Simulation

Matthew Hefner

December 5, 2018

Page 490 #5 Part A

```
import math
def testrand_M(s, n):
  count = [0 for i in range(10)]
  k = 16807
  j = 2147483647
  for i in range(n):
    s = k * s % j
    x = s / j
    count[math.floor(x * 10)] += 1
  print(count)
for i in range(1, 11):
  testrand_M(i, 10000)
```

```
## [993, 1007, 998, 958, 1001, 1049, 989, 963, 1026, 1016]
## [1023, 1019, 992, 1004, 979, 982, 980, 1004, 970, 1047]
## [969, 995, 971, 965, 1056, 1034, 1009, 996, 1000, 1005]
## [1032, 973, 987, 1012, 1000, 996, 992, 982, 998, 1028]
## [1027, 960, 1018, 995, 1007, 981, 959, 986, 1062, 1005]
## [1008, 995, 1040, 964, 954, 1013, 1022, 943, 1008, 1053]
## [1014, 978, 976, 975, 983, 1035, 1061, 953, 1010, 1015]
## [1012, 1016, 992, 973, 968, 1001, 997, 1013, 1002, 1026]
## [949, 969, 976, 1062, 997, 1016, 975, 991, 1034, 1031]
## [983, 1025, 961, 958, 1017, 987, 1024, 1033, 1028, 984]
```

Looks like reasonable randomness to me!

Page 496 #5

```
import random
import math
def f(x, y, z):
  return y * x * x + z * math.log(y) + math.exp(x)
def integral(n):
  integral = 0.0
  for i in range(n):
    x = random.uniform(-1, 1)
    y = random.uniform(3, 6)
    z = random.uniform(0, 2)
    integral += f(x, y, z)
  regionSize = (2) * (3) * (2)
  print("Estimate:", integral * regionSize / n)
  print("Rough error estimate:", 1 / math.sqrt(n))
integral(100)
```

```
## Estimate: 49.27710380723866
## Rough error estimate: 0.1
```

```
integral(1000)
```

```
## Estimate: 49.946157321216
## Rough error estimate: 0.03162277660168379
```

```
integral(10000)
```

```
## Estimate: 50.17198593455242
## Rough error estimate: 0.01
```

Maples symbolic solution is $24ln(2) + 12ln(3) + 6 + 6e - 6e^{-1}$; 49.921294121181621205 at 20 decimal point accuracy. Only the last estimate is within our rough error estimate, but the estimates are generally fairly close.

Page 505 #17

```
import random
import math
def drunk(n):
  count = 0
  for trial in range(n):
    x = 0
    y = 0
    for i in range(50):
      step = random.uniform(0, 1)
      if step < 1 / 6:
        x += 1
      elif step < .5:
        x -= 1
      elif step <.75:
        y +=1
      else:
        y -= 1
    if math.sqrt(x ** 2 + y ** 2) > 20:
      count += 1
  print(count / n)
drunk(10000)
```

```
## 0.0115
```

It looks like he winds up over 20 feet away slightly over 1% of the time.

$\sum_{i=1}^{n}$