DigiPen Institute of Technology, Bilbao

MAT300 Curves & Surfaces
Spring, 2020
Project 1. Deadline: 2/10/2020

---

1 Main description of the project

The project consists of coding routines to obtain polynomial curves in 2D and 3D.

Given $n+1$ points $(t_0, P_0),\ (t_1, P_1),\ \ldots,\ (t_n, P_n)$ with $t_i \neq t_j$ for $i \neq j$, there is a unique polynomial of degree at most n passing through the points. The idea is to select $n+1$ points in $\mathbb{R}^2$ or $\mathbb{R}^3$. Next select a mesh of nodes over a closed interval $[a, b] \in \mathbb{R}$. Finally apply interpolation to obtain the polynomial curve. The different interpolation methods here are: Gauss-Jordan, Lagrange and Newton.

1.1 Input data

The input data will be the following:

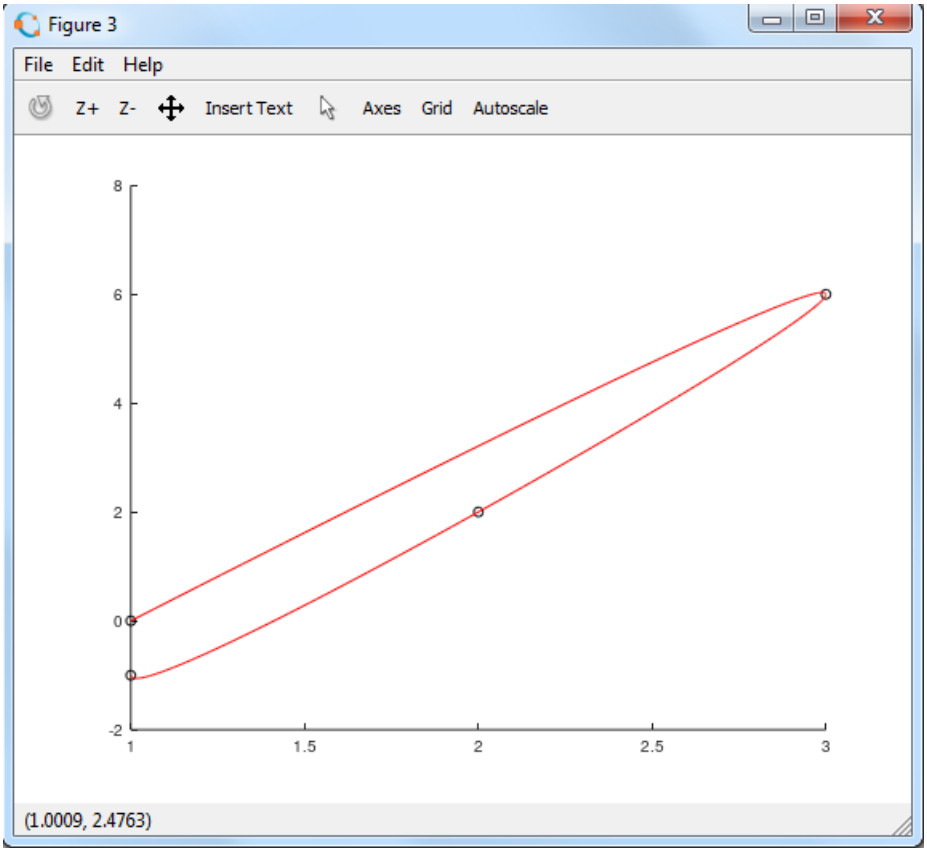
- Interpolation points P_0, P_1, ..., P_n.
- Dimension (2 or 3).
- Desired mesh for interpolation.
- Desired method for interpolation.
- Number of nodes for the output mesh.

