# Data Fitting Report

Exercise 3

Comparison between interpolating polynomial and natural cubic spline

Cesare De Cal

Professor: Annie Cuyt

Assistant Professor: Ferre Knaepkens

# 1 Introduction

An imaginary chemistry experiment produces the following data:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $x_i$ | -1 | -0.96 | -0.86 | -0.79 | 0.22 | 0.50 | 0.93 |
| $f_i$ | -1.000 | -0.151 | 0.894 | 0.986 | 0.895 | 0.500 | -0.306 |

The goal of this exercise is to use these data points to compute and plot the interpolating polynomial together with the natural cubic spline, and to report what it is observed. In the following sections I am going to describe the tools used to compute and plot the interpolating polynomial and the natural cubic spine, and draw conclusions based on the plot.

## 2 Tools

The following programming language and libraries have been used in this exercise:

- Python 3.7
- SciPy

The following NumPy methods of the SciPy environment have been used in this exercise:

- numpy.array
- numpy.linspace
- numpy.polynomial.polynomial

The following Matplotlib methods of the SciPy environment have been used in this exercise:

- matplotlib.pyplot.plot
- matplotlib.pyplot.legend
- matplotlib.pyplot.show

This report was written in LaTeX. The results produced by the SciPy library were checked using Maple.

## 3 Computing the interpolating polynomial

To compute the interpolating polynomial, I use the Lagrange method which is one of the methods saw in class. I'll also calculate the roundoff errors and the errors on the coefficients.
The polynomial found is:

The coefficients are:

- 0.04365005085
- 16.0766955610565
- -0.048402197837630
- -20.10047681678335740
- 0.0168806991536721320
- 5.02946373595240442320
- -0.006446071786954468800

Here's a graph showing the 6th degree interpolating polynomial: