Simulation Step 2 Generate Delivery Data

June 30, 2024

[1]:
```
import matplotlib.pyplot as plt
import random
```

1 Generating Delivery Data

[2]:
```
def generateDeliveries(p, C, days, seed=0):
    ## p is the average number of parcels per day per customer
    ## C is the number of customers to be served
    ## days is the number of days for which data are to be generated.
    random.seed(seed)
    deliveries = [ [ ] for _ in range(days) ]
    for c in range(C):
        arr = 0
        while True:
            arr += random.expovariate(p)
            day = int(arr)
            if day>=days:
                break
            deliveries[day].append(c)
    return deliveries
```

2 Demo Example

[3]:
```
D = generateDeliveries(0.2, 10, 5, seed=42)
D
```

[3]:
```
[[1, 4, 5, 8], [1, 5], [1], [4, 4, 6], [4, 7]]
```

Over 5 days, a total of 12 parcels are to be delivered:

- On the first day deliver one parcel to each of customers 1, 4, 5, and 8.
- On the second day deliver one parcel to each of customer 1 and 5.
- On the third day deliver one parcel to customers 1.
- On the fourth day deliver two parcels to customer 4 and one parcel to customer 6.
- And on the fiths day one parcel to each to customers 4 and 7.

3 Statistic Plots

[4]:
```
import scipy.stats as stats
```

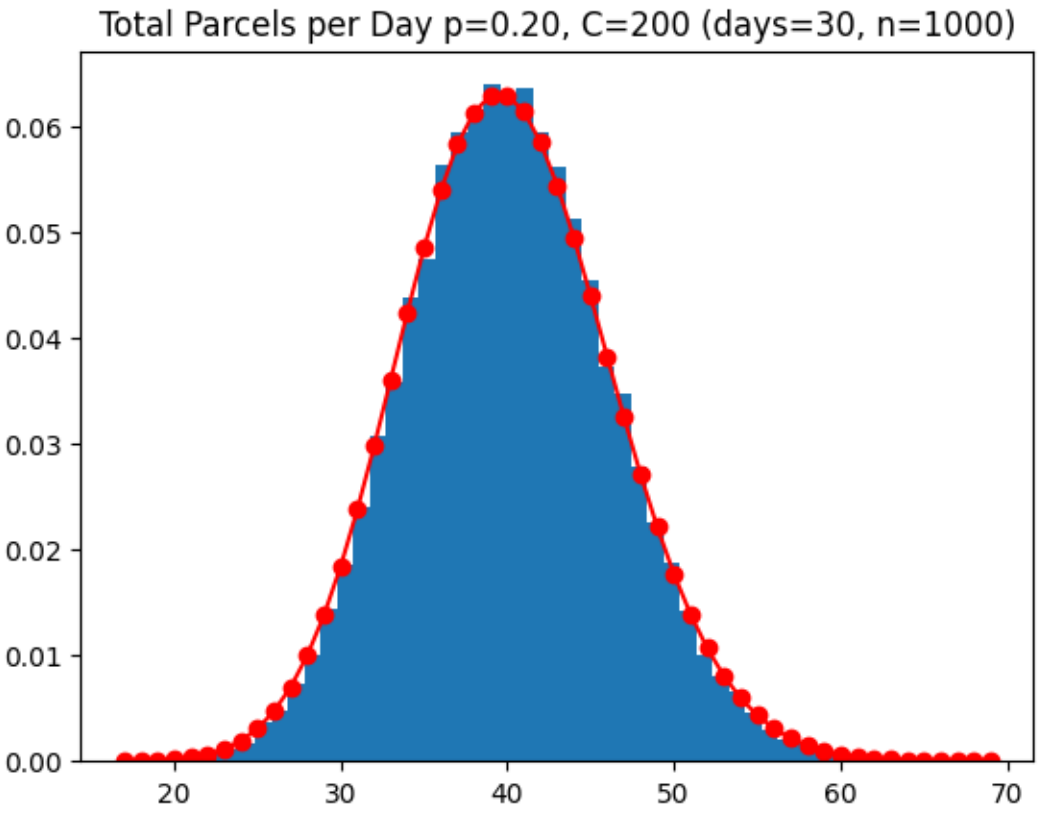
[5]:
```
def simulateDeliveriesPerDay(p, C=100, days=100, n=1000):
    deliveries = []
    for seed in range(n):
        D = generateDeliveries(p, C, 50, seed=seed)
        deliveries += [len(d) for d in D]

    mind = min(deliveries)
    maxd = max(deliveries)

    plt.hist(deliveries, bins=maxd-mind+1, density=True)
    plt.title(f"Total Parcels per Day p={p:4.2f}, C={C:d} (days={days:d}, n={n:
 ↪d})")
    poisson=stats.poisson(p*C)
    X = range(mind, maxd+1)
    plt.plot( X, [ poisson.pmf(x) for x in X ], color='red', marker='o')
    plt.show()
```

