installed on cars, as it is compulsory today for airplanes.

About the Random Walk test for Functional Autoregressive Processes a natural future work we are considering is to apply the test systematically to stock prices and to other econometry time series. That is, to perform a research similar to [Nelson and Plosser, 1982] but considering modern high frequency data which can be often fruitfully described as functional data. It would be also interesting to extend the Random Walk test to Functional Autoregressive processes of order higher than one.

From a more general and philosophical standpoint, it is opinion of the author that Functional Data Analysis should relay on stronger hypothesis if we want our tests on functional data to be strong as much as their unfunctional correspondent. The current point of view is instead well summarized by [Valderrama, 2007], which I cite: "... *when we concern to functional data analysis (FDA) we mean that there are not any hypothesis on the probability distribution of the stochastic processes underlying the data, but only sample information*". For me this has to change, as the key of the development in unfunctional regression is the reasonable assumption that ε_i are normally distributed, an analogy has to be developed for the functional case. My best guess is that the way to go is to express the error as Stochastic Differential Equation. The commonly used Brownian Motion and Brownian bridge are inadequate to express an error, who would use a Random Walk as an error in the unfunctional case?

In conclusion, If I will have the opportunity to work on functional data in the future I will try to parallel the beautiful theory of unfunctional regression with normal errors in the functional case. Where normal errors need to be substituted with something appropriate. If the hypotheses will be precise and strict, the test output will be clear. If the hypotheses are vague, the test output will be vague. If there are not hypotheses, if there is not a probabilistic framework, then there is nothing to test, and Statistics dissolves into Data Science.

Bibliography

[Adams, 2003] Adams, R. (2003). *Calcolo Differendizle 2*. Casa Editrice Ambrosiana.

[Ash and Gardner, 1975] Ash, R. B. and Gardner, M. F. (1975). *Topics in stochastic processes*, volume 27. Academic Pr.

[Battaglia, 2005] Battaglia, F. (2005). Outliers in functional autoregressive time series. *Statistics & probability letters*, 72(4):323–332.

[Besse et al., 2000] Besse, P. C., Cardot, H., and Stephenson, D. B. (2000). Autoregressive forecasting of some functional climatic variations. *Scandinavian Journal of Statistics*, 27(4):673–687.

[Bosq, 2000] Bosq, D. (2000). *Linear processes in function spaces: theory and applications*, volume 149. Springer.

[Box and Jenkins, 1970] Box, G. E. and Jenkins, G. M. (1970). *Time series analysis: Forecasting and control*. Holden-Day.

[Candes and Tao, 2007] Candes, E. and Tao, T. (2007). The dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351.

[Chatterjee and Hadi, 2006] Chatterjee, S. and Hadi, A. S. (2006). *Regression Analysis by Example*, volume 607. John Wiley & Sons.

[Chen et al., 1998] Chen, S. S., Donoho, D. L., and Saunders, M. A. (1998). Atomic decomposition by basis pursuit. *SIAM journal on scientific computing*, 20(1):33–61.

[Cho et al., 2013] Cho, H., Goude, Y., Brossat, X., and Yao, Q. (2013). Modeling and forecasting daily electricity load curves: a hybrid approach. *Journal of the American Statistical Association*, 108(501):7–21.

[Cuevas, 2014] Cuevas, A. (2014). A partial overview of the theory of statistics with functional data. *Journal of Statistical Planning and Inference*, 147:1–23.

[Damon and Guillas, 2002] Damon, J. and Guillas, S. (2002). The inclusion of exogenous variables in functional autoregressive ozone forecasting. *Environmetrics*, 13(7):759–774.

[Dauxois et al., 1982] Dauxois, J., Pousse, A., and Romain, Y. (1982). Asymptotic theory for the principal component analysis of a vector random function: some applications to statistical inference. *Journal of multivariate analysis*, 12(1):136–154.

[De Boor, 2001] De Boor, C. (2001). *A practical guide to splines*, volume 27. Springer Verlag.

[Dickey and Fuller, 1979] Dickey, D. A. and Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, 74(366a):427–431.

[Dickey et al., 1984] Dickey, D. A., Hasza, D. P., and Fuller, W. A. (1984). Testing for unit roots in seasonal time series. *Journal of the American Statistical Association*, 79(386):355–367.

[Eck and Hodges, 2001] Eck, J. and Hodges, S. (2001). Whiplash: a review of a commonly misunderstood injury. *The American journal of medicine*.

[Efron et al., 2004] Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *The Annals of statistics*, 32(2):407–499.

[Faraway, 1997] Faraway, J. (1997). Regression analysis for a functional response. *Technometrics*.

[Febrero-Bande and Oviedo de la Fuente, 2012] Febrero-Bande, M. and Oviedo de la Fuente, M. (2012). Statistical computing in functional data analysis: the r package fda. usc. *Journal of Statistical Software*, 51(4):1–28.

[Feller, 1957] Feller, W. (1957). *An introduction to probability theory and its applications. Vol. 1 .* John Wiley.

[Ferraty and Romain, 2011] Ferraty, F. and Romain, Y. (2011). *The Oxford handbook of functional data analysis*. OUP Oxford.

[Ferraty and Vieu, 2006] Ferraty, F. and Vieu, P. (2006). *Nonparametric functional data analysis: theory and practice*. Springer Verlag.

[Ferretti and Romo, 1996] Ferretti, N. and Romo, J. (1996). Unit root bootstrap tests for ar (1) models. *Biometrika*, 83(4):849–860.

[Fuller, 1976] Fuller, W. A. (1976). *Introduction to statistical time series*.

[Gabrys et al., 2010] Gabrys, R., Horváth, L., and Kokoszka, P. (2010). Tests for error correlation in the functional linear model. *Journal of the American Statistical Association*, 105(491):1113–1125.

[Gohberg et al., 1990] Gohberg, I., Goldberg, S., and Kaashoek, M. (1990). Classes of linear operators, 1990.

[Grant and Boyd, 2008] Grant, M. and Boyd, S. (2008). *Graph implementations for nonsmooth convex programs*. Lecture Notes in Control and Information Sciences. Springer-Verlag Limited. http://stanford.edu/~boyd/graph_dcp.html.

[Grant and Boyd, 2011] Grant, M. and Boyd, S. (2011). *CVX: Matlab Software for Disciplined Convex Programming, version 1.21*.

[Grenander, 1981] Grenander, U. (1981). *Abstract inference*. Wiley New York.

[Guillas et al., 2011] Guillas, S., Bakare, A., Morley, J., and Simons, R. (2011). Functional autoregressive forecasting of long-term seabed evolution. *Journal of Coastal Conservation*, 15(3):337–351.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., and Tibshirani, R. (2009). *The elements of statistical learning*. Springer.

[Hoerl and Kennard, 1970] Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: applications to nonorthogonal problems. *Technometrics*, 12(1):69–82.

[Hong and Lian, 2011] Hong, Z. and Lian, H. (2011). Inference of genetic networks from time course expression data using functional regression with lasso penalty. *Communications in Statistics - Theory and Methods*, 40(10):1768–1779.

[Horváth et al., 2010] Horváth, L., Hušková, M., and Kokoszka, P. (2010). Testing the stability of the functional autoregressive process. *Journal of Multivariate Analysis*, 101(2):352–367.

[Horváth and Kokoszka, 2012] Horváth, L. and Kokoszka, P. (2012). *Inference for functional data with applications*, volume 200. Springer Science & Business Media.

[Hughes, 1996] Hughes, B. D. (1996). *Random walks and random environments*. Clarendon Press Oxford.

[Iacus, 2009] Iacus, S. M. (2009). *Simulation and inference for stochastic differential equations: with R examples*. Springer.

[Iglesias et al., 2007] Iglesias, A., Ipanaqué, R., and Urbina, R. (2007). Symbolic manipulation of bspline basis functions with mathematica. *Computational Science–ICCS 2007*, pages 194–202.

[Ihaka, 2010] Ihaka, R. (2010). R: Lessons learned, directions for the future. In *Joint Statistical Meetings*. The Authors.

[Ihaka and Lang, 2008] Ihaka, R. and Lang, D. T. (2008). Back to the future: Lisp as a base for a statistical computing system. In *COMPSTAT 2008*, pages 21–33. Springer.

[James et al., 2013] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*. Springer.

[Jolliffe, 2002] Jolliffe, I. (2002). *Principal component analysis*. Wiley Online Library.

[Kokoszka and Reimherr, 2013] Kokoszka, P. and Reimherr, M. (2013). Determining the order of the functional autoregressive model. *Journal of Time Series Analysis*, 34(1):116–129.

[Kokoszka and Zhang, 2010] Kokoszka, P. and Zhang, X. (2010). Improved estimation of the kernel of the functional autoregressive process. Technical report, Technical Report. Utah State University.

[Kraft et al., 2011] Kraft, M., Anders, K., Malm, S., and Ydenius, A. (2011). Influence of Crash Severity on Various Whiplash Injury Symptoms: A Study Based on Real-Life Rear-End Crashes with Recorded Crash Pulses. *Folksam Research and Karolinska Institutet, Sweden*, pages 1–8.

[Kristoufek, 2013] Kristoufek, L. (2013). Bitcoin meets google trends and wikipedia: Quantifying the relationship between phenomena of the internet era. *Scientific reports*, 3.

[López-Pintado and Romo, 2009] López-Pintado, S. and Romo, J. (2009). On the concept of depth for functional data. *Journal of the American Statistical Association*, 104(486):718–734.

[Matsui and Konishi, 2011] Matsui, H. and Konishi, S. (2011). Variable selection for functional regression models via the L1 regularization. *Computational Statistics & Data Analysis*.

[Meese and Rogoff, 1983] Meese, R. A. and Rogoff, K. (1983). Empirical exchange rate models of the seventies: Do they fit out of sample? *Journal of international economics*, 14(1):3–24.

[Meinshausen et al., 2007] Meinshausen, N., Rocha, G., and Yu, B. (2007). Discussion: A tale of three cousins: Lasso, l2boosting and dantzig. *The Annals of Statistics*, pages 2373–2384.

[Nakamoto, 2008] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28.

[Nelson and Plosser, 1982] Nelson, C. R. and Plosser, C. R. (1982). Trends and random walks in macroeconmic time series: some evidence and implications. *Journal of monetary economics*, 10(2):139–162.

[Phillips and Perron, 1988] Phillips, P. C. and Perron, P. (1988). Testing for a unit root in time series regression. *Biometrika*, 75(2):335–346.

[Ramsay et al., 2009] Ramsay, J., Hooker, G., and Graves, S. (2009). *Functional data analysis with R and MATLAB*. Springer Verlag.

[Ramsay and Silverman, 2005] Ramsay, J. and Silverman, B. (2005). *Functional data analysis*. Springer, New York.

[Seber and Lee, 2003] Seber, G. and Lee, A. (2003). *Liner Regression Analysis*. John Wiley & Sons, second edition edition.

[Shumway et al., 2000] Shumway, R. H., Stoffer, D. S., and Stoffer, D. S. (2000). *Time series analysis and its applications*, volume 3. Springer New York.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

[Topi and Tucker, 2014] Topi, H. and Tucker, A., editors (2014). *Computing Handbook. Information Systems and Information Technology*. CRC Press, third edition edition.

[Valderrama, 2007] Valderrama, M. J. (2007). An overview to modelling functional data. *Computational Statistics*, 22(3):331–334.

[Wasserman, 2003] Wasserman, L. (2003). *All of statistics*. Springer New York.

[Weisberg, 1985] Weisberg, S. (1985). *Applied Linear Regression*. Wiley New York, second edition edition.

[Wold, 1975] Wold, H. (1975). Soft modeling by latent variables: the nonlinear iterative partial least squares approach. *Perspectives in probability and statistics, papers in honour of MS Bartlett*, pages 520–540.

[Zhao et al., 2012] Zhao, Y., Ogden, R. T., and Reiss, P. T. (2012). Wavelet-based lasso in functional linear regression. *Journal of Computational and Graphical Statistics*, 21(3):600–617.

[Zou and Hastie, 2005] Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.

Appendix A

A.1 FAR(1) Simulations

b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.91	0.62	0.27	0.18	0.06	0.03
	100	1	1	0.98	0.68	0.3	0.16	0.05
	200	1	1	1	1	0.69	0.14	0.04
	500	1	1	1	1	1	0.35	0.05
bb	50	1	0.97	0.78	0.37	0.17	0.06	0.05
	100	1	1	0.99	0.84	0.51	0.15	0.06
	200	1	1	1	0.99	0.89	0.23	0.06
	500	1	1	1	1	1	0.46	0.03

b2

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.93	0.61	0.3	0.26	0.16	0.11
	100	1	1	0.98	0.71	0.31	0.18	0.08
	200	1	1	1	1	0.63	0.16	0.06
	500	1	1	1	1	1	0.33	0.08
bb	50	1	0.92	0.58	0.31	0.2	0.49	0.34
	100	1	1	1	0.76	0.33	0.24	0.13
	200	1	1	1	1	0.87	0.23	0.13
	500	1	1	1	1	1	0.43	0.05

b4

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.93	0.59	0.22	0.07	0.02	0.05
	100	1	1	0.99	0.68	0.3	0.04	0.09
	200	1	1	1	0.99	0.63	0.15	0.06
	500	1	1	1	1	1	0.29	0.05
bb	50	1	0.98	0.85	0.37	0.28	0.04	0.05
	100	1	1	1	0.86	0.44	0.13	0.06
	200	1	1	1	1	0.88	0.11	0.09
	500	1	1	1	1	1	0.46	0.02

b5

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.87	0.57	0.24	0.12	0.13	0.13
	100	1	1	0.99	0.64	0.3	0.11	0.15
	200	1	1	1	0.99	0.63	0.14	0.08
	500	1	1	1	1	1	0.28	0.06
bb	50	1	0.96	0.62	0.17	0.23	0.34	0.31
	100	1	1	1	0.78	0.37	0.23	0.16
	200	1	1	1	1	0.84	0.25	0.11
	500	1	1	1	1	1	0.44	0.03

b7

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.96	0.73	0.35	0.2	0.06	0.03
	100	1	1	0.98	0.72	0.35	0.17	0.05
	200	1	1	1	1	0.76	0.14	0.04
	500	1	1	1	1	1	0.42	0.06
bb	50	1	0.99	0.83	0.46	0.21	0.07	0.05
	100	1	1	1	0.95	0.56	0.17	0.04
	200	1	1	1	1	0.92	0.25	0.06
	500	1	1	1	1	1	0.51	0.03

b8

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.96	0.69	0.35	0.27	0.16	0.1
	100	1	1	0.99	0.76	0.39	0.2	0.06
	200	1	1	1	1	0.72	0.18	0.07
	500	1	1	1	1	1	0.37	0.08
bb	50	1	0.97	0.75	0.39	0.25	0.53	0.35
	100	1	1	1	0.8	0.41	0.25	0.13
	200	1	1	1	1	0.93	0.23	0.13
	500	1	1	1	1	1	0.52	0.05

b10

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.91	0.64	0.27	0.17	0.06	0.03
	100	1	1	0.98	0.66	0.3	0.16	0.05
	200	1	1	1	1	0.69	0.14	0.04
	500	1	1	1	1	1	0.35	0.05
bb	50	1	0.98	0.78	0.38	0.18	0.09	0.06
	100	1	1	0.99	0.84	0.52	0.15	0.06
	200	1	1	1	0.99	0.89	0.22	0.06
	500	1	1	1	1	1	0.47	0.03

b11

Error	n	0.5	0.7	0.8	0.9	0.95	0.99	1.00
bm	50	1	0.93	0.63	0.31	0.26	0.16	0.11
	100	1	1	0.98	0.71	0.31	0.18	0.09
	200	1	1	1	1	0.62	0.16	0.07
	500	1	1	1	1	1	0.34	0.08
bb	50	1	0.92	0.59	0.31	0.19	0.51	0.36
	100	1	1	1	0.76	0.34	0.24	0.13
	200	1	1	1	1	0.89	0.24	0.13
	500	1	1	1	1	1	0.46	0.04

k1b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	0.88	0.72	0.5
	100	1	1	1	0.99	0.94	0.63
	200	1	1	1	1	1	0.75
	500	1	1	1	1	1	0.89
bb	50	1	0.95	0.9	0.65	0.36	0.19
	100	1	1	1	0.91	0.69	0.33
	200	1	1	1	1	0.95	0.35
	500	1	1	1	1	1	0.66

k1b2

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	0.98	0.86	0.75	0.64
	100	1	1	1	1	0.9	0.64
	200	1	1	1	1	1	0.75
	500	1	1	1	1	1	0.87
bb	50	1	0.96	0.9	0.64	0.36	0.2
	100	1	1	1	0.92	0.68	0.33
	200	1	1	1	1	0.95	0.36
	500	1	1	1	1	1	0.67

k1b7

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	0.92	0.78	0.54
	100	1	1	1	1	0.97	0.7
	200	1	1	1	1	1	0.83
	500	1	1	1	1	1	0.94
bb	50	1	0.96	0.94	0.74	0.43	0.2
	100	1	1	1	0.97	0.74	0.4
	200	1	1	1	1	0.97	0.43
	500	1	1	1	1	1	0.72

k2b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	0.93	0.88	0.59	0.4	0.22
	100	1	1	1	0.88	0.6	0.28
	200	1	1	1	1	0.93	0.42
	500	1	1	1	1	1	0.6
bb	50	1	0.97	0.98	0.83	0.53	0.37
	100	1	1	1	0.99	0.84	0.46
	200	1	1	1	1	0.97	0.55
	500	1	1	1	1	1	0.8

k2b2

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	0.97	0.81	0.57	0.44	0.31
	100	1	1	1	0.86	0.6	0.35
	200	1	1	1	0.99	0.92	0.41
	500	1	1	1	1	1	0.58
bb	50	1	0.98	0.98	0.82	0.53	0.37
	100	1	1	1	0.99	0.83	0.47
	200	1	1	1	1	0.97	0.54
	500	1	1	1	1	1	0.8

k3b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1
bb	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1

k3b2

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1
bb	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1

k4b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	0.9	0.72	0.48
	100	1	1	1	1	0.97	0.66
	200	1	1	1	1	1	0.87
	500	1	1	1	1	1	0.98
bb	50	1	1	1	0.95	0.83	0.69
	100	1	1	1	1	1	0.81
	200	1	1	1	1	1	0.96
	500	1	1	1	1	1	0.99

k4b2

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	0.98	0.87	0.78	0.67
	100	1	1	1	1	0.91	0.63
	200	1	1	1	1	1	0.83
	500	1	1	1	1	1	0.96
bb	50	1	1	1	0.96	0.84	0.68
	100	1	1	1	1	1	0.79
	200	1	1	1	1	1	0.96
	500	1	1	1	1	1	1

k5b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	0.93	0.88	0.6	0.42	0.21
	100	1	1	1	0.87	0.6	0.27
	200	1	1	1	1	0.92	0.41
	500	1	1	1	1	1	0.6
bb	50	1	0.97	0.93	0.73	0.44	0.26
	100	1	1	1	0.97	0.75	0.39
	200	1	1	1	1	0.96	0.47
	500	1	1	1	1	1	0.74

k5b2

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	0.96	0.82	0.56	0.45	0.32
	100	1	1	1	0.84	0.61	0.34
	200	1	1	1	0.99	0.91	0.39
	500	1	1	1	1	1	0.6
bb	50	1	0.97	0.91	0.72	0.44	0.27
	100	1	1	1	0.97	0.78	0.41
	200	1	1	1	1	0.96	0.46
	500	1	1	1	1	1	0.74

k6b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1
bb	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1

k7b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1
bb	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1

k8b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	1	1	0.98
	100	1	1	1	1	1	0.99
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1
bb	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1

k9b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	0.99	0.82	0.67	0.42
	100	1	1	1	0.99	0.91	0.49
	200	1	1	1	1	1	0.66
	500	1	1	1	1	1	0.81
bb	50	1	0.96	0.95	0.75	0.45	0.24
	100	1	1	1	0.97	0.77	0.42
	200	1	1	1	1	0.97	0.46
	500	1	1	1	1	1	0.71

k10b1

Error	n	0.5	0.7	0.8	0.9	0.95	0.99
bm	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1
bb	50	1	1	1	1	1	1
	100	1	1	1	1	1	1
	200	1	1	1	1	1	1
	500	1	1	1	1	1	1

Appendix B

B.1 Details about the Confidence Band Plot

Given three functions with domain in $[0, 1]$

$$\begin{cases} \beta_0(t) = 30\, t\, (1-t)^{3/2} \\ \beta_1(t) = 3 \sin(t) \\ \beta_2(t) = 5 \cos(t) \end{cases}, \tag{B.1}$$

we define a set of 30 observations as $Y_i(t) = \beta_0(t) + \beta_1(t)\, X_{i,1} + \beta_2(t)\, X_{i,2} + \varepsilon_i(t)$, where X_i are random vectors of length 30 such that $X_{i,j} \sim \text{Uniform}(-1, 1)$ and every $\varepsilon_i(t)$ is the linear interpolation of a sequence of 100 iid random variables distributed as a Normal$(0, 0.5)$. After that, knowing $Y_i(t), X_1, X_2$ and another spurious regressors X_3 build as the previous X_i, we fit by Least Squares the linear model

$$Y_i(t) = \beta_0(t) + \beta_1(t)\, X_{i,1} + \beta_2(t)\, X_{i,2} + \beta_3(t)\, X_{i,3}\ ,$$

and find the estimated $\{\hat{\beta}_0(t), \hat{\beta}_1(t), \hat{\beta}_2(t), \hat{\beta}_3(t)\}$. We expect to find that $\hat{\beta}_i(t) \approx \beta_i(t)$ for $i = 0...2$ and $\hat{\beta}_3(t) \approx 0$. We are interested in proving that $\beta_3(t) = 0$. In Figure B.1 we can see all the $\hat{\beta}_i(t)$, $\beta_3(t)$ is the smallest green line. In Figure B.2 we can see 200 bootstrap replications $\hat{\beta}_3^*(t)$ obtained resampling the residuals $\varepsilon_i(t)$. Figure B.3 is a obtained by selecting the set D of the 90% most deep curves from Figure B.2 and then drawing the MinMax band, which is, for every t_i, the interval $[\min(f(t_i)), \max(f(t_i))]_{f \in D}$.

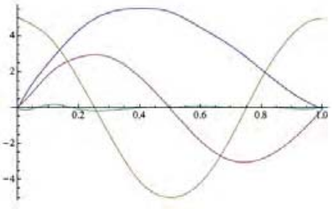


Figure B.1.

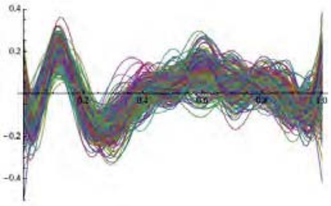


Figure B.2.

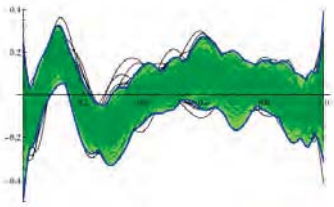


Figure B.3.

B.2 Shaken Lasso Simulations

The experimental results about the Shaken Lasso are collected in a high definition color “pdf” file which is available at: https://db.tt/0FK2bnVW . In some versions of this document the results are included in a final addendum at the very end.

All experiment results are displayed one per page in a standardized format. For each experiment, on the top of each page are reported the number of regressors that have to be recognized as useful, the error type and its magnitude, the random seed and the minimum found for the cross validation error. When Rule 2 is used, it is said how much λ has been moved from the minimum position.

The following table collects some comments for the interpretation of each experiment. At the beginning the comments will be extensive, then they will become more terse since there are only four possible kinds of outputs and interpretations.