The data will be introduced in a script file like this:

```
%-------------------------------------------------------------
% MAT300 Curves and surfaces
% Julia Sánchez Sanz
% julia.sanchez@digipen.edu
% 15/5/2019
%
% Script input data project 1
%-------------------------------------------------------------

PX=[1 2 3 1];  % x-coordinate interpolation points
PY=[-1 2 6 0]; % y-coordinate interpolation points
%PZ=[2 1 1 4]; % z-coordinate interpolation points (un-comment)

Dimension=2; % dimension 2 or 3 (consistent with PZ)

meshdigit=0; % 0 regular [0,n], 1 regular [0,1], 2 Chebyshev [-1,1]

methoddigit=0; % 0 Gauss-Jordan, 1 Lagrange, 2 Newton

outputnodes=150; % number of nodes for the output mesh
```

1.2 Expected output

When the user runs the program, a figure should appear with the graph of the polynomial curve and the interpolation points. The plot will be a 2D or 3D plot depending on the input data.



1.3 Main function

The main function of the project will be **interpolation.m**

The function will check which interpolation method is the one to use, and will call to functions **gaussjordanmethod.m**, **lagrangemethod.m** or **newtonmethod.m** for solving the interpolation problem.

1.4 The meshes of nodes

There will be three possibilities for the meshes of nodes:

- Regular mesh in $[0, n]$ satisfying $0 = t_0 < t_1 < \ldots < t_n = n$ and same distance among nodes.
- Regular mesh in $[0, 1]$ satisfying $0 = t_0 < t_1 < \ldots < t_n = 1$ and same distance among nodes.

- Chebysheb extremal mesh (ordered forward) in $[-1, 1]$ where

$$t_j = -\cos\left(\frac{j\pi}{n}\right), \qquad j = 0, 1, \ldots, n. \tag{1}$$

The computation of the mesh can be implemented in a function **meshselection.m** or it can be implemented in each of the numerical methods (Gauss-Jordan, Lagrange, Newton).

1.5 Interpolation techniques

1.5.1 Gauss-Jordan gaussjordanmethod.m

The interpolant polynomial as a linear combination of the standard basis for P_n is

$$p(x) = a_0 + a_1 t + a_2 t^2 + \ldots + a_n t^n \tag{2}$$

For each interpolation point (t_i, P_i) you have to create a linear equation, and so obtain a linear system of $n + 1$ equations of the type

$$\begin{cases} a_0 + a_1 t_0 + a_2 t_0^2 + \ldots + a_n t_0^n = P_0, \\ a_0 + a_1 t_1 + a_2 t_1^2 + \ldots + a_n t_1^n = P_1, \\ \vdots \\ a_0 + a_1 t_n + a_2 t_n^2 + \ldots + a_n t_n^n = P_n. \end{cases} \tag{3}$$

for each coordinate (x, y and z). By solving system (3) applying the Gauss-Jordan algorithm, you will obtain the coefficients a_i for $i = 0, 1, \ldots, n$ of the interpolant polynomial in the standard basis of P_n (for each coordinate).

Finally, you have to construct and evaluate the polynomial at the points of the output mesh to obtain the graph of the curve.

1.5.2 Lagrange lagrangemethod.m

Here you will use the Lagrange basis for P_n which is the set of polynomials

$$L_j^n(t) = \prod_{i \neq j} \frac{t - t_i}{t_j - t_i}, \qquad i, j = 0, 1, \ldots, n, \tag{4}$$

where the t_i and t_j for $i, j = 0, 1, \ldots, n$ are the nodes of the mesh.

For each point t of the output mesh you have to evaluate the Lagrange polynomials and construct the interpolant polynomial curve through the linear combination

$$p(t) = \sum_{j=0}^{n} P_j L_j^n(t). \tag{5}$$

You have to do this for each coordinate (x, y and z), and plot the graph of the curve.

1.5.3 Newton newtonmethod.m

You have to use the Newton basis for P_n and divided differences.

For the nodes in the mesh $t_0,\ t_1,\ \ldots,\ t_n$ the Newton basis is the one formed by the $n+1$ polynomials $N_0(t) = 1$ and

$$N_j(t) = \prod_{i=0}^{j-1}(t - t_i), \qquad j = 1, \ldots, n. \tag{6}$$

The divided difference $f[t_0, t_1, \ldots, t_n]$ is computed in the following recursive way:

$$\begin{array}{lll}
f[t_0] = P_0 & & \\
 & f[t_0,t_1] = \frac{f[t_1]-f[t_0]}{t_1-t_0} & \\
f\,[t_1] = P_1 & & f[t_0,t_1,t_2] = \frac{f[t_1,t_2]-f[t_0,t_1]}{t_2-t_0} \\
 & f[t_1,t_2] = \frac{f[t_2]-f[t_1]}{t_2-t_1} & \\
f\,[t_2] = P_2 & & \\
 & & \\
\vdots & \vdots & \ldots \\
 & & \\
f\,[t_{n-2}] = P_{n-2} & & \\
 & f[t_{n-2},t_{n-1}] = \frac{f[t_{n-1}]-f[t_{n-2}]}{t_{n-1}-t_{n-2}} & \\
f\,[t_{n-1}] = P_{n-1} & & f[t_{n-2},t_{n-1},t_n] = \frac{f[t_{n-1},t_n]-f[t_{n-2},t_{n-1}]}{t_n-t_{n-2}} \\
 & f[t_{n-1},t_n] = \frac{f[t_n]-f[t_{n-1}]}{t_n-t_{n-1}} & \\
f\,[t_n] = P_n & &
\end{array}$$

The interpolant polynomial in the Newton basis is given by the linear combination

$$p(t) = \sum_{j=0}^{n} f[t_0, \ldots, t_j]N_j(t). \tag{7}$$

Notice that the divided difference only depend on the mesh of nodes, and not on the output mesh. Therefore you have to compute the divided differences for each coordinate (x, y and z) and evaluate (7) at the output mesh to obtain the graph of the curve.

1.6 Graphs

In octave 2D plots are computed with the command **plot**. 3D plots are obtained with **plot3**, sometimes they appear planar plots that you have to rotate.

2 Submission instructions

The programming projects will be done in groups of three people (and one of two because you are eleven new students in class).

Please, submit all project parts on the Moodle page for MAT300 in a zip. The name of the zip will be

MAT300_ project1_ "surnames of all members of group"

You should include in the zip:

- Octave codes for each part of the project.
- A **readme.text** file specifying which programs the zip contains, in which program is each task of the project, the authors and instructions for running the code.
- A **explanations.pdf** file with the mathematical background. This file will contain at least 4 and at most 6 pages. In this pdf you should present:
  - **the description of the problem:** what is the mathematical problem that the code solves (one sentence or two), what are the inputs and the expected outputs.
  - **the explanation of the numerical methods that you use to tackle the problem:** here you have to present the theory of every method you use, step by step, using the proper terminology, a correct notation and understandable mathematical formulation (you can use latex or any other editor). The explanations should be carried out for the general case, and not for particular examples. You can use the slides, the books of reference in the syllabus, or internet for information, but do not forget to include the references in the bibliography. If you look at Wikipedia as a source of information, be sure you understand the material, notation and formulation before doing copy-paste, and be sure that material is consistent with the contents of the course. The notation has to be consistent along the whole document, and if possible the same as the used in class and in the codes. Please, use a common form along the document (same font, same size and so on).
  - **the relation between the above methods and your code:** you should link the above descriptions to the files and lines of code (what the code does and in which order, what is done-where is done).
  - **examples showing that your code indeed works:** you should compare your obtained results with analytical solutions. It worth to compute for a short number of nodes interpolant polynomials by hand and evaluate them in some points and compare with the computations.
  - **observations:** behavior of curves, problems of computations due to round off errors and machine precision, time of computation, comparison among different methods (which is faster) and other additional information that you may consider relevant.

  - **bibliography:** include the references that you used as literature.

3 Grading policy

The total grade of this programming assignment will be computed as follows:

- Codes 60%:
  - Gauss-Jordan works 8%
  - Lagrange works 14%
  - Newton works 15%
  - Meshes are correct 4%
  - Outputs in 2D and 3D are correct 4%
  - Codes are structured and clean. They implement Octave matrix and vector algebra 5%
  - Codes are well commented with headers and mathematical explanations with correct terminology in each line 10%
- Readme 4%
- Document 36%:
  - Description of the problem 3%
  - Mathematical explanation of the numerical methods that you use to tackle the problem: 20%
  - The relation between the above methods and your code: 5%
  - Examples: 3%
  - Observations: 5%

Concerning late submissions I will restrict to the late policy included in the syllabus.