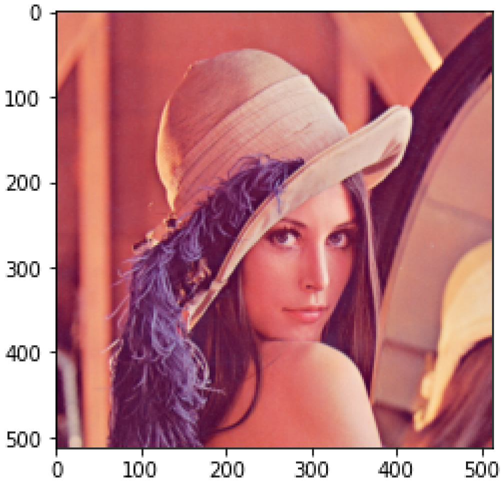
DIP Assignment 06

In [320]:

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

In [321]:

```
img = plt.imread("lena_color.tiff")
plt.imshow(img, cmap=plt.cm.bone)
plt.show()
img.shape
```



Out[321]:

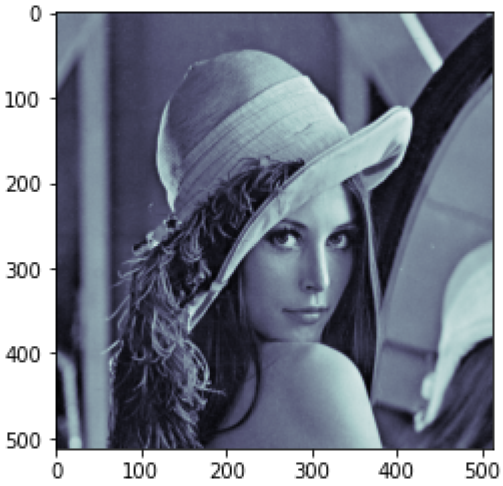
```
(512, 512, 3)
```

In [322]:

```
image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

In [323]:

```
plt.imshow(image, cmap=plt.cm.bone)
plt.show()
image.shape
```



Out[323]:

```
(512, 512)
```

In [324]:

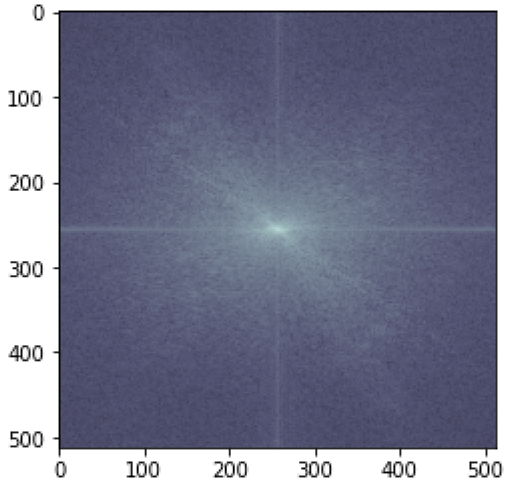
```
imgr , imgc = image.shape
```

In [325]:

```
dft = cv2.dft(np.float32(image),flags = cv2.DFT_COMPLEX_OUTPUT)
imageft = np.fft.fftshift(dft)
magspec = 20*np.log(cv2.magnitude(imageft[:,:,0],imageft[:,:,1]))
```

In [326]:

```
plt.imshow(magspec, cmap=plt.cm.bone)
plt.show()
image.shape
```



Out[326]:

```
(512, 512)
```

Frequency Domain Smoothing Filters

In [327]:

```
D0 = 20
```

In [328]:

```
def Euclidean(imgr,imgc,u,v):
    D_uv = math.sqrt((u-imgr/2)**2 + (v-imgc/2)**2)

    return D_uv
```

#1 Ideal Low Pass Filter

In [329]:

```
def ILPFTransFunc(D0,D_uv):
    if D_uv <= D0:
        return 1
    else:
        return 0
```

In [330]:

```
ilpf = np.zeros((imgr,imgc,2))
ms = np.zeros((imgr,imgc,2))
```

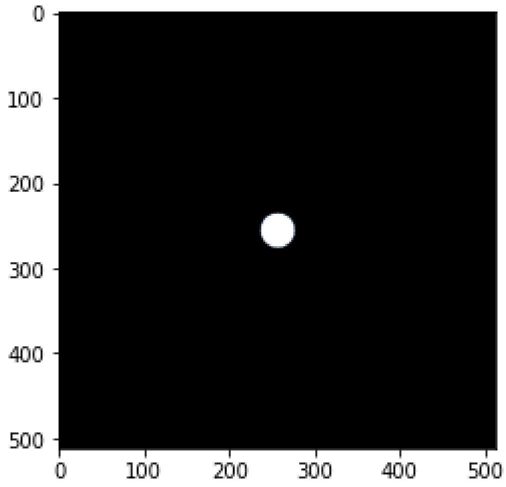
In [331]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = ILPFTransFunc(D0,D_uv)

        ilpf[u][v] = H_uv
```

In [332]:

```
ms = cv2.magnitude(ilpf[:,:,0],ilpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

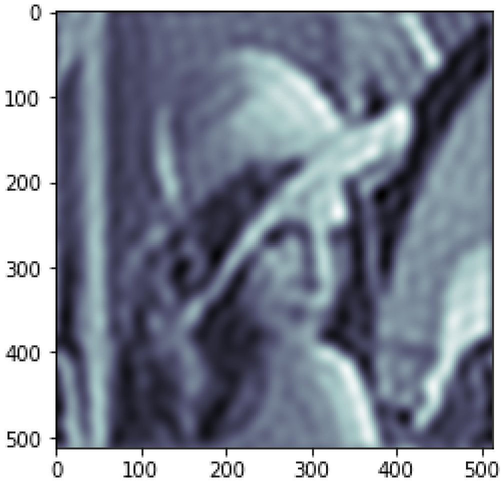


In [333]:

