Data Approximation

Anouk Allenspach, Rodolphe Farrando

December 15, 2017

1 Goal

The goal of this project is to find a polynomial approximation of some data. Several types of approximation can be made. The goal of the user can be to either find a trend for the data (fitting) or to interpolate the data with one of the other methods. Several possibilities are available to do an interpolation, the first one is a simple polynomial interpolation, where the degree is equal to the number of points minus 1. The second one, is a piecewise interpolation, which can can be a spline or a simple piecewise interpolation.

2 Requirements

There are three requirements to using the program:

- A gcc compiler (We used Apple machine to run the program, it uses Clang instead of gcc).
- The CMake software (the version 3.10 was used to run the tests) to link the files is necessary to compile the program
- An external library is used to solve linear systems, namely the Eigen library.

3 Method

To approximate the data, the principal method that has been used is the least squares method. The method minimizes the error between the true values and the estimated ones, that is:

$$min \sum_i (y_i - p(x_i))^2 \tag{1}$$

$p(x_i) = a_0 + a_1 \cdot x_i + \ldots + a_m \cdot x_i^m$ being the interpolated polynomial at x_i of degree m. By deriving this equation with respect to all the coefficient of the polynomial, the following linear system is obtained:

$$\begin{bmatrix} \sum_i x_i^0 & \sum_i x_i^1 & \cdots & \sum_i x_i^m \\ \sum_i x_i^1 & \sum_i x_i^2 & \cdots & \sum_i x_i^{m+1} \\ \vdots & \ddots & \ddots & \vdots \\ \sum_i x_i^m & \cdots & \sum_i x_i^{2m-1} & \sum_i x_i^{2m} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} \sum_i y_i \\ \sum_i y_i x_i \\ \vdots \\ \sum_i y_i x_i^m \end{bmatrix} \tag{2}$$

The linear system $Xa = b$ can the be solved by computing $a = X^{-1}b$.
For the spline interpolation, `https://www.math.uh.edu/ jingqiu/math4364/spline.pdf` the following paper has been used.

4 Usage of the program

4.1 Input file

The program developed allows the user to give a .csv file with some necessary inputs inside. Depending on what is in the file, the program will compute what the user has asked. The .csv files **must be** in special directory: `cmake-build-debug/code/`. They should have the following form:

data.csv	Approximation Degree	Type

The *data.csv* file contains the points coordinates in two columns, first the x coordinates, followed by the y coordinates. The type is the approximation specified by the user; it can be either *Fitting*, *Interpolation*, *Piecewise* or *PiecewiseContinuous* - the Piecewise continuous is also called spline interpolation. Finally, the approximation degree is the degree of the polynomial which approximates the data.

4.2 Interactive main

The interactive main allows the user to interact with a small program that gives the results wanted. Just compile the program and enter the name of the configuration file in the console and the program will print out the coefficient, the function(s) and the error.

4.3 Create a new main

The user just has to construct an object Points that can the be used to construct the interpolation object that he needs and finally, he can call any of the method that are in the approximation class to obtain the solution.

5 Features description

Points (class): This class creates an object that can instantiate the class Approximation. There is two ways to create an object Points, either with a .csv file ore directly with vector and number.
Approximation (class): The approximation class has three derived classes: Fitting, Interpolation and PiecewiseInterpolation. It will give the coefficients of the polynomial, display the function(s), as well as the error associated.
Function Approximation (class): A class which allows to create a configuration, as well as a point file for a function specified by the user. This can then be used to approximate the points created. To be instantiated, this class needs a .csv file that has the following information:

function name	start interval	end interval	nbr. of points to generate	method	degree

6 Tests

A series of main are testing all the possible interpolation. All this tests returns the same output: the interpolation function(s) and the error compared to the points given. For the 4 first test we generate a .csv file using the FunctionApprox class. We define a function, an interval, the number of points and the degree of interpolation. For these tests all necessary files are in the right folder.
Test 1→ Fitting Method: $f(x) = sin(x/2)$, number of points = 20 , degree = 5, interval = [0 1]
Test 2 → Interpolation Method: $f(x) = log(x)$, number of points = 6, degree = 5, interval [1 6]
Test 3 → Piece-wise Method: $f(x) = cos(x) * sin(x/2)$, number of points = 16, degree = 5, interval = [0 1]
Test 4 → Piece-wise ContinuousMethod: $f(x) = e^{(x/2)} - x^3$number of points = 20, degree = 3, interval = [3 10]
Test 5 → Function approximation class: Small test which shows how this class works
Test 6 → Empty config: Small test which throws an error because of an empty config.csv file

7 Remarks

When the degree of interpolation is high ($\geq$ 7 approximatively), the polynomial is badly conditioned and the points are not exactly interpolated. Indeed, the condition number of the matrix X in the linear system $Xa = b$ is big. It results in difficulties when inverting the matrix and the solution is thus flawed. The user needs to interpolate only few points to have a good result. The same happens for the fitting of the data, the degree shouldn't be too high otherwise same problems could occur.

8 ToDos

There are many ways this program can be extended. For example, a more robust method should be found for approximating higher degrees. Also, it might be interesting to add some simple scans in order to detect points that follow a linear function, or another well know trend. This could be done by pre testing if the points belong to one of these types of functions. In order for the user to have a better understanding of what the solution to his/her data approximation problem looks like, it could be useful to implement a method that plots the points, as well as the function that approximates them, this would add to the user friendliness of the program.