Author: Elena Cano Lázaro

September 2016

Enhancement of a metaheuristic algorithm for trigger error minimization in earthquake events considering geological issues

Trabajo Fin de Máster UOC-URV

Máster en Ingeniería Computacional y Matemática

Directors: Ángel A. Juan, Laura Calvet, Jessica de Armas

Enhancement of a metaheuristic algorithm for trigger error minimization in earthquake events considering geological issues

Summary

Natural risks, such as earthquakes, are high loss events that exceed the financial capacity of the insurance market.

The insurance and reinsurance markets, governments and others, use some financial instruments, called Catastrophe Bonds, to cede risks derived from earthquakes to the capital market. Once an event has occurred, a Post Event Loss Calculation is initiated, in order the investors to know if the principal of the bound should or should not be disbursed. There are different calculation techniques that use physical parameters of the earthquake events, such as location, magnitude and depth of the hypocenter, to determine losses. All these methods are more or less affected by a trigger error, but they have the advantage that they allow insurers to know about the disbursement of their investment in a very short time, which would also redound to a faster reconstruction of the damaged areas.

The aim of this study is to review some of these existing techniques, two statistics and one metaheuristic optimization algorithm, create a model for each of them, propose an improvement of the metaheuristic algorithm, and compare the results given by each technique.

A geological point of view will be given to understand the preparation of the data, to interpret the models, and to explain the metaheuristic improved model.

List of Contents

Chapter 1. Introduction

1.1. Justification and context of the Project: starting point and achievements

When a catastrophic earthquake occurs, it usually causes damages or destruction of buildings, structures and personal properties. But it can also cause injured people or even deaths. Thus, consequences of an earthquake are measured in economic losses and/ or lives.

Once a catastrophe happens, emergency services must act to help the victims, implementing an emergency management model. And after all affected people have received proper cares and their basic necessities, a reconstruction labor must start to restore to citizens their normalcy.

Emergency services economical expenses can be supported by Governments and international aid. However, the amount of money needed for reconstruction may be so huge that the insurance market has no economical capacity to deal with it. This is the reason why financial markets developed few decades ago the securitization of catastrophic events, creating the catastrophe bonds, or cat-bonds.

A cat-bond is a financial instrument used by the insurance and reinsurance market for covering a specific geographical area by means of transferring risks derived from natural catastrophes to the capital market, who invests in these bonds. These bonds are, therefore, high risk investments that release countries from losses which could not assume on their own.

When issuing a bond, a special purpose vehicle (SPV) is created. This is a new subsidiary company with whom the sponsor (usually a reinsurance company) will contract the cat-bond. The contract will specify a threshold of losses such that, if it is surpassed, the principal of the bond, say, its nominal value, will be disbursed, and, if it is not surpassed, the principal will not be disbursed.

And here comes the main problem. In case an earthquake occurs, the losses should be known as soon as possible so that the disbursement for reconstruction is available in a short term. But losses are given by the Seismic Risk. And the Seismic Risk is the combination of three concepts:

- ***Seismic hazard***, this is, the probability of occurrence of an earthquake, available in the geological services of most of the countries.
- ***Exposure***, this is, the existence or not of constructions and population in the geographical area.
- ***Vulnerability*** of the constructions in the geographical area, which refers to the fragility of constructions, this is, the possibility of constructions of being affected.

Probability and exposure are relatively easy to be determined, however, vulnerability is not known in most of the cases, and it requires a detailed study of each of the buildings and structures, which could take a very long time, not available in case of a catastrophe.

A solution for this is to develop a method that calculates whether payment should be made or not, taking into account the physical parameters of the earthquake. This can be done by analyzing historical registers of earthquake events, where physical parameters, as well as losses, have been registered. As mentioned before, there are different calculation techniques that use physical parameters such as location of the earthquake, magnitude and depth of the hypocenter, to determine losses. However, since all these methods are more or less affected by a trigger error, it is important to find a method that minimizes it.

The starting point of this study has been to review some of the techniques that automatically calculate losses. Two of them are statistic methods – logistic regression and classification trees –, proposed by *de Armas, Calvet, Franco, Lopeman, Juan. (2015)* [2], and another one is a metaheuristic method, proposed by *Franco (2010)* [1].

Next step has been to reproduce all the calculation methods described in the bibliographical references [1] and [2] using Matlab software. Special mention has to be made here to the metaheuristic model, which has been created from scratch, constructing one by one all steps related by *Franco (2010)* [1].

The aim of this study is to propose a metaheuristic method that improves the results given by the genetic algorithm proposed by *Franco (2010)* [1]. To this end, new metaheuristic model has been implemented. A comprehensive comparison has been performed among by *Franco (2010)* [1], the 'improved genetic algorithm' and the statistical models. As a result, some conclusions have been drawn and a few interesting lines of future research have been identified.

1.2. Basic concepts

a) Calculation mechanism: the trigger mechanism

The three mentioned approaches, as proposed in papers [1] and [2], have been selected due to the fact that all of them consider the same input and output parameters, and, in consequence, they can be easily compared.

The payment mechanism consists in a ***binary calculation method*** which determines whether the cat-bond should trigger (result = 1) or not trigger (result = 0), depending on some physical parameters of the earthquake. This method reduces the possibility of manipulation of the parameters by any party, increasing transparency and eliminating moral hazard. Physical parameters usually registered in all earthquakes database are: coordinates, hypocenter depth, magnitude and intensity.

- The ***coordinates*** are represented by X and Y, which correspond to the ***longitude*** and the ***latitude*** of the earthquake epicenter, respectively.
- The ***hypocenter depth*** refers to the depth at which the origin of the earthquake occurs. The shallower the depth is, the more violent is the earthquake; the deeper it is, the more far the effects are felt (considering in both of them the same type of materials).
- The ***magnitude*** of an earthquake is a measure to quantify the energy of the earthquake, calculated by the seismometer registers.
- The ***intensity*** is a measure to quantify the effects of an earthquake on the Earth's surface, humans, objects of nature, and man-made structures. These measures are given in scale intensities based on observations that go from not-felt to total destruction. However, since effects of the earthquake depend on the vulnerability of the population, structures and buildings, its value will vary depending on the characteristics of the place where the event occurs, and cannot be taken as an objective and invariable parameter. Due to this reason, this parameter will not be taken into account for determining the physical parameters of the earthquake.

According to this, the set of earthquake events addressed in this study is characterized by **three** different **parameters**: ***location coordinates (X and Y)***, ***magnitude (M)*** and ***depth of the*** earthquake ***hypocenter (D)***. Thus, an event i will be characterized by its coordinates x_i and y_i, its magnitude m_i, and its hypocenter depth d_i.

Depending on the value of these physical parameters, a binary trigger mechanism can be designed in order to determine if the earthquake will trigger or not trigger the cat bond. This response is represented by parameter B_i'. $B_i'=1$ means that the cat bond has to be disbursed. $B_i'=0$ means that the cat bond will not be disbursed.

In an ideal behavior, the earthquake should only trigger the cat bond if losses produced by the event surpass the monetary threshold stablished in the insurance contract. Then, to know if this mechanism is properly designed, it needs to be compared with historical events where losses have been registered. These events will be also characterized by a parameter l_i, which represent the economic losses. In this case, a new variable B_i is considered, which represents whether event losses have in fact surpass the monetary threshold (L) or not. If they have surpassed it, $B_i=1$, if they have not, $B_i=0$.

$$B_i = \begin{cases} 0, \; if \; l_i < L \\ 1, \; if \; l_i \geq L \end{cases}$$

The objective of the cat-bond trigger error minimization techniques is to minimize discrepancies between B_i' and B_i. Any discrepancies are measured as 1:

$$E^+ = \sum_{\forall i} \begin{cases} 1, \; B_i^{'} > B_i \\ 0, \; B_i^{'} = B_i \end{cases}$$

$$E^- = \sum_{\forall i} \begin{cases} 1, \; B_i^{'} < B_i \\ 0, \; B_i^{'} = B_i \end{cases}$$

And the sum or errors would be the sum of all the positive and the negative contributions:

$$E = E^+ + E^-$$

Taking this into account, a database of historical earthquake events including above mentioned physical parameter, as well as losses generated, is needed to design algorithms of models with each technique. Physical parameters can be easily found in the geological or seismic services of many countries around the world. However, due to the fact that there are almost no registers in the first half of the XX century, and up to the 1980's they are limited, a complete database including not only physical, but also monetary losses are quite difficult to be found. This fact will be mentioned again later.

b) Geological concepts

What exactly is an earthquake?

Earthquakes are ground movements generated by a readjustment of the Earth's crust at both sides of an active geological fault. A geological fault is a fissure on the Earth's crust. The crust areas situated at each of the sides of the fault receive the name of blocks (or lips), which will remain immobile if the fault is inactive, or could slip relatively along the fault plane if the fault is active. Faults separate each block along a plane, called the fault plane. This plane is defined by its strike (φ, the azimuth of the fault from north where it intersects a horizontal surface) and dip (δ, the angle from the horizontal), and therefore, the fault plane could be either inclined at an angle, either vertical.

The block positioned over the fault plane would be called the 'hanging wall', and the block positioned under the fault plane would be called the 'foot wall'.

Depending on direction of the stresses, blocks would be placed different along the fault plane, so that several types of faults can be distinguished:

- ***Transcurrent faults (or strike-slip faults)***: there is no hanging wall or foot wall, both blocks are at the same level, since they are caused by shear stresses. They are illustrated on Figure 1.
- ***Normal faults***: the hanging wall moves down relative to the footwall. This fault motion is caused by tensional forces and results in extension (Figure 2)
- ***Reverse faults and overthrusts***: the hanging wall moves up relatively to the footwall. This fault motion is caused by compressional forces and results in shortening. (Figure 3).

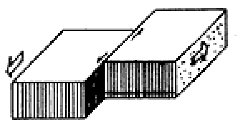


Figure 1. Transcurrent fault. Source [11]

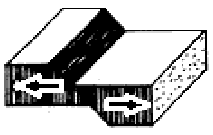


Figure 2. Normal fault. Source [11]

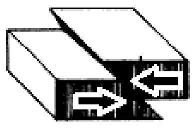


Figure 3. Reverse fault. Source [11]

When blocks move one with respect to the other, the stresses are supported by the ground till a rupture is produced. The position of the initial point of a seismic rupture is called ***hypocenter***, and the point of the Earth's Surface located over it is called ***epicenter***. This rupture is propagated to the nearby materials, so that there is a volume of rock which movement causes the seism, and where the fault goes through. This volume is called the ***seismic focus***.

Thinking in how the ground at both sides of the fault behaves when an earthquake happens, it is easy to visualize that the ground located towards the direction of the movement will suffer compression forces, the one located in the opposite direction will suffer dilatation forces, and the one located in a direction perpendicular to the movement will suffer shear forces. These interactions produce two types of seismic waves:

- ***P waves*** (or primary waves), which produce the compression and dilatation of the ground.
- ***S waves*** (or secondary waves), which are slower than previous ones and produce shear forces on the ground.

The interaction of waves P and S with the Earth's surface produce another type of waves, called superficial waves, which are the most destructive ones in the earthquake events, and are also divided in two:

- ***Rayleigh waves***, the slowest ones, that produce vertical and horizontal movements on the ground surface, parallel to the direction of the movement. They are caused by P and S vertical waves.
- ***Love waves***, whose velocity is between the S and the Rayleigh waves, and which produce horizontal movements towards the propagation direction. They are caused by S horizontal waves.

Thus, depending on the fault strike and the direction of the force, the pattern of distribution of waves S and P will vary, and the compression and dilatation zones will also vary. Consequently, this will condition the type and distribution of superficial waves, that will cause the damages.

This pattern of distribution of waves S and P is called the ***focal mechanism***. The focal mechanism could also be defined as a graphic representation of the possible solutions for the breakup that a seism originates, and the configuration of the tectonic forces of the area, or, in other words, as a set of information that describes the possible mechanism of the fault in the focus where energy is liberated in the form of seismic waves.

The compression and dilatation zones – or the focal mechanism – caused by faults are commonly represented in what is known as 'beach balls', where dark colored sections represent compression, and light colored sections represent dilatation. Beach balls for each type of fault are represented in the following figure:

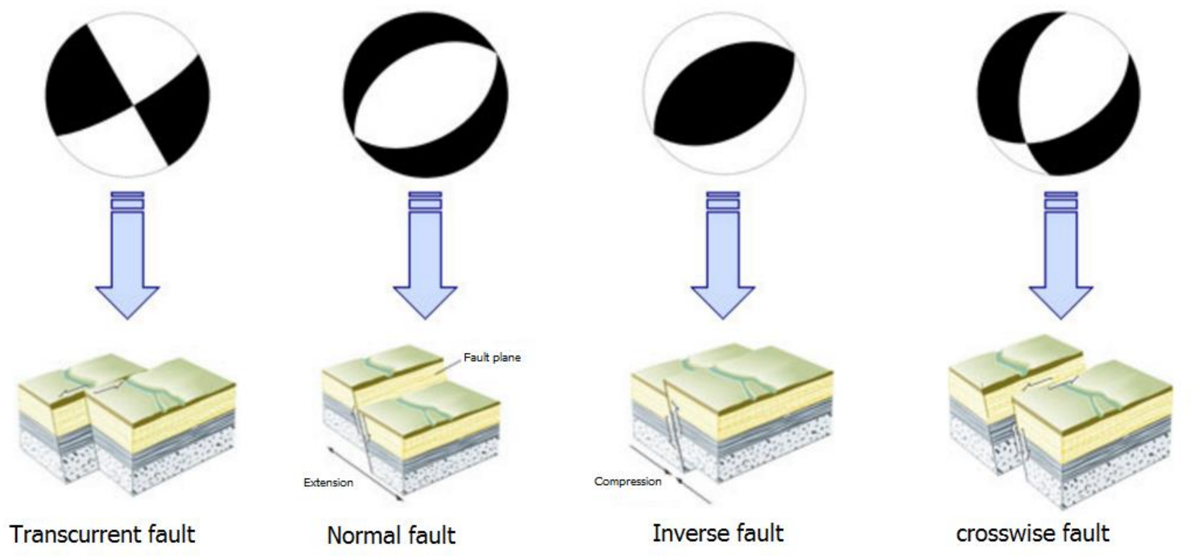


Figure 4. Representation of the most common focal mechanisms and their generating faults. Source: FUNVISIS (Fundación Venezolana de Investigaciones Sismológicas)

But the characteristics of the dynamic effects of an earthquake do not only depend on the focal mechanism. They also depend on the ***energy liberated in the hypocenter*** (measured by the magnitude), the ***distance to the hypocenter***, the ***characteristics of the ground*** where the waves propagate – especially those referred to the most superficial ones –, and the existing ***constructions***. Effects would be more destructive the higher is the energy liberated, the closer is the area to the hypocenter, the softer the soil is and the weaker the constructions are.

Another thing must be taken into account: due to the fact that faults are caused by tectonic stresses, crust deformations are associated to these faults. ***Normal faults are associated to gravens*** or big depressions, and ***reverse faults and overthusts are associated to mountainous areas***. The direction of the depressions and the mountains is the same as the direction of the main fault, since deformations and break ups are controlled by the same tectonic effort. At the same time, geographical features control in many cases the different types of soils. For example, depressions might contain a water course in the deepest part that will deposit flood deposits in its lower course; colluviums will be deposited in the mountainsides; and so on…

And additionally, geographical features could also constitute barriers for the human settlements.

All this means that vulnerable areas are not distributed in a disperse manner, but concentrated, and, moreover, the focal mechanism affecting them will be different in disperse areas and similar in nearby areas.

In summary, ***effects of an earthquake would be conditioned by the focal mechanism, the vulnerability of the area, and the arrangement of the geographical elements in the area***, which would likely follow the main fault direction.

c) Statistical learning

Statistical learning refers to a vast set of tools for understanding data. Imagine there is a set of predictor parameters $X_1, X_2, \ldots, X_p$ - physical parameters of the earthquake in this study –, and a quantitative response Y is observed; the objective of statistical learning is to find a function f that expresses the relationship between the predictor parameters and the response, such that:

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

being ϵ a random error term, which is independent of X and has a mean of zero.

The two statistical tools that are being used in this study are ***Binary logistic regression*** and ***Classification trees***. These are explained in the following two subsections.

c) i) Binary logistic regression

Logistic regression models are statistical methods that show the relation between:

– a dependent variable, which in current study will be a binary variable.
– one or more independent or explanatory variables.

The study of this relation produces two main outcomes:

– To quantify the significance of the relation between each independent variable and the dependent variable.
– To classify the individuals in one of the two binary categories of the dependent variable (0 and 1 in present study).

Logistic regression is an instrument that can be used either to explain, either to predict behaviors of new variables and their results. If it is used in an explanatory way, dependent and independent variables are gathered from observations. In case it is used in a predictive way, a fictitious dependent variable (0 or 1 in present study) is chosen, and the model would predict the probability of the fictitious variable to happen.

Having a binary dependent variable Y, and independent variables X_1, X_2, ..., X_k, a linear regression expresses this relation in the form:

$$Y = f(\beta_1 + \beta_2 X_2 + \cdots + \beta_n X_n) + u \quad (1)$$

Where β coefficients are the parameters that measure the influence of the independent variables on the dependent one, u is a disturbance term, and f is a real function that depends on the linear expression $\beta_1+\beta_2 X_2+\cdots+\beta_n X_n$.

This form will comply with the following expression:

$$E[Y] = P(Y = 1) = f(\beta_1 + \beta_2 X_2 + \cdots + \beta_n X_n) + u \quad (2)$$

Meaning that the success probability (probability of Y being 1), is equal to a function that depends on the combination of a number of independent variables.

Two models that can be used to explain or predict the behavior of a binary dependent variable are Logit model and Probit model.

Logit model uses the logic distribution for function f :

$$f(z) = \frac{\exp(z)}{1 + \exp(z)} \qquad (3)$$

Substituting in (2), the probability of success will be given by:

$$E[Y] = P(Y = 1) = \frac{\exp(\beta_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}{1 + \exp(\beta_1 + \beta_2 X_2 + \cdots + \beta_n X_n)} \quad (4)$$

Whereas in Probit model (also called Normit), f is the distribution function of a standard normal in Probit model (also called Normit):

$$f(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} exp(-t^2/2)dt$$

Substituting in (2), the probability of success will be given by:

$$E[Y] = P(Y = 1) = \int_{-\infty}^{\beta_1+\beta_2 X_2+\cdots+\beta_n X_n} \frac{1}{\sqrt{2\pi}} exp(-t^2/2)dt$$

Logit model has been the one chosen for current study.

c) ii) Classification trees

Decision tree learning is a statistic method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. Tree models where the target variable can take a finite set of values are called **classification trees**. This study is focused on classification trees, since the target variable is binary.

Classification trees are diagrams represented by a root on top (the initial group of instances), nodes, branches descending from nodes, and leaves. Each node of the tree specifies an attribute of the instances. Each branch starting from a node, represent all possible values for this attribute. Leaves are dead ends, nodes that do not generate new branches.

Thus, there is only one root and, departing from the root, there will be – in this specific study – two branches: the one that represent instances that give a response equal to 0 for the node attribute, and the one that represent instances that give a response equal to 1 for the node attribute. Just like the root, nodes also generate two branches, classifying instances depending on the response they give. Branches whose instances cannot be separated in new attributes end up in leaves. A model is finished once all its instances are located in leaves.

Following the variables mentioned in the previous section, attributes of classification tress are constituted by the independent variables X_1, X_2, …, X_k, and classifiers (or responses) that determine the direction of the instances in branches, are constituted by the dependent variable Y.

Appropriate problems where to use classification trees have the following characteristics:

- Instances are described by a fixed set of attributes (location, magnitude and hypocenter depth, in the case study).
- The target function has a finite set of values (binary, 0 and 1, in this study).
- Disjunctive descriptions may be required.

– The training data may contain errors. Decision trees are robust to errors, both in classification of the training examples and errors in the attribute values that describe this examples.
– The training data may contain missing attribute values.

According to these characteristics, the case study perfectly fits with a classification tree model.

d) Metaheuristic

The word Metaheuristic is composed by two Greek words:

- The word ***ευριστικειν***, meaning the art of discovering new strategies or rules to solve problems.
- The prefix ***μετα***, meaning "after", or "beyond".

Both of them together define a set of search methodologies that can be defined as upper level general methodologies used as guiding strategies in designing solutions for specific computational optimization problems.

Metaheuristics allow to tackle large-size problem instances by delivering satisfactory solutions in a reasonable time.

Application of metaheuristics falls into a large number of areas; some them are:

- Engineering design, topology optimization and structural optimization in electronics and VLSI, aerodynamics, fluid dynamics, telecommunications, automotive, and robotics.
- Machine learning and data mining in bioinformatics and computational biology, and finance.
- System modeling, simulation and identification in chemistry, physics, and biology; control, signal, and image processing.
- Planning in routing problems, robot planning, scheduling and production problems, logistics and transportation, supply chain management, environment, and so on.

d) i) Evolutionary algorithms

The metaheuristic methodology used in this study is a type of population-based metaheuristics called ***Evolutionary algorithm (EA)***, which consists of an iterative improvement in a population of solutions.

Selection processes in evolutionary algorithms are based in processes that simulate biological reproduction. The starting point is a randomly generated group of individuals, called initial population. Each of these individuals represent a potential solution to the problem. An objective function associates a fitness value with every individual, indicating its suitability to the problem, so that individuals with better fitness are selected with higher probability. An iterative optimization algorithm is applied to these individuals, that will evolve following the schemes proposed by Darwin, based in the notion of competition, and would generate a new population. This new population is integrated into the current one using some selection procedures. The search process is stopped when a given condition is satisfied (stopping criterion).

d) ii) Genetic algorithms

Genetic algorithms (GA) have been developed by J. Holland in the 1970s (University of Michigan, USA) to understand the adaptive processes of natural systems. Genetic algorithms are a very popular class of EA traditionally associated with the use of a binary representation.

A GA usually applies a crossover operator to two solutions that plays a major role, plus a mutation operator that randomly modifies the individual contents to promote diversity. Thus, the model is created as follows:

- The initial population is composed by individuals that are called 'parents'.
- Parents are combined in pairs – crossover – to produce next generation – 'children' –.
- Random changes might be produced —*mutation*— in a percentage of the children.

As a type of EA, a GA considers each generation as a population of solutions, codified as 'chromosomes'. 'Chromosomes' are groups of parameters or 'gens'. Each of these chromosomes will have associated a goodness or fitness value, that quantifies its validity as a solution to the problem. Depending on this value, the individuals will have more or less opportunities to reproduce.

Thus, a function for selecting individuals of each generation needs to be specified. From one generation to the other, an elitist strategy can be selected. In that case, the best individuals of each generation will be copied to the next one, to avoid losing them.

The genetic algorithm stops once a good solution is met. To do that, the algorithm needs to incorporate a method able to decide when this condition is reached.

e) Geographic Information System (GIS)

A Geographic Information System, commonly known by its acronym GIS, is an organized integration of hardware, software, geographical data and people, designed to capture, store, handle, analyze, model and represent information geographically referenced. Its aim is to solve problems of planning and management of many technical areas. It can also be defined as a model of a part of the reality referred to a land system of coordinates, constructed to satisfy specific information needs.

QGIS software [13] has been used as a help tool for representing the data of this study. QGIS is an open GIS system created by the Open Source Geospatial Foundation (OSGeo). It runs over Linux, Unix, Mac OSX, Windows and Android, and stands many formats and functionalities of vector data, raster data and databases. It is also a volunteer driven project that accepts contributions from users.

Its use is similar to other commonly used GIS software. The free software, as well as a complete manual and tutorials can be downloaded from the official website www.qgis.org .

f) Matlab software

Matlab ("MATrix LABoratory") [10] is a tool for numerical computation and visualization and a fourth generation programming language. The basic data element is a matrix. Matlab allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

A vast library of prebuilt toolboxes is available to implement the code for the user model, but Matlab language also provides features of traditional programming languages.

Matlab platform has been used in this study to implement all the models: the ones created using a statistics techniques and the ones created used metaheuristics.

1.3. Research goals and Objectives

The final goal of this study is to improve the genetic algorithm proposed by *Franco (2010)* [1] by taking into account some geological arguments. Nevertheless, this improved algorithm should be compared not only with the referred algorithm, but also with some other existing techniques, so that its validity is properly tested.

According to its chronological sequence, this paper has the four following four objectives:

1. Reviewing some existing techniques to minimize cat-bond trigger error.
   Techniques reviewed have been logistic regression, classification trees, and a genetic algorithm. A study of each method has been performed at a first stage.

2. Creating a model for each technique.
   Matlab software was used for every model. The construction of the logistic regression and the decision tree has been made with Matlab functions '*fitctree*' and '*glmfit*'. The evolutionary algorithm has been constructed from scratch, following the steps given in *Franco (2010)* [1].
   Every model is constructed to be used in any place, anytime and for any event, working in an automatic and with the only requirement of some specific input parameters.

3. Proposing an improvement of the genetic algorithm.
   This improvement is based on geological arguments, and the results obtained are also explained from a geological point of view.

4. Comparing results between the different models.
   Total error, accuracy, sensitivity and computational time, are some of the aspects that have been analyzed.

1.4. Document structure

This document is structured as follows:

- The present chapter (Chapter 1), explains the main concepts and ideas to understand the study that is being developed in the following chapters.
- Chapter 2 details the methodology followed, from data collection, to the execution of the models.
- Chapter 3 presents the results obtained with each of the models, compares them according to some evaluation methodologies, and gives an interpretation of the results from a geological point of view.
- Chapter 4 gathers up the conclusions of this study and makes a proposal to tackle future research lines.

Chapter 2. Methodology and model configuration

2.1. Data collection

The models presented in the following subchapters, are being tested on earthquakes occurred in Turkey from the early XX century, up to date.

Data have been collected from the *National Centers for Environmental Information (NCEI) of the United States of America* [14], which offers a wide and open database of earthworks around the World. Databases for all the countries have not only data of the physical parameters of the earthquakes, but also, data of the economic damages, valued either in a quantitative, either in a qualitative way.

The open earthquake catalog of the Turkey Republic has also been consulted, however, since it did not include economic damages, this data has not been taken into account.

Earthworks from 1912 to the most recent one (2012) registered in the NCEI have been used to construct the study database. First of all, a GIS database, using QGIS free software, has been created, to locate them all. Next, an Excel database has been elaborated, according to the following steps:

1. Events without hypocenter depth have been removed.
2. Magnitude has been selected according to the following priority order: moment magnitude, hypocenter magnitude, and magnitude notice at the nearest location.
3. Those events where damages data are void, have been checked in the GIS, to confirm that there were no significant towns close to them. One event dated on 1905 whose epicenter was located on the top of a town has been removed.
4. For those events where damages were measured in a qualitative way (1 for minor damages, 4 to mayor damages, 2 and 3 for intermediate ones), a logic estimation of quantitative damages has been made, in accordance with the pattern observed in those with quantitative damages. In order this to be done, a weight of 2 has been assigned to life losses number and the number of destroyed constructions, and a weight of 1 has

been assigned to the number of injured and the number of constructions damaged. A relation of the number or every attribute (life losses, destroyed constructions, injured and damages constructions) and the average value for the rank of the attribute at its corresponding category (1 to 4) has been calculated. Since economic damages have also been assigned one of the 4 categories, the weight average of each attribute has been calculated, and multiplied by the average economic damages of its corresponding category.

Location of all events can be observed in the following figure:

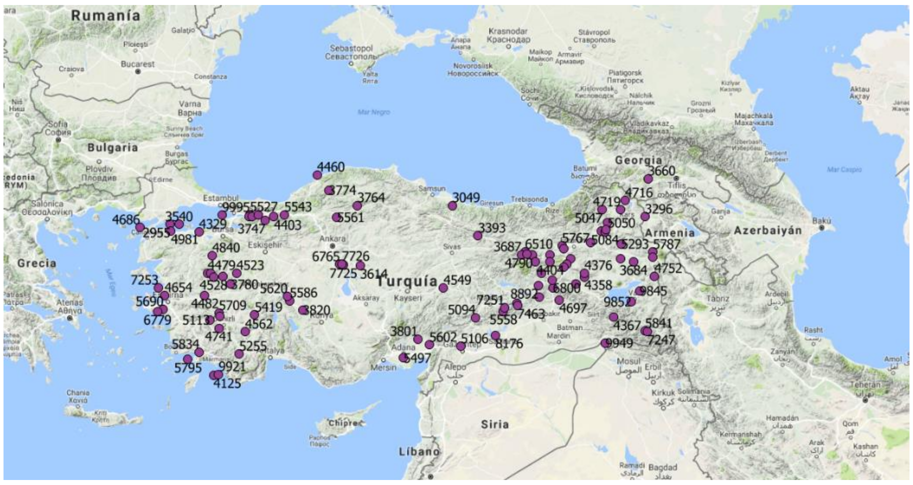


Figure 5. Location of database events over Google Physical OpenLayer. Source: QGIS (numbers attached to the event points are the NCEI event number identification)

After all these steps, the .xlsx database obtained has seven columns of attributes: event number (ID), year of occurrence ($YEAR$), longitude (X), latitude (Y), magnitude (M), depth of hypocenter (D) and economic losses (L).

A new column 'PAY' is introduced, assuming an economic threshold of 20 million dollars. Events whose economic losses are under this threshold, will have a 0 in this column, otherwise, PAY column will have a value of 1. To select this threshold, the following criteria have been considered:

- To have a ten percent of the events, approximately triggering the cat-bond, and the other 90% percent of the events not triggering the cat-bond. This criterion would be an upper bound limit in the case scenario.

- To have approximately 1 out of 3 events that causes economic losses triggering the cat-bond and approximately 2 out of 3 events that causes economic losses not triggering the cat-bond. This criterion would be an upper bound limit the case scenario.

- To choose an economical amount significant enough for insurance companies and Governments, so that cat-bonds financing could be required. This criterion would be a lower bound limit the case scenario.

Taking into account all the above explanations, 103 sample events out of the initial 161 events have been definitively selected. Coordinates, magnitude, hypocenter depth of the earthquake and economic losses, have been the attributes of each event incorporated in the study database from the *National Centers for Environmental Information (NCEI)* database. Intensity of the event has not been incorporated due to the reasons explained in section 1.1. Coordinates, magnitude and hypocenter depth will be the variables used in all models. Losses have been used to create the column 'PAY', which will be the outcome in every model.

As a result, 11 out of 103 events trigger the cat bond and 92 out of those 103 events do not trigger the cat-bond.

2.2. Data partition

Inductive techniques generally work over two (sometimes three) samples of population. Considering that the construction and the validation of the models is being performed in an inductive way, the dataset has been partitioned in two:

- Training sample: sample for constructing the different models.
- Test sample: sample for testing the model itself.

Both models should be representative to guarantee the quality of the model. To obtain representative samples, function $cvpartition(n,'HoldOut',p\ )$ of Matlab has been used, aiming to create a random partition for holdout validation on n observations. This partition divides the observations into a training set and a test (or holdout) set. The parameter 'p' must be a scalar between 0 and 1, which selects approximately '$p*n$' observations for the test set, being 'n' the total number of samples.

At first stage, it was thought to choose a parameter p = 0.4, so that the 40% of the sample (41 events) was to be used for the training data, and the 60% of the sample (62 events) was to be used for the test data. However, 62 events were though limited for the test sample. In consequence, a value of p = 0.25 has been used for the current study, and therefore, the training sample is composed by 25 events, and the test sample, by 78 events. This way, the test sample is considered to be big enough to return complete and representative results.

Thus, 8 events out of the 78 test observations would trigger the cat-bond for the specifies threshold, and 70 of them would not.

As said, Matlab software has been used for creating all four models. The way each one has been elaborated is explained in the following subchapters.

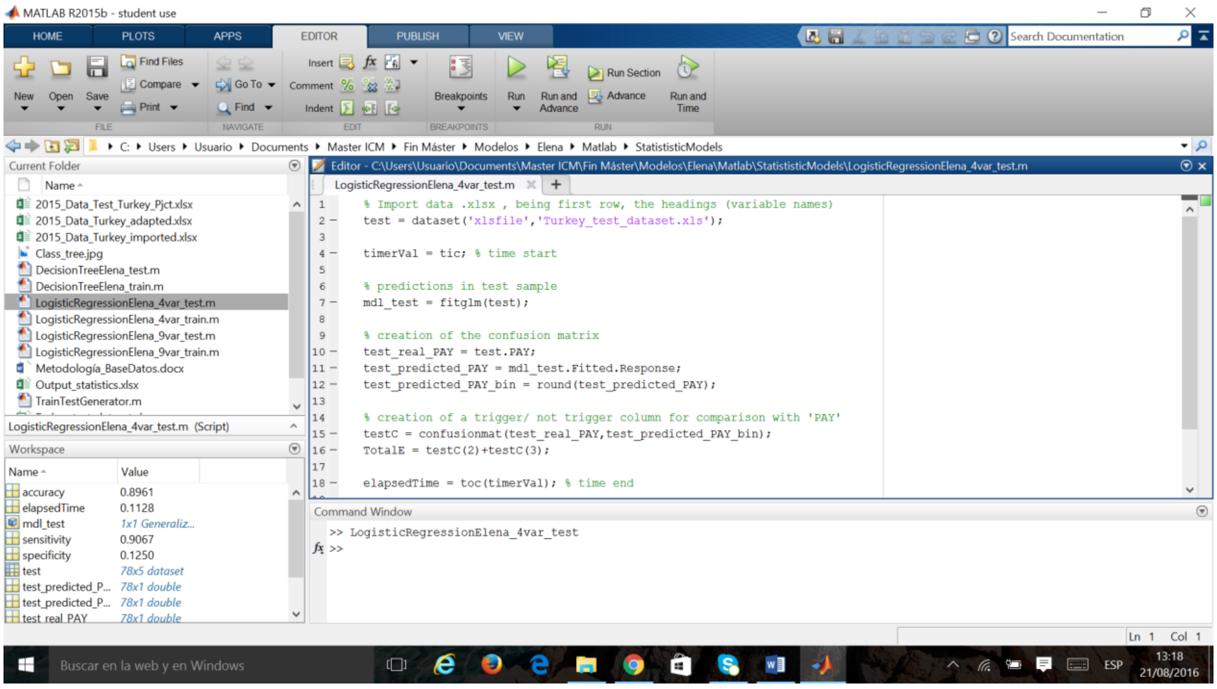
2.3. Logistic regression model

Matlab has '*fitglm*' function to create generalized linear regression model. Specifically, '*fitglm(TBL)*' returns a generalized linear model fit to variables in the table or dataset array '*TBL*', which in the case study is the training sample database.

By default, '*fitglm*' takes the last variable as the response variable, then, last column collects the attribute *PAY*. The input variables are the independent variables, or, as mentioned, the physical parameters of the earthquake events: '*X*', '*Y*', '*M*' and '*D*'. Columns *ID* and *YEAR* have not been taken into account.

To clarify this model, two outcomes have been pointed at the end of the script: predicted payments and the total error. This will be further commented in chapter 5.

There are two versions for this logistic regression model: the first one considers the four explanatory variables as presented above. The second one considers, in addition to those four variables, a pair-wise interaction of some of them '*M*D*','*M*X*','*M*Y*','*D*Y*' and '*X*Y*'.