Case	Comments
1	**Solved with Rule 1**. Functional Lasso is enough to decide which regressors to choose. The estimated parameters $\beta_3, ..., \beta_6$ correspond to regressors to discard. There are two estimations, Least Squares in dashed red lines and Lasso in black. Lasso estimators for $\beta_3, ...,$ β_6 are always inside the $[-0.01,\ 0.01]$ barrier so we can conclude they must be considered null. We observe also that β_1 and β_2 shape correspond exactly to initial parameters in the first plot on the top left part.
2	**Solved with Rule 1**. in this case Lasso estimators for $\beta_3, ..., \beta_6$ are shrunk very near to zero, well inside the $[-0.01, 0.01]$ barrier.
3	Solved with Rule 1.
4	Solved with Rule 1.
5	**Solved with Rule 1 and Rule 2**. In this case Rule 1 is sufficient to decide β_5 is null, but not sufficient to decide with certainty if $\{\beta_3,\ \beta_4,\ \beta_6\}$ are also null. Indeed, Lasso estimators for regressors $\beta_3, \beta_4, \beta_5$ escape the $[-0.01, 0.01]$ barrier. In this case we shake the Lasso solution increasing a bit the penalization term λ, from 0.07 to $\{0.08, 0.09, 0.1\}$. For each of these new values of λ there are new estimators for each of the β_i. These new estimators are drawn in dashed black lines and they are visible only if they do not overlap exactly to previous estimators. We observe that dashed black lines are visible only in $\beta_3, \beta_4, \beta_6$ plots and also, that the dashed lines tend to enter, the $[-0.01, 0.01]$ barrier. We conclude that these betas are to be considered null because on increasing a bit the penalization term, all the effect concentrate on their annihilation leaving the other parameters unchanged.
6	Solved with Rule 1 and Rule 2.
7	Solved with Rule 1 and Rule 2.
8	Solved with Rule 1.
9	Solved with Rule 1 and Rule 2.
10	Solved with Rule 1 and Rule 2.
11	Solved with Rule 1 and Rule 2.
12	Solved with Rule 1 and Rule 2.
13	Solved with Rule 1.
14	Solved with Rule 1 and Rule 2.
15	Solved with Rule 1 and Rule 2.
16	Solved with Rule 1 and Rule 2.
17	Solved with Rule 1 and Rule 2.
18	Solved with Rule 1 and Rule 2.
19	Solved with Rule 1 and Rule 2.

20 Solved with Rule 1 and Rule 2.

21 Solved with Rule 1.

22 Solved with Rule 1. In this case the cross validation error is not smooth. In a single analysis situation it would be desirable to tune the cross validation train and test set and the λ values grid in such a way that the cross validation error has a minimum on a smooth function.

23 Solved with Rule 1.

24 Solved with Rule 1 and Rule 2. In this case the use of Rule 2 is really only a reinforcement, Rule 1 alone gave quite a direct answer and β_6 simply touched the $[-0.01, 0.01]$ barrier.

25 Solved with Rule 1 and Rule 2.

26 Solved with Rule 1 and Rule 2.

27 Solved with Rule 1 and Rule 2.

28 Solved with Rule 1. see comments on case 22.

29 Solved with Rule 1. see comments on case 22.

31 Solved with Rule 1.

32 Solved with Rule 1 and Rule 2.

33 Solved with Rule 1.

34 Solved with Rule 1 and Rule 2.

35 **Not solved by Rule 1 and Rule 2**. Applying Rule 1 we think only β_6 could be null. We apply Rule 2, β_6 shrinks but even large parameters do so. The problem is most apparent in β_3, that parameter is large, very far from the [−0.01, 0.01] barrier, it is not expected to be null so it is not expected to shrink. Instead we can see the dashed lines in it. Moreover, the dashed lines are visible also in β_2 and β_3. We conclude the method is not helpful in this case because we expect by Rule 1 that only parameter β_6 could be zero, so we expect that on increasing the penalization term λ a bit the shrinking should affect only β_6. This does not happen.

36 Solved with Rule 1 and Rule 2.

37 Solved with Rule 1 and Rule 2.

46 Solved with Rule 1. No regressors is to be dropped, β_i are all very far from the the $[-0.01, 0.01]$ rejection barrier.

51 Solved with Rule 1 and Rule 2.

52 Solved with Rule 1 and Rule 2.

53 Solved with Rule 1 and Rule 2.

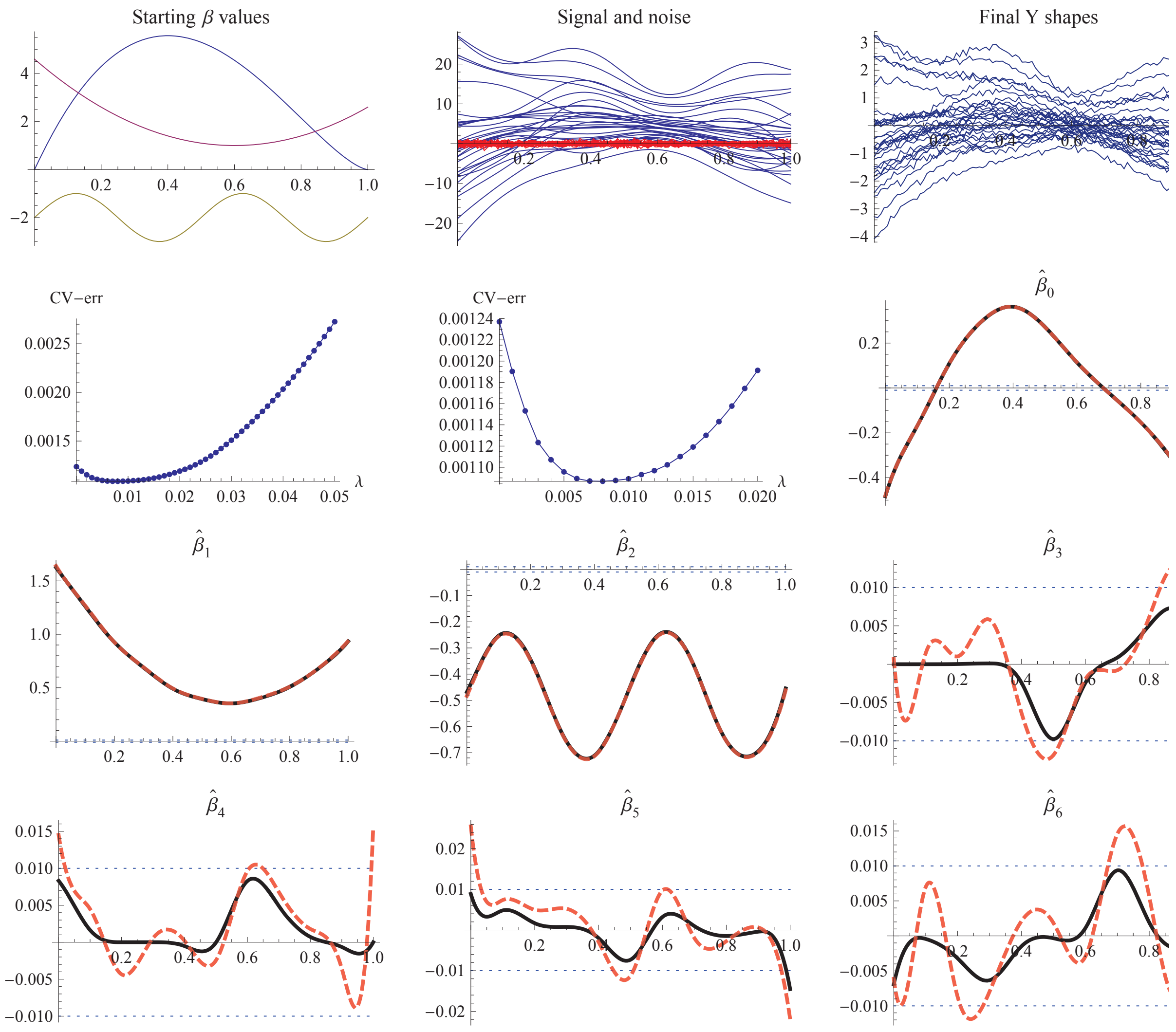
56 Solved with Rule 1 and Rule 2.

Addendum

Shaken Lasso Simulations

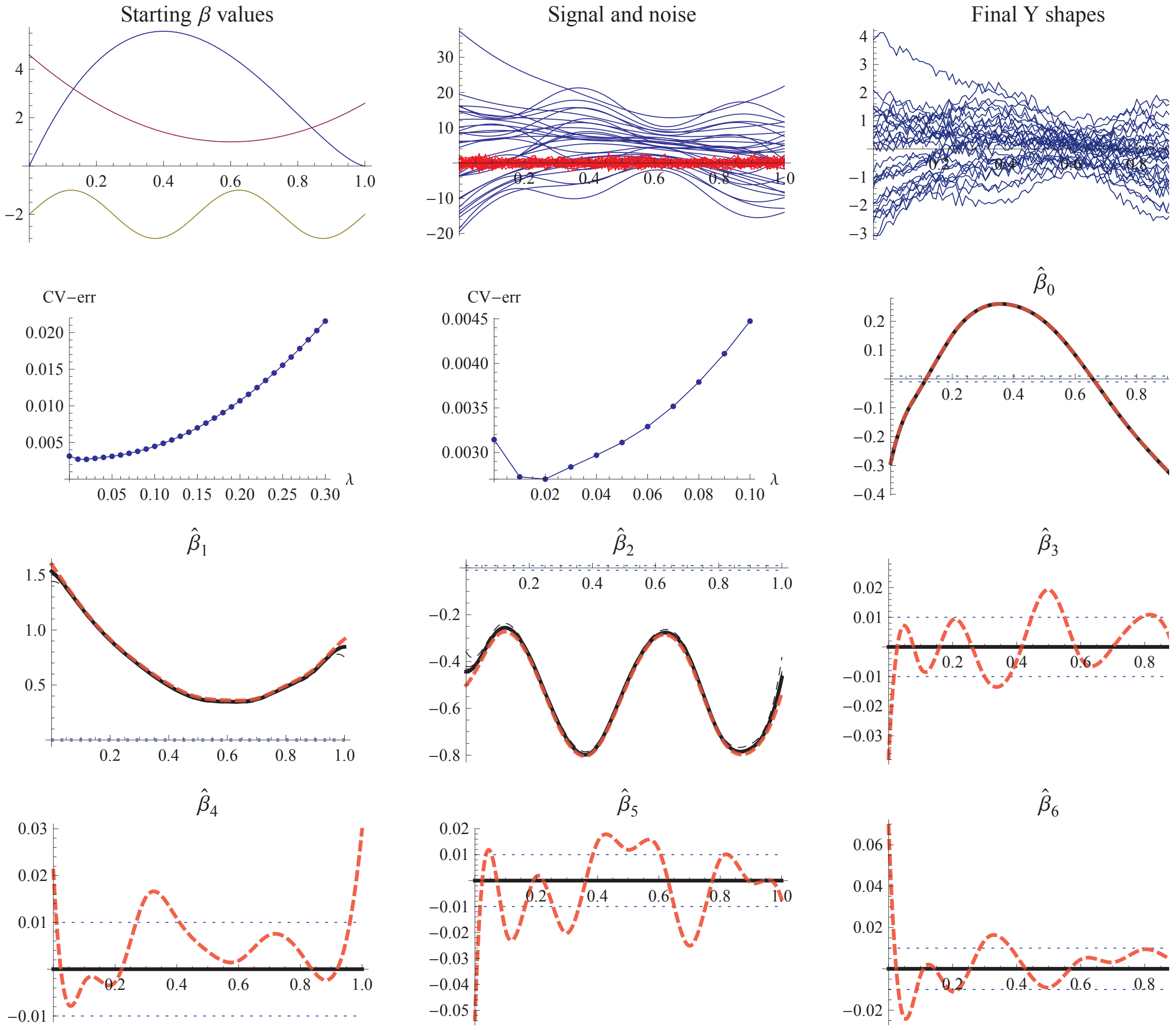
Case 1

- Number of regressors to select: 2.
- Erorr type: White noise, sd=0.5; Random seed=133.
- Lambda for minimum cv-error: 0.007.



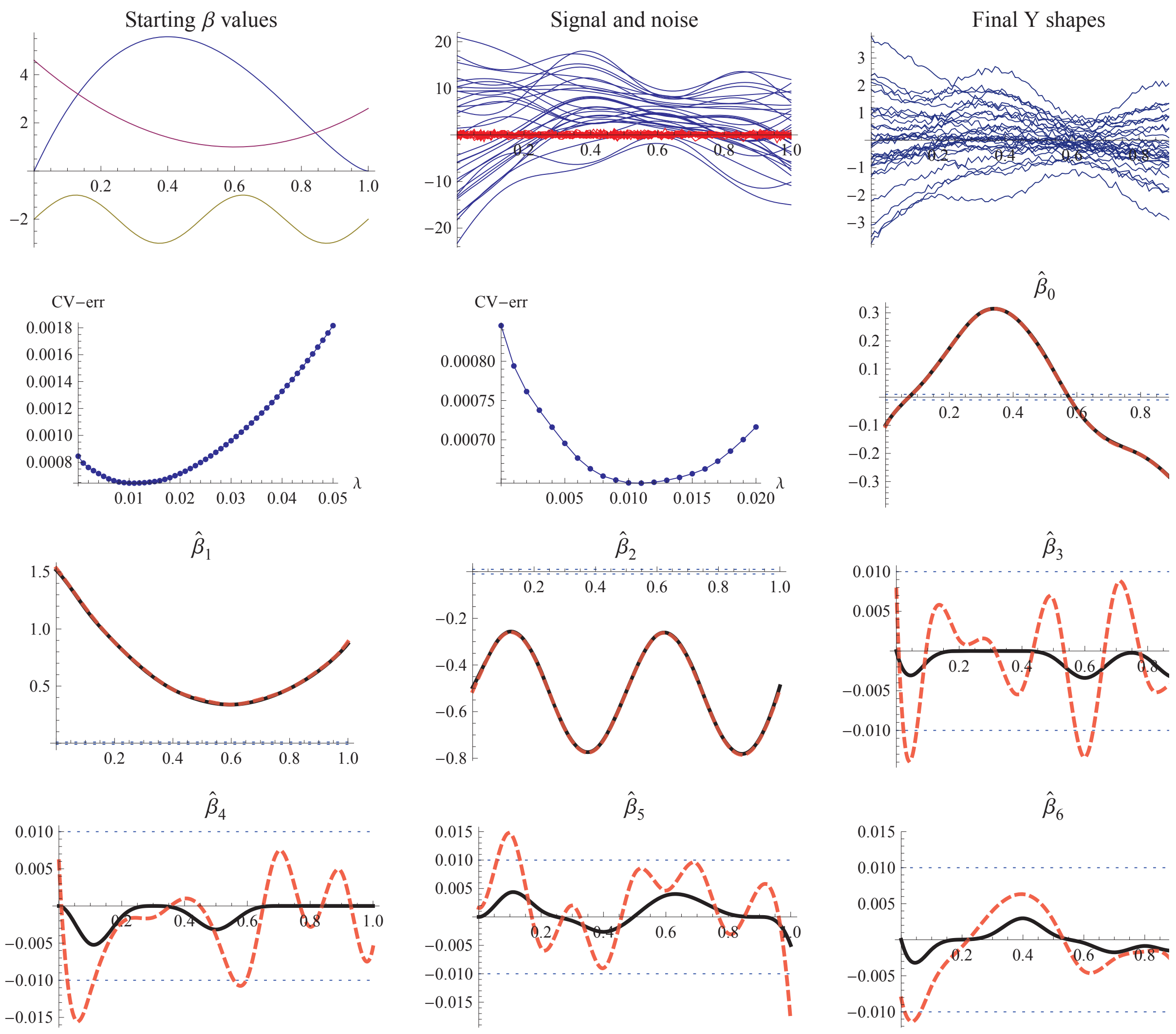
Case 2.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=0.5Randomseed=123.
- Lambdaforminimumcv-error0.006.



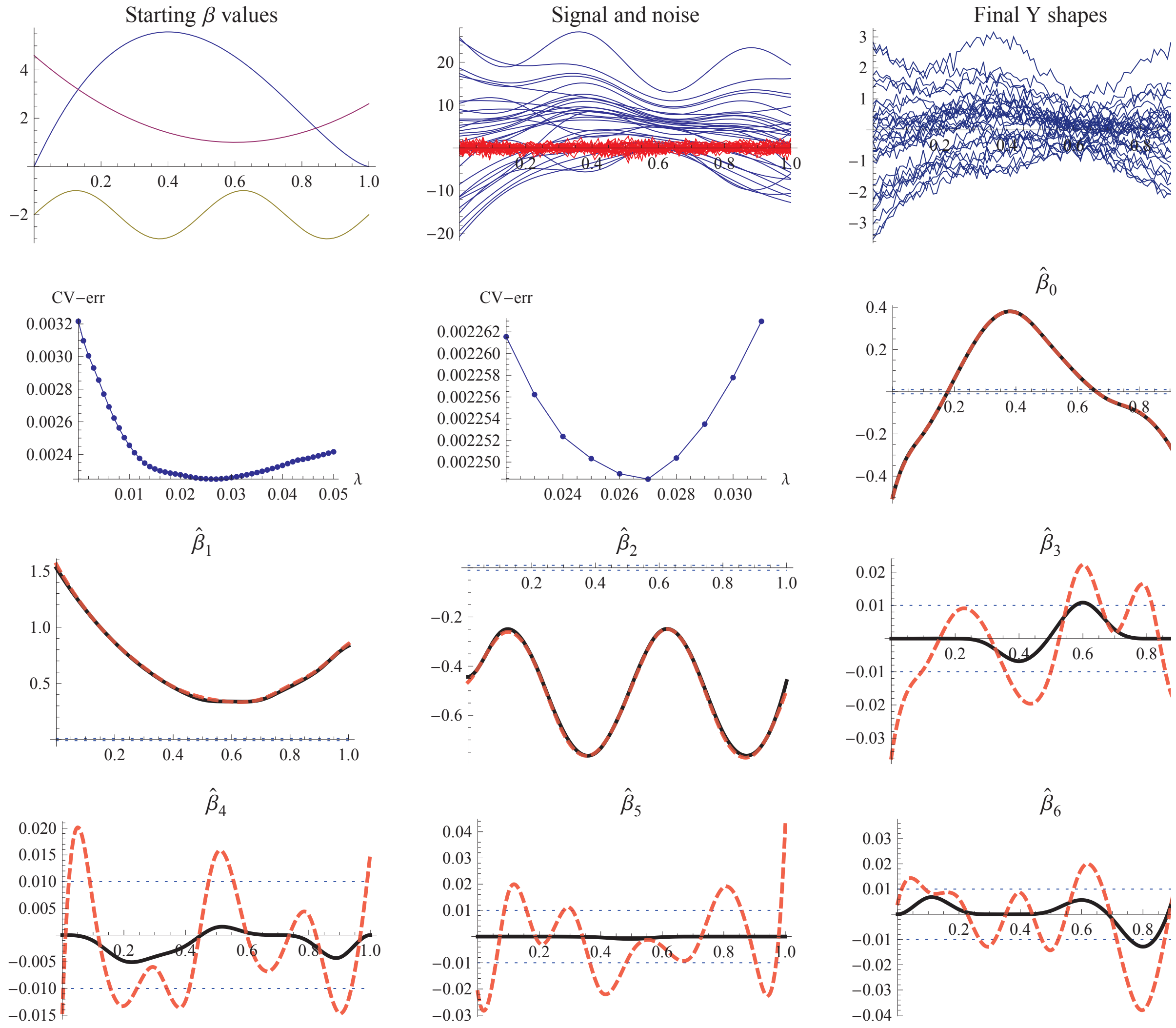
Case 3.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=0.5Randomseed=143.
- Lambdaforminimumcv-error0.011.



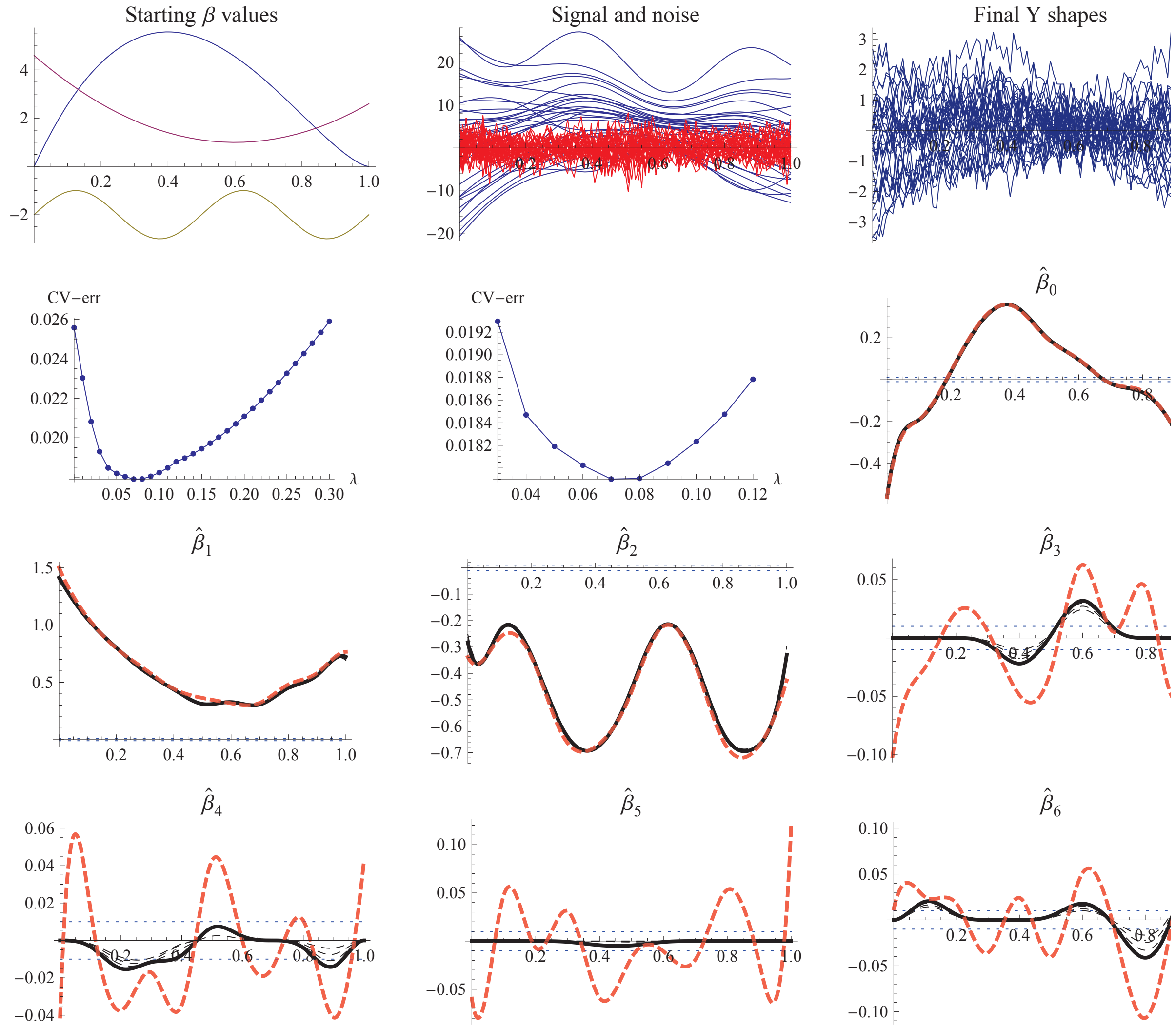
Case 4.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=1Randomseed=123.
- Lambdaforminimumcv-error0.027.



