PHYS 411

Assignment 1

Import required packages:

```
[2]   import random
      import numpy as np
      import scipy as sci
      from scipy.stats import norm
      import matplotlib.pyplot as plt
      from IPython.display import Math, display
```

Question 2: Monte Carlo simulation of the Central Limit theorem

Generate Data:

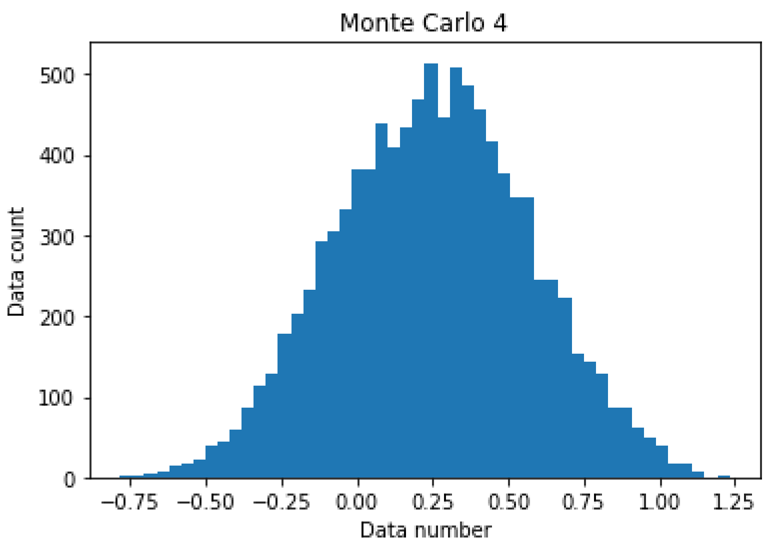
```
[11]  mean = 0.5
      datanumber = 10000
      n_bins = 50

      # Generate an array A1
      # Then subtract 0.5 from all the values
      A0 = np.random.rand(datanumber)
      A1 = np.random.rand(datanumber)
      A2 = A0 + A1 - 0.5

      # Doing it 4, 8, and 16 more times:
      n4 = [4, 8, 16]

      A3 = A1
      for i in range(0, n4[0]):
          T1 = np.random.rand(datanumber)
          A3 = A3 + T1 - 0.5

      A4 = A1
      for j in range(0, n4[1]):
          T2 = np.random.rand(datanumber)
          A4 = A4 + T2 - 0.5

      A5 = A1
```

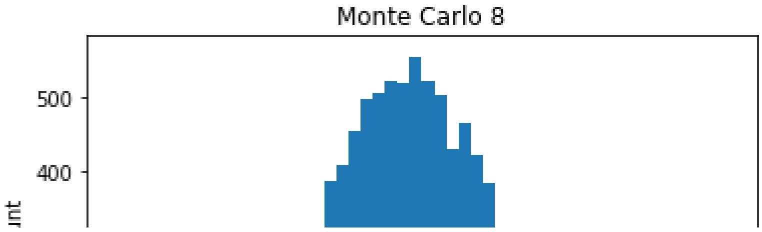
```python
    for k in range(0, n4[2]):
        T3 = np.random.rand(datanumber)
        A5 = A5 + T3 - 0.5
```

Plot Data:

[12]

```python
plt.figure(1)
plt.hist(A1, bins=n_bins)
plt.xlabel(r'Data number')
plt.ylabel(r'Data count')
plt.title(r'Uniform probability distribution between 0 and 1')

plt.figure(2)
plt.hist(A2/np.sqrt(2), bins=n_bins)
plt.xlabel(r'Data number')
plt.ylabel(r'Data count')
plt.title(r'Monte Carlo 2')

plt.figure(3)
plt.hist(A3/np.sqrt(n4[0]), bins=n_bins)
plt.xlabel(r'Data number')
plt.ylabel(r'Data count')
plt.title(r'Monte Carlo 4')

plt.figure(4)
plt.hist(A4/np.sqrt(n4[1]), bins=n_bins)
plt.xlabel(r'Data number')
plt.ylabel(r'Data count')
plt.title(r'Monte Carlo 8')

plt.figure(5)
plt.hist(A5/np.sqrt(n4[2]), bins=n_bins)
plt.xlabel(r'Data number')
plt.ylabel(r'Data count')
plt.title(r'Monte Carlo 16')
```





As we can see, as the graphs approach a Gaussian as the sample count increases. This is a result of probability. Data in this graph is generated in a range from 0 to 1, then the values are counted and plotted on a histogram. A second set of data is generated and added to the original set, this process is repeated multiple times. The histogram will approach a Gaussian because it is unlikely for a value in the data set to be repeatedly assigned an extreme value at one end of the spectrum, hence, decreasing the data count at the extremes with each successive data set. Values from the repeated addition of data sets will tend to average towards the expected value, therefore, data counts close to the expected value will increase. Thus, the data will tend towards the normal distribution (a Gaussian) as the sample size increases.

Question 3: Flipping two coins

[8]

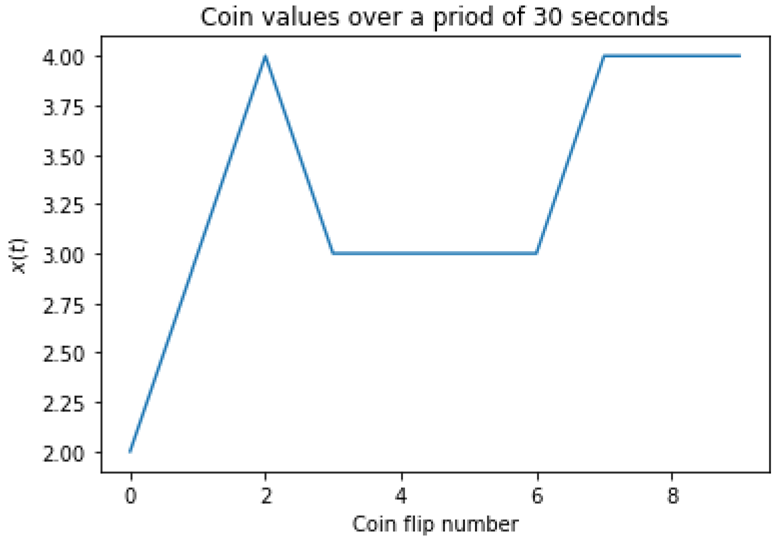
```
t = 60*60   # Total flipping time
p = 3       # Period between flips

# Generate coin flips
B0 = np.random.randint(1, 3, int(t/p)+2)
B1 = []

# Add the coin values
for n in range(0, len(B0)-1):
    b = B0[n] + B0[n+1]
    B1.append(b)

print(B1[:10])
```

```
[2, 3, 4, 3, 3, 3, 3, 4, 4, 4]
```

a) Possible outcome of x(t) for the first 30s:

```
[9]  plt.figure(6)
     plt.plot(B1[:10])
     plt.xlabel(r'Coin flip number')
     plt.ylabel(r'$x(t)$')
     plt.title(r'Coin values over a priod of 30 seconds')
     print(B1[:10])
```

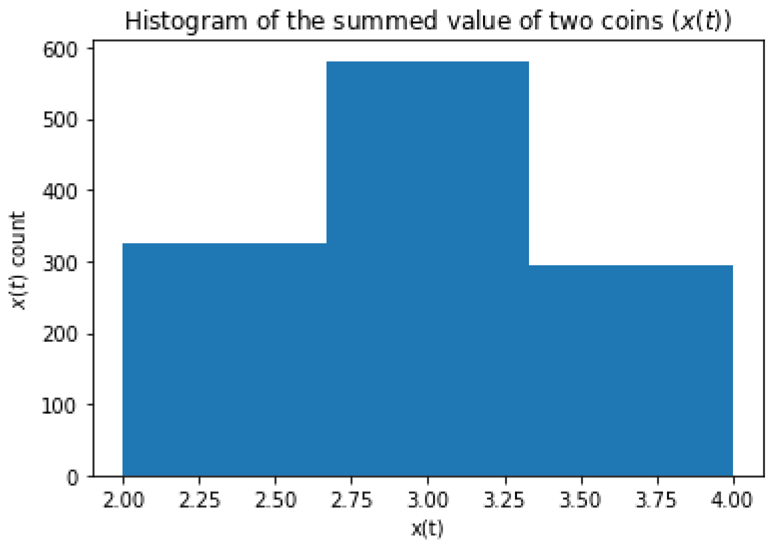
```
[2, 3, 4, 3, 3, 3, 3, 4, 4, 4]
```



b) A likely histogram of x(t)

```
[10]  plt.figure(7)
      plt.hist(B1, bins=3)
      plt.xlabel(r'x(t)')
      plt.ylabel(r'$x(t)$ count')
      plt.title(r'Histogram of the summed value of two coins ($x(t)$)')
```

```
Text(0.5,1,'Histogram of the summed value of two coins ($x(t)$)')
```



c) Probability density function $p(x)$, and probability distribution function $P(x)$

Probability density function: $p(x)$

$$p(x) = \frac{1}{4}\delta(x-2) + \frac{1}{2}\delta(x-3) + \frac{1}{4}\delta(x-4)$$

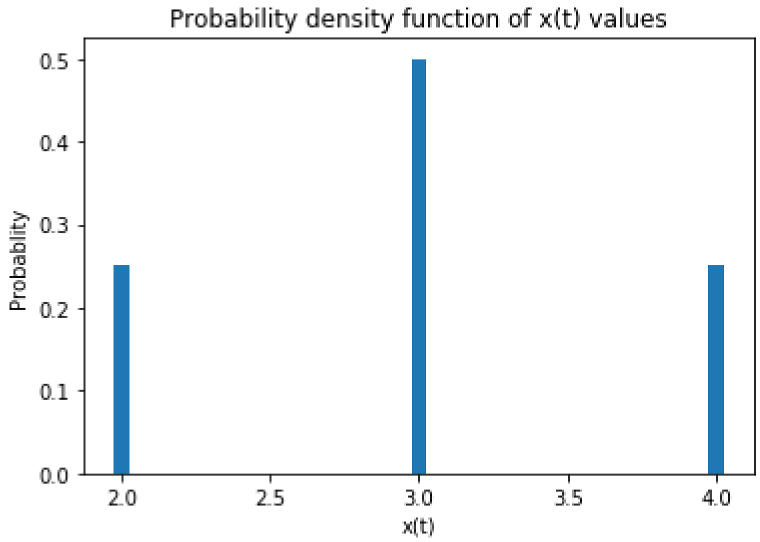
Probability distribution function: $P(x)$

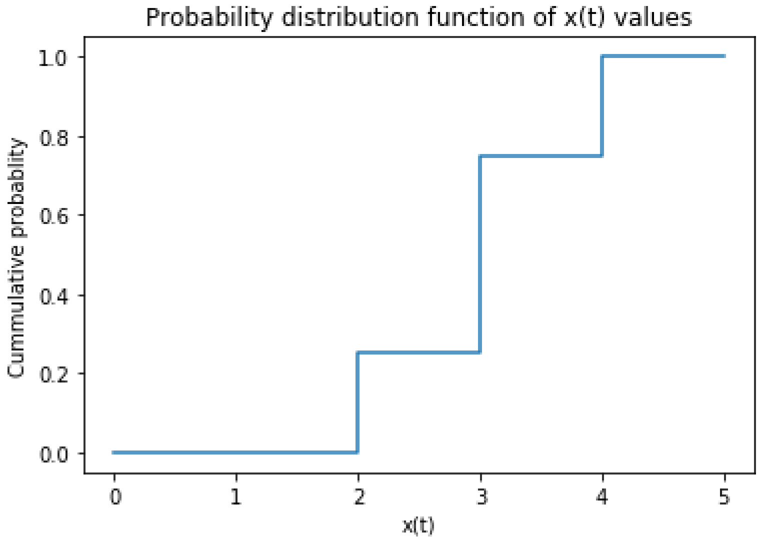
$$P(x) = \begin{cases} \frac{1}{4} & x < 2 \\ \frac{3}{4} & 2 \leq x < 3 \\ 1 & x \geq 3 \end{cases}$$

[11]

```
rang = [2, 3, 4]
RANG = [0, 2, 3, 4, 5]
pdf = [0.25, 0.5, 0.25]
PDF = [0, 0, 0.25, 0.75, 1]

# Plot the probability density function
plt.figure(8)
plt.bar(rang, pdf, width=0.05)
plt.xlabel(r'x(t)')
plt.ylabel(r'Probablity')
plt.title(r'Probability density function of x(t) values')

# Plot the probability distribution function
plt.figure(9)
plt.step(RANG, PDF)
plt.xlabel(r'x(t)')
plt.ylabel(r'Cummulative probablity')
plt.title(r'Probability distribution function of x(t) values')
```

```
Text(0.5,1,'Probability distribution function of x(t) values')
```





d) The mean μ_x of $x(t)$, and the variance σ_x^2 of $x(t)$

```
[12]  # Find the mean
      mean = round(np.mean(B1), 1)
      display(Math(r'\mu_{x}=%.2f' % mean))

      # Find the varience
      var = round(np.var(B1), 1)
      display(Math(r'\sigma_{x}^{2}=%.2f' % var))
```

$$\$\$\mu_x = 3.00\$\$$$

$$\$\$\sigma_x^2 = 0.50\$\$$$

Question 4: Comparison of minute-resolution and hourly-resolution data

```
[3]  # Load the data
     D1 = np.loadtxt('/Users/Kev/Documents/Uvic/Python/PHYS 411 - Time Series /
     D2 = np.loadtxt('/Users/Kev/Documents/Uvic/Python/PHYS 411 - Time Series /

     # Check data dimensions
     print(D1.shape, type(D1))
     print(D2.shape, type(D2))
```

```
(2979363,) <class 'numpy.ndarray'>
(84723, 38) <class 'numpy.ndarray'>
```

1)

Generate graphs:

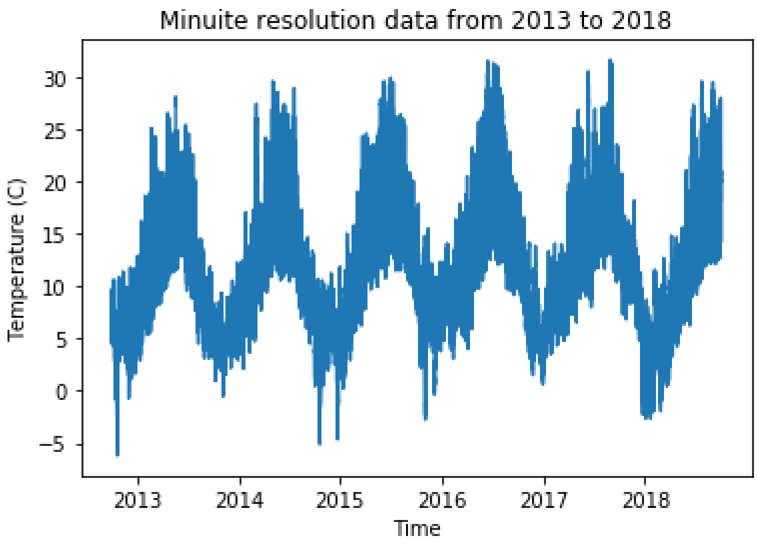
[76]

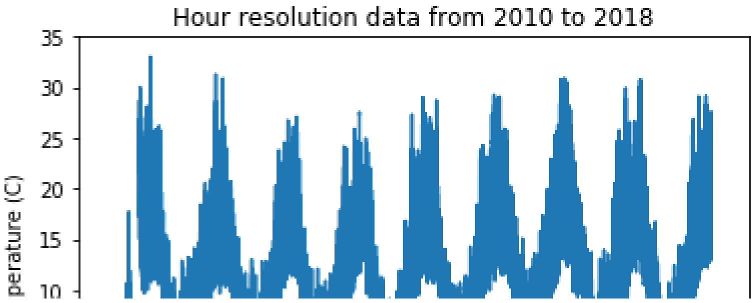
```
date1 = ['2013', '2014', '2015', '2016', '2017', '2018']
date2 = ['2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017',

ypos1 = np.array([0, 1, 2, 3, 4, 5])*len(D1[3:])/6 + len(D1[3:])/24
ypos2 = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])*((D2[84722:,0]-D2[3,0])[(
```

[77]

```
# Temprature
plt.figure(10)
plt.plot(D1[3:])
plt.xlabel(r'Time')
plt.ylabel(r'Temperature (C)')
plt.xticks(ypos1, date1)
plt.title(r'Minuite resolution data from 2013 to 2018')

# Temperature all stataions
# UVic Science Building
# Longitude: 236.691
# Latitude: 48.462
# Column: 36 => Index: 25
plt.figure(11)
plt.plot(D2[3:,0],D2[3:,35])
plt.xlabel(r'Time')
plt.ylabel(r'Temperature (C)')
plt.xticks(ypos2, date2)
plt.title(r'Hour resolution data from 2010 to 2018')
```

```
Text(0.5,1,'Hour resolution data from 2010 to 2018')
```





Calculate averages and standard deviation:

```
[9]  # Find the mean
     d1 = D1[3:]
     d2 = D2[3:,35]

     MnM1 = np.nanmean(d1)
     display(Math(r'\mu_{M}=%.5f' % MnM1))

     HrM1 = np.nanmean(d2)
     display(Math(r'\mu_{H}=%.5f' % HrM1))

     # Find the Standard Deviation
     MnV1 = np.nanvar(d1)
     display(Math(r'\sigma_{M}=%.5f' % MnV1**0.5))

     HrV1 = np.nanvar(d2)
     display(Math(r'\sigma_{H}=%.5f' % HrV1**0.5))
```

$$\mu_M = 11.29531$$

$$\mu_H = 11.19736$$

$$\sigma_M = 5.60895$$

$$\sigma_H = 5.53058$$

The average temperature for the minute resolution data is $\mu_M = 11.29530^\circ C$, with a standard deviation of $\sigma_M = 5.60895^\circ C$.

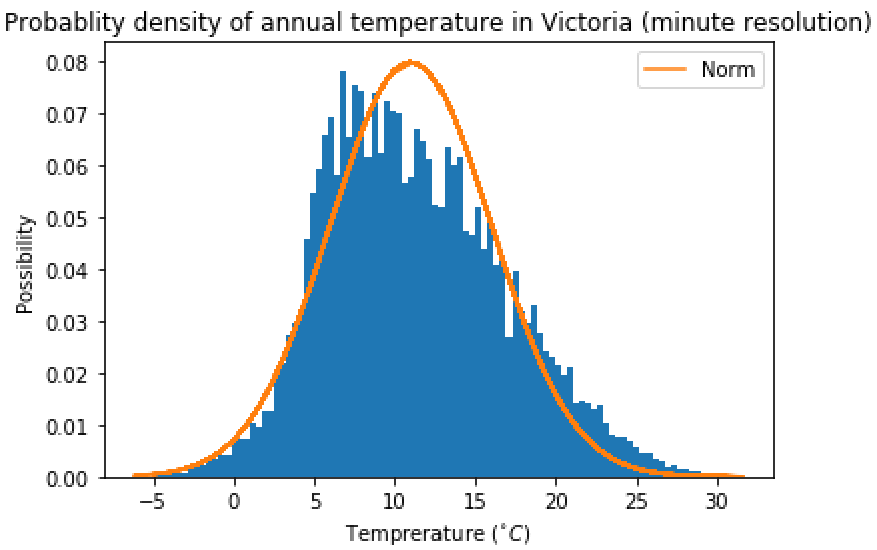
The average temperature for the hourly resolution data is $\mu_H = 11.19736^\circ C$, with a standard deviation of $\sigma_H = 5.53058^\circ C$.

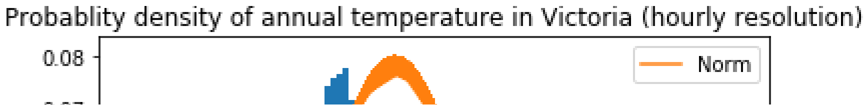
Generate histograms:

```python
[13]  # Generate the histograms, probability density functions, and normal distr
      # Mn_pdf = sci.stats.norm.pdf(d1, int(MnM1), int(MnV1**0.5))
      # Hr_pdf = sci.stats.norm.pdf(d2, int(HrM1), int(HrV1**0.5))
      Mn_norm = norm.pdf(d1, int(MnM1), int(MnV1**0.5))
      Hr_norm = norm.pdf(d2, int(HrM1), int(HrV1**0.5))

      plt.figure(12)
      plt.hist(d1[~np.isnan(d1)], bins=100, density=1)
      # plt.plot(d1, Mn_pdf, label='PDF')
      plt.plot(d1, Mn_norm, label='Norm')
      plt.xlabel(r'Temprerature ($^{\circ} C$)')
      plt.ylabel(r'Possibility')
      plt.title(r'Probablity density of annual temperature in Victoria (minute re
      plt.legend()

      plt.figure(13)
      plt.hist(d2[~np.isnan(d2)], bins=100, density=1)
      # plt.plot(d2, Hr_pdf, label='PDF')
      plt.plot(d2, Hr_norm, label='Norm')
      plt.xlabel(r'Temprerature ($^{\circ} C$)')
      plt.ylabel(r'Possibility')
      plt.title(r'Probablity density of annual temperature in Victoria (hourly re
      plt.legend()
```

```
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-
packages/scipy/stats/_distn_infrastructure.py:876: RuntimeWarning: invalid
value encountered in greater_equal
  return (self.a <= x) & (x <= self.b)
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-
packages/scipy/stats/_distn_infrastructure.py:876: RuntimeWarning: invalid
value encountered in less_equal
  return (self.a <= x) & (x <= self.b)
```

```
<matplotlib.legend.Legend at 0x118fd5048>
```





2) Data from 2012

[33]
```
start = int(len(D2)*3/16 + 3 + len(D2)/72)
fin = int(len(D2)*5/16 + 3 + len(D2)/72)

D3 = D2[start:fin, 35]
```

Calculate average and standard deviation:

[34]
```
# Find the mean
HrM2 = np.nanmean(D3)
display(Math(r'\mu_{H2012}=%.5f' % HrM2))

# Find the Standard Deviation
HrV2 = np.nanvar(D3)
display(Math(r'\sigma_{M2012}=%.5f' % HrV2**0.5))
```

$$\mu_{H2012} = 9.85886$$

$$\sigma_{M2012} = 5.07574$$

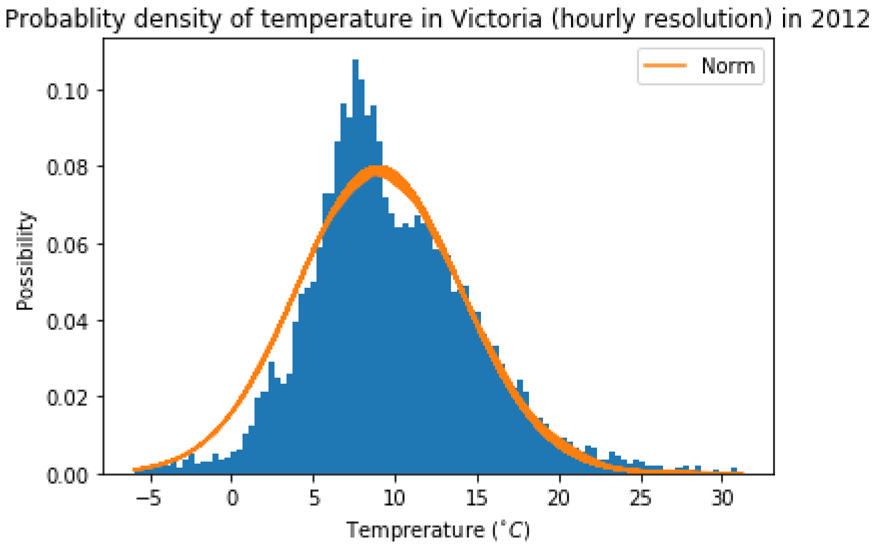
The average temperature for the minute resolution data is $\mu_M = 9.85886^\circ C$, with a standard deviation of $\sigma_M = 5.07574^\circ C$.

Generate histogram:

[35]
```
Hr2_norm = norm.pdf(D3, int(HrM2), int(HrV2**0.5))

plt.figure(15)
plt.hist(D3[~np.isnan(D3)], bins=100, density=1)
# plt.plot(d1, Mn_pdf, label='PDF')
plt.plot(D3, Hr2_norm, label='Norm')
plt.xlabel(r'Temprerature ($^{\circ} C$)')
plt.ylabel(r'Possibility')
plt.title(r'Probablity density of temperature in Victoria (hourly resolut
plt.legend()
```

```
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-
packages/scipy/stats/_distn_infrastructure.py:876: RuntimeWarning: invalid
value encountered in greater_equal
  return (self.a <= x) & (x <= self.b)
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-
packages/scipy/stats/_distn_infrastructure.py:876: RuntimeWarning: invalid
value encountered in less_equal
  return (self.a <= x) & (x <= self.b)
```

```
<matplotlib.legend.Legend at 0x117faef60>
```



Discussion:

From the three probability density graphs, we can see the temperature in Victoria tends toward being cold, as the probability distribution displayed more weight towards the right side of the normal distribution. Average temperature in 2012 was colder than usual, with an average of $9.85886\,^{\circ}C$ compared to the averages of $11.29530\,^{\circ}C$ and $11.19736\,^{\circ}C$ from the minute and hour resolution graphs, respectively. Despite being colder, temperatures in 2012 were more stable with a standard deviation of $5.07574\,^{\circ}C$ compared to that of $5.60895\,^{\circ}C$ and $5.53058\,^{\circ}C$.