COVID19-Bayesian-Inference

By krsmanian1972@gmail.com

In [1]:

```
# Importing the Required library components

%matplotlib inline
import pymc3 as pm
from IPython.core.pylabtools import figsize
from matplotlib import pyplot as plt
import scipy.stats as stats
import numpy as np
import theano.tensor as tt
import seaborn as sns
```

Number of People Affected Per Day

We obtain the count of the number of people affected per day with COVID-19 from https://en.wikipedia.org/wiki/2020_coronavirus_pandemic_in_India (https://en.wikipedia.org/wiki/2020_coronavirus_pandemic_in_India)

The label for this count, hereafter, will be referred to as New Cases Per Day.

The Period of Data considered for the study is from 1st Apr to 17th Apr.

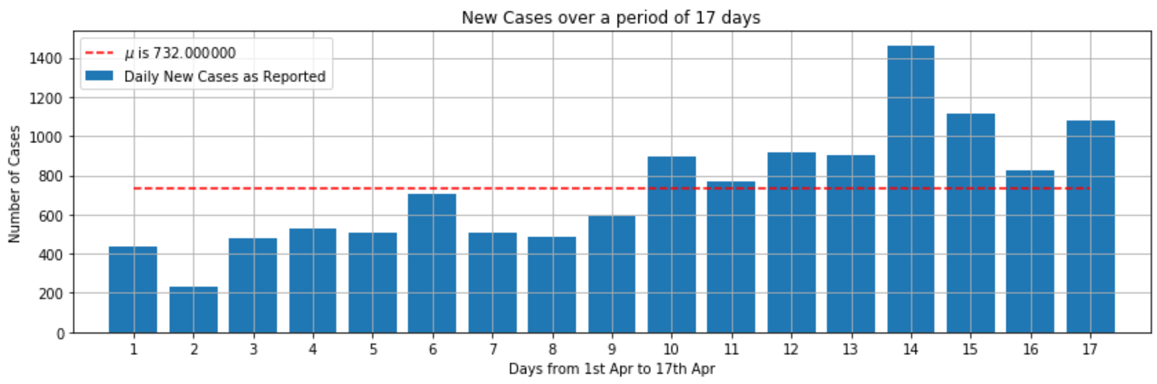
In [2]:

```
# Representation of the Daily Cases

daily_new_cases = np.array([437,235,478,525,505,704,508,485,591,896,768,918,905,
1463,1118,826,1076])
date_arr = np.array([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17])

total = daily_new_cases.sum()
total_days = len(daily_new_cases)
lambda0 = daily_new_cases.mean().round(0)
print("The Number of observed Days: {}".format(total_days))
print("The average count of New Cases Per Day: {}".format(lambda0))
print("The Total number of Reported Cases: {}".format(total))
```

```
The Number of observed Days: 17
The average count of New Cases Per Day: 732.0
The Total number of Reported Cases: 12438
```

In [3]:

```
# Visualization of the Daily Cases

figsize(14,4)
plt.bar(date_arr,daily_new_cases,label="Daily New Cases as Reported")

plt.title("New Cases over a period of {} days".format(total_days))
plt.plot((1,total_days),(lambda0,lambda0),"r--",label="$\mu$ is %f" %lambda0)
plt.xlabel("Days from 1st Apr to 17th Apr")
plt.ylabel("Number of Cases")
plt.xlim(0,total_days+1)
plt.xticks(date_arr)

plt.legend()
plt.grid()
```



Change-Point Analysis

Being the data size is very small, eye-balling of the data tells us that the rate of growth from 1st to 9th looks different from that of 10th to 17th.

However, we need to validate if the given data is a mix of more than one "distribution" and where that change could potentially occur through a method.

First, A quick introduction to certain jargon.

Distribution?.

In our context, the events are the number of people infected by the disease per day. We need a distribution to model the obtained data.

Poisson Distribution is appropriate for modeling the number of times an event occurs in an interval of time.

The Probability Mass Function (pmf) of Poisson Distribution is

$$f(k;\lambda) = \Pr(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

λ is the average number of events in an interval. k is the daily number of new cases (437,235,...,1076)

X ~ Pois(λ) is the shorter version to emphasize that Poisson distribution is a Function of the Parameter Lambda.

Change-point?.

I mean the date at which, the characteristic of the distribution changes.

Scenario-1

You are a developer. And hence cloud-watching your Amazon RDS may be one of our hobbies. The CPU utilization metric shows a sudden spike. You are proactive. You turn the slow-query log and identify the slow running query. Give a fix like a hero and again observe the CPU Utilization chart. You are happy upon seeing the CPU utilization is back to the "accepted normal" mark.

Scenario-2

Your supplier is very prompt for the first 100 days in a row. He is Prompt in delivering by 10:00 am. The accepted normal is 10:00 am. You see him arrive late by an hour for the last 5 days in a row. Will you act or not?

The Key Points are :

You are assigning importance to the latest events.

You are not complacent about the historical average.

You detect the change against an acceptable limit.

In our case we do not have an acceptable limit. I wish it to be zero.

Several Lambdas:

Q: What is Lamdba? A: The Rate of New Cases Detected Per Day

Q: Can there be more than one Lambda that satisfy the PMF equation? Yes, we are expecting a distribution for the Lambda itself.

Q: And are we expecting two such distributions? A: Yes, I'm expecting the given data is a mix of two Poisson Distribution. Hence there can be more than one Lamdba Distributions.

Modeling the scenario using PyMC3 - a Bayesian Modeling Framework

Now our objective is to capture the Lambdas that drive our data.

The probabilistic design for the given context in the reverse order.

1. The observation is X - The discrete set of random variables. The count of new cases per day.

2. X ~ Pois(λ)
   We can model our observation, X, with Poisson Distribution. But,We need the parameter λs for that.

   Who can provide the λs?

   From the Exponential Distribution.

3. λ ~ Exp(alpha)

   We can draw samples for Lambda from the Exponential Distribution.

   Exponential Distribution is to model the "amount of time until" an event occurs.
   Poisson Distribution is to model the Rate of the event.

   We call this parameter of the exponential distribution as alpha

4. The relationship between alpha and lambda is alpha = 1/λ
   We know that the Expected Value of a Poisson Distribution is the Average of the Events.
   EV = The average count of New Cases Per Day = 732.0
   Hence the prior for alpha = 1.0/732.0

   alpha is the hyper-parameter for lambda.

Note:

We can of course, draw samples for Lambda from Gamma Distribution as well. Let me prove later that there is no difference between both the strategies, because Exponential Distribution is a special case of Gamma Distribution.

In [4]:

```
# the avg_model encompasses all the observed data

with pm.Model() as avg_model:
    alpha = 1.0/daily_new_cases.mean() # Prior.
    lambda_for_whole = pm.Exponential("lambda_for_whole", alpha)
    obs_whole = pm.Poisson("obs_whole",lambda_for_whole,observed=daily_new_cases
)
```

In [5]:

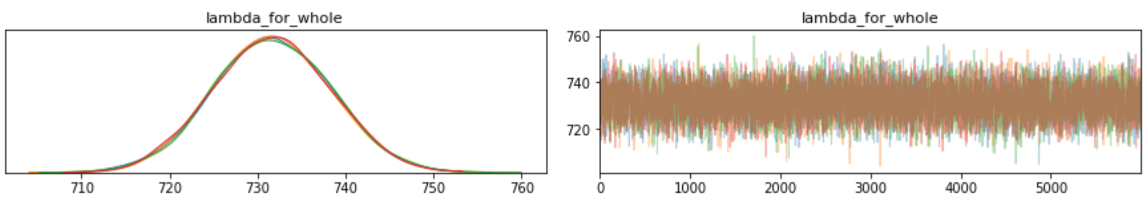
```
# PyMC3 has the implementations of the family of Markov-Chain-Monte-Carlo algori
thms.
# NUTS is a flavor of the family of MCMC. No U-Turn Sampling.
# We have plans to create our own implementation of an MCMC later with RUST lang
after a couple of days.

# Now do simulation to draw around 6000 samples of Lambdas.

with avg_model:
    trace_avg = pm.sample(6000,tune=1500)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (4 chains in 4 jobs)
NUTS: [lambda_for_whole]
Sampling 4 chains, 0 divergences: 100%|██████████| 30000/30000 [00:0
3<00:00, 7670.73draws/s]
```

In [6]:

```
# Alright, MCMC Proposed a distribution of the probable lambdas for the Poisson
 Distribution.
# Before we proceed, it is important to validate if the sampling process is Conv
erged.
# Note The peaks of all the four chains of lambdas is nearer to the average 730.

pm.traceplot(trace_avg);
```

```
/Users/harinimaniam/neural/lib/python3.7/site-packages/arviz/plots/b
ackends/matplotlib/distplot.py:38: UserWarning: Argument backend_kwa
rgs has not effect in matplotlib.plot_distSupplied value won't be us
ed
  "Argument backend_kwargs has not effect in matplotlib.plot_dist"
```



In [7]:

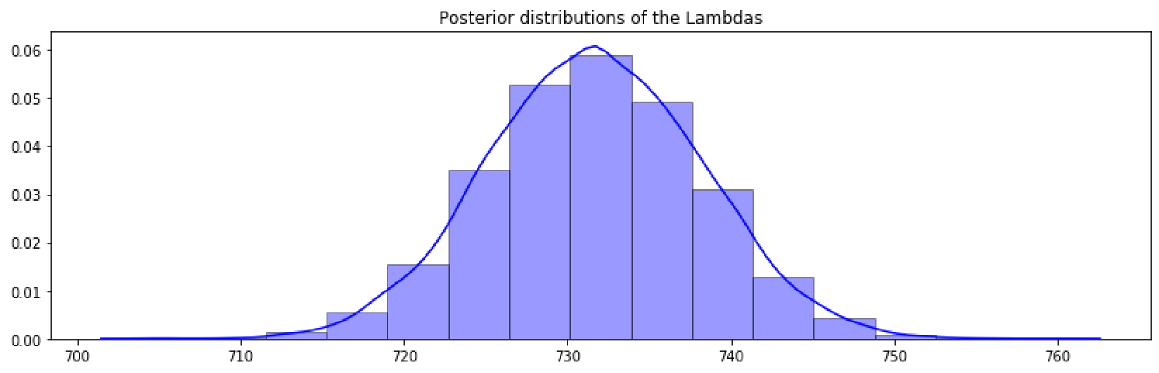
```
# Convergence is confirmed by the r_hat of the parameter is 1.0
# We get certain high-level statistics about the sampling as well.

pm.summary(trace_avg)
```

Out[7]:

	mean	sd	hpd_3%	hpd_97%	mcse_mean	mcse_sd	ess_mean	ess
lambda_for_whole	731.662	6.567	718.81	743.484	0.065	0.046	10347.0	1034

In [8]:

```
avg_lambdas = trace_avg["lambda_for_whole"]
plot_1 = sns.distplot(avg_lambdas, hist=True, kde=True, bins=15, color = 'blue',
hist_kws={'edgecolor':'black'})
plot_1 = plot_1.set_title("Posterior distributions of the Lambdas")
```



Bayesian Vs Frequentist

1. The bayesian framework offers you a probability distribution of the Lamdba, instead of a single value.
2. Each bar of the Posterior distribution is a relative probability of the number of new cases to observe.
3. The Uncertainty is the Key to the inference.

Inference - 1

It is TOOOOOO early to conclude that we have around 60% of chances for any expected number of new cases.

Now it is time to detect the Changepoint.

Remember our key point, as decided earlier, "You are assigning importance to the latest events"

The true rate of new cases, the true lambda, is shadowed by the history.

Distilling the current distribution of lambda(s) is our immediate goal. Note the Plural - Bayesian.

In [9]:

```
# Obtain posterior samples from two Exponential Distributions.

# Use the switch function to randomly collect lambdas before and after a randoml
y selected date
# Random Ethana Random Sir!!!

# The idea is if there is no change then both the lambda distribution will be ne
arly identical.
# If there is a change we will end-up in collecting two lambda distributions

# We will get change-point (tau) also.

total_days = len(daily_new_cases)
with pm.Model() as cp_model:
    alpha = 1.0/daily_new_cases.mean()

    lambda_1 = pm.Exponential("lambda_1", alpha)
    lambda_2 = pm.Exponential("lambda_2", alpha)

    tau = pm.DiscreteUniform("tau",lower=0,upper=total_days-1)

    idx = np.arange(total_days)
    lambda_switch = pm.math.switch(tau>idx,lambda_1,lambda_2)

    obs_cp = pm.Poisson("obs_cp",lambda_switch,observed=daily_new_cases)
```

In [10]:

```
# Do the simulation
# Watch for the selection of NUTS for the Continuous Distribution (Exponential D
istribution)
# Watch for the selection of Metropolis for Discrete Distribution (Discrete Unif
orm)
with cp_model:
    trace_cp = pm.sample(10000,tune=5000)
```

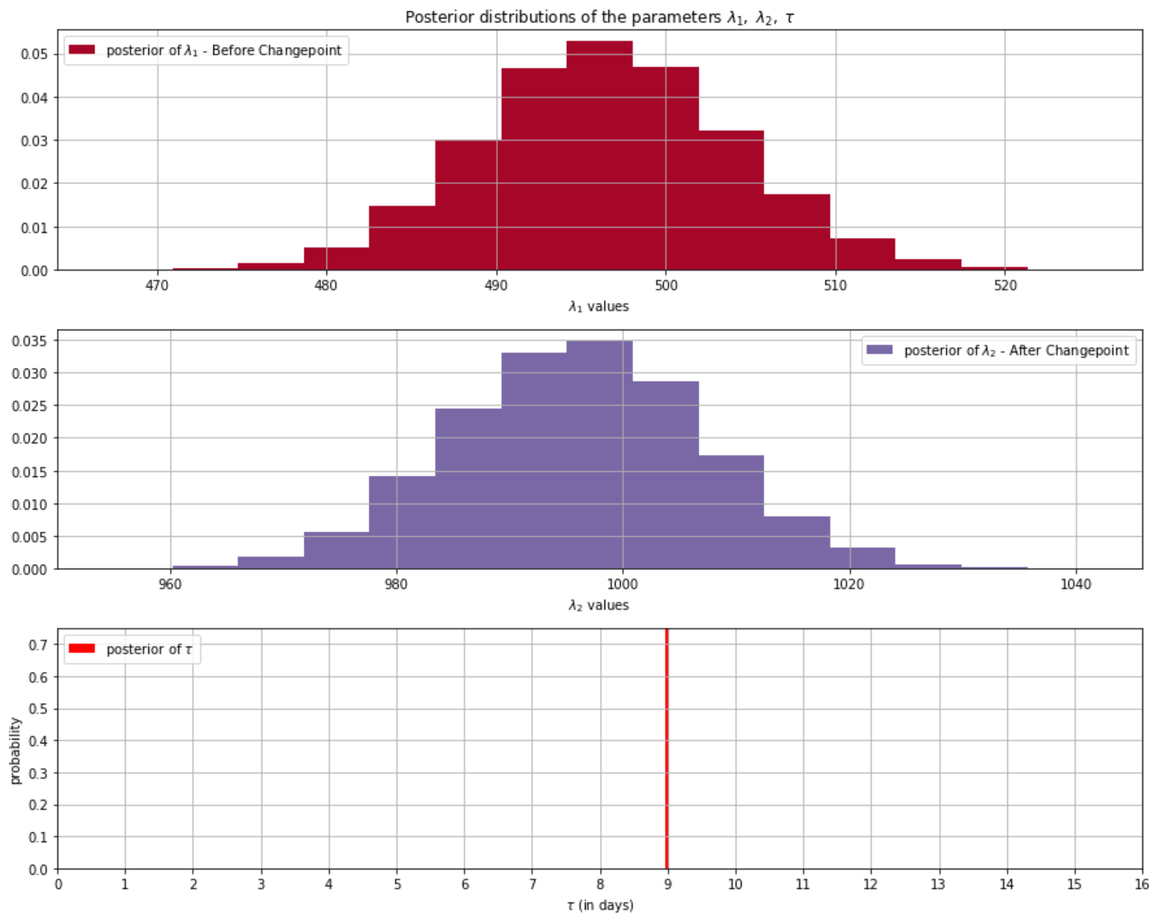
```
Multiprocess sampling (4 chains in 4 jobs)
CompoundStep
>NUTS: [lambda_2, lambda_1]
>Metropolis: [tau]
Sampling 4 chains, 0 divergences:  75%|███████▌  | 45000/60000 [00:4
9<00:16, 907.78draws/s]
/Users/harinimaniam/neural/lib/python3.7/site-packages/xarray/core/n
putils.py:223: RuntimeWarning: All-NaN slice encountered
  result = getattr(npmodule, name)(values, axis=axis, **kwargs)
```

In [11]:

```
# Time to collect the samples
lambda_1_samples = trace_cp['lambda_1']
lambda_2_samples = trace_cp['lambda_2']
tau_samples = trace_cp['tau']
```

In [12]:

```
# Plot the Posterior Parameters lambda_1, lambda_2 and Tau as Histogram of Proba
bility Density
# The Changepoint is tau is also a probability distribution.
# Look for potential changepoints

figsize(12.5, 10)

ax=plt.subplot(311)
plt.title(r"""Posterior distributions of the parameters $\lambda_1,\;\lambda_
2,\;\tau$""")
plt.hist(lambda_1_samples, bins=15,label="posterior of $\lambda_1$ - Before Chan
gepoint", color="#A60628", density=True)
plt.legend(loc="upper left")
plt.xlabel("$\lambda_1$ values")
plt.grid()

ax = plt.subplot(312)
plt.hist(lambda_2_samples, bins=15, label="posterior of $\lambda_2$ - After Chan
gepoint", color="#7A68A6", density=True)
plt.legend(loc="upper right")
plt.xlabel("$\lambda_2$ values")
plt.grid()

ax=plt.subplot(313)
w = 1.0 / tau_samples.shape[0] * np.ones_like(tau_samples)
plt.hist(tau_samples, bins=total_days, label=r"posterior of $\tau$",color="r", w
eights=w )
plt.xticks(np.arange(total_days))
plt.legend(loc="upper left")
plt.ylim([0, .75])
plt.xlabel(r"$\tau$ (in days)")
plt.ylabel("probability");

plt.grid()
plt.tight_layout()
```



Change-point Detected

We have successfully separated the two distributions.

Thanks to the hierarchical model of PyMC3 and to the many authors I referred to.

Lambda-2 is the latest trend. Which we can use to make the predictions by sampling from the Poisson Distribution.

As we have guessed already the distribution splits on the Ninth Day of April

Please wait before performing the predictions. We have a couple of activities to perform to prove our theory.

Box Plot the Cases of Before and After the Change-Point

We are interested in the new_cases those are on or after the changepoint.

In [13]:

```
change_point_index = 9
old_cases = daily_new_cases[0:change_point_index]
new_cases = daily_new_cases[change_point_index:]
new_dates = date_arr[9:]
```

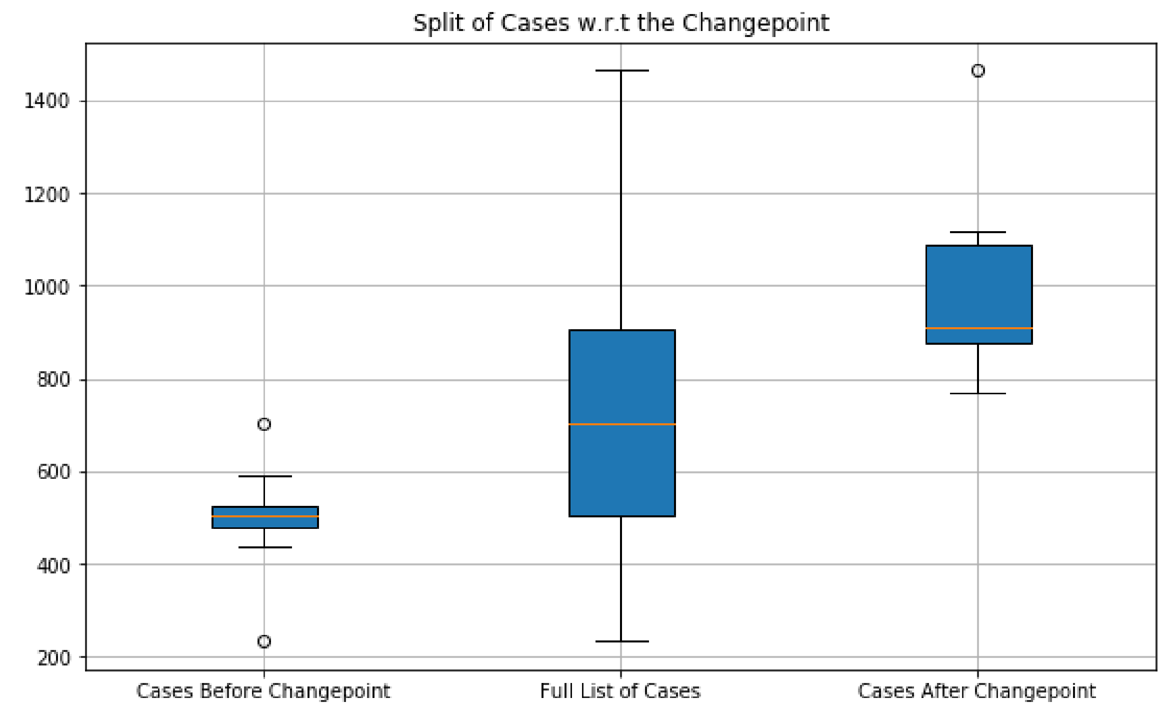
In [14]:

```
size = len(new_cases)
mean = new_cases.mean()

print("Number of New Cases {}".format(size))
print("Number of Date Array {}".format(len(new_dates)))
print("Average Number of New Cases {}".format(mean))
```

```
Number of New Cases 8
Number of Date Array 8
Average Number of New Cases 996.25
```

In [15]:

```
fig, ax = plt.subplots(figsize=(10,6))
plt.title("Split of Cases w.r.t the Changepoint")
ax.boxplot([old_cases,daily_new_cases,new_cases],patch_artist=True);
ax.set_xticklabels(['Cases Before Changepoint', 'Full List of Cases', 'Cases Aft
er Changepoint'])
plt.grid()
```



Modeling the New Cases After the Switchpoint - Gamma Distribution

Now, it is time to prove to the audience that both the Gamma distribution and Exponential Distribution are equally good to model the posterior parameter, Lambdas, of the Poisson Distribution.

The reason I've used the exponential distribution is that it requires a single hyper-parameter alpha. But the Gamma distribution requires two hyper-parameters.

However it is very interesting to define the modeling.

Again the model definition in the reverse order.

1. The observation is X - The discrete random variables. The count of new cases per day. I take the cases after the changepoint now.
2. A Gamma Distribution, to propose samples for the Poisson Distribution. It requires two-parameters viz., alpha and beta. I defer discussing about alpha and beta for another occasion.
3. alpha and beta are functions of two other parameters. We call them mu and sigma
4. Draw samples for sigma from the exponential distribution. I set the prior is 1.0
5. Draw samples for mu from another Gamma distribution. With alpha set to 1.5 and beta set to 1.0/size

In [2]:

```
# A Hack for resuming the analysis from here when Jupyter breaks
new_cases = np.array([ 896,768,918,905,1463,1118,826,1076])
size=len(new_cases)
```

In [3]:

```
# The Hiarchial model definition. I call this as gamma_model

with pm.Model() as gamma_model:

    mu_ = pm.Gamma('mu', alpha=1.5, beta=1.0/size)
    sigma_ = pm.Exponential('sigma', 1.0)

    alpha_ = mu_**2/sigma_**2
    beta_ = mu_/sigma_**2

    lambda_ = pm.Gamma("lambda", alpha_,beta_)
    obsx = pm.Poisson("obsx",lambda_,observed=new_cases)
```

In [4]:

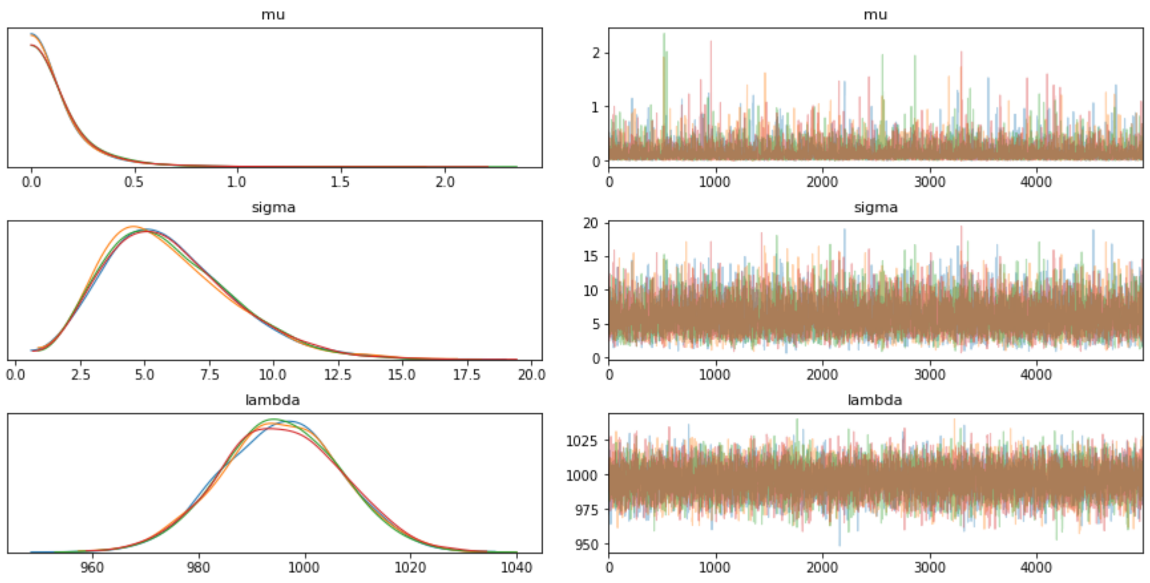
```
# Simulate the Gamma model
with gamma_model:
    tracex = pm.sample(5000,tune=1000)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (4 chains in 4 jobs)
NUTS: [lambda, sigma, mu]
Sampling 4 chains, 0 divergences: 100%|██████████| 24000/24000 [00:0
7<00:00, 3353.26draws/s]
```

In [5]:

```
# Ignore the warnings.
# We are interested in the convergence of the model. Refer the RHat

pm.traceplot(tracex);
```

```
/Users/harinimaniam/neural/lib/python3.7/site-packages/arviz/plots/b
ackends/matplotlib/distplot.py:38: UserWarning: Argument backend_kwa
rgs has not effect in matplotlib.plot_distSupplied value won't be us
ed
  "Argument backend_kwargs has not effect in matplotlib.plot_dist"
/Users/harinimaniam/neural/lib/python3.7/site-packages/arviz/plots/b
ackends/matplotlib/distplot.py:38: UserWarning: Argument backend_kwa
rgs has not effect in matplotlib.plot_distSupplied value won't be us
ed
  "Argument backend_kwargs has not effect in matplotlib.plot_dist"
/Users/harinimaniam/neural/lib/python3.7/site-packages/arviz/plots/b
ackends/matplotlib/distplot.py:38: UserWarning: Argument backend_kwa
rgs has not effect in matplotlib.plot_distSupplied value won't be us
ed
  "Argument backend_kwargs has not effect in matplotlib.plot_dist"
```



In [6]:

```
# Convergence is validated by R_Hat
pm.summary(tracex)
```

Out[6]:

	mean	sd	hpd_3%	hpd_97%	mcse_mean	mcse_sd	ess_mean	ess_sd	ess_l
mu	0.147	0.161	0.002	0.419	0.002	0.001	6986.0	6986.0	64
sigma	5.976	2.470	1.843	10.546	0.030	0.021	6926.0	6926.0	67
lambda	995.632	11.288	974.508	1016.802	0.120	0.085	8897.0	8897.0	89

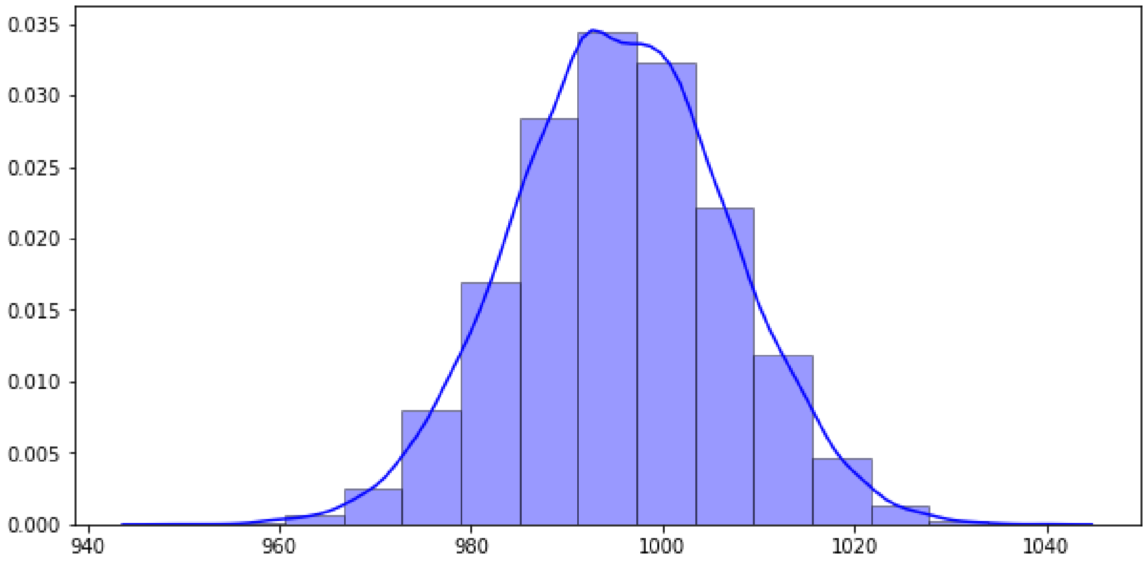
In [7]:

```
lambda_samples = tracex['lambda']
```

In [8]:

```
fig, ax = plt.subplots(figsize=(10,5))
sns.distplot(lambda_samples, hist=True, kde=True, bins=15, color = 'blue',hist_k
ws={'edgecolor':'black'})
print("Posterior Lambdas collected from Gamma Distribution for the Poisson Distr
ibution")
```

```
Posterior Lambdas collected from Gamma Distribution for the Poisson
Distribution
```



Modeling the New Cases Post Changepoint - Just the Exponential Distribution to propose Lambdas

This is the first exercise we did in the begining but with the entire data. Now we are interested only in the post changepoint data.

Remember Occam's Razor. I call this model as "Simple Model".

In [10]:

```
with pm.Model() as simple_model:
    alpha = 1.0/new_cases.mean()
    lambda_1 = pm.Exponential("lambda_1", alpha)
    obs_simple = pm.Poisson("obs_simple",lambda_1,observed=new_cases)
```

In [11]:

```
with simple_model:
    trace_simple = pm.sample(4000,tune=1500)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (4 chains in 4 jobs)
NUTS: [lambda_1]
Sampling 4 chains, 0 divergences: 100%|██████████| 22000/22000 [00:0
3<00:00, 7319.10draws/s]
```

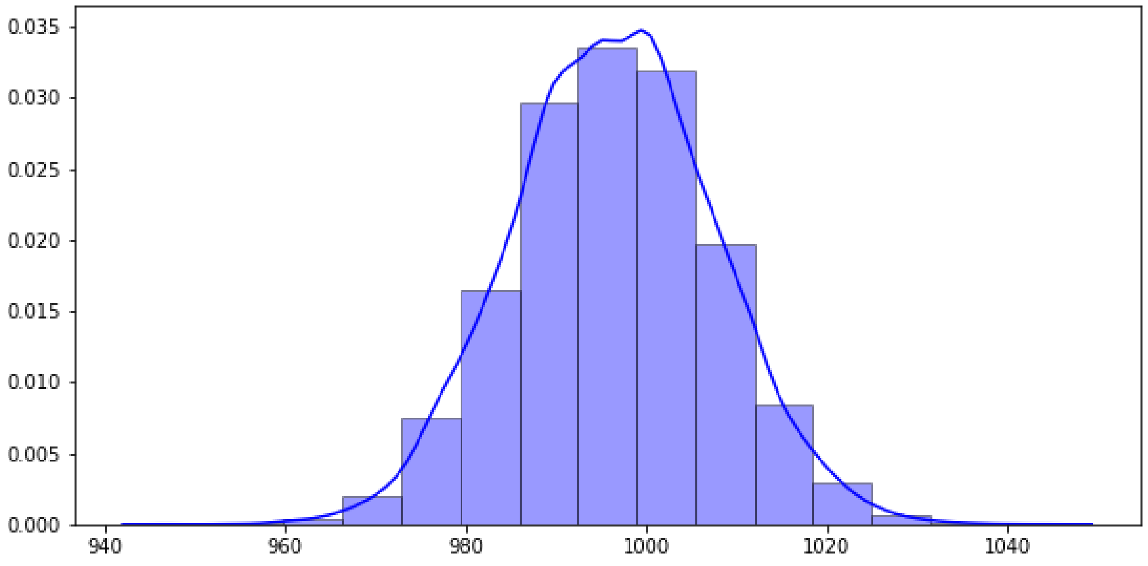
In [12]:

```
lambda_1_samples = trace_simple['lambda_1']
```

In [13]:

```
fig, ax = plt.subplots(figsize=(10,5))
sns.distplot(lambda_1_samples, hist=True, kde=True, bins=15, color = 'blue',hist
_kws={'edgecolor':'black'})
print("Posterior Lambdas collected from Expoential Distribution for the Poisson
 Distribution")
```

```
Posterior Lambdas collected from Expoential Distribution for the Poi
sson Distribution
```



In [16]:

```
pm.summary(trace_simple)
```

Out[16]:

	mean	sd	hpd_3%	hpd_97%	mcse_mean	mcse_sd	ess_mean	ess_sd	es
lambda_1	996.396	11.187	975.814	1017.563	0.138	0.098	6529.0	6528.0	(

Proof of Gamma and Exponential are equally appropriate

The reader can compare now the Posterior of Lambda from the Gamma distribution and Exponential Distribution are almost identical.

Inference

Now we have the posterior parameters for the Poisson Distributed Daily New Cases after the Switch Point.

So, we are ready to answer certain questions using the lambdas that we have collected.

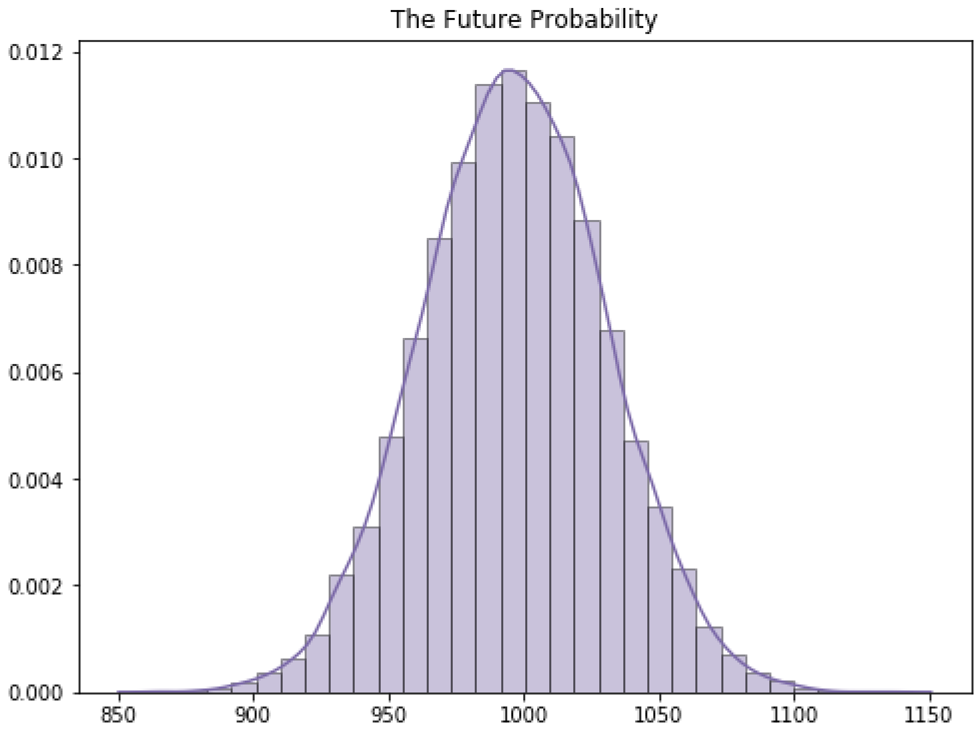
Q: What is the expected number of new cases given the current context?

The bayesian answer will be again a distribution as captured below.

In [20]:

```
future = np.random.poisson(lam=trace_simple['lambda_1'])
fig, ax = plt.subplots(figsize=(8, 6))
sns.distplot(future, color="#7A68A7", bins=30, ax=ax,hist_kws={'edgecolor':'blac
k'})
plt.title("The Future Probability")
print("The Probability of daily new new cases")
```

```
The Probability of daily new new cases
```



Okay. We understand. There is no single answer.

How many cases we can expect between 900 to 1000 ?

In [21]:

```
def how_many(future,start,end):
    count = 0
    for x in future:
        if(x>= start and x<= end):
            count+=1
    return count/len(future)


print("Probability of daily new cases between 900 and 1000 : {}".format(how_many
(future,900,1000)))
print("Probability of daily new cases between 1000 and 1200 : {}".format(how_man
y(future,1000,1200)))

less_than_900 = (future < 900).astype(int).mean()
less_than_1000 = (future < 1000).astype(int).mean()
above_1200 = (future > 1200).astype(int).mean()

print("Probability of daily news cases less than 900 : {}".format(less_than_900
))
print("Probability of daily news cases less than 900 : {}".format(less_than_1000
))

print("Probability of daily news cases above 1200 : {}".format(above_1200))
```

```
Probability of daily new cases between 900 and 1000 : 0.5445
Probability of daily new cases between 1000 and 1200 : 0.466625
Probability of daily news cases less than 900 : 0.002125
Probability of daily news cases less than 900 : 0.533375
Probability of daily news cases above 1200 : 0.0
```

Good News & ...

We have more than 50% chances for the number of cases to reduce to less than 900.

Having said that, we have experienced a changepoint where the rates are pretty high.

Social Distancing is the only option to help to break the chain of transmission or to reduce the

எதிரதாக் காக்கும் அறிவினார்க் கில்லை அதிர வருவதோர் நோய்.

No terrifying calamity will affect the wise, who foresee and guard against evils.

Thanks

In [ ]: