Understanding & Computing Riemann Sums (in an iPhone app)

Andrew Owen - yumaikas94@gmail.com - www.junglecoder.com
Michael MacCallum - mickmaccallum@comcast.net

Overview

- Riemann Sums
- Programming
- Regex

Motivations

- Create an application that can…
  - Check our homework
  - Solve an interesting problem

Objectives

- Intuitive usage (TI-83+ like syntax)
- Freedom
  - No Internet required
  - No rate limiting
  - Integrate on the go! (But not while driving)

What is a Riemann Sum?

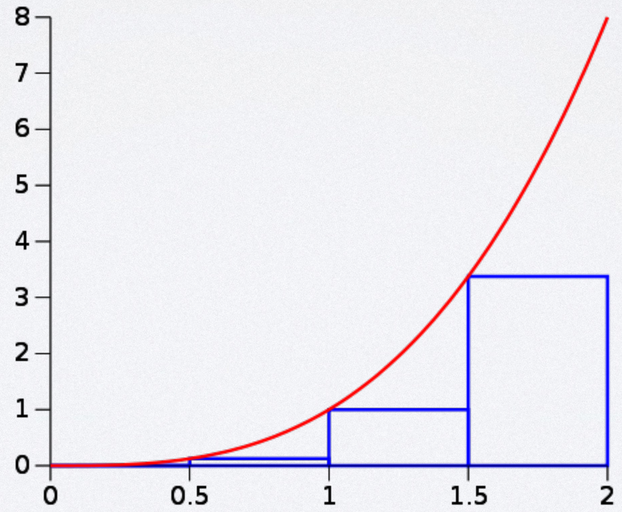
- Numerical method for approximating integrals
- Calculated by summing the area of N rectangles or trapezoids below a function

Four Methods to Calculate Riemann Sums

- Left
- Right
- Middle
- Trapezoidal

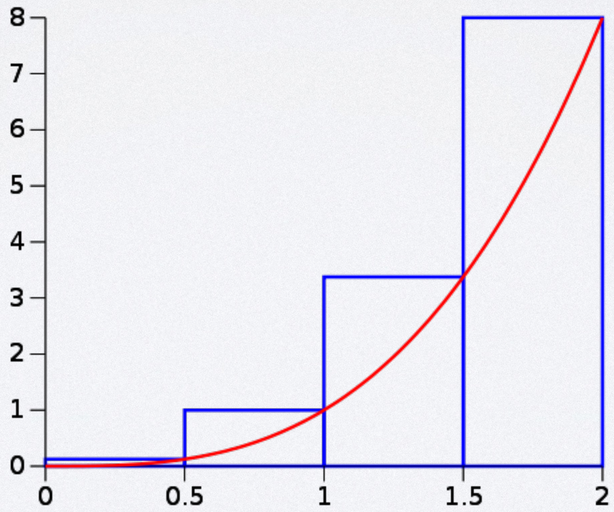
Left Sum

$$\Delta x[f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \cdots + f(b - \Delta x)]$$



