Exercise 5

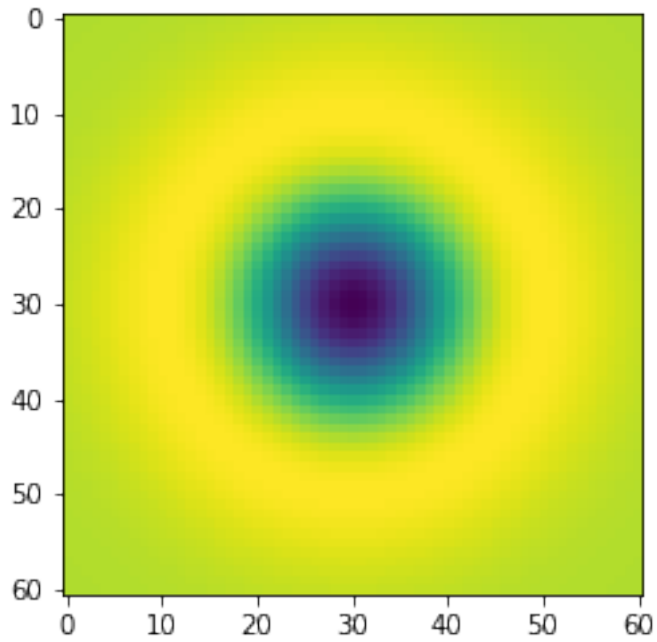
March 9, 2022

0.0.1 Name: Chathuranga M.M.P.

0.0.2 Index No: 190108X

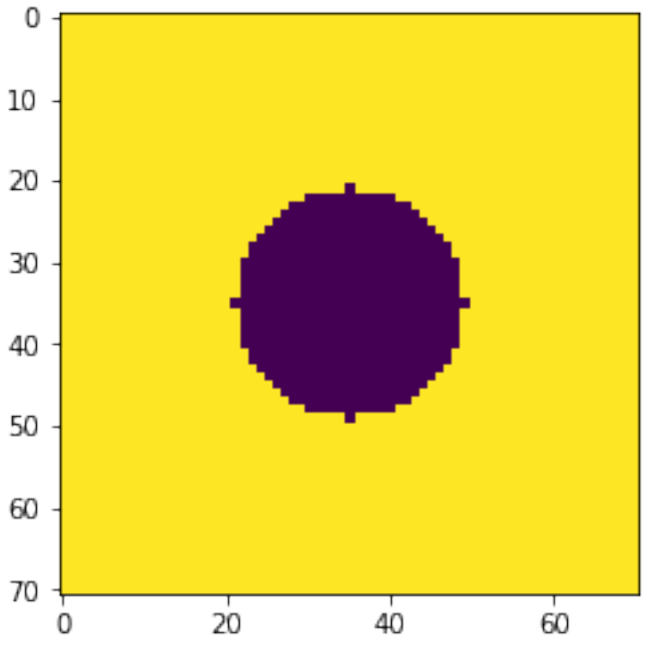
[7]:
```
# Question 1

import matplotlib.pyplot as plt
import numpy as np
import math
%matplotlib inline

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

[7]:
```
<matplotlib.image.AxesImage at 0x1b4c1f0a9d0>
```



[10]:
```
# Question 2

import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline

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

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

r=w//5 # 141
f*=X**2+Y**2>r**2

plt.imshow(f)
```

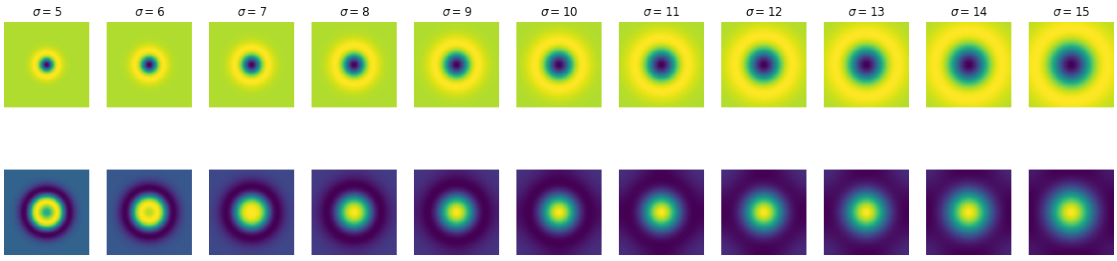
[10]:
```
<matplotlib.image.AxesImage at 0x1b4c213c8b0>
```



[22]:
```
s=11
fig,ax=plt.subplots(2,s,figsize=(20,5))
scalespace=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*sigma
    X,Y=np.meshgrid(np.arange(-hw,hw+1,1),np.arange(-hw,hw+1,1))
    log=1/(2*math.pi*sigma**2)*(X**2/sigma**2+Y**2/sigma**2-2)*np.
 ↪exp(-(X**2+Y**2)/(2*sigma**2))
    f_log=cv.filter2D(f,-1,log)
    scalespace[:,:,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(scalespace,axis=None),scalespace.shape)
print(indices)
print(sigmas[indices[2]])
```

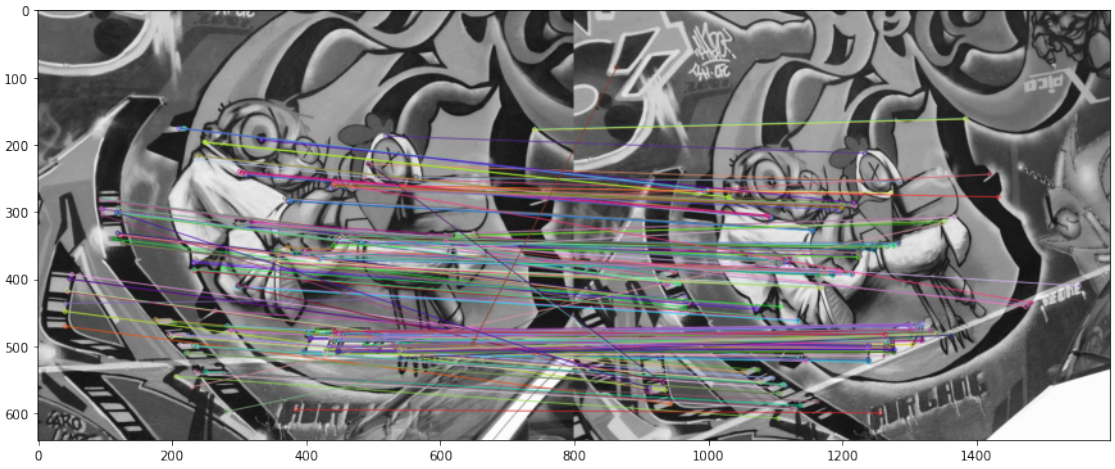
```
(35, 35, 5)
10
```



[22]:
```
# Question 3

import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline

img1=cv.imread(r'img1.ppm',cv.IMREAD_GRAYSCALE)
assert img1 is not None
img2=cv.imread(r'img2.ppm',cv.IMREAD_GRAYSCALE)
assert img2 is not None

orb_detect = cv.ORB_create()

key_p1, des1 = orb_detect.detectAndCompute(img1,None)
key_p2, des2 = orb_detect.detectAndCompute(img2,None)
```

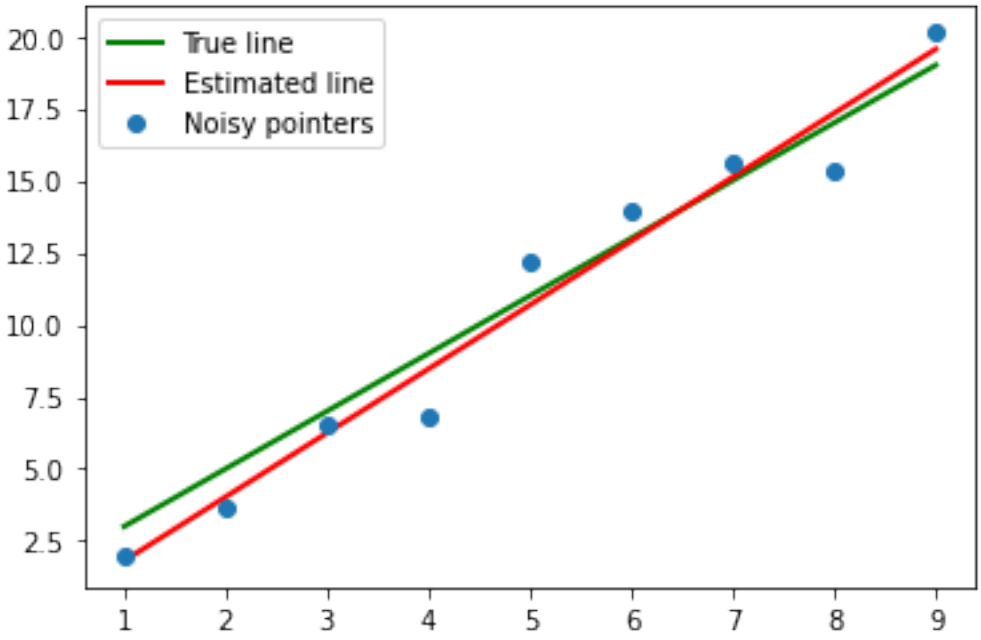
```
bfmatcher = cv.BFMatcher(cv.NORM_HAMMING, crossCheck=True)

matches = bfmatcher.match(des1,des2)
matches = sorted(matches, key = lambda x:x.distance)
img3 = cv.drawMatches(img1,key_p1,img2,key_p2,matches[:],None,flags=cv.
 ↪DrawMatchesFlags_NOT_DRAW_SINGLE_POINTS)

fig,ax=plt.subplots(1,1,figsize=(15,15))
ax.imshow(img3)
```

[22]: <matplotlib.image.AxesImage at 0x219923b5790>



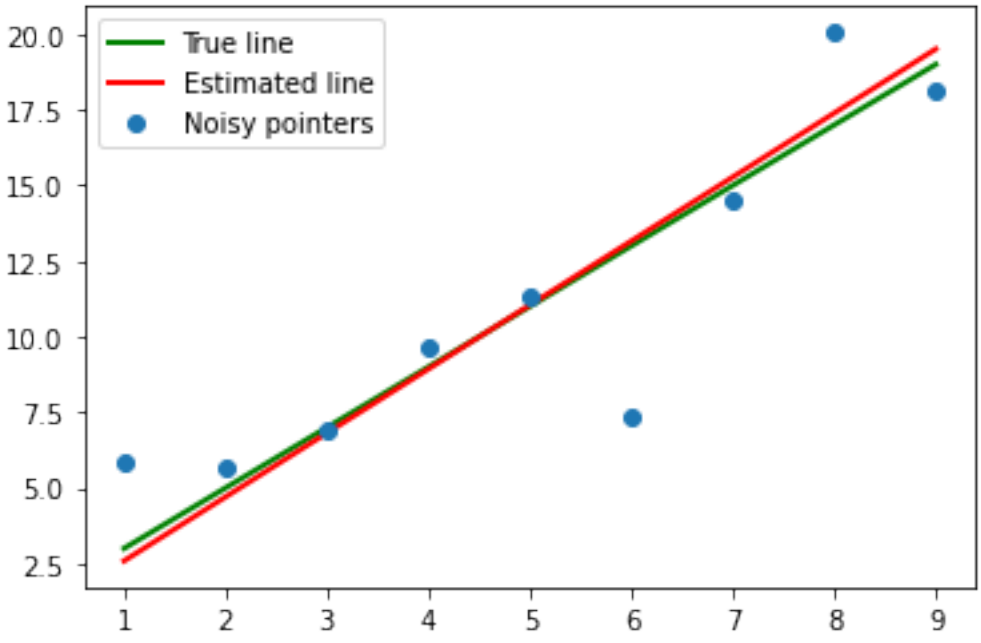
[5]:
```
# Question 4

import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline

m = 2 # Line equation : y = m*x + c . m i s the s lope . c i s the int e r c ep␣
 ↪t .
c = 1
x = np . arange (1 ,10 , 1)
n = 2.*np . random. randn ( len ( x ) )
o = np . zeros ( x . shape )
# o[=1] = 20
y = m*x + c + n + 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=2,label=r'True␣
↪line')
plt.
↪plot([x[0],x[-1]],[mstar*x[0]+cstar,mstar*x[-1]+cstar],color='r',linewidth=2,label=r'Estima
↪line')
plt.plot(x,y,'o',label='Noisy pointers')
plt.legend()
plt.show()
```



[26]:
```
# Question 5

import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline

m = 2 # Line equation : y = m*x + c . m i s the s lope . c i s the int e r c ep␣
↪t .
c = 1
x = np . arange (1 ,10 , 1)
n = 2.*np . random. randn ( len ( x ) )
o = np . zeros ( x . shape )
```

```
# o[=1] = 20
y = m*x + c + n + o

n=len(x)

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]])
M,V=np.linalg.eig(U)
ev_corresponding_to_smallest_ev=V[:,np.argmin(M)]

a=ev_corresponding_to_smallest_ev[0]
b=ev_corresponding_to_smallest_ev[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=2,label=r'True␣
 ↪line')
plt.
 ↪plot([x[0],x[-1]],[mstar*x[0]+cstar,mstar*x[-1]+cstar],color='r',linewidth=2,label=r'Estima
 ↪line')
plt.plot(x,y,'o',label='Noisy pointers')
plt.legend()
plt.show()
```


True line
Estimated line
Noisy pointers
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
1
2
3
4
5
6
7
8
9