|  |
| --- |
|  |
| A Journey of a Travelling Salesman in Spain and France |
| A Report of a Travelling Salesman Problem |
| **Roger G.**  Instructor: **Mr. Wees**  Course: **Gr. 10 Mathematics** |



|  |
| --- |
|  |

Table of Contents

[Introduction 3](#_Toc247728487)

[The 8 Cities 3](#_Toc247728488)

[The Hunt for Coordinates 4](#_Toc247728489)

[Finding the Distances 4](#_Toc247728490)

[Solving the Hamiltonian Path 6](#_Toc247728491)

[Conclusion 7](#_Toc247728492)

[Evaluation 7](#_Toc247728493)

# Introduction

During my Mathematics 10 class, the teacher assigned us with a travelling salesman problem (TSP). A TSP is a problem which has you define a number of points in a grid and find the shortest distance in which you can visit all the points, once. It is called a TSP because salesmen often need to travel to different locations, but only need to visit them once, thus calling for the need to find the fastest way through the salesman track. In essence, the TSP is a Hamiltonian path. Wikipedia describes the TSP as a problem combinatorial optimization, mainly studied in operations research and theoretical computer science.

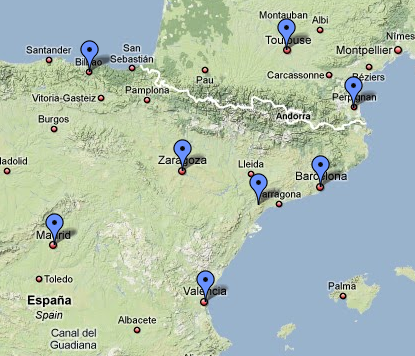
So my goal was to select 8 points in a grid (which are 8 cities on the globe), to find the distances between them and then find the shortest distance track that visits each city only once. Once the cities were selected, I found out their coordinates using a JavaScript prompt inserted into the Google™ Maps URL bar, and a message popped out with the latitude and longitude coordinates (χ and у coordinates). Then I used an algorithm to find the distance between all of those points (in pairs), and proceeded to find the track to all of the points through the shortest distance.

# The 8 Cities

I then chose 8 cities scattered through a rather small area (due to the fact that if they are scattered all over the world, the track between 2 far-apart cities might not be precise in some maps and curve due to the Earth’s spherical shape). The area I chose for my cities is the South-Eastern part of France (close to the Pyrenees) and the North-Eastern portion of Spain (also close to the Pyrenees) and the center of Spain. Inside this area, the cities that were chosen were chosen in a rather arbitrary method. I chose cities where I have visited, and in some cases where the city’s name sounded cool, and some of them where I have family – so there is no logical trend to the cities. The cities/villages are as follows:

* Barcelona – capital of Catalonia, an autonomous community in the North-Easter part of Spain.
* Valencia – capital of the autonomous community of Valencia, also in Spain.
* Madrid – capital of Spain and capital of the autonomous community of Madrid. Located in the most centric area in Spain.
* Bilbao – home of the arts museum, Guggenheim of Bilbao. Also the capital of the autonomous community of Biscay, Spain.
* Capçanes – capital of nothing, a small rural agricultural city in the heart of Catalonia’s wine-making fields. Home of a population of roughly 300, where all of them seem to be family, one way or another.
* Zaragoza – capital of the autonomous community of Aragón.
* Perpignan – city in France, have family there but know nothing about the city itself.
* Toulouse – a city in Southern France, on the banks of River Garonne.

Once the 8 cities were established, a Google™ Map was created and called “Travelling Salesman 2”, and may be found on this shortened URL: **http://is.gd/4KwlR**. Here is also a copy of the map:



# The Hunt for Coordinates

Now that the cities have been established, their coordinates were found. First, they were pinned in Google™ Maps, then one city was selected and a JavaScript source code was inserted into the URL bar and the Latitude and Longitudinal coordinates for the selected city appeared as a pop-up system message. The code was as follows:

javascript:void(prompt('',gApplication.getMap().getCenter()));

Then a table was created with all the respective coordinates. All of the coordinates have been rounded to display 2 decimal digits.

|  |  |
| --- | --- |
| Barcelona | (42.33 , 2.38) |
| Valencia | **(39.66 , -0.08)** |
| Madrid | **(40.68 , -3.11)** |
| Bilbao | **(43.27 , -2.92)** |
| Capçanes | **(41.11 , 0.78)** |
| Zaragoza | **(42.00 , -0.04)** |
| Perpignan | **(42.76 , 2.87)** |
| Toulouse | **(45.00 , 4.94)** |

# Finding the Distances

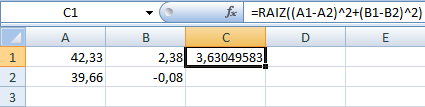
The next step that was taken to solve this TSP was to create and fill a distances table with all the distances between pairs of cities. The equation that was used is as follows:

*D* represents *Distance*

*X* and *y* represent their correspondent in a city’s coordinates (*x* – Latitude, *y* – Longitude)

At first, the equation was computed using a calculator, but then it was proven ineffective and quite time-consuming to do all the calculations of the equation shown above, *28 times!* So another method was found. The mighty power of **Microsoft Office Excel 2007!** The formula for the cell that calculated for the distance and solved that equation is as follows:

=RAIZ**(**(A1-A2)^2+(B1-B2)^2**)**

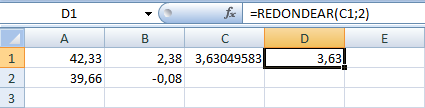


NOTE: the command “RAIZ” is equivalent to the command “SQRT”, because I have MS-Office in Spanish, and *Square Root* in Spanish is *Raíz Cuadrada.* To avoid confusion, the formula could also be written as:

=**(**(A1-A2)^2+(B1-B2)^2**)**^(1/2)

In this previous example, **C1** calculated the distance between *Barcelona* and *Valencia.* **A1** held the value of *x* in Barcelona’s Latitude coordinate, **A2** held its *y* or Longitude coordinate. Same with **A2** and **B2**, just with Valencia’s coordinates. As you may have noted, the end result was also rounded to 2 decimal places. I could have done this manually, but I felt the need for some more Excel Power! **D1** now became the playing field for rounding the **C1** value to 2 decimal places. The formula for **D1** is as follows:

=REDONDEAR(C1;2)



NOTE: again, the command “REDONDEAR” is equivalent to the command “ROUND”. In English Operating Systems, it would have been written as:

=ROUND(C1;2)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Barcelona | Valencia | Madrid | Bilbao | Capçanes | Zaragoza | Perpignan | Toulouse |
| Barcelona |  | 3,63 | 5,73 | 5,38 | 1,76 | 2,44 | 0,65 | 3,70 |
| Valencia |  |  | 3,20 | 4,59 | 1,68 | 2,34 | 4,28 | 7,33 |
| Madrid |  |  |  | 2,60 | 3,91 | 3,41 | 6,33 | 9,13 |
| Bilbao |  |  |  |  | 4,28 | 3,15 | 5,81 | 8,05 |
| Capçanes |  |  |  |  |  | 1,21 | 2,66 | 5,69 |
| Zaragoza |  |  |  |  |  |  | 3,00 | 5,81 |
| Perpignan |  |  |  |  |  |  |  | 3,05 |
| Toulouse |  |  |  |  |  |  |  |  |

Once the spreadsheet was complete, a table was created with all the results rounded to the 2nd decimal place. Here is the table:

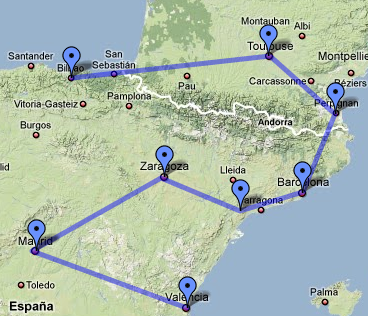
# Solving the Hamiltonian Path

Alright, we are almost done. Now we have a nice clear table of all the distances between the cities – we are ready to solve the Hamiltonian Path (or Travelling Salesman Problem, for the sake of simplicity). So now – how is the shortest track found? One method is using pure brute force search, or in other words tackle every single possible combination. But then I took a look at the actual possible solutions. The calculation leading to all the possible combinations is as follows (the factorial):

Where *n* represents the number of cities, so *n*=8:

In other words, if I were to go and take a look at all the possibly available paths, it would take me a very long time to find the shortest path amongst the *40,320* possible solutions, and a teacher would be very upset because of a student’s late work, so another method was chosen to find the shortest track.

As opposed to all the previous work done in this project, the method chosen was simple, yet looks quite effective – at least for this scenario. A path was chosen by eye, pure estimate.



This is the path that was speculated to be the shortest track visiting all 8 cities. The path starts in Bilbao and ends in Valencia, passing through in order, Toulouse, Perpignan, Barcelona, Capçanes, Zaragoza and Madrid. The total distance was then found by adding all the distances by pairs (i.e. Bilbao-Toulouse + Toulouse-Perpignan, etc.). The result is:

The result is 21.33, as shown above.

# Conclusion

The result that was achieved from using the “guessing” method is that using the *Bilbao-Toulouse-Perpignan-Barcelona-Capçanes-Zaragoza-Madrid-Valencia* combination, the total distance that our Mr. Salesman would have traveled is **21.33**.

I am sure there are plenty of other methods out there, but for this scenario (and considering that the time-frame given for this project was fairly limited), the most viable method was chosen to be the speculating method.

I am quite sure that Mr. Salesman’s boss would have been proud of him.

# Evaluation

There were some difficulties at first, when the method of finding the distance between 2 coordinates wasn’t fully understood, but that was solved the next day through a little trip to the Math teacher, and I asked away and the problem was solved.

There was one more problem, where it was entirely my fault. There was a small error made in all of the calculations at first, where I interpreted the distance algorithm as

Instead of the correct version (note that the middle brackets are being added and not multiplied, which was the initial mistake):

This proves to me once more that mere mistakes that seem trivial at first... are not that fun to deal with. It is like the airplane and pilot metaphor: the plane is flying on a 180º course, the pilot is careless and deviates to course 181º, it seems OK at first (I mean, it’s only one degree), but after hours of flying, he might be kilometres away from his initial course. With that I want to prove that small mistakes are not that small in mathematics – exactly the most exact sciences of them all.