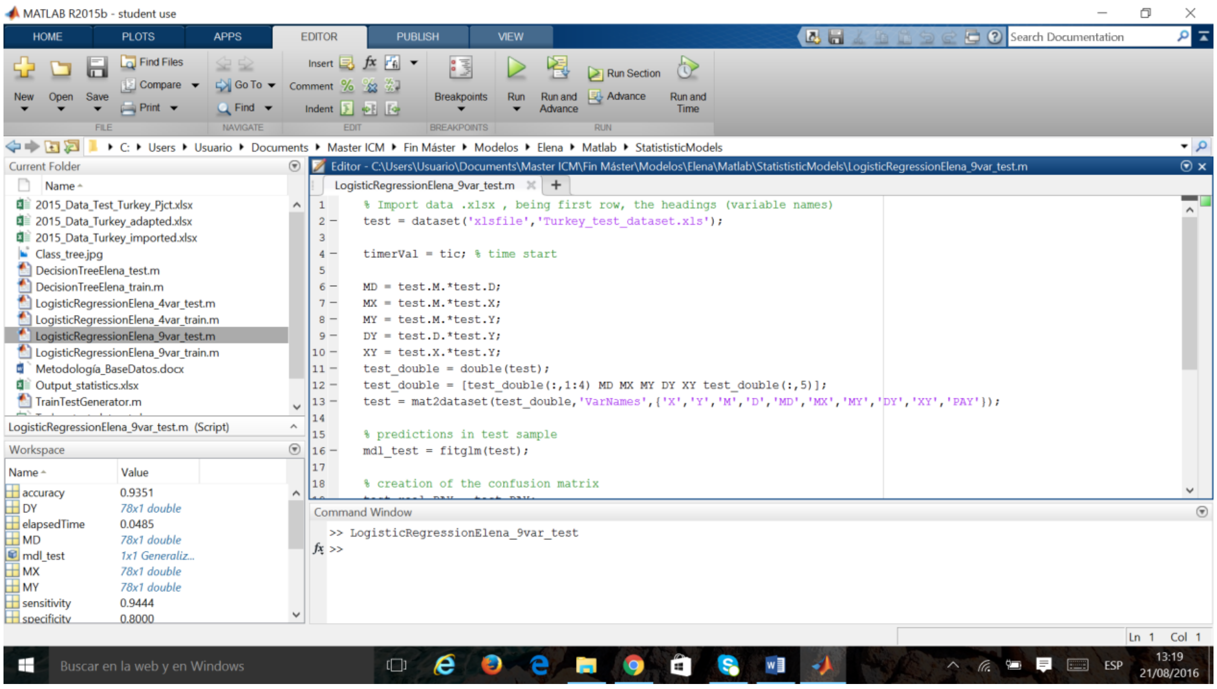
Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\StatististicModels\LogisticRegressionElena_4var_test.m

```
% Import data .xlsx , being first row, the headings (variable names)
test = dataset('xlsfile','Turkey_test_dataset.xls');

timerVal = tic; % time start

% predictions in test sample
mdl_test = fitglm(test);

% creation of the confusion matrix
test_real_PAY = test.PAY;
test_predicted_PAY = mdl_test.Fitted.Response;
test_predicted_PAY_bin = round(test_predicted_PAY);

% creation of a trigger/ not trigger column for comparison with 'PAY'
testC = confusionmat(test_real_PAY,test_predicted_PAY_bin);
TotalE = testC(2)+testC(3);

elapsedTime = toc(timerVal); % time end
```

```
>> LogisticRegressionElena_4var_test
>>
```

```
Name              Value
accuracy          0.8961
elapsedTime       0.1128
mdl_test          1x1 Generaliz...
sensitivity       0.9067
specificity       0.1250
test              78x5 dataset
test_predicted_P...  78x1 double
test_predicted_P...  78x1 double
test_real_PAY     78x1 double
```

Figure 6. Screenshot displaying the 4 variable logistic regression model



Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\StatististicModels\LogisticRegressionElena_9var_test.m

```
% Import data .xlsx , being first row, the headings (variable names)
test = dataset('xlsfile','Turkey_test_dataset.xls');

timerVal = tic; % time start

MD = test.M.*test.D;
MX = test.M.*test.X;
MY = test.M.*test.Y;
DY = test.D.*test.Y;
XY = test.X.*test.Y;
test_double = double(test);
test_double = [test_double(:,1:4) MD MX MY DY XY test_double(:,5)];
test = mat2dataset(test_double,'VarNames',{'X','Y','M','D','MD','MX','MY','DY','XY','PAY'});

% predictions in test sample
mdl_test = fitglm(test);

% creation of the confusion matrix
```

```
>> LogisticRegressionElena_9var_test
>>
```

```
Name              Value
accuracy          0.9351
DY                78x1 double
elapsedTime       0.0485
MD                78x1 double
mdl_test          1x1 Generaliz...
MX                78x1 double
MY                78x1 double
sensitivity       0.9444
specificity       0.8000
```

Figure 7 Screenshot displaying the 9 variable logistic regression model

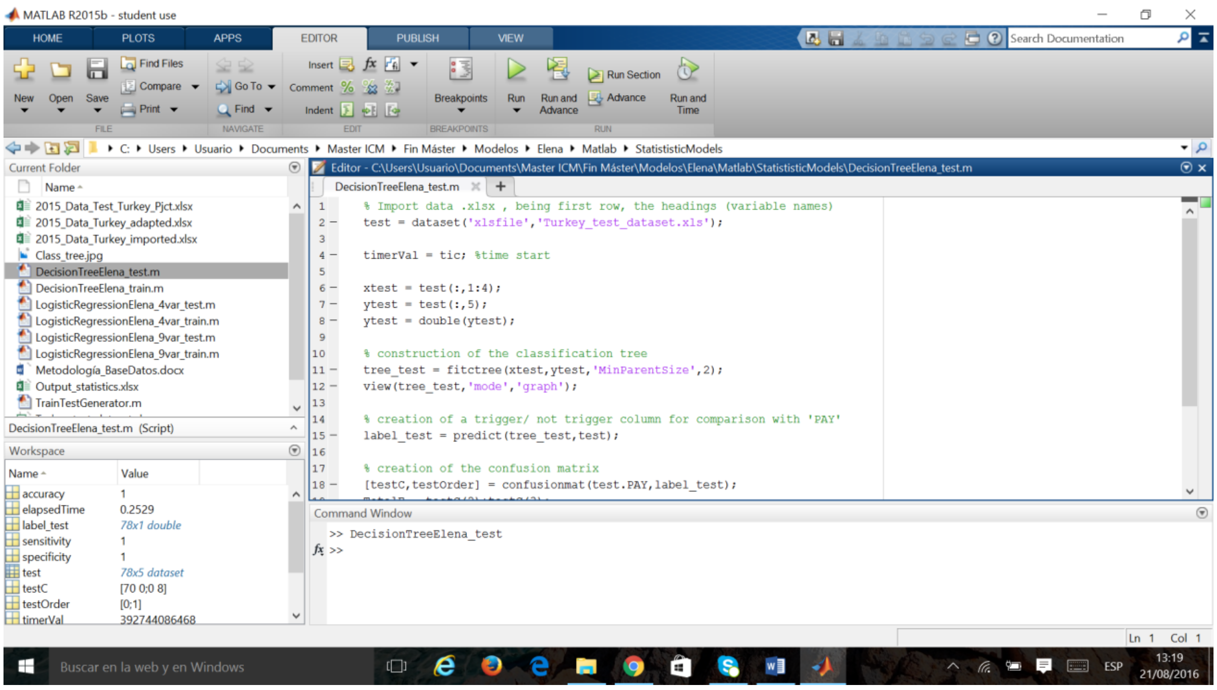
2.4. Classification tree

The classification tree model has been constructed using ‘*fitctree*’ Matlab function.

‘*fitctree(TBL,ResponseVarName)*’ returns a fitted binary decision tree based on the input variables ‘*X*’, ‘*Y*, ‘*M*’ and ‘*D*’ and the output (dependent variable ‘*PAY*’) of the training sample database. The input variables are inserted as a dataset table in place of *TBL*, and the output variable *PAY* is inserted as the ‘*ResponseVarName*’. Same as in the logistic regression model, columns *ID* and *YEAR* have not been taken into account.

The returned binary tree splits branching nodes based on the values of column *PAY*.

To clarify this model, the same outcomes pointed in the logistic regression model (predicted payments and the total error) have been remarked here. This will also be further commented in chapter 5, when comparing results.


MATLAB R2015b - student use
HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
Search Documentation
New
Open
Save
Find Files
Compare
Print
FILE
Go To
Find
NAVIGATE
Insert
Comment
Indent
EDIT
Breakpoints
BREAKPOINTS
Run
Run and Advance
Run Section
Advance
Run and Time
RUN
C: ▸ Users ▸ Usuario ▸ Documents ▸ Master ICM ▸ Fin Máster ▸ Modelos ▸ Elena ▸ Matlab ▸ StatististicModels
Current Folder
Name
Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\StatististicModels\DecisionTreeElena_test.m
DecisionTreeElena_test.m
Command Window
Buscar en la web y en Windows
Ln 1 Col 1
ESP
13:19
21/08/2016

2015_Data_Test_Turkey_Pjct.xlsx
2015_Data_Turkey_adapted.xlsx
2015_Data_Turkey_imported.xlsx
Class_tree.jpg
DecisionTreeElena_test.m
DecisionTreeElena_train.m
LogisticRegressionElena_4var_test.m
LogisticRegressionElena_4var_train.m
LogisticRegressionElena_9var_test.m
LogisticRegressionElena_9var_train.m
Metodología_BaseDatos.docx
Output_statistics.xlsx
TrainTestGenerator.m

DecisionTreeElena_test.m (Script)

Workspace

Name	Value
accuracy	1
elapsedTime	0.2529
label_test	78x1 double
sensitivity	1
specificity	1
test	78x5 dataset
testC	[70 0;0 8]
testOrder	[0;1]
timerVal	392744086468

```
% Import data .xlsx , being first row, the headings (variable names)
test = dataset('xlsfile','Turkey_test_dataset.xls');

timerVal = tic; %time start

xtest = test(:,1:4);
ytest = test(:,5);
ytest = double(ytest);

% construction of the classification tree
tree_test = fitctree(xtest,ytest,'MinParentSize',2);
view(tree_test,'mode','graph');

% creation of a trigger/ not trigger column for comparison with 'PAY'
label_test = predict(tree_test,test);

% creation of the confusion matrix
[testC,testOrder] = confusionmat(test.PAY,label_test);
```

```
>> DecisionTreeElena_test
>>
```

Figure 8. Screenshot displaying the classification tree model

2.5. Basic genetic algorithm

The 'basic genetic algorithm' is a genetic algorithm model constructed with Matlab. However, the genetic algorithm functions available in the software has not been used. The reason of not using it is to create an algorithm that fully complies with the one described in [1], pp. 993-996.

The 'basic genetic algorithm' works in a geographical area divided in a number of squared boxes, with a chosen side length. This length can be modified as desired. The smaller the side, the more number of boxes the area will be divided in.

Each box contains the earthquake events within its X and Y limits. These earthquake events will be located according to their longitude (X) and latitude (Y), and will be characterized by their magnitude (M) and hypocenter depth(D). Since there is a register of economic losses in every event, a list of the $[M, D]$ vectors that have trigger the cat bond in each box can be generated. A logical supposition would be to think that, once having selected the event with the minimum magnitude and maximum hypocenter depth, all the events whose magnitude m_i is over that minimum and whose depth d_i hypocenter is over that maximum, would trigger the cat bond. However, this does not always happen in a real situation.

Explaining this with previously mentioned parameters B_i' and B_i, ideally, a vector $[M_k, D_k]$ (being k the box number) could be selected in every box such that:

$$B_i' = \begin{cases} 0,\ if\ m_i < M_k\ or\ d_i > D_k \text{ does not trigger the cat bond} \\ 1,\ if\ m_i \geq M_k\ and\ d_i \leq D_k \text{ triggers the cat bond} \end{cases}$$

But on the other hand, variable B_i would represent whether event losses (l_i) have in fact surpass the monetary threshold (L) or not.

$$B_i = \begin{cases} 0,\ if\ l_i < L \\ 1,\ if\ l_i \geq L\ ) \end{cases}$$

Sometimes B_i' and B_i discrepancies occur, then they would be registered as trigger errors for every box.

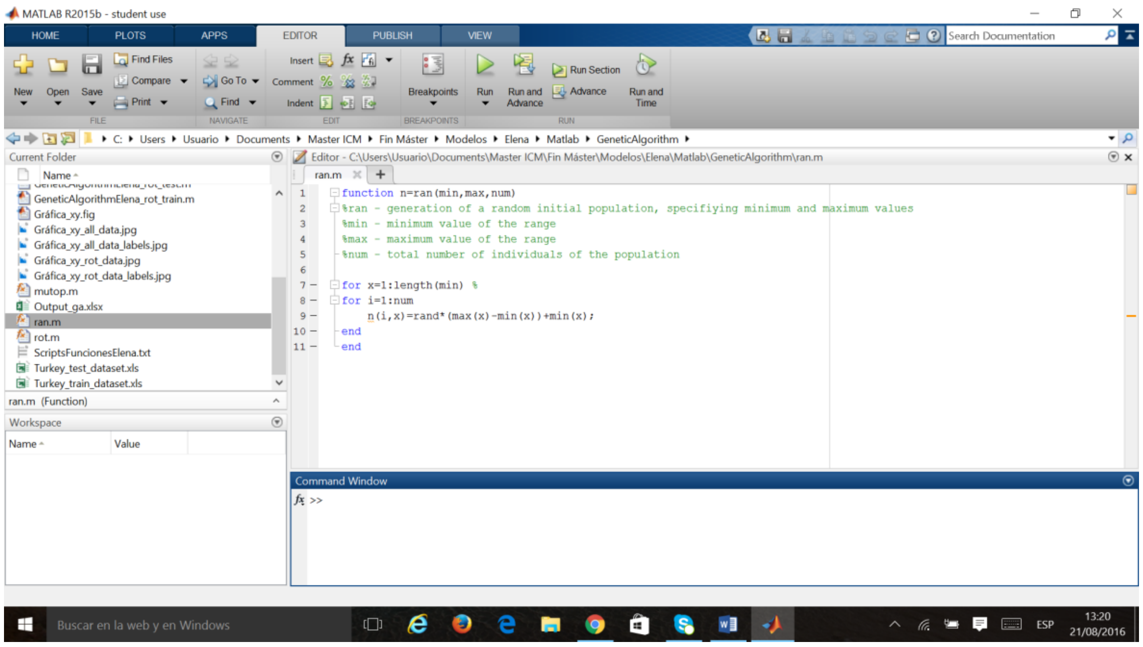
Another concept must be explained to understand this model: the concept of zones. Whereas the geographical area is divided in a number of boxes K, magnitude M and hypocenter depth D are selected to define zones. Each zone is characterized by a vector $[M_z\ D_z]$. The number of zones are less or equal than the number of boxes, so that one or more boxes could belong to one zone, but one box cannot belong to more than one zone.

The target of the 'basic genetic algorithm' is to find an arbitrary number of zones for all boxes that minimize the trigger error.

But, how exactly has the model been constructed? Well, the explanation will be divided in the same steps followed in the model script:

1. Data preparation. A function is used to import data from the Excel database.

2. Selection of the initial population: an initial population of 10 individuals is created randomly, within a specified range of values, defined by the minimum magnitude and the maximum depth existing in the database. Function '*ran.m*' has been created to get this step.

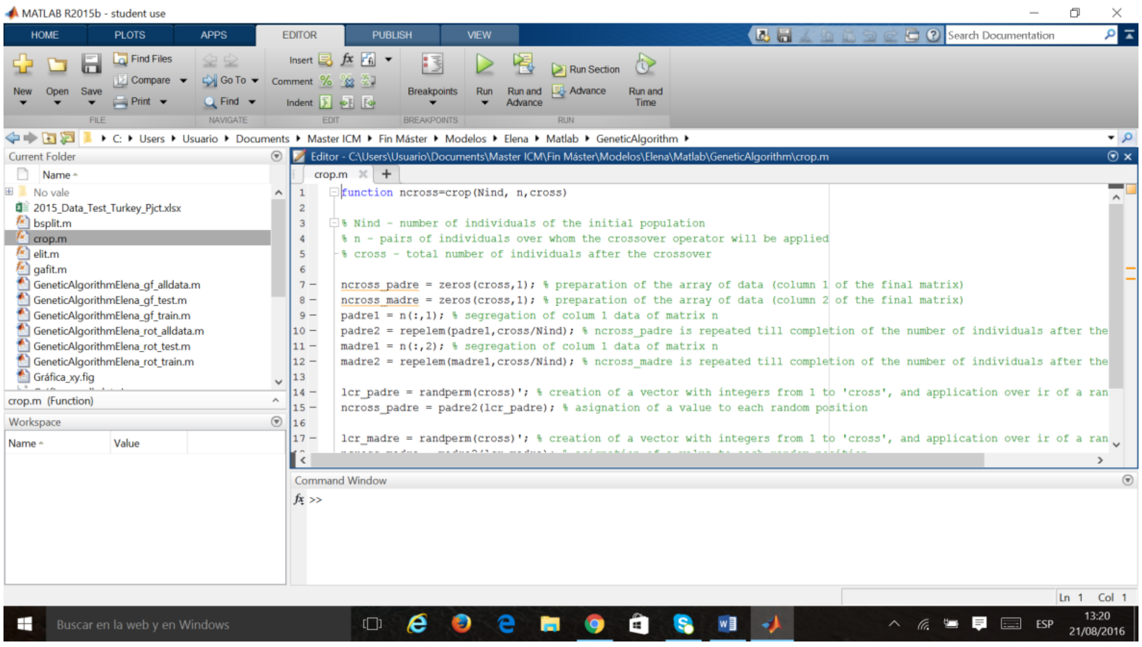


Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\GeneticAlgorithm\ran.m

```
function n=ran(min,max,num)
%ran - generation of a random initial population, specifiying minimum and maximum values
%min - minimum value of the range
%max - maximum value of the range
%num - total number of individuals of the population

for x=1:length(min) %
for i=1:num
    n(i,x)=rand*(max(x)-min(x))+min(x);
end
end
```

