INTERPOLACIÓN POLINÓMICA

CONTENIDO

- INTRODUCCIÓN
- EQUIPO
- LAGRENGE
- NEWTON
- CONCLUSIONES

INTRODUCCIÓN

La interpolación polinómica es una técnica usada para estimar valores entre puntos conocidos. El método de Lagrange utiliza una fórmula explícita basada en polinomios base, útil para pocos datos. En cambio, el método de Newton usa diferencias divididas y permite añadir nuevos puntos fácilmente. Ambos son fundamentales en el análisis numérico para aproximar funciones.

EQUIPO DE TRABAJO

Brandon García Ordaz

Ivan Pedro Suarez

Alexis Martin Cortes de Lucio

Ingrid Ortega Ramos

Brandon Joshua Labastida Jimenez

Priscila Cortés Ramírez

Oliver Espinosa Sánchez

INTERPOLACIÓN POLINÓMICA DE LAGRANGE

OBJETIVO

Aproximar una función utilizando un único polinomio que pasa exactamente por un conjunto de puntos dados.

IDEA PRINCIPAL

Construye un polinomio único que interpola una serie de puntos (x0,y0),(x1,y1),...,(xn,yn), usando sumas de productos de términos conocidos como polinomios base de Lagrange.

FÓRMULA GENERAL

$$P(x) = \sum_{i=0}^{n} y_i \cdot L_i(x)$$

FÓRMULA GENERAL

Donde:

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

VENTAJAS

- No requiere derivadas.
- Fórmula explícita y directa.
- Fácil de implementar.
- Buena aproximación con pocos puntos.

DESVENTAJAS

- Si se agregan más puntos, el polinomio debe volver a calcularse desde cero.
- No es eficiente para muchos puntos (puede causar oscilaciones: Fenómeno de Runge).

EJEMPLO SIMPLE

Sean los puntos:

(1,2), (2,3), (4,1)

El polinomio de Lagrange sería:

$$P(x) = 2 \cdot L_0(x) + 3 \cdot L_1(x) + 1 \cdot L_2(x)$$

EJEMPLO SIMPLE

🧮 Paso 1: Construimos los polinomios base

- Para $L_0(x)$:

Usamos todos los puntos menos el primero (es decir, usamos x = 2 y x = 4):

$$L_0(x) = \frac{x-2}{1-2} \cdot \frac{x-4}{1-4} = \frac{x-2}{-1} \cdot \frac{x-4}{-3} = (2-x) \cdot \left(\frac{4-x}{3}\right)$$

EJEMPLO SIMPLE

- **Para $L_1(x)$:**

Usamos todos los puntos menos el segundo (usamos x = 1 y x = 4):

$$L_1(x) = \frac{x-1}{2-1} \cdot \frac{x-4}{2-4} = (x-1) \cdot \frac{x-4}{-2} = (x-1)\left(-\frac{x-4}{2}\right)$$

- **Para $L_2(x)$:**

Usamos x = 1 y x = 2:

$$L_2(x) = \frac{x-1}{4-1} \cdot \frac{x-2}{4-2} = \frac{x-1}{3} \cdot \frac{x-2}{2}$$

EJEMPLO SIMPLE

🧮 Paso 2: Sustituimos en el polinomio total

$$P(x) = 2 \cdot L_0(x) + 3 \cdot L_1(x) + 1 \cdot L_2(x)$$

Sustituimos cada $L_i(x)$:

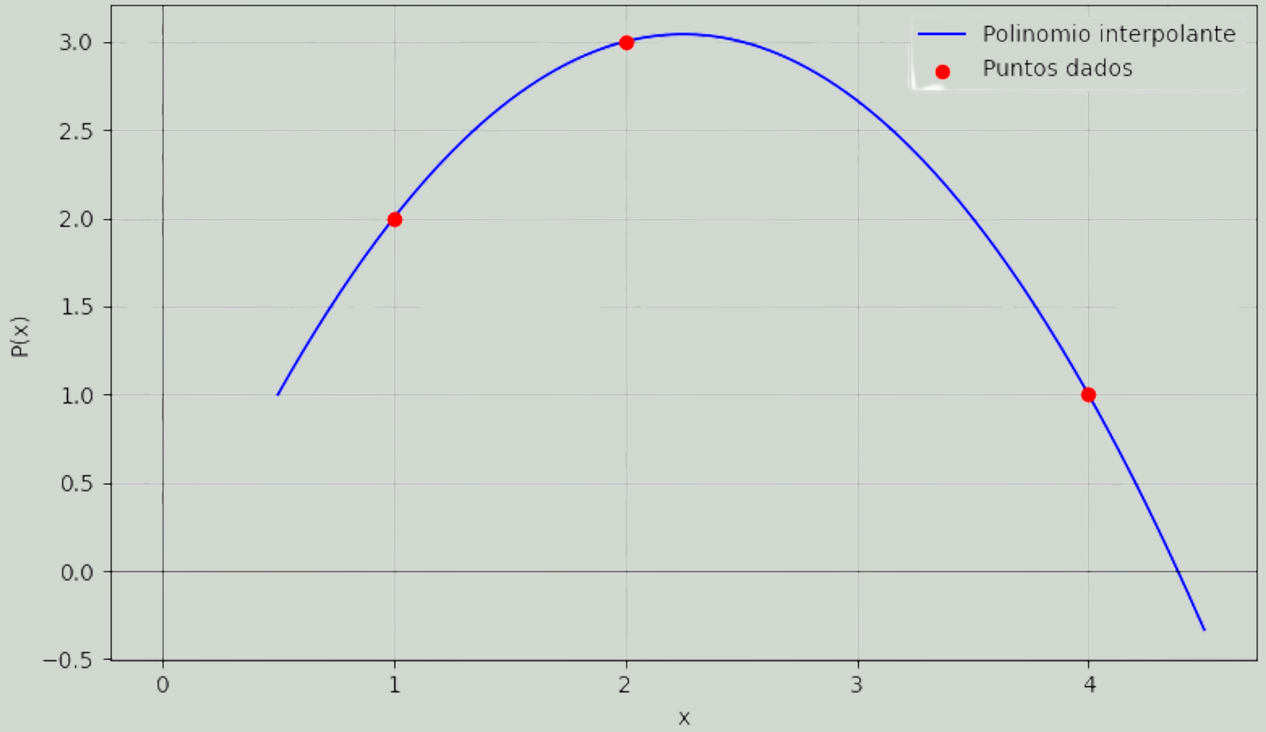
$$P(x) = 2 \cdot \left(\frac{x - 2}{-1} \cdot \frac{x - 4}{-3} \right) + 3 \cdot \left((x - 1) \cdot \frac{x - 4}{-2} \right) + 1 \cdot \left(\frac{x - 1}{3} \cdot \frac{x - 2}{2} \right)$$

EJEMPLO SIMPLE

El polinomio interpolante que pasa por los puntos (1,2),(2,3),(4,1) es:

$$P(x) = -\frac{2}{3}x^2 + 3x - \frac{1}{3}$$

G R Á F I C A


Polinomio interpolante
Puntos dados
P(x)
x
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
0
1
2
3
4

PSEUDOCÓDIGO

```
// Entrada de datos
IMPRIMIR "Ingrese el número de puntos:"
LEER n
// Crear arreglos para almacenar los puntos
CREAR arreglo x de tamaño n de tipo real
CREAR arreglo y de tamaño n de tipo real
IMPRIMIR "Ingrese los puntos (x, y):"
// Bucle para leer las coordenadas de cada punto
PARA i DESDE 0 HASTA n-1 HACER
  IMPRIMIR "x[" + i + "] = "
  LEER x[i]
  IMPRIMIR "y[" + i + "] = "
  LEER y[i]
FIN PARA
// Mostrar los polinomios base de Lagrange
IMPRIMIR "\nPolinomios de base de Lagrange:"
// Bucle para cada punto
PARA i DESDE 0 HASTA n-1 HACER
  IMPRIMIR "L_" + i + "(x) = "
  LLAMAR SUBALGORITMO imprimirPolinomioBaseLagrange(x, i)
  IMPRIMIR "" // Nueva línea
FIN PARA
// Mostrar la forma general del polinomio interpolante
IMPRIMIR "\nEl polinomio interpolante P(x) es la suma de y_i * L_i(x):"
// Bucle para construir la suma
PARA i DESDE 0 HASTA n-1 HACER
  IMPRIMIR y[i] con formato de 2 decimales
  IMPRIMIR " * L_" + i + "(x)"
  SI i < n - 1 ENTONCES
    IMPRIMIR " + "
  FIN SI
FIN PARA
IMPRIMIR "" // Nueva línea

// Cerrar la entrada
// (Implícito en el final del algoritmo principal)

FIN ALGORITMO

// SUBALGORITMO imprimirPolinomioBaseLagrange
// Entrada: arreglo de reales x, entero i
SUBALGORITMO imprimirPolinomioBaseLagrange(x, i)
  n = LONGITUD(x)
  CREAR cadena sb (StringBuilder)
```

PSEUDOCÓDIGO

```
  // Bucle para construir el numerador del polinomio base
  PARA j DESDE 0 HASTA n-1 HACER
    SI i NO ES IGUAL A j ENTONCES
      CONCATENAR "(x - " a sb
      CONCATENAR x[j] a sb
      CONCATENAR ")" a sb
      SI j < n - 1 Y i NO ES IGUAL A n - 1 Y i NO ES IGUAL A j + 1 ENTONCES
        CONCATENAR " * " a sb
      FIN SI
    FIN SI
  FIN PARA

  CONCATENAR " / " a sb
  denominador = 1 // Inicializar el denominador

  // Bucle para calcular el denominador
  PARA j DESDE 0 HASTA n-1 HACER
    SI i NO ES IGUAL A j ENTONCES
      denominador = denominador * (x[i] - x[j])
    FIN SI
  FIN PARA

  CONCATENAR "(" a sb
  CONCATENAR denominador a sb
  CONCATENAR ")" a sb
  IMPRIMIR sb
FIN SUBALGORITMO
```

CÓDIGO EN JAVA

```java
import java.util.Arrays;
import java.util.Scanner;

public class InterpolacionLagrange {

    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println(x:"Ingrese el número de puntos:");
        int n = scanner.nextInt();

        double[] x = new double[n];
        double[] y = new double[n];

        System.out.println(x:"Ingrese los puntos (x, y):");
        for (int i = 0; i < n; i++) {
            System.out.print("x[" + i + "] = ");
            x[i] = scanner.nextDouble();
            System.out.print("y[" + i + "] = ");
            y[i] = scanner.nextDouble();
        }

        System.out.println(x:"\nPolinomios de base de Lagrange:");
        for (int i = 0; i < n; i++) {
            System.out.print("L_" + i + "(x) = ");
            imprimirPolinomioBaseLagrange(x, i);
            System.out.println();
        }

        System.out.println(x:"\nEl polinomio interpolante P(x) es la suma de y_i * L_i(x):");
        for (int i = 0; i < n; i++) {
            System.out.printf(format:"%.2f * L_%d(x)", y[i], i);
            if (i < n - 1) {
                System.out.print(s:" + ");
            }
        }
        System.out.println();
        scanner.close();
    }

    public static void imprimirPolinomioBaseLagrange(double[] x, int i) {
        int n = x.length;
        StringBuilder sb = new StringBuilder();
        for (int j = 0; j < n; j++) {
            if (i != j) {
                sb.append(str:"(x - ").append(x[j]).append(str:")");
                if (j < n - 1 && i != n - 1 && i != j + 1) {
                    sb.append(str:" * ");
                }
            }
        }
        sb.append(str:" / ");
        double denominador = 1;
        for (int j = 0; j < n; j++) {
            if (i != j) {
                denominador *= (x[i] - x[j]);
            }
        }
        sb.append(str:"(").append(denominador).append(str:")");
        System.out.print(sb.toString());
    }
}
```

INTERPOLACIÓN DE NEWTON

CONCEPTO

La interpolación de Newton es un método numérico que utiliza diferencias divididas para construir un polinomio que pasa por un conjunto de puntos dados. Su estructura recursiva permite agregar nuevos datos sin recalcular completamente el polinomio, siendo eficiente y versátil en diversas aplicaciones.

IDEA PRINCIPAL

Construir un polinomio de interpolación de forma progresiva (o regresiva) usando diferencias divididas, aprovechando la estructura triangular que facilita agregar más puntos sin rehacer todo el cálculo.

FÓRMULA GENERAL

$$P_n(x)=f[x_0]+f[x_0,x_1](x-x_0)+f[x_0,x_1,x_2](x-x_0)(x-x_1)+\ldots$$
$$+f[x_0,\ldots,x_n](x-x_0)\ldots(x-x_{n-1})$$

VENTAJAS

- Fácil actualización: Permite añadir nuevos puntos sin recalcular todo el polinomio.
- Eficiencia computacional: Requiere menos operaciones que otros métodos, como el de Lagrange.
- Flexibilidad: Funciona con datos no equidistantes ni ordenados.

DESVENTAJAS

- Complejidad para muchos puntos: Puede volverse complicado y propenso a errores con grandes conjuntos de datos.
- El resultado puede variar si se cambia el orden de los puntos.
- Menor precisión en extrapolaciones: La precisión disminuye al estimar valores fuera del rango de los datos conocidos.

EJEMPLO PRACTICO

Tenemos 3 mediciones de temperatura:

- 9:00 → 15°C
- 12:00 → 21°C
- 15:00 → 18°C

Queremos estimar la temperatura a las 13:30 (x = 13.5).

PASO 1: CÁLCULO DE DIFERENCIAS DIVIDIDAS DE PRIMER ORDEN

$$f[x_1,x_2] = (f(x_2) - f(x_1))/(x_2 - x_1)$$

Estas representan la tasa de cambio entre dos puntos consecutivos.

Para calcular **f[9,12]:**

- Tomamos los valores f(9) = 15 y f(12) = 21
- Aplicamos la fórmula: f[9,12] = (f(12) - f(9))/(12 - 9)
- Sustituyendo: f[9,12] = (21 - 15)/(12 - 9) = 6/3 = 2

Esto significa que la temperatura aumenta 2°C por hora entre las 9:00 y 12:00

Para calcular **f[12,15]:**

- Tomamos los valores f(12) = 21 y f(15) = 18
- Aplicamos la fórmula: f[12,15] = (f(15) - f(12))/(15 - 12)
- Sustituyendo: f[12,15] = (18 - 21)/(15 - 12) = -3/3 = -1

Esto significa que la temperatura disminuye 1°C por hora entre las 12:00 y 15:00

DIFERENCIAS DIVIDIDAS DE SEGUNDO ORDEN

Estas miden cómo cambia la tasa de cambio.

$f[12,15] = -1$

$f[9,12] = 2$

Para calcular f[9,12,15]:

$$f[x_1,x_2,x_3] = (f[x_2,x_3] - f[x_1,x_2])/(x_3 - x_1)$$

- Aplicamos la fórmula: **f[9,12,15] = (f[12,15] - f[9,12])/(15 - 9)**
- Sustituyendo: f[9,12,15] = (-1 - 2)/(15 - 9) = -3/6 = -0.5

Esto indica que hay un cambio en la tendencia de la temperatura (de subir a bajar)

CONSTRUCCIÓN DEL POLINOMIO DE NEWTON

Estas miden cómo cambia la tasa de cambio.

El polinomio de Newton para 3 puntos tiene la forma:

$$P(x) = f(x_0) + f\underline{x_0,x_1} + f\underline{x_0,x_1,x_2}(x-x_1)$$

Donde:

- $x_0 = 9$, $x_1 = 12$, $x_2 = 15$
- $f(x_0) = f(9) = 15$
- $f[x_0,x_1] = f[9,12] = 2$ (calculado en el paso anterior)
- $f[x_0,x_1,x_2] = f[9,12,15] = -0.5$ (calculado en el paso anterior)

Sustituyendo estos valores en la fórmula:

$$P(x) = 15 + 2(x-9) + (-0.5)(x-9)(x-12)$$

EVALUACIÓN DEL POLINOMIO PARA X = 13.5

$$P(x) = 15 + 2(x-9) + (-0.5)(x-9)(x-12)$$

Ahora sustituimos x = 13.5 en el polinomio que acabamos de construir:

$$P(13.5) = 15 + 2(13.5-9) + (-0.5)(13.5-9)(13.5-12)$$

La temperatura estimada a las 13:30 es aproximadamente 20.6°C.

P S E U D O C Ó D I G O

Entrada: Arreglos x[n], y[n] con los puntos
Salida: Polinomio interpolado evaluado en un valor dado

Crear matriz DD[n][n] para diferencias divididas
Para i = 0 hasta n-1:
    DD[i][0] ← y[i]
Para j = 1 hasta n-1:
    Para i = 0 hasta n-j-1:
        DD[i][j] ← (DD[i+1][j-1] - DD[i][j-1]) / (x[i+j] - x[i])
Leer valor de x_eval a evaluar
resultado ← DD[0][0]
producto ← 1
Para i = 1 hasta n-1:
    producto ← producto * (x_eval - x[i-1])
    resultado ← resultado + DD[0][i] * producto
Retornar resultado

CÓDIGO EN JAVA

CONCLUSIONES

Tanto la interpolación de Lagrange como la de Newton son herramientas clave para construir polinomios que ajusten datos conocidos y permitan estimar valores intermedios con precisión. Aunque cada método tiene sus ventajas según el contexto simplicidad en Lagrange y flexibilidad en Newton, ambos son fundamentales para resolver problemas donde no se conoce la función exacta, facilitando así el análisis y modelado en ingeniería y ciencias aplicadas.

INTERPOLACIÓN POLINÓMICA

MÉTODOS NUMÉRICOS

INTERPOLACIÓN Y AJUSTE DE FUNCIONES