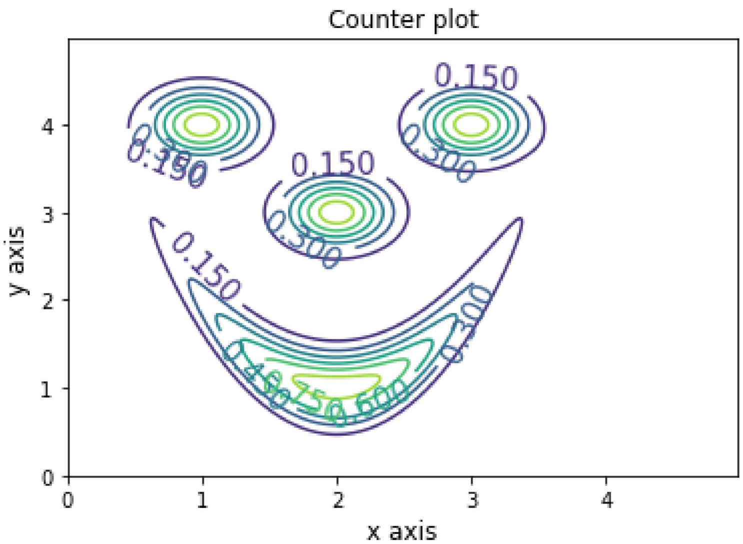
In your experiments you found that the phenomenon you are measuring is described by the following equation:

Z = np.exp(-((X-1)**2+(Y-4)**2)/0.15) +
np.exp(-((X-3)**2+(Y-4)**2)/0.15) +
np.exp(-((X-2)**2+(Y-3)**2)/0.15) +
np.exp(-(X-2)**2) * np.exp(-(Y - ((X-2)**2+1))*2/0.15)* *Note: For your convenience, the equation is written with np.exp referring to the exponentiation function ex from the numpy package name shortened to np for convenience. Also,* * is the power function. The '\' is because python needs it for equations that continue over multiple lines - not necessary in other languages. Also, to be complete, be sure to show the contour plot for the entire range for which this function has interesting features to observe.

1. Contour plot 1). Make a contour plot. Make sure to add labels in the plot or a legend for colors on the contours. You can choose either a filled contour plot or colored lines, your choice. 2). Do this for an additional color mapping: e.g. hot/cold or black/white.

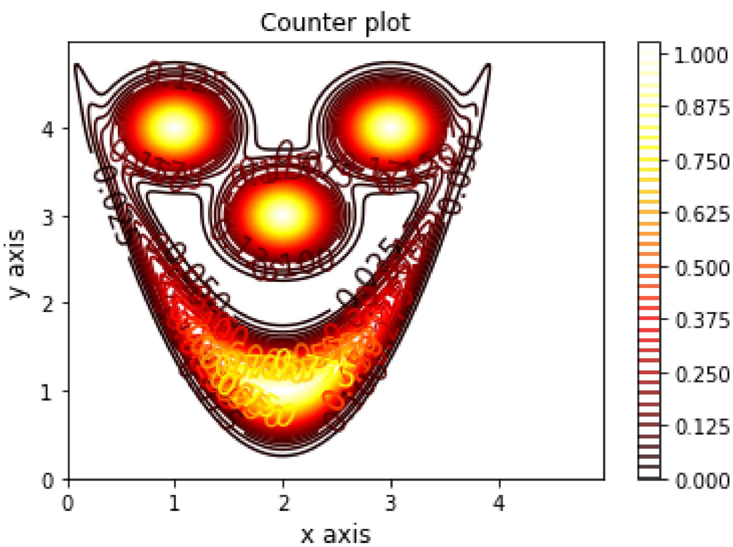
In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm
```

In [4]:
```
d=0.020
a=np.arange(0,5.0,d)
b=np.arange(0,5.0,d)
J,K=np.meshgrid(a,b)
L = np.exp(-((J-1)**2+(K-4)**2)/0.15) +\
np.exp(-((J-3)**2+(K-4)**2)/0.15) +\
np.exp(-((J-2)**2+(K-3)**2)/0.15) +\
np.exp(-(J-2)**2)*np.exp(-(K - ((J-2)**2+1))**2/0.15)
fig,ax=plt.subplots(1,1)
ab=ax.contour(J,K,L)
ax.clabel(ab,inline=True,fontsize=15)
ax.set_title("Counter plot")
plt.xlabel("x axis",fontsize=12)
plt.ylabel("y axis",fontsize=12)
plt.show()
```



2.color mapping

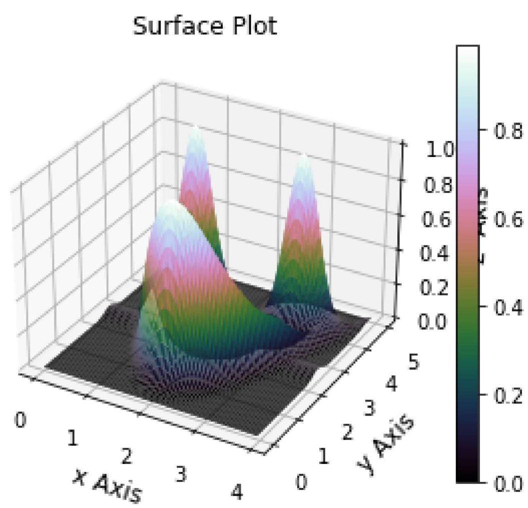
In [6]:
```
d=0.020
a=np.arange(0,5.0,d)
b=np.arange(0,5.0,d)
J,K=np.meshgrid(a,b)
L = np.exp(-((J-1)**2+(K-4)**2)/0.15) +\
np.exp(-((J-3)**2+(K-4)**2)/0.15) +\
np.exp(-((J-2)**2+(K-3)**2)/0.15) +\
np.exp(-(J-2)**2)*np.exp(-(K - ((J-2)**2+1))**2/0.15)
fig,ax=plt.subplots(1,1)
ab=ax.contour(J,K,L,40,cmap=plt.cm.hot)
plt.colorbar(ab)
ax.clabel(ab,inline=True,fontsize=15)
ax.set_title("Counter plot")
plt.xlabel("x axis",fontsize=12)
plt.ylabel("y axis",fontsize=12)
plt.show()
```



2. Surface plots (or mesh plots) 1). Using the same data set as before, create a surface plot. Also be sure to choose an appropriate color mapping to help in interpretation. If you can't make a surface plot, a mesh plot (where the surface is not filled in) will suffice. 2). Generate at least one additional viewpoint of the surface that may also be helpful in providing insights.

1.surface plot

In [9]:
```
from mpl_toolkits import mplot3d
d=0.05
a=np.arange(0,4.0,d)
b=np.arange(0,5.0,d)
J,K=np.meshgrid(a,b)
L=np.exp(-((J-1)**2+(K-4)**2)/0.15) + \
np.exp(-((J-3)**2+(K-4)**2)/0.15) + \
np.exp(-((K-2)**2+(K-3)**2)/0.15) + \
np.exp(-(J-2)**2) * np.exp(-(K-((J-2)**2+1))**2/0.15)
ax = plt.axes(projection='3d')
ab=ax.plot_surface(J,K,L,cstride=1,rstride=1,cmap='cubehelix')
plt.colorbar(ab)
ax.set_title('Surface Plot');
plt.xlabel("x Axis",fontsize =12)
plt.ylabel("y Axis",fontsize =12)
ax.set_zlabel("z  Axis",fontsize =12)
plt.show()
```



2 Additional view point

In [12]:
```
d=0.05
a=np.arange(0,4.0,d)
b=np.arange(0,5.0,d)
J,K=np.meshgrid(a,b)
L=np.exp(-((J-1)**2+(K-4)**2)/0.15) + \
np.exp(-((J-3)**2+(K-4)**2)/0.15) + \
np.exp(-((K-2)**2+(K-3)**2)/0.15) + \
np.exp(-(J-2)**2) * np.exp(-(K-((J-2)**2+1))**2/0.15)
ax = plt.axes(projection='3d')
ab=ax.plot_surface(J,K,L,cstride=1,rstride=1,cmap='viridis')
plt.colorbar(ab)
ax.set_title('Surface Plot');
plt.xlabel("x Axis",fontsize =12)
plt.ylabel("y Axis",fontsize =12)
ax.set_zlabel("z  Axis",fontsize =12)
ax.view_init(50,20)
plt.show()
```