Gliwice, January 14, 2024

Mathematical and computer modeling of various technical and scientific problems – Documentation of the final laboratory project

Comparative analysis of methods for approximating functions

Authors	Jakub Pietrasik Bartosz Lewandowicz Szymon Hankus
Field of study	Informatics – sem. V

1 Project objective

The goal of the project is to explain, present, and compare three methods of approximating functions either from a set of given points or by approximating the function itself. The methods that we chose are:

1. Interpolation using Lagrange polynomials
2. Fourier series
3. The least squares method

2 Lagrange polynomials

Lagrange polynomials are polynomials $w(x)$ that interpolate a given set of points $P = \{(x_0, y_0),\ (x_1, y_1), \dots ,\ (x_n, y_n)\}$ Which means that they satisfy the following condition:

$$\forall_{(x_i,y_i)\in P}\ w(x_i) = y_i$$

In other words, when a polynomial **interpolates** the dataset, it passes through all of the points in said dataset. The Lagrange interpolating polynomial is the unique polynomial of lowest degree that interpolates a given set of data

Langrage polynomials for a given set of data are calculated using the following formula

$$w(x) = \sum_{i=0}^{n} y_i \cdot \prod_{\substack{0\le j\le n \\ j\neq i}} \frac{x - x_j}{x_i - x_j}$$

In the next section, I would like to demonstrate how this formula can be obtained and that it is actually quite straightforward and intuitive.

2.1 How to arrive at the formula for Lagrange polynomials?

This formula might seem daunting at first, but let us first consider a polynomial $w_i(x)$ that **goes through only one point** (x_i, y_i) **and is equal to 0 for the rest of the points**.

1. For a given pair (x_i, y_i), it has to be equal to y_i

2. For each pair (x_j, y_j) other than (x_i, y_i), it has to be equal to 0

Let's start with condition no. 2. It tells us that the polynomial $w(x)$ should be equal to 0 for all x_j inputs, that are not x_i. That is actually quite simple – all we have to do is construct a polynomial in factored form with the roots being all the x_j values other than x_i.

$$w_i(x) = \prod_{\substack{0 \le j \le n \\ j \neq i}} (x - x_j)$$

Let's now consider a simplified condition no. 1 – let's find a polynomial that is equal to 1 for (x_i, x_i), but also satisfies condition no. 2. That is actually also very simple – we just have to divide the polynomial that we have already constructed by $(x_i - x_j)$

$$w_i(x) = \prod_{\substack{0 \le j \le n \\ j \neq i}} \frac{(x - x_j)}{(x_i - x_j)}$$

Now,

$$w_i(x_i) = \prod_{\substack{0 \le j \le n \\ j \neq i}} \frac{(x_i - x_j)}{(x_i - x_j)} = 1$$

In order for the polynomial to satsify cond. no. 1, all we have to do is multiply it by y_i

$$w_i(x) = y_i \prod_{\substack{0 \le j \le n \\ j \neq i}} \frac{(x - x_j)}{(x_i - x_j)}$$

Condition no. 1 is satisfied,

$$w_i(x_i) = y_i \prod_{\substack{0 \le j \le n \\ j \neq i}} \frac{(x_i - x_j)}{(x_i - x_j)} = y_i \cdot 1 = y_i$$

Since both of the conditions are satisfied, $w_i(x)$ interpolates one point – (x_i, y_i) and is equal to 0 for all the other points in the dataset.

Below is a sample dataset comprising three points $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ and the values that polynomials w_i take for each of them.

	x_1	x_2	x_3
w_1	y_1	0	0
w_2	0	y_2	0
w_3	0	0	y_3

To find a polynomial that interpolates all of the points, we simply create $w_i(x)$ for every point and take their sum.

$$w(x) = \sum_{i=0}^{n} w_i(x)$$

Which expands to

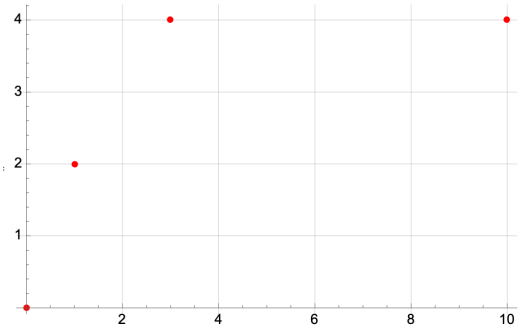
$$w(x) = \sum_{i=0}^{n} y_i \cdot \prod_{\substack{0 \le j \le n \\ j \neq i}} \frac{x - x_j}{x_i - x_j}$$

We have arrived at the exact formula for Lagrange polynomials.

2.2 Examples

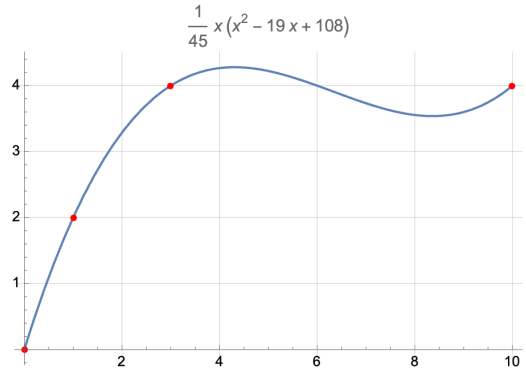
Let us consider a very simple set of four points

```
points = {{0, 0}, {1, 2}, {3, 4}, {10, 4}};
Show[
 ListPlot[points, PlotStyle → Red, GridLines → Automatic]
]
```



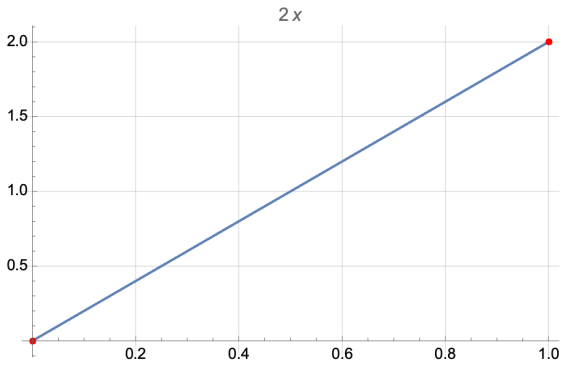
Now, let's use the `Langrage` module that we have written, and create a Lagrange polynomial interpolating the above-mentioned set of points.

```
points = {{0, 0}, {1, 2}, {3, 4}, {10, 4}};
Show[
 Plot[Lagrange[points], {x, 0, 10}, PlotRange → Full,
   PlotLabel → Simplify[Lagrange[points]]],
 ListPlot[points, PlotStyle → Red, PlotRange → Full],
 GridLines → Automatic
]
```



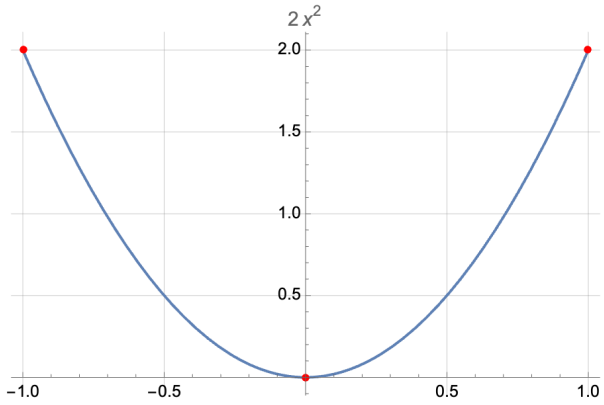
As was previously mentioned, the Lagrange interpolating polynomial is a polynomial the of lowest degree that interpolates a given set of data. Thus, feeding two points into our module will yield a linear function that interpolates these points.

```
points = {{0, 0}, {1, 2}};
Show[
 Plot[Lagrange[points], {x, 0, 1}, PlotRange → Full,
   PlotLabel → Simplify[Lagrange[points]]],
 ListPlot[points, PlotStyle → Red, PlotRange → Full],
 GridLines → Automatic
]
```



Similarly, three points will yield a quadratic (parabolic) function.

```
points = {{-1, 2}, {0, 0}, {1, 2}};
Show[
 Plot[Lagrange[points], {x, -1, 1}, PlotRange → Full,
   PlotLabel → Simplify[Lagrange[points]]],
 ListPlot[points, PlotStyle → Red, PlotRange → Full],
 GridLines → Automatic
]
```



2.3 Pros and cons of using Lagrange polynomials for function approximation

Pros

- The function goes exactly through the set of points given as the input,
- The formula is quite easy to understand, as I tried to demonstrate in section 2.2,
- Even when the points are not evenly spaced, it can be used to find the interpolating function.

Cons

- As new points are added to the dataset, the function quickly becomes computationally complex,
- Unsuitable for cases where the general shape of the function is known e.g. when we know the function we are looking for should be roughly logarithmic, it's better to use the least-squares function approximation

3 Fourier series

Fourier series is a powerful mathematical tool used to represent periodic functions as a sum of sine and cosine functions. Named after the French mathematician Joseph Fourier, who introduced the concept in the 19th century, Fourier series find applications in various scientific and engineering domains, particularly in the analysis of periodic signals such as sound waves and electrical signals.

For a periodic function $f(t)$ with period L, the Fourier series is given by:

$$f(t) = a_0 + \sum_{n=1}^{\infty} \left(a_n \cos\left(\frac{2\pi nt}{L}\right) + b_n \sin\left(\frac{2\pi nt}{L}\right)\right)$$

The coefficients a_0, a_n, and b_n are determined by the following integrals:

$$a_0 = \frac{1}{L}\int_0^L f(t)\,dt$$

$$a_n = \frac{2}{L}\int_0^L f(t)\cos\left(\frac{2\pi nt}{L}\right)\,dt$$

$$b_n = \frac{2}{L}\int_0^L f(t)\sin\left(\frac{2\pi nt}{L}\right)\,dt$$

These coefficients represent the amplitudes of the different sine and cosine components in the Fourier series. The term $\frac{2\pi nt}{L}$ in the trigonometric functions ensures that the series captures the periodic nature of the function.

Example of using Fourier Series to approximate a function:

What will be the Fourier series of the function $f(x) = 1 - x^2$ in the interval $[-1, 1]$?

Given,

$$f(x) = 1 - x^2, \quad [-1, 1]$$

The Fourier series of the function $f(x)$ in the interval $[-L, L]$, i.e. $-L \leq x \leq L$, is written as:

$$f(x) = A_0 + \sum_{n=1}^{\infty} A_n \cos\left(\frac{n\pi x}{L}\right) + \sum_{n=1}^{\infty} B_n \sin\left(\frac{n\pi x}{L}\right)$$

Here,

$$A_0 = \frac{1}{2L}\int_{-L}^{L} f(x)\,dx$$

$$A_n = \frac{1}{L}\int_{-L}^{L} f(x)\cos\left(\frac{n\pi x}{L}\right)\,dx, \quad n > 0$$

$$B_n = \frac{1}{L}\int_{-L}^{L} f(x)\sin\left(\frac{n\pi x}{L}\right)\,dx, \quad n > 0$$

Now, by applying the formula for $f(x)$ in the interval $[-1, 1]$:

$$f(x) = \frac{1}{2}\cdot 1\cdot\int_{-1}^{1}(1-x^2)\,dx + \sum_{n=1}^{\infty}\frac{1}{1}\int_{-1}^{1}(1-x^2)\cos(n\pi x)\,dx\cdot\cos(n\pi x)$$

$$+\sum_{n=1}^{\infty}\frac{1}{1}\int_{-1}^{1}(1-x^2)\sin(n\pi x)\,dx\cdot\sin(n\pi x)$$

Now simplifying the definite integrals,

$$= \frac{1}{2}\cdot\frac{4}{3} + \sum_{n=1}^{\infty}\frac{-4(-1)^n\pi^2n^2}{n^2\pi^2}\cos(n\pi x) + \sum_{n=1}^{\infty}0\cdot\sin(n\pi x)$$

$$= \frac{2}{3} + \sum_{n=1}^{\infty}\frac{4(-1)^n}{n^2}\cos(n\pi x)$$

Thus, the Fourier series of the function $f(x) = 1 - x^2$ in the interval $[-1, 1]$ is: $\frac{2}{3} + \sum_{n=1}^{\infty}\frac{4(-1)^n}{n^2}\cos(n\pi x)$

3.1 Real-life Applications of Fourier Series

1. **Audio Processing:**
   - **Music Compression:** Fourier series is used in audio compression algorithms (like MP3) for efficient representation and compression of audio signals.
2. **Telecommunications:**
   - **Signal Transmission:** Fourier series is used in techniques like Frequency Division Multiplexing (FDM) to transmit multiple signals simultaneously over a communication channel.
3. **Electrical Engineering:**
   - **Power Systems:** Fourier series is employed in power systems analysis to study periodic waveforms and analyze harmonic content in AC circuits.
4. **Image Processing:**
   - **Image Compression:** Fourier series is used in image compression algorithms (like JPEG) for efficient representation and compression of images.
5. **Physics:**
   - **Wave Analysis:** Fourier series is used in physics to analyze and describe periodic phenomena, such as the vibration of strings or the behavior of waves.
6. **Medical Imaging:**
   - **MRI (Magnetic Resonance Imaging):** Fourier Transform techniques, related to Fourier series, are fundamental in MRI for data transformation and image reconstruction.
7. **Chemistry:**
   - **Spectroscopy:** Fourier Transform Infrared (FTIR) spectroscopy uses techniques related to Fourier series for the analysis of molecular absorption and emission of light.

4 The least squares method

The least squares fitting method is a widely used technique in statistical analysis and regression modeling. It aims to find the best-fitting line or curve for a set of data points by minimizing the sum of the squares of the vertical distances (residuals) between the data points in the set and the corresponding points on the fitted line or curve.

4.1 How to fit a line to a set of points?

When trying to find the line of best fit for a given data set, we will be using a linear function $f(x) = ax + b$ as a basis for the least squares method.

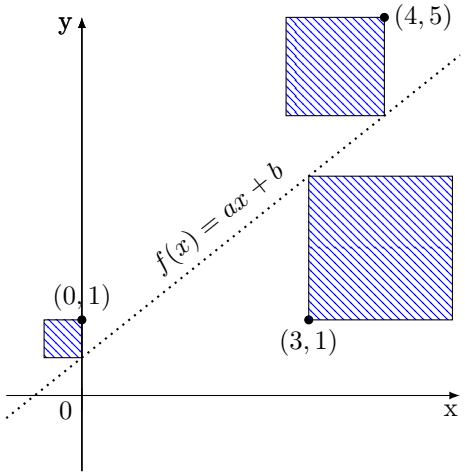
Let's imagine that we are looking for a line $f(x) = ax + b$ that best fits a set of points : $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. In order to measure how much the line deviates from the points, we can introduce residuals (vertical distances) $\xi_i = |f(x_i) - y_i|$.

The problem of finding that line boils down to minimizing the sum of the squares of the residuals.

4.2 Geometric interpretation

Let's imagine that we are looking for a line $f(x) = ax + b$ that best fits a set of points : $\{(0, 1), (3, 1), (4, 5)\}$. In order to find this line, we have to find a formula for the squares of the residuals and minimize it.

Geometrically, it presents as follows:



The distance in the vertical direction between the point and the line is the residual ξ of that point. The squares represent ξ^2 – which is the parameter that we're trying to minimize.

4.3 The optimization problem

The parameter that we're trying to minimize (optimize) is the sum of the squares of the residuals $R^2 = \sum_{i=1}^{n} \xi_i^2$ with respect to the parameters of the linear function a, b.

$$R^2 = \sum_{i=1}^{n} \xi_i^2 = \sum_{i=1}^{n} [y_i - (ax_i + b)]^2$$

The condition for R^2 to be a minimum is that the derivatives of R^2 with respect to

a and b are both equal to zero.

$$\frac{\partial R^2}{\partial a} = -2\sum_{i=1}^{n} x_i[y_i - (ax_i + b)] = 0$$
$$\frac{\partial R^2}{\partial b} = -2\sum_{i=1}^{n} [y_i - (ax_i + b)] = 0$$

Which we can quickly verify with Wolfram Mathematica

```
In[1]:= D[(y - a * x - b) ^ 2, a]
Out[1]= -2 x (-b - a x + y)

In[2]:= D[(y - a * x - b) ^ 2, b]
Out[2]= -2 (-b - a x + y)
```

This leads to the equations

$$na + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$$
$$a\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i$$

Which ultimately leads to the solution for a and b

$$a = \frac{\sum_{i=1}^{n} y_i \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$
$$b = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

4.4 Examples

```
data = {{1.3, 2.2}, {2.7, 3.5}, {3.2, 5.8}, {4.5, 6.7}, {5.9, 9.2}}
Show[
 ListPlot[data, PlotStyle → Red],
 Plot[Fit[data, {1, x}, x] /. x → xv, {xv, 0, 10}]]
```

```
{{1.3, 2.2}, {2.7, 3.5}, {3.2, 5.8}, {4.5, 6.7}, {5.9, 9.2}}
```

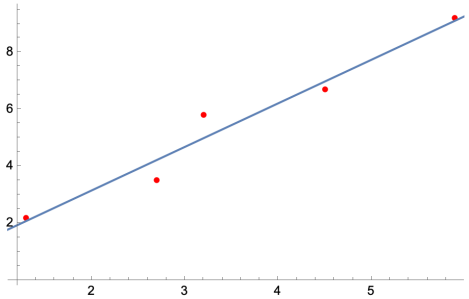


Figure 1: A linear function fit to the dataset

```
data = {{1.3, 8.2}, {2.7, 3.5}, {3.2, 2.8}, {4.5, 6.7}, {5.9, 9.2}}
Show[
 ListPlot[data, PlotStyle → Red],
 Plot[Fit[data, {1, x, x^2}, x] /. x → xv, {xv, 0, 10}]]
```

```
{{1.3, 8.2}, {2.7, 3.5}, {3.2, 2.8}, {4.5, 6.7}, {5.9, 9.2}}
```

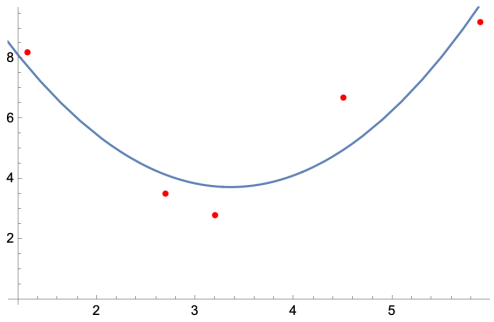


Figure 2: A parabolic function fit to the dataset

4.5 Pros and cons of using the least squares method for function approximation

Pros

- For cases where the general shape of the function is known e.g. when we know the function we are looking for should be roughly parabolic, it gives us the best fitting function.
- It's not very computationally complex.

Cons

- The function is not guaranteed to go exactly through any of the points in the dataset.

5 Summary

Enclosures

- File with the program (`Pietrasik_Lewandowicz_Hankus.nb`)