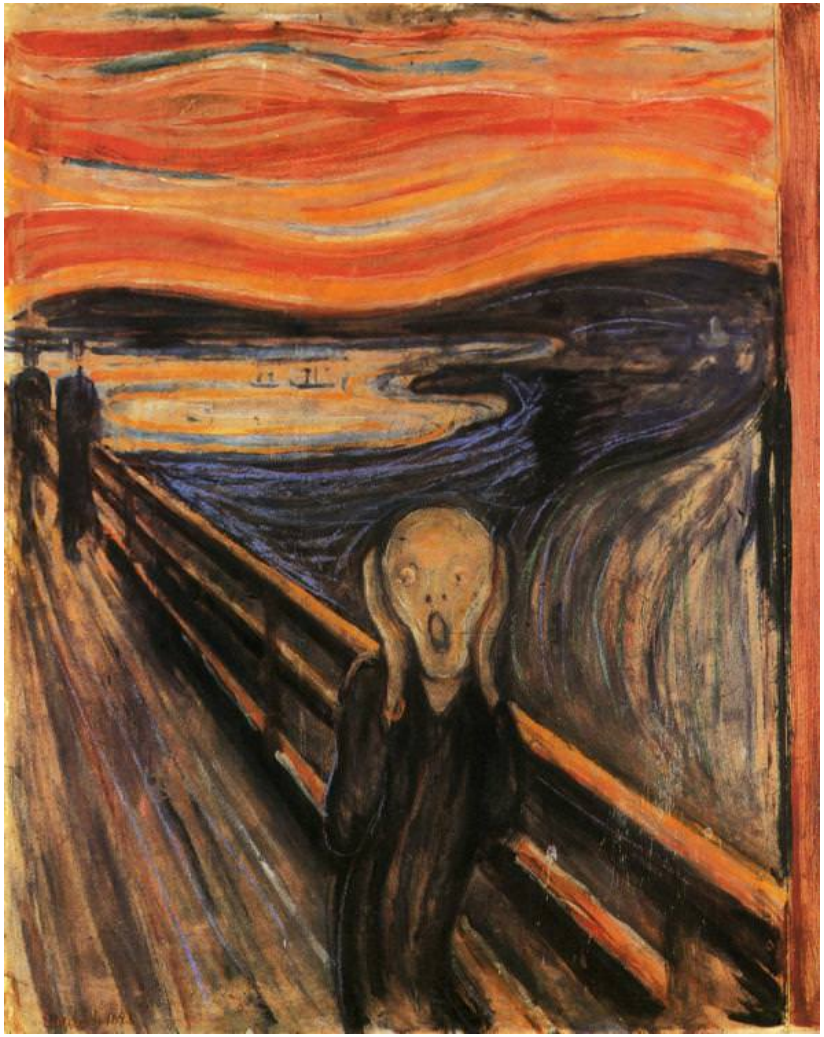
Happy Halloween



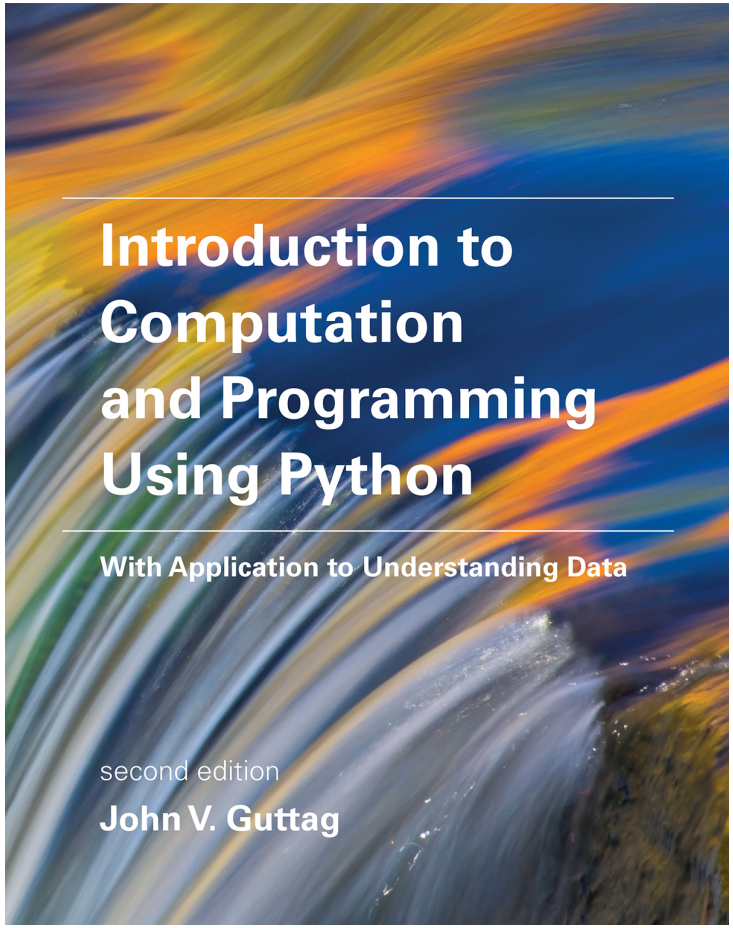
STOCHASTIC THINKING

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

6.0002 LECTURE 4

Assigned Reading

- Today:
  - Sections 15.1-15.3
  - Section 15.5
- Monday:
  - Chapter 14



https://mitpress.mit.edu/sites/default/files/Guttag_errata_revised_083117.pdf

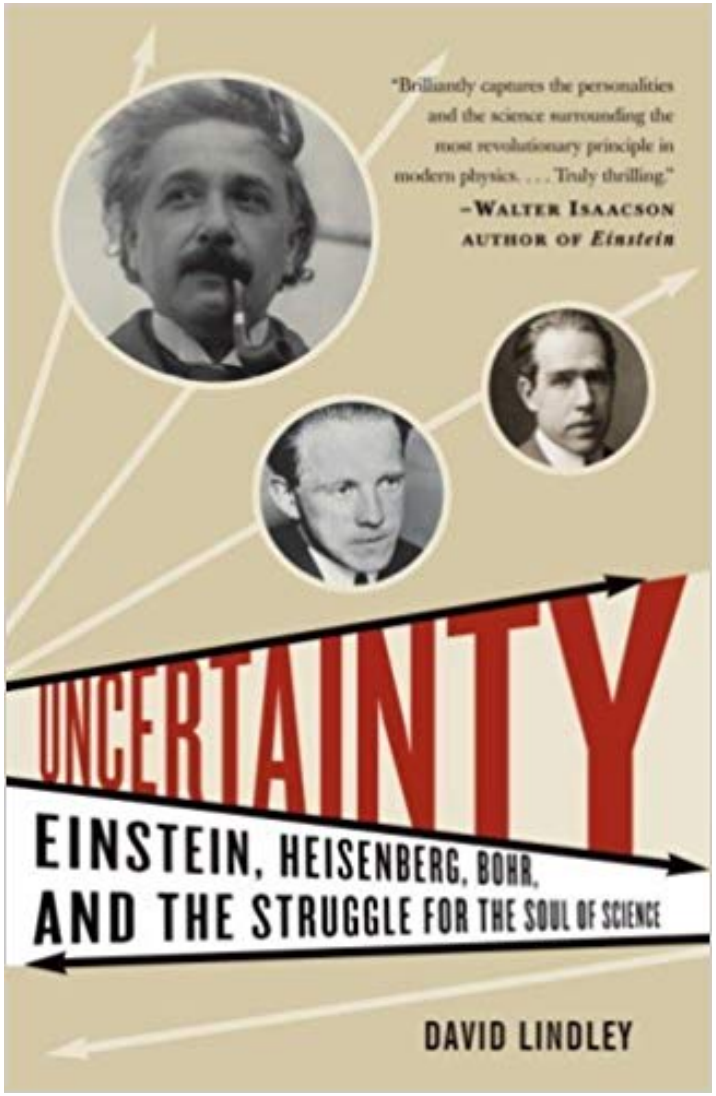
The Simple World of Newtonian Mechanics

- Every effect has a cause
- The physical world can be understood causally

- 18th century mathematics
  - 18.01, 18.02
- 19th century physics
  - 8.01

1643 – 1727

Two Centuries Later



Copenhagen Doctrine

- Bohr and Heisenberg 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”

"Anyone who is not shocked by Quantum Theory has not understood it."

Many Were Shocked

