Университет ИТМО
Факультет программной инженерии и компьютерной техники

Лабораторная работа №5

«Вычислительная математика»

Выполнил:
Студент группы P32102
Гулямов Т.И.

Преподаватель:
Рыбаков С.Д.

Санкт-Петербург
2023

Цель лабораторной работы

Цель лабораторной работы: решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

Порядок выполнения работы

1. Вычислительная реализация задачи
2. Программная реализация задачи

Вычислительная реализация задачи

Исходные данные:

x	0.50	0.55	0.60	0.65	0.70	0.75	0.80
y	1.5320	2.5356	3.5406	4.5462	5.5504	6.5559	7.5594

Конечные разности:

x	y	∆y	∆2y	∆3y	∆4y	∆5y	∆6y
0.50	1.5320	1.0036	0.0014	-0.0008	-0.0012	0.0059	-0.0166
0.55	2.5356	1.005	0.0006	-0.002	0.0047	-0.0107	
0.60	3.5406	1.0056	-0.0014	0.0027	-0.006		
0.65	4.5462	1.0042	0.0013	-0.0033			
0.70	5.5504	1.0055	-0.002				
0.75	6.5559	1.0035					
0.80	7.5594						

Формула Ньютона:

$t = (x - x_0)/h = (0.502 - 0.50)/0.05 = - 0.04$

$N_n(x) = y_i + t\Delta y_i + \frac{t(t-1)}{2!}\Delta^2 y_i + ... + \frac{t(t-1)...(t-n+1)}{n!}\Delta^n y_i$

$N(0.502) = y_i + t\Delta y_i + \frac{t(t-1)}{2!}\Delta^2 y_i + ... + \frac{t(t-1)(t-2)(t-3)(t-4)(t-5)}{6!}\Delta^6 y_i$

$N(0.502) = 1.5320 + 0.04 * 1.0036 + \frac{0.04(0.04-1)}{2!}0.0014 + ...$

$N(0.502) = 1.5320 + 0.0401 - 0.00003 + 0.00002 + 0.00001 + 0.00004 + 0.0001 = 1.57229$

Формула Гаусса:

$t = (x - x_0)/h = (0.645 - 0.65)/0.05 = - 0.1$

$P_n(x) = y_0 + t\Delta y_{-1} + \frac{t(t+1)}{2!}t\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}t\Delta^3 y_{-2} + ... + \frac{(t+n-1)...(t-n+1)}{(2n)!}\Delta^{2n-1} y_{-n} + \frac{(t+n)(t+n-1)...(t-n+1)}{(2n)!}\Delta^2$

$P(0.645) = 4.5462 - 0.01 * 1.0056 - \frac{-0.09}{2}0.001 - \frac{0.099}{6}0.002 + \frac{0.1881}{24}0.0047 - \frac{0.39501}{120}0.0059 + \frac{1.14}{720}$

$P(0.645) = 4.445714$

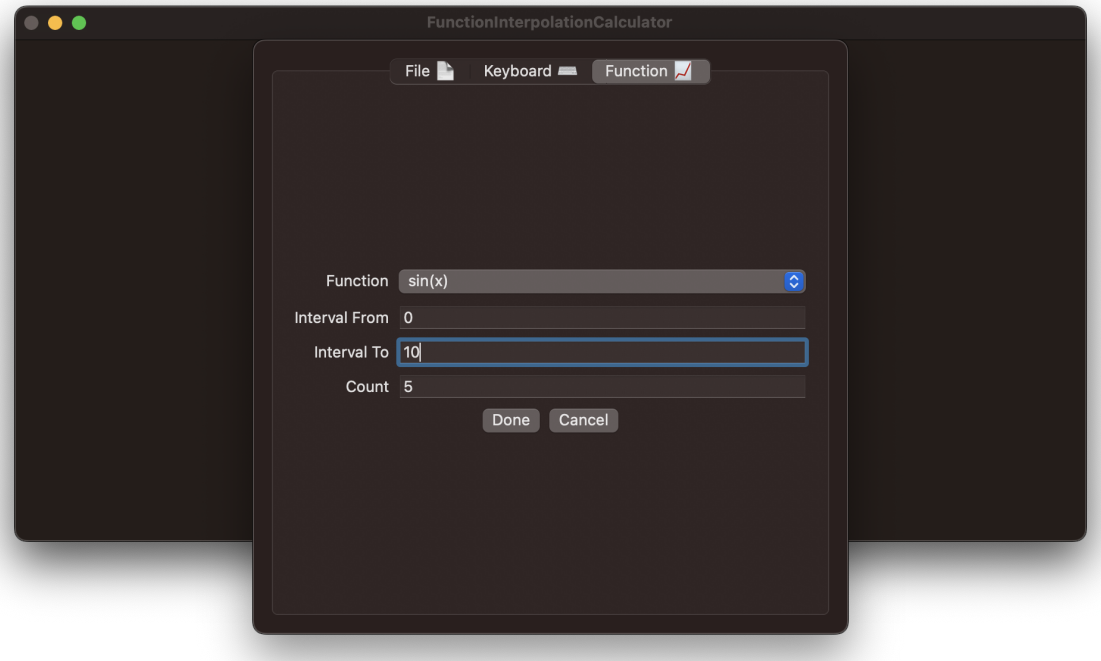
Листинг программы

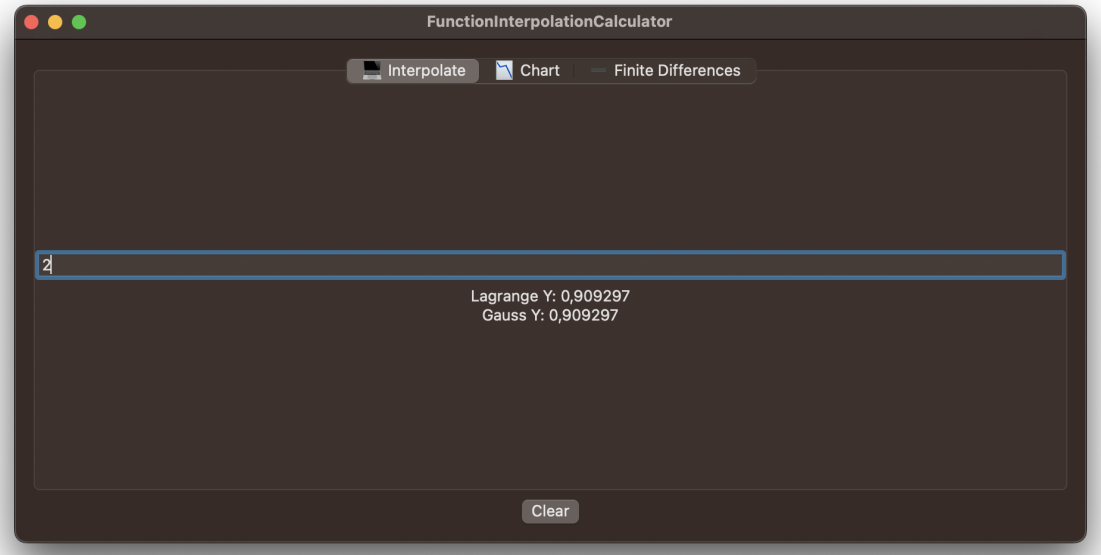
```
struct LagrangePolynomial {
  let points: [Point]

  func callAsFunction(_ x: Double) -> Double {
    self.points.reduce(0) { sum, i in
      sum + i.y * self.points.reduce(1) { mult, j in
        i == j ? mult : mult * (x - j.x) / (i.x - j.x)
      }
    }
  }
}
```

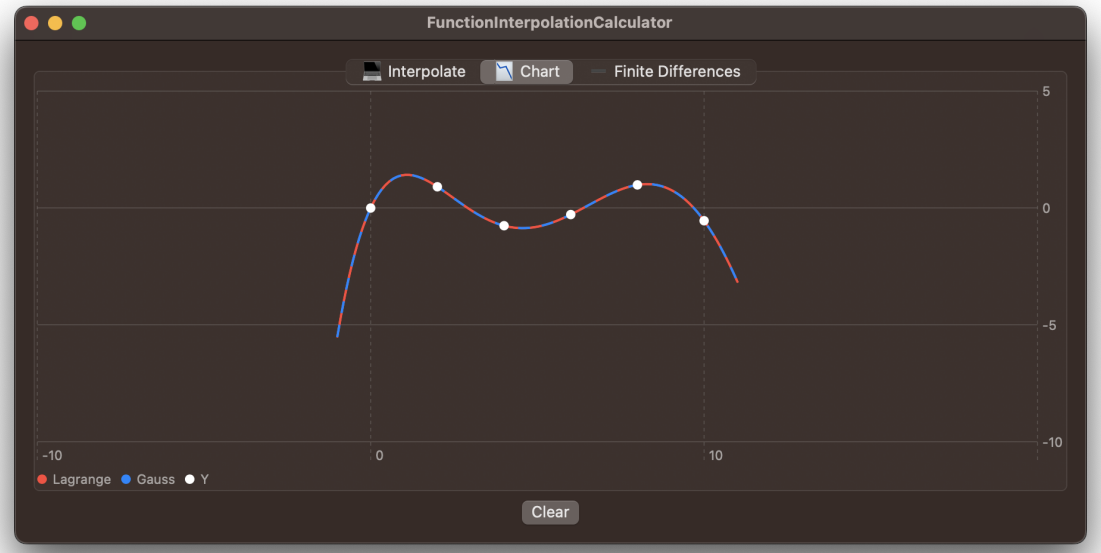
```
struct GaussPolynomial {
  enum Error: Swift.Error {
    case incorrectPoints
  }

  init(points: [Point]) throws {
    guard points.count > 2 else {
      throw Error.incorrectPoints
    }

    guard points
      .map(\.x).slidingWindow2()
      .map(-).allEquals({ abs($0 - $1) < eps })
    else {
      throw Error.incorrectPoints
    }

    self.middlePointForward = points[(points.count - 1) / 2]
    self.middlePointBackward = points[points.count / 2]
    self.stepLength = points[1].x - points[0].x
    self.count = points.count
    self.differences = FiniteDifferences(
      points.map(\.y)
    )
  }

  let count: Int
  let stepLength: Double
  let middlePointForward: Point
  let middlePointBackward: Point
  let differences: FiniteDifferences

  var middlePoint: Point {
```

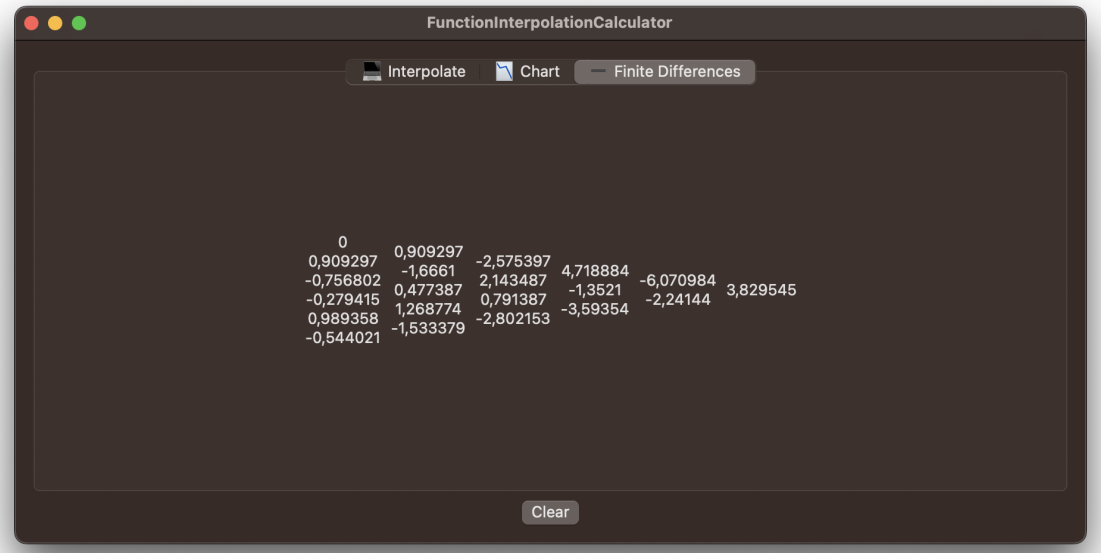
```
    .init(
      x: (self.middlePointForward.x + self.middlePointBackward.x) / 2,
      y: (self.middlePointForward.y + self.middlePointBackward.y) / 2
    )
  }

  func callAsFunction(_ x: Double) -> Double {
    let isForward = x > self.middlePoint.x
    let t = isForward
      ? (x - self.middlePointForward.x) / self.stepLength
      : (x - self.middlePointBackward.x) / self.stepLength

    var numerator = 1.0
    var denominator = 1.0
    var result = isForward
      ? self.middlePointForward.y
      : self.middlePointBackward.y

    for i in 1..<self.count {
      // Calculate finite difference
      let difference = isForward
        ? self.differences[forward: i, -(i / 2)]
        : self.differences[backward: i, -(i /+ 2)]

      // Calculate factorial
      denominator *= Double(i)

      // Calculate numerator
      switch (isForward, i.isMultiple(of: 2)) {
      case (false, true), (true, false):
        numerator *= t + Double((i - 1) /+ 2)
      case (false, false), (true, true):
        numerator *= t - Double((i - 1) /+ 2)
      }

      result += difference * numerator / denominator
    }

    return result
  }
}
```

Результаты выполнения программы









Вывод

Во время выполнения лабораторной работы познакомился с методом интерполяции функции. Научился использовать и реализовывать программно метод Гаусса и метод Лагранжа. Получил ценные знания, которые несомненно пригодятся в будущем.