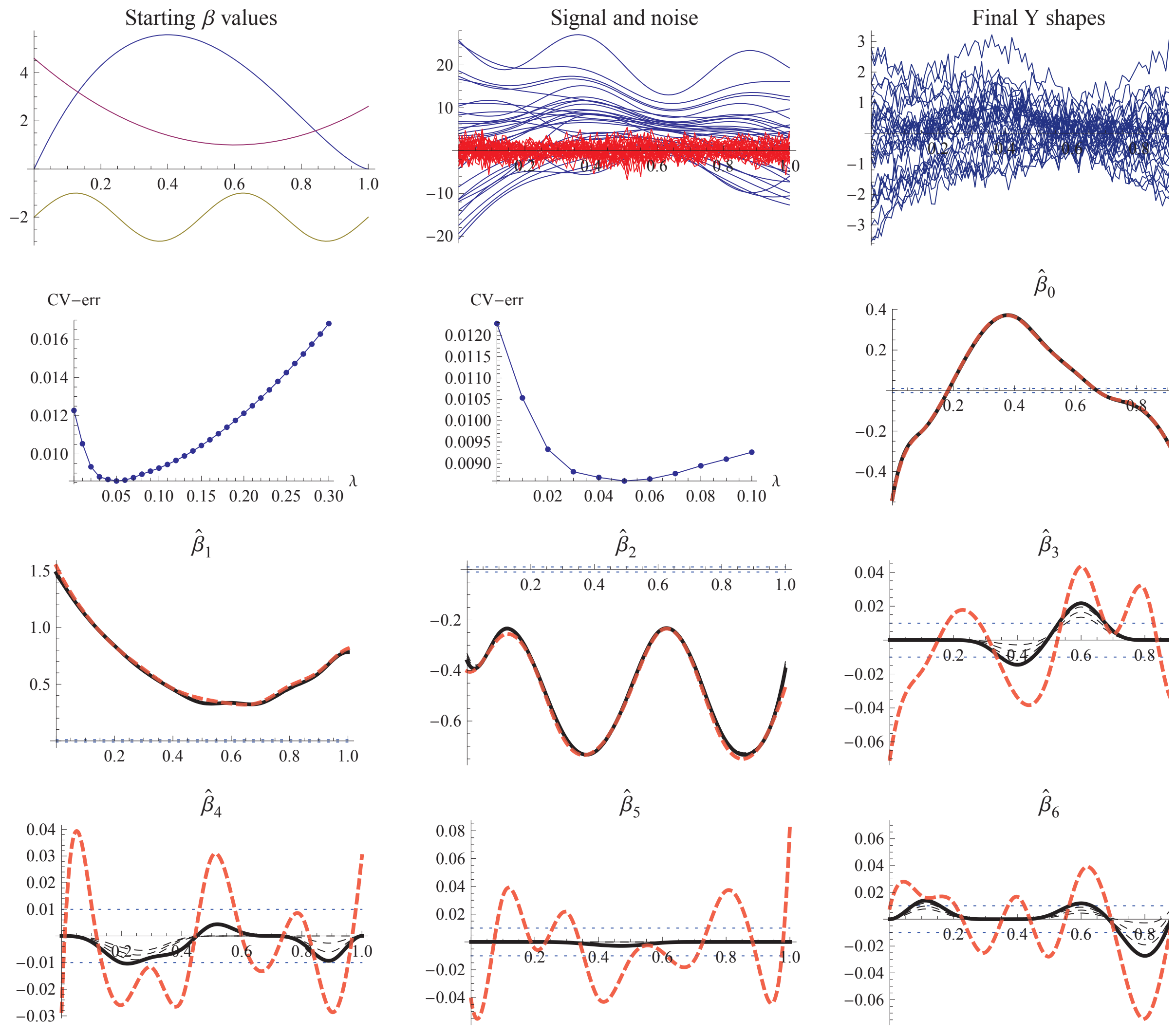
Case 5

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=3Randomseed=123.
- Lambdaforminimumcv-error0.07.Otherlambdas0.08,0.09,0.1.



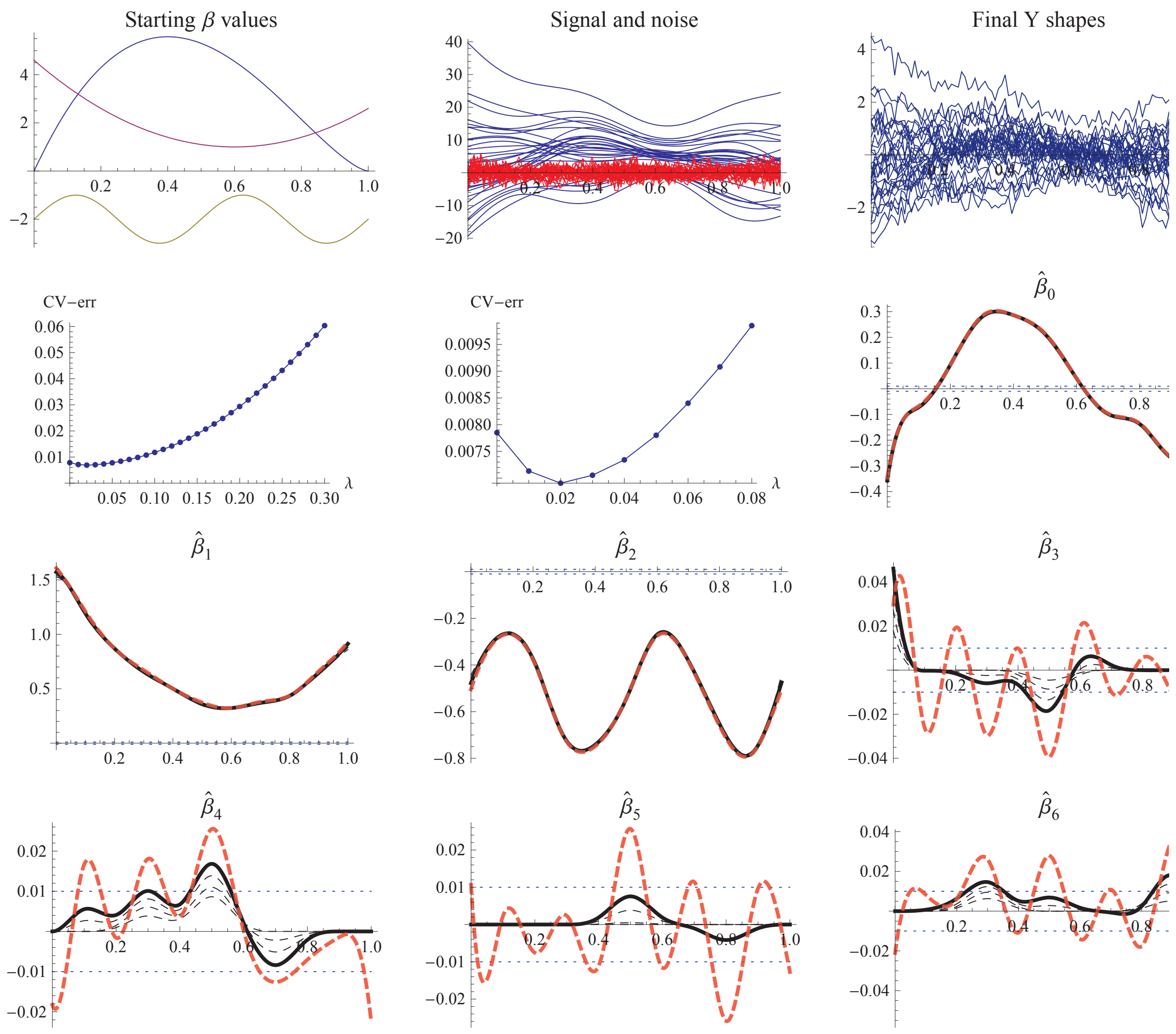
Case 6.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=2,Randomseed=123.
- Lambdaforminimumcv-error:0.05.Otherlambdas:0.06,0.07,0.08.



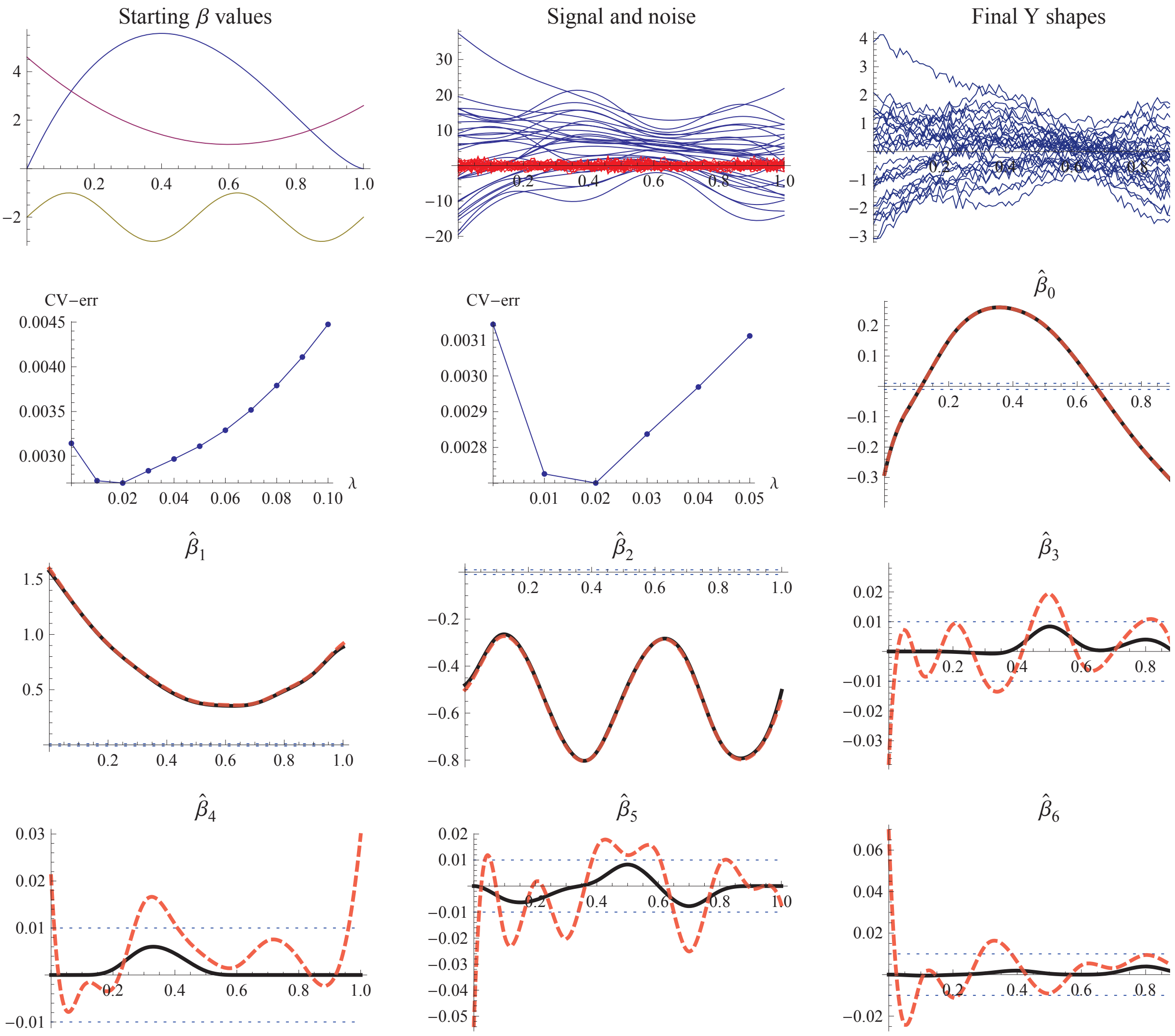
Case 7

- Number of regressors to select: 2.
- Erorr type: White noise, sd=2, Random seed=153.
- Lambda for minimum cv-error: 0.02. Other lambdas: 0.03, 0.04, 0.05.



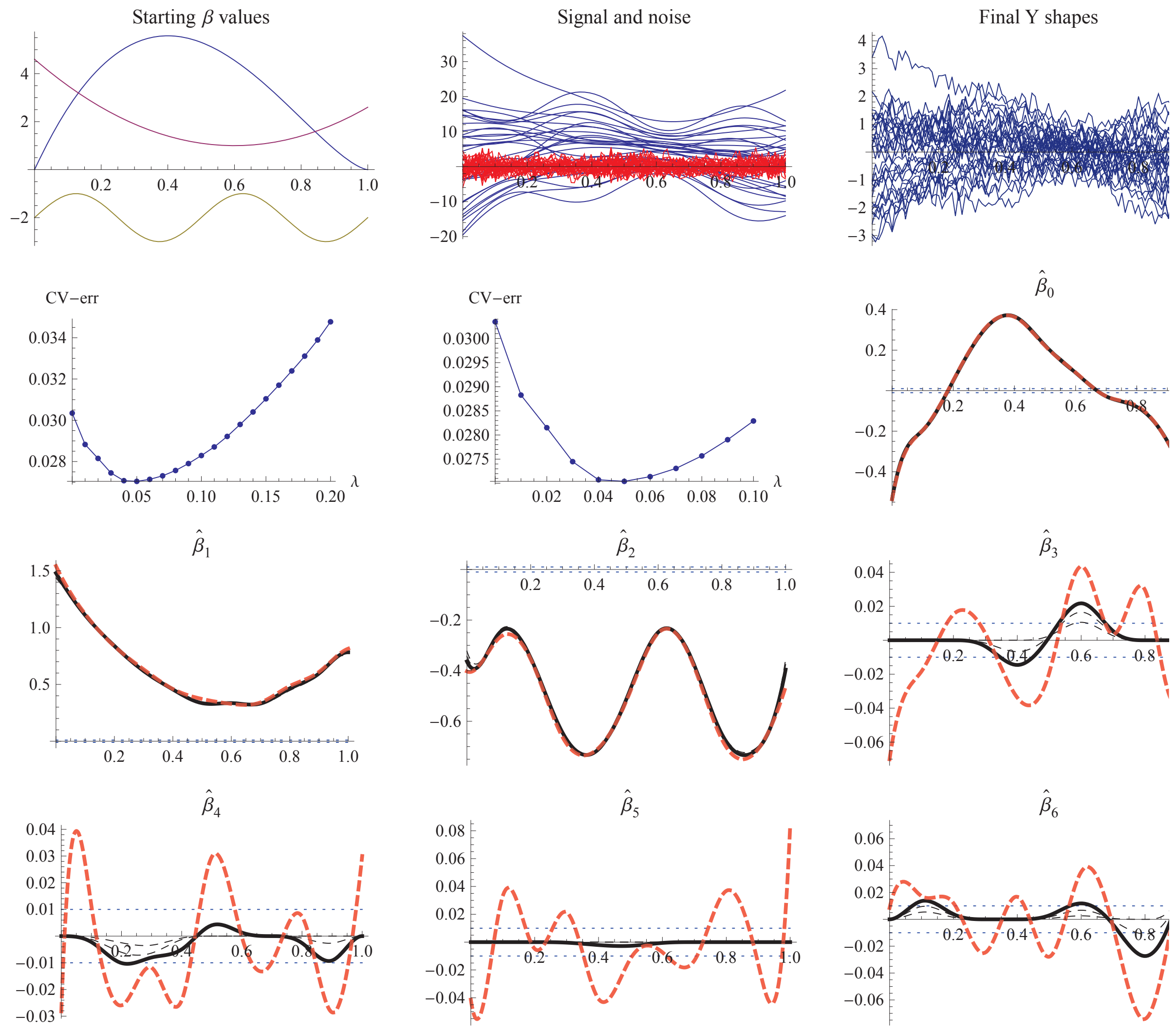
Case 8.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=1,Randomseed=1231.
- Lambdaforminimumcv-error0.02.



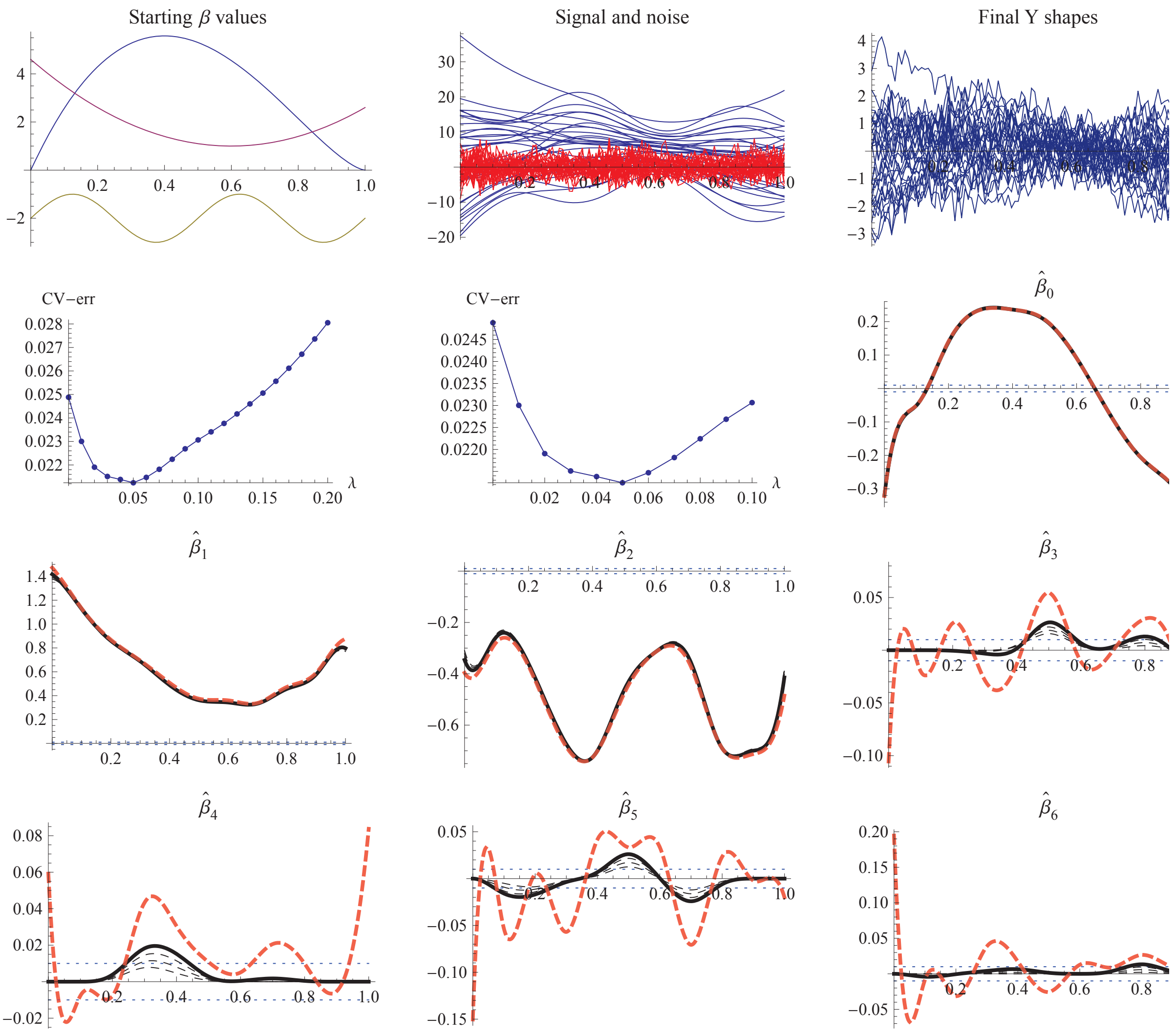
Case 9

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=2,Randomseed=1231.
- Lambdaforminimumcv-error0.05.Otherlambdas:0.07,0.09.



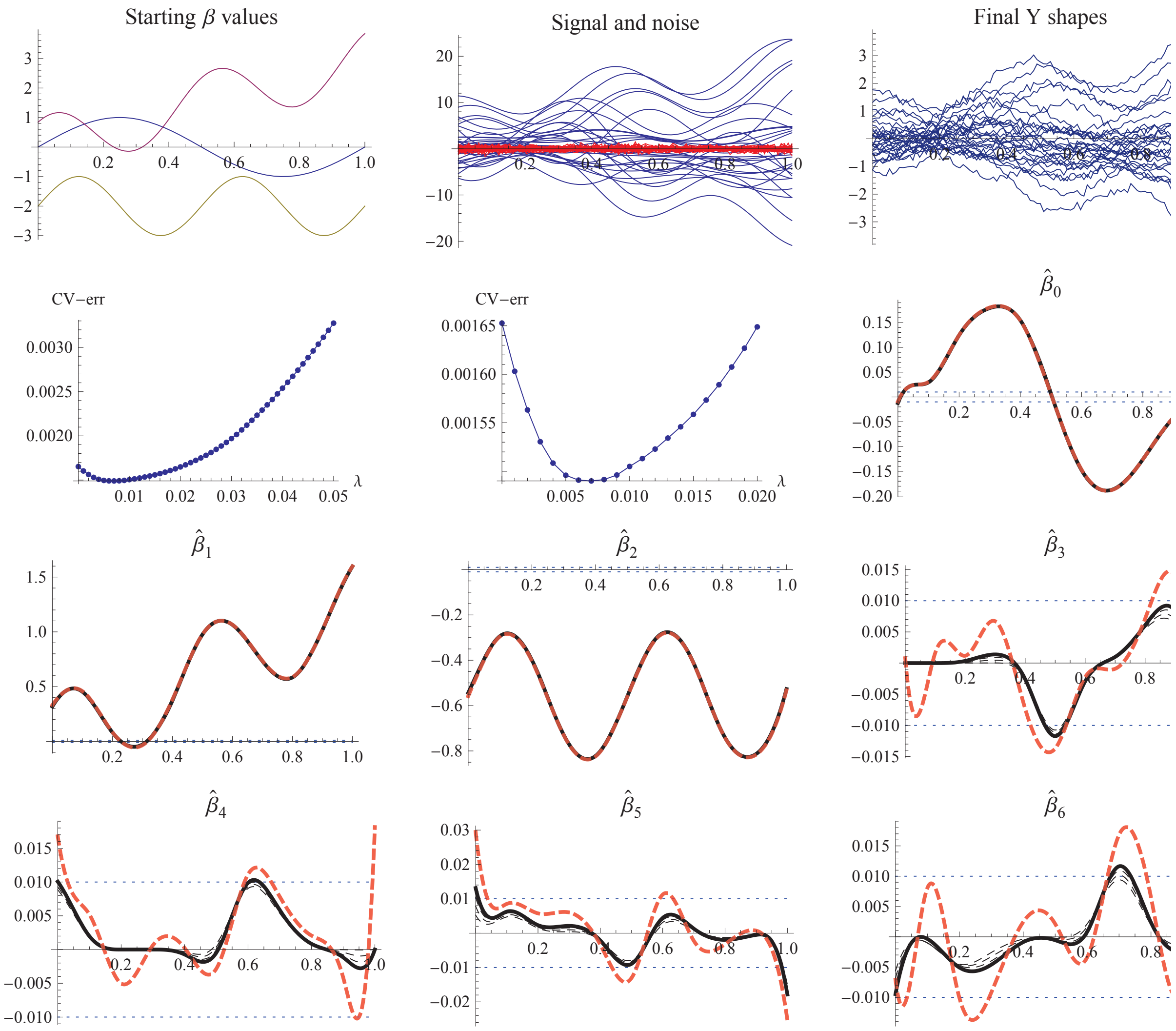
Case 10.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=3,Randomseed=1231.
- Lambdaforminimumcv-error0.05.Otherlambdas:0.06,0.07,0.08.