[6]:
```
def simulateDeliveriesPerCustomer(p, days=100, C=100, n=1000, log=False):
    total = [ 0 for c in range(C) ]
    mind = maxd = int(p*days)
    counts = [0]
    assert(len(counts) == maxd-mind+1)
    for seed in range(n):
        deliveries = [ 0 for c in range(C) ]
        D = generateDeliveries(p, C, days, seed=seed)
        for d in D:
            for c in d:
                deliveries[c] += 1

        # extend count to the left if required
        for i in range(mind-min(deliveries)):
            counts = [0] + counts
        # extend count to the right if required
        for i in range(max(deliveries)-maxd):
            counts = counts + [0]
        mind = min(mind, int(min(deliveries)))
        maxd = max(maxd, int(max(deliveries)))
        assert(len(counts) == maxd-mind+1)

        for c in range(C):
            counts[deliveries[c]-mind] += 1

    if not log:
```

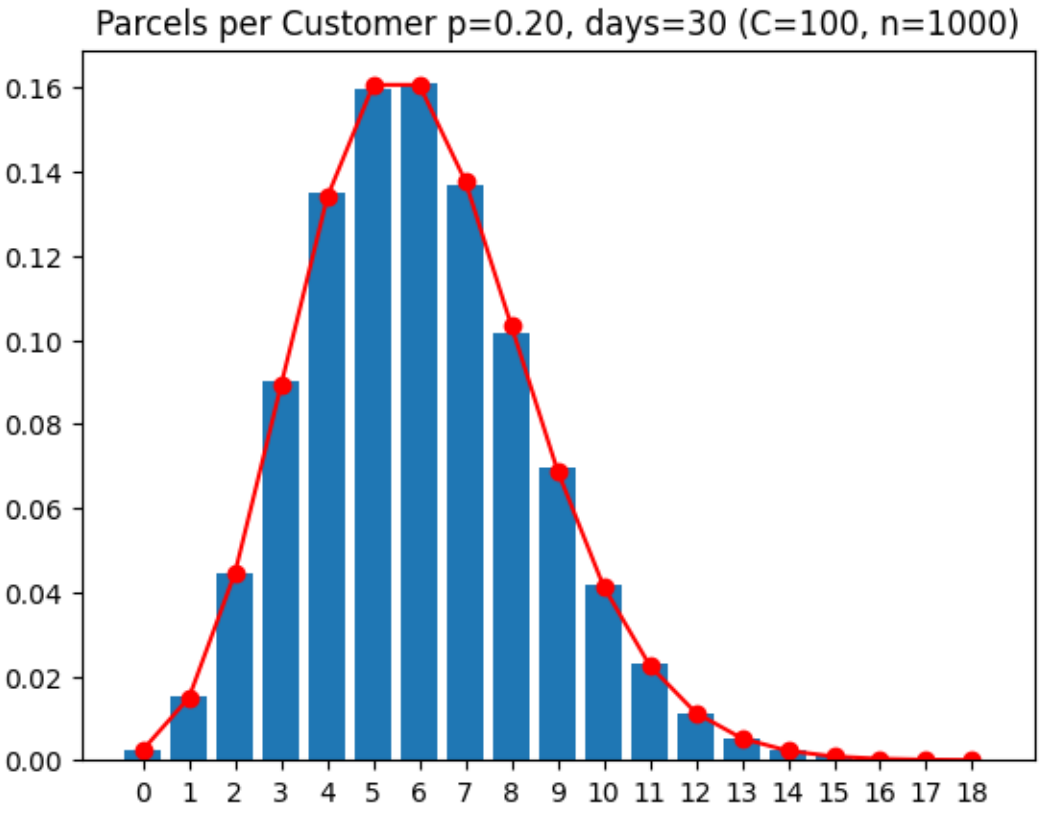
```
        for i in range(len(counts)):
            counts[i] = counts[i]/(n*C)

    plt.bar(x=[str(i) for i in range(mind, maxd+1)], height=counts, log=log)
    plt.title(f"Parcels per Customer p={p:4.2f}, days={days:d} (C={C:d}, n={n:
↪d})")

    poisson=stats.poisson(p*days)
    X = range(mind, maxd+1)
    Y = [ poisson.pmf(x) for x in X ]

    if log:
        Y = [ n*C*y for y in Y ]

    plt.plot( X, Y, color='red', marker='o')
    plt.show()
```

4 Statistical Analysis

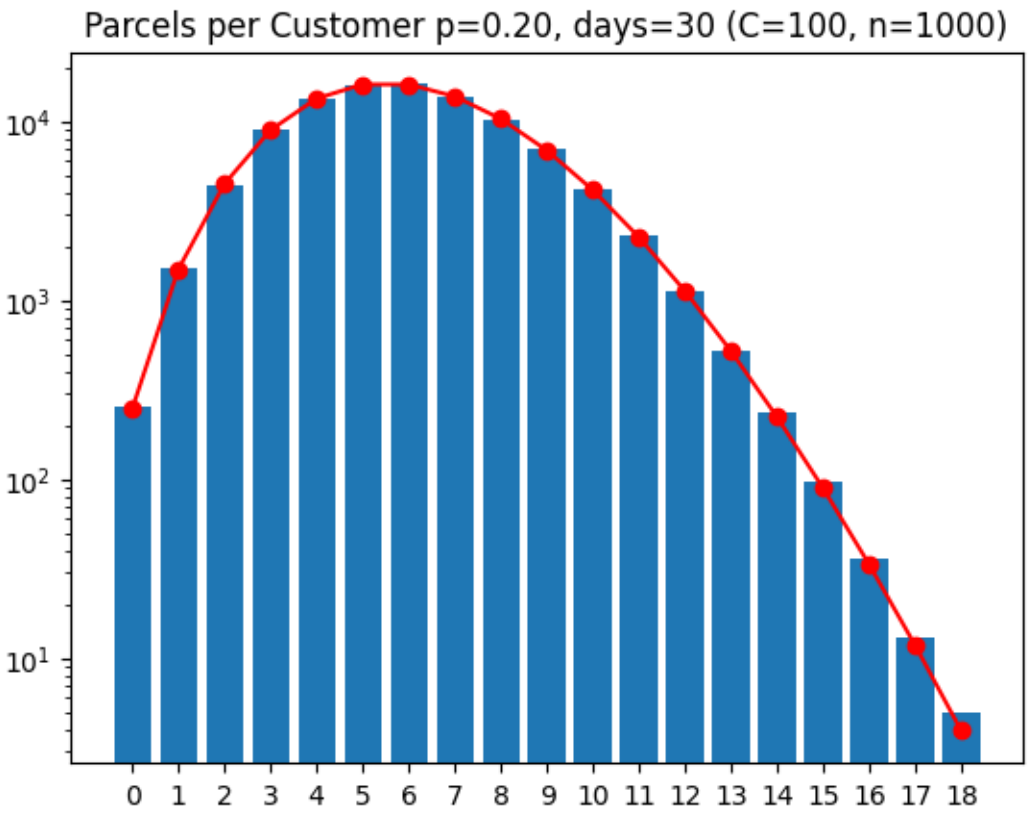
[7]:
```
simulateDeliveriesPerDay(0.2, C=200, days=30)
```



[8]:
```
simulateDeliveriesPerCustomer(0.2, days=30)
```



[9]:
```
simulateDeliveriesPerCustomer(0.2, days=30, log=True)
```


Parcels per Customer p=0.20, days=30 (C=100, n=1000)
10^4
10^3
10^2
10^1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18