```
fshift = imageft*ilpf
f_ishift = np.fft.ifftshift(fshift)
imgilpf = cv2.idft(f_ishift)
imgilpf = cv2.magnitude(imgilpf[:,:,0],imgilpf[:,:,1])
```

In [334]:

```
plt.imshow(imgilpf, cmap=plt.cm.bone)
plt.show()
```



#2 Butterworth Low Pass Filter

In [335]:

```
def BLPFTransFunc(D0,D_uv,n):
    return (1/(1+(D_uv/D0)**(2*n)))
```

In [336]:

```
blpf = np.zeros((imgr,imgc,2))
ms = np.zeros((imgr,imgc,2))
```

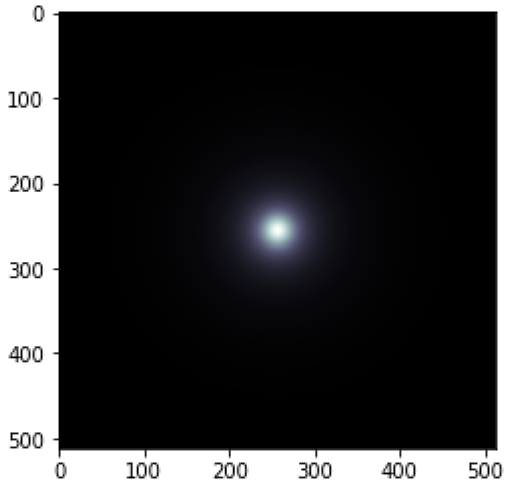
In [337]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = BLPFTransFunc(D0,D_uv,1)

        blpf[u][v] = H_uv
```

In [338]:

```
ms = cv2.magnitude(blpf[:,:,0],blpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

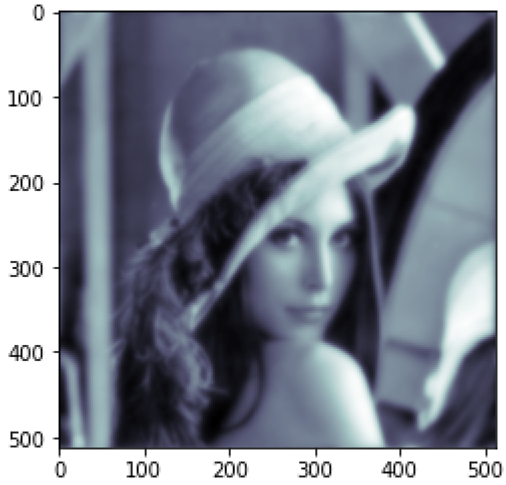


In [339]:

```
fshift = imageft*blpf
f_ishift = np.fft.ifftshift(fshift)
imgblpf = cv2.idft(f_ishift)
imgblpf = cv2.magnitude(imgblpf[:,:,0],imgblpf[:,:,1])
```

In [340]:

```
plt.imshow(imgblpf, cmap=plt.cm.bone)
plt.show()
```



#3 Gaussian Low Pass Filter

In [341]:

```
def GLPFTransFunc(D0,D_uv):
    return (math.exp(-1*((D_uv)**2)/(2*(D0**2))))
```

In [342]:

```
glpf = np.zeros((imgr,imgc,2))
ms = np.zeros((imgr,imgc,2))
```

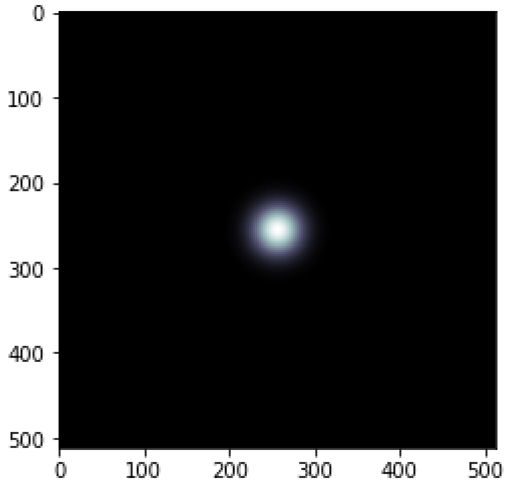
In [343]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GLPFTransFunc(D0,D_uv)

        glpf[u][v] = H_uv
```

In [344]:

```
ms = cv2.magnitude(glpf[:,:,0],glpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

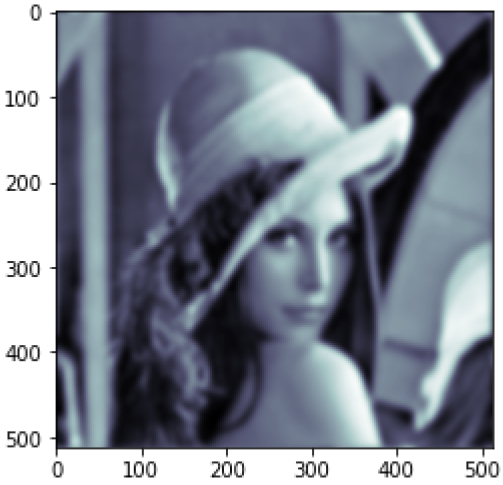


In [345]:

```
fshift = imageft*glpf
f_ishift = np.fft.ifftshift(fshift)
imgglpf = cv2.idft(f_ishift)
imgglpf = cv2.magnitude(imgglpf[:,:,0],imgglpf[:,:,1])
```

In [346]:

```
plt.imshow(imgglpf, cmap=plt.cm.bone)
plt.show()
```



For the given cut-off frequency value D0 = 20, We clearly see the smoothing quality of Gaussian is the best, followed by Butterworth and finally Ideal Low Pass Filter

Ringing Effect of Ideal Low Pass Filter

In [347]:

```
D0 = 25
```

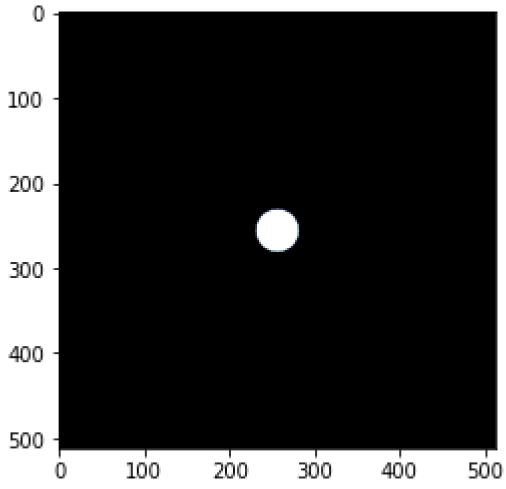
In [348]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = ILPFTransFunc(D0,D_uv)

        ilpf[u][v] = H_uv
```

In [349]:

```
ms = cv2.magnitude(ilpf[:,:,0],ilpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

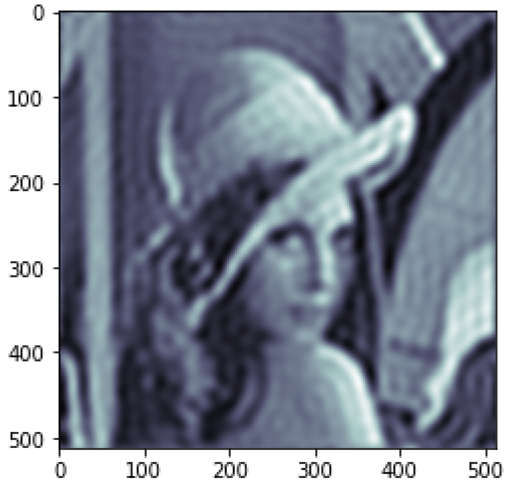


In [350]:

```
fshift = imageft*ilpf
f_ishift = np.fft.ifftshift(fshift)
imgilpf = cv2.idft(f_ishift)
imgilpf = cv2.magnitude(imgilpf[:,:,0],imgilpf[:,:,1])
```

In [351]:

```
plt.imshow(imgilpf, cmap=plt.cm.bone)
plt.show()
```



Ringing Effect: Notice the rippling artifact around the edges of the objects in the image. That is the 'ringing effect'.

Butterworth Low Pass Filters for different cut-off frequencies (n = 2)

In [352]:

```
n = 2
```

For D0 = 5

In [353]:

```
D0 = 5
```

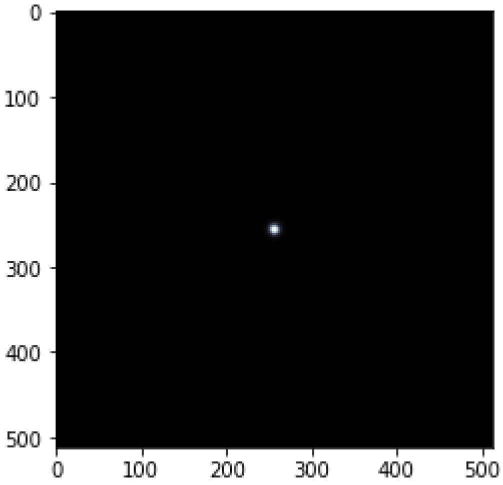
In [354]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = BLPFTransFunc(D0,D_uv,n)

        blpf[u][v] = H_uv
```

In [355]:

```
ms = cv2.magnitude(blpf[:,:,0],blpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

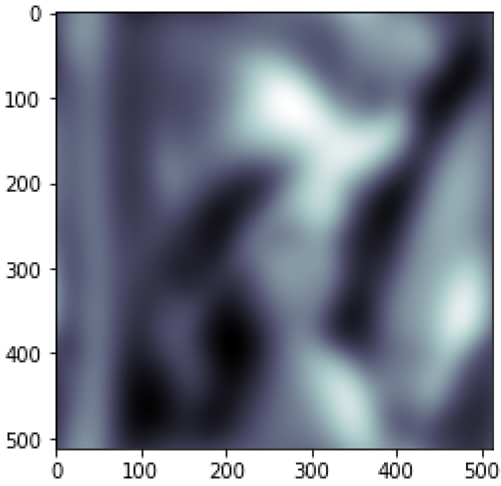


In [356]:

```
fshift = imageft*blpf
f_ishift = np.fft.ifftshift(fshift)
imgblpf = cv2.idft(f_ishift)
imgblpf = cv2.magnitude(imgblpf[:,:,0],imgblpf[:,:,1])
```

In [357]:

```
plt.imshow(imgblpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 15

In [358]:

```
D0 = 15
```

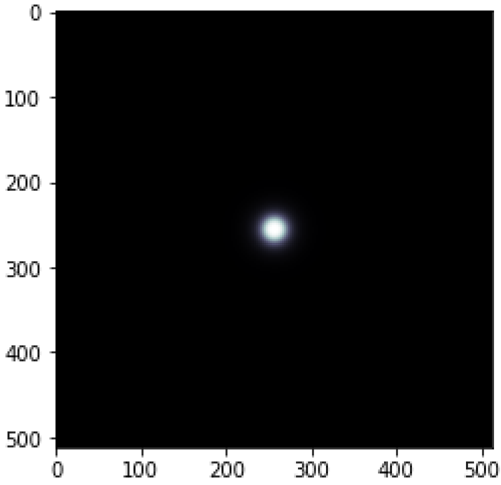
In [359]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = BLPFTransFunc(D0,D_uv,n)

        blpf[u][v] = H_uv
```

In [360]:

```
ms = cv2.magnitude(blpf[:,:,0],blpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

