****

**APPLICATIONS OF POLYNOMIAL INTERPOLATION**

**REPORT BY**: Suman Mitra

**ROLL NO.**: 2020202018

**DEPARTMENT**: CSIS (PG1-2020)

**SUBJECT**: LINEAR ALGEBRA

**PROJECT GROUP**: 8

**DATE**: 21st Nov, 2020

**ABSTRACT:**

In this report, I will be focusing on one of the computation methods for polynomial interpolation: **Lagrange Polynomials.** Other than this, I will also briefly explain the **Vandermonde Method** and **Newton Polynomials**, though they will be covered in details by other team members.

The applications of **Polynomial Interpolation** like **Karatsuba** multiplication**, Tom-Cook** multiplication, Secure Multi Party Computation **(SMPC)**, **Secret Sharing** algorithm etc. will be again covered by other team members.

1. **INTRODUCTION**

**Polynomial interpolation** is a method of estimating values between known data points. When graphical data contains a gap, but data is available on either side of the gap or at a few specific points within the gap, an estimate of values within the gap can be made by interpolation.

The simplest method of interpolation is to draw straight lines between the known data points and consider the function as the combination of those straight lines. This method, called **linear interpolation**, usually introduces considerable error. Therefore, a more precise approach uses a **polynomial** function to connect the points. Polynomials are used to approximate complicated curves.

There are mainly three techniques for finding interpolating polynomials:

* The Vandermonde method. (easy to calculate and generalize)
* Lagrange polynomials. (easy to find manually)
* Newton polynomials. (efficient to implement when using Horner's rule)

In this report, I will mainly focus on the **Lagrange polynomials**. I will also briefly touch upon the **Vandermonde method** and **Newton polynomials**, though these topics will be covered by other team members in details.

1. **POLYNOMIAL INTERPOLATION AS A LINEAR PROBLEM:**

**Definition:** Given points which are pair-wise disjoint (i.e. no two points are equal) and a function defined at these points. We can find a polynomial (where = Set of all polynomials with degree equal to or less than ) such that:

Though the polynomial may not fit the function very well outside the given points If we choose to represent as a linear combination of monomials, then the problem of finding is equivalent to solving a linear problem with variables (the coefficients of the monomials) and constraints:

We can rewrite this system of equations as:

where, ; and

The matrix formed here is known as **Vandermonde Matrix.**

**2.1. THE VANDERMONDE METHOD**

The Vandermonde method is the most straight-forward method which may be used to find an interpolating polynomial. The substitution of the points into the desired polynomial yields a system of linear equations in the coefficients of the polynomial which may then be solved using Gaussian elimination or PLU decomposition. This technique also generalizes for interpolating non-polynomial functions or points in more than two dimensions.

Suppose we have points (2, 5), (3, 6), (7, 4) and we want to fit a quadratic polynomial through these points. The general form of a quadratic polynomial is:

.

Thus, if we were to simply evaluate at these three points, we get three equations:

Therefore, we may draw the obvious generalization that given points , we can find a polynomial of degree which passes through those points by doing the following:

* Writing down the general polynomial of degree .
* Evaluating the polynomial at the points .
* Solving the resulting system of linear equations.

Rather than performing all of these operations, we will simply write down the problem in the form where is the vector of values, **c** is the vector of coefficients, and is th**e Vandermonde matrix**.

* 1. **DRAWBACKS OF THIS APPROACH**

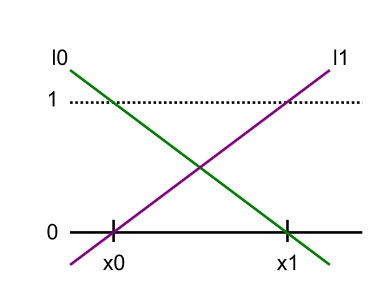
Transforming the problem of finding a polynomial into an equivalent linear problem has helped us to understand polynomial interpolation. However, it has some significant drawbacks as a method for solving polynomial interpolation problems. Those drawbacks are:

* Solving a linear system of equations takes a significant amount of time.
* This system of linear equations is often ill-conditioned and prone to round-off errors. In other words, although the **Vandermonde** matrix is **non-singular**, it may be close to **singular**, which makes it hard to solve accurately on a computer.
* It is not well-suited to a common kind of exploratory analysis. Suppose we are given . We solve , and we get a solution . Suppose that we do not like the solution for some reason, so we add a new point . Now we make a new system , and solve for . We have to do this from scratch. We cannot reuse all the computation we already invested to solve . If is large, this could be very bad.

This equivalence between solving a linear problem and finding a polynomial rests on our representation of as a linear combination of monomials. To find a better method, we should consider alternate representations. We shall mainly talk about one such class of representations for polynomial interpolation which are called **Lagrange Polynomials**.

1. **INTERPOLATION BY LAGRANGE POLYNOMIALS**

Given the interpolation points suppose we can find polynomials of degree . These polynomials are fixed and independent of the function values , and satisfy the following condition:

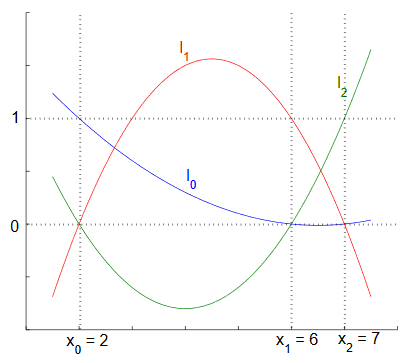
For example, given and , two such polynomials are shown in Figure 1.



* 1. **USING LAGRANGE POLYNOMIAL FOR INTERPOLATION**

Any polynomial can be represented as a linear combination of Lagrange Polynomials of degree . To illustrate the use of Lagrange Polynomials, consider the following example:

**Example-1:** Given the points , and the corresponding function values , we have to find an interpolating polynomial .





Consider that, if we had three quadratic Lagrange polynomials (as shown in **Figure 2**) with:

If we multiply each polynomial by the corresponding function value and then add them together, the resulting polynomial will match at each of the input points. So, we can define as:

Evaluating at the input point , we get:

In general, given a set of input points the corresponding points on the graph of the function and the Lagrange polynomials for our input points , we can define the following polynomial that interpolates :

What we have actually done is, initially, when we first formulated the problem of polynomial interpolation, we did not specify any particular representation of polynomials. When we started to do computation, we needed a computationally accessible definition, so we choose the most natural basis that we all are familiar with: **monomials**. For example, we used the monomials as a basis for . But this is only one basis. ***The Lagrange polynomials also form a basis for the space of polynomials. The difference is that the Lagrange basis is chosen so as to make our problem easier.***

* 1. **FINDING LAGRANGE POLYNOMIALS**

So, we can find a polynomial given its roots and a point with ) = 1. Consider that, if a polynomial has roots , , then it must have factors (and (). We can get our polynomial by first multiplying the factors together, obtaining a polynomial with the correct roots. We can then multiply by a constant scaling factor so that ) = 1.

To illustrate, consider **Example-1**. Sinceis **0** at **6** and **7**, it must have factors ( **- 6**) and ( **- 7**). If we consider ( **- 6**). ( **- 7**) evaluated at = 2, we see that its value will be (2−6).(2−7). So, if we divide by this value, we will have the correct polynomial:

We can define and similarly:

In general, we can calculate each Lagrange polynomial for a set of input points as:

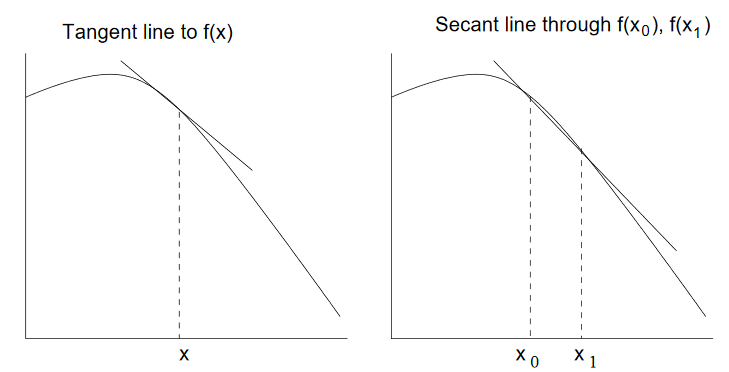
* 1. **ANALYSIS OF INTERPOLATION BY LAGRANGE POLYNOMIALS**

So we have seen that we can solve a polynomial interpolation problem quite easily using a linear combination of Lagrange polynomials. We will now analyze how easy it is to use the resulting polynomial for numerical computation. For example:

We might argue that evaluating the polynomial is fairly straight forward. However, it is easy to see that evaluating the derivative could be very complicated (consider using the product rule on terms with factors each).