Figure 9. Screenshot displaying ran.m function

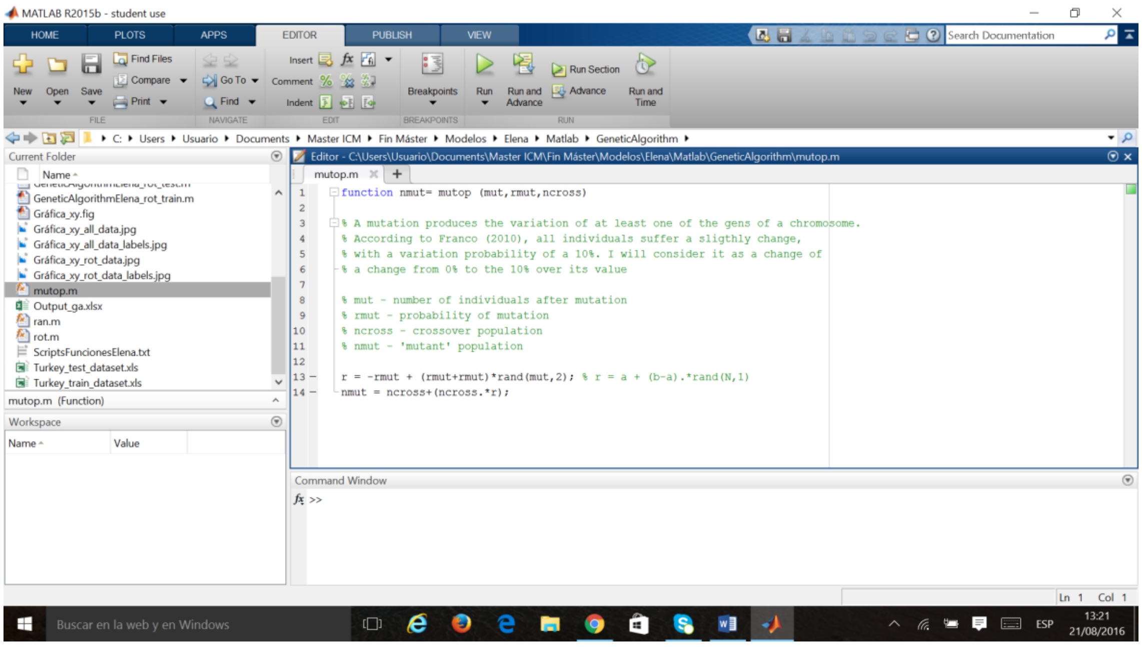
3. Application of a crossover operator over the initial population, generating 70 new individuals. To get that, initial population is copied the necessary times, and a permutation is applied to each column (M and D), so that each individual is combined randomly. Function '*crop.m*' has been created to take this step.



```
function ncross=crop(Nind, n,cross)

% Nind - number of individuals of the initial population
% n - pairs of individuals over whom the crossover operator will be applied
% cross - total number of individuals after the crossover

ncross_padre = zeros(cross,1); % preparation of the array of data (column 1 of the final matrix)
ncross_madre = zeros(cross,1); % preparation of the array of data (column 2 of the final matrix)
padre1 = n(:,1); % segregation of colum 1 data of matrix n
padre2 = repelem(padre1,cross/Nind); % ncross_padre is repeated till completion of the number of individuals after the
madre1 = n(:,2); % segregation of colum 1 data of matrix n
madre2 = repelem(madre1,cross/Nind); % ncross_madre is repeated till completion of the number of individuals after the

lcr_padre = randperm(cross)'; % creation of a vector with integers from 1 to 'cross', and application over ir of a ran
ncross_padre = padre2(lcr_padre); % asignation of a value to each random position

lcr_madre = randperm(cross)'; % creation of a vector with integers from 1 to 'cross', and application over ir of a ran
```

Figure 10. Screenshot displaying crop.m function

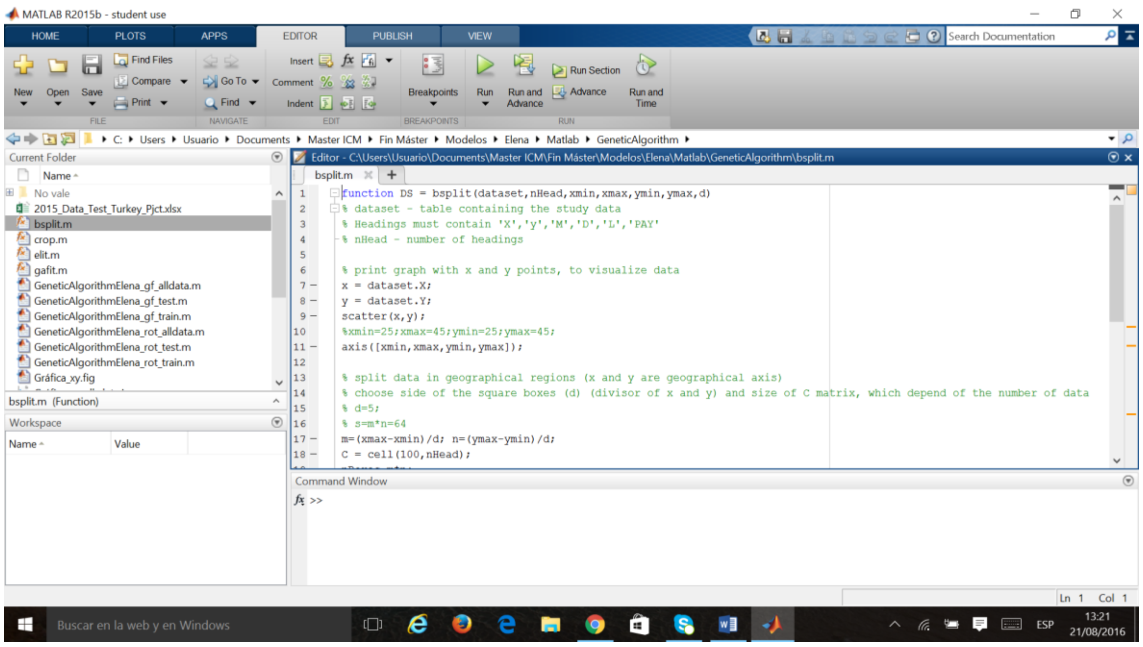
4. Application of a mutation operator over the crossed individuals. Every crossed individual suffers a mutation consisting of a random variation from a 0 to 10% of the individual values of M and D. Function '*mutop.m*' has been created to get this step.



```
function nmut= mutop (mut,rmut,ncross)

% A mutation produces the variation of at least one of the gens of a chromosome.
% According to Franco (2010), all individuals suffer a sligthly change,
% with a variation probability of a 10%. I will consider it as a change of
% a change from 0% to the 10% over its value

% mut - number of individuals after mutation
% rmut - probability of mutation
% ncross - crossover population
% nmut - 'mutant' population

r = -rmut + (rmut+rmut)*rand(mut,2); % r = a + (b-a).*rand(N,1)
nmut = ncross+(ncross.*r);
```

Figure 11. Screenshot displaying mutop.m function

5. Division of the geographical area in boxes. Function ‘*bsplit.m*’ has been created to divide the geographical area in a number of boxes that depend on the desired length of the box side.



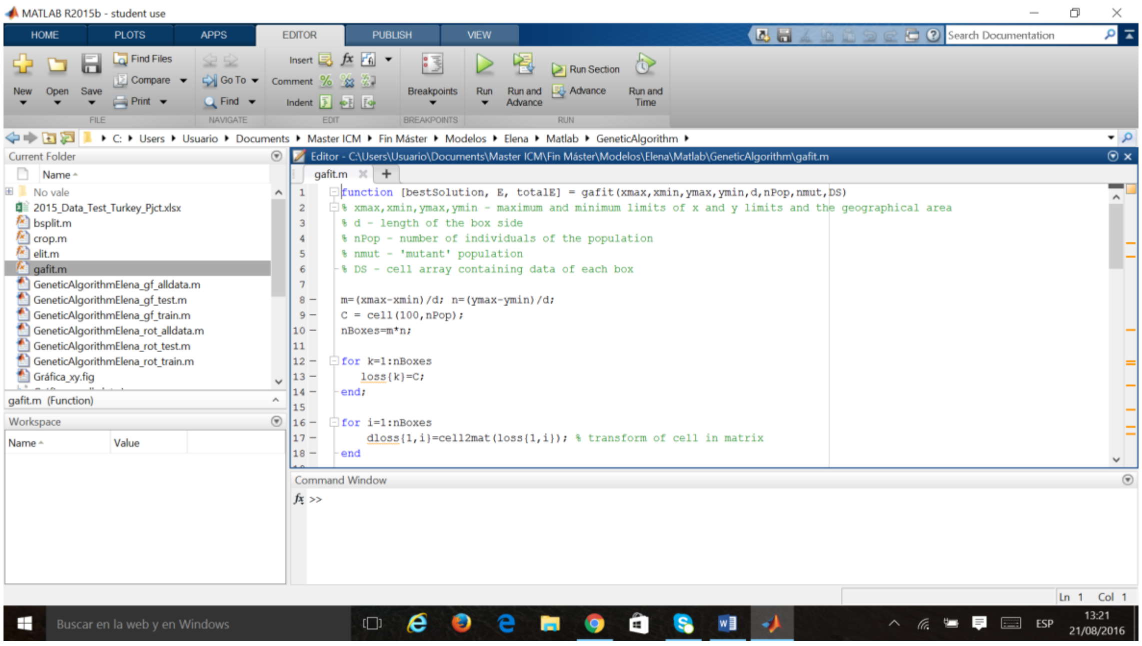
Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\GeneticAlgorithm\bsplit.m

```
function DS = bsplit(dataset,nHead,xmin,xmax,ymin,ymax,d)
% dataset - table containing the study data
% Headings must contain 'X','y','M','D','L','PAY'
% nHead - number of headings

% print graph with x and y points, to visualize data
x = dataset.X;
y = dataset.Y;
scatter(x,y);
%xmin=25;xmax=45;ymin=25;ymax=45;
axis([xmin,xmax,ymin,ymax]);

% split data in geographical regions (x and y are geographical axis)
% choose side of the square boxes (d) (divisor of x and y) and size of C matrix, which depend of the number of data
% d=5;
% s=m*n=64
m=(xmax-xmin)/d; n=(ymax-ymin)/d;
C = cell(100,nHead);
```

Figure 12. Screenshot displaying bsplit.m function

All these listed steps set up all the necessary data to apply the fitness function. Now, next step is:

1. To create fitness function, which will check the mutated population over each box, selecting the most suitable individuals. Function ‘*gafit.m*’ has been created to this aim.

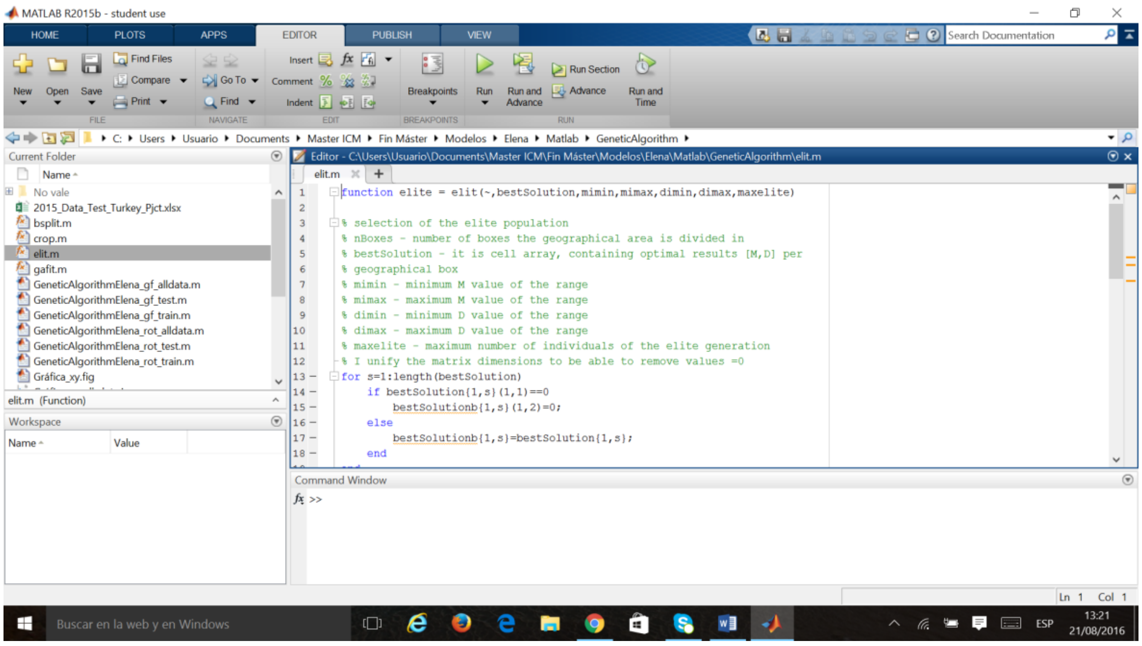


Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\GeneticAlgorithm\gafit.m

```
function [bestSolution, E, totalE] = gafit(xmax,xmin,ymax,ymin,d,nPop,nmut,DS)
% xmax,xmin,ymax,ymin - maximum and minimum limits of x and y limits and the geographical area
% d - length of the box side
% nPop - number of individuals of the population
% nmut - 'mutant' population
% DS - cell array containing data of each box

m=(xmax-xmin)/d; n=(ymax-ymin)/d;
C = cell(100,nPop);
nBoxes=m*n;

for k=1:nBoxes
   loss{k}=C;
end;

for i=1:nBoxes
    dloss{1,i}=cell2mat(loss{1,i}); % transform of cell in matrix
end
```

Figure 13. Screeshot displaying gafit.m function

2. Selection of an elite population of 10 individuals. The closest values to one individual are eliminated, till the 10 most different values amongst the whole population remain. Function '*elit.m*' has been created to this aim.

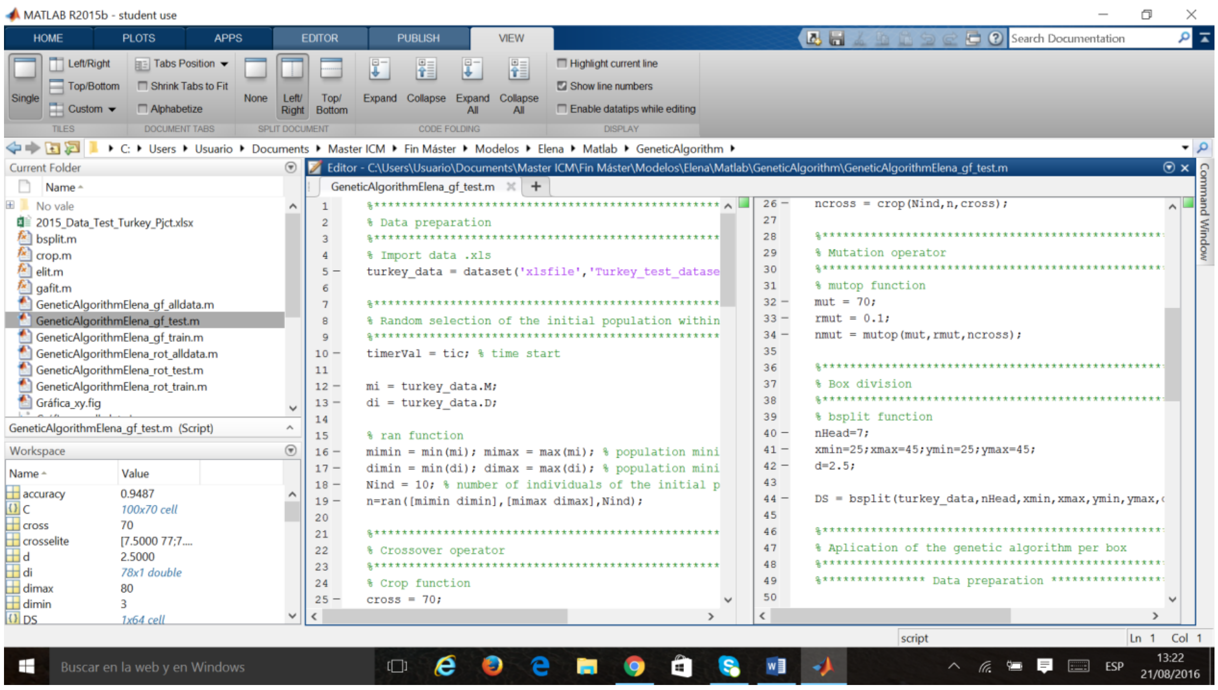


Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\GeneticAlgorithm\elit.m

```
function elite = elit(~,bestSolution,mimin,mimax,dimin,dimax,maxelite)

% selection of the elite population
% nBoxes - number of boxes the geographical area is divided in
% bestSolution - it is cell array, containing optimal results [M,D] per
% geographical box
% mimin - minimum M value of the range
% mimax - maximum M value of the range
% dimin - minimum D value of the range
% dimax - maximum D value of the range
% maxelite - maximum number of individuals of the elite generation
% I unify the matrix dimensions to be able to remove values =0
for s=1:length(bestSolution)
    if bestSolution{1,s}(1,1)==0
        bestSolutionb{1,s}(1,2)=0;
    else
        bestSolutionb{1,s}=bestSolution{1,s};
    end
```

Figure 14. Screenshot displaying elit.m function

3. Steps 6 and 7 are repeated in a loop till it converges in an isolated value or till the chosen number of iterations is reached.



```
%*****************************************************
% Data preparation
%*****************************************************
% Import data .xls
turkey_data = dataset('xlsfile','Turkey_test_datase

%*****************************************************
% Random selection of the initial population within
%*****************************************************
timerVal = tic; % time start

mi = turkey_data.M;
di = turkey_data.D;

% ran function
mimin = min(mi); mimax = max(mi); % population mini
dimin = min(di); dimax = max(di); % population mini
Nind = 10; % number of individuals of the initial p
n=ran([mimin dimin],[mimax dimax],Nind);

%*****************************************************
% Crossover operator
%*****************************************************
% Crop function
cross = 70;
ncross = crop(Nind,n,cross);

%*****************************************************
% Mutation operator
%*****************************************************
% mutop function
mut = 70;
rmut = 0.1;
nmut = mutop(mut,rmut,ncross);

%*****************************************************
% Box division
%*****************************************************
% bsplit function
nHead=7;
xmin=25;xmax=45;ymin=25;ymax=45;
d=2.5;

DS = bsplit(turkey_data,nHead,xmin,xmax,ymin,ymax,

%*****************************************************
% Aplication of the genetic algorithm per box
%*****************************************************
%*************** Data preparation ********************

```

Name	Value
accuracy	0.9487
C	100x70 cell
cross	70
crosselite	[7.5000 77;7....
d	2.5000
di	78x1 double
dimax	80
dimin	3
DS	1x64 cell

Figure 15. Screenshot displaying part of the ‘basic genetic algorithm’ as in [1] model

2.6. Improved genetic algorithm

The ‘improved genetic algorithm’ consists of an optimization of the ‘basic genetic algorithm’, by integrating geological knowledge inside the model.

As explained in 1.2.b), ***effects of an earthquake would be conditioned by the focal mechanism, the vulnerability of the area, and the arrangement of the geographical elements in the area***, which would likely follow the main fault direction.

The following section gives an example on how the earthquake effects could be conditioned by all these mentioned elements.

a) An illustrated example

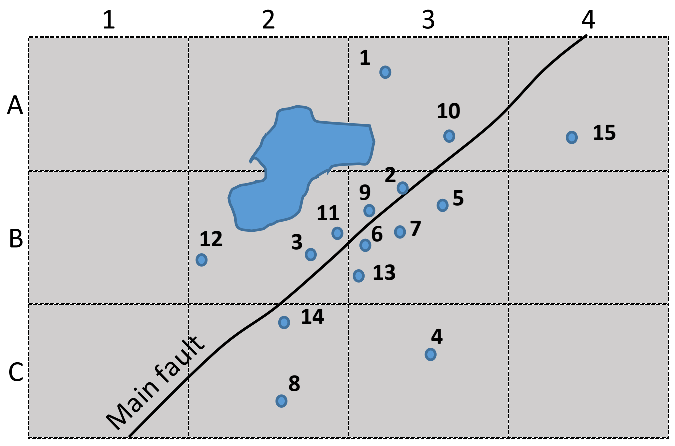
Imagine there is an invented region, with a populated area shaded, as shown in Figure 16. There is a main active fault, oblique to the 'x' axis, which causes the most important earthquakes in the area, and some other minor faults, not represented, and associated to the main one, that can also cause some earthquakes.

