In [1]:

```
# Library
import numpy as np
import matplotlib.pyplot as plt
```

In [15]:

```
#(1) Generate arrays for A matrix and b vector using NumPy library
A = np.array([[1, 1, -2],[3, 3, -5],[3, 2, -10], [1,1,-7], [-4,-4,11]])
b = np.array([2, 7, 2, -3, -4])

print('\n','1. A is')
print(A)

print('\n','2. b is')
print(b)
```

```
 1. A is
[[  1   1  -2]
 [  3   3  -5]
 [  3   2 -10]
 [  1   1  -7]
 [ -4  -4  11]]

 2. b is
[ 2  7  2 -3 -4]
```

In [16]:

```
#(2) Transpose A and apply matrix multiplication with the original matrix(A'A)
transA = A.T
print('\n','1. Transpose A is')
print(transA)
ATA = np.dot(transA,A)
print('\n','2. Trans(A) * A is')
print(ATA)
```

```
 1. Transpose A is
[[  1   3   3   1  -4]
 [  1   3   2   1  -4]
 [ -2  -5 -10  -7  11]]

 2. Trans(A) * A is
[[ 36  33 -98]
 [ 33  31 -88]
 [-98 -88 299]]
```

In [25]:

```
#(3) Calculate x as the solution of A'Ax = A'b
# x = inv(A'A)*A'*b
ATAinv = np.linalg.inv(ATA)
print('\n','1. inverse(trans(A)*A) is')
print(ATAinv)
B = np.dot(transA,b)
print('\n','2. trans(A)*b is')
print(B)
x_ = np.linalg.solve(ATA, B)
print(x_)
x = np.dot(ATAinv,B)
print('\n','3. the solution is')
print(x)
```

```
 1. inverse(trans(A)*A) is
[[ 2.03604806 -1.65954606  0.17890521]
 [-1.65954606  1.54873164 -0.08811749]
 [ 0.17890521 -0.08811749  0.03604806]]

 2. trans(A)*b is
[ 42  40 -82]
[ 4.46194927 -0.52603471  1.03337784]

 3. the solution is
[ 4.46194927 -0.52603471  1.03337784]
```

In [27]:

```
#(4) Calculate the norm of Ax - b with line-by-line coding and using Numpy library
diffs = []
diff_mat = np.dot(A, x) - b

for i in range(5):
    diff = np.dot(A[i],x_) - b[i]
    diffs.append(diff)
    print('%dth row Ax-b is %0.2f' %(i, diffs[-1]))
print('\n numpy norm option: 2-norm(defalt)')
print(' Ax - b is', diffs)
norm1 = np.linalg.norm(diffs,1)
norm2 = np.linalg.norm(diffs,2)
norminf = np.linalg.norm(diffs,np.inf)

print('\n 1-norm is',norm1)
print(' 2-norm is',norm2)
print(' infinity-norm is',norminf)
```

```
[-1.30841121e-01 -3.59145527e-01  3.55271368e-14 -2.97730307e-01
 -3.76502003e-01]
0.37650200267022654
0th row Ax-b is -0.13
1th row Ax-b is -0.36
2th row Ax-b is 0.00
3th row Ax-b is -0.30
4th row Ax-b is -0.38

 numpy norm option: 2-norm(defalt)
 Ax - b is [-0.1308411214953269, -0.3591455273698241, 1.2434497875801753e-14, -0.29773030707611126, -0.3765020026
702217]

 1-norm is 1.1642189586114964
 2-norm is 0.6135975901763525
 infinity-norm is 0.3765020026702217
```

In [7]:

```
#2. Drawing Graphs
from mpl_toolkits.axes_grid1 import make_axes_locatable
# 1) parameter setting
a = 20
b = 0.2
c = 2*np.pi
bb = -4
ub = 4
N  = 100

x = np.arange(bb,ub,1/N)
y = np.arange(bb,ub,1/N)
X, Y = np.meshgrid(x, y)
```

In [8]:

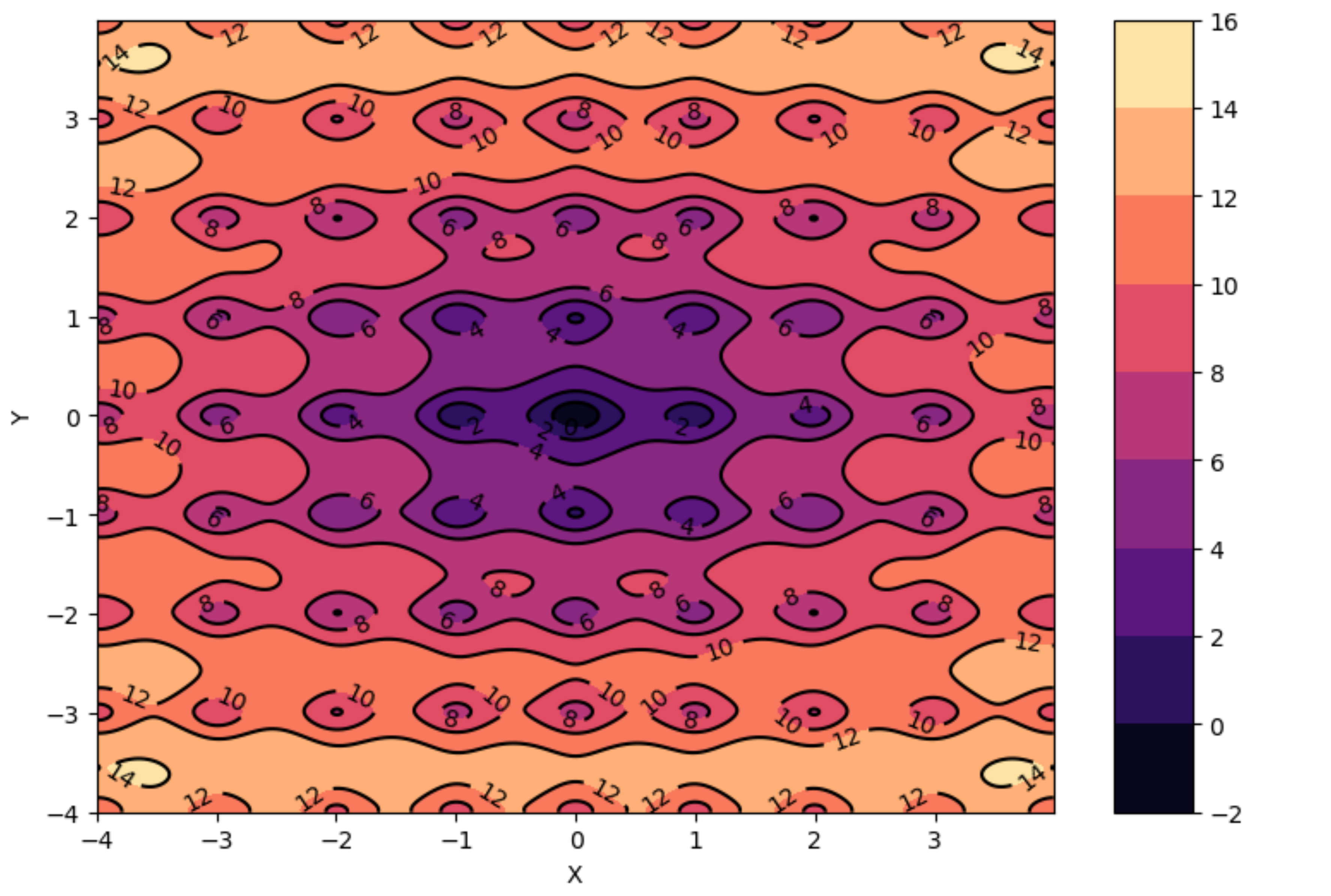
```
# 2) Setting the function
Z1 = -a * np.exp(-b * np.sqrt(0.5 * X**2 + Y**2))
Z2 = np.exp(0.5 * np.cos(c*X) + np.cos(c*Y)) - a - np.exp(1)
Z = Z1 - Z2
```

In [39]:

```
fig = plt.figure(1, figsize=(9,6))
cont = plt.contourf(X,Y,Z, cmap='magma')
plt.xlabel('X')
plt.ylabel('Y')
cs=plt.contour(X,Y,Z,colors='k')
fig.colorbar(cont, shrink=1, aspect=10)
plt.clabel(cs)
```

Out[39]:

```
<a list of 99 text.Text objects>
```



In [10]:

```
z1 = -a * np.exp(-b * np.sqrt(0.5 * x**2 + y**2))
z2 = np.exp(0.5 * np.cos(c*x) + np.cos(c*y)) - a - np.exp(1)
z = z1 - z2
```

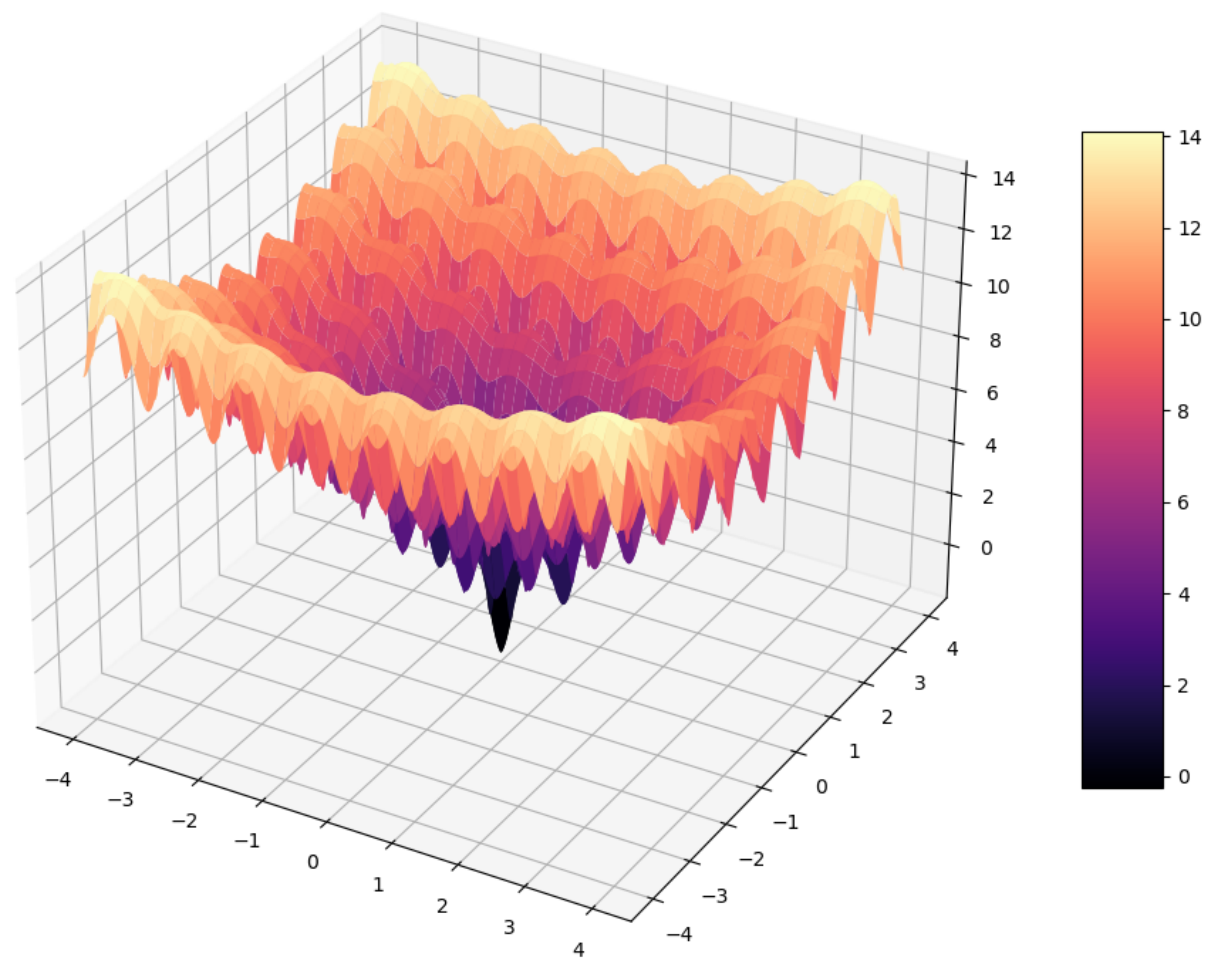
In [41]:

```
from mpl_toolkits.mplot3d import Axes3D
fig= plt.figure(figsize=(12, 12))
ax=fig.add_subplot(111, projection='3d')
surf=ax.plot_surface(X,Y,Z, cmap='magma')

fig.colorbar(surf, shrink=0.5, aspect=8)
plt.tight_layout
```

Out[41]:

```
<function matplotlib.pyplot.tight_layout(*, pad=1.08, h_pad=None, w_pad=None, rect=None)>
```



In [12]:

```
ax.scatter(X,Y,Z,cmap='jet_r')
plt.show()
```