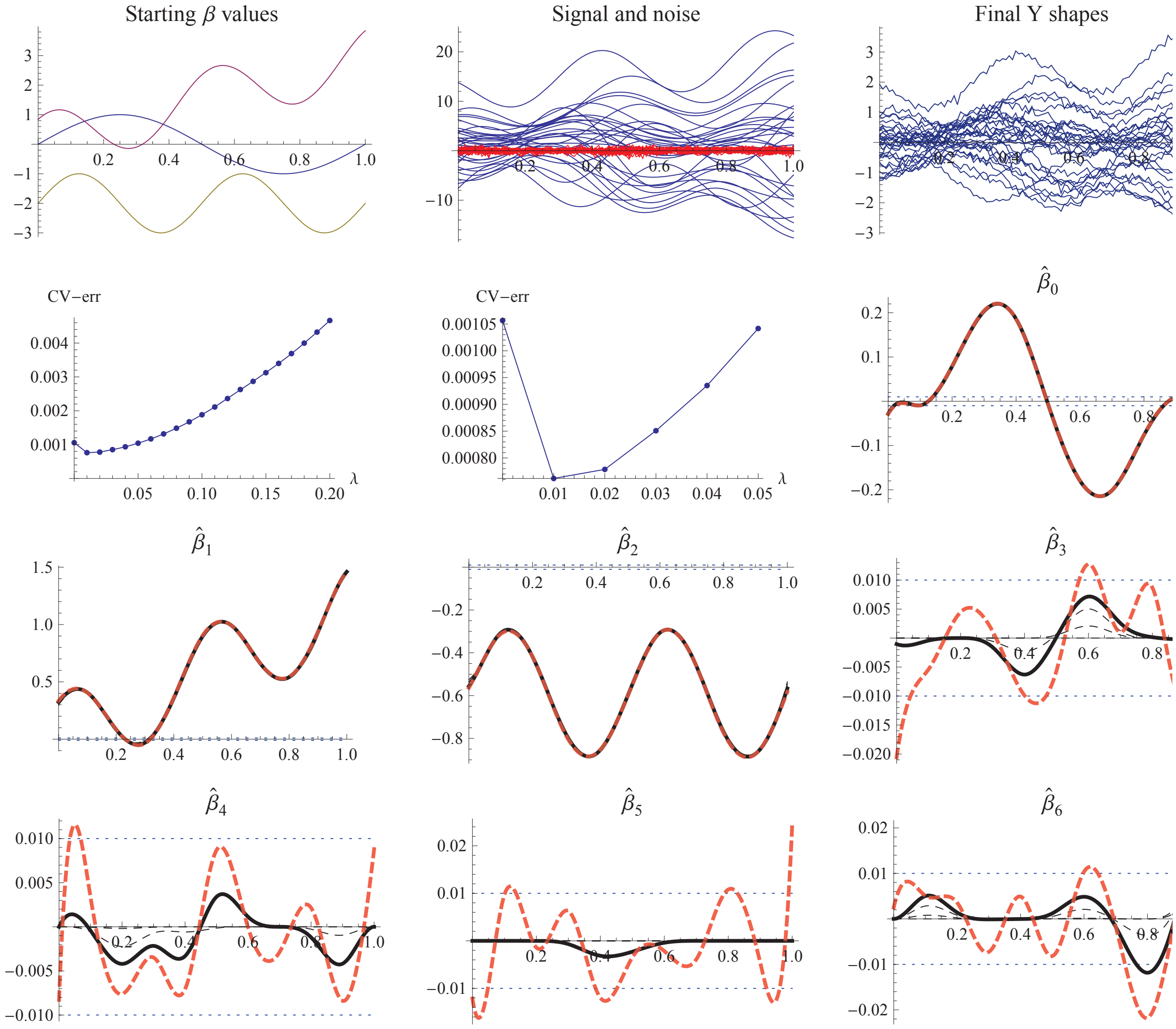
Case 11.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=0.5,Randomseed=133.
- Lambdaforminimumcv-error0.007.Otherlambdas:0.008,0.009,0.01.



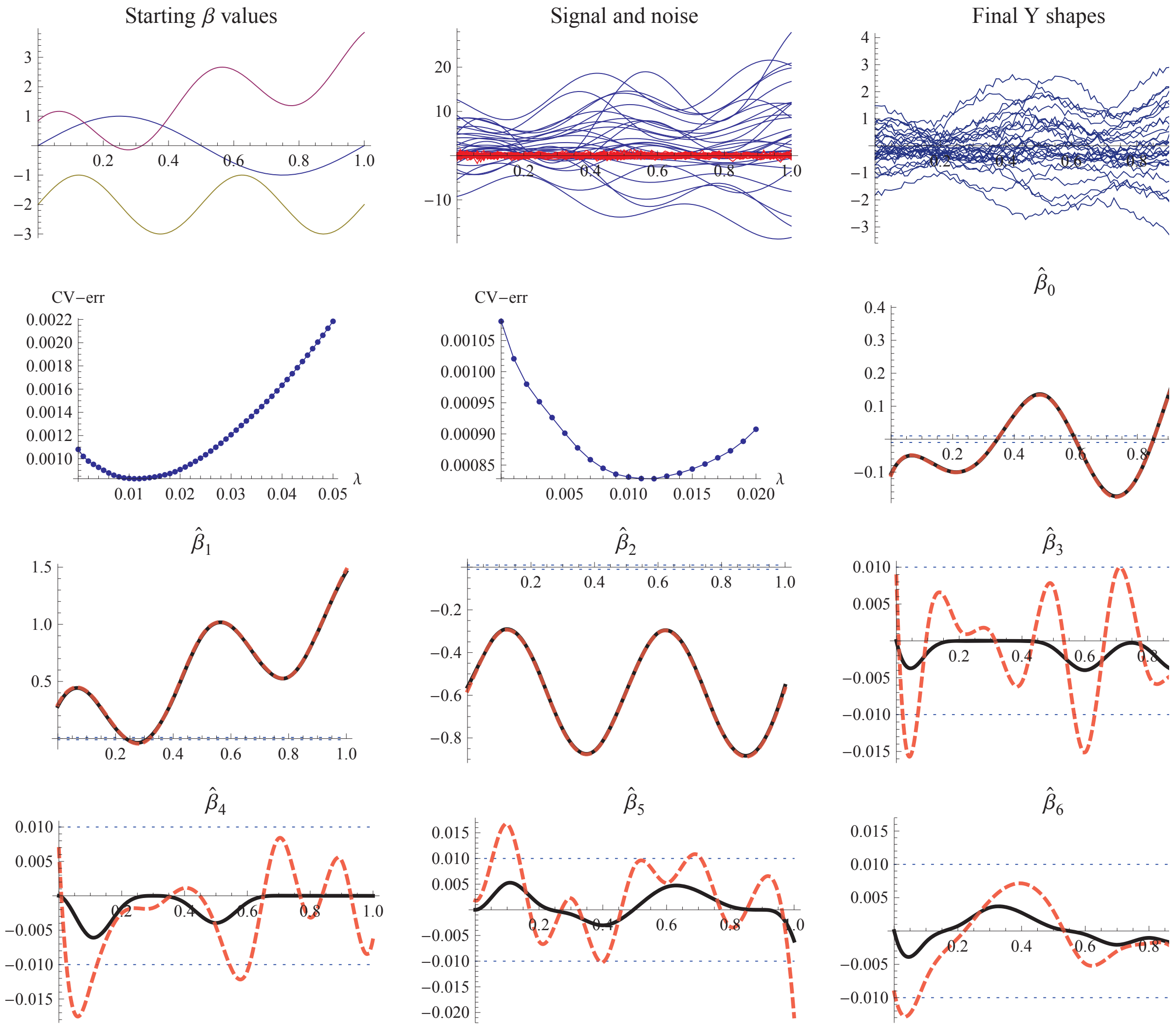
Case 12.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise,sd=0.5,Randomseed=123.
- Lambdaforminimumcv-error:0.01.Otherlambdas:0.02,0.03,0.04.



Case 13.

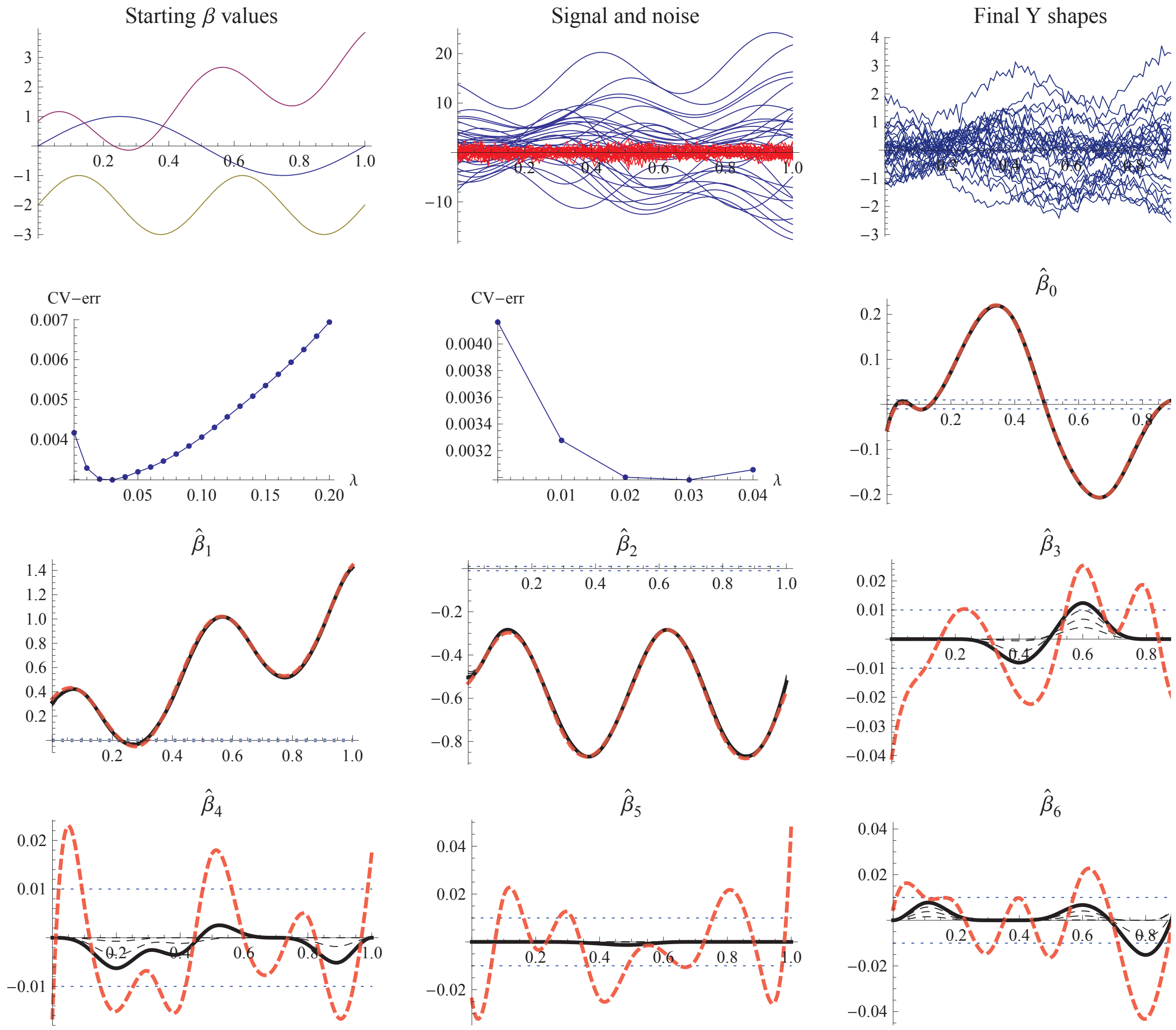
- Numberof regressorstoselect:2.
- Erorrtype:Whitenoise, sd=0.5,Randomseed=143.
- Lambdaforminimumcv-error0.012.



Case 14.

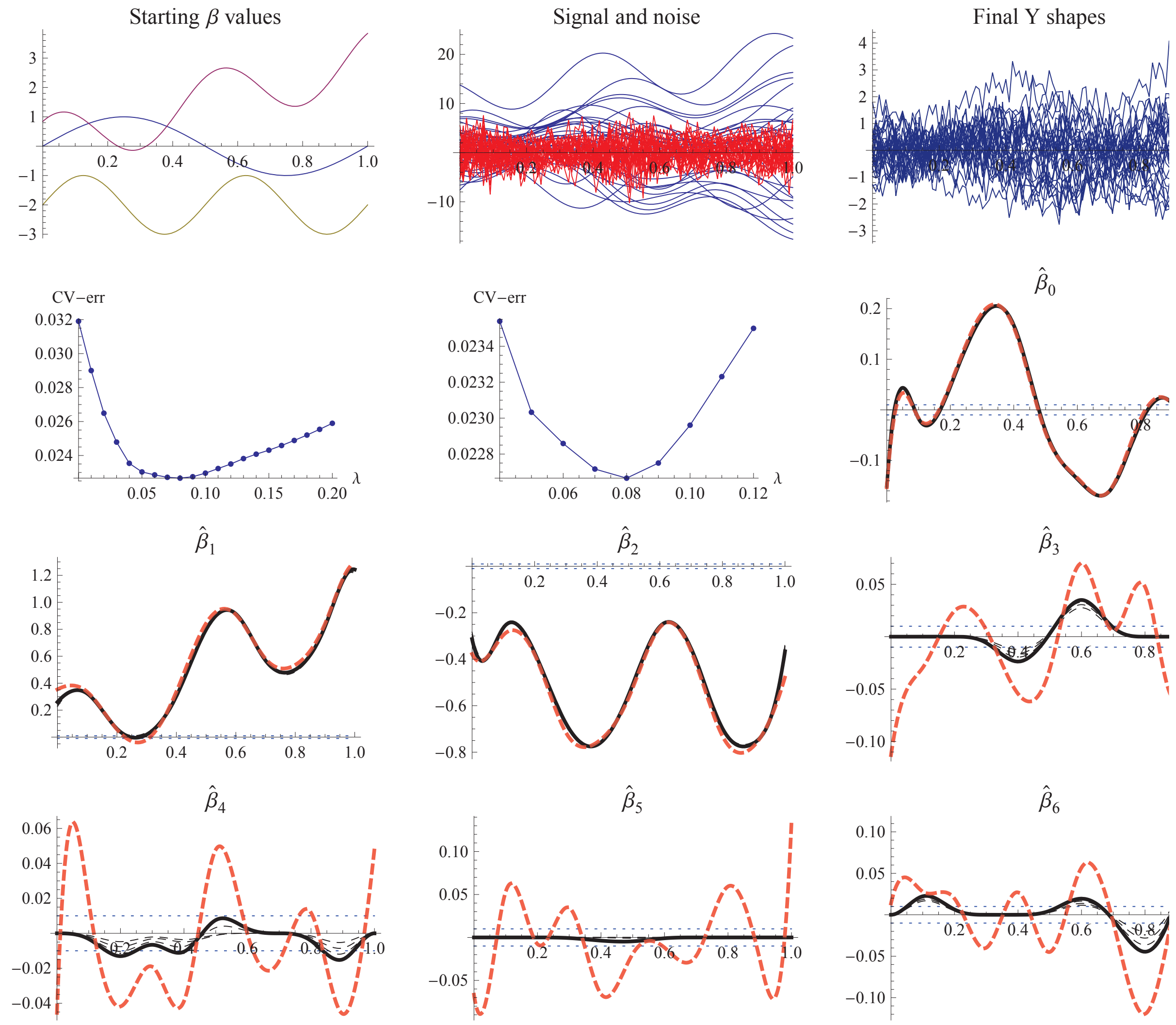
- Number of regressors to select: 2.
- Erorr type: White noise, sd=1, Random seed=123.
- Lambda for minimum cv-error: 0.03. Other lambdas: 0.04, 0.05, 0.06.

Out[280]=



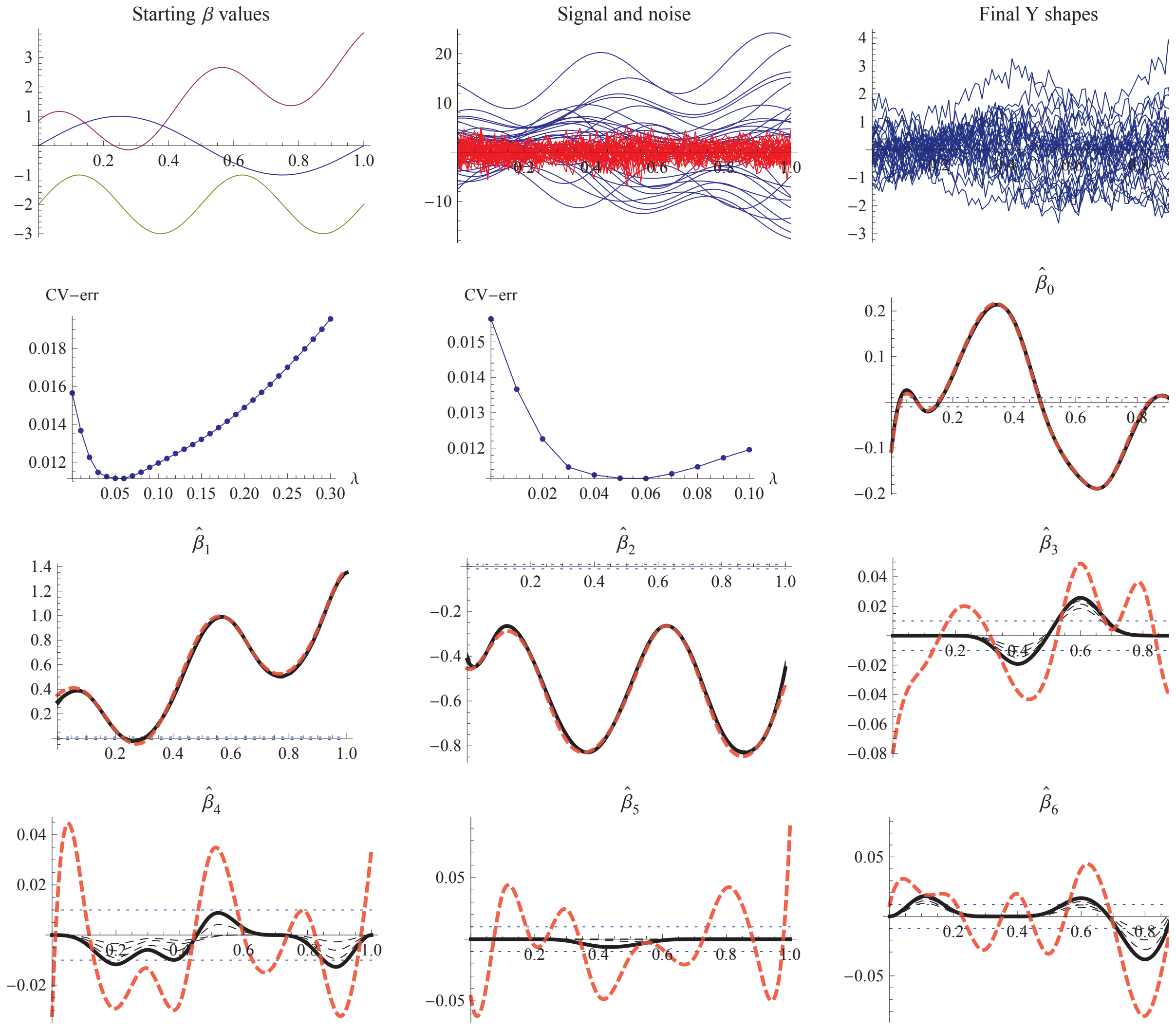
Case 15.

- Numberof regressorstoselect:2.
- Erorrtype:Whitenoise, sd=3.Randomseed=123.
- Lambdaforminimumcv-error0.08.Otherlambdas:0.09,0.10,0.11.



Case 16.

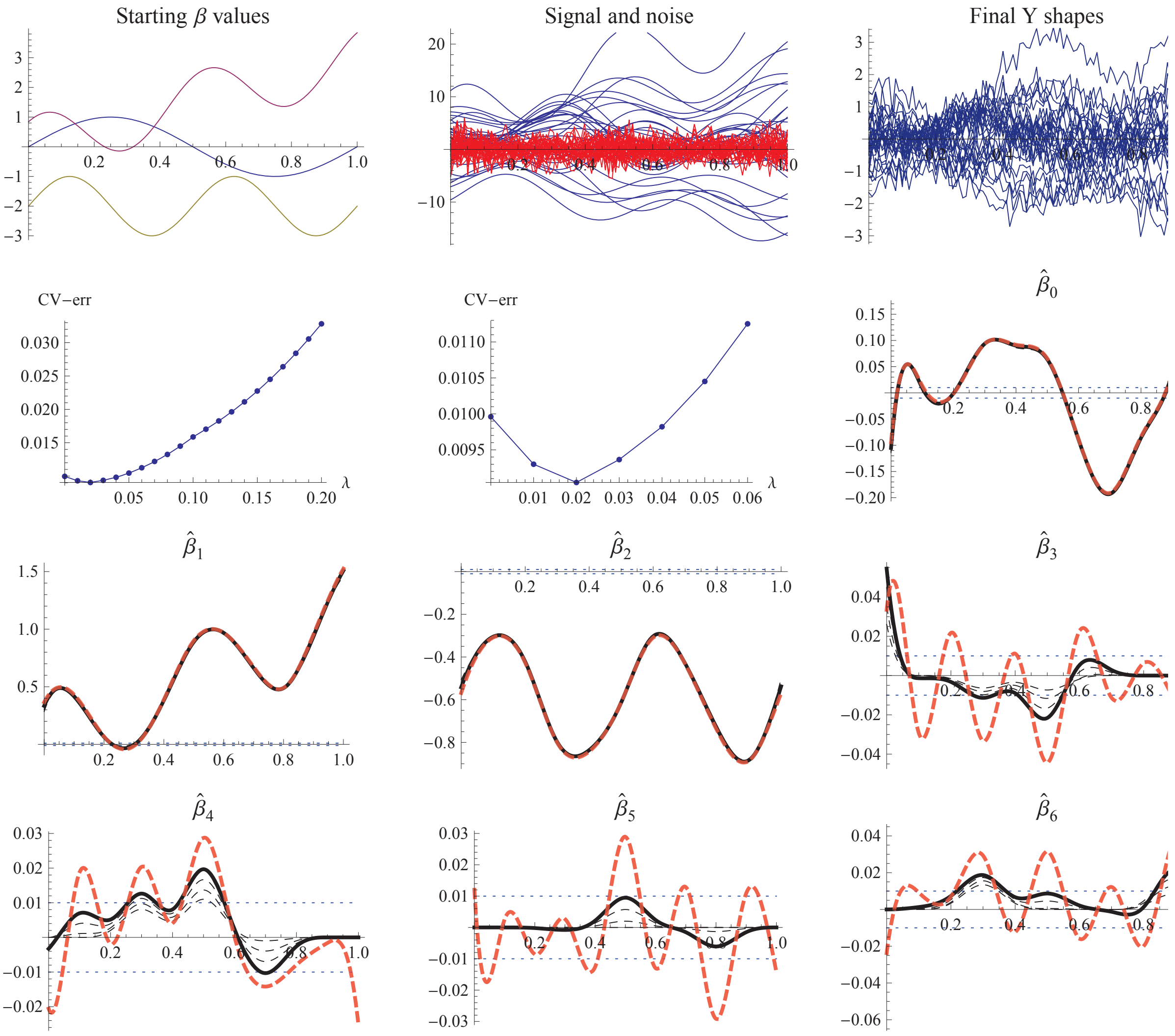
- Numberof regressorstoselect:2.
- Erorrtype:Whitenoise, sd=2,Randomseed=123.
- Lambdaforminimumcv-error0.05.Otherlambdas:0.06,0.07,0.08.



Case 17.

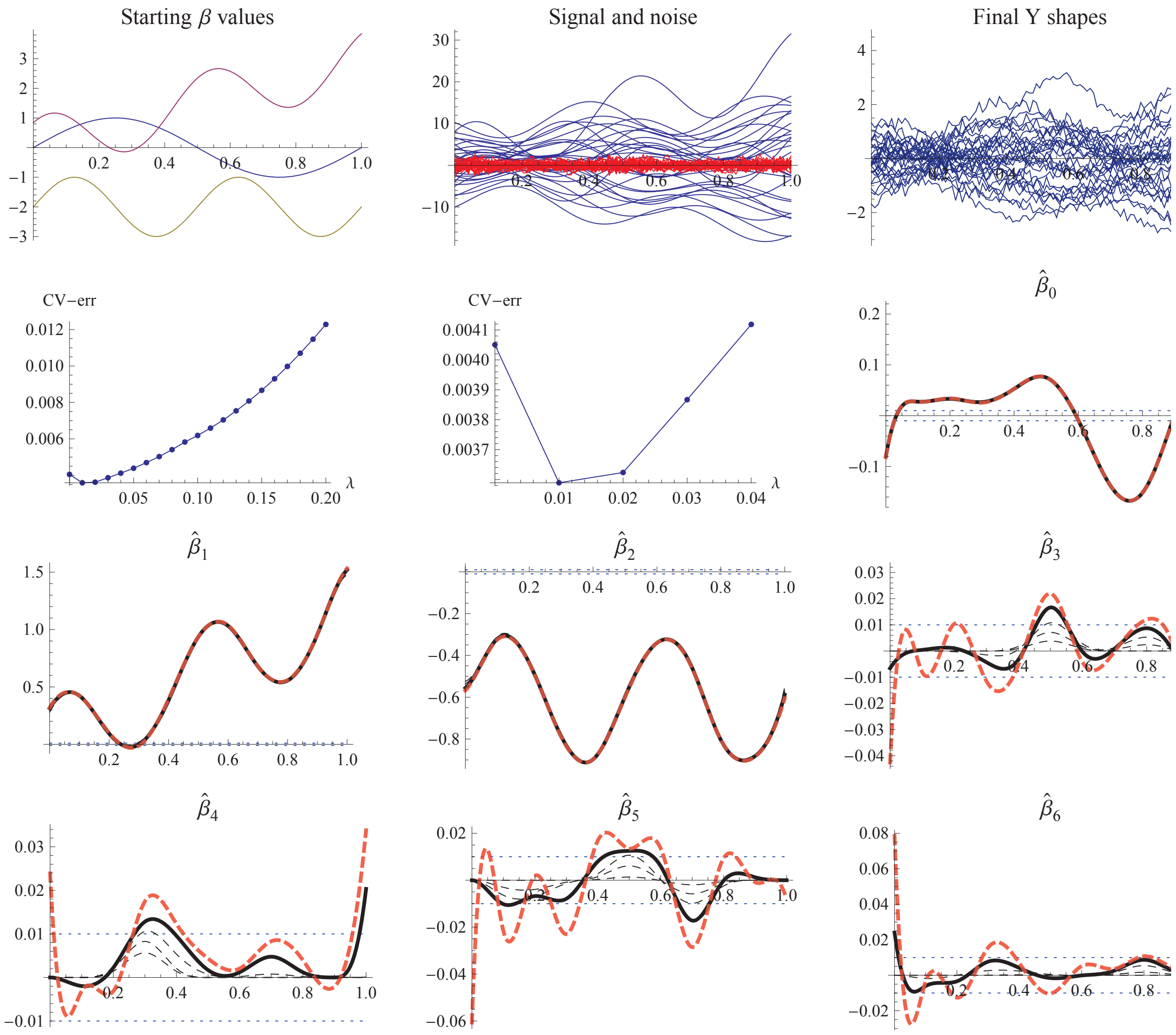
- Number of regressors to select: 2.
- Erorr type: White noise, sd = 2. Random seed = 153.
- Lambda for minimum cv-error: 0.02. Other lambdas: 0.03, 0.04, 0.05.

Out[234]=



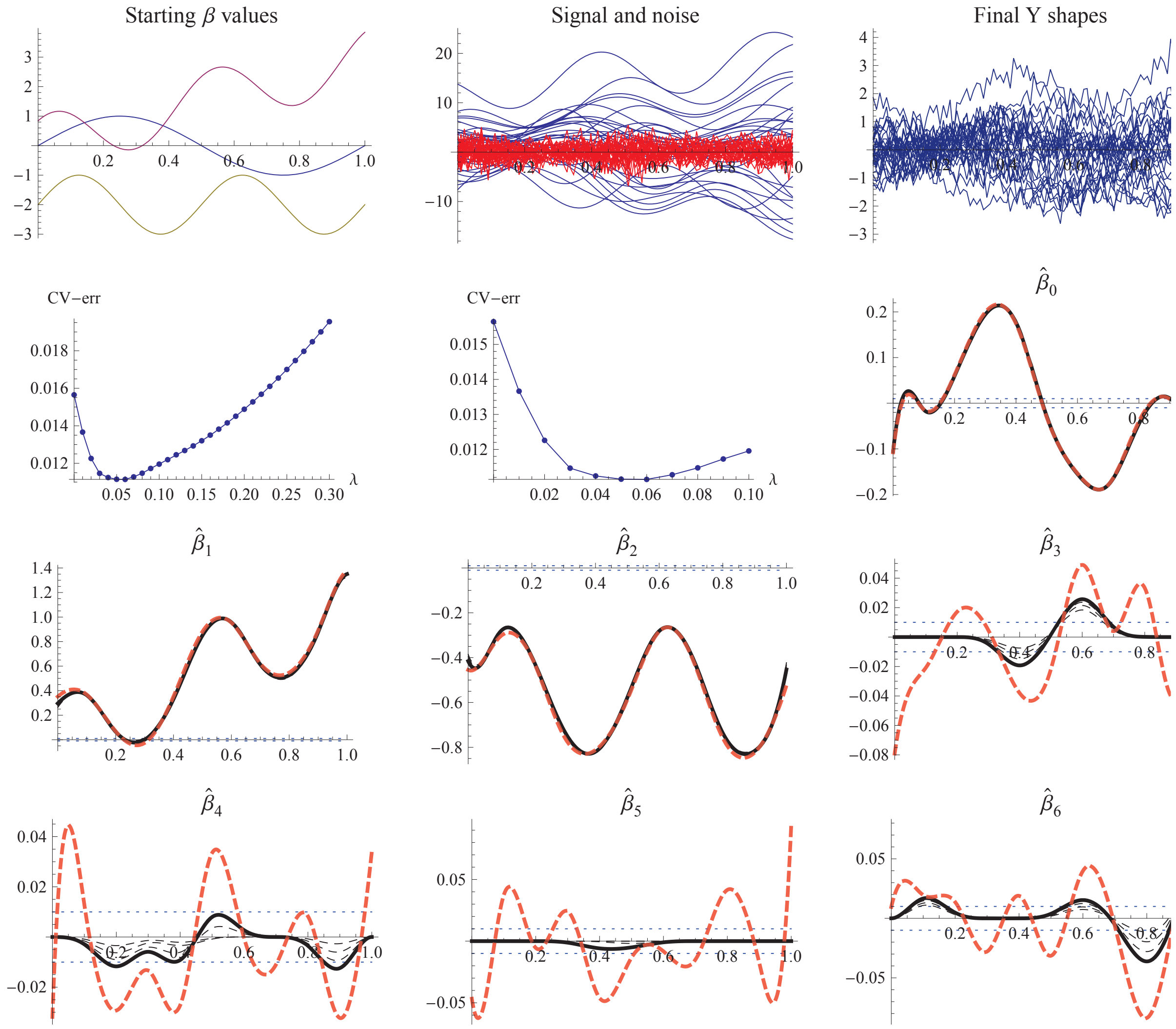
Case 18.

- Numberof regressorsto select:2.
- Erorrtype:Whitenoise, sd=1.Randomseed=1231.
- Lambdaforminimumcv-error0.01.Otherlambdas:0.02,0.03,0.04.



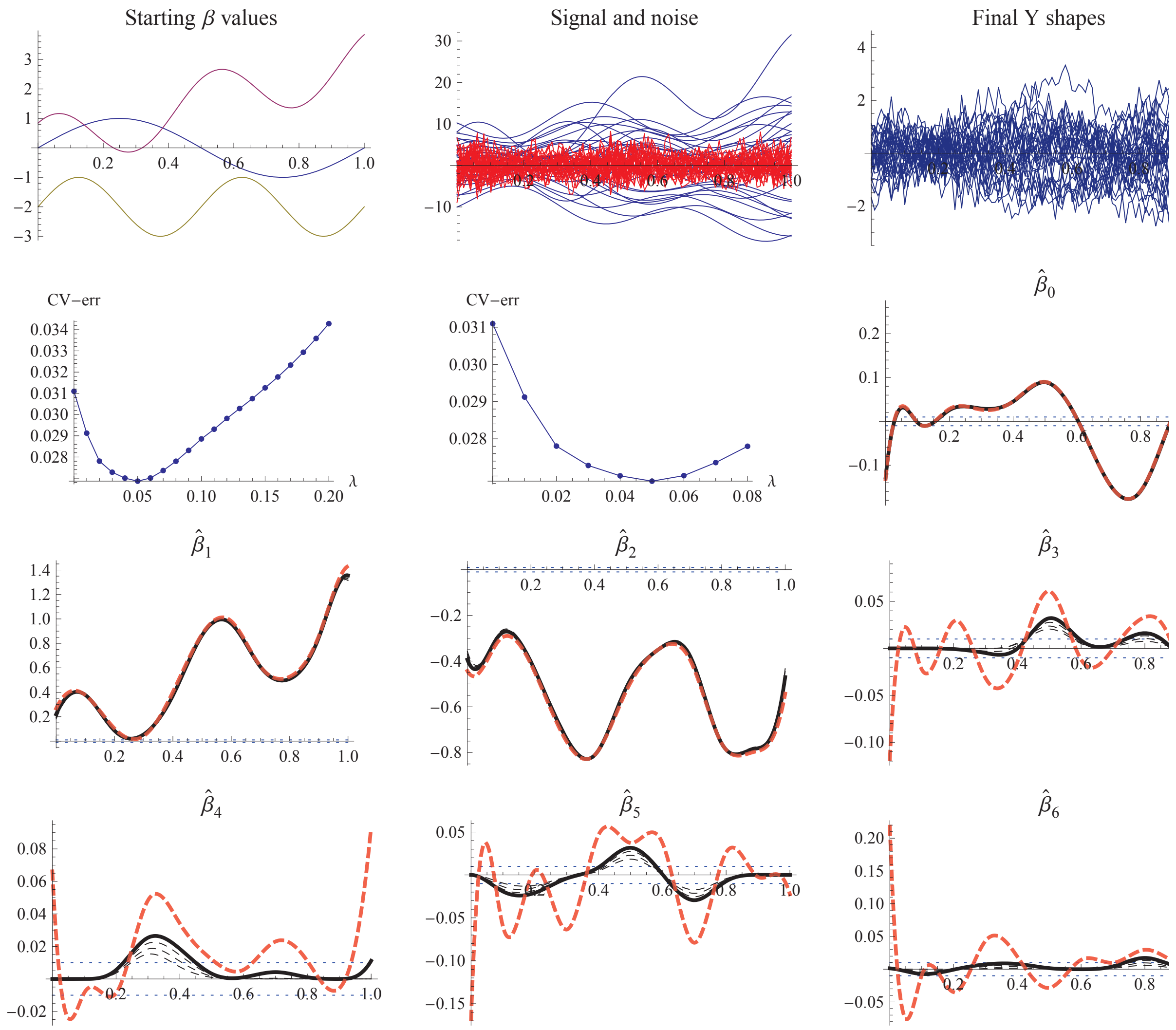
Case 19.

- Number of regressors to select: 2.
- Erorr type: White noise, sd=2, Random seed=1231.
- Lambda for minimum cv-error 0.03. Other lambdas: 0.04, 0.05.



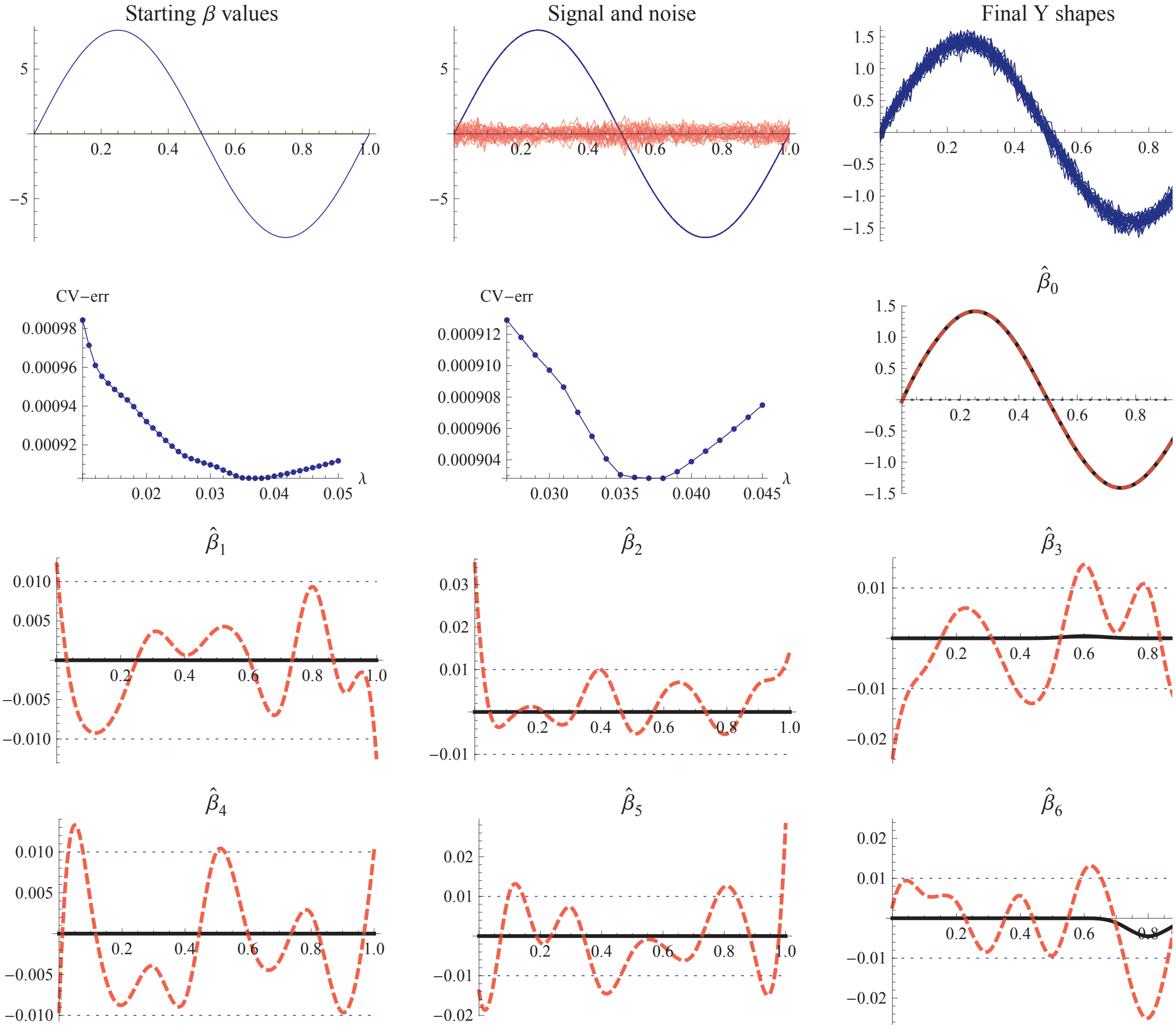
Case 20.

- Numberof regressorstoselect:2.
- Erorrtype:Whitenoise, sd=3,Randomseed=1231.
- Lambdaforminimumcv-error:0.05.Otherlambdas:0.06,0.07,0.08.



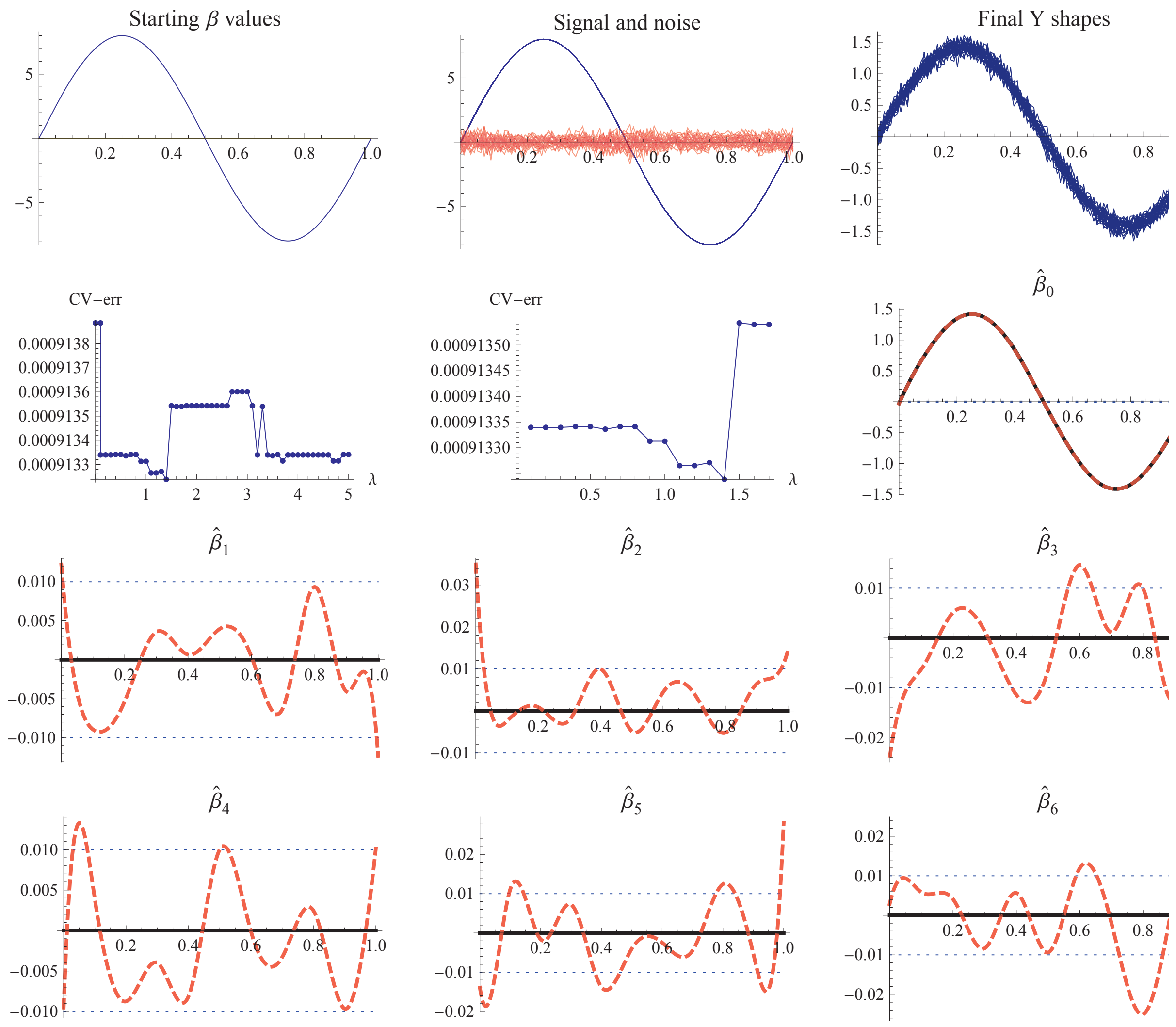
Case 21.

- Numberof regressorsto select:0.
- Erorrtype:Whitenoise, sd=0.5Randomseed=133.
- Lambdaforminimumcv-error0.037.



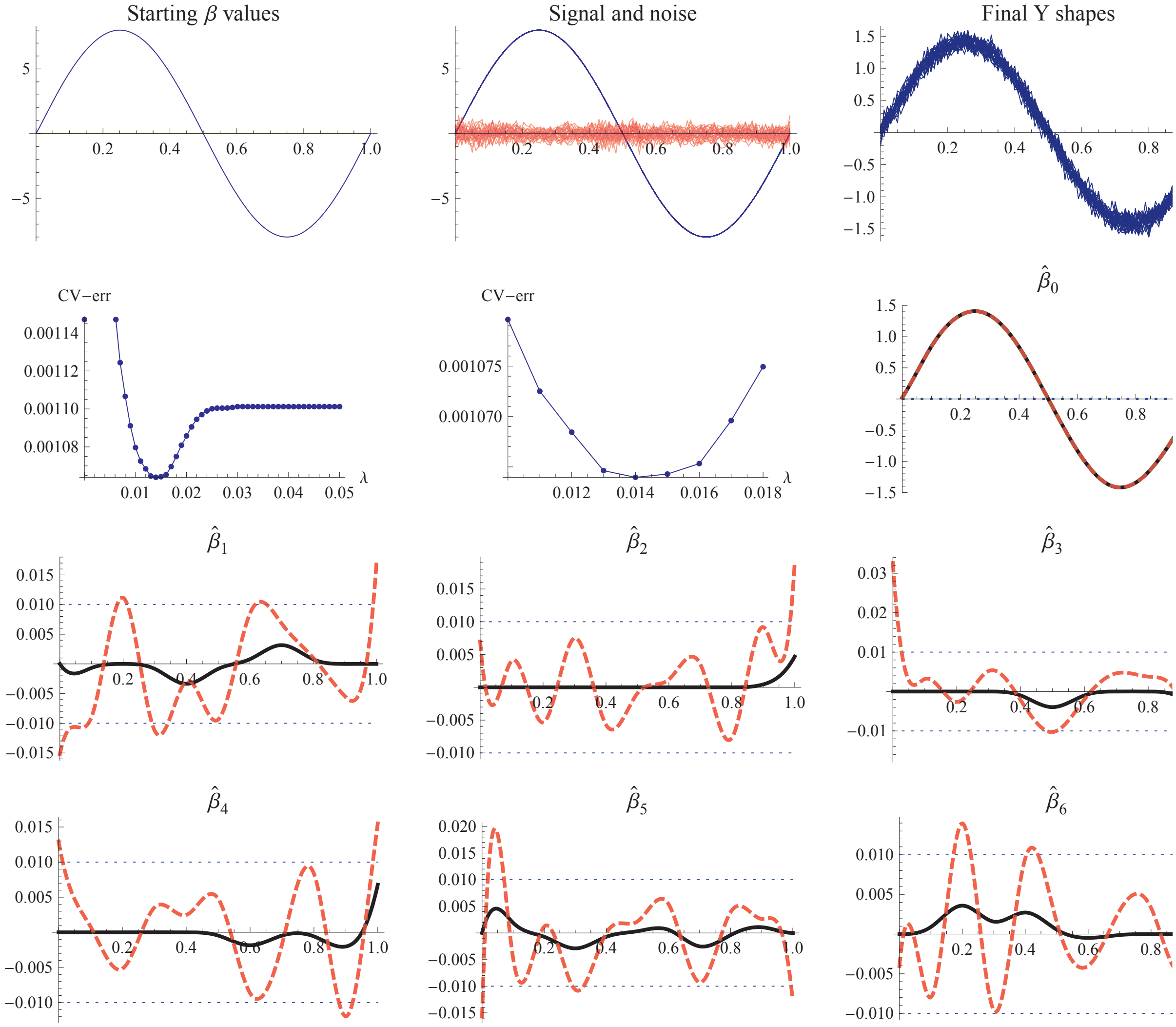
Case 22.

- Numberof regressorsto select:0.
- Erorrtype:Whitenoise, sd=0.5Randomseed=123.
- Lambdaforminimumcv-error:1.5.



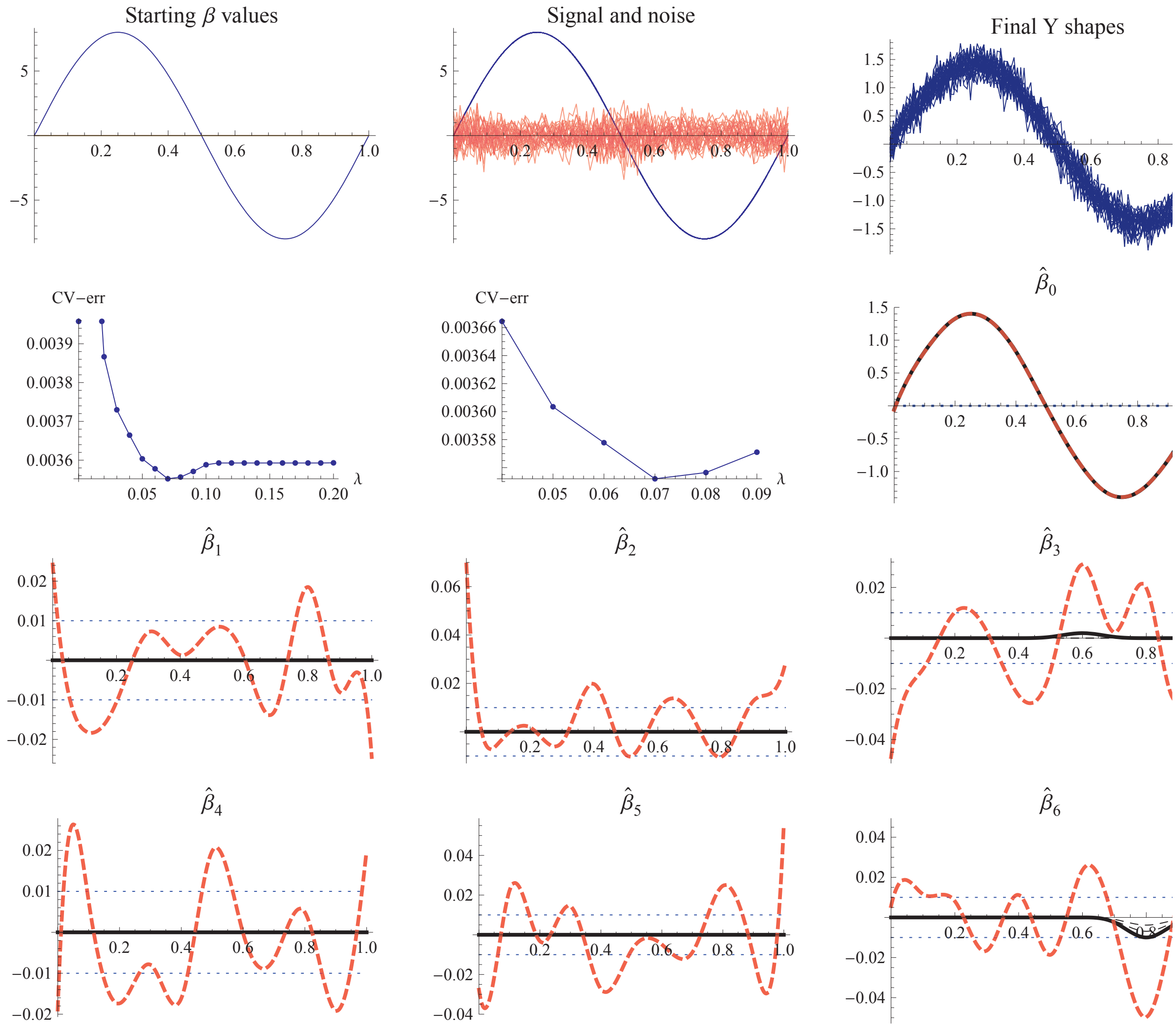
Case 23.

- Numberof regressorsto select:0.
- Erorrtype:Whitenoise, sd=0.5Randomseed=143.
- Lambdaforminimumcv-error0.014.



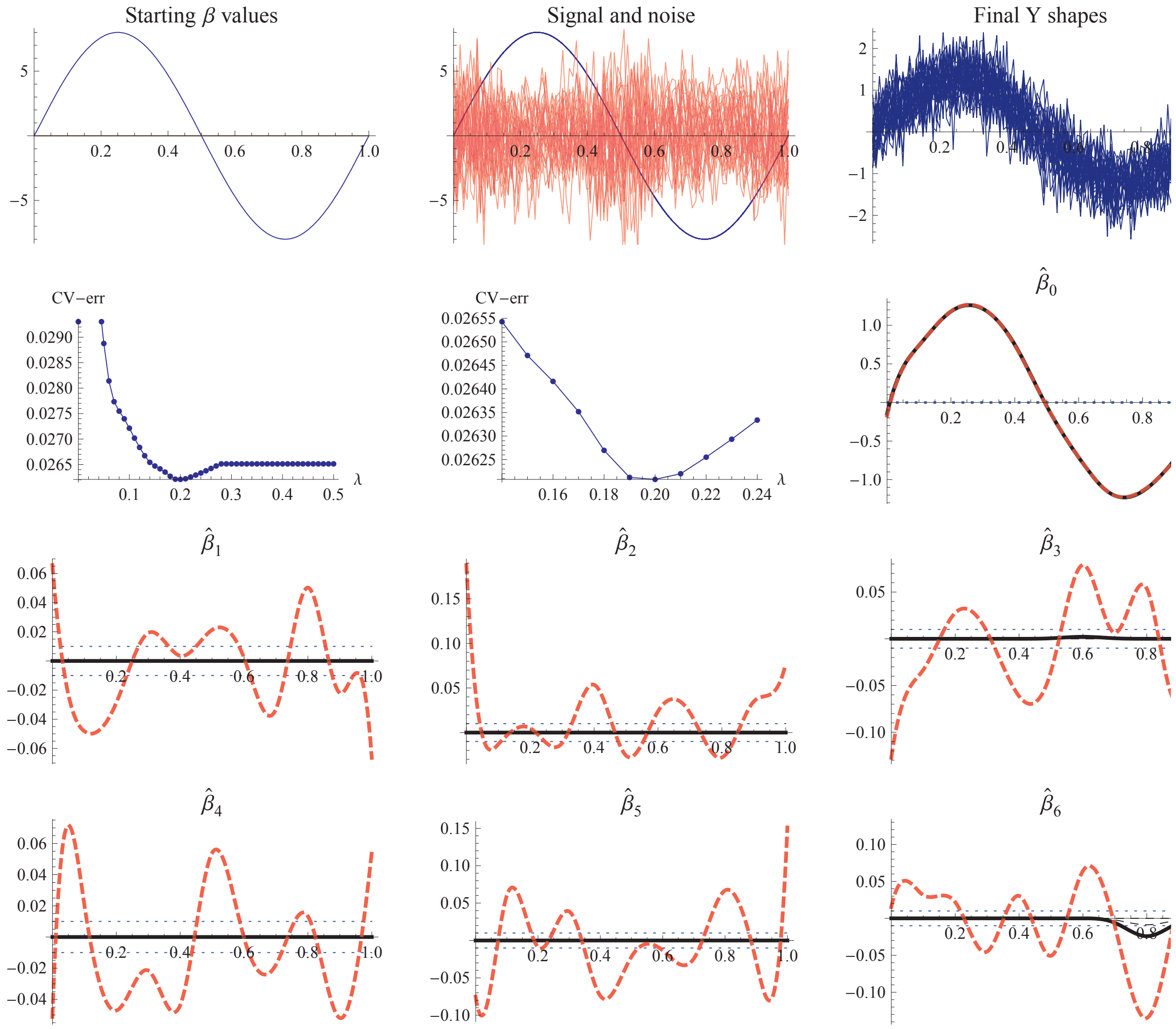
Case 24.

- Numberof regressorsto select:0.
- Erorrtype:Whitenoise, sd=1.Randomseed=123.
- Lambdaforminimumcv-error0.07.Otherlambdas:0.08,0.09



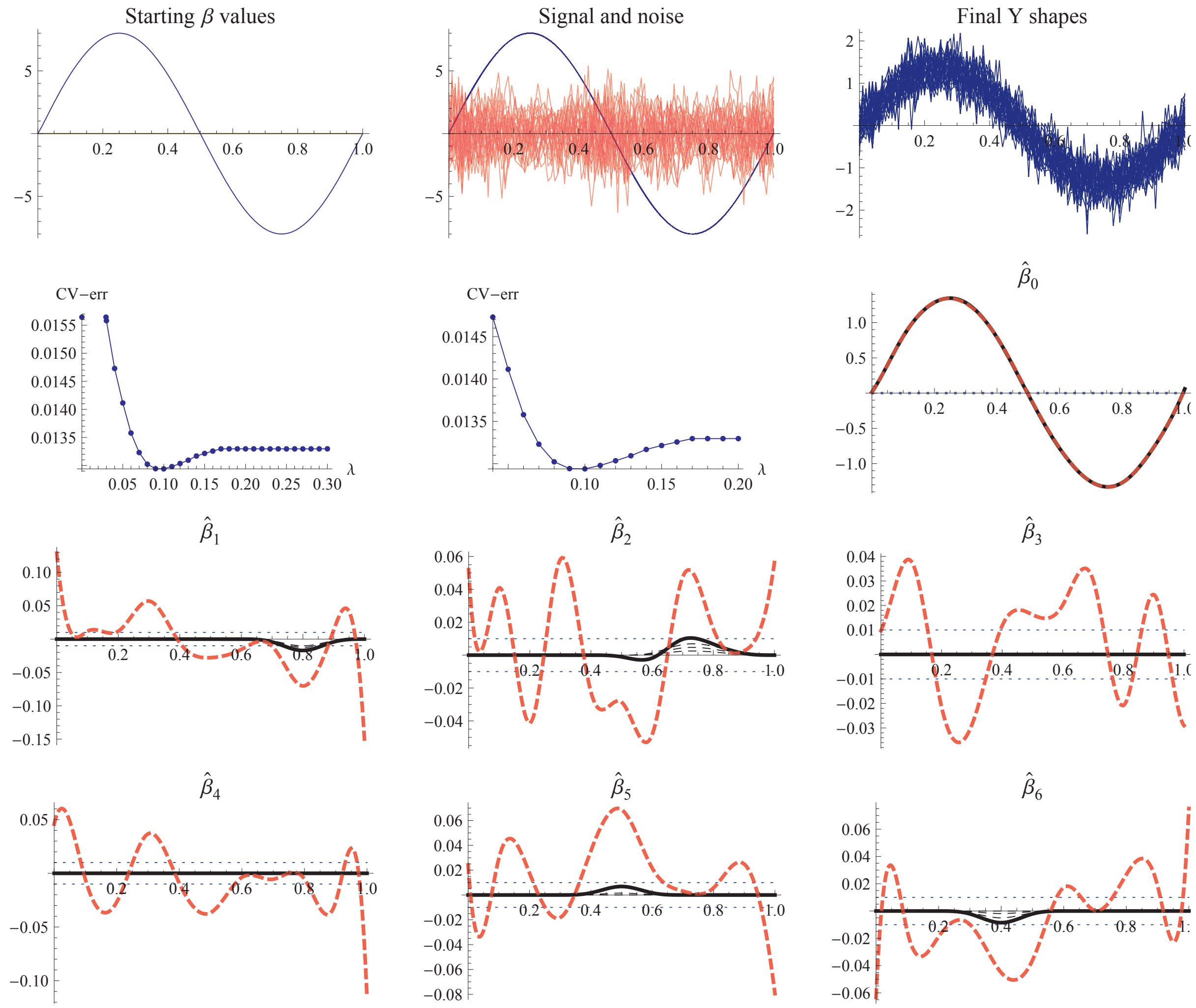
Case 25.

- Numberof regressorsto select:0.
- Erorrtype:Whitenoise, sd=3.Randomseed=123.
- Lambdaforminimumcv-error0.2.Otherlambdas:0.25,0.3.



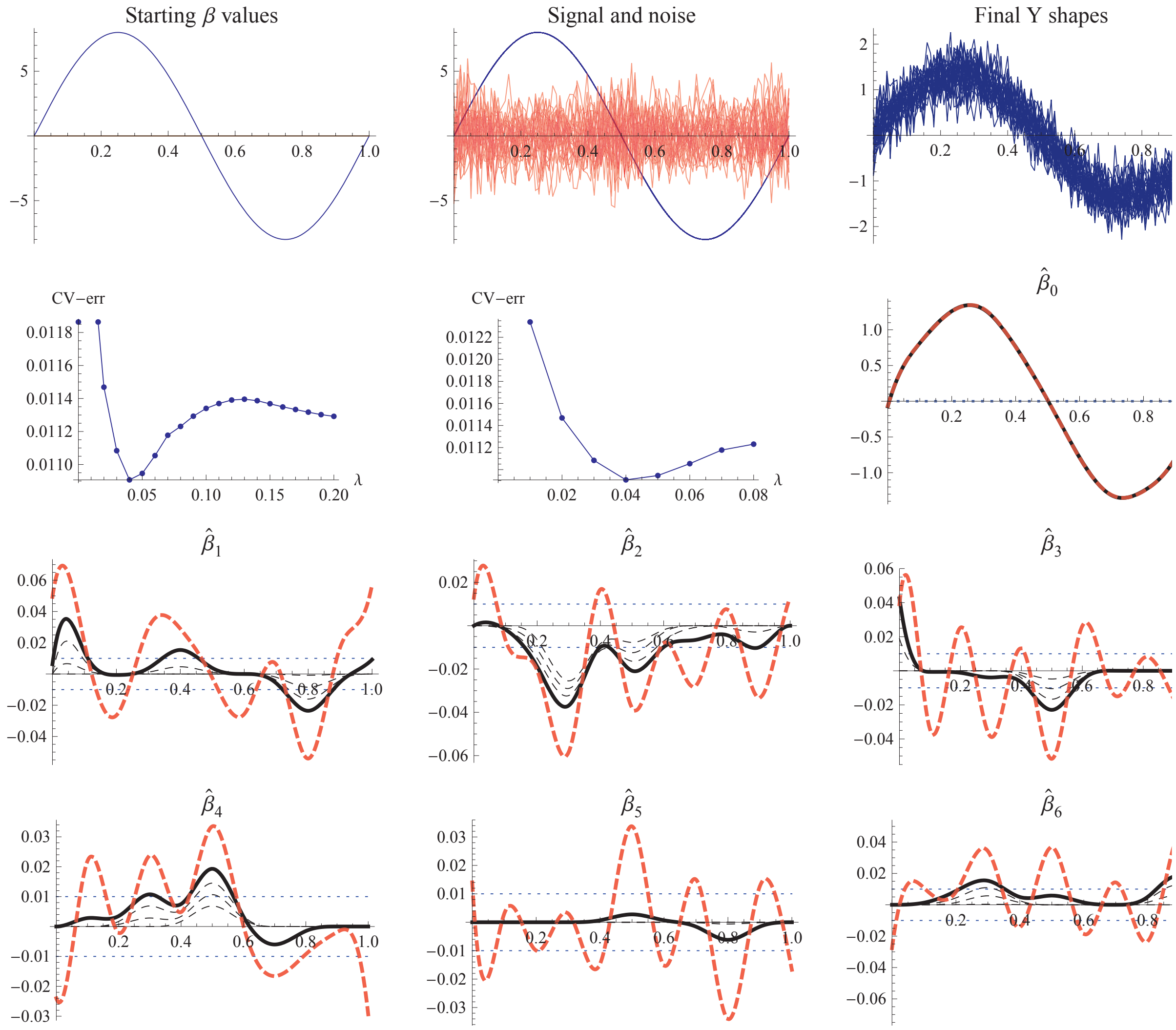
Case 26.

- Number of regressors to select: 0.
- Erorr type: White noise, sd=2. Random seed = 123.
- Lambda for minimum cv-error: 0.1. Other lambdas: 0.11, 0.12, 0.13.



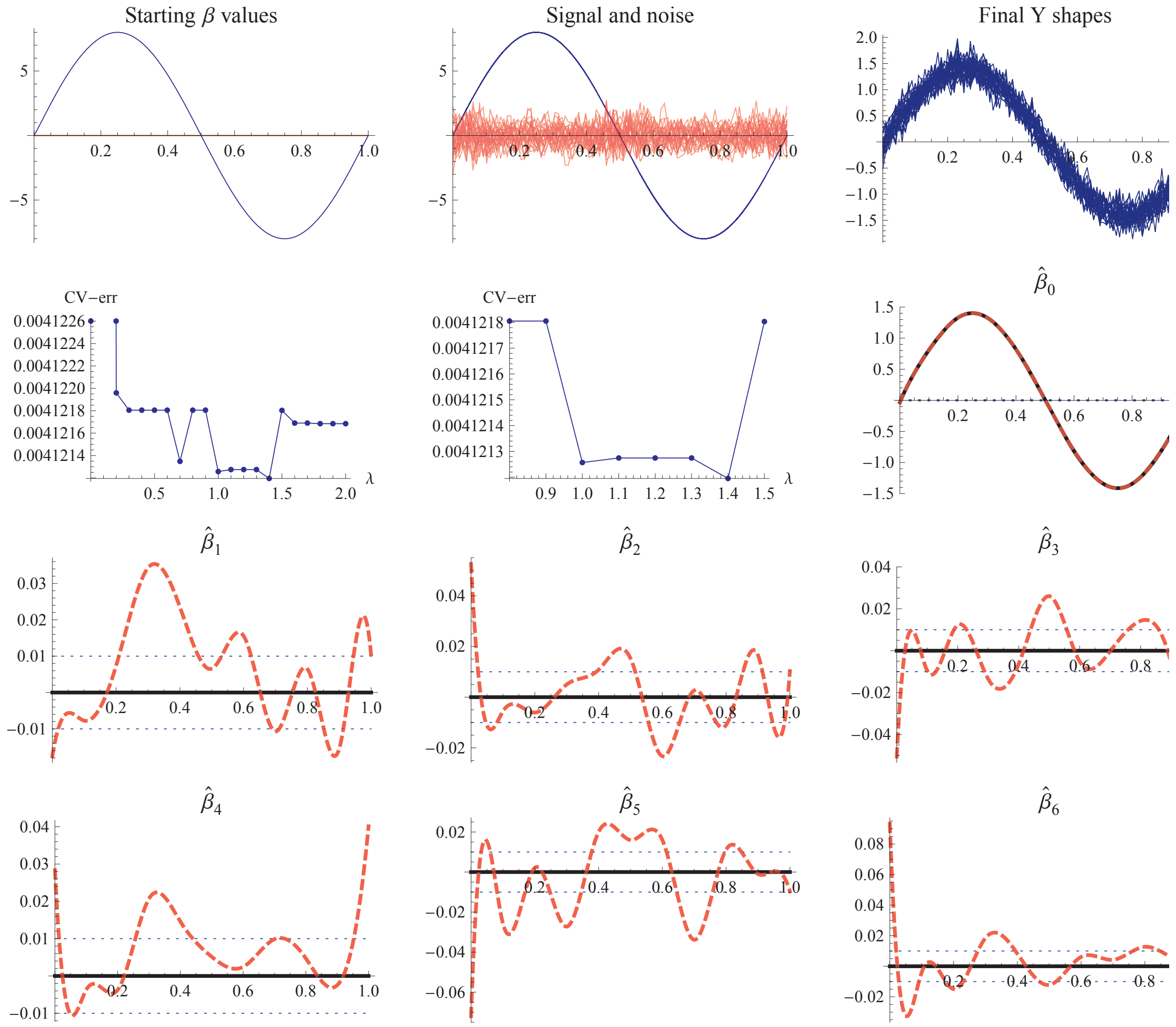
Case 27.

- Number of regressors to select: 0.
- Erorr type: White noise, sd=2. Random seed=153.
- Lambda for minimum cv-error: 0.04. Other lambdas: 0.06, 0.08, 0.1.



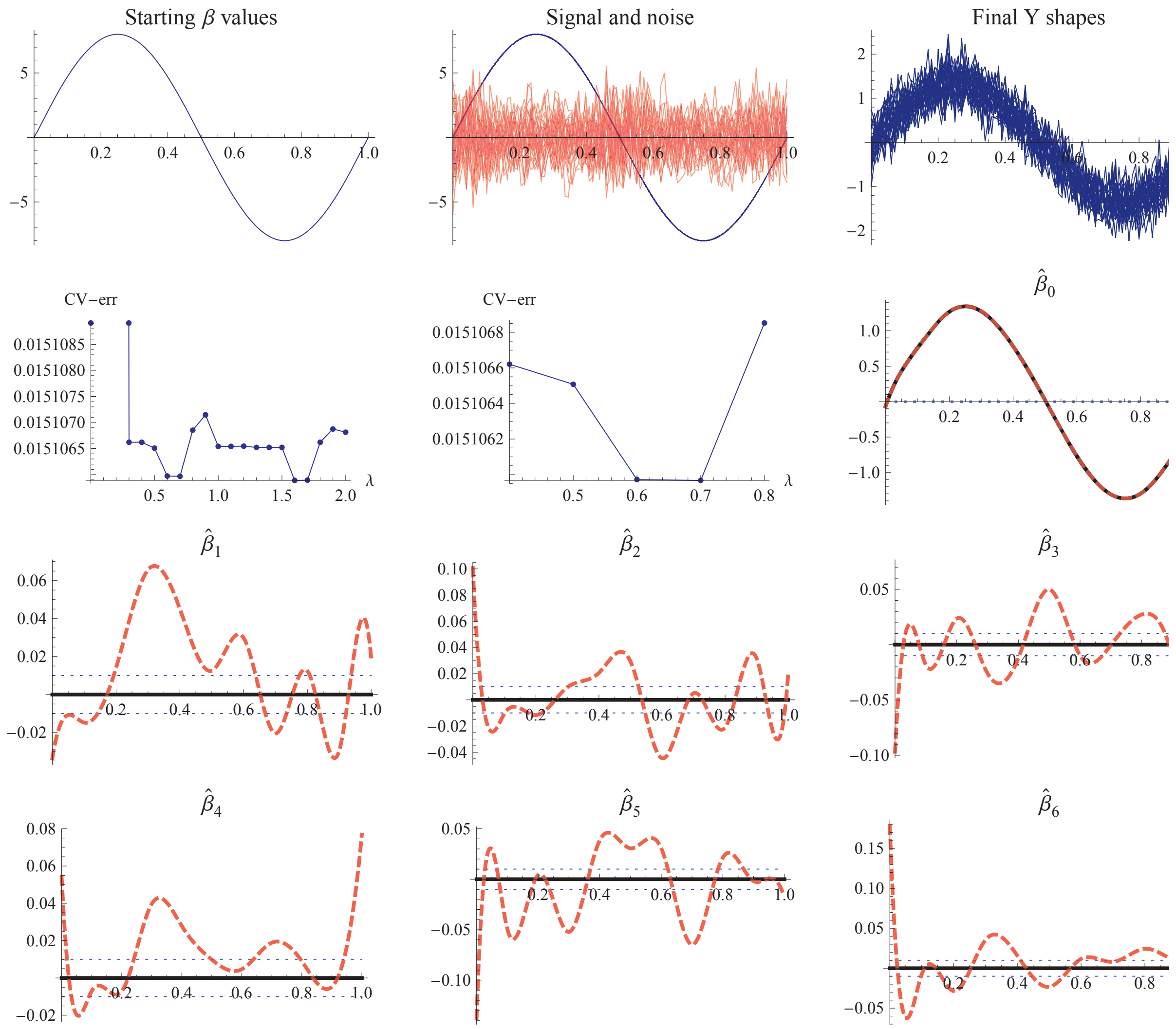
Case 28.

- Numberof regressorsto select:0.
- Erorrtype:Whitenoise, sd=1.Randomseed=1231.
- Lambdaforminimumcv-error1.4.



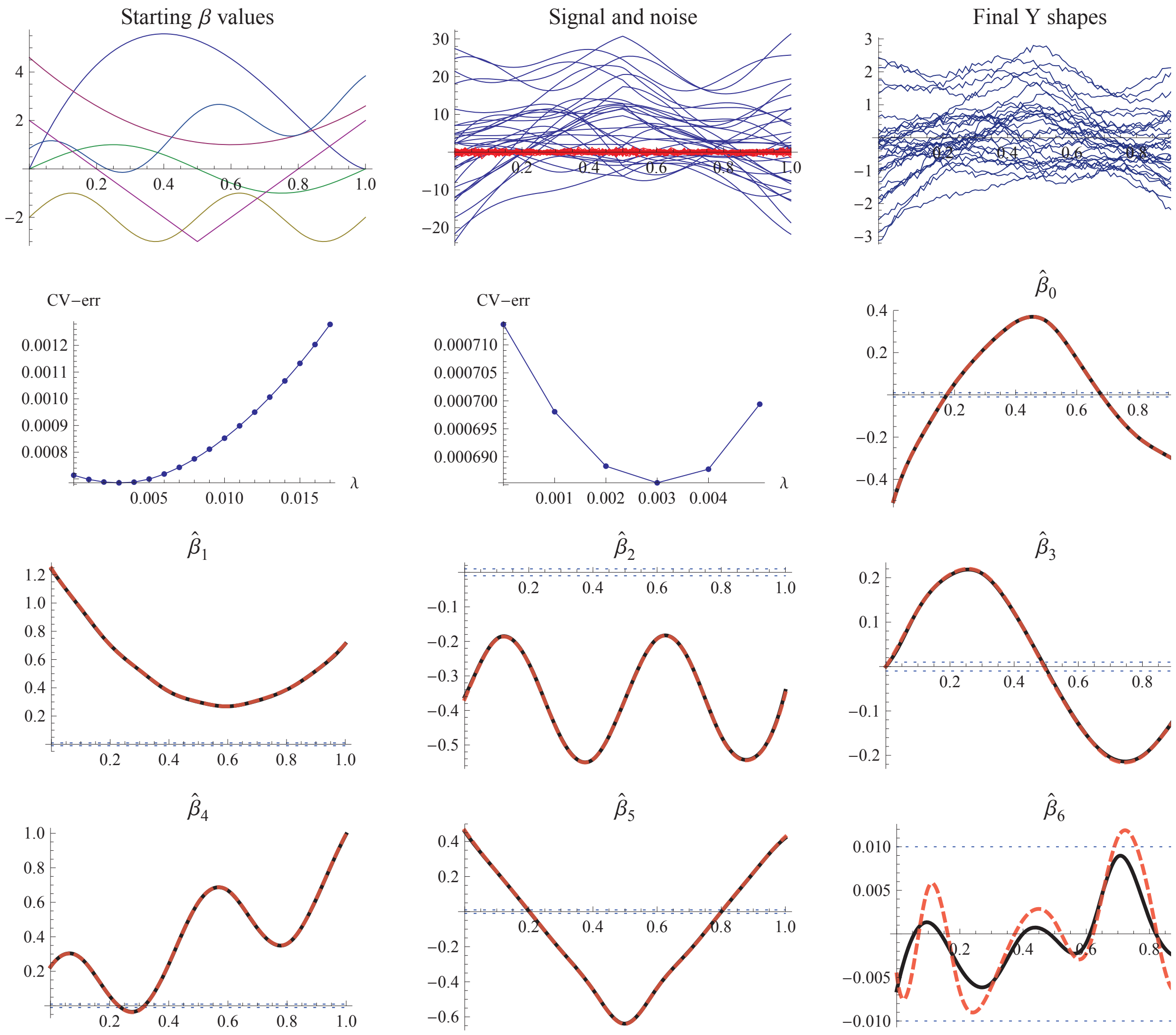
Case 29.

- Number of regressors to select: 0.
- Erorr type: White noise, sd=2. Random seed=1231.
- Lambda for minimum cv-error: 0.6.



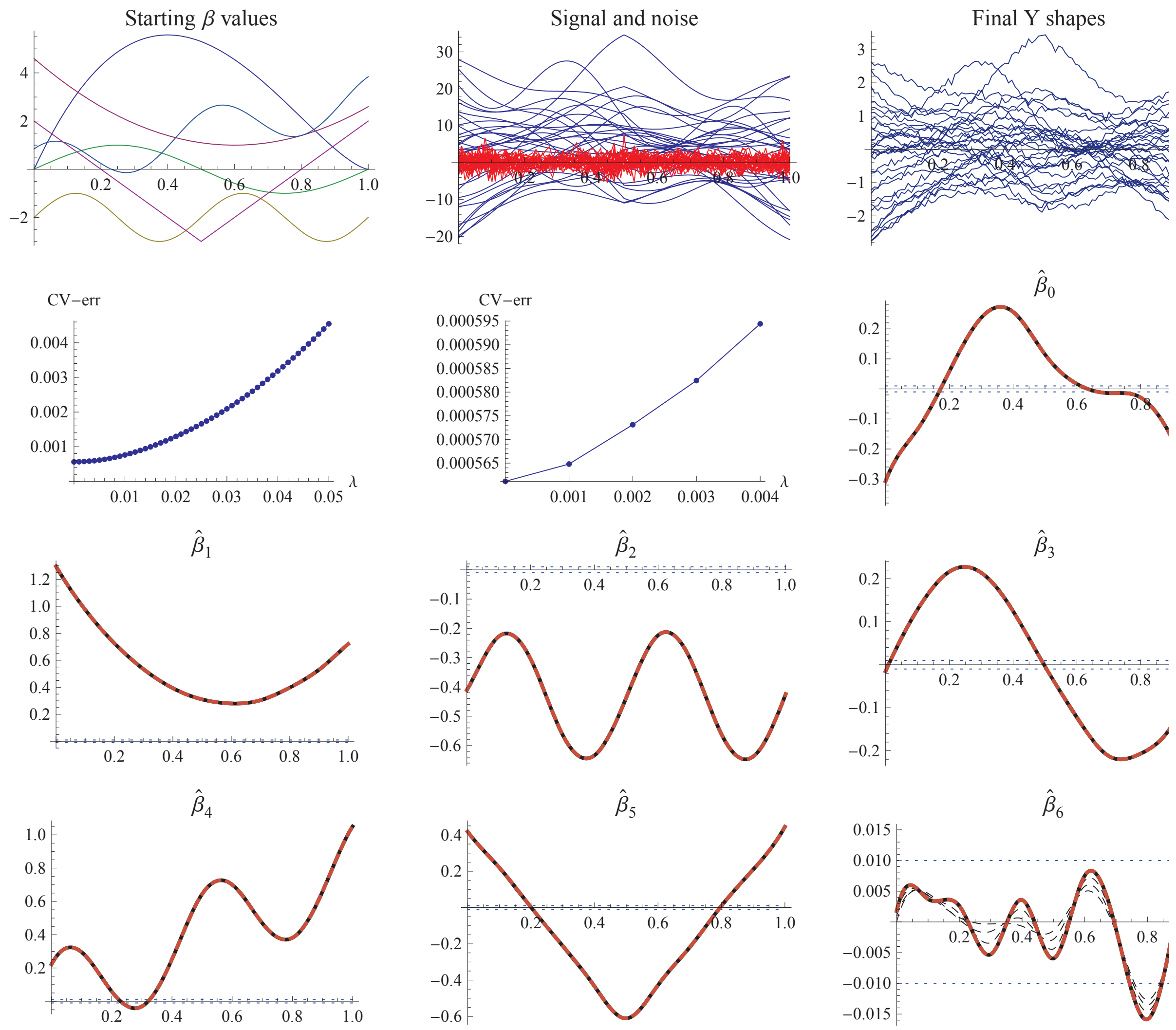
Case 31.

- Numberof regressorsto select:5.
- Erorrtype:Whitenoise, sd=0.5,Randomseed=133.
- Lambdaforminimumcv-error0.003.



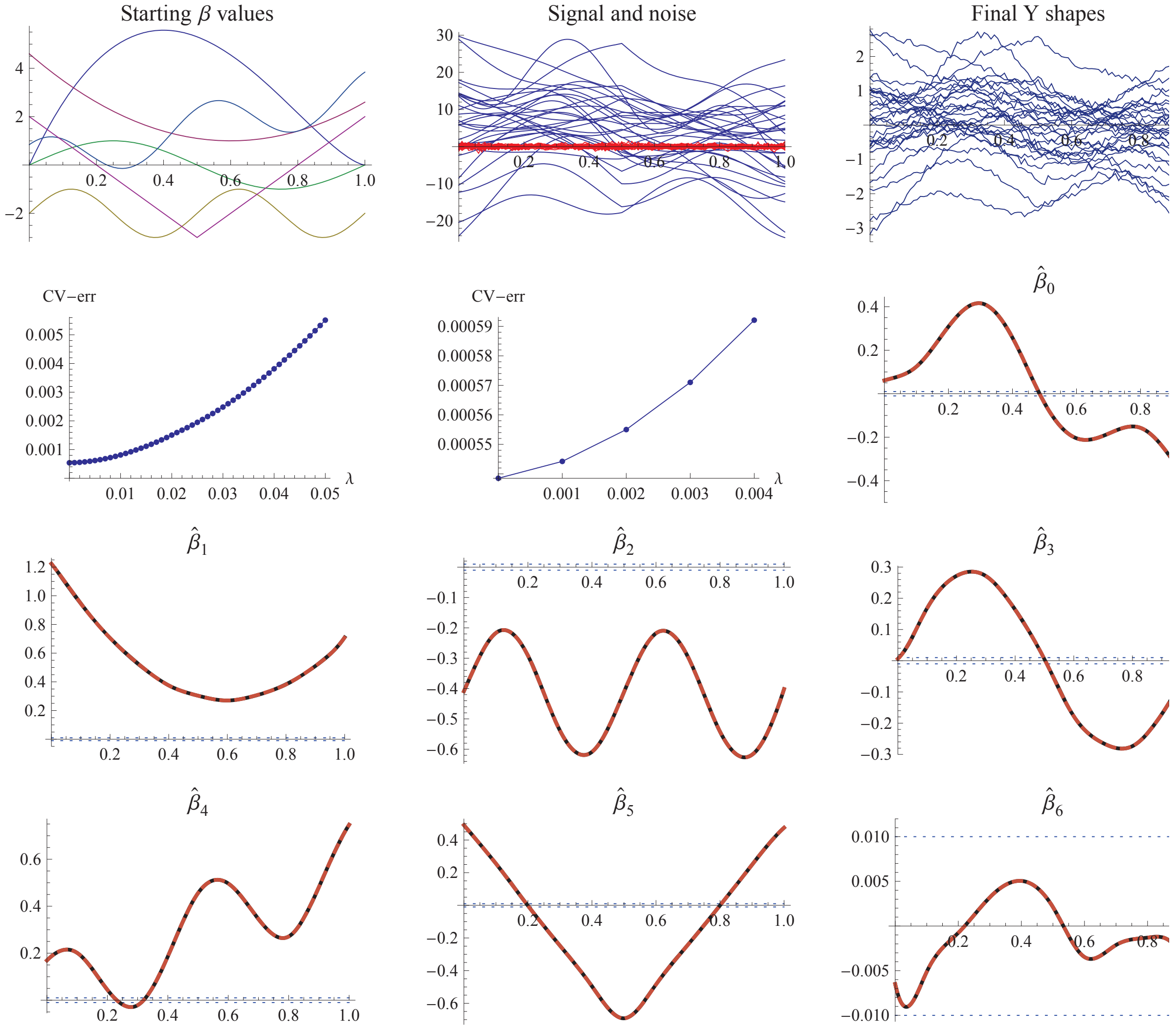
Case 32.

- Numberof regressorsto select:5.
- Erorrtype:Whitenoise, sd=0.5,Randomseed=123.
- Lambdaforminimumcv-error0. Otherlambdas:0.001,0.002,0.003.



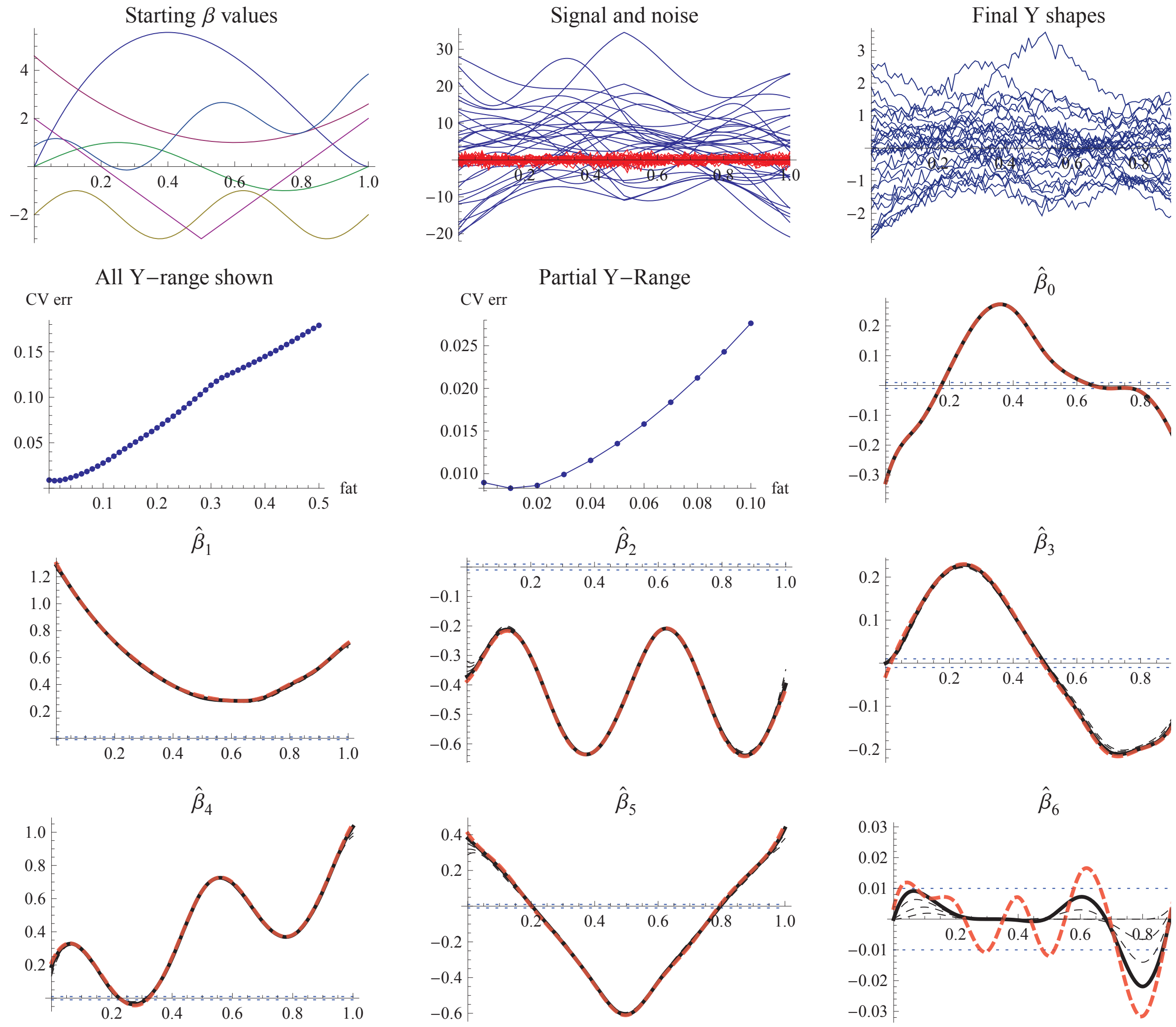
Case 33.

- Number of regressors to select: 5.
- Erorr type: White noise, sd=0.5, Random seed=143.
- Lambda for minimum cv-error 0.



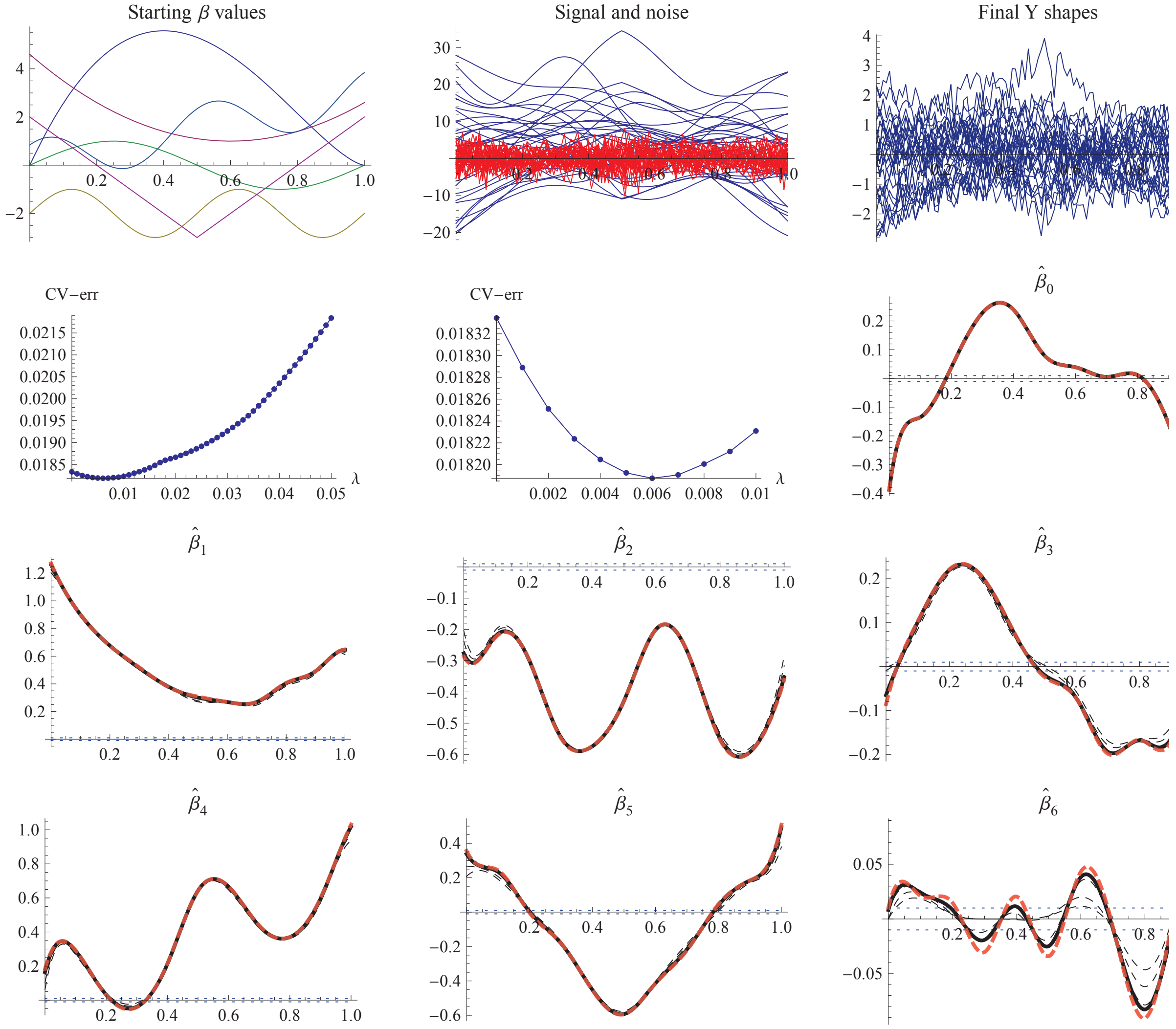
Case 34.

- Number of regressors to select: 5.
- Erorr type: White noise, sd=1, Random seed=123.
- Lambda for minimum cv-error: 0.01. Other lambdas: 0.02, 0.03, 0.04.



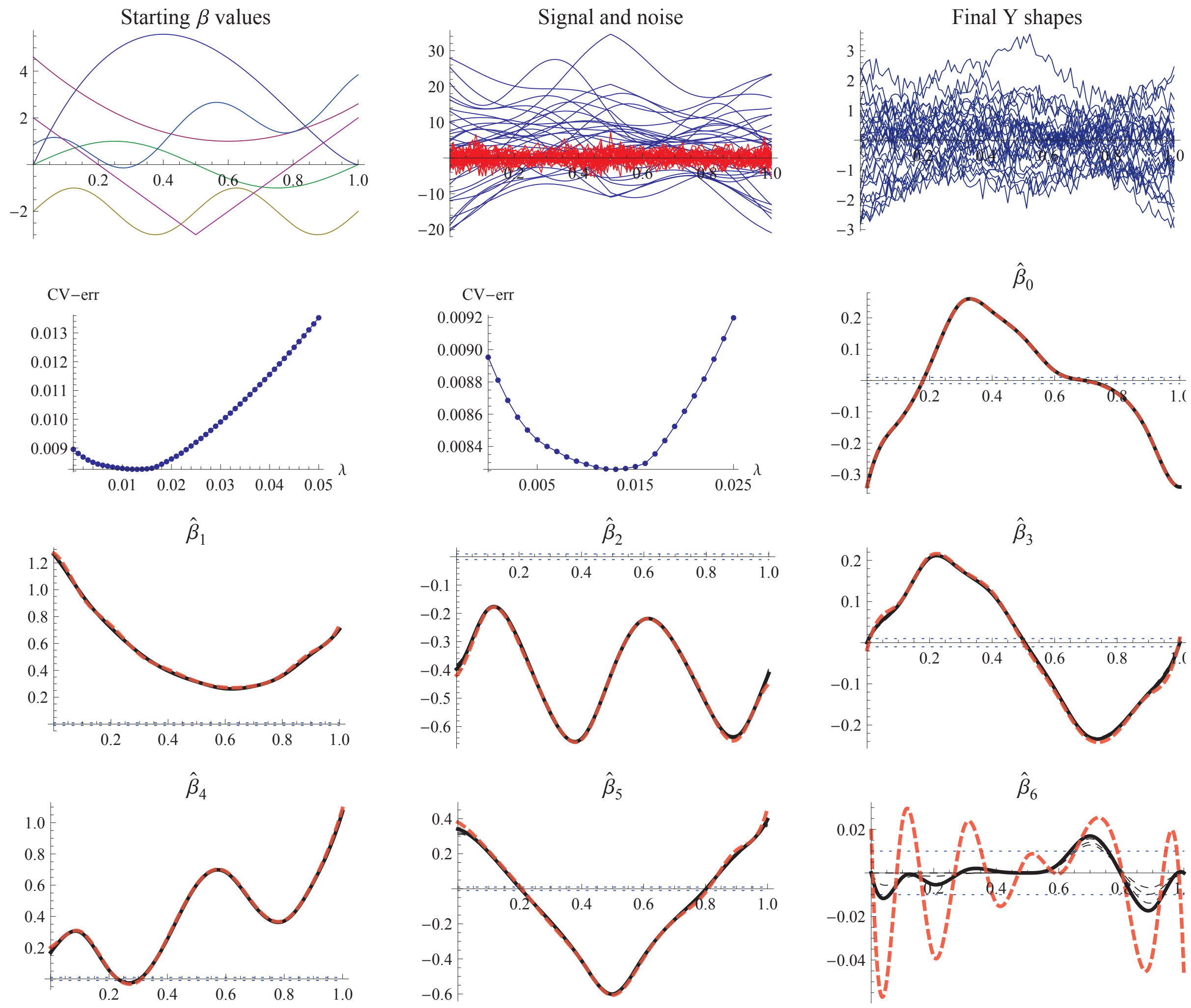
Case 35.

- Numberof regressorstoselect:5.
- Erorrtype:Whitenoise, sd=3,Randomseed=123.
- Lambdaforminimumcv-error:0.006.Otherlambdas:0.01,0.03,0.05.