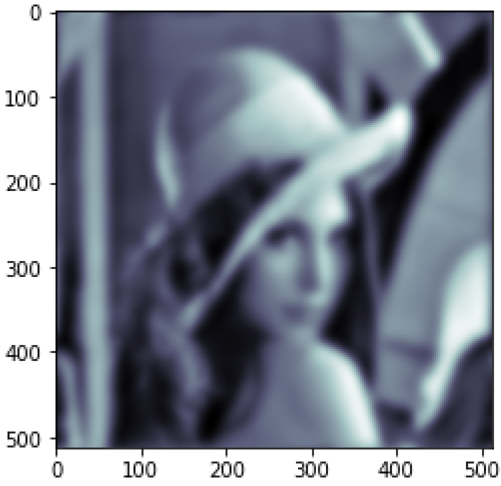


In [361]:

```
fshift = imageft*blpf
f_ishift = np.fft.ifftshift(fshift)
imgblpf = cv2.idft(f_ishift)
imgblpf = cv2.magnitude(imgblpf[:,:,0],imgblpf[:,:,1])
```

In [362]:

```
plt.imshow(imgblpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 30

In [363]:

```
D0 = 30
```

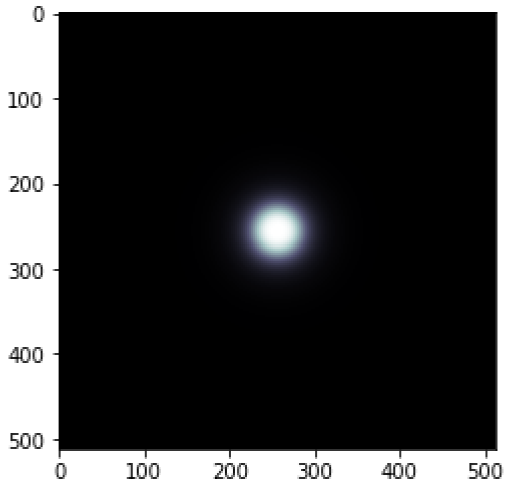
In [364]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = BLPFTransFunc(D0,D_uv,n)

        blpf[u][v] = H_uv
```

In [365]:

```
ms = cv2.magnitude(blpf[:,:,0],blpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

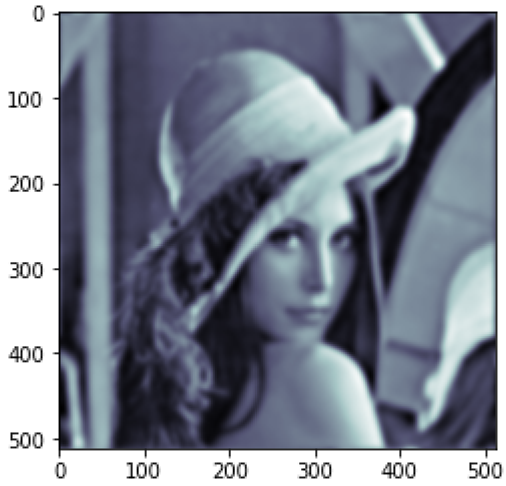


In [366]:

```
fshift = imageft*blpf
f_ishift = np.fft.ifftshift(fshift)
imgblpf = cv2.idft(f_ishift)
imgblpf = cv2.magnitude(imgblpf[:,:,0],imgblpf[:,:,1])
```

In [367]:

```
plt.imshow(imgblpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 90

In [368]:

```
D0 = 90
```

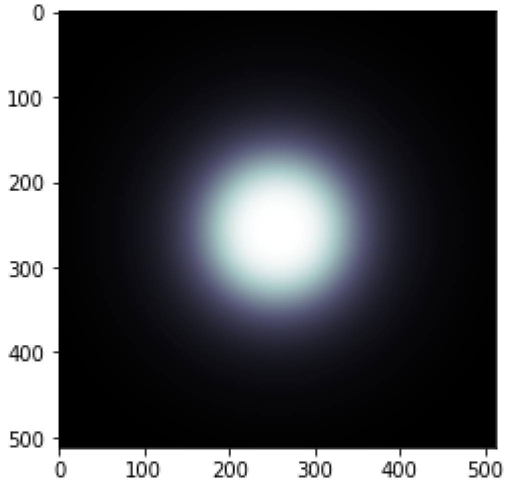
In [369]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = BLPFTransFunc(D0,D_uv,n)

        blpf[u][v] = H_uv
```

In [370]:

```
ms = cv2.magnitude(blpf[:,:,0],blpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

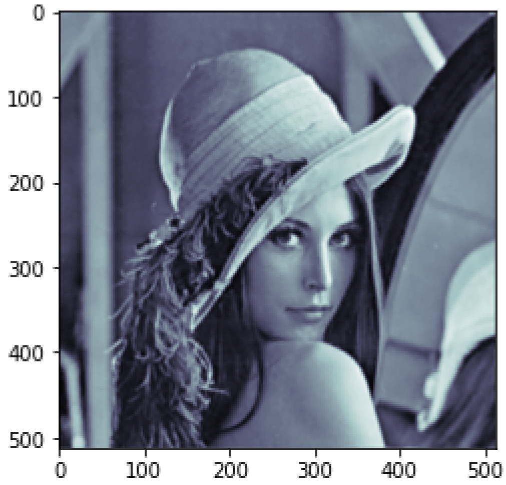


In [371]:

```
fshift = imageft*blpf
f_ishift = np.fft.ifftshift(fshift)
imgblpf = cv2.idft(f_ishift)
imgblpf = cv2.magnitude(imgblpf[:,:,0],imgblpf[:,:,1])
```

In [372]:

```
plt.imshow(imgblpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 120

In [373]:

```
D0 = 120
```

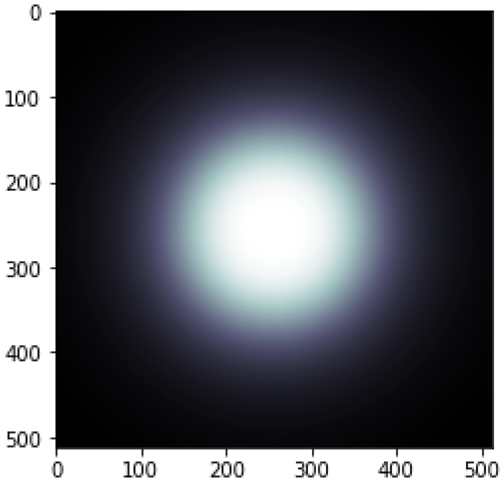
In [374]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = BLPFTransFunc(D0,D_uv,n)

        blpf[u][v] = H_uv
```