- Einstein and Schrodinger objected
  - “God does not play dice with the universe” – Albert Einstein

DOES IT REALLY MATTER?

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

The Moral

- The world may or may not be inherently unpredictable
- But lack of knowledge doesn't allow accurate predictions
- Therefore, we might as well 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

Why Should We Care about Stochastics?

- Want to model systems where have choices in state transitions
  - Where will a molecule move when a fluid is heated?
- Want to model systems where measurement is uncertain
  - How tall are you?
- Want to model systems where can't measure entire system
  - Will proposition 3 pass?
- Etc.

Stochastic Processes

- In general, a system defined by a set of state variables, and processes that determine the next set of values
- In a stochastic system, the process for determining next state might depend on both the previous states **and some random element**

Two Specifications

```
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])
```

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

Trying It

```
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))
```

Predictive or causal non-determinism?

How probable is the output 11111?

Probability is About Counting

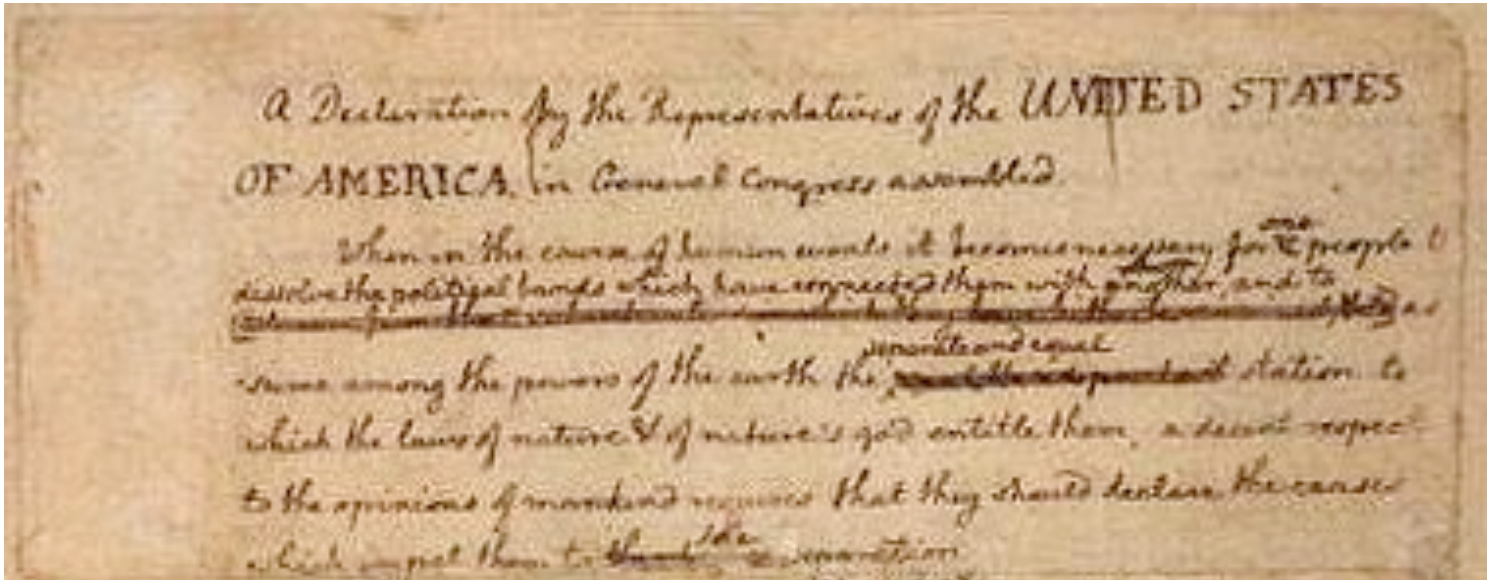
- Count the number of possible events
- Count the number of events that have the property of interest
- Divide second by the first
- Probability of 11111?
  - All events: 11111, 11112, 11113, …, 11121, 11122, …, 66666
  - Ratio: 1/(6**5)
  - ~0.0001286
- Probability of 12345?

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



Some Probability Algebra

- P(A) = $\frac{\text{number of events in A}}{\text{number of events in U}}$
- P(A and B) = P(A)P(B) if independent
  - What is probability of flipping a head and rolling a 5 on a die?
  - P(head) = ½
  - 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

- P(A) = $\frac{\text{number of events in A}}{\text{number of events in U}}$
- P(A and B) = P(A)P(B) if independent
  - What is probability of flipping a head and rolling a 5 on a die?
  - P(head) = ½
  - 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) ?
  - 1 – P(¬A and ¬B)

Independent vs. Dependent Probabilities

- Deal two cards from a standard deck of 52 cards
- What is the probability that the first card is a king and the second a queen?
  - 4/52 * 4/52?
  - 4/52 * 4/51?
  - Something else?
- With replacement
  - 4/52 * 4/52
- Without replacement?

Using Simulation to Estimate Probabilities

- First of many simulations we will see
- Talk more carefully about simulation in future lectures
- Simulation
  - Run many trials in which we select one value from the universe of possible values
  - For each trial, compute some properties of value
  - 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%100000 == 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))
```

Output of Simulation

- Actual probability = 0.0001286
- Estimated probability = 0.0
- How did I **know** that this is what would get printed?
  - actual probability comes directly from math, but estimated probability?
- Why did simulation give me the **wrong** answer?
  - 6**5 is a big universe (7776 events), so 1000 trials unlikely to observe one event in that universe with enough frequency to estimate
    - Even if I observed once, estimate would then be 0.001

Let's try 1,000,000 trials

Morals

- 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

The Birthday Problem

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

The Birthday Problem

- What if there are 367 people in the group?
  - Use the “pigeonhole principle”
- What about fewer people?

The Birthday Problem

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

- Probability that n (<367) people each have different birthday

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

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

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

Don't worry about why it's this formula

- 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))
```

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

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

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))
```

One trial

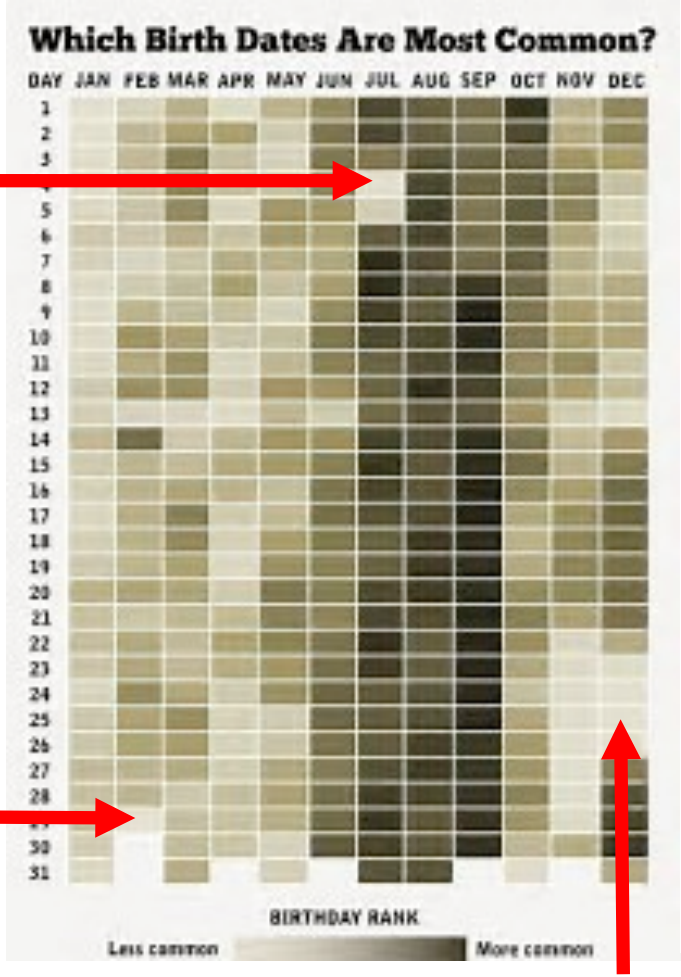
Set of trials

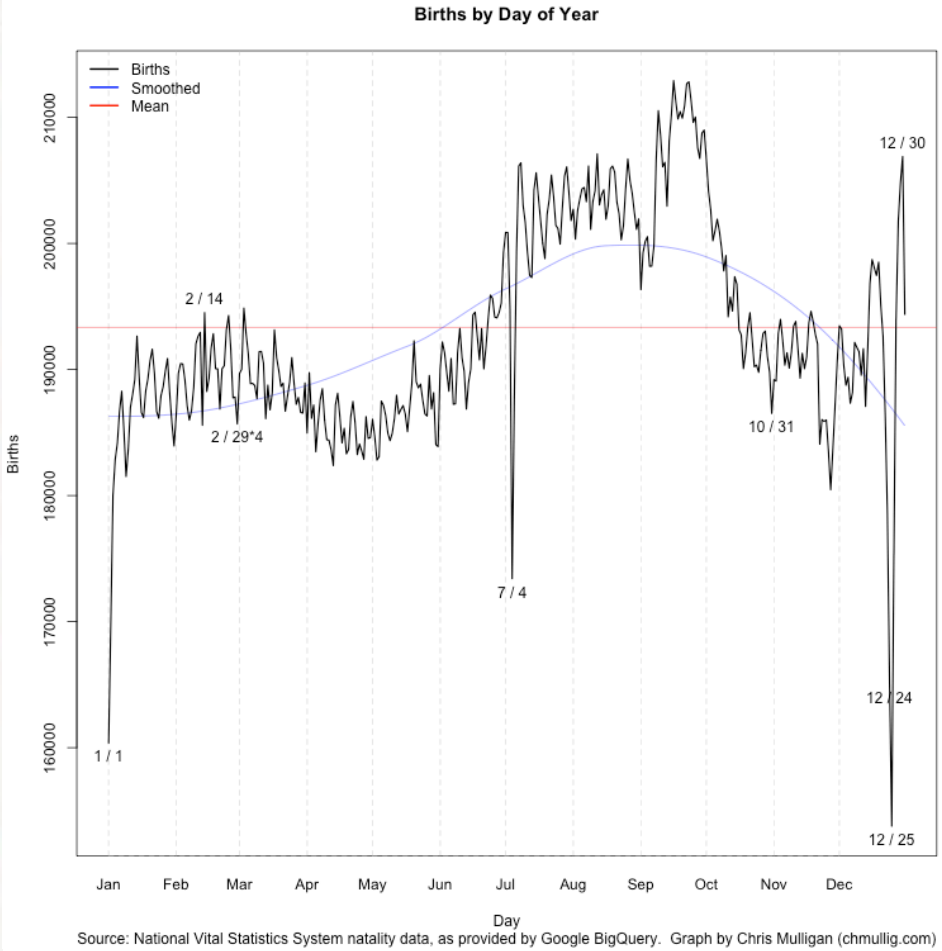
Suppose we want the probability of 3 people sharing a birthday?

Why 3 Is Much Harder Analytically

- For 2 people, the complementary problem is “all birthdays are distinct”
- For 3 people, the complementary problem is a complicated disjunction
  - All birthdays distinct or
  - One pair and the rest distinct or
  - Two pairs and the rest distinct or
  - …
- But using the simulation is dead trivial!!

But All Dates Are NOT Equally Likely





Excerpts of Data

	A	B	C
1	month	day	births
2	1	1	160369
3	1	2	169896
4	1	3	180036
5	1	4	182854
6	1	5	184145
7	1	6	186726
8	1	7	188277
9	1	8	185186
10	1	9	181511
11	1	10	183668

52	2	20	190051
53	2	21	186860
54	2	22	190061
55	2	23	190298
56	2	24	193160
57	2	25	194281
58	2	26	191623
59	2	27	187750
60	2	28	187812
61	2	29	46420

How Does this Affect Probabilities?

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

Approximating Using a Simulation

```
def getBdays(toPlot = False):
    inFile = open('Births.csv')
    inFile.readline() #discard first line
    numBirths = []
    for l in inFile:
        line = l.split(',')
        numBirths.append(int(line[2][:-1]))
    possibleDates = []
    d = {}
    for i in range(len(numBirths)):
        possibleDates += [i]*(numBirths[i])
        d[i+1] = numBirths[i]
    if toPlot:
        #some plotting code
    return possibleDates
```

Create a very long list

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)))
```

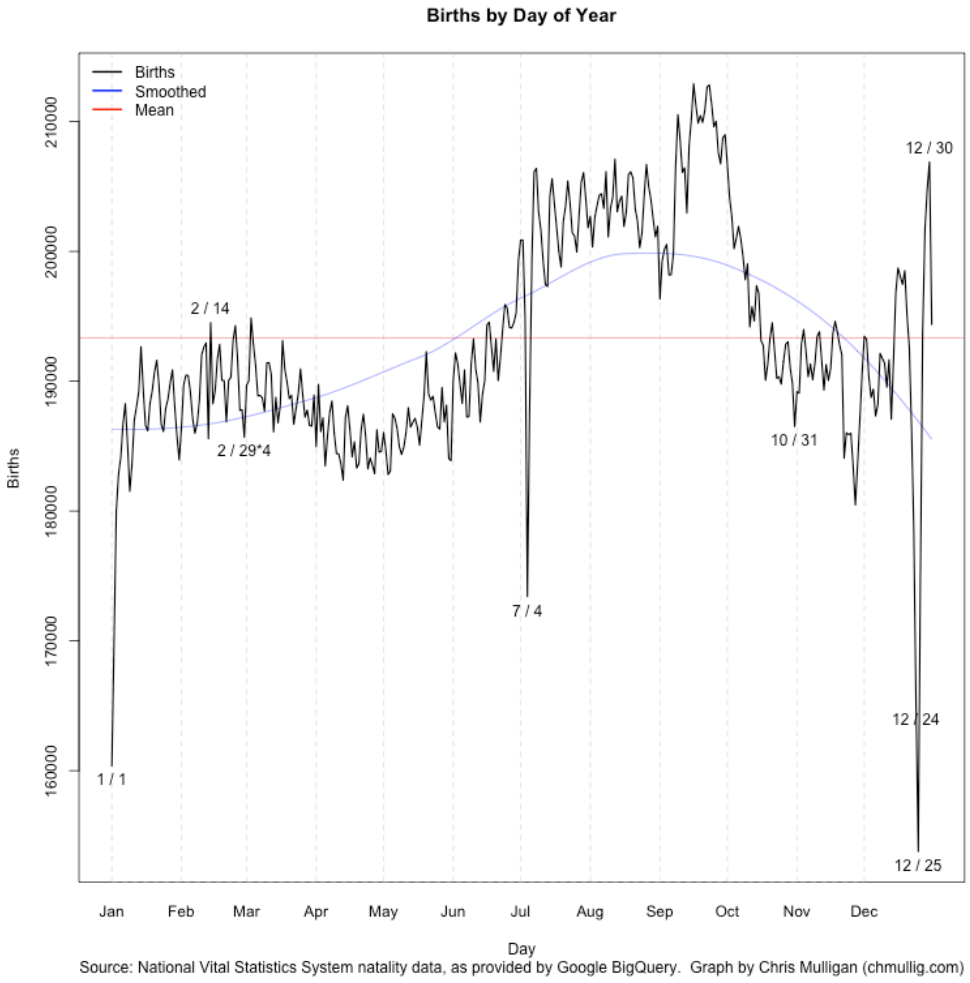
Looking at Results

Given apparent variance

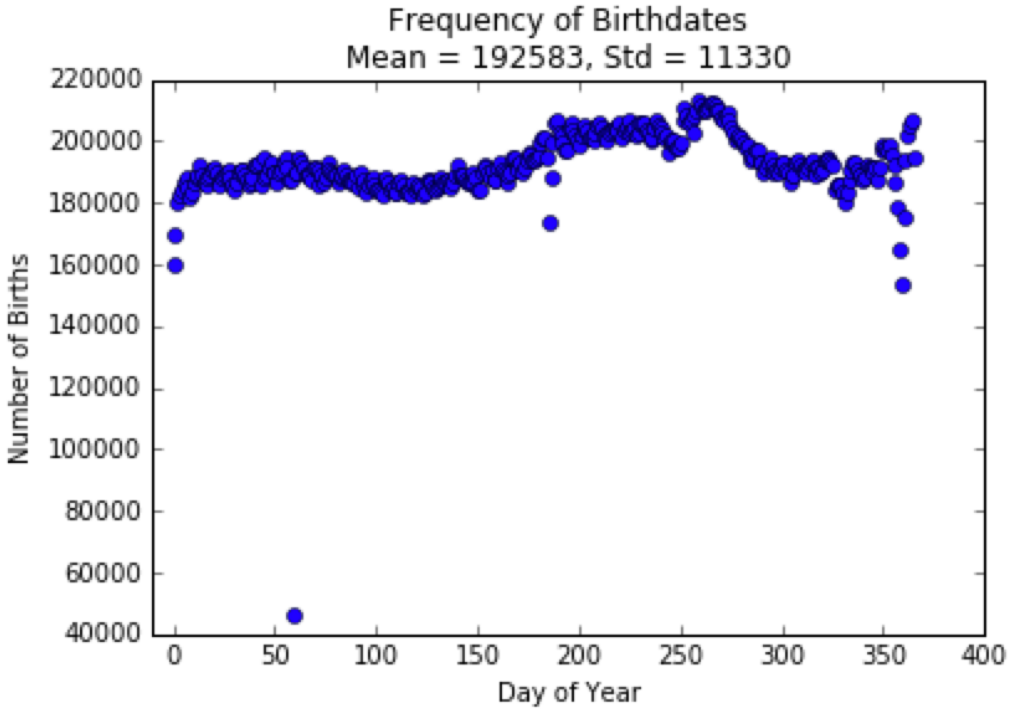
- Shouldn't differences be bigger
- Or at least in same direction

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 it consistently
- Much more on this later in the term

2M Trials

```
For 8 probability of 3 shared birthdays:
  Estimated for actual distribution: 0.0003855
  Estimated for uniform distribution: 0.000433
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
```

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 often better
- 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, and
  - Estimating probability of observing particular event (e.g., rolling 11111, finding two people with shared birthday)
- Much more on simulation models to come


QUIZ TIME

Note on Quiz

- No programming
- ONLY thing on your screen should be quiz
  - **No IDE**
- Don’t forget to check out