CS221 Section 1

Foundations

Roadmap

Matrix Calculus

Python

Complexity

Recurrence Relations

Probability Theory

Notation and Basic Properties

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

$$\text{"}\mathbf{v}^2\text{"} = \|\mathbf{v}\|_2^2 = \mathbf{v} \cdot \mathbf{v} = \mathbf{v}^T\mathbf{v}$$

$$(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$$

$$(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$$

Matrix Calculus

Compute $\nabla_{\mathbf{w}} f(\mathbf{w})$

$$f(\mathbf{w}) = \mathbf{a} \cdot \mathbf{w} + b \|\mathbf{w}\|_2^2 + \mathbf{w}^\top C \mathbf{w}$$

Matrix Calculus

Compute $\nabla_{\mathbf{w}} f(\mathbf{w})$

$$f(\mathbf{w}) = \mathbf{a} \cdot \mathbf{w} + b \|\mathbf{w}\|_2^2 + \mathbf{w}^\top C \mathbf{w}$$

Observe that:

$$\nabla_{\mathbf{w}} \mathbf{a} \cdot \mathbf{w} = \mathbf{a}$$

$$\nabla_{\mathbf{w}} \|\mathbf{w}\|_2^2 = \nabla_{\mathbf{w}} \mathbf{w} \cdot \mathbf{w} = 2\mathbf{w}$$

$$\nabla_{\mathbf{w}} \mathbf{w}^\top C \mathbf{w} = (C + C^\top) \mathbf{w}$$

Matrix Calculus

example: Find the gradient of $g : \mathbf{R}^m \to \mathbf{R}$,

$$g(y) = \log \sum_{i=1}^{m} \exp(y_i).$$

Matrix Calculus

example: Find the gradient of $g : \mathbf{R}^m \to \mathbf{R}$,

$$g(y) = \log \sum_{i=1}^{m} \exp(y_i).$$

solution.

$$\nabla g(y) = \frac{1}{\sum_{i=1}^{m} \exp y_i} \begin{bmatrix} \exp y_1 \\ \vdots \\ \exp y_m \end{bmatrix}$$

Roadmap

Matrix Calculus

Python

Complexity

Recurrence Relations

Probability Theory

Syntactic Sugar

```
a = "Can I skip a CS221 homework? No dice!!"
```

The **split** command creates a list from a string with blank spaces as delimiters by default. You can also specify a different delimiter.

```
b = a.split()
b
```

```
['Can', 'I', 'skip', 'a', 'CS221', 'homework?', 'No', 'dice!!']
```

```
c = [len(_) for _ in b]
c
```

```
[3, 1, 4, 1, 5, 9, 2, 6]
```

Note that `_` is a valid variable name (usually for one-time use). An equivalent for-loop would be:

```
c = []
for _ in b:
    c.append(len(_))
c
```

```
[3, 1, 4, 1, 5, 9, 2, 6]
```

Syntactic Sugar

```
# Task: Find sum of X values less than 5

points = [(1, 2), (2, 6), (3, 3), (8, 9)]

X_total = 0
for point in points:
    if point[0] < 5:
        X_total = X_total + point[0]

X_total
```

```
6
```

```
sum([x for x, _ in points if x < 5])
```

```
6
```

enumerate is a useful built-in:

```
a = enumerate(['a', 'b', 'c'])

list(enumerate(['a', 'b', 'c']))
```

```
[(0, 'a'), (1, 'b'), (2, 'c')]
```

Syntactic Sugar

```
c = [len(_) for _ in b]

c
```

```
[3, 1, 4, 1, 5, 9, 2, 6]
```

```
c[2]    # The index starts from 0
```

```
4
```

```
c[-1]   # You can count backward from the right: c[-1] is the last element
```

```
6
```

```
c[1:4]  # Indexes are inclusive:exclusive
```

```
[1, 4, 1]
```

```
c[:4]   # = c[0:4]
```

```
[3, 1, 4, 1]
```

```
c[4:]   # = c[4:len(c)]
```

```
[5, 9, 2, 6]
```

```
c[:-1] # = c[0:len(c)-1], cutting out the last element in the list
```

```
[3, 1, 4, 1, 5, 9, 2]
```

References

- Official Documentation (has a tutorial):

  https://docs.python.org/

- Learn X in Y minutes:

  http://learnxinyminutes.com/docs/python/

- You don’t need to know numpy. But if you want to:

  http://nbviewer.ipython.org/gist/rpmuller/5920182

Roadmap

Matrix Calculus

Python

Complexity

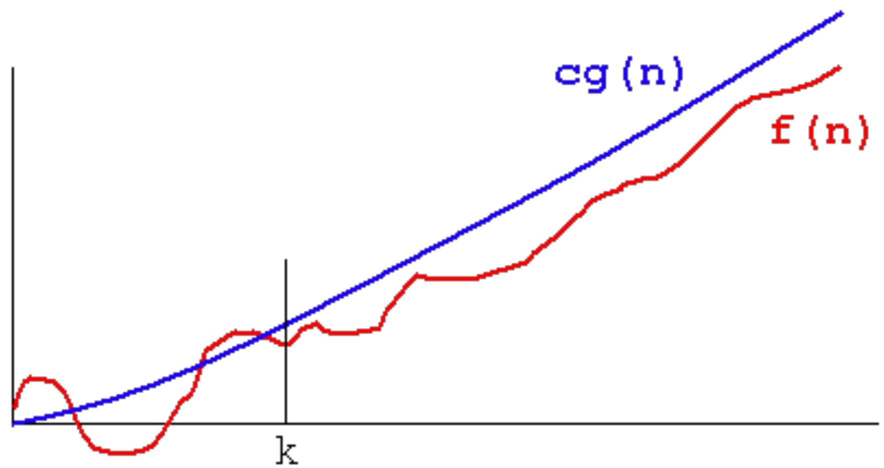
Recurrence Relations

Probability Theory

Time Complexity

Big O Notation:

We say f(n) is O(g(n)) if there exist positive constants C and K such that $0 \le f(n) \le Cg(n)$ for all $n \ge k$ (where C, K are constants)



Time Complexity

Examples:

```
max_val = -infinite
for i = 1 to n:
    max_val = max(arr[i], max_val)
```

```
for i = 1 to n - 1:
    for j = 1 to n - i:
        if arr[j] > arr[j+1]:
            swap(arr[j], arr[j+1])
```

Note that: $1 + 2 + \ldots + k = k(k + 1)/2$

Roadmap

Matrix Calculus

Python

Complexity

Recurrence Relations

Probability Theory

Coin Payment

Problem

Suppose you have an unlimited supply of coins with values 2, 3, and 5 cents

How many ways can you pay for an item costing 12 cents?
How about n cents?

Coin Payment

Recurrence Relation: Break down into smaller problems

Memoization: Remember what you already calculated

Roadmap

Matrix Calculus

Python

Complexity

Recurrence Relations

Probability Theory

Random Variables

Discrete:

$$\mathbb{P}(A = a) \qquad \text{or} \qquad p_A(a)$$

Continuous:

$$\cancel{\mathbb{P}(A = a)}$$

$$f_A(a)$$

$$\mathbb{P}(A \leq c) = \int_{--}^{c} f_A(a)\, da$$

Random Variables

	A = 0	A = 1	A = 2	A = 3
B = 0	0.1	0.25	0.1	0.05
B = 1	0.15	0	0.15	0.2

- What is $\mathbb{P}(A = 2)$
- What is $\mathbb{P}(A = 2 \mid B = 1)$

Random Variables

Independence:

$$\forall a, b, \quad \mathbb{P}(A = a, B = b) = \mathbb{P}(A = a)\mathbb{P}(B = b)$$

$$\forall a, b, \quad f_{A,B}(a, b) = f_A(a) f_B(b)$$

Expectation:

$$\mathbb{E}[A] = \sum_a a\,\mathbb{P}[A = a]$$

$$\mathbb{E}[A] = \int a f_A(a)\,da$$

Random Variables

	$\mathbf{A = 0}$	$\mathbf{A = 1}$	$\mathbf{A = 2}$	$\mathbf{A = 3}$
$\mathbf{B = 0}$	0.1	0.25	0.1	0.05
$\mathbf{B = 1}$	0.15	0	0.15	0.2

- Are A and B independent?
- What are $\mathbb{E}[A]$, $\mathbb{E}[B]$, $\mathbb{E}[A + B]$

Linearity of Expectation: $\mathbb{E}[A + B] = \mathbb{E}[A] + \mathbb{E}[B]$

True even when A and B are dependent!

Hat Toss

Problem

Suppose *n* hatted people toss their hats into the air and pick up one hat at random

In expectation, how many people get their own hats back?

Hint: linearity of expectation

Coin Tossing

Problem

You are given a fair coin with sides heads and tails.

What is the expected number of flips until you get 3 heads in a row?

How about *n* heads in a row?

Questions?