Thus, with Lagrange polynomials, we have struck on a polynomial representation that makes it easy to solve polynomial interpolation problems, but potentially difficult to use once the problem has been solved.

Now we will see that how important it is that the polynomial representation we end up with is easy to use. Consider that we have already used polynomial interpolation without knowing it, when we approximated the derivative of a function.





Recalling that the derivative of a function at a point is simply the slope of the tangent line at that point. We can approximate the tangent line by taking two points , and interpolating a degree polynomial (a secant line) that approximates the function between these two points (as shown in **Figure-3**). We can then approximate the slope at by taking the derivative of the secant line. If we look at this problem as a polynomial interpolation problem, we might get a better estimate by interpolating over a larger number of points. In cases like this, clearly it is important that the representation of the resulting polynomial is easy to use.

* 1. **UNIQUENESS OF SOLUTIONS**

The process of interpolation by Lagrange polynomials shows that we can always find a solution to a polynomial interpolation problem.

Recalling that polynomial interpolation is equivalent to solving the linear problem:

From linear algebra, we know that the solution to this problem hinges on whether or not the matrix is singular. If is singular, then whether or not the equation has a solution depends on :

Is non-singular ∀ equation 5 has a unique solution.

is singular

Now, since we know that we can always solve a polynomial interpolation problem using Lagrange polynomials, we know that equation 5 always has a solution, regardless of . Since this is true, cannot be singular, and therefore, we know that not only does equation 5 always have a solution, but that solution is always unique.

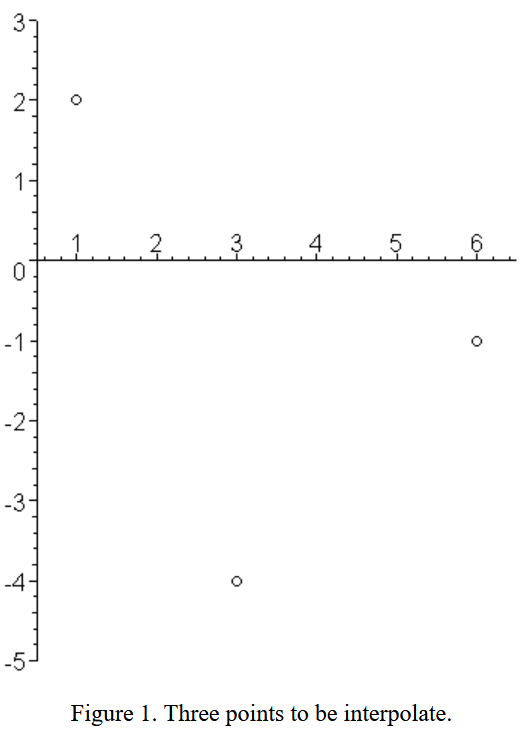
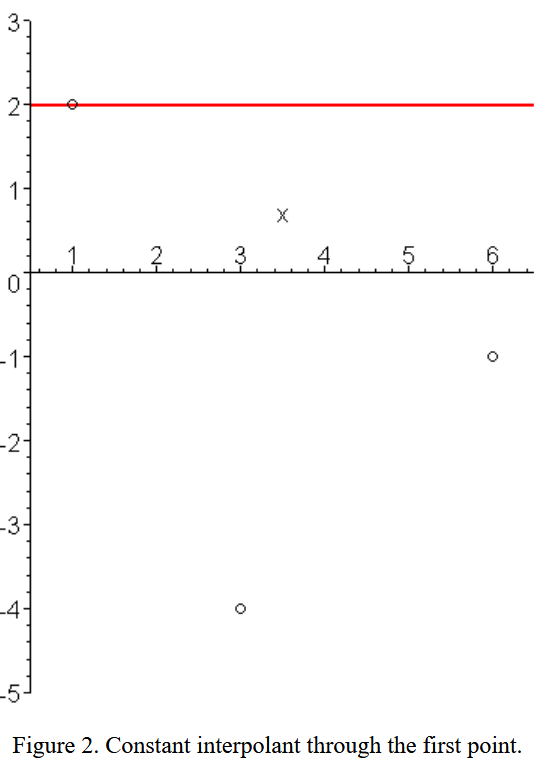
1. **NEWTON POLYNOMIALS**

With Lagrange polynomials, we found the interpolating polynomial by taking each point and finding a polynomial with the property that:

* , and
* for

This technique is the most easily implemented by humans (at least for linear and quadratic interpolating polynomials), but is also the least useful programmatically. Also, if we found the Lagrange polynomial passing through the points , this gives us very little information for finding the Lagrange polynomial which passes through these points together with a new point .

Let us, however, use this idea: given points, how do we find the polynomial which passes through the given points and another point ?

To demonstrate this, we will find the interpolating quadratic polynomial which passes through the three points in Figure 1.

We will do this by finding a sequence of polynomials , , and of degrees 0, 1, and 2 where the last will interpolate all three points.

**Constant (n = 1):**

To do this, let us start from the beginning with a single point . The polynomial passing through this point is . For sample points, is shown above in Figure 2.

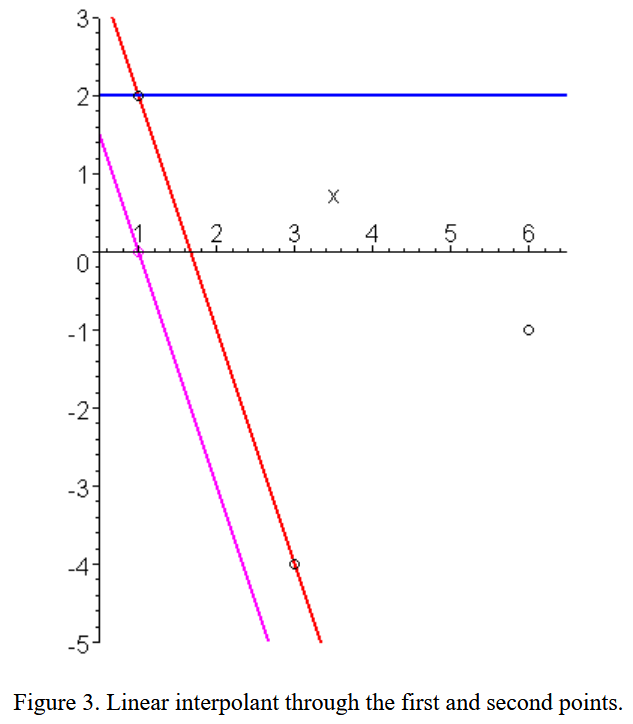
**Linear (n = 2):**

Having found , we will find by defining a polynomial Δ such that:

The first requirement makes it quite clear that the delta-polynomial must be of the form . We will use the second condition to determine :

Dividing through by , we get:

Given the example points, , and Figure 3 shows in blue, in magenta, and summing these together, we get that passes through both the first and second points.



**Quadratic:**

Given that the function passes through two points, we want to find a function such that passes through all three points.

By solving, we will get an equation like this:

where,  **;** and  **.**

In general, if we define the interpolating Newton polynomial of degree as:

1. **CONTRIBUTION BY TEAM MEMBERS**
2. **Krishnapriya:** Introduction to Polynomial Interpolation
3. **Abhiram:** Computation method- Vandermonde Matrix.
4. **Amanpreet:** Application- Karatsuba Algorithm.
5. **Ayushi:** Application- Tom-Cook Multiplication.
6. **Navya:** Application- Shamir’s Secret Sharing Scheme.
7. **Pradeep:** Application- Secure multi-party computation (SMPC).
8. **REFERENCES:**

* <https://en.wikipedia.org/wiki/Lagrange_polynomial>
* <https://ece.uwaterloo.ca/~dwharder/NumericalAnalysis/05Interpolation/>
* <https://www.youtube.com/watch?v=B2wqZKZv3O0>
* <https://www.youtube.com/watch?v=XK4G5Ndy-m8>
* <https://www2.math.uconn.edu/~leykekhman/courses/MATH3795/Lectures/Lecture_14_poly_interp.pdf>