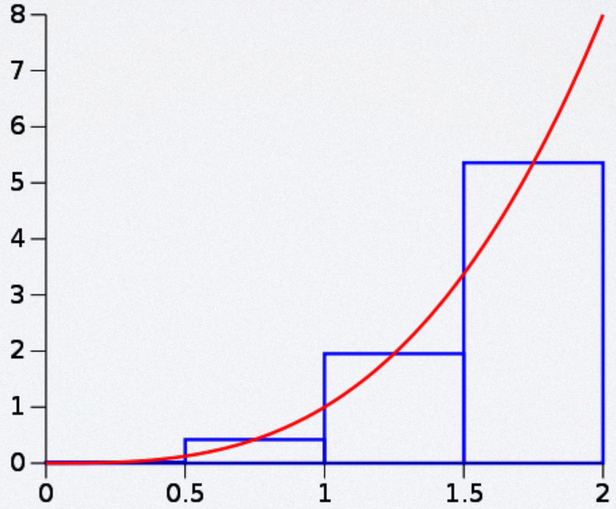
Right Sum

$$\Delta x[f(a+\Delta x)+f(a+2\Delta x)+\cdots+f(b)]$$



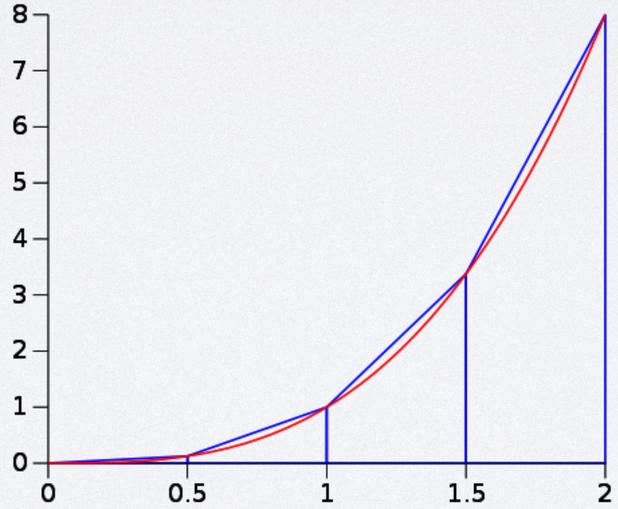
Middle Sum

$$\Delta x[f(a+\frac{\Delta x}{2})+f(a+\frac{3\Delta x}{2})+\cdots+f(b-\frac{\Delta x}{2})]$$



Trapezoid Method

$$\frac{\Delta x}{2}[f(a) + 2f(a + \Delta x) + 2f(a + 2\Delta x) + 2f(a + 3\Delta x) + \cdots + f(b)]$$



What isn't a Riemann Sum?

- Any function that isn't integrable
- Any definite integral with a vertical asymptote (ex: $\int_0^b \frac{1}{x} dx$)
- Areas larger than double max (around 1.797^308 for programs)

Implementation

- Used 3rd party math parsers
- Evaluate functions for every value of N * Delta X
- Summed the results

Implementation

- Math Parsers
  - Objective-C - DDMathParser
    - https://github.com/davedelong/DDMathParser/
  - C# - YAMP
    - https://github.com/FlorianRappl/YAMP

Implementation

```
double h = (b - a) / n;

for (unsigned int i = 0 ; i < n ; i ++) {

   @autoreleasepool {

        double x = a + i * h;

        NSDictionary * const substitutions = @{var: @(x)};

        NSNumber *eval = [expression evaluateWithSubstitutions:substitutions evaluator:nil error:&parseError];

        double y = [eval doubleValue];

        sum += y;
    }
}

sum *= h;
```

Challenges

- Converting formulas for intuitive syntax
- Resolving performance issues
- Worked around floating point rounding errors

Converting formulas for Intuitive syntax

- Aiming for TI-83-like syntax
- “^” → “**” - DDMathParser (Objective-C)
- “2x” → “2*(x)” - YAMP (C#)
- “2sinx” → “2*sin(x)” - YAMP (C#)

Resolving Performance Issues

- Comparing the programs
- Objective-C was 100x slower than C#
- Formula was being re-parsed for every value of x instead of being reused.
- Fixing this lead to a 50x speed up

Worked Around Floating Point Rounding Errors

- Computers don't do fractions the way we do
- $0.1 + 0.1 \approx 0.2$

Conclusion

- Definite integrals can be calculated on a mobile device, without the need to use the Internet or the Wolfram platform.

Open Source

This presentation - https://github.com/NSPostWhenIdle/RiemannSumsProject

Python Implementation - https://github.com/NSPostWhenIdle/PythonRiemannSums

C# Implementation - https://github.com/yumaikas/RiemannSums

Objective-C Implementation - https://github.com/NSPostWhenIdle/Riemann-Sums