UNIT 5:

Numerical Differentiation and Integration

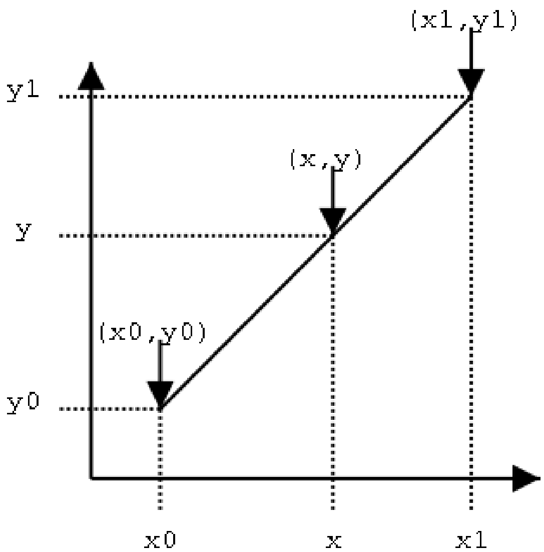
Linear Interpolation Formula

Linear interpolation is a method useful for curve fitting using linear polynomials. It helps in building new data points within the range of a discrete set of already known data points. Therefore, the Linear interpolation is the simplest method for estimating a channel from the vector of the given channel's estimates. It is very useful for data prediction, data forecasting, market research, and many other mathematical and scientific applications. This article will elaborate on this concept with Linear Interpolation Formula and suitable examples. Let us learn it!

What is Linear Interpolation?

Interpolation is a method for estimating the value of a function between any two known values. Often some relationship is there, and with the help of experiments at a range of values to predict other values. Interpolation is useful to estimate the function of the un-tabulated points. Interpolation is useful to estimate any desired value at some specific known coordinate point.

Linear interpolation is useful while searching for a value between given data points. Therefore mathematician considers it as "filling in the gaps" for a given data values in tabular format. The strategy for linear interpolation is to use a straight line to connect the given data points on positive as well as the negative side of the unknown point.



Often, Linear interpolation is not accurate for non-linear data. If the points in the data set to change by a large value, then linear interpolation may not give a good estimate. Also, it involves estimating a new value by connecting two adjacent known values with a straight line.

Formula of Linear Interpolation

Its simplest formula is given below:

$y = y_1 + (x - x_1)(y_2 - y_1) x_2 - x_1$

This formula is using coordinates of two given values to find the best fit curve as a straight line. Then this will give any required value of y at a known value of x.

In this formula, we are having terms as:

- x_1 and y_1 are the first coordinates
- x_2 and y_2 are the second coordinates
- x is the point to perform the interpolation
- y is the interpolated value.

Solved Examples for Linear Interpolation Formula

Q.1: Find the value of y at x = 4 given some set of values (2, 4), (6, 7).

Solution: Given the known values are,

$x=4$ $x_1=2$ $x_2=6$ $y_1=4$; $y_2=7$

The interpolation formula is,

$y=y_1+(x-x_1)(y_2-y_1)/x_2-x_1$

i.e. $y=4+(4-2)\times(7-4)(6-2)$

y = 112

Lagrange Interpolation Formula

Lagrange Interpolation Formula finds a polynomial called Lagrange Polynomial that takes on certain values at an arbitrary point. It is an n^{th} degree polynomial expression to the function f(x). The interpolation method is used to find the new data points within the range of a discrete set of known data points.

Lagrange Interpolation Formula

Given few real values x_1, x_2, x_3, …, x_n and y_1, y_2, y_3, …, y_n and there will be a polynomial P with real coefficients satisfying the conditions $P(x_i) = y_i$, $\forall i = \{1, 2, 3, …, n\}$ and degree of polynomial P must be less than the count of real values i.e., degree(P) < n. The Lagrange Interpolation formula for different orders i.e., n^{th} order is given as,

