Call Center Data Modelling

Posterior Confidence Intervals

Preprocessing

In [1]:

```
import numpy as np
import scipy as sp
import scipy.stats as sts
import matplotlib.pyplot as plt

%matplotlib inline
```

In [2]:

```
# Load the data set containing durations between calls arriving at the call
# center during 1 day. All values are in minutes.
waiting_times_day = np.loadtxt('call_center.csv')

# Display some basic information about the data set.
print('Size of data set:', len(waiting_times_day))
print('First 3 values in data set:', waiting_times_day[:3])
print('Sum of data set:', sum(waiting_times_day))
```

```
Size of data set: 5856
First 3 values in data set: [30.   3.4  3.2]
Sum of data set: 1441.6838153800093
```

In [3]:

```
# Make 24 empty lists, one per hour.
waiting_times_per_hour = [[] for _ in range(24)]

# Split the data into 24 separate series, one for each hour of the day.
current_time = 0
for t in waiting_times_day:
    current_hour = int(current_time // 60)
    current_time += t
    waiting_times_per_hour[current_hour].append(t)
```

Model

Likelihood: Exponential Distribution as it is the simplest Model that fits the shape of the data

Prior: Gamma distribution as it is the conjugate prior and is very customizable.

Prior parameters: Since I have very low confidence in my prior estimations, I will set alpha = 0.001, beta = 0.001. This is so that any new data that is evaluated will be much larger relative to these values and hence will quickly/ easily override any effect that the prior parameters might have on the posterior.

Calculating Posteriors

In [4]:

```
# for each hour
postparams_by_hour = [None for _ in range(24)]
for i,times in enumerate(waiting_times_per_hour):
    # update posterior
    ## Posterior = gamma(alpha+n, beta+sum(x))
    alpha = 0.001 + len(times)
    beta = 0.001 + sum(times)
    postparams_by_hour[i] = (alpha,beta)
```

In [5]:

```
meanLambda_by_hour = [None for _ in range(24)]
test = [None for _ in range(24)]

CI_by_hour = [None for _ in range(24)]
for i,postparams in enumerate(postparams_by_hour):
    # mean is alpha*1/beta
    meanLambda_by_hour[i] = sts.gamma.mean(postparams[0],scale=1/postparams[1])
    CI_by_hour[i] = sts.gamma.interval(0.98,postparams[0],scale=1/postparams[1])
```

In [6]:

```
meanLambda_by_hour
```

Out[6]:

```
[0.08211687821218043,
 0.05989431295938684,
 0.0783410138248848,
 0.21717651529545884,
 0.4277113388495008,
 0.8903092506425226,
 1.5809405488769377,
 2.8763245772201405,
 4.257946738995599,
 5.735802392371304,
 8.297251791809659,
 15.3897440125383,
 14.293158361432559,
 6.359111750941879,
 3.090235559866644,
 3.449882696221966,
 4.381057246707557,
 6.984538535640818,
 8.81700118530327,
 6.658450159710849,
 2.277963375937035,
 0.817874210205587,
 0.3260093560007171,
 0.27823954928010014]
```

In [7]:

```
CI_by_hour
```

Out[7]:

```
[(0.021010976501357217, 0.1905741173766468),
 (0.012330265998853741, 0.15039762172137183),
 (0.023313163061751994, 0.17114613158238112),
 (0.07889839164578151, 0.4343361636035827),
 (0.2570052810111013, 0.6466270752684866),
 (0.6307090239981112, 1.1992303299694524),
 (1.224724863267216, 1.987105586670805),
 (2.3921604780349064, 3.40936419319257),
 (3.6611754146965585, 4.90400602944679),
 (5.041950133042432, 6.478541448276308),
 (7.4552276367636425, 9.188469120468492),
 (14.23649658678494, 16.59197490149489),
 (13.182547882215983, 15.452761116026544),
 (5.626773835151244, 7.140400598881653),
 (2.5863724765171474, 3.6432046541860212),
 (2.9166815039065934, 4.032081183391074),
 (3.7771954736577253, 5.0338972437855585),
 (6.215342689560669, 7.802752985135049),
 (7.95137125847753, 9.731460806802628),
 (5.908517434237448, 7.4573321087582976),
 (1.8497935116311295, 2.7550056801108136),
 (0.5752191877669777, 1.1076108701155347),
 (0.18064693931681702, 0.5190890258828414),
 (0.14227267236370433, 0.46505143544502114)]
```

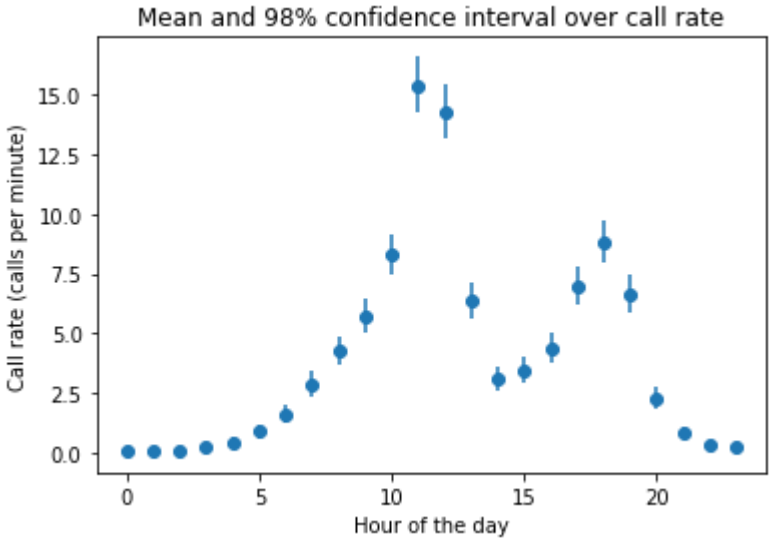
Plotting Posterior Confidence Intervals

In [8]:

```
CI_by_hour
error_bars = [[None for _ in range(24)],[None for _ in range(24)]]
for i, ci in enumerate(CI_by_hour):
    error_bars[0][i],error_bars[1][i] = (meanLambda_by_hour[i]-ci[0],ci[1]-meanLambd
```

In [9]:

```
fig, ax = plt.subplots()
x = np.arange(0,24)
ax.errorbar(x, meanLambda_by_hour, yerr=error_bars,fmt='o')
plt.xlabel("Hour of the day");
plt.ylabel("Call rate (calls per minute)");
plt.title("Mean and 98% confidence interval over call rate");
```



Presenting Findings

This plot describes the estimated number of calls we expect per minute (y-axis) differentiated by hour of the day (x-axis). Hour 0 represents midnight, and hour 23 represemts 11pm. The minimum call rate is approximately 0, and the maximum 16.

Focusing on the dots in the plot, we can see that almost all of the calls will occur during waking hours (5am to 9pm) with the call rate being almost 0 during the late night (9pm to 5am). The call rate peaks in the early afternooon (12pm) and early evening (5pm), with the afternoon peak being significantly larger. There is a dip in the call rate after lunch time (3pm).

The uncertainty of our estimates can be evaluated through the lines visible above and below each dot in the plot. There is a 98% chance that the line contains the true average call rate. However, this is only based on the data that we have. Thus, there is always a danger that we are missing crucial information or that the information we used is wrong. That said, assuming that the data we have is representative of the actual call rates, we can be very certain that these estimates are accurate.

In [ ]:

```
```