In [375]:

```
ms = cv2.magnitude(blpf[:,:,0],blpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

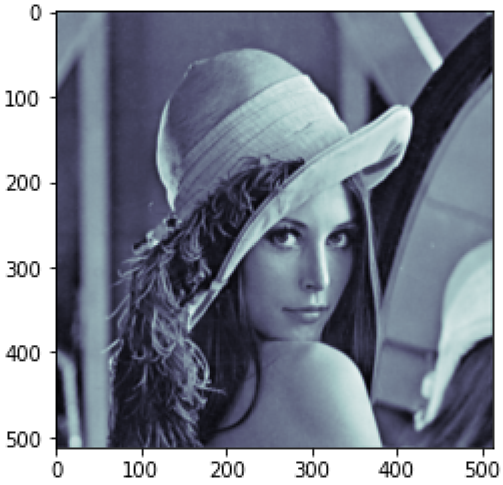


In [376]:

```
fshift = imageft*blpf
f_ishift = np.fft.ifftshift(fshift)
imgblpf = cv2.idft(f_ishift)
imgblpf = cv2.magnitude(imgblpf[:,:,0],imgblpf[:,:,1])
```

In [377]:

```
plt.imshow(imgblpf, cmap=plt.cm.bone)
plt.show()
```



Gaussian Low Pass Filters for different cut-off frequencies

For D0 = 5

In [378]:

```
D0 = 5
```

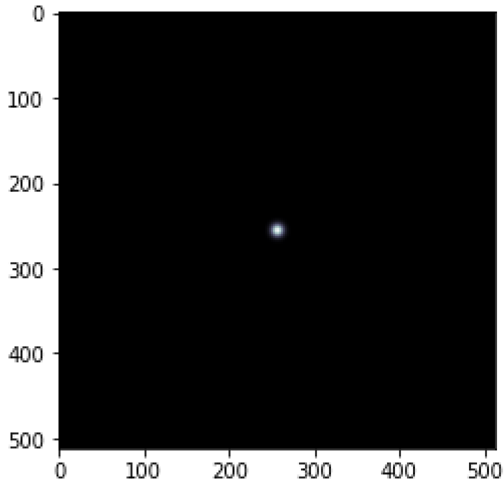
In [379]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GLPFTransFunc(D0,D_uv)

        glpf[u][v] = H_uv
```

In [380]:

```
ms = cv2.magnitude(glpf[:,:,0],glpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

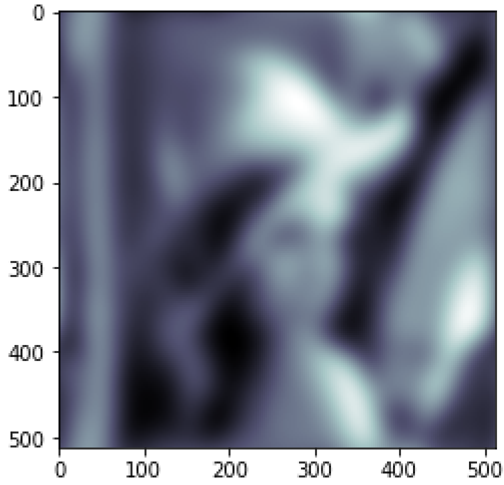


In [381]:

```
fshift = imageft*glpf
f_ishift = np.fft.ifftshift(fshift)
imgglpf = cv2.idft(f_ishift)
imgglpf = cv2.magnitude(imgglpf[:,:,0],imgglpf[:,:,1])
```

In [383]:

```
plt.imshow(imgglpf, cmap=plt.cm.bone)
plt.show()
```



D0 = 15

In [384]:

```
D0 = 15
```

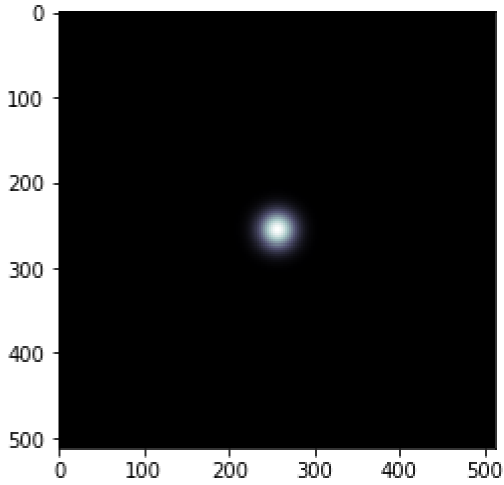
In [385]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GLPFTransFunc(D0,D_uv)

        glpf[u][v] = H_uv
```

In [386]:

```
ms = cv2.magnitude(glpf[:,:,0],glpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

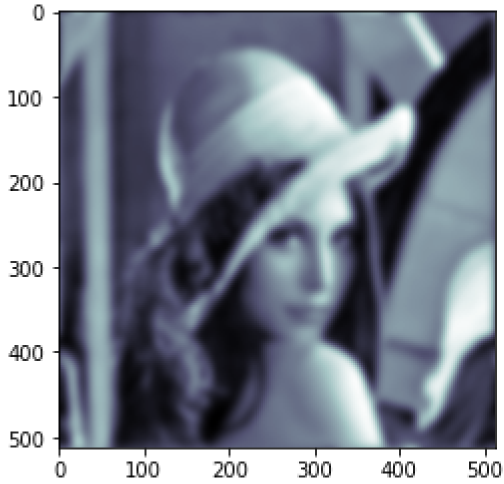


In [387]:

```
fshift = imageft*glpf
f_ishift = np.fft.ifftshift(fshift)
imgglpf = cv2.idft(f_ishift)
imgglpf = cv2.magnitude(imgglpf[:,:,0],imgglpf[:,:,1])
```

In [388]:

```
plt.imshow(imgglpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 30

In [389]:

```
D0 = 30
```

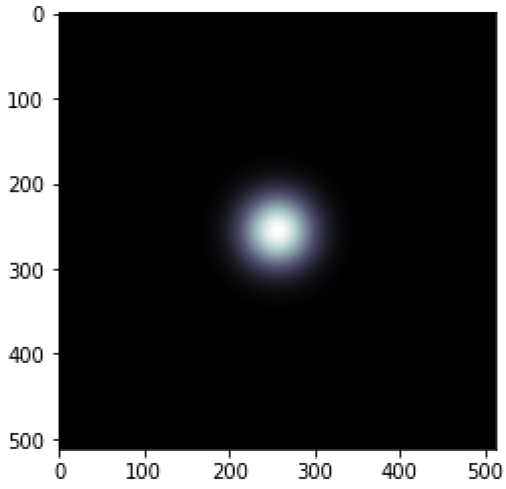
In [390]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GLPFTransFunc(D0,D_uv)

        glpf[u][v] = H_uv
```

In [391]:

```
ms = cv2.magnitude(glpf[:,:,0],glpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

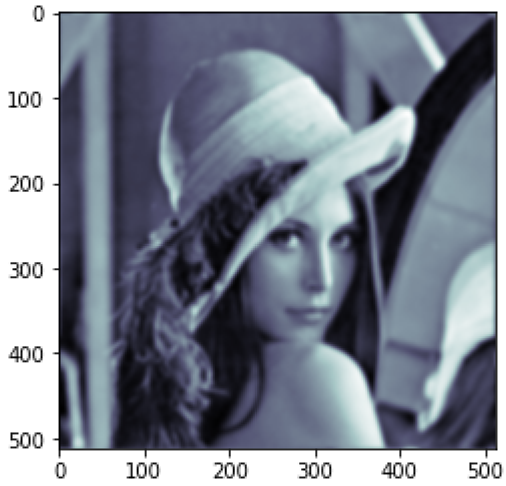


In [392]:

```
fshift = imageft*glpf
f_ishift = np.fft.ifftshift(fshift)
imgglpf = cv2.idft(f_ishift)
imgglpf = cv2.magnitude(imgglpf[:,:,0],imgglpf[:,:,1])
```

In [393]:

```
plt.imshow(imgglpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 90

In [395]:

```
D0 = 90
```

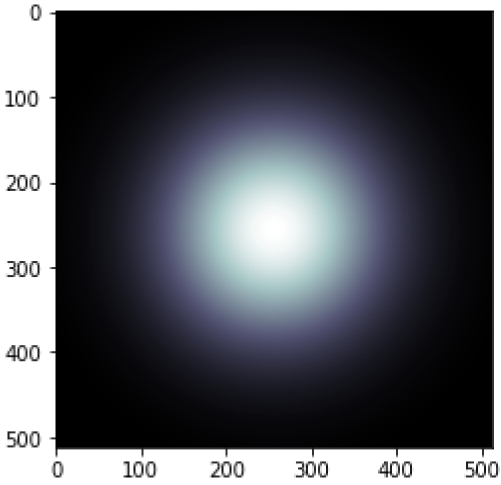
In [396]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GLPFTransFunc(D0,D_uv)

        glpf[u][v] = H_uv
```

In [397]:

```
ms = cv2.magnitude(glpf[:,:,0],glpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

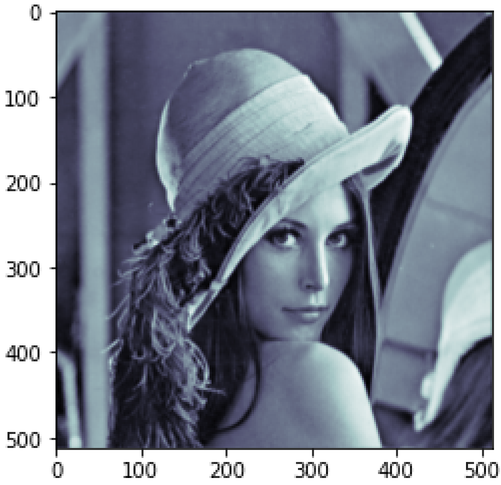


In [398]:

```
fshift = imageft*glpf
f_ishift = np.fft.ifftshift(fshift)
imgglpf = cv2.idft(f_ishift)
imgglpf = cv2.magnitude(imgglpf[:,:,0],imgglpf[:,:,1])
```

In [399]:

```
plt.imshow(imgglpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 120

In [400]:

```
D0 = 120
```

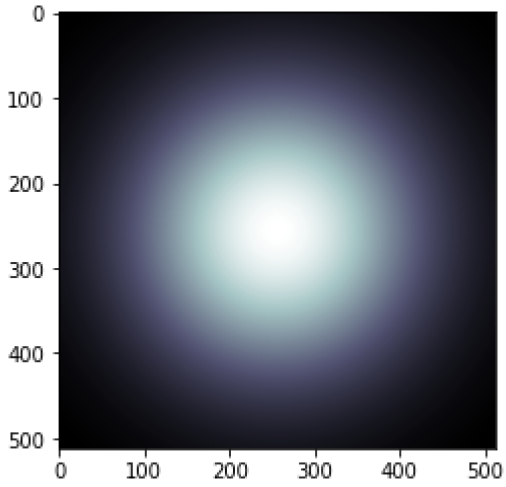
In [401]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GLPFTransFunc(D0,D_uv)

        glpf[u][v] = H_uv
```

In [402]:

```
ms = cv2.magnitude(glpf[:,:,0],glpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