Lagrange Interpolation Formula for n^{th} order is-

If the Degree of the polynomial is 1 then,

Lagrange Interpolation Formula for 1^{st} order polynomial is-

Similarly for 2^{nd} Order polynomial, the Lagrange Interpolation formula is-

Proof of Lagrange Theorem
Let's consider a nth degree polynomial of given form

Substitute observations x_i to get A_i
Put $x = x_0$ then we get A_0
$f(x_0) = y_0 = A_0(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)\ldots(x_0 - x_n)$
$A_0 = y_0/(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)\ldots(x_0 - x_n)$
By substituting $x = x_1$ we get A_1
$f(x_1) = y_1 = A_1(x_1 - x_0)(x_1 - x_2)(x_1 - x_3)\ldots(x_1 - x_n)$
$A_1 = y_1/(x_1 - x_0)(x_1 - x_2)(x_1 - x_3)\ldots(x_1 - x_n)$
In similar way by substituting $x = x_n$ we get A_n
$f(x_n) = y_n = A_n(x_n - x_0)(x_n - x_1)(x_n - x_2)\ldots(x_n - x_{n-1})$
$A_n = y_n/(x_n - x_0)(x_n - x_1)(x_n - x_2)\ldots(x_n - x_{n-1})$
If we substitute all values of A_i in function f(x) where i = 1, 2, 3, …n then we get Lagrange Interpolation Formula.

Let's look into a few sample questions on Lagrange Interpolation Formula.

Sample Problems

Question 1: Find the value of y at x = 2 for the given set of points (1, 2),(3, 4)
Solution:
Given,

$(x_0, y_0) = (1, 2)$
$(x_1, y_1) = (3, 4)$
$x = 2$

As per the 1^{st} order Lagrange Interpolation Formula,

= (-2/-2) + (4/2)

= 1 + 2

y = 3

Newton Interpolation Polynomial

Newton's interpolation polynomial is a method used to approximate a function with a polynomial that passes through given data points. It is named after Sir Isaac Newton, who contributed to its development along with other mathematicians like John Wallis and James Gregory.

Given a set of data points (x_i, y_i), where x_i is the independent variable and y_i is the corresponding dependent variable, Newton's interpolation polynomial can be used to find an approximate polynomial function P(x) that passes through these points. The polynomial is represented in the form:

$P(x) = c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \dots + c_n(x - x_0)(x - x_1)\dots(x - x_{n-1})$

where $c_0, c_1, \dots, c_n$ are coefficients determined from the given data points. The interpolation polynomial has a degree of at most n, where n is the number of data points minus 1.

The process of constructing the Newton interpolation polynomial involves using divided difference coefficients, which are recursively calculated from the given data points. The divided difference coefficients are used to compute the coefficients $c_0, c_1, \dots, c_n$ in the polynomial expression. The divided differences can be expressed as follows:

$f[x_i] = y_i$ $f[x_i, x_{i+1}, \dots, x_j] = (f[x_{i+1}, x_{i+2}, \dots, x_j] - f[x_i, x_{i+1}, \dots, x_{j-1}]) / (x_j - x_i)$

where $f[x_i, x_{i+1}, \dots, x_j]$ represents the divided difference coefficient associated with the data points $(x_i, y_i), (x_{i+1}, y_{i+1}), \dots, (x_j, y_j)$.

The coefficients $c_0, c_1, \dots, c_n$ can be determined as follows:

$c_0 = f[x_0]$ $c_1 = f[x_0, x_1]$ $c_2 = f[x_0, x_1, x_2]$ … $c_n = f[x_0, x_1, \dots, x_n]$

Once the coefficients are calculated, the polynomial P(x) can be constructed and used for interpolation or approximation within the range of the given data points.

Newton's interpolation polynomial is a powerful tool for approximating functions from discrete data points and is commonly used in numerical analysis and interpolation problems.

Differentiating Continuous functions

To differentiate a continuous function, you apply the rules of differentiation to find its derivative. The derivative of a function represents the rate of change of the function with respect to its independent variable (usually denoted as x).

The standard rules of differentiation are as follows:

1. Constant Rule: d/dx [c] = 0, where c is a constant.
2. Power Rule: d/dx [x^n] = n * x^(n-1), where n is any real number.
3. Sum/Difference Rule: d/dx [f(x) + g(x)] = d/dx [f(x)] + d/dx [g(x)] d/dx [f(x) - g(x)] = d/dx [f(x)] - d/dx [g(x)]
4. Product Rule: d/dx [f(x) * g(x)] = f(x) * d/dx [g(x)] + g(x) * d/dx [f(x)]
5. Quotient Rule: d/dx [f(x) / g(x)] = (g(x) * d/dx [f(x)] - f(x) * d/dx [g(x)]) / (g(x))^2
6. Chain Rule: d/dx [f(g(x))] = f'(g(x)) * g'(x), where f'(g(x)) represents the derivative of f with respect to its inner function g(x), and g'(x) is the derivative of g(x).
7. Trigonometric Functions: d/dx [sin(x)] = cos(x) d/dx [cos(x)] = -sin(x) d/dx [tan(x)] = sec^2(x)
8. Exponential Function: d/dx [e^x] = e^x

Using these rules, you can differentiate more complex functions by breaking them down into simpler components. Remember that a continuous function is one that has no abrupt jumps or holes in its graph, meaning it can be continuously drawn without lifting the pen.

