Exercise 05

190713X - L.H.N.WIJEWARDENA

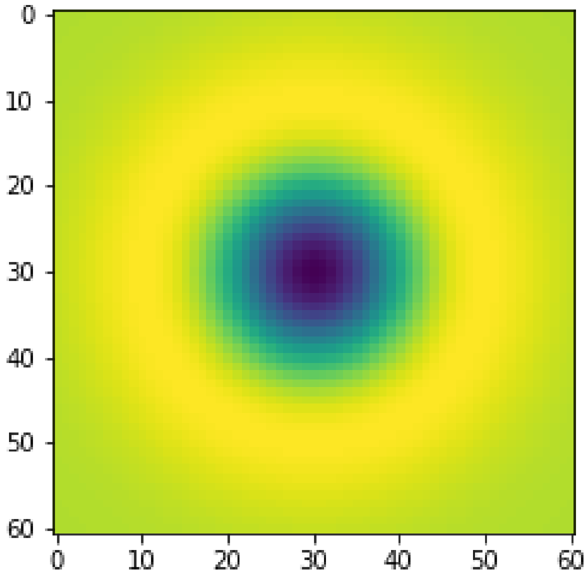
Question 1

In [ ]:

```
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt

sigma = 10
hw = 3*sigma
X, Y = np.meshgrid(np.arange(-hw,hw+1,1),np.arange(-hw,hw+1,1))
log = 1/(2*np.pi*sigma**2)*(X**2/(sigma**2)+Y**2/(sigma**2)-2)*np.exp(

plt.imshow(log)
```

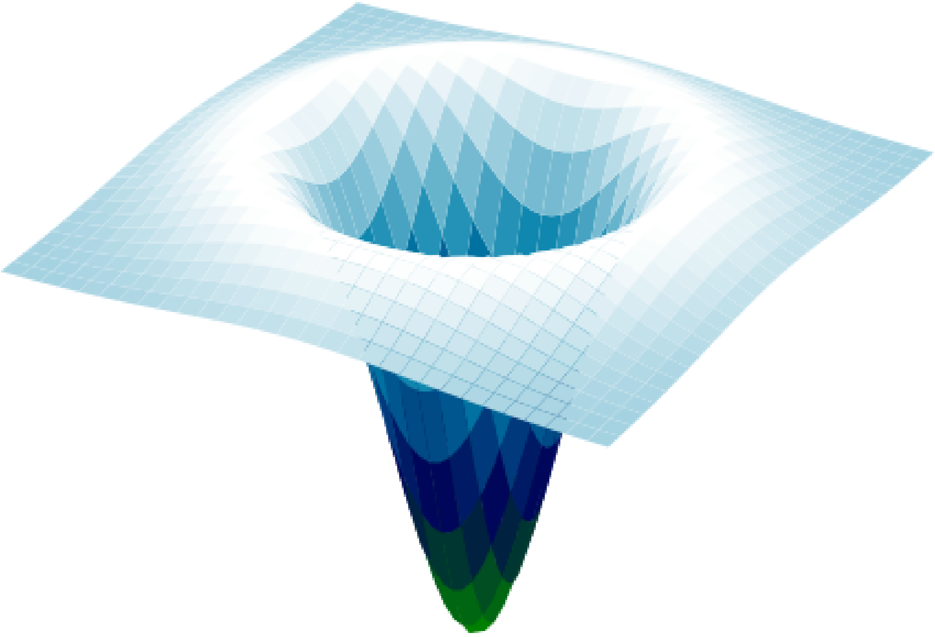
Out[ ]:

```
<matplotlib.image.AxesImage at 0x2615b6cf040>
```



In [ ]:

```
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter

fig = plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')

surf = ax.plot_surface(X, Y, log, cmap=cm.ocean, linewidth=0, antialia
ax.zaxis.set_major_locator(LinearLocator(10))
ax.zaxis.set_major_formatter(FormatStrFormatter('%.02f'))
plt.axis('off')
plt.show()
```



Question 2

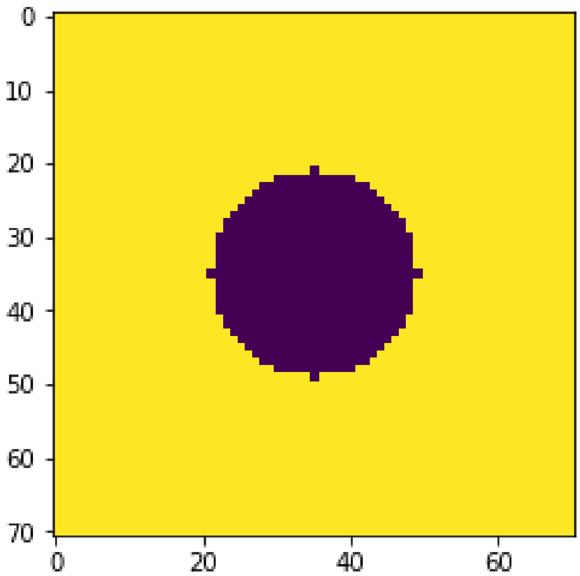
In [ ]:

```
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt

w,h =71,71
hw = w//2
hh = h//2

f = np.ones((h,w), dtype=np.float32)*255
X, Y = np.meshgrid(np.arange(-hh,hh+1,1),np.arange(-hw,hw+1,1))
```

```
r = w//5
f*= X**2 +Y**2 > r**2

plt.imshow(f)
```

Out[ ]:

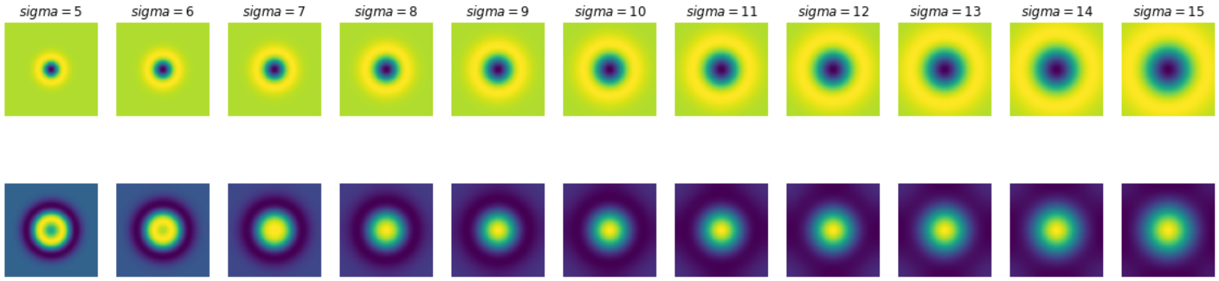
```
<matplotlib.image.AxesImage at 0x1b4b8168730>
```



In [ ]:

```
s =11
fig, ax = plt.subplots(2, s, figsize=(20,5))
scale_space = np.empty((h,w,s),dtype=np.float32)
sigmas = np.arange(5,16,1)
for i,sigma in enumerate(np.arange(5,16,1)):
    log_hw =3*np.max(sigmas)
    X, Y = np.meshgrid(np.arange(-hw,hw+1,1),np.arange(-hw,hw+1,1))
    log = 1/(2*np.pi*sigma**2)*(X**2/(sigma**2) + Y**2/(sigma**2)-2)*r
    f_log = cv.filter2D(f,-1,log)
    scale_space[:,:,i]=f_log
    ax[0,i].imshow(log)
    ax[0,i].axis('off')
    ax[0,i].set_title(r'$sigma={}$'.format(sigma))
    ax[1,i].imshow(f_log)
    ax[1,i].axis('off')

indices = np.unravel_index(np.argmax(scale_space, axis=None),scale_spa
print(indices)
print(sigmas[indices[2]])
```

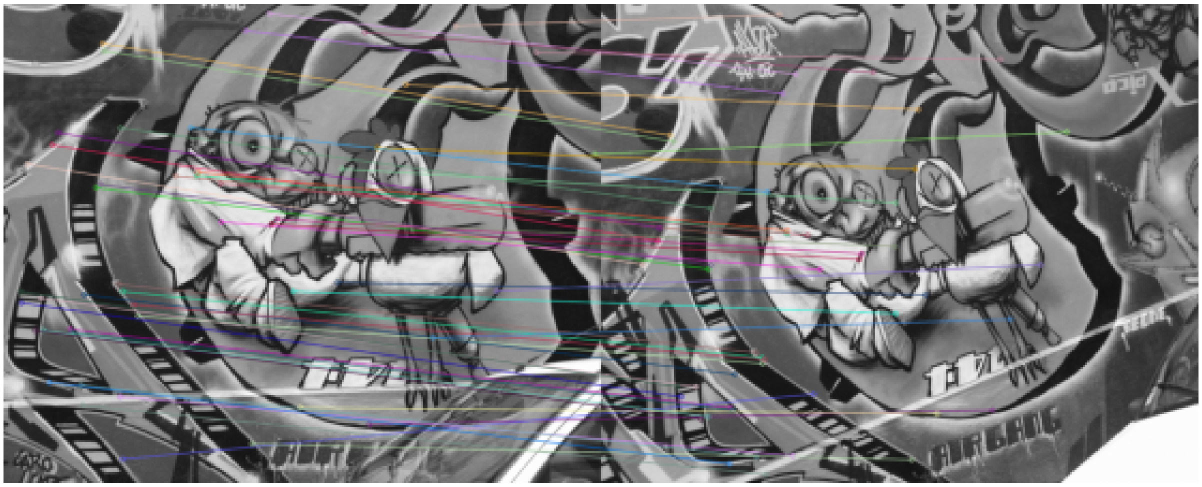
```
(35, 35, 5)
10
```



Question 3

In [ ]:

```
img1 = cv.imread('img1.ppm')
img2 = cv.imread('img2.ppm')

img1 = cv.cvtColor(img1, cv.COLOR_BGR2GRAY)
img2 = cv.cvtColor(img2, cv.COLOR_BGR2GRAY)

sift = cv.SIFT_create()
keypoint_1, descriptors_1 = sift.detectAndCompute(img1,None)
keypoint_2, descriptors_2 = sift.detectAndCompute(img2,None)

bf = cv.BFMatcher(cv.NORM_L1, crossCheck=True)

matches = bf.match(descriptors_1,descriptors_2)
matches = sorted(matches, key = lambda x:x.distance)

fig, ax = plt.subplots(figsize = (10,10))
ax.axis('off')
img3 = cv.drawMatches(img1, keypoint_1, img2, keypoint_2, matches[:50]
plt.imshow(img3)
plt.show()
```



Question 4

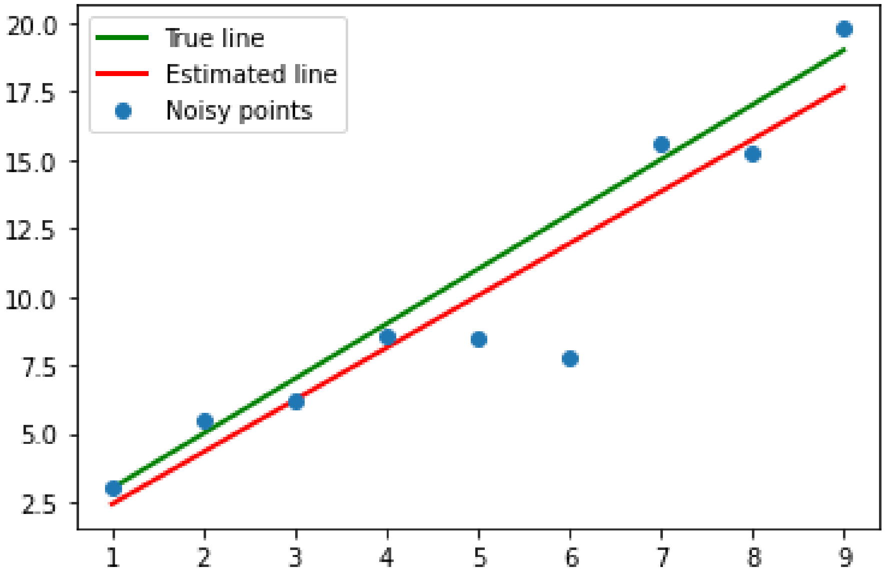
In [ ]:

```
m = 2 # Line equation : y = m*x + c . m is the slope . c is th
```

```
c = 1
x = np.arange (1 ,10 , 1)
np.random.seed(45)
noise = 2.*np.random.randn(len(x))
o = np.zeros(x.shape)
# o [=1] = 20
y = m*x + c + noise + o

n = len(x)
X = np.concatenate([x.reshape(n,1), np.ones((n,1))], axis = 1)
B = np.linalg.pinv(X.T @ X) @ X.T @ y
mstar = B[0]
cstar = B[1]

plt.plot([x[0],x[-1]],[m*x[0] + c, m*x[-1] + c],color = 'g', linewidth
plt.plot([x[0],x[-1]],[mstar*x[0] + cstar, mstar*x[-1] + cstar],color
plt.plot(x,y, 'o', label = 'Noisy points')
plt.legend()
```

Out[ ]:

```
<matplotlib.legend.Legend at 0x2d0de182880>
```



Question 5

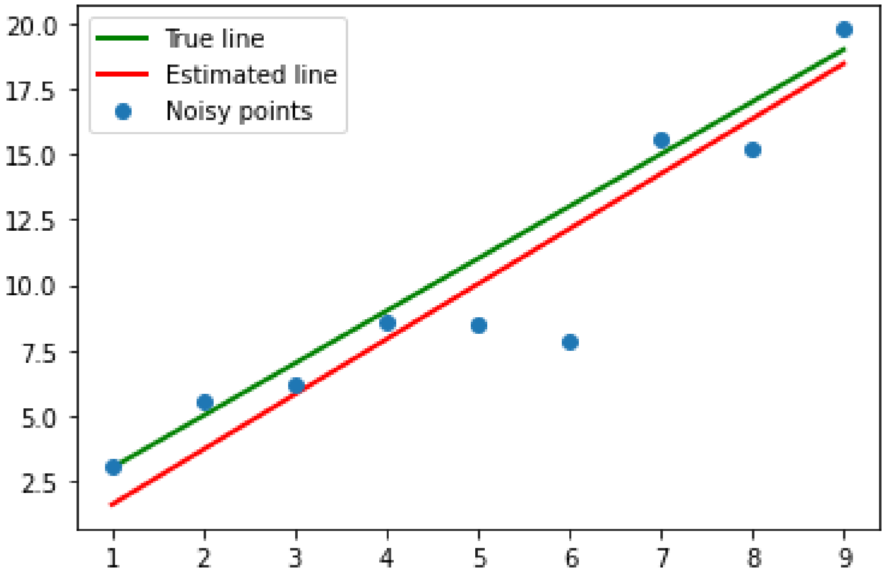
In [ ]:

```
m = 2 # Line equa t ion : y = m*x + c . m i s the s l o p e . c i s th
c = 1
x = np.arange (1 ,10 , 1)
np.random.seed(45)
noise = 2.*np.random.randn(len(x))
o = np.zeros(x.shape)
# o [=1] = 20
y = m*x + c + noise + o

u11 = np.sum((x-np.mean(x))**2)
```

```
u12 = np.sum((x-np.mean(x))*(y-np.mean(y)))
u21 = u12
u22 = np.sum((y-np.mean(y))**2)

U =np.array([[u11, u12],[u21,u22]])
W ,V = np.linalg.eig(U)
ev_for_smallest = V[:,np.argmin(W)]

a = ev_for_smallest[0]
b = ev_for_smallest[1]
d = a*np.mean(x) + b*np.mean(y)

mstar = -a/b
cstar = d/b

plt.plot([x[0],x[-1]],[m*x[0] + c, m*x[-1] + c],color = 'g', linewidth
plt.plot([x[0],x[-1]],[mstar*x[0] + cstar, mstar*x[-1] + cstar],color
plt.plot(x,y, 'o', label = 'Noisy points')
plt.legend()
```

Out[ ]:

```
<matplotlib.legend.Legend at 0x2d0dbe04f70>
```


True line
Estimated line
Noisy points