Fifteen events have been simulated in the region. They have been assigned a magnitude, a hypocenter depth and an amount of losses, taking into account 1) the geological criteria on earthquake effects, 2) the distance to vulnerable populated areas, so that:

- The closer an event to the populated area, the more destructive it is
- The higher the magnitude, the more destructive the earthquake is
- The shallower the depth, the more destructive the earthquake is given the same magnitude

Afterwards, a threshold of 10 billion euros has been chosen to select those events that would trigger the cat-bond (Bi=1), and those that would not (Bi=0).

This is shown in the figure, and in its adjacent table:



ID	mi	di (km)	Li 10^3M€	Bi
1	5.6	12	0.0	0
2	6.1	10	11.0	1
3	6.5	29	15.0	1
4	4.8	8	0.0	0
5	6.8	16	12.0	1
6	6.4	30	7.0	0
7	6.9	25	17.0	1
8	6.1	27	0.0	0
9	6.1	3	14.0	1
10	6.3	29	5.0	0
11	6.0	28	4.0	0
12	5.9	6	1.0	0
13	6.7	30	10.0	1
14	6.1	4	3.0	0
15	5.7	1	0.0	0

Figure 16. Imaginary region and fault, with associated events and their attribute table

Observe that the region has been divided in twelve (12) boxes, each one of which contains a number of events. If events are grouped, a labor of assigning each box a zone could be done, as shown in the following figure:

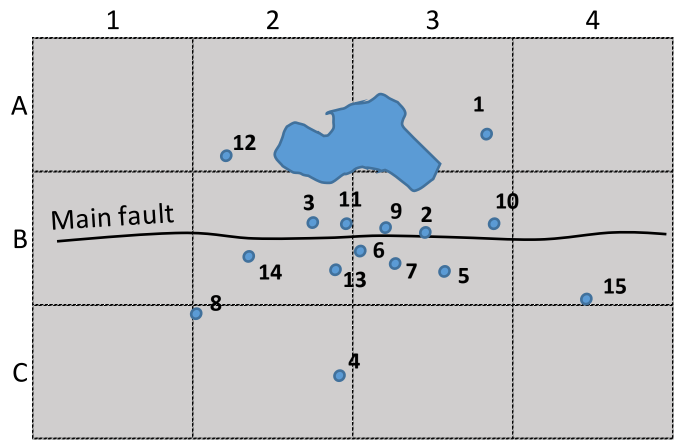
	1	2	3	4
A			[6.4 29] e=0	[6.4 29] e=0
B		[6.4 29] e=0	[6.1 25] e=1	
C		[6.4 29] e=0	[6.4 29] e=0	

ID	mi	di (km)	Li 10^3M€	Bi	Mz	Dz
2	6.1	10	11.0	1	6.1	25
5	6.8	16	12.0	1		
6	6.4	30	7.0	0		
7	6.9	25	17.0	1		
9	6.1	3	14.0	1		
13	6.7	30	10.0	1		
1	5.6	12	0.0	0	6.4	29
10	6.3	29	5.0	0		
15	5.7	1	0.0	0	5.7	1
3	6.5	29	15.0	1	6.5	29
11	6.0	28	4.0	0		
12	5.9	6	1.0	0		
8	6.1	27	0.0	0	6.2	4
14	6.1	4	3.0	0		
4	4.8	8	0.0	0	4.8	8

Figure 17. Grouping of Figure 16 events and assignation of zones to each box in the form [Magnitude Depth], e(error)

All boxes are assigned a zone that returns a zero error, but one: the box in dark yellow. A magnitude of 6.1 and a hypocenter depth of 30 would make event 13 trigger the cat bond, but would make event 6 not to trigger the cat-bond. However, if an upper magnitude is chosen, events 2 and 9 would not trigger the cat-bond, and the error would increase from 1 to 2.

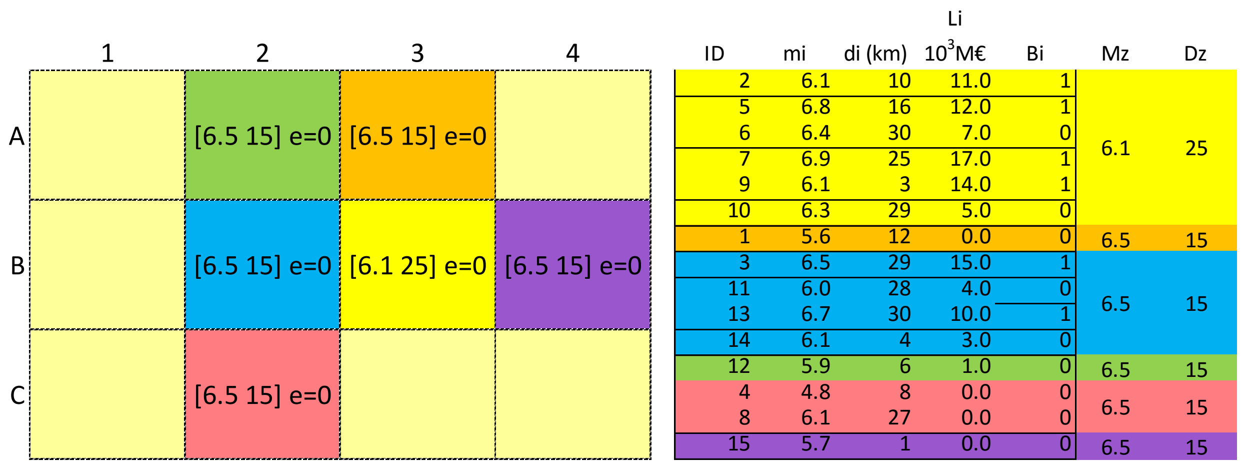
As explained in the previous subsection, earthquake effects are highly conditioned by the fault direction. Then, what would happen if the axis would be rotated till making axis 'x' parallel to the fault direction? The response can be seen in the following figure:



ID	mi	di (km)	Li 10^3M€	Bi
1	5.6	12	0.0	0
2	6.1	10	11.0	1
3	6.5	29	15.0	1
4	4.8	8	0.0	0
5	6.8	16	12.0	1
6	6.4	30	7.0	0
7	6.9	25	17.0	1
8	6.1	27	0.0	0
9	6.1	3	14.0	1
10	6.3	29	5.0	0
11	6.0	28	4.0	0
12	5.9	6	1.0	0
13	6.7	30	10.0	1
14	6.1	4	3.0	0
15	5.7	1	0.0	0

Figure 18. Rotated imaginary region and fault, with associated events and their attribute table

Events would be grouped in a different way and, as seen in Figure 19, ***the characteristics of the events and of the geographical area for each cell have been homogenized***, so that trigger error has become zero in all boxes:



	1	2	3	4
A		[6.5 15] e=0	[6.5 15] e=0	
B		[6.5 15] e=0	[6.1 25] e=0	[6.5 15] e=0
C		[6.5 15] e=0		

ID	mi	di (km)	Li 10^3M€	Bi	Mz	Dz
2	6.1	10	11.0	1	6.1	25
5	6.8	16	12.0	1		
6	6.4	30	7.0	0		
7	6.9	25	17.0	1		
9	6.1	3	14.0	1		
10	6.3	29	5.0	0		
1	5.6	12	0.0	0	6.5	15
3	6.5	29	15.0	1	6.5	15
11	6.0	28	4.0	0		
13	6.7	30	10.0	1		
14	6.1	4	3.0	0		
12	5.9	6	1.0	0	6.5	15
4	4.8	8	0.0	0	6.5	15
8	6.1	27	0.0	0		
15	5.7	1	0.0	0	6.5	15

Figure 19. Grouping of Figure 18 events and assignation of zones to each box in the form [Magnitude Depth], e(error)

This means that, since the earthquake effects are conditioned by the main fault direction, when making the fault direction parallel to the ‘x’ axis, the model would be optimized in a preliminary stage, and the behavior of the genetic algorithm model after rotating the events in the data preparation stage, would be expected to improve, minimizing the trigger error and obtaining more accurate solutions.

b) The area of study

Having a look to a Turkey tectonic map (see Figure 20), some issues can be observed:

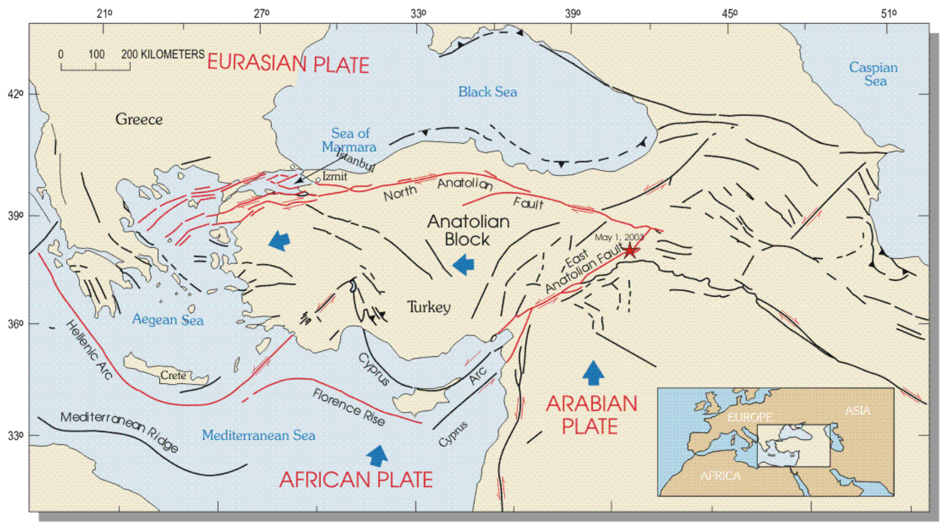


Figure 20. Main active faults in Turkey region. Source: U.S. Geological Survey website http://earthquake.usgs.gov/earthquakes/eqarchives/year/2003/2003_05_01_maps.php

Three main active faults control the seismic activity of the country:

- North Anatolian Fault (20ºN at its most active area).
- East Anatolian Fault (30ºN, approximately).
- Helenic Arc (40ºN in the Turkey section).

The main earthquakes occurred in Turkey have been located over these mentioned faults. And all these faults have a similar direction with respect to the North, from 20 to 40ºN.

A graph of the event location of the complete Turkey database would return an area of dispersed points, from 25º to 45º longitude and from 36º to 42º latitude (see Figure 21).

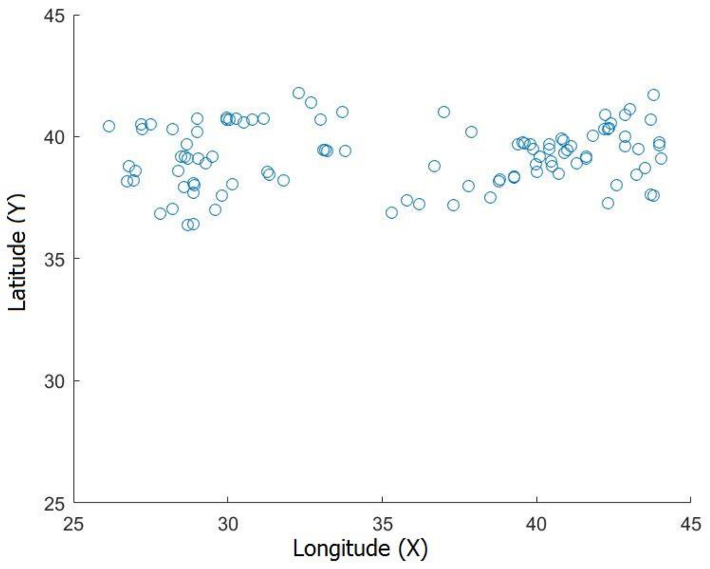


Figure 21. Representation of Turkey database events in a X (longitude) – Y (latitude) chart

What would happen if data would be rotated? In this case-study, points would be similarly dispersed (see Figure 22) if they were rotated 30ºS (30ºN is the average direction of the three main faults in Turkey), but, as explained, it is to be expected that events are more homogeneously distributed along boxes, according to their physical parameters, which are strongly dependent on their X and Y attributes, as it can be inferred from the geological exposure.

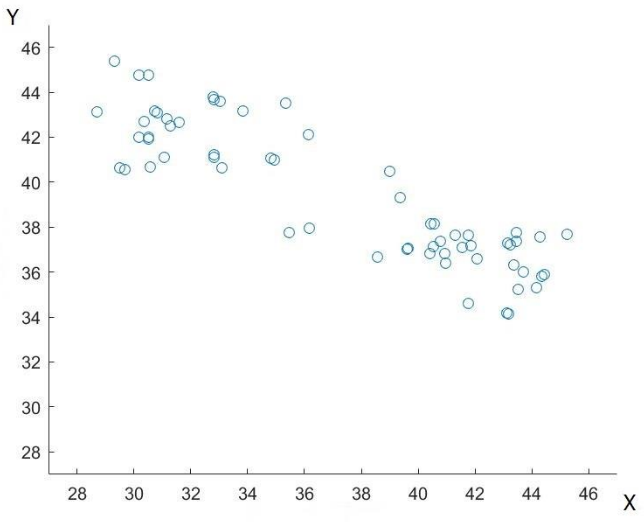


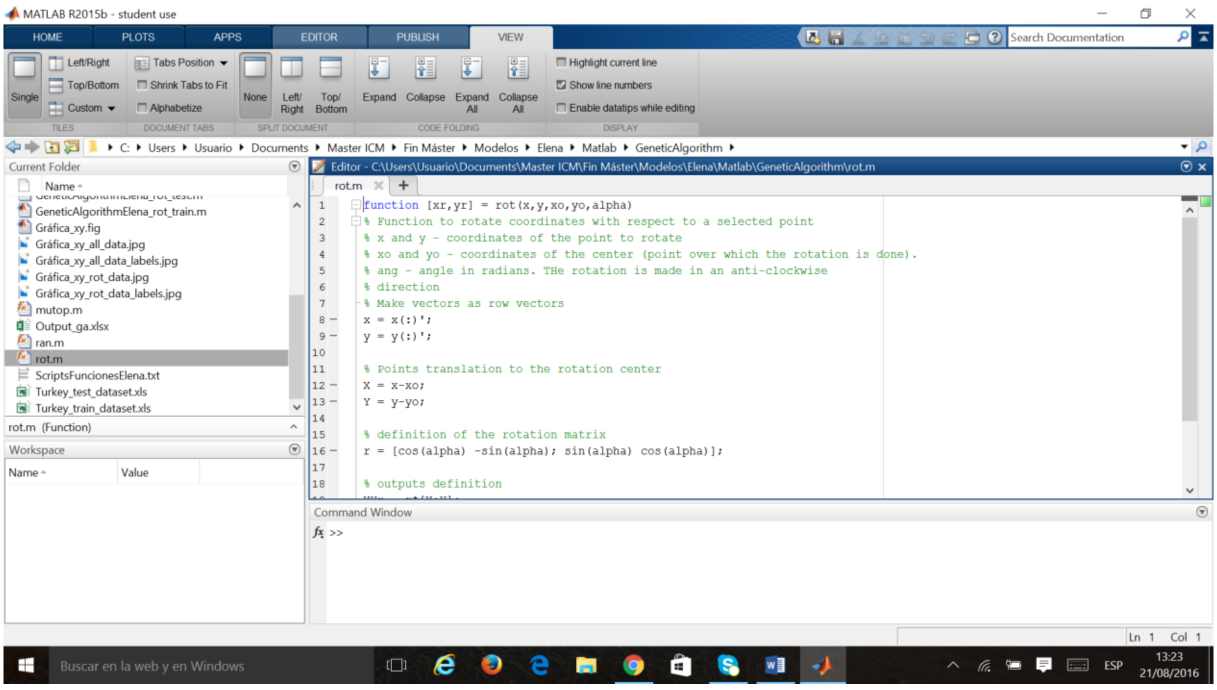
Figure 22. Representation of Turkey earthquake events in a X – Y chart, after rotating the data 30ºS

X and Y axis in Figure 22 do not exactly represent longitude and latitude as in figure Figure 21, but some new axis rotated 30ºS with respect to the longitude and latitude.

c) Construction of the 'improved genetic algorithm'

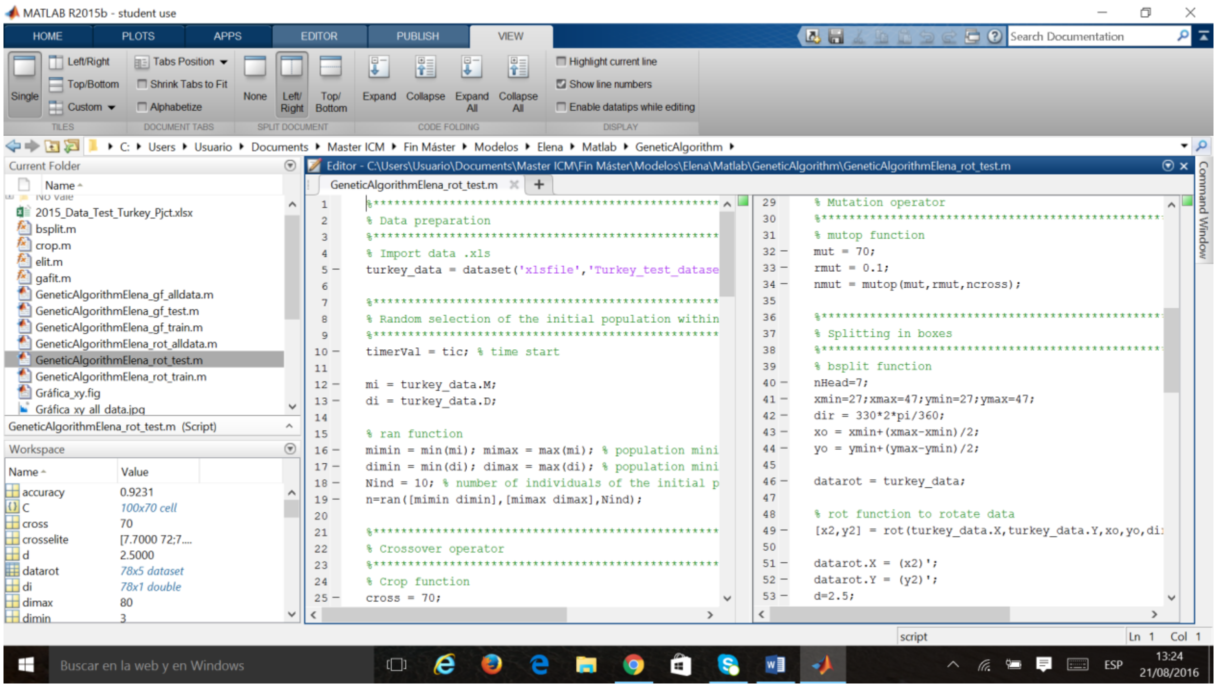
All statements and assumptions explained in 1.2.b), 2.6.a) and 2.6.b) have been taken into account for the elaboration of the 'improved genetic algorithm'.

In broad terms, 'improved genetic algorithm' has been constructed the same way than the 'basic genetic algorithm', with the peculiarity that their instances have been rotated 30º clockwise. The rotation has been implemented by creating the function '*rot.m*'. Rotation angle could be modified, so that it could be applied to any other faults anywhere.


Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\GeneticAlgorithm\rot.m

```
function [xr,yr] = rot(x,y,xo,yo,alpha)
% Function to rotate coordinates with respect to a selected point
% x and y - coordinates of the point to rotate
% xo and yo - coordinates of the center (point over which the rotation is done).
% ang - angle in radians. THe rotation is made in an anti-clockwise
% direction
% Make vectors as row vectors
x = x(:)';
y = y(:)';

% Points translation to the rotation center
X = x-xo;
Y = y-yo;

% definition of the rotation matrix
r = [cos(alpha) -sin(alpha); sin(alpha) cos(alpha)];

% outputs definition
```

Figure 23. Screenshot displaying rot.m function


MATLAB R2015b - student use
HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
Search Documentation
C: ▸ Users ▸ Usuario ▸ Documents ▸ Master ICM ▸ Fin Máster ▸ Modelos ▸ Elena ▸ Matlab ▸ GeneticAlgorithm ▸
Current Folder
Workspace
Editor - C:\Users\Usuario\Documents\Master ICM\Fin Máster\Modelos\Elena\Matlab\GeneticAlgorithm\GeneticAlgorithmElena_rot_test.m
GeneticAlgorithmElena_rot_test.m
Command Window
script
Ln 1 Col 1

Current Folder:

```
2015_Data_Test_Turkey_Pjct.xlsx
bsplit.m
crop.m
elit.m
gafit.m
GeneticAlgorithmElena_gf_alldata.m
GeneticAlgorithmElena_gf_test.m
GeneticAlgorithmElena_gf_train.m
GeneticAlgorithmElena_rot_alldata.m
GeneticAlgorithmElena_rot_test.m
GeneticAlgorithmElena_rot_train.m
Gráfica_xy.fig
Gráfica_xy_all_data.jpg
```

GeneticAlgorithmElena_rot_test.m (Script)

Workspace:

Name	Value
accuracy	0.9231
C	100x70 cell
cross	70
crosselite	[7.7000 72;7....
d	2.5000
datarot	78x5 dataset
di	78x1 double
dimax	80
dimin	3

```
%****************************************************
% Data preparation
%****************************************************
% Import data .xls
turkey_data = dataset('xlsfile','Turkey_test_datase

%****************************************************
% Random selection of the initial population within
%****************************************************
timerVal = tic; % time start

mi = turkey_data.M;
di = turkey_data.D;

% ran function
mimin = min(mi); mimax = max(mi); % population mini
dimin = min(di); dimax = max(di); % population mini
Nind = 10; % number of individuals of the initial p
n=ran([mimin dimin],[mimax dimax],Nind);

%****************************************************
% Crossover operator
%****************************************************
% Crop function
cross = 70;
```

```
% Mutation operator
%****************************************************
% mutop function
mut = 70;
rmut = 0.1;
nmut = mutop(mut,rmut,ncross);

%****************************************************
% Splitting in boxes
%****************************************************
% bsplit function
nHead=7;
xmin=27;xmax=47;ymin=27;ymax=47;
dir = 330*2*pi/360;
xo = xmin+(xmax-xmin)/2;
yo = ymin+(ymax-ymin)/2;

datarot = turkey_data;

% rot function to rotate data
[x2,y2] = rot(turkey_data.X,turkey_data.Y,xo,yo,di

datarot.X = (x2)';
datarot.Y = (y2)';
d=2.5;
```

Figure 24. Screenshot displaying part of the ‘improved genetic algorithm’ model

Chapter 3. Results and discussion

To understand how results have been calculated, a confusion matrix is presented for each model, representing B as the real values (outcomes of the trigger mechanism according to real losses), and B' as the computed values (outcomes of the trigger mechanism according to the physical parameters). This confusion matrix is expressed as:

	$B'=0$	$B'=1$
$B=0$	TP	FP
$B=1$	FN	TN

Table 1. Confusion matrix

Being TN the true positives, FP false negatives, FN false negatives, and TP true positives.

- The ***true positives*** are those where none, physical parameters (B') or losses (B), trigger the cat-bond ($B'=B=0$).
- The ***true negatives*** are those where both, physical parameters (B') and losses (B), do trigger the cat-bond ($B'=B=1$).
- The ***false positives*** are those where physical parameters (B') trigger the cat-bond but losses (B) do not trigger the cat-bond ($B'=0\neq B=1$).
- ***False negatives*** are those where physical parameters (B') do not trigger the cat-bond but losses (B) trigger the cat-bond ($B'=1\neq B=0$).

Results of each model will be given in terms of accuracy, sensitivity, specificity, total error and computational time.

Accuracy measures the positives and negatives correctly identified. It is defined as the relation between the sum of true positives and true negatives and the sum of all outputs. It is a general indicator of the behavior of the model.

$$Accuracy = \frac{True\ positives + True\ negatives}{True\ positives + True\ negatives + False\ positives + False\ negatives}$$

Sensitivity measures the proportion of positives correctly identified. It is defined as the relation between the true positive values and the sum of true positives and false negatives. A low sensitivity means that the cat-bond for some specific parameters would trigger even though the economic threshold of losses is not reached, and consequently, the model would be more favorable to sponsors than to investors. A high sensitivity would mean that the model would behave correctly for those cases where the cat-bond should not trigger.

$$Sensitivity = \frac{True\ positives}{True\ positives + False\ negatives}$$

Specificity measures the proportion of negatives correctly identified. It is defined as the relation between the true negative values and the sum of false negatives and true negatives. A low specificity would mean that the cat-bond for some specific parameters would not trigger even though the economic threshold of losses is reached, and consequently, the model would be more favorable to investors than to sponsors. A high specificity would mean that the model would behave correctly for those cases where the cat-bond should trigger.

$$Specificity = \frac{True\ negatives}{False\ positives + True\ negatives}$$

Additionally, a tic toc has been inserted in each model for the time elapse returned in seconds.

3.1. Logistic regression

Two different logistic regression models have been estimated, one considering 4 independent variables (x coordinate, y coordinate, magnitude and hypocenter depth), and another one considering 9 independent variables, as given in *de Armas, Calvet, Franco, Lopeman, Juan. (2015)* [2]. The dependent variable is the output ‘PAY’ in both cases.

For a success probability over 0.5, the event is classified in the group 1; otherwise, it is assigned to group 0.

Taking into account all this, logistic regression models has return the following results:

a) 4 variables model

The following output illustrates the linear regression and model fit for the four predictors:

Logistic regression 4 values				
	Estimate	SE	tStat	P-value
(Intercept)	-1.3128	0.9980	-1.3154	0.1925
X	-0.0005	0.0053	-0.0841	0.9332
Y	0.0065	0.0267	0.2446	0.8074
M	0.1897	0.0375	5.0550	3.153E-06
D	0.0027	0.0019	1.4500	0.1514

Table 2. 4 variables linear regression model outputs

Being,

- Intercept - independent term
- Estimate – estimate of the βi coefficients for each one of the terms
- SE - standard error of the estimated coefficient βi
- tStats - quotient between the estimate and the SE
- p value – p-values associated to the estimates. Probability of finding the observed results when the null hypothesis ($\beta i=0$) is true. A commonly used significance level (α) to reject the null hypothesis is 5%. Below it, the error in case of rejecting the null hypothesis would be low, above it, it would increase.

p-values for the X and Y estimates are over 80%, well above the significance level. This means that we could not reject the null hypothesis and, therefore, X and Y coordinates are much less significant for the model than the other variables. P-value for the D estimate is much lower than X and Y estimates, although still above the 5% significance level; therefore, the null hypothesis could also be rejected in this case.

Nevertheless, the model returns a very low p-value for the M variable estimates. According to this, the magnitude is the one that contributes the most to determine the model outputs.

Although the importance of the earthquake magnitude is essential when measuring earthquake consequences, not considering any of the other parameters could derive in major errors or difficulties in understanding a damaged scenario. And, in fact, removing all variables but M from the model derives in worse results.

Confusion matrix and final results for this model are collected in the following tables:

Confusion Matrix 4 variables

	$B'=0$	$B'=1$
$B=0$	68	1
$B=1$	7	1

Total Error 4 variables	8

Table 3. Confusion matrix and total error for the 4 variables logistic regression model

Accuracy	0.8961
Sensitivity	0.9067
Specificity	0.1250
Elapsed time	0.0526

Table 4. Results for the 4 variables logistic regression model

The accuracy of the model is acceptable, although it does not reach the 90%. With respect to the sensitivity, the value is also acceptable since most of the events neither trigger the cat-bond nor reach the losses threshold. However, False Negatives are more than desired.

Specificity is more sensitive to results, since the number of events where losses amount overpasses the losses threshold is low (8 out of 77). Results show that the model does not properly choose the events that trigger the cat-bond, moreover, it is conservative and would be favorable to investors, and prejudicial to the sponsors.

b) 9 variables model

The 9 variables logistic regression model has been created to prove if the association of some of the independent variables could change the predictions obtained on the 4 variables model. When an association of variables has an effect on the model predictions, it means that they interact among them. That is the reason why this association is called 'variable interaction'.

Should variables increase their significance by interacting among them, it may mean that magnitude should be not considered as the only variable that contributes to the outcomes.

Several trials have been made over the test model, to choose the most significant variable interactions for the lower trigger error of the model. Finally, 'X', 'Y', 'M' and 'D' variables have been considered, adding interactions: '$M*D$','$M*X$','$M*Y$','$D*Y$' and '$X*Y$'.

Table 5 illustrates the linear regression and model fit for the nine predictors:

Logistic regression 9 values				
	Estimate	SE	tStat	P-value
(Intercept)	19.4483	11.4978	1.6915	0.0954
X	-0.1918	0.1819	-1.0541	0.2956
Y	-0.5795	0.2981	-1.9438	0.0561
M	-1.9988	1.3302	-1.5026	0.1376
D	0.0793	0.0745	1.0638	0.2912
MD	0.0045	0.0023	1.9630	0.0538
MX	-0.0131	0.0064	-2.0590	0.0434
MY	0.0648	0.0339	1.9129	0.0600
DY	-0.0026	0.0020	-1.3087	0.1951
XY	0.0069	0.0047	1.4611	0.1487

Table 5. 9 variables linear regression model outputs

Differences in significance have been reduced with respect to the 4 variable model. Although all variables but MX variable are above the 5% significance level, rejecting the null hypothesis in any of them has been proved to get accuracy, sensitivity and specificity worse, as well as increase the trigger error.

Confusion matrix and final results for this model are collected in the following tables:

Confusion Matrix 9 variables

	$B'=0$	$B'=1$
$B=0$	68	1
$B=1$	4	4

Total Error 9 variables	5

Table 6. Confusion matrix and total error for the 4 variables logistic regression model

Accuracy	0.9351
Sensitivity	0.9444
Specificity	0.8000
Elapsed time	0.0494

Table 7. Results for the 4 variables logistic regression model

Values of accuracy, sensitivity and specificity are all of them acceptable, although specificity value is lower than the other two. This implies that the model would generally be favorable to investors and not so favorable to sponsors.

3.2. Classification tree

Classification tree model has returned the following chart:

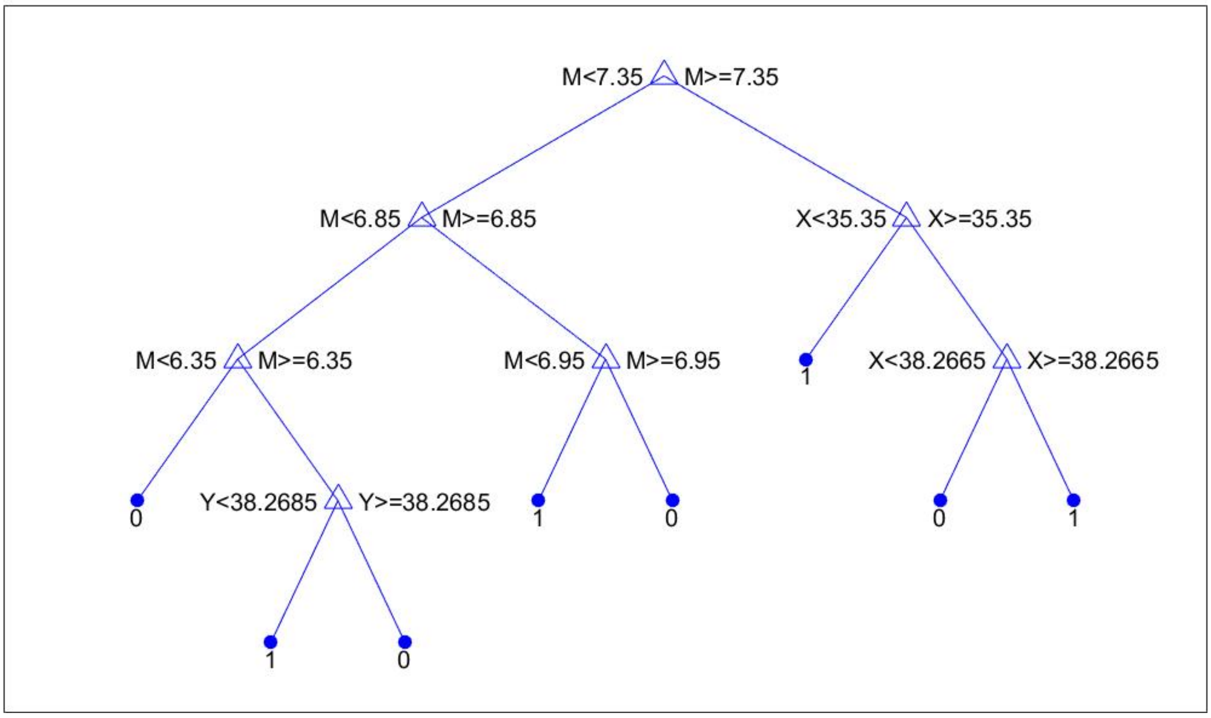


Figure 25. Classification tree results

Each node of the classification tree represents a value. Branches that depart from the node to the right, collect all samples equal or greater than the node value. Those branches departing to the left, collect all samples below the node value. Dead end branches finalize in a node indicating 0 – for those samples that trigger the cat-bond – or 1 – for those samples that do not trigger the cat-bond.

The most critical value seems to be the magnitude and, once the sample has been classified according to their magnitude, their coordinates contribute to the subsequent selections. Hypocenter depth has not been taken into account as a tree classifier in this case study, probably because it has a high variability for similar values of magnitude.

Confusion matrix and results of the classification tree model are shown in the following tables:

Confusion Matrix class. tree

	$B'=0$	$B'=1$
$B=0$	70	0
$B=1$	0	8

Total Error tree	0

Table 8. Confusion matrix and total error for the classification tree

Accuracy	1.0000
Sensitivity	1.0000
Specificity	1.0000
Elapsed time	0.1784

Table 9. Results for the classification tree

As it can be seen in the tables, the model works perfectly for all the observations, being accuracy, sensitivity and specificity equal to one. Therefore, it is expected to behave properly, being impartial for both parties, investors and sponsors.

3.3. Basic genetic algorithm

Due to the inherent characteristics of the genetic algorithm, the results obtained each time the model is run slightly change. For that reason, each model has been run 15 times, to obtain representative results in the algorithm responses. This results are: the 'Best Solution', the total error, computational time, accuracy, sensitivity and specificity. Average values of each term will be the ones taken as valid for interpretation and comparison with the other models. This will also be applicable to the 'improved genetic algorithm'.

In current and the 'improved genetic algorithm', the most important output obtained are the values of magnitude and hypocenter depth $[Mz\ Dz]$ that minimizes the trigger error, which has been called the '$Best\ Solution$'. This solution is calculated in an iterative way such that, in every iteration, the individual which behaves best in each box 'elite individuals' is selected. All these 'elite' are gathered and, once a cross and a mutation operators are applied to them, a new analysis is done again over the mutated elite population. This is run 25 times or till a unique solution is found. In the case study, a unique solution is found most of the times, this means, one zone is found for all boxes. Should the number of iteration be reduced, the number of zones would be more than one, in many cases.

Due to space limitations, confusion matrix of each ran model is not being included here or in the 'improved genetic algorithm'. The outcomes of these confusion matrixes will be represented in the model accuracy, sensitivity and specificity.

	Basic GA 4 x 4 grid (16 boxes)		
	BestSol	Total E	Comp. time
1	[7.1 69]	6	1.9307
2	[7.2 62]	7	1.8025
3	[6.2 77]	15	2.0999
4	[7.6 67]	7	1.9871
5	[6.4 60]	14	2.4313
6	[7.6 66]	7	2.086
7	[7.4 76]	5	1.9175
8	[7.2 67]	6	1.9360
9	[6.7 66]	7	2.0356
10	[6.4 58]	14	1.9521
11	[6.9 73]	8	2.2320
12	[6.9 74]	8	1.9619
13	[7.8 64]	8	2.2768
14	[7.2 51]	7	1.9207
15	6.3 79]	14	2.1188
Average performance		8.87	2.0459

	Basic GA 8 x 8 grid (64 boxes)		
	BestSol	Total E	Comp. time
1	[7.7 71]	8	2.0785
2	[7.2 51]	7	2.0782
3	[7.5 69]	5	2.0526
4	[7.2 72]	6	2.1402
5	[6.9 73]	8	2.2382
6	[7.2 65]	6	2.2804
7	[7.7 78]	8	2.1596
8	[6.7 65]	7	2.2966
9	[7.2 79]	6	2.1968
10	[7.2 72]	6	2.3329
11	[7.5 72]	5	2.0894
12	[6.9 71]	8	2.0144
13	[7.2 80]	6	2.3141
14	[7.7 74]	8	2.0843
15	[7.2 69]	6	2.1128
Average performance		6.67	2.1646

Table 10. Best solutions, error and elapsed time for GA [1]

	Basic GA 4 x 4		
	Accuracy	Sensitivity	Specificity
1	0.9231	0.9571	0.6250
2	0.9231	0.9571	0.6250
3	0.8077	1.0000	0.3478
4	0.9103	0.9200	0.6667
5	0.8205	0.9667	0.3333
6	0.9103	0.9200	0.6667
7	0.9359	0.9452	0.8000
8	0.9231	0.9571	0.625
9	0.9103	0.9846	0.5385
10	0.8205	0.9667	0.3333
11	0.8974	0.9559	0.5000
12	0.8974	0.9559	0.5000
13	0.8974	0.8974	-
14	0.9103	0.9437	0.5714
15	0.8205	1	0.3636
AV.	0.89	0.96	0.5355

	Basic GA 8 x 8		
	Accuracy	Sensitivity	Specificity
1	0.8974	0.9079	0.5000
2	0.9103	0.9437	0.5714
3	0.9359	0.9452	0.8000
4	0.9231	0.9571	0.6250
5	0.8974	0.9559	0.5000
6	0.9231	0.9571	0.6250
7	0.8974	0.9079	0.5000
8	0.9103	0.9846	0.5385
9	0.9231	0.9571	0.6250
10	0.9231	0.9571	0.6250
11	0.9359	0.9452	0.8000
12	0.8974	0.9559	0.5000
13	0.9231	0.9571	0.6250
14	0.8974	0.9079	0.5000
15	0.9231	0.9571	0.6250
AV.	0.91	0.95	0.5973

Table 11. Accuracy, sensitivity and specificity for GA [1]

These results show that the 'basic genetic algorithm' model has an acceptable accuracy and the sensitivity value shows that it properly behaves for the events that would not trigger the cat-bond. However, specificity value is low, and therefore, events that trigger the cat-bond do not match properly with those that surpass the losses threshold, favoring the investors and prejudicing the sponsors.

3.4. Improved genetic algorithm

Same steps than in the ‘basic genetic algorithm’ have been followed to run the improved GA model. These are shown in tables below:

improved GA 4 x 4

	BestSol	Total E	Comp. time
1	[7 59]	8	2.7370
2	[6.7 65]	7	2.0919
3	[7.2 69]	6	2.7148
4	[7.4 75]	5	2.8792
5	[6.9 67]	8	2.3650
6	[7.3 73]	4	2.1490
7	[7.6 61]	7	2.8602
8	[7 71]	7	2.2747
9	[6.2 67]	15	1.8283
10	[6.9 53]	9	2.1288
11	[7.6 70]	7	2.3025
12	[6.9 72]	8	2.0217
13	[7.4 61]	5	3.4399
14	[7.7 45]	9	2.8689
15	[7.7 64]	8	2.5478
Average performance		7.53	2.4806

improved GA 8 x 8

	BestSol	Total E	Comp. time
1	[7.2 70]	6	2.2306
2	[7.4 72]	5	2.2265
3	[7.6 80]	7	2.1400
4	[7 78]	7	2.2441
5	[7.1 72]	6	2.6595
6	[7.7 69]	8	2.1884
7	[7 74]	7	2.3053
8	[7.7 69]	8	2.4266
9	[7.4 60]	6	2.2564
10	[7.6 55]	8	2.1091
11	[6.7 77]	7	2.0159
12	[7.4 77]	5	2.2606
13	[7.4 69]	5	2.0870
14	[7.5 64]	5	2.3223
15	[6.7 77]	7	2.3088
Average performance		6.47	2.2521

Table 12. Best solutions, error and elapsed time for the improved GA

	improved GA 4 x 4		
	Accuracy	Sensitivity	Specificity
1	0.8974	0.9429	0.5000
2	0.9103	0.9846	0.5385
3	0.9231	0.9571	0.6250
4	0.9359	0.9452	0.8000
5	0.8974	0.9559	0.5000
6	0.9487	0.9583	0.8333
7	0.9103	0.9200	0.6667
8	0.9103	0.9565	0.5556
9	0.8077	1.0000	0.3478
10	0.8846	0.9420	0.4444
11	0.9103	0.9200	0.6667
12	0.8974	0.9559	0.5000
13	0.9359	0.9452	0.800
14	0.8846	0.8961	0
15	0.8974	0.9079	0.500
AV.	0.9000	0.9500	0.5519

	improved GA 8 x 8		
	Accuracy	Sensitivity	Specificity
1	0.9231	0.9571	0.6250
2	0.9359	0.9452	0.8000
3	0.9103	0.920	0.6667
4	0.9103	0.9565	0.5556
5	0.9231	0.9571	0.6250
6	0.8974	0.9079	0.5000
7	0.9103	0.9565	0.5556
8	0.8974	0.9079	0.5000
9	0.9231	0.9324	0.7500
10	0.8974	0.9079	0.5000
11	0.9103	0.9846	0.5385
12	0.9359	0.9452	0.8000
13	0.9359	0.9452	0.8000
14	0.9359	0.9452	0.8000
15	0.9103	0.9846	0.5385
AV.	0.9200	0.9400	0.6370

Table 13. Accuracy, sensitivity and specificity for the improved GA

As for the 'basic genetic algorithm', accuracy and sensitivity are acceptable, and therefore, the model properly behaves for the events that would not trigger the cat-bond. However, specificity value is low, then, events that trigger the cat-bond do not match properly with those that surpass the losses threshold, favoring the investors and prejudicing the sponsors.

3.5. Models overview

This summary table allows to check the results of all models at a glance:

	LR 4var	LR 9var	Clas.tree	Ba.GA 4x4	Ba.GA 8x8	Imp.GA 4x4	Imp.GA 8x8
Accuracy	0.8961	0.9351	1.0000	0.8900	0.9100	0.9000	0.9200
Sensitivity	0.9067	0.9444	1.0000	0.9600	0.9500	0.9500	0.9400
Specificity	0.1250	0.8000	1.0000	0.5355	0.5973	0.5519	0.6370
Total Error	8	5	0	9	7	8	6
Elapsed time	0.0526	0.0494	0.1784	2.0459	2.1646	2.4806	2.2521

Table 14. Summary table, comparing results between all models

According to above summary table, the model that brings best results is the classification tree.

4 variable logistic regression model is one of the less accurate models, as well as the less specific one. It would be the one that worst fits to reality. Nevertheless, when adding an interaction of some of the variables as done in the 9 variable logistic regression model, its behavior improves in a very significant way, returning acceptable values of accuracy, sensitivity and specificity.

With reference to the genetic algorithms, they all return lower results than the 9 variable logistic regression model and the classification tree. However, there are some noticeable differences between them. Stablishing a parallelism between the two 4x4 boxes genetic algorithms and the two 8x8 boxes genetic algorithms, next observations are recognized:

- The 'improved genetic algorithm' is slightly more accurate in both 4x4 and 8x8 model,
- Whereas sensitivity is a bit lower in the 'improved genetic algorithm' than in the 'basic genetic algorithm'.
- The more significant difference resides in the specificity, which .02 to .04 higher in the 'improved genetic algorithm' with respect to the other one.

Values of sensitivity are high in any case, then, the model would adequately perform in the cases where the earthquake parameters do not trigger the cat-bond. Nevertheless, the slightly lower value in the 'improved genetic algorithm' means that it would be slightly more favorable to sponsors than to investors than in the 'basic genetic algorithm'. Then, it some cases the cat-bond would trigger even when losses do not reach the stablished economic threshold.

On the contrary, specificity, is markedly higher in the 'improved genetic algorithm', although still it is low and in both cases would prejudice the sponsors. The difference in models would be more favorable to investors in the 'basic genetic algorithm', and less favorable to investors in the 'improved genetic algorithm'.

The 'improved genetic algorithm' model could be considered better fitted to reality, not only because its results are slightly better, but also because its lower value of sensitivity could be somehow understood as better compensated with its specificity value.

With respect to the computational time, logistic regression models are the fastest ones. On the opposite side, the genetic algorithms are the slower ones, due a bigger quantity of steps followed in calculations, as well as the splitting in boxes during the data preparation. The 'improved genetic algorithm' is a bit slower than the 'basic genetic algorithm' because data needs to be rotated before the splitting of the geographical area is done, which results in an additional step to be executed by the model.

Chapter 4. Conclusions

4.1. Conclusions of the study

Cat-bonds are the solution to insure natural catastrophic events such as earthquakes, since they are bonds issued by the capital market, who have capacity enough to finance the great economic losses derived from a catastrophic event of great magnitude.

Once an event has occurred, a Post Event Loss Calculation is initiated, and this can be done using many different methodologies. This study has focused in logistic regression, classification trees and genetic algorithms. Two genetic algorithm models have been studied:

- The first one, called 'basic genetic algorithm' would analyze a geographical area, dividing it in squared boxes with 'x' and 'y' axis parallel to the Earth's parallels (longitude) and the Earth's meridians (latitude).
- The second one, called 'improved genetic algorithm', would analyze a geographical area, dividing it in squared boxes, with 'x' axis parallel to the main fault direction. This would presumably gather events with similar physical characteristics and areas of similar vulnerability and therefore, make boxes have more homogeneous earthquake consequences.

The analysis has shown that the model that brings better results is the classification tree. Logistic regression model for a reduced number of variables is one of the less accurate models, as well as the less specific one, whereas a logistic regression model where interactions of variables are added to the independent variable set, returns acceptable values of accuracy, sensitivity and specificity.

Both genetic algorithms return lower results than the 9 variable logistic regression model and the classification tree. However, there are some noticeable differences between the two genetic models:

- Minimization of the trigger error is better achieved by the 'improved genetic algorithm'.
- Improved GA is more accurate than 'basic genetic algorithm'
- Sensitivity is a bit lower in the 'improved genetic algorithm' than in the 'basic genetic algorithm', although over 90% in both cases.

- The more significant difference resides in the specificity, which is .02 to .04 higher in the improved algorithm with respect to the other one, being low in both cases (from 50% to 65%).

The 'Improved genetic algorithm' model could be considered better fitted to reality, not only because its results are slightly better, but also because its lower value of sensitivity could be somehow understood as better compensated with its specificity value. This means that, although the model does not behave ideally when the mechanism should trigger the cat-bond, favoring the investors, it behaves better in the case of the improved GA as well as it is slightly more favorable to sponsors when the mechanism should not trigger the cat-bond.

Therefore, when data is rotated till making the main fault direction parallel to the boxes axis, trigger error is minimized due to a homogenization of the characteristics of the events grouped in each box. As a consequence, the model returns better results, being more accurate and specific.

4.2. Applications for the insurance activity

The global objective of all the models that have been implemented here is to predict whether a cat-bond will be triggered or not in a future event, so that, an insurance contract that includes clear scenarios for the principal disbursement, could be signed.

Although Classification Tree is performing better than the other models, there is a clear advantage of the genetic algorithms above the statistics models when applying them to insurance purposes:

- Logistic models in this study have been elaborated in an explanatory way, explaining the behavior of the trigger mechanism for the historical earthquakes. But the models could also be used to predict the trigger mechanism for future events. Nevertheless, the use of logistic regression in a predictive way is not immediate, since it requires of some calculations that are not easy to understand to those people who are not familiar with math disciplines. The use of formulas or mathematical procedures in the contract to determine whether disbursement is to be done or not, could lead to discussion or difficulties.

Discussions could arise for not being an intuitive, or understandable method for all parties. And difficulties may come once an event occurs years after the signing of the contract if any of the statistical outcomes or procedures of the contract are not clear.

- In the case of the classification tree and the genetic algorithms, they all could be incorporated in the contract before signing it, so that no calculations are needed to be done after a new event happens, being more immediate and intuitive than logistic regression.
    - Regarding the classification tree, the event would be located in any of the final nodes for some specific characteristics, determined by the upper branches. But one inconvenient is found here; should the classification tree take a coordinate as an initial node, some events could be directly classified as never triggering the cat bond.

      Imagine there is an area where no buildings or structures exists. It is not vulnerable at all and an earthquake would not produce any damage if it remains empty. But what if a new neighborhood or a new motorway is constructed there after the signature of the contract? The contract would be invalid for the new scenario, fact that could create may conflicts and controversies coming from both parties during the discussion of the contract.
    - The genetic algorithms avoid any of the controversies mentioned above. Area located within a box would be assigned a zone by the genetic algorithm model. The region, divided by boxes and their corresponding zones, could be incorporated to the contract before signing it. All events that reach or overpass the zone parameters would trigger the cat-bond and all event that do not reach the zone parameters would not trigger the cat-bond. Since there is always a zone with a pair of magnitude and hypocenter depth [Mz Dz] associated, the contract would be valid even for areas that are not built at the moment of the contract. Controversies regarding new constructions would very likely not arise in this case, then, it would generate less conflicts than the classification tree model.

4.3. Future lines of research

The number of events used for the model construction in this study is limited due to the difficulty of finding data of economic losses. Collecting data to obtain a complete database would probably require time as for a thesis study, which exceeds the purpose and time of the current study. Nevertheless, it would be ideal to check and compare the results of both genetic algorithm models over a complex database that would include a much higher number of events.

Moreover, the region of Turkey, where the study has been developed, is a very complicated area from a tectonic point of view. The whole country forms what is known as the Anatolian Tectonic plate. This plate is surrounded by the African Plate on the South, the European Plate on the north and the Arabian plate on the East. The African plate is colliding Turkey, while the other two constitute transcurrent borders. Therefore, the region of Turkey is complicated combination of compression, distension and shear stresses.

Then, in order to understand better the differences between both models, future research lines should focus in:

- A complex earthquake database, including a vast number of events.
- A region where areas of high vulnerability of buildings are clearly differenced from areas of low vulnerability of buildings.
- A region where areas of soft soils and saturated soils are clearly differenced from areas of hard soils and rocks.
- A region where the main fault is very well-defined, and stresses have a well-defined main component, preferable compressional.

A region that fulfilled all these aspects would allow to have a clearer overview of the results and to make further interpretations.

The aim of the study was to compare the improved genetic algorithm, which includes geological interpretations, with the basic genetic algorithm and other calculation techniques. Logistic regression and classification trees had been chosen for this comparison, following bibliographical reference [2] Nevertheless, some other calculation techniques could be used to further study the minimization of trigger error, such as the Nearest Neighbors Classifier, the Naïve Bayes Classifier, Linear and Quadratic Discriminant Analyses or Neural Networks.

BIBLIOGRAPHY

1. Guillermo Franco (2010). **Minimization of Trigger Error in Cat-in-a-Box Parametric Earthquake Catastrophe Bonds with an Application to Costa Rica.** *Earthquake Spectra. 26(4), 983-998*

2. Jesica de Armas, Laura Calvet, Guillermo Franco, Madeleine Lopeman, Ángel A. Juan. (2015). **Minimizing Trigger Error in Parametric Earthquake Catastrophe Bonds via Statistical Approaches.** *Modeling and Simulation in Engineering, Economics and Management. Volume 254 of the series Lecture Notes in Business Information Processing pp 167-175*

3. Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani (2013). **An Introduction to Statistical Learning**. *Springer Texts in Statistics*

4. Ángel Alejandro Juan Pérez, Renatas Kizys, Luis María Manzanedo Del Hoyo (2002). **Regresión Logística binaria**. *Proyecto e-Math*

5. Tom Mitchell (1997). **Machine Learning**. *McGraw Hill*

6. Brett Lantz (2013). **Machine Learning with R**. *Packt Publishing*

7. Marcos Gestal, Daniel Rivero, Juan Ramón Rabuñal, Julián Dorado, Alejandro Pazos (2010). **Introducción a los Algoritmos Genéticos y la Programación Genética**. *Universidade da Coruña, Servizo de Publicacións*

8. El-Ghazali Talbi (2009). **Metaheuristics. From design to implementation**. *John Wiley & Sons, Inc.*

9. Bertrand Liaudet (2008) **4 :Modélisation**. *Cours de Data mining EPF - 4ème année - IAP*

10. Peter M. Shearer (2009). **Introduction to Seismology. Second Edition.** *Cambridge University Press*

11. Alejandro Nava (2002). **Terremotos**. *Fondo De Cultura Economica USA.* http://bibliotecadigital.ilce.edu.mx/sites/ciencia/volumen1/ciencia2/34/html/terrem.html

12. Matlab R2016a documentation. www.mathworks.com

13. QGIS User Guide. Release 2.2. QGIS Project. 2004. www.qgis.com

14. Significant Earthquake Database. National Centers for Environmental Information. https://www.ngdc.noaa.gov/nndc/struts/form?t=101650&s=1&d=1ç

APPENDIX 1. MATLAB SCRIPTS

APPENDIX 1.1. Training set and a test set database split script

```
% Import data .xlsx , being first row the headings (variable
names)
turkey_table=readtable('2015_Data_Test_Turkey_Pjct.xlsx','Read
RowNames',true);

% Delete the last row of the table, should an additional row
be added
% Convert table in matrix
turkey_data=table2dataset(turkey_table(:,[2:5 7]));

% Cross validation - divide sample in two sets: test and
training
cv = cvpartition(length(turkey_data),'HoldOut',0.75);

all_data = double(turkey_data);
train = all_data(training(cv),:);
test = all_data(test(cv),:);
train =
mat2dataset(train(:,:),'VarNames',{'X','Y','M','D','PAY'});
test =
mat2dataset(test(:,:),'VarNames',{'X','Y','M','D','PAY'});

export(test,'XLSfile','Turkey_test_dataset');
export(train,'XLSfile','Turkey_train_dataset');
```

APPENDIX 1.2. Classification tree script

```
% Import data .xlsx , being first row, the headings (variable
names)
test = dataset('xlsfile','Turkey_test_dataset.xls');

timerVal = tic; %time start

xtest = test(:,1:4);
ytest = test(:,5);
ytest = double(ytest);

% construction of the classification tree
tree_test = fitctree(xtest,ytest,'MinParentSize',2);
view(tree_test,'mode','graph');

% creation of a trigger/ not trigger column for comparison
with 'PAY'
label_test = predict(tree_test,test);

% creation of the confusion matrix
[testC,testOrder] = confusionmat(test.PAY,label_test);
TotalE = testC(2)+testC(3);

elapsedTime = toc(timerVal); %time end

% results in accuracy, sensitivity and specificity
accuracy =
(testC(1)+testC(4))./(testC(1)+testC(2)+testC(3)+testC(4));
sensitivity = testC(1)./(testC(1)+testC(2));
specificity = testC(4)./(testC(3)+testC(4));
```

APPENDIX 1.3. 4 variables logistic regression model script

```
% Import data .xlsx , being first row, the headings (variable
names)
test = dataset('xlsfile','Turkey_test_dataset.xls');

timerVal = tic; % time start

% predictions in test sample
mdl_test = fitglm(test);

% creation of the confusion matrix
test_real_PAY = test.PAY;
test_predicted_PAY = mdl_test.Fitted.Response;
test_predicted_PAY_bin = round(test_predicted_PAY);

% creation of a trigger/ not trigger column for comparison
with 'PAY'
testC = confusionmat(test_real_PAY,test_predicted_PAY_bin);
TotalE = testC(2)+testC(3);

elapsedTime = toc(timerVal); % time end

% results in accuracy, sensitivity and specificity
accuracy =
(testC(1)+testC(4))./(testC(1)+testC(2)+testC(3)+testC(4));
sensitivity = testC(1)./(testC(1)+testC(2));
specificity = testC(4)./(testC(2)+testC(4));
```

APPENDIX 1.4. 9 variables logistic regression model script

```
% Import data .xlsx , being first row, the headings (variable
names)
test = dataset('xlsfile','Turkey_test_dataset.xls');

timerVal = tic; % time start

MD = test.M.*test.D;
MX = test.M.*test.X;
MY = test.M.*test.Y;
DY = test.D.*test.Y;
XY = test.X.*test.Y;
test_double = double(test);
test_double = [test_double(:,1:4) MD MX MY DY XY
test_double(:,5)];
test =
mat2dataset(test_double,'VarNames',{'X','Y','M','D','MD','MX',
'MY','DY','XY','PAY'});

% predictions in test sample
mdl_test = fitglm(test);

% creation of the confusion matrix
test_real_PAY = test.PAY;
test_predicted_PAY = mdl_test.Fitted.Response;
test_predicted_PAY_bin = round(test_predicted_PAY);

% creation of a trigger/ not trigger column for comparison
with 'PAY'
[testC,testOrder] =
confusionmat(test_real_PAY,test_predicted_PAY_bin);
TotalE = testC(2)+testC(3);

elapsedTime = toc(timerVal); % time end

% results in accuracy, sensitivity and specificity
accuracy =
(testC(1)+testC(4))./(testC(1)+testC(2)+testC(3)+testC(4));
sensitivity = testC(1)./(testC(1)+testC(2));
specificity = testC(4)./(testC(3)+testC(4));
```

APPENDIX 1.5. ‘Basic genetic algorithm’ script

```
%*************************************************************
************
% Data preparation
%*************************************************************
************
% Import data .xls
turkey_data = dataset('xlsfile','Turkey_test_dataset.xls');

%*************************************************************
************
% Random selection of the initial population within the
selected range
%*************************************************************
************
timerVal = tic; % time start

mi = turkey_data.M;
di = turkey_data.D;

% ran function
mimin = min(mi); mimax = max(mi); % population minimum and
maximum limits of M
dimin = min(di); dimax = max(di); % population minimum and
maximum limits of D
Nind = 10; % number of individuals of the initial population
n=ran([mimin dimin],[mimax dimax],Nind);

%*******************************************************
% Crossover operator
%*******************************************************
% Crop function
cross = 70;
ncross = crop(Nind,n,cross);

%*******************************************************
% Mutation operator
%*******************************************************
% mutop function
mut = 70;
rmut = 0.1;
nmut = mutop(mut,rmut,ncross);
```

```
%********************************************************
% Box division
%********************************************************
% bsplit function
nHead=7;
xmin=25;xmax=45;ymin=25;ymax=45;
d=2.5;

DS = bsplit(turkey_data,nHead,xmin,xmax,ymin,ymax,d);

%*********************************************************
% Aplication of the genetic algorithm per box
%*********************************************************
%*************** Data preparation ******************

nPop = length(nmut);

m=(xmax-xmin)/d; n=(ymax-ymin)/d;
C = cell(100,nPop);
nBoxes=m*n;

%************ fitness function ***************

% fitness function for the first population of mutant
individuals

fitness1 = gafit(xmax,xmin,ymax,ymin,d,nPop,nmut,DS);

% selection of the elite population - elit function
maxelite=10;
elite =
elit(nBoxes,fitness1,mimin,mimax,dimin,dimax,maxelite);

% function fitness applied again, till convergence or
iteration completion
rmut=0.1;
for i=1:25
    x=size(elite);
    if x(1)>1
        crosselite = crop(length(elite),elite,length(elite));
        mutelite = mutop(length(elite),rmut,crosselite);
        [fitelite, E] =
gafit(xmax,xmin,ymax,ymin,d,length(mutelite),mutelite,DS);
        elite =
elit(nBoxes,fitelite,mimin,mimax,dimin,dimax,maxelite);
```

```
            roundelite1 = round(elite(:,1),1);
            roundelite2 = round(elite(:,2));
            elite = [roundelite1,roundelite2];
            elite = sort(elite);
            row = find(diff(elite(:,1)+elite(:,2))==0);
            elite(row,1)=0;
            elite(row,2)=0;
            elite(~any(elite,2),:) = [];
        end
    end

    elapsedTime = toc(timerVal); % time end

    % creation of trigger/ not trigger colum for comparison with
    'PAY'
    loc = find(turkey_data.M>elite(1)&turkey_data.D<elite(2));
    Predicted_PAY = zeros(length(turkey_data),1);
    Predicted_PAY(loc)=1;

    % creation of the confusion matrix
    testC = confusionmat(turkey_data.PAY,Predicted_PAY);
    TotalE = testC(2)+testC(3);

    % results in accuracy, sensitivity and specificity
    accuracy =
    (testC(1)+testC(4))./(testC(1)+testC(2)+testC(3)+testC(4));
    sensitivity = testC(1)./(testC(1)+testC(2));
    specificity = testC(4)./(testC(3)+testC(4));
```

APPENDIX 1.6. ‘Improved genetic algorithm’ script

```
%**************************************************************
************
% Data preparation
%**************************************************************
************
% Import data .xls
turkey_data = dataset('xlsfile','Turkey_test_dataset.xls');

%**************************************************************
************
% Random selection of the initial population within the
selected range
%**************************************************************
************
timerVal = tic; % time start

mi = turkey_data.M;
di = turkey_data.D;

% ran function
mimin = min(mi); mimax = max(mi); % population minimum and
maximum limits of M
dimin = min(di); dimax = max(di); % population minimum and
maximum limits of D
Nind = 10; % number of individuals of the initial population
n=ran([mimin dimin],[mimax dimax],Nind);

%********************************************************
% Crossover operator
%********************************************************
% Crop function
cross = 70;
ncross = crop(Nind,n,cross);

%********************************************************
% Mutation operator
%********************************************************
% mutop function
mut = 70;
rmut = 0.1;
nmut = mutop(mut,rmut,ncross);
```

```
%*********************************************************
% Splitting in boxes
%*********************************************************
% bsplit function
nHead=7;
xmin=27;xmax=47;ymin=27;ymax=47;
dir = 330*2*pi/360;
xo = xmin+(xmax-xmin)/2;
yo = ymin+(ymax-ymin)/2;

datarot = turkey_data;

% rot function to rotate data
[x2,y2] = rot(turkey_data.X,turkey_data.Y,xo,yo,dir);

datarot.X = (x2)';
datarot.Y = (y2)';
d=2.5;

DSrot = bsplit(datarot,nHead,xmin,xmax,ymin,ymax,d);

%**********************************************************
% Aplication of the genetic algorithm per box
%**********************************************************
%*************** Data preparation *******************

nPop = length(nmut);

m=(xmax-xmin)/d; n=(ymax-ymin)/d;
C = cell(100,nPop);
nBoxes=m*n;

%************ fitness function ****************

% fitness function for the first population of mutant
individuals
fitness1 = gafit(xmax,xmin,ymax,ymin,d,nPop,nmut,DSrot);

% selection of the elite population - elit function
maxelite=10;
eliterot =
elit(nBoxes,fitness1,mimin,mimax,dimin,dimax,maxelite);

% function fitness applied again, till convergence or
iteration completion
```

```
rmut=0.1;
for i=1:25
    x=size(eliterot);
    if x(1)>1
        crosselite =
crop(length(eliterot),eliterot,length(eliterot));
        mutelite = mutop(length(eliterot),rmut,crosselite);
        [fitelite, E] =
gafit(xmax,xmin,ymax,ymin,d,length(mutelite),mutelite,DSrot);
        eliterot =
elit(nBoxes,fitelite,mimin,mimax,dimin,dimax,maxelite);
        roundelite1 = round(eliterot(:,1),1);
        roundelite2 = round(eliterot(:,2));
        eliterot = [roundelite1,roundelite2];
        eliterot = sort(eliterot);
        row = find(diff(eliterot(:,1)+eliterot(:,2))==0);
        eliterot(row,1)=0;
        eliterot(row,2)=0;
        eliterot(~any(eliterot,2),:) = [];
    end
end

elapsedTime = toc(timerVal); % time end

% creation of trigger/ not trigger colum for comparison with
'PAY'
loc =
find(turkey_data.M>eliterot(1)&turkey_data.D<eliterot(2));
Predicted_PAY = zeros(length(turkey_data),1);
Predicted_PAY(loc)=1;

% creation of the confusion matrix
testC = confusionmat(turkey_data.PAY,Predicted_PAY);
TotalE = testC(2)+testC(3);

% results in accuracy, sensitivity and specificity
accuracy =
(testC(1)+testC(4))./(testC(1)+testC(2)+testC(3)+testC(4));
sensitivity = testC(1)./(testC(1)+testC(2));
specificity = testC(4)./(testC(3)+testC(4));
```

APPENDIX 1.7. ran.m function script

```
function n=ran(min,max,num)
%ran - generation of a random initial population, specifiying
minimum and maximum values
%min - minimum value of the range
%max - maximum value of the range
%num - total number of individuals of the population

for x=1:length(min) %
for i=1:num
    n(i,x)=rand*(max(x)-min(x))+min(x);
end
end
```

APPENDIX 1.8. crop.m function script

```
function ncross=crop(Nind, n,cross)

% Nind - number of individuals of the initial population
% n - pairs of individuals over whom the crossover operator
will be applied
% cross - total number of individuals after the crossover

ncross_padre = zeros(cross,1); % preparation of the array of
data (column 1 of the final matrix)
ncross_madre = zeros(cross,1); % preparation of the array of
data (column 2 of the final matrix)
padre1 = n(:,1); % segregation of colum 1 data of matrix n
padre2 = repelem(padre1,cross/Nind); % ncross_padre is
repeated till completion of the number of individuals after
the crossover = cross
madre1 = n(:,2); % segregation of colum 1 data of matrix n
madre2 = repelem(madre1,cross/Nind); % ncross_madre is
repeated till completion of the number of individuals after
the crossover = cross

lcr_padre = randperm(cross)'; % creation of a vector with
integers from 1 to 'cross', and application over ir of a
random permutation
ncross_padre = padre2(lcr_padre); % asignation of a value to
each random position

lcr_madre = randperm(cross)'; % creation of a vector with
integers from 1 to 'cross', and application over ir of a
random permutation
ncross_madre = madre2(lcr_madre); % asignation of a value to
each random position

ncross = [ncross_padre,ncross_madre]; % merging of the two
vectors in a matrix with a number of individuals = 'cross'
```

APPENDIX 1.9. mutop.m function script

```
function nmut= mutop (mut,rmut,ncross)

% A mutation produces the variation of at least one of the
gens of a chromosome.
% According to Franco (2010), all individuals suffer a
sligthly change,
% with a variation probability of a 10%. I will consider it as
a change of
% a change from 0% to the 10% over its value

% mut - number of individuals after mutation
% rmut - probability of mutation
% ncross - crossover population
% nmut - 'mutant' population

r = -rmut + (rmut+rmut)*rand(mut,2); % r = a + (b-
a).*rand(N,1)
nmut = ncross+(ncross.*r);
```

APPENDIX 1.10. bsplit.m function script

```
function DS = bsplit(dataset,nHead,xmin,xmax,ymin,ymax,d)
% dataset - table containing the study data
% Headings must contain 'X','y','M','D','L','PAY'
% nHead - number of headings

% print graph with x and y points, to visualize data
x = dataset.X;
y = dataset.Y;
scatter(x,y);
%xmin=25;xmax=45;ymin=25;ymax=45;
axis([xmin,xmax,ymin,ymax]);

% split data in geographical regions (x and y are geographical
axis)
% choose side of the square boxes (d) (divisor of x and y) and
size of C matrix, which depend of the number of data
% d=5;
% s=m*n=64
m=(xmax-xmin)/d; n=(ymax-ymin)/d;
C = cell(100,nHead);
nBoxes=m*n;
for k=1:nBoxes
   M{k}=C;
end;
for i=1:nBoxes

DS{1,i}=mat2dataset(M{1,i},'VarNames',{'YEAR','X','Y','M','D',
'L','PAY'});
end;

% data to delimit boxes
jx=0:1:(m-1);lx=1:1:m;jy=0:1:(n-1);ly=1:1:n;
ax=jx*d+xmin; bx=lx*d+xmin; ay=jy*d+xmin; by=ly*d+xmin;

for s=1
    for p=1:m
        for q=1:n
            DS{1,s} =
dataset(dataset.X>ax(p)&dataset.X<bx(p)&dataset.Y>ay(q)&datase
t.Y<by(q),:);
            s=s+1;
        end
    end
end;
```

APPENDIX 1.11. rot.m function

```
function [xr,yr] = rot(x,y,xo,yo,alpha)
% Function to rotate coordinates with respect to a selected
point
% x and y - coordinates of the point to rotate
% xo and yo - coordinates of the center (point over which the
rotation is done).
% ang - angle in radians. THe rotation is made in an anti-
clockwise
% direction
% Make vectors as row vectors
x = x(:)';
y = y(:)';

% Points translation to the rotation center
X = x-xo;
Y = y-yo;

% definition of the rotation matrix
r = [cos(alpha) -sin(alpha); sin(alpha) cos(alpha)];

% outputs definition
XYr = r*[X;Y];
xor = XYr(1,:);
yor = XYr(2,:);

xr = xor+xo;
yr = yor+yo;
```

APPENDIX 1.12. gafit.m function script

```
function [bestSolution, E, totalE] =
gafit(xmax,xmin,ymax,ymin,d,nPop,nmut,DS)
% xmax,xmin,ymax,ymin - maximum and minimum limits of x and y
limits and the geographical area
% d - length of the box side
% nPop - number of individuals of the population
% nmut - 'mutant' population
% DS - cell array containing data of each box

m=(xmax-xmin)/d; n=(ymax-ymin)/d;
C = cell(100,nPop);
nBoxes=m*n;

for k=1:nBoxes
   loss{k}=C;
end;

for i=1:nBoxes
    dloss{1,i}=cell2mat(loss{1,i}); % transform of cell in
matrix
end

% I have to include all these loops, presizing matrix,
otherwise I cannot operate with matrixes, columns or rows
for s=1:nBoxes
    x=size(DS{1,s});
    for j=1:nPop
        for i=1:x(1)
            if DS{1,s}.M(i)>nmut(j,1)&&DS{1,s}.D(i)<nmut(j,2)
                dloss{1,s}(i,j)=1;
            else
                dloss{1,s}(i,j)=0;
            end
        end
    end
end

for s=1:nBoxes
    x=size(DS{1,s});
    for j=1:nPop
        for i=1:x(1)
```

```
            result{1,s}(i,j) = dloss{1,s}(i,j) ==
DS{1,s}.PAY(i);
        end
    end
end;

for s=1:length(result)
    x=size(DS{1,s});
    for j=1:nPop
        for i=1:x(1)
            E{1,s}(i,j) = result{1,s}(i,j)==0;
            Esum{1,s}(:,j) = sum(E{1,s}(:,j));
        end
    end
end;

for s=1:length(result)
    Emin{1,s}=min(Esum{1,s});
end;

mat_Emin=cell2mat(Emin);
totalE=sum(mat_Emin);

for s=1:length(result)
    location{1,s} = find(Esum{1,s} == Emin{1,s});
end

for s=1:length(result)
    if sum(DS{1,s}.PAY)==0
        bestSolution{1,s}=0;
    else
        bestSolution{1,s} = nmut(location{1,s},:);
    end
end
```

APPENDIX 1.13. elit.m function script

```
function elite =
elit(~,bestSolution,mimin,mimax,dimin,dimax,maxelite)

% selection of the elite population
% nBoxes - number of boxes the geographical area is divided in
% bestSolution - it is cell array, containing optimal results
[M,D] per
% geographical box
% mimin - minimum M value of the range
% mimax - maximum M value of the range
% dimin - minimum D value of the range
% dimax - maximum D value of the range
% maxelite - maximum number of individuals of the elite
generation
% I unify the matrix dimensions to be able to remove values =0
for s=1:length(bestSolution)
    if bestSolution{1,s}(1,1)==0
        bestSolutionb{1,s}(1,2)=0;
    else
        bestSolutionb{1,s}=bestSolution{1,s};
    end
end
% I concatenate matrixes of all individuals (once removed
cells without a solution (which had values =0)
eliteb=vertcat(bestSolutionb{:,:})
eliteb(~any(eliteb,2), : ) = [];

% I remove inividuals with magnitudes over the historical
ones, since a new
% mutation will be applied to results, then, results over
historical ones
% may arise again after mutation

elite = eliteb;
x=size(elite);

for i=1:x(1)
    if elite(i,1)>mimax
        elite(i,1)=0;
        elite(i,2)=0;
    end
end
```

```
for i=1:x(1)
    if elite(i,2)>dimax
        elite(i,1)=0;
        elite(i,2)=0;
    end
end
elite( ~any(elite,2), : ) = [];

% I remove the clossest values - non-usefuls - to select the
elite ones (more variety)
a=size(elite);

for i=1:a(1)
    x=size(elite);
    elitediff = zeros(x(1),3);
    elitediff(:,1) = elite(:,1)/(mimax-mimin);
    elitediff(:,2) = elite(:,2)/(dimax-dimin);
    elitediff(:,3) = elitediff(:,1) + elitediff(:,2);
    out=min(elitediff(:,3));
    row=find(elitediff(:,3)==out);
    if x(1)>maxelite
        elite(row,1)=0;
        elite(row,2)=0;
        elite( ~any(elite,2), : ) = [];
    end
end
```