Mathematical Examples of Functions that Produce and Consume Functions

(or, more concisely)

Higher-order Functions in Math

Quiz Feedback

- *“I hope my mental math serves me well”*
  - Student will need to clarify (by email)
- *“Are we going to use large amounts of calculus in this class?”*
  - No. We focus on more foundational math
- *“… I feel like I'm getting a little more lost every chapter.”*
  - In class show of hand says many thought the same
  - Please 1) ask more questions in class, 2) use quiz for asking questions, and 2) email discussion and teachers lists.

Higher-order Functions (HOFs)

- Producing a function
  - add: number -> (number -> number)
- Consuming a function
  - map: (X -> Y) [X] -> [Y]
- Doing both
  - map’: (X -> Y) -> ([X] -> [Y])
- Street terms: “call backs” and “objects”

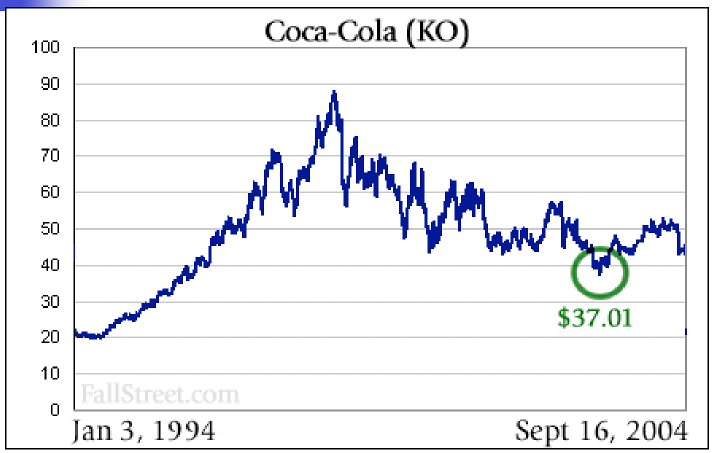
Plan for today

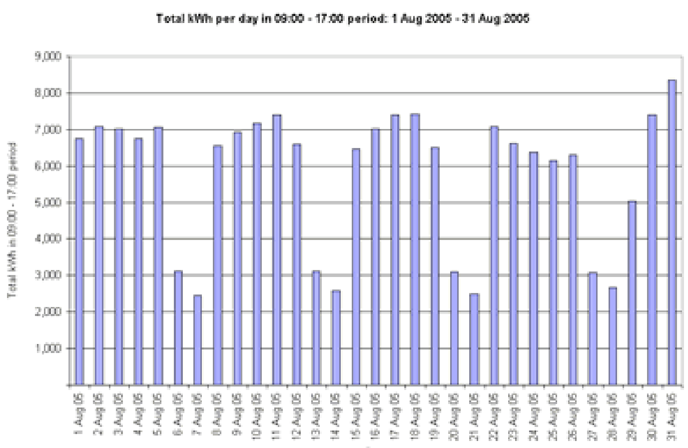
There are many real-world problems that are solved effectively and elegantly with HOFs.

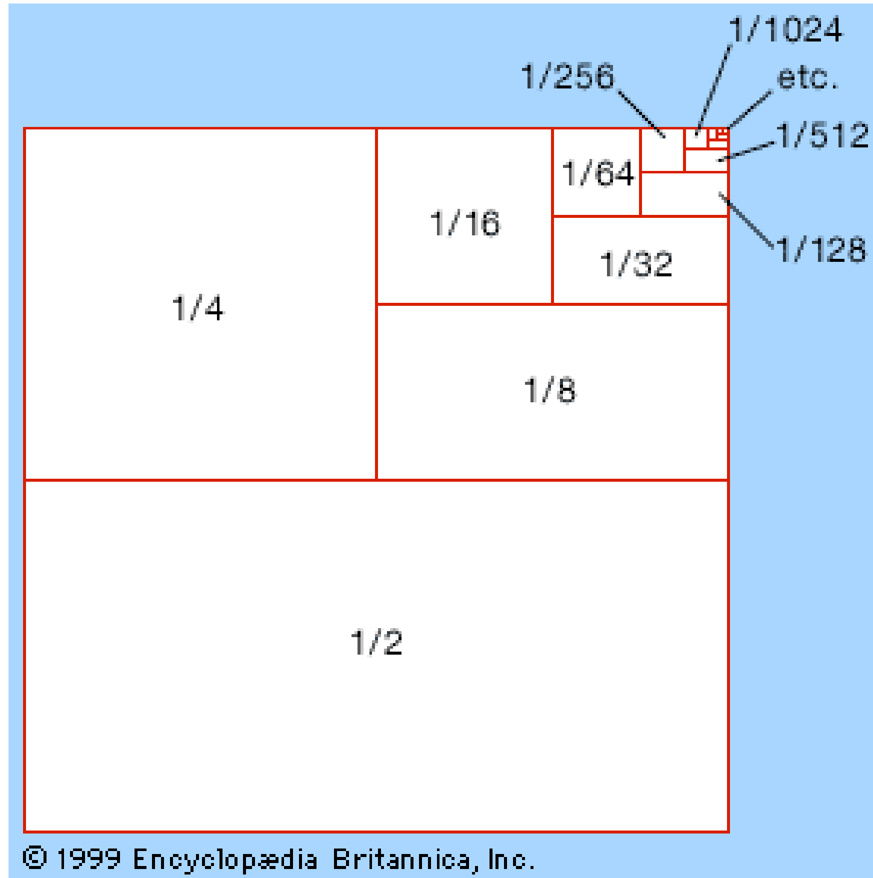
Examples:

- Sequences and Series
- Integration (area under function)
- Differentiation (slope of line)

Sequences and Series







Sequences and Series

- Lists can represent finites sequences only
  - [1, 1/2, 1/4, 1/8]

- Functions can represent even *infinite* ones
  - f(0) = 1, f(1) = 1/2, f(2) =1/4, ...
  - (define (f n) (/ 1 (expt 2 n)))

Other Examples of Sequences

```
; series : N -> R
(define (f-numbers n) n)

(define (f-even n) (* 2 n))

(define (f-odd n) (+ 1 (* 2 n)))

(define (f-squares n) (* n n))
```

What Can we Do with Sequences?

```
; let N be natural numbers
; let R be real numbers
; sum : N (N -> R) -> R
 (define (sum n f)
    (cond
      [(= n 0) (f n)]
      [else (+ (f n)
                (sum (- n 1) f))]))
```

Examples

```
  (sum 3 f-even)
= ... (+ 0 2 4 6) = 12

  (sum 4 f-squares)
= ... (+ 0 1 4 9 16) = 30
```

More Examples (In class)

- `series-map:  (R->R) (N -> R) -> (N -> R)`
- `series-map:  (X->Y) (N -> X) -> (N -> Y)`

- `apply-taylor:  (N->R) N R -> R`

- `product:  (N->R) N -> R`

- `sum-upto :  (N->R) -> (N->R)`

Example (test case)

```
(sum-upto <3,2,3,4,5...>)
= ... <3, 5, 8, 12, 17, ...>

// Note:  <...> stands for a
//   function representing a
//   sequence (for readability)
```

Types of Series

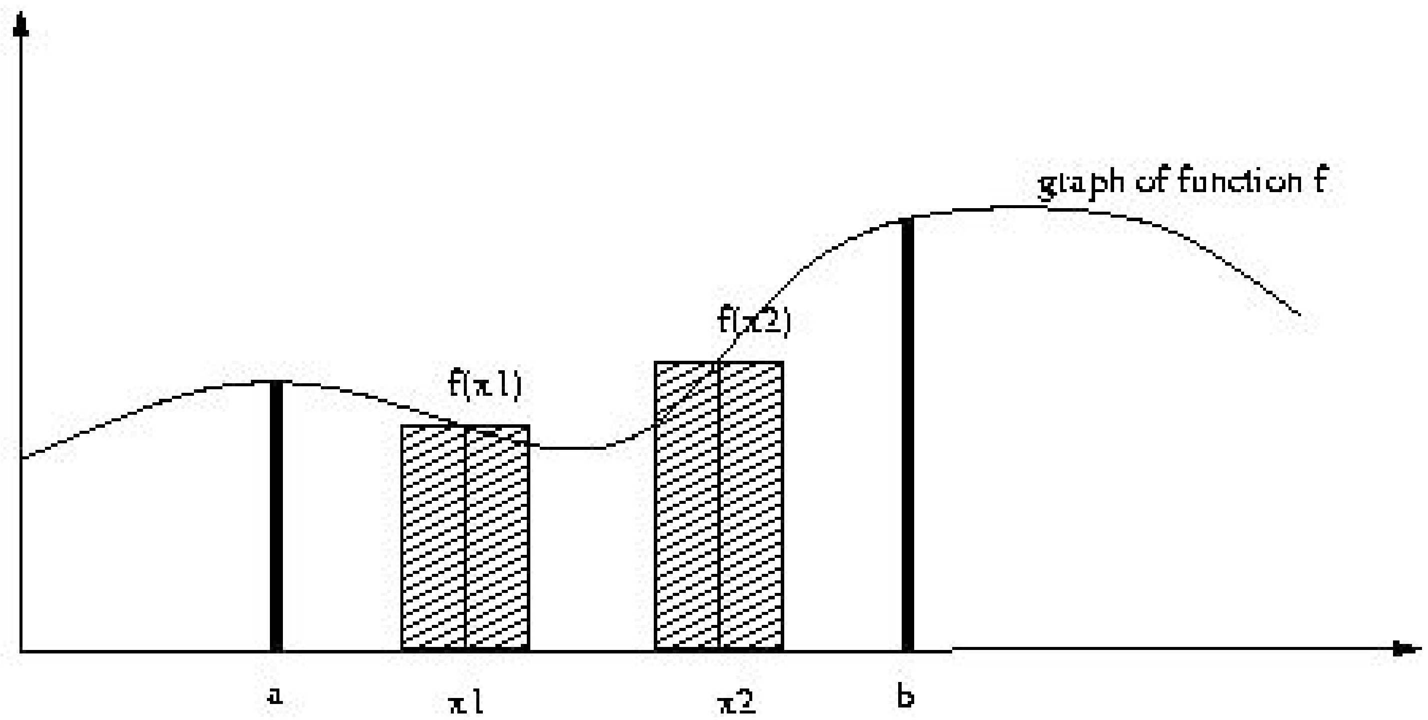
- Arithmetic Series
  - 1, 1, 1, 1, 1, ...
  - 1, 2, 3, 4, 5, ...
  - 0, 4, 8, 12, 16, ...
- Geometric Series
  - 1, 1, 1, 1, 1, ...
  - 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, ...
  - 1, .5, .25, .125, ...

Integration

- Basically: Area under a function between two points
- Extremely important in modeling physical systems
  - Even today, still challenging from the computing point of view

```
integrate:  (R -> R) R R -> R
```

Numerical Integration

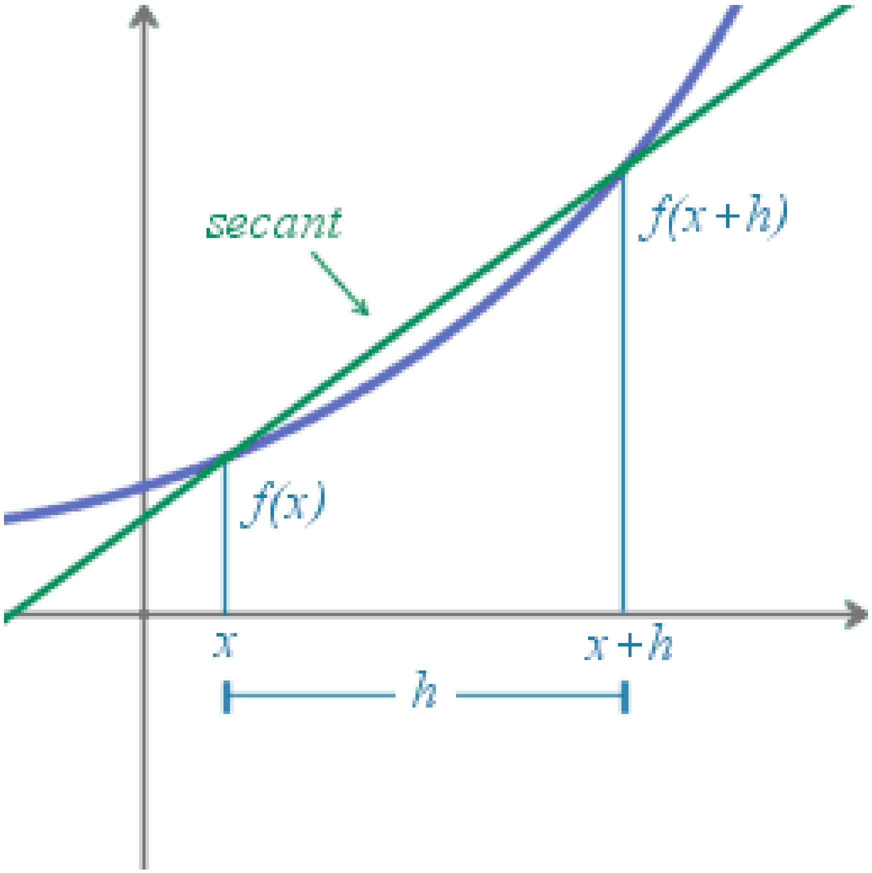


Differentiation

- Basically: Slope of a function at some point
- Similarly important (and challenging)
  - Fundamental relation to integration
    - (Notice, we see a lot of "g(f(x))=x" equations!)

```
diff:  (R -> R) R -> R
```

Numerical Differentiation



For Next Class

- Homework due next Friday
- Reading:
  - Ch 24: Defining functions on the fly
- Quiz on reading