Keep in mind that finding the derivative of a continuous function is an important concept in calculus and has many practical applications in fields such as physics, engineering, economics, and more.

Differentiating tabulated functions

When you have a tabulated function (also known as discrete data points), the process of differentiation is slightly different from differentiating continuous functions. Instead of applying the standard rules of differentiation, you use numerical differentiation techniques to approximate the derivatives at specific data points.

The simplest and most commonly used method for numerical differentiation of tabulated functions is the finite difference method. There are two main finite difference approximations:

1. Forward Difference: The forward difference formula is used to approximate the derivative at a particular data point (x_i) using the data points (x_i, y_i) and (x_{i+1}, y_{i+1}) that are adjacent to it.

   $f'(x_i) \approx (y_{i+1} - y_i) / (x_{i+1} - x_i)$
   This method provides an approximation of the derivative at the data point x_i based on the slope of the secant line between the adjacent data points.
2. Central Difference: The central difference formula is used to approximate the derivative at a particular data point (x_i) using the data points (x_{i-1}, y_{i-1}), (x_i, y_i), and (x_{i+1}, y_{i+1}) surrounding it.

   $f'(x_i) \approx (y_{i+1} - y_{i-1}) / (x_{i+1} - x_{i-1})$
   This method provides a more accurate approximation of the derivative by taking into account the slopes on both sides of the data point x_i.

It's important to note that numerical differentiation using finite difference methods introduces some errors due to the discrete nature of the data points. The accuracy of the approximation depends on the step size (the difference between adjacent x-values) and the smoothness of the function being approximated. Smaller step sizes generally result in better approximations, but they can also lead to more significant round-off errors.

When using numerical differentiation, it's crucial to be aware of these limitations and choose an appropriate step size based on the data and the desired level of accuracy. Additionally, other numerical differentiation techniques, such as higher-order difference formulas, can also be used to improve accuracy and reduce errors.

Newton- cotes methods of integration

Newton-Cotes methods are numerical techniques used to approximate the definite integral of a function over a specified interval. These methods use equally spaced points (nodes) within the interval and approximate the integral by constructing a polynomial that passes through these points. The integral of the polynomial is then taken as an approximation of the original integral. The two most commonly used Newton-Cotes methods are the Trapezoidal Rule and the Simpson's Rule.

1. Trapezoidal Rule: The Trapezoidal Rule is the simplest Newton-Cotes method and approximates the area under the curve by approximating it with trapezoids. It divides the interval [a, b] into equally spaced segments and approximates the function as a straight line within each segment.

The integral is then approximated by the sum of the areas of the trapezoids. The formula for the Trapezoidal Rule is as follows:

$\int[a, b] f(x) dx \approx h * [f(a) + 2 * (f(a + h) + f(a + 2h) + ... + f(a + (n-1)h)) + f(b)]$

where h is the step size (distance between adjacent nodes), n is the number of segments, and $h = (b - a) / n$.

2. Simpson's Rule: Simpson's Rule is a more accurate Newton-Cotes method that approximates the function within each segment with a quadratic polynomial (parabola). It divides the interval [a, b] into equally spaced segments and fits a parabola through three consecutive points within each segment.

The integral is then approximated by the sum of the areas under the parabolas. The formula for Simpson's Rule is as follows:

$\int[a, b] f(x) dx \approx h/3 * [f(a) + 4 * f(a + h) + 2 * f(a + 2h) + ... + 2 * f(a + (n-2)h) + 4 * f(a + (n-1)h) + f(b)]$

where h is the step size (distance between adjacent nodes), n is the number of segments, and $h = (b - a) / n$.

Both the Trapezoidal Rule and Simpson's Rule offer reasonably accurate approximations for well-behaved functions, but Simpson's Rule generally provides higher accuracy for smooth functions due to its use of quadratic polynomials instead of straight lines. However, it's essential to keep in mind that the accuracy of these methods depends on the number of segments used (higher n leads to better approximations) and the smoothness of the function being integrated.