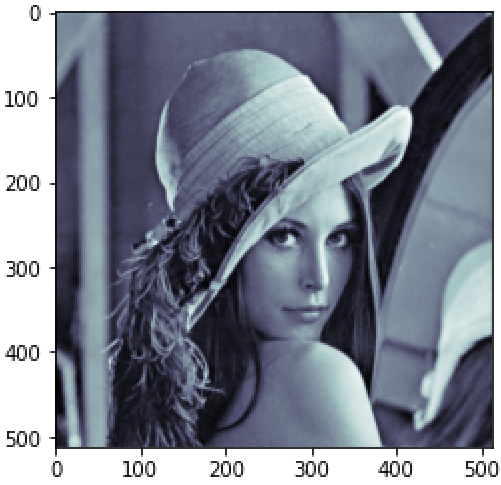


In [403]:

```
fshift = imageft*glpf
f_ishift = np.fft.ifftshift(fshift)
imgglpf = cv2.idft(f_ishift)
imgglpf = cv2.magnitude(imgglpf[:,:,0],imgglpf[:,:,1])
```

In [404]:

```
plt.imshow(imgglpf, cmap=plt.cm.bone)
plt.show()
```



Frequency Domain Sharpening Filters

In [405]:

```
D0 = 20
```

#1 Ideal High Pass Filter

In [406]:

```
def IHPFTransFunc(D0,D_uv):
    if D_uv <= D0:
        return 0
    else:
        return 1
```

In [408]:

```
ihpf = np.zeros((imgr,imgc,2))
ms = np.zeros((imgr,imgc,2))
```

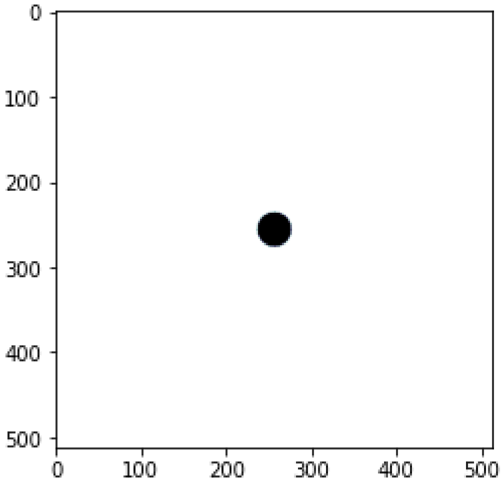
In [409]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = IHPFTransFunc(D0,D_uv)

        ihpf[u][v] = H_uv
```

In [410]:

```
ms = cv2.magnitude(ihpf[:,:,0],ihpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

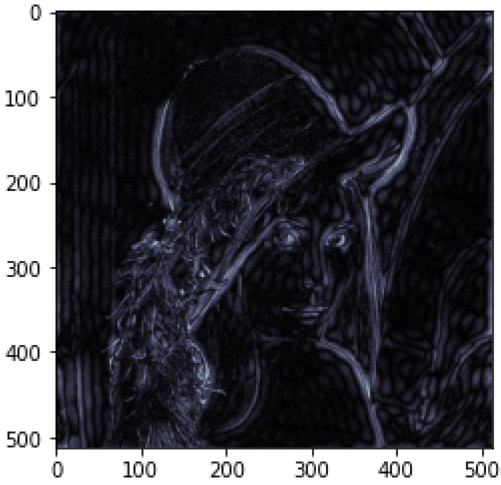


In [411]:

```
fshift = imageft*ihpf
f_ishift = np.fft.ifftshift(fshift)
imgihpf = cv2.idft(f_ishift)
imgihpf = cv2.magnitude(imgihpf[:,:,0],imgihpf[:,:,1])
```

In [412]:

```
plt.imshow(imgihpf, cmap=plt.cm.bone)
plt.show()
```



#2 Butterworth High Pass Filter

In [413]:

```
def BHPFTransFunc(D0,D_uv,n):
    return (1/(1+(D0/D_uv)**(2*n)))
```

In [414]:

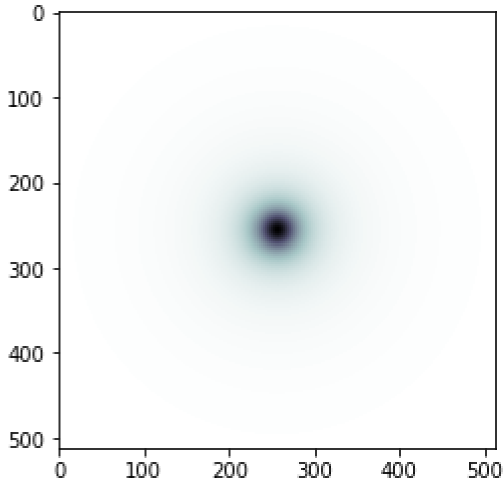
```
bhpf = np.zeros((imgr,imgc,2))
ms = np.zeros((imgr,imgc,2))
```

In [415]:

```
for u in range(imgr):
    for v in range(imgc):
        if (u == imgr/2 and v == imgc/2):
            bhpf[u][v] = 0
        else:
            D_uv = Euclidean(imgr,imgc,u,v)
            H_uv = BHPFTransFunc(D0,D_uv,1)
            bhpf[u][v] = H_uv
```

In [416]:

```
ms = cv2.magnitude(bhpf[:,:,0],bhpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

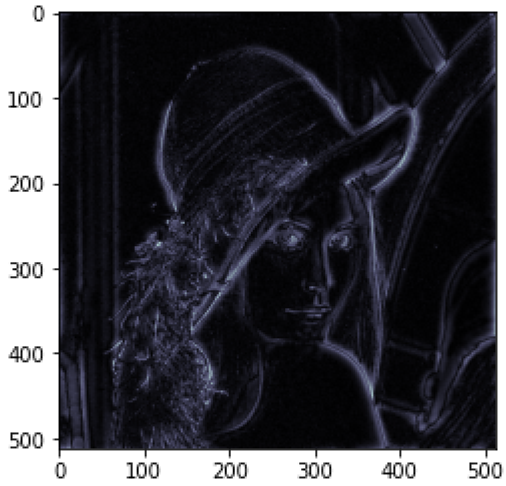


In [417]:

```
fshift = imageft*bhpf
f_ishift = np.fft.ifftshift(fshift)
imgbhpf = cv2.idft(f_ishift)
imgbhpf = cv2.magnitude(imgbhpf[:,:,0],imgbhpf[:,:,1])
```

In [418]:

```
plt.imshow(imgbhpf, cmap=plt.cm.bone)
plt.show()
```



#3 Gaussian High Pass Filter

In [419]:

```
def GHPFTransFunc(D0,D_uv):
    return (1 - math.exp(-1*((D_uv)**2)/(2*(D0**2))))
```

In [420]:

```
ghpf = np.zeros((imgr,imgc,2))
ms = np.zeros((imgr,imgc,2))
```

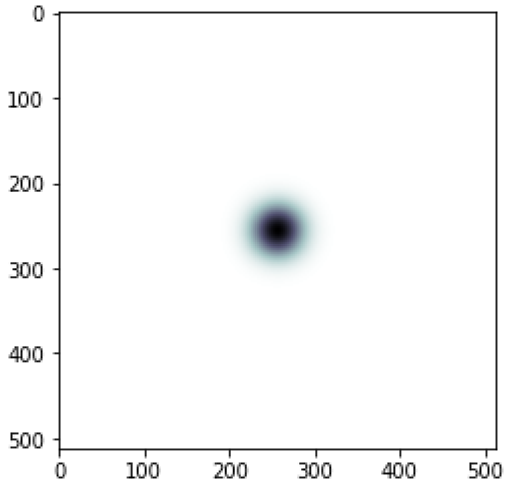
In [421]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GHPFTransFunc(D0,D_uv)

        ghpf[u][v] = H_uv
```

In [422]:

```
ms = cv2.magnitude(ghpf[:,:,0],ghpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

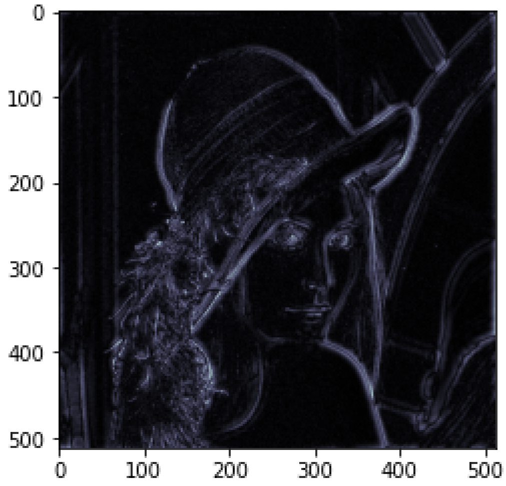


In [423]:

```
fshift = imageft*ghpf
f_ishift = np.fft.ifftshift(fshift)
imgghpf = cv2.idft(f_ishift)
imgghpf = cv2.magnitude(imgghpf[:,:,0],imgghpf[:,:,1])
```

In [424]:

```
plt.imshow(imgghpf, cmap=plt.cm.bone)
plt.show()
```



For the given cut-off frequency value D0 = 20, We clearly see the sharpening quality of Gaussian is the best, followed by Butterworth and finally Ideal High Pass Filter

Ringing Effect of Ideal High Pass Filter

In [425]:

```
D0 = 25
```

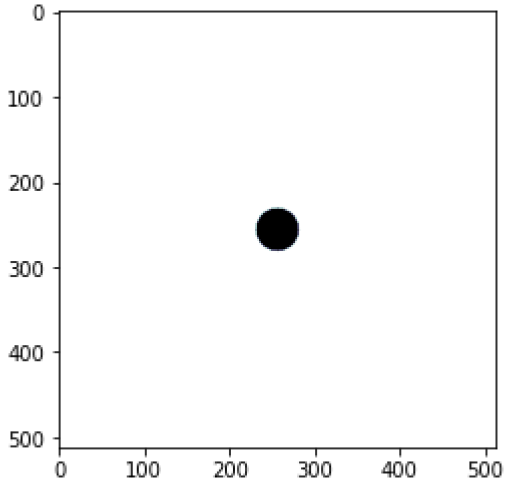
In [426]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = IHPFTransFunc(D0,D_uv)

        ihpf[u][v] = H_uv
```

In [427]:

```
ms = cv2.magnitude(ihpf[:,:,0],ihpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

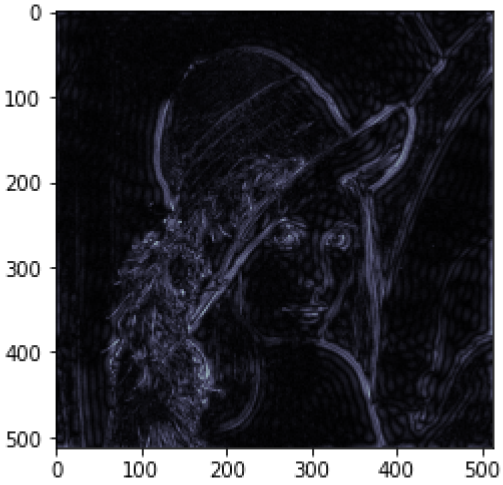


In [428]:

```
fshift = imageft*ihpf
f_ishift = np.fft.ifftshift(fshift)
imgihpf = cv2.idft(f_ishift)
imgihpf = cv2.magnitude(imgihpf[:,:,0],imgihpf[:,:,1])
```

In [429]:

```
plt.imshow(imgihpf, cmap=plt.cm.bone)
plt.show()
```



Ringing Effect: Notice the rippling artifact around the edges of the objects in the image. That is the 'ringing effect'.

Butterworth High Pass Filters for different cut-off frequencies (n = 2)

In [430]:

```
n = 2
```

For D0 = 5

In [431]:

