STOCHASTIC THINKING

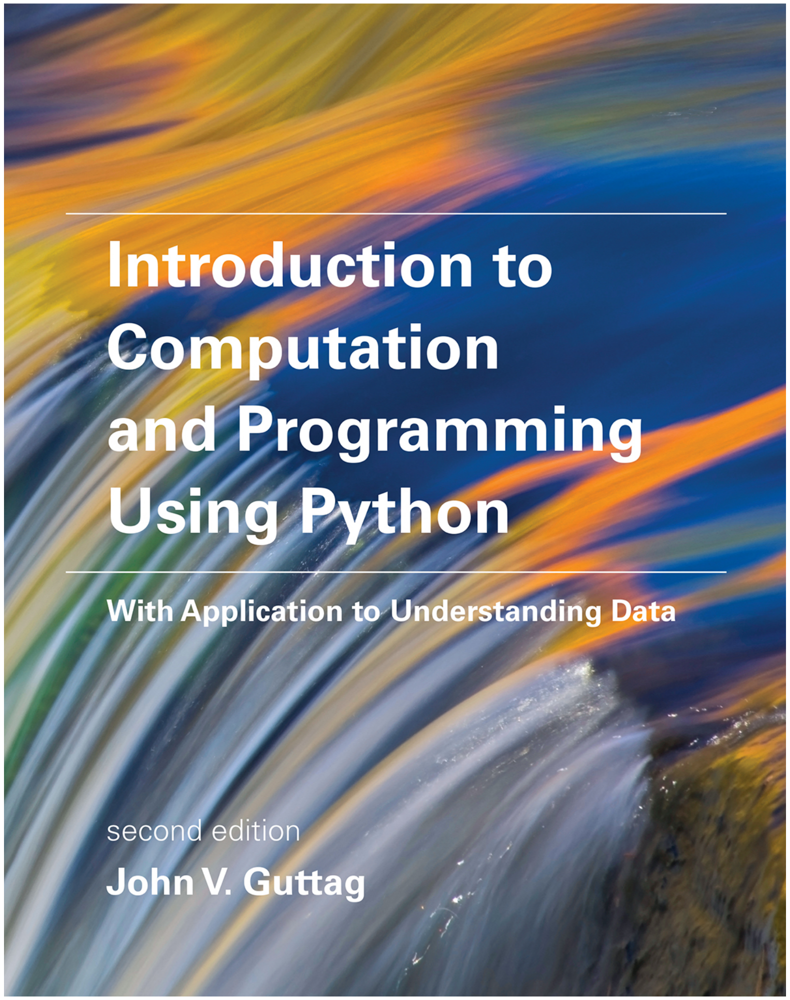
(download slides and .py files to follow along!)

Eric Grimson

MIT Department Of Electrical Engineering and Computer Science

Assigned Reading

- Today:
  - Sections 15.1-15.3
  - Section 15.5
- Next lecture:
  - Chapter 14



The Simple World of Newtonian Mechanics

- Want to build computational models of physical world
- Every effect has a cause, so physical world can be understood causally
  - e.g., Newton's three laws of motion
- 18th century mathematics
  - 18.01, 18.02
- 18th-19th century physics
  - 8.01

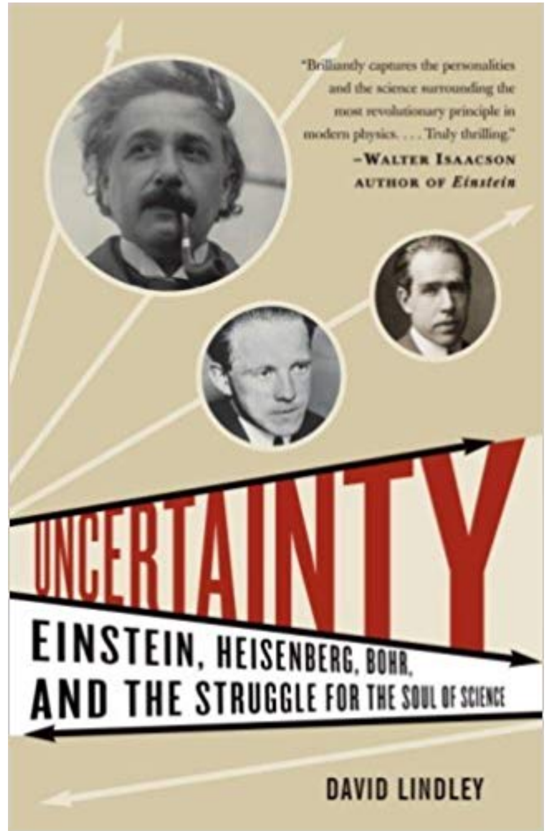
1643 – 1727

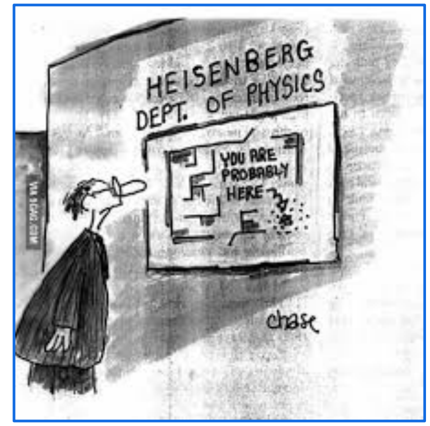
A side comment on Newton and campus closures:

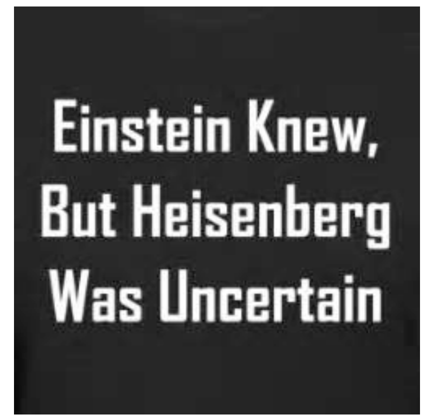
- Plague forced closure of Cambridge from 1665-1667
- Newton went home to family's country house (Woolsthorpe Manor)
- Poll: which, if any, of the following did he accomplish during that time?
  - Invented calculus
  - Derived first version of laws of gravity
  - Discovered foundations of optics of colors

Two+ Centuries Later

- In early 20th century, atomic and subatomic observations seemed to defy classical mechanical explanations
  - E.g., do electrons behave as particles or waves (or both)?
- Led to introduction of quantum mechanics, and assertion that one cannot precisely measure a particle's physical properties
- This had impact beyond physics – what are philosophical implications if one cannot measure things precisely or cannot be sure that an observed effect had a specific cause?





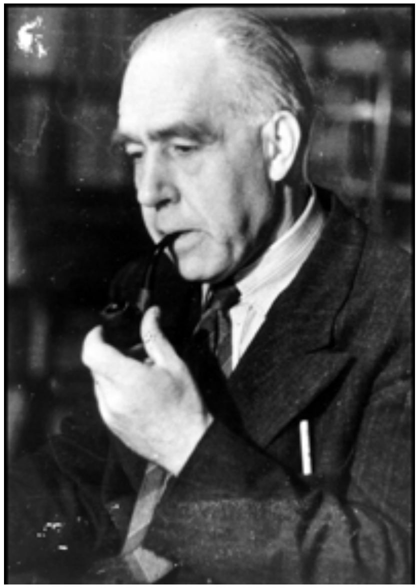


- 20th century physics
  - 8.04, 8.05, 8.06

Copenhagen Interpretation

- Heisenberg and Bohr argued that at its most fundamental level, the behavior of the physical world cannot be predicted
  - For example, cannot precisely measure position and momentum of a particle at the same time
- Fine to make statements of the form "x is highly likely to occur," but not of the form "x is certain to occur"

"The more precise the measurement of position, the more imprecise the measurement of momentum, and vice versa."



"Those who are not shocked when they first come across quantum theory cannot possibly have understood it. ...If you think you can talk about quantum theory without feeling dizzy, you haven't understood the first thing about it."

Many Were Indeed Shocked

- Einstein and Schrodinger objected
  - "I, at any rate, am convinced that He [God] does not throw dice."– Albert Einstein
  - Bohr, in response, said, "Einstein, don't tell God what to do."

Poll:
Heisenberg (1932), Bohr (1922), Einstein (1921) and Schrodinger (1933 - shared) all won the Nobel prize for Physics. Which one had a son that also won the Nobel prize for Physics?

One of 6 cases of two generations winning different Nobel prizes

DOES IT REALLY MATTER?

- Suppose we flip a coin twice
- Could we correctly predict whether the flips would yield
  - 2 heads?
  - 2 tails?
  - 1 head and 1 tail?
- Need to know accurately:
  - weight distribution of coin
  - velocity and acceleration of thumb
  - orientation of coin on thumb before flip
  - air flow around coin
  - height above landing spot
  - other effects

The Moral

- The world may or may not be inherently unpredictable
- But lack of knowledge prevents precise predictions
- Therefore, treat the world as inherently unpredictable

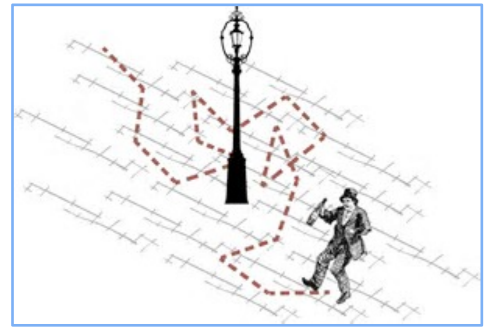
Causal nondeterminism – some events truly random

Predictive nondeterminism – in principle might be able to predict, but don’t have enough information. There is chaos, but not randomness

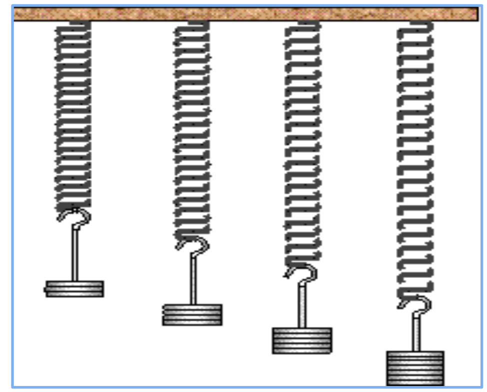
Use a **stochastic** process to represent systems or phenomena that seem to change in a random way

Why Should We Care about Stochastics?

- model systems where have choices in state transitions
  - e.g., a drunk wandering in a field, with random probability of taking a step in any direction at any point in time; motion of particles in a fluid



- model systems where measurement is uncertain or noisy
  - e.g., physical model of a spring stretching; epidemiology studies connecting mortality to environmental factors (e.g., smoking)



- model systems where can't measure entire system
  - e.g., polling voter preference before election; estimating public support for initiatives (e.g., mitigating climate change)

- draw **statistical conclusions** about outcomes
  - how likely is an outcome to occur, and how confident are we in that estimate?
  - do for systems that are not completely random, but are not exactly predicted by model, and for random systems

Stochastic Processes

- In general, a system often can be defined by a set of state variables, and processes that determine the transition to a next set of values for those variables
  - Causal processes
- In a **stochastic** system, the process for determining next state might depend both on the previous states of the process **and on some random element**
  - Predictive nondeterminism
- This will require a change in how we think about computational models of processes

Two Specifications for a Process

```
def rollDie():
    """returns a int between 1 and 6"""

def rollDie():
    """returns a random int between 1 and 6"""
```

Any implementation that satisfies the second specification would also satisfy the first.

But one that satisfies the first specification might or might not satisfy the second

```
def rollDie():
    """returns an int between 1 and 6"""
    return 3

import random

def rollDie():
    """returns a random int between 1 and 6"""
    return random.choice([1,2,3,4,5,6])
```

Trying It (simulate rolling 5 dice)

```
def rollDie():
    """returns a random int between 1 and 6"""
    return random.choice([1,2,3,4,5,6])

def testRoll(n):
    result = ''
    for i in range(n):
        result = result + str(rollDie())
    return result

for i in range(10):
    print(testRoll(5))
```

Poll: Is this an example of predictive or causal non-determinism?

How probable is the output 11111?

Probability is About Counting

- Count the total number of possible events (often called the universe of events)
- Count the number of events that have the property of interest
- Divide second number by the first
- Probability of rolling 11111?
  - All events: 11111, 11112, 11113, …, 11121, 11122, …, 66666
  - Ratio: 1/(6**5)
  - ~0.0001286
- Probability of 12345?

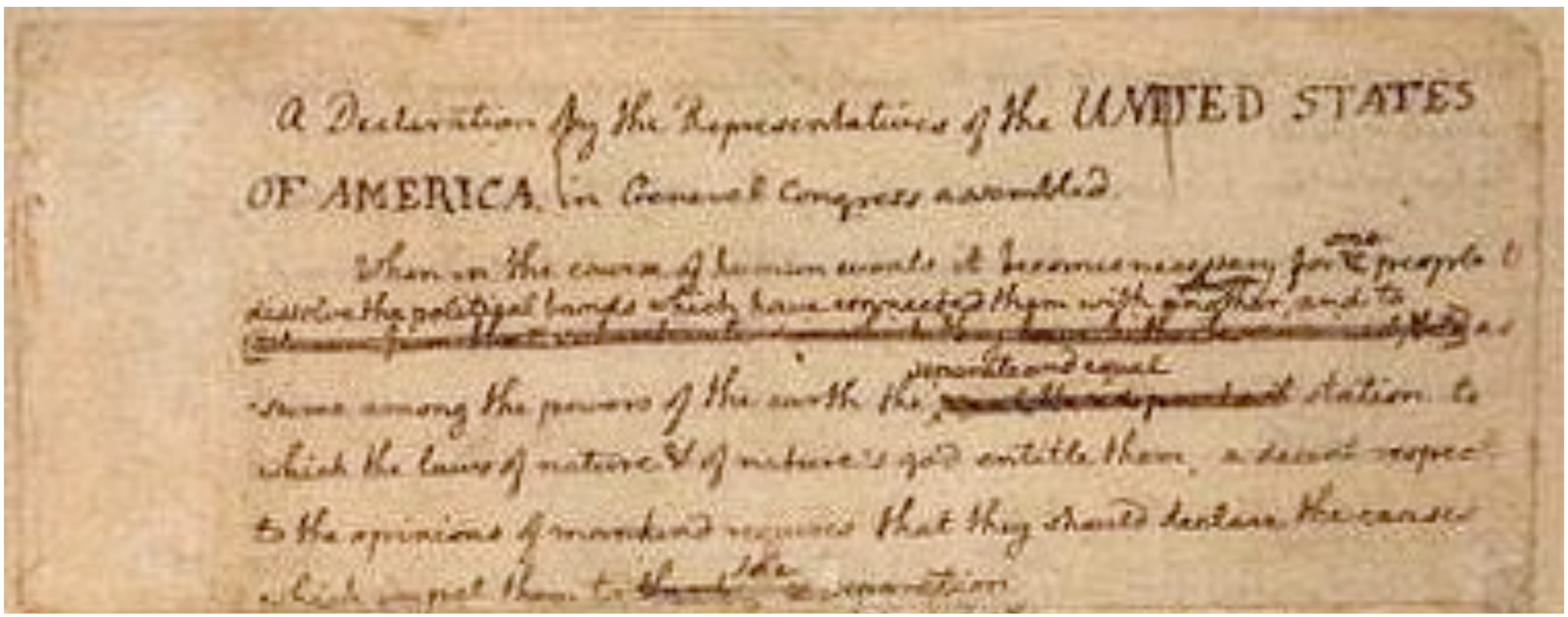
Same probability

Some Basic Facts about Probability

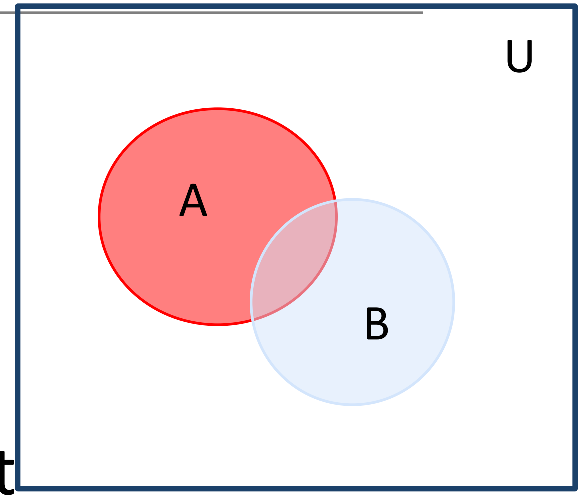
- Probabilities are always in the range **0 to 1**. 0 if impossible, and 1 if guaranteed
- If the probability of an event occurring is **p**, the probability of it not occurring must be **1-p**
- When events are **independent** of each other, the probability of all of the events occurring is equal to the **product** of the probabilities of each of the events occurring

Independence

- Two events are **independent** if the outcome of one event has no influence on the outcome of the other
- Independence should not be taken for granted
  - Is your score on the micro quiz after today's lecture independent of how well you did on the first Pset?



Some Probability Algebra



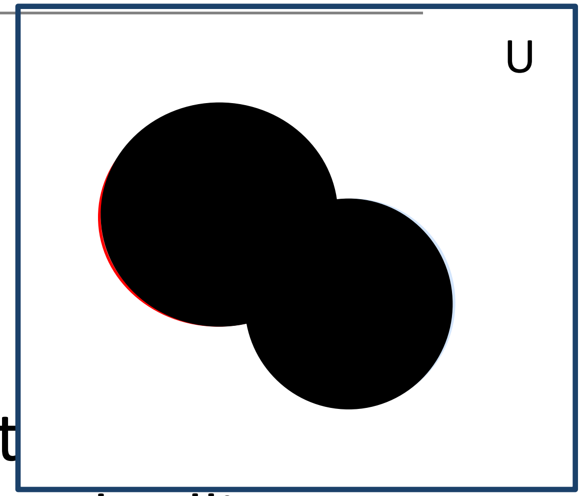
- Given a universe U of all possible events
- $P(A) = \dfrac{\text{number of events in A}}{\text{number of events in U}}$
- P(A and B) = P(A)P(B), if A and B independent
  - What is probability of flipping a head on a coin and rolling a 5 on a die?
  - P(head) = 1/2
  - P(roll 5) = 1/6
  - Product is 1/12

H1, H2, H3, H4, H5, H6
T1, T2, T3, T4, T5, T6

- P(A or B) ?

P(A) + P(B) ?

Some Probability Algebra



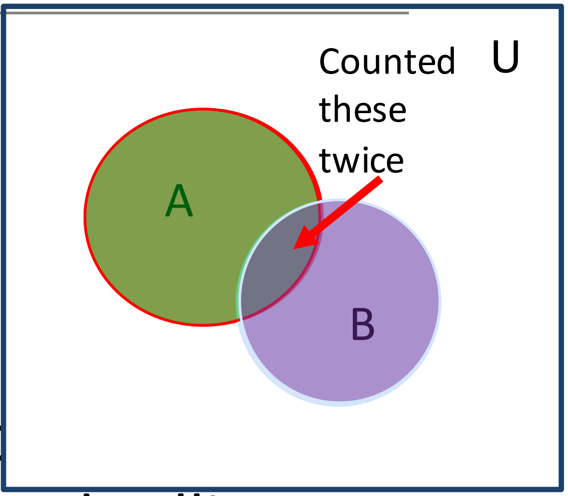
- Given a universe U of all possible events
- P(A) = $\frac{\text{number of events in A}}{\text{number of events in U}}$
- P(A and B) = P(A)P(B), if A and B independent
  - What is probability of flipping a head on a coin and rolling a 5 on a die?
  - P(head) = 1/2
  - P(roll 5) = 1/6
  - Product is 1/12
- P(A or B) ?
  - 1 – P(¬A and ¬B)
  - 1 – P(¬A)P(¬B)

H1, H2, H3, H4, H5, H6
T1, T2, T3, T4, T5, T6

P(head or rolling 5)?

1 - (1/2 * 5/6) = 7/12

Some Probability Algebra



- Given a universe U of all possible events
- P(A) = $\frac{\text{number of events in A}}{\text{number of events in U}}$
- P(A and B) = P(A)P(B), if A and B independent
  - What is probability of flipping a head on a coin and rolling a 5 on a die?
  - P(head) = 1/2
  - P(roll 5) = 1/6
  - Product is 1/12
- P(A or B) ?
  - 1 – P(¬A and ¬B)
  - P(A) + P(B) – P(A and B)
  - P(A) + P(B) – P(A)P(B)

H1, H2, H3, H4, $\boxed{\text{H5}}$, H6
T1, T2, T3, T4, $\boxed{\text{T5}}$, T6

P(head or rolling 5)?

1 - (1/2 * 5/6) = 7/12

1/2 + 1/6 – 1/12 = 7/12

Independent vs. Dependent Probabilities

- Deal two cards at random from a standard deck of 52 cards
- What is the probability that the first card is a king and the second a king?
- With replacement
  - 4/52 * 4/52
  - These are independent events
- Without replacement?
  - Really asking what is the probability that the second card is a king, given that the first card was a king
  - 4/52 * 3/51
  - These are **not** independent events

Conditional Probabilities

- probability of A, given B occurred
- $P(A|B) = \dfrac{P(A \text{ and } B)}{P(B)}$

- example – let D_1 be value of roll of first die
  and – let D_2 be value of roll of second die
- what is $P(D_1 = 2 \mid D_1 + D_2 <= 5)$?

Conditional Probabilities

- what is $P(D_1 = 2 \mid D_1 + D_2 <= 5)$?
- $P(D_1 = 2) = 6/36 = 1/6$
- $P(D_1 + D_2 <=5) = 10/36$
- $P(D_1 = 2 | D_1 + D_2 <=5)$
  $= 3/10$

	1	2	3	4	5	6
1						
2						
3						
4						
5						
6						

But also

- $$\frac{P(D_1 = 2 \text{ and } D_1 + D_2 <=5)}{P(D_1 + D_2 <=5)}$$

$$= \frac{3/36}{10/36}$$

$$= 3/10$$

Note that we cannot use

$P(D_1 = 2 \text{ and } D_1 + D_2 <=5) = P(D_1)P(D_1 + D_2 <=5)$

because events are **not** independent

Using Simulation to Estimate Probabilities

- Going to explore simulation of rolling a group of dice many times
- First of many simulations we will see
- Talk more carefully about simulation in future lectures; especially when dealing with problems that do not have analytic solutions
- **Simulation**
  - Run many trials in which we select one event from the universe of possible events
  - For each trial, compute some properties of event
  - Report some statistics about the properties over the set of trials

A Simulation of Die Rolling

```
def runSim(goal, numTrials):
    total = 0
    for i in range(numTrials):
        if i != 0 and i%1000 == 0:
            print('Starting trial', i)
        result = ''
        for j in range(len(goal)):
            result += str(rollDie())
        if result == goal:
            total += 1
    print('Actual probability of', goal, '=',
          round(1/(6**len(goal)), 8))
    estProbability = round(total/numTrials, 8)
    print('Estimated Probability of', goal, '=',
          round(estProbability, 8))

runSim('11111', 1000)
```

Roll appropriate # of dice

Count # of hits

Return observed probability

Output of Simulation

- Actual probability = 0.0001286
- Estimated probability = 0.0
- Actual probability comes directly from math, but estimated probability?
- Why did simulation give me the **wrong** answer?
  - 6**5 = 7776 possibilities, so 1000 trials unlikely to observe one event in that universe with enough frequency to estimate accurately
  - Even if observed once, estimate would then be 0.001

Let's try 1,000,000 trials

Morals

Aesop
620 BC – 564 BC

- Moral 1: it takes a lot of trials to get a good estimate of the frequency of occurrence of a **rare** event. We'll talk lots more in later lectures about how to **know** when we have enough trials to trust the estimate
- Moral 2: one should not confuse the **sample probability** with the actual probability
- Moral 3: there was really no need to do this by simulation, since there is a perfectly good closed form answer. We will see many examples where this is not true, where only simulation can provide answers about processes and outcomes

The Birthday Problem

- What is the probability of at least two people in a group having the same birthday?
- If there are 30 people in a room, should you be surprised if two share a birthday?

Poll:
What do you think are the odds of two people, out of a group of 30, sharing a birthdate?

The Birthday Problem

- What if there are 367 people in the group?
  - Use the "pigeonhole principle" – if you try to put m items into n containers and m > n, then at least one container must have more than 1 item
- What about fewer then 366 people in the group?

The Birthday Problem

- If assume each birthdate equally likely, then can use "inclusion/exclusion principle"

- Probability that n (<367) people each have different birthday (but no twins allowed!)

$$\frac{366}{366} * \frac{365}{366} * \frac{364}{366} * \cdots * \frac{366-n+1}{366} = \frac{366!}{366^n * (366-n)!}$$

- Probability that at least two of n people have same birthday:

$$1 - \frac{366!}{366^n * (366-n)!}$$

For n = 23 this is about .5063

For n = 30 this is about .7053

- Without assumption of equal likelihood, **VERY** complicated

The Birthday Problem


```
import math

def trueProb(numPeople):
    #assumes each birth date equally probable
    numerator = math.factorial(366)
    denom = (366**numPeople)*math.factorial(366-numPeople)
    return 1 - numerator/denom

for i in (2, 4, 8, 16, 32, 64, 128, 256):
    print('Probability of a shared birthday with', i,
          'people =', round(trueProb(i), 8))
```

Results

```
Probability of a shared birthday with 2 people = 0.0027
Probability of a shared birthday with 4 people = 0.0163
Probability of a shared birthday with 8 people = 0.0741
Probability of a shared birthday with 16 people = 0.2829
Probability of a shared birthday with 32 people = 0.7524
Probability of a shared birthday with 64 people = 0.9971
Probability of a shared birthday with 128 people = 1.0000
Probability of a shared birthday with 256 people = 1.0000
```

Approximation Using a Simulation

```
def sameDate(numPeople, numSame, possibleDates):
    birthdays = [0]*366
    for p in range(numPeople):
        birthDate = random.choice(possibleDates)
        birthdays[birthDate] += 1
    return max(birthdays) >= numSame
```

Create array of 0's, one for each possible day

Count hits for each day

Is day with most hits big enough?

One trial

```
def birthdayProb(numPeople, numSame, numTrials,
                 possibleDates):
    numHits = 0
    for t in range(numTrials):
        if t%100000 == 0:
            print('Starting trial', t)
        if sameDate(numPeople, numSame, possibleDates):
            numHits += 1
    return numHits/numTrials
```

Set of trials

```
possibleDates = range(366)
for i in (2, 4, 8, 16, 32, 64, 128, 256):
    print('Est. probability of a shared birthday with', i,
          'people =', round(birthdayProb(i, 2, 1000, possibleDates), 8))
```

Results

```
Probability of a shared birthday with 2 people = 0.0027
Probability of a shared birthday with 4 people = 0.0163
Probability of a shared birthday with 8 people = 0.0741
Probability of a shared birthday with 16 people = 0.2829
Probability of a shared birthday with 32 people = 0.7524
Probability of a shared birthday with 64 people = 0.9971
Probability of a shared birthday with 128 people = 1.0000
Probability of a shared birthday with 256 people = 1.0000
```

```
Est. probability of a shared birthday with 2 people = 0.0030
Est. probability of a shared birthday with 4 people = 0.0190
Est. probability of a shared birthday with 8 people = 0.0730
Est. probability of a shared birthday with 16 people = 0.3020
Est. probability of a shared birthday with 32 people = 0.7070
Est. probability of a shared birthday with 64 people = 0.9960
Est. probability of a shared birthday with 128 people = 1.0000
Est. probability of a shared birthday with 256 people = 1.0000
```

Why 3 Is Much Harder Analytically

- Suppose we want the probability of 3 people sharing a birthday, out of some group of size N?
- For 2 people, the complementary problem is “all birthdays are distinct”
- For 3 people, the complementary problem is a complicated disjunction of cases, and hard to describe analytically

- But using the simulation is **dead trivial!!**

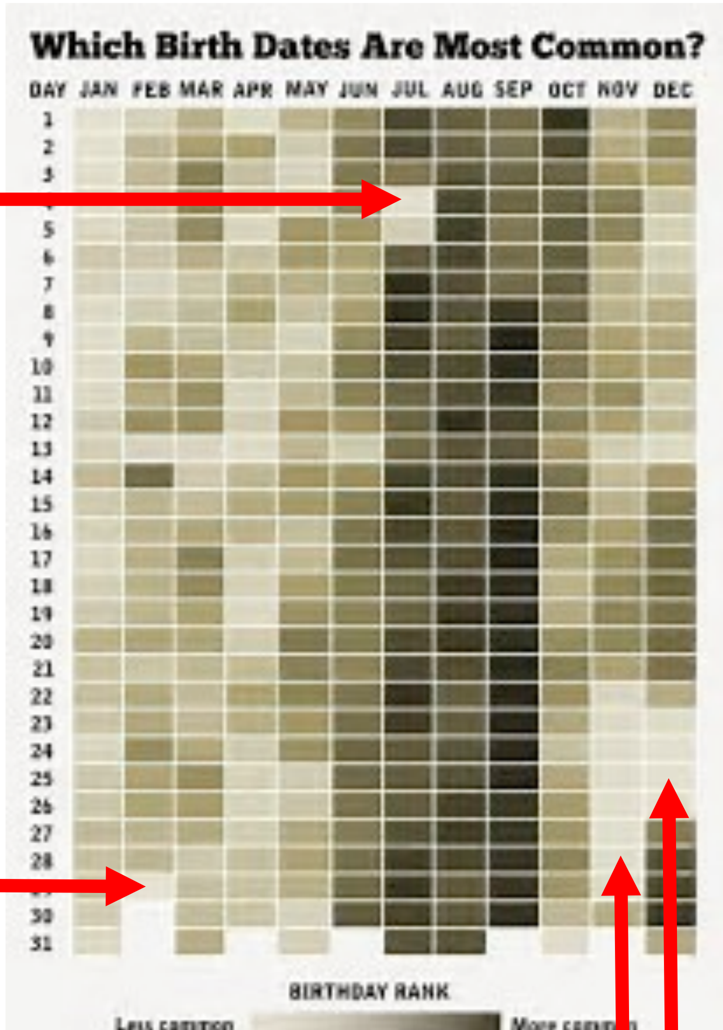
Results

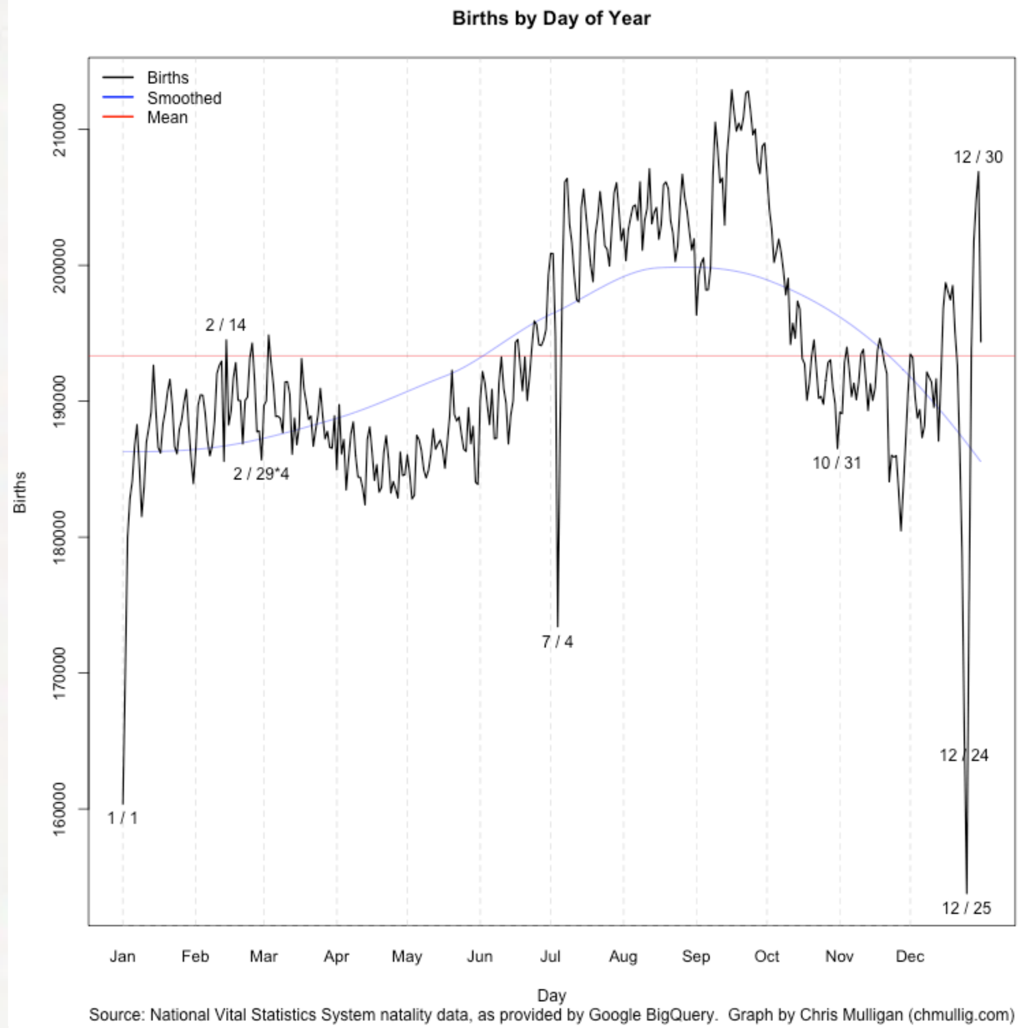
```
Est. probability of a shared birthday with 2 people = 0.0030
Est. probability of a shared birthday with 4 people = 0.0190
Est. probability of a shared birthday with 8 people = 0.0730
Est. probability of a shared birthday with 16 people = 0.3020
Est. probability of a shared birthday with 32 people = 0.7070
Est. probability of a shared birthday with 64 people = 0.9960
Est. probability of a shared birthday with 128 people = 1.0000
Est. probability of a shared birthday with 256 people = 1.0000
```

```
Est. probability of three sharing a birthday among 2 people = 0.0000
Est. probability of three sharing a birthday among 4 people = 0.0000
Est. probability of three sharing a birthday among 8 people = 0.0000
Est. probability of three sharing a birthday among 16 people = 0.0020
Est. probability of three sharing a birthday among 32 people = 0.0400
Est. probability of three sharing a birthday among 64 people = 0.2530
Est. probability of three sharing a birthday among 128 people = 0.8820
Est. probability of three sharing a birthday among 256 people = 1.0000
```

Note how a simulation is answering a question we cannot answer analytically

But All Dates Are NOT Equally Likely

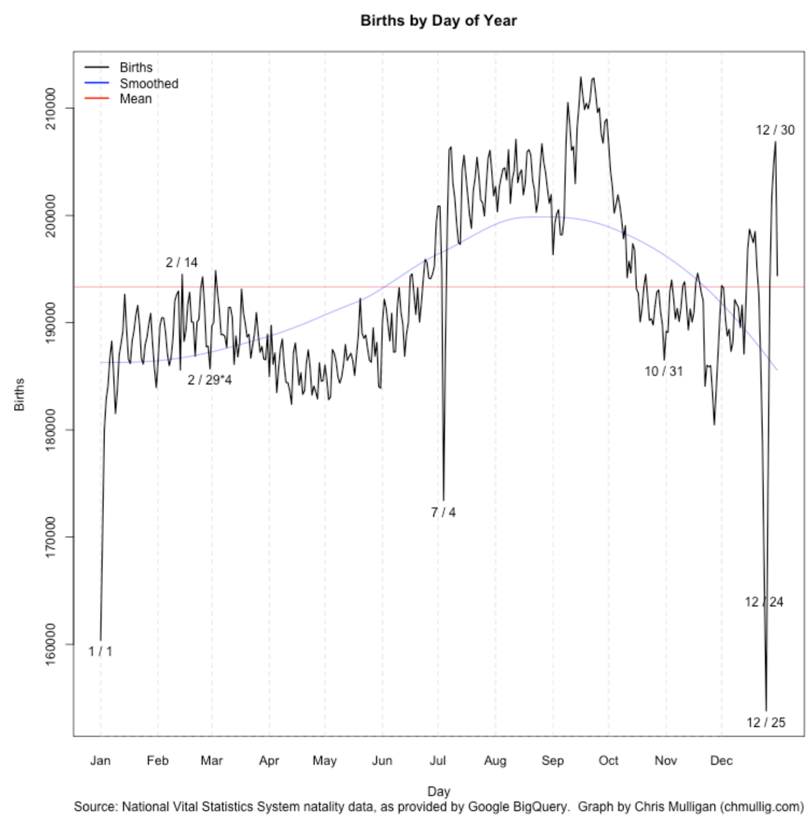




Poll: why are some birthdays less common?

How Does this Affect Probabilities?

- Do you expect a big change?
- Again, adjusting analytic model is a pain
- Adjusting simulation model easy



Poll:
Given this distribution of birthdays, do you expect a big change in probability of sharing a birthdate?

Excerpt of Data

month,day,births
1,1,160369
1,2,169896
1,3,180036
1,4,182854
1,5,184145
1,6,186726
1,7,188277
1,8,185186
1,9,181511
1,10,183668

Approximating Using a Simulation

```
def getBdays(toPlot = False):
    inFile = open('Births.csv')
    inFile.readline() #discard first line
    numBirths = [int(line.split(',')[2][:-1])\
                 for line in inFile]
    possibleDates = []
    for i in range(len(numBirths)):
        possibleDates += [i]*(numBirths[i])
    if toPlot:
        d = {i+1:numBirths[i] for i in range(len(numBirths))}
        vals = [d[k] for k in d.keys()]
        plt.plot(vals, 'bo')
        plt.xlim(-10, plt.xlim()[1])
        plt.xlabel('Day of Year')
        plt.ylabel('Number of Births')
        mean = 'Mean = ' + str(int(sum(vals)/len(vals)))
        std = 'Std = ' + str(int(numpy.std(vals)))
        plt.title('Frequency of Birthdates\n' +
                  mean + ', ' + std)
    return possibleDates
```

List of ints, each is number of births on a particular day

For each day (indexed from start of year) create numBirths copies of that index

160,369 0's, then
168,896 1's, then
180,036 2's, then…

Very long list, but picking an element from it at random will reflect actual probability of a birth on that day

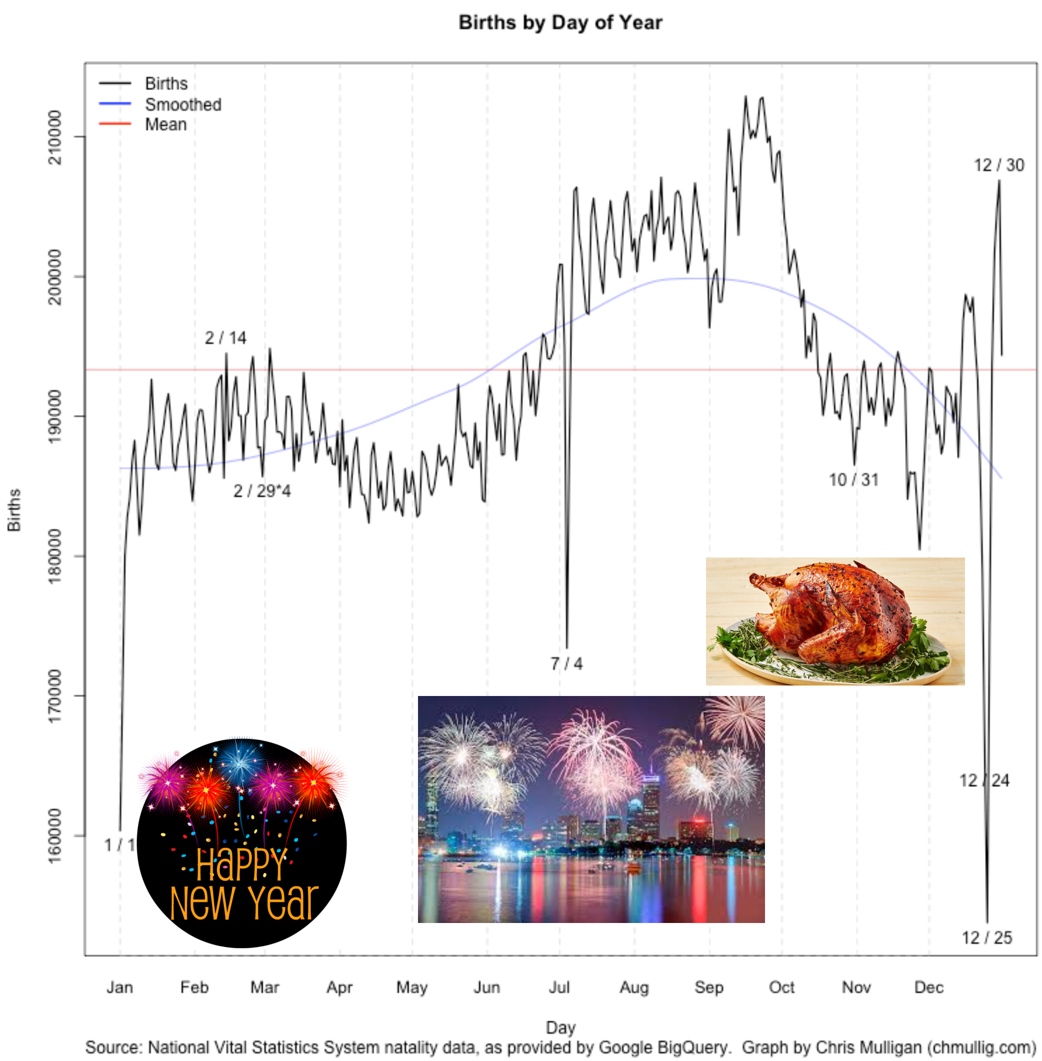
Approximating Using a Simulation

```
possibleDates = getBdays(False)
numShared = 3
for numPeople in [8, 16, 32, 64, 128, 256]:
    print('For', numPeople, 'probability of',
          numShared, 'shared birthdays:')
    print('  Estimated for actual distribution:',
          birthdayProb(numPeople, numShared, 10000,
                       possibleDates))
    print('  Estimated for uniform distribution:',
          birthdayProb(numPeople, numShared, 10000,
                       range(366)))
```

Results

```
For 16 probability of 3 shared birthdays:
  Estimated for actual distribution:  0.0038
  Estimated for uniform distribution: 0.0026
For 32 probability of 3 shared birthdays:
  Estimated for actual distribution:  0.035
  Estimated for uniform distribution: 0.0358
For 64 probability of 3 shared birthdays:
  Estimated for actual distribution:  0.2432
  Estimated for uniform distribution: 0.2533
For 128 probability of 3 shared birthdays:
  Estimated for actual distribution:  0.8839
  Estimated for uniform distribution: 0.8822
For 256 probability of 3 shared birthdays:
  Estimated for actual distribution:  1.0
  Estimated for uniform distribution: 1.0
```

Looking at Results



Given apparent variance

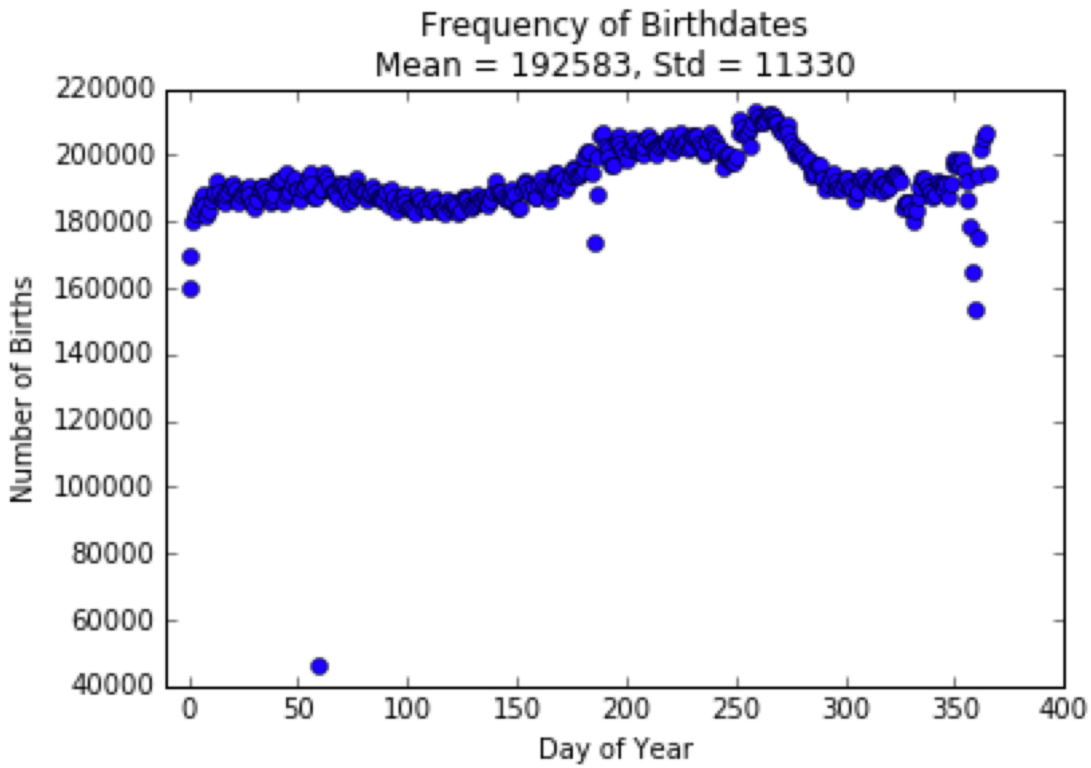
Shouldn't difference between actual and uniform distribution be larger?

Or at least in consistent direction?

Numbers on y-axis hard to read

Note y axis scaling

A Better Plot



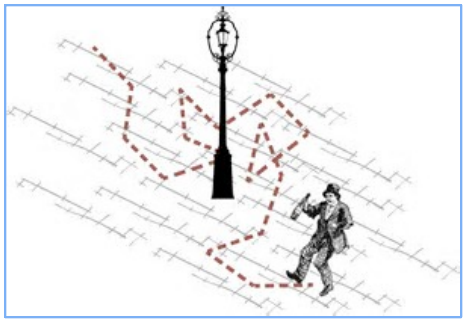
- Effect small
  - Frequency is pretty similar with a few exceptions
- Not a large enough sample to see effect consistently
- Much more on this later in the term

2M Trials

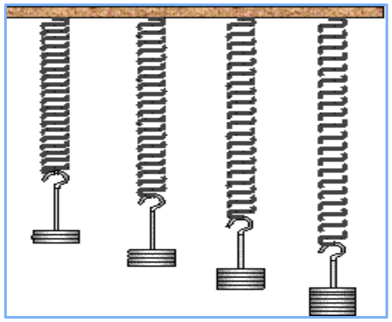
```
For 16 probability of 3 shared birthdays:
  Estimated for actual distribution: 0.00416
  Estimated for uniform distribution: 0.0040215
For 32 probability of 3 shared birthdays:
  Estimated for actual distribution: 0.0348315
  Estimated for uniform distribution: 0.0341425
For 64 probability of 3 shared birthdays:
  Estimated for actual distribution: 0.246033
  Estimated for uniform distribution: 0.243804
For 128 probability of 3 shared birthdays:
  Estimated for actual distribution: 0.8808115
  Estimated for uniform distribution: 0.879002
For 256 probability of 3 shared birthdays:
  Estimated for actual distribution: 1.0
  Estimated for uniform distribution: 0.9999995
```

Can Simulation Address Real Problems?

- You might conclude that simulation is only useful to thinking about toy problems, like flipping coins, rolling die, or looking at party games like shared birthdays
- In the coming lectures we are going to use these simulation methods to look at real world problems



Model systems where have choices in state transitions – random walks



Model systems where measurement is uncertain or noisy – linear regression

Model systems where can't measure entire system – Central Limit Theorem

An Aside on Hashing

- Earlier in 6.0001 we used dicts as a very efficient way to store and retrieve data
- Given that keys are arbitrary, how does Python efficiently retrieve a value associated with a key?
- Uses a hash table:
  - Convert a key to an integer within a particular range (called a hash function) – range would be size of dict
    - Typically mapping a large range of values to a smaller one
  - Use integer to index into a list
  - Store or retrieve value at that spot in the list
  - Indexing is constant time, so if hash function is fast to compute, method is very efficient – which is why dict's are efficient data structures

An Aside on Hashing

- Can generalize the idea to both store and retrieve data from a list using arbitrary keys
- Ideal hash function:
  - Has a range from 1 to n
  - Produces a uniform distribution of keys, i.e., probability that any key is mapped to any integer is 1/n
- If we do K insertions into a hash table with n slots, what is chance of a collision of two keys mapping to the same index?
- This is just the birthday problem!!

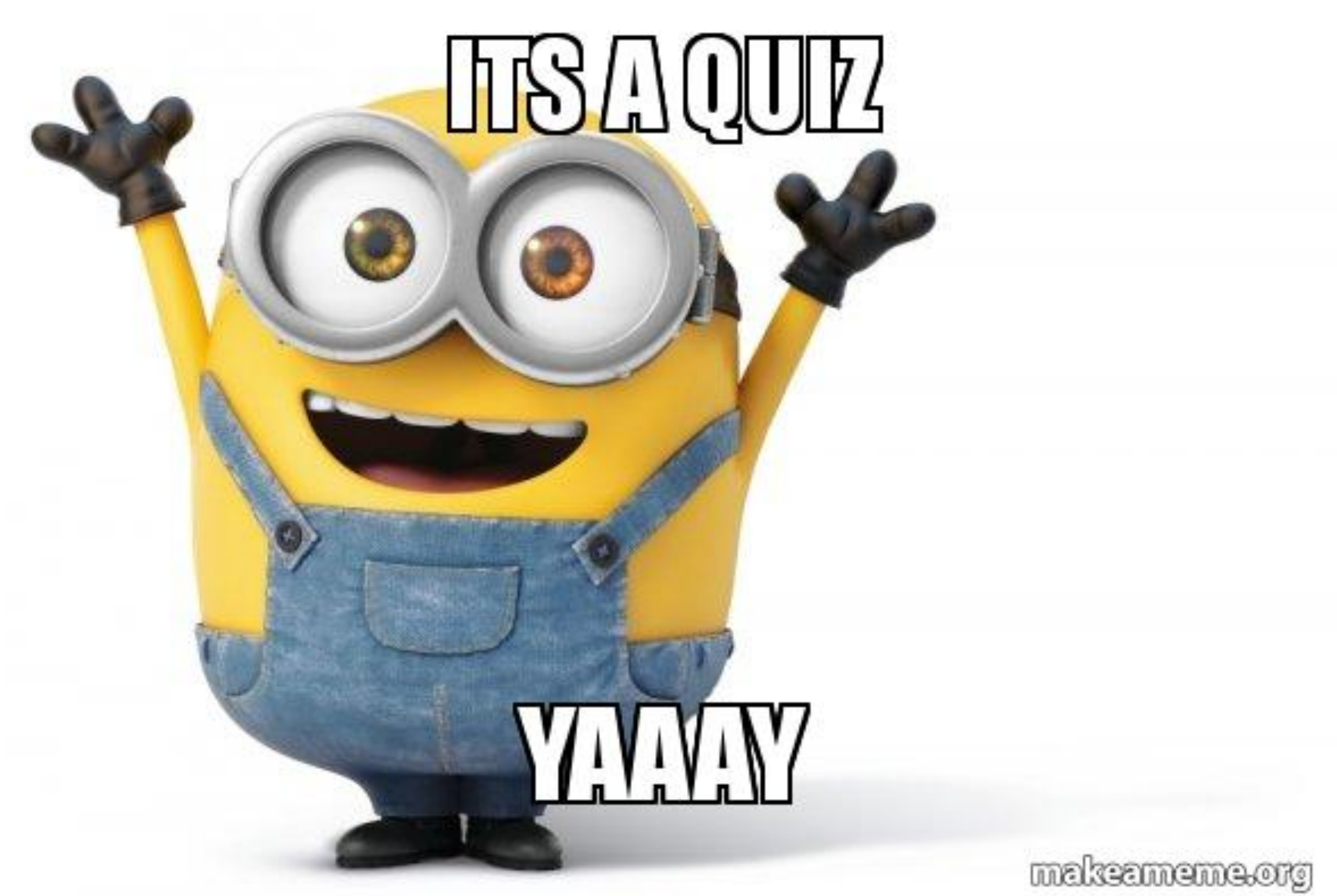
Stochastics Useful for Measuring Event Probability

Can stochastics help us answer questions like: how likely is this to happen?

Given that this occurred on **March 14, 2017**, we might want to be careful about assumptions of independent randomly positioned players – as opposed to some non-random effect

Summary

- As far as we can tell, the world is stochastic
- Therefore models need to estimate probabilities
- Analytic models feasible for reasonably simple situations
- For more complex situations, simulations may be best (or only) option
- Simulate stochastic process by:
    - Defining an event (e.g., rolling a die N times, flipping a coin N times, randomly picking birthdays for N people),
    - Running some number of trials,
    - Estimating probability of observing particular event (e.g., rolling 11111, finding three people with shared birthday)
- Much much **much** more on simulation models to come


ITS A QUIZ
YAAAY
makeameme.org

Microquiz Logistics

- The quiz will be available to start at 15:55 EDT, on days noted in the calendar, and due within 12 hours. Once you start, you have 30 minutes to complete it. Be sure you have your IDE open and your allowed materials ready.
- You may use class notes, slides, and code files. You may **NOT** look up anything on the Internet. You may **NOT** discuss the quiz with anyone for the 12 hours that the quiz is available.
- Once you click on the 'Start Quiz' button, your 30 minute time starts, and cannot be stopped.
- For additional details, see the course Stellar site:

http://stellar.mit.edu/S/course/6/sp20/6.0002/