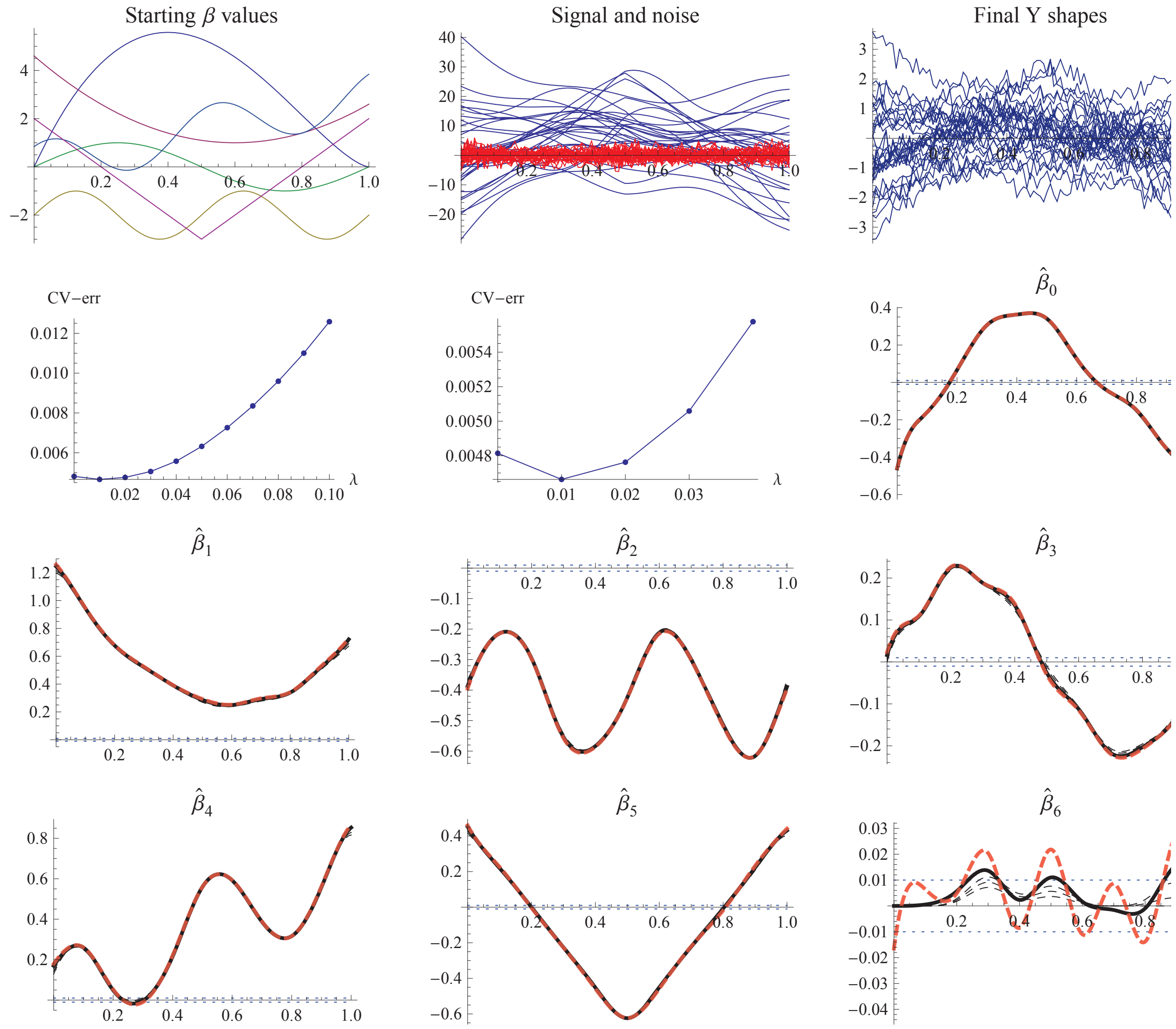
Case 36.

- Number of regressors to select: 5.
- Erorr type: White noise, sd=2, Random seed = 123.
- Lambda for minimum cv-error: 0.015. Other lambdas: 0.018, 0.020, 0.022, 0.025.



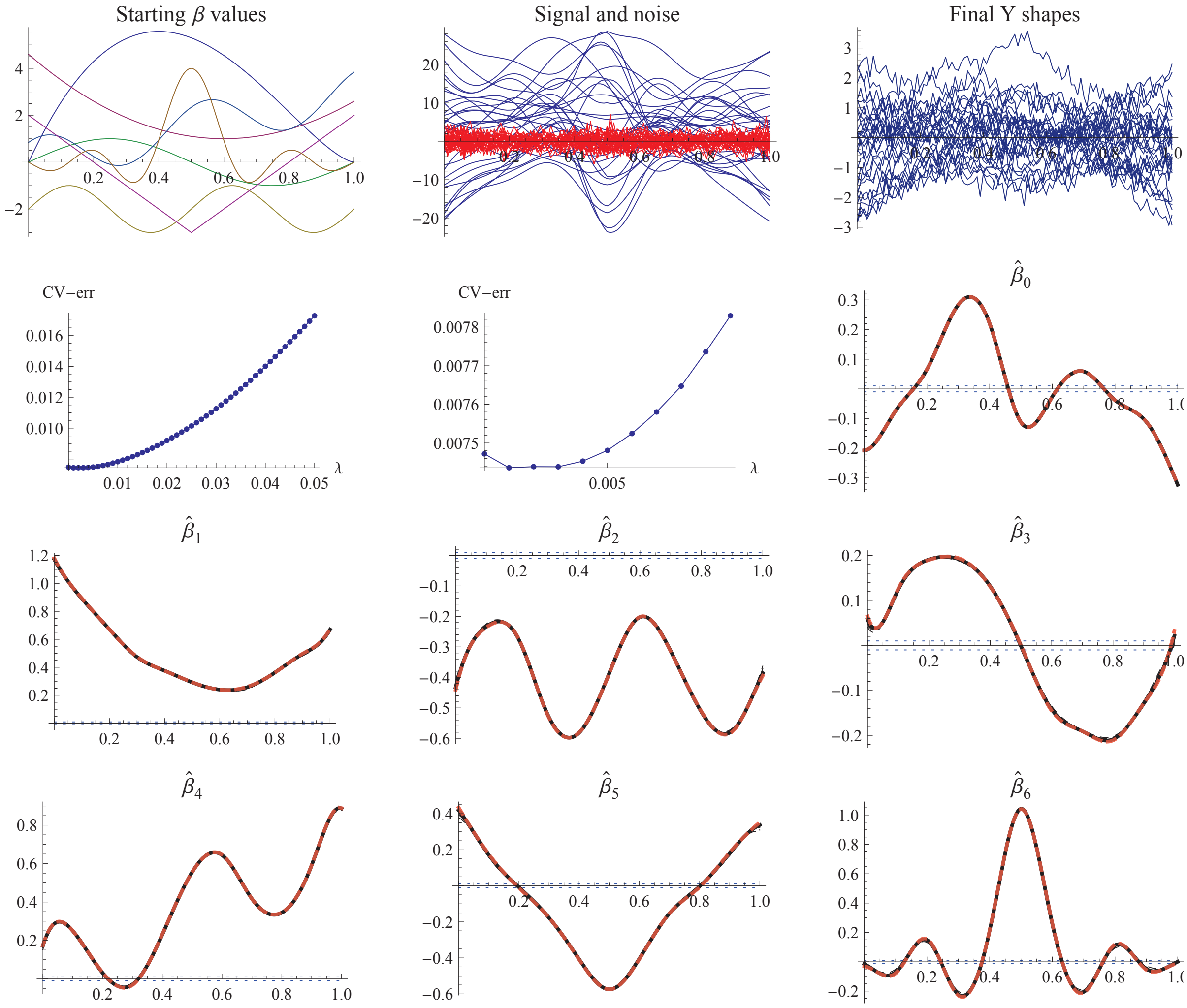
Case 37.

- Numberof regressorstoselect:5.
- Erorrtype:Whitenoise, sd=2,Randomseed=153.
- Lambdaforminimumcv-error:0.01.Otherlambdas:0.02,0.03,0.04.



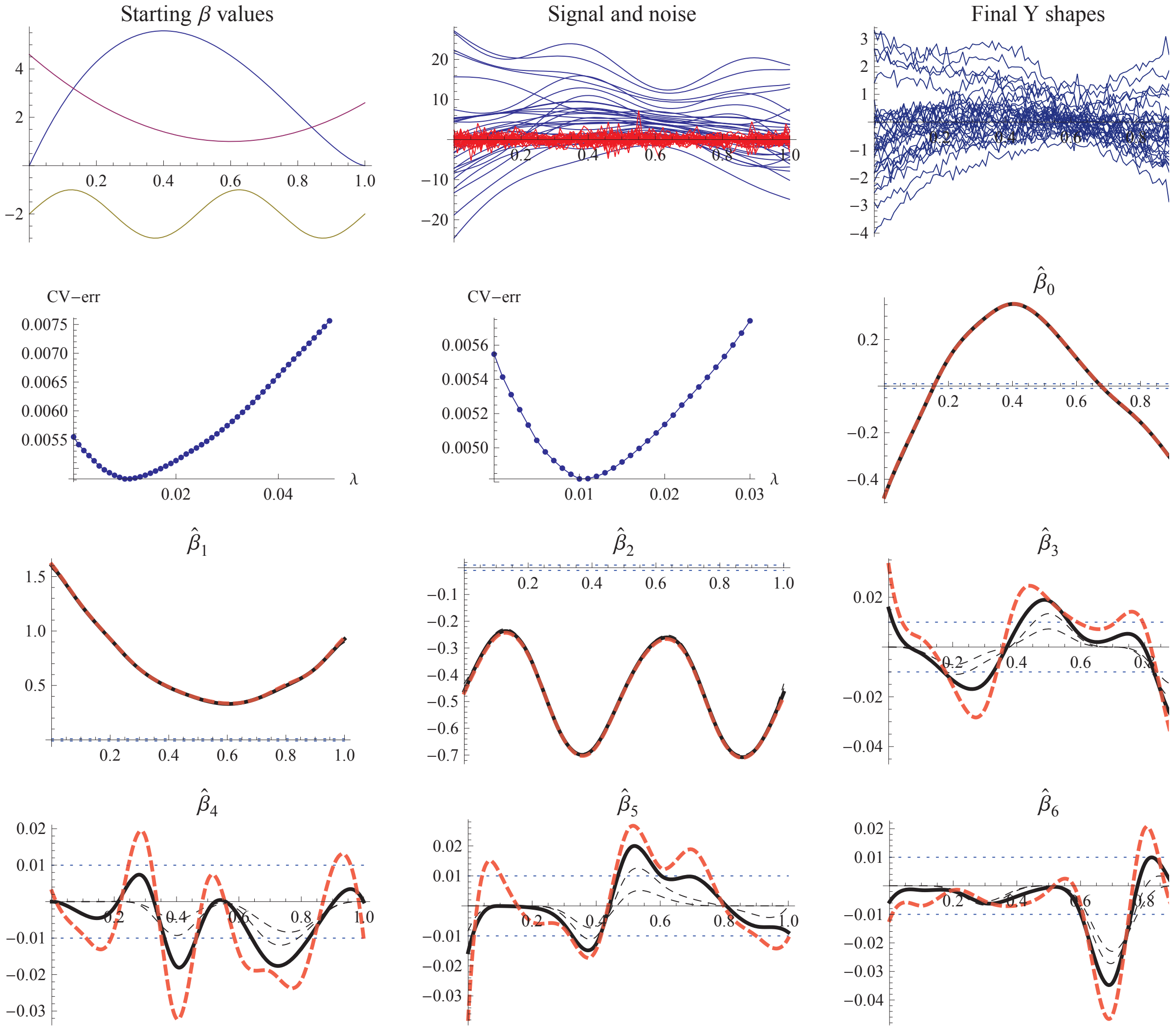
Case 46.

- Number of regressors to select: 6.
- Erorr type: White noise, sd=2, Random seed = 123.
- Lambda for minimum cv-error: 0.003. Other lambdas: 0.006, 0.010, 0.015.



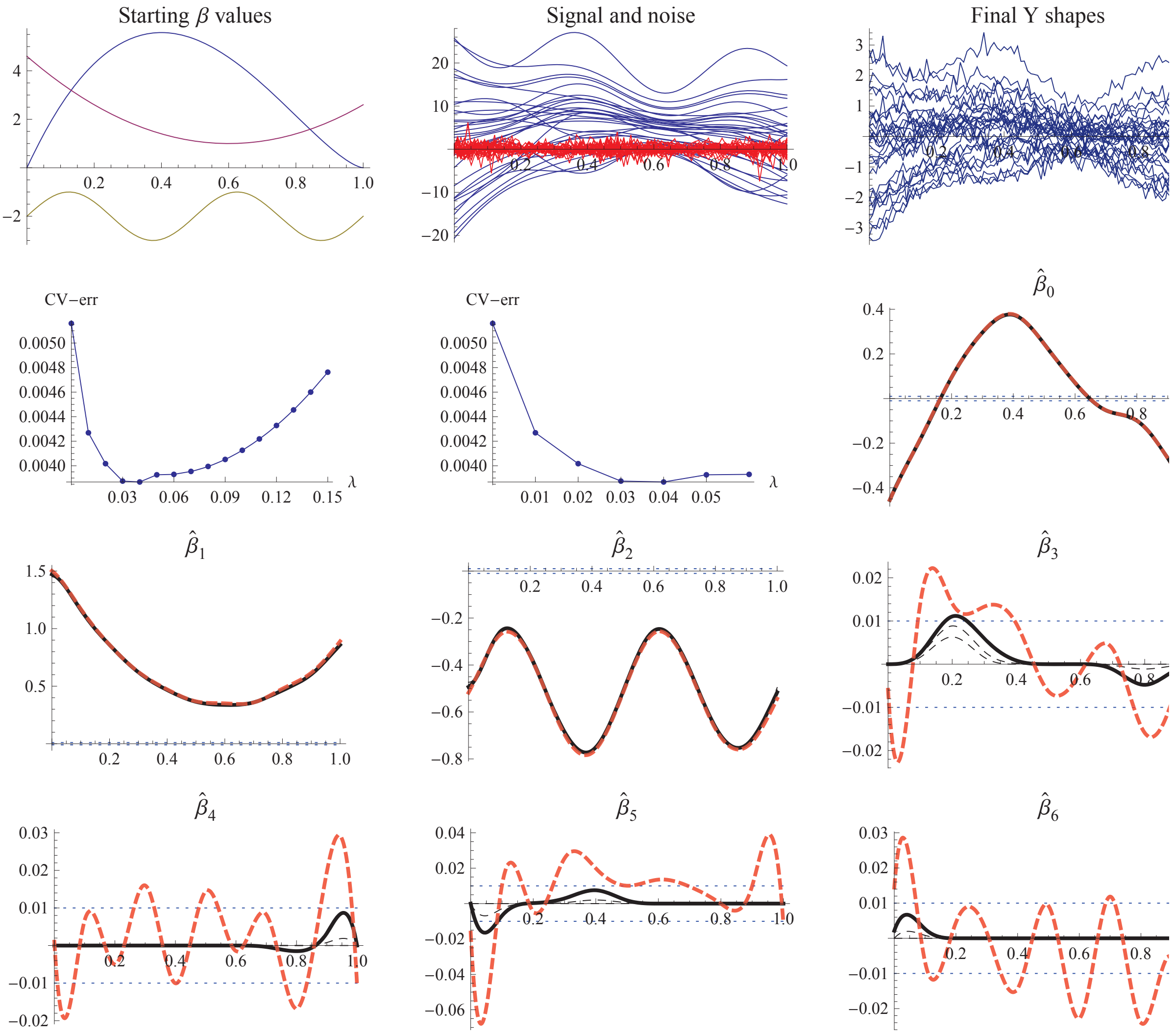
Case 51.

- Number of regressors to select: 2.
- Erorr type: Whte Noise with Student t(6), Random seed = 133.
- Lambda for minimum cv-error: 0.01. Other lambdas: 0.02, 0.03.



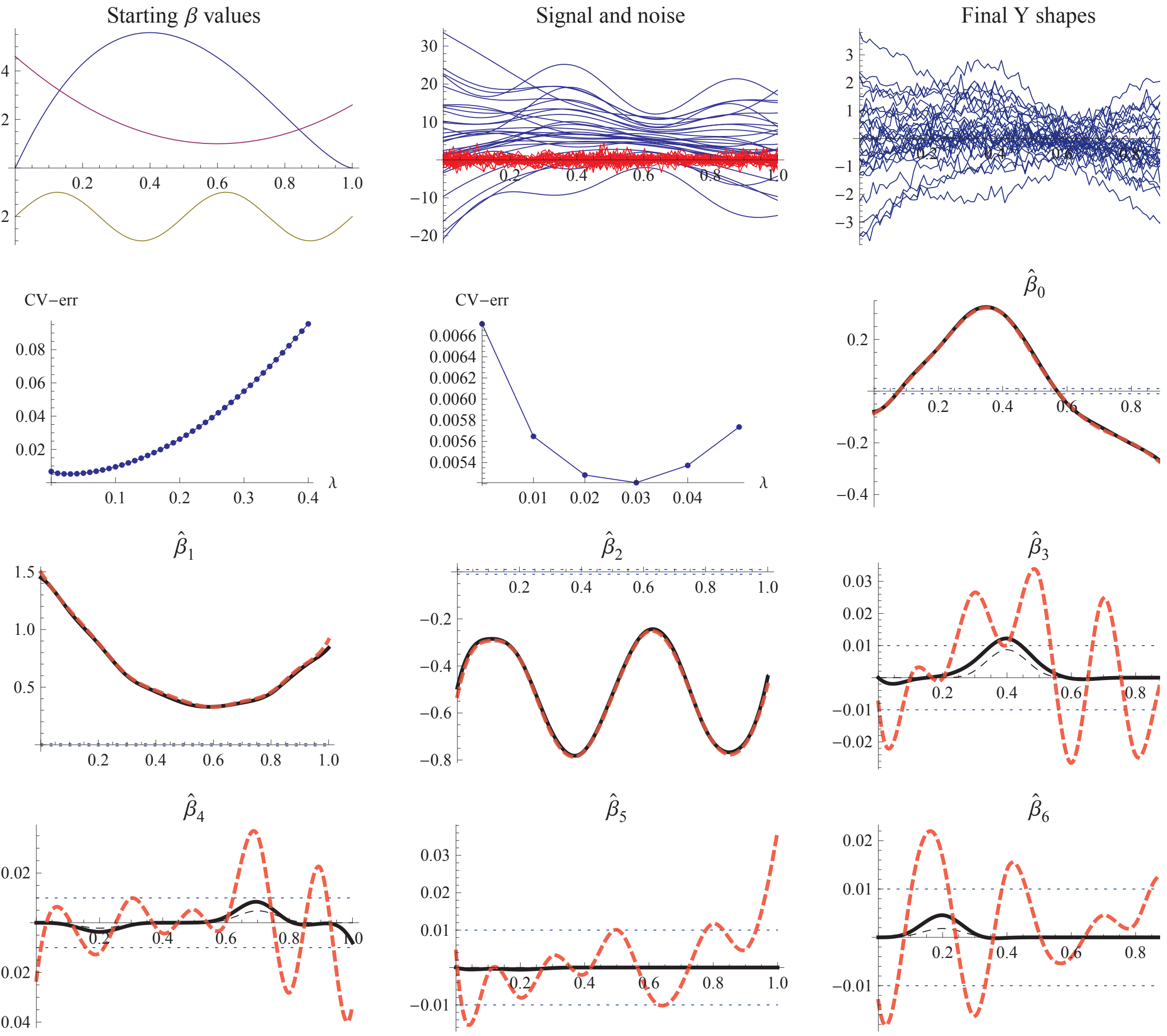
Case 52.

- Number of regressors to select: 2.
- Erorr type: Whte Noise with Student t(6), Random seed = 123.
- Lambda for minimum cv-error: 0.03. Other lambdas: 0.04, 0.05.



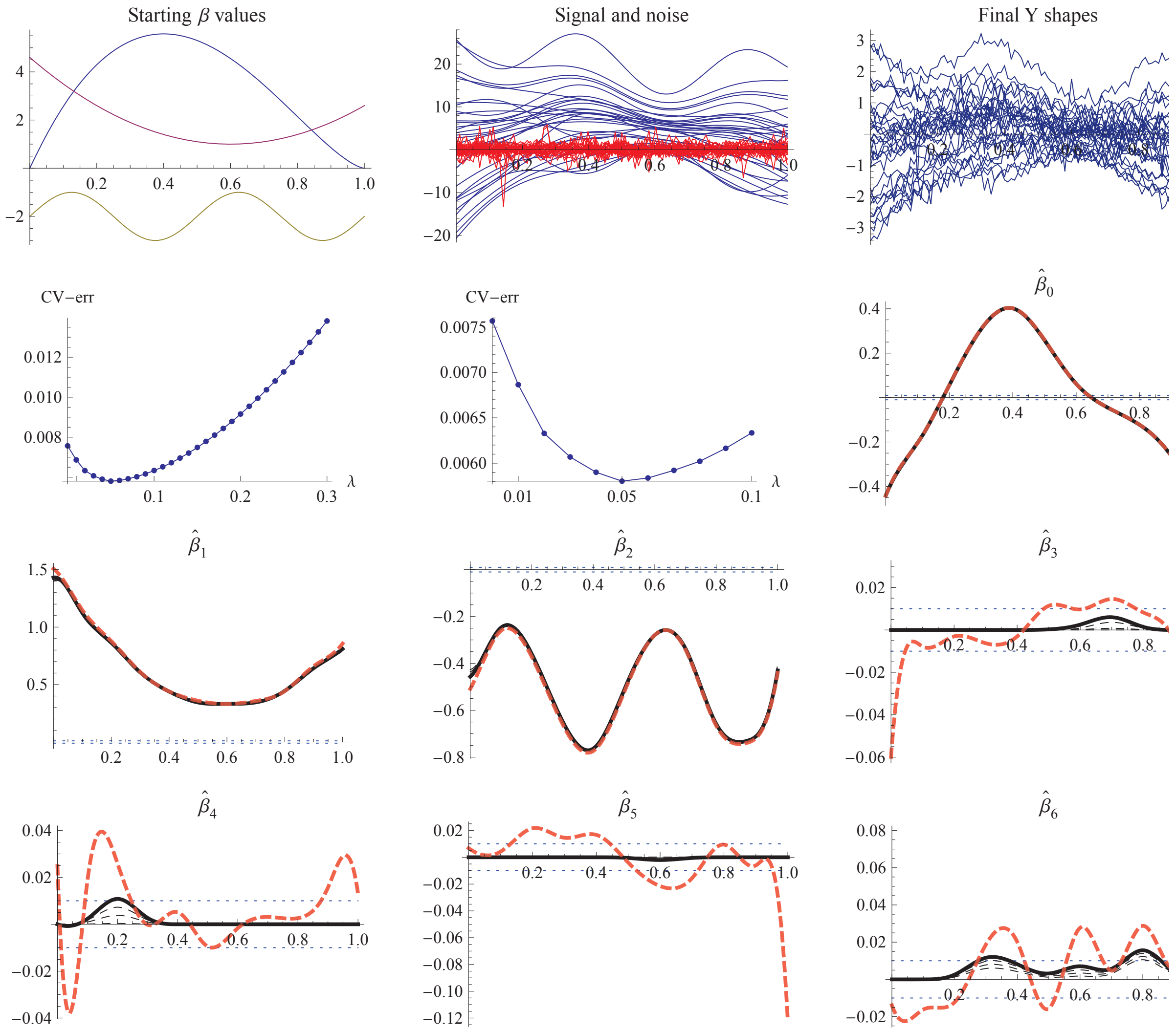
Case 53.

- Numberof regressorstoselect:2.
- Erorrtype:WhteNoisewithStudentt(6),Randomseed=143.
- Lambdaforminimumcv-error0.03.Otherlambdas:0.04.



Case 56.

- Numberof regressorstoselect:2.
- Erorrtype:WhteNoisewithStudentt(4),Randomseed=123.
- Lambdaforminimumcv-error0.05.Otherlambdas:0.06,0.07,0.08.


Starting β values
Signal and noise
Final Y shapes
CV−err
λ
CV−err
λ
$\hat{\beta}_0$
$\hat{\beta}_1$
$\hat{\beta}_2$
$\hat{\beta}_3$
$\hat{\beta}_4$
$\hat{\beta}_5$
$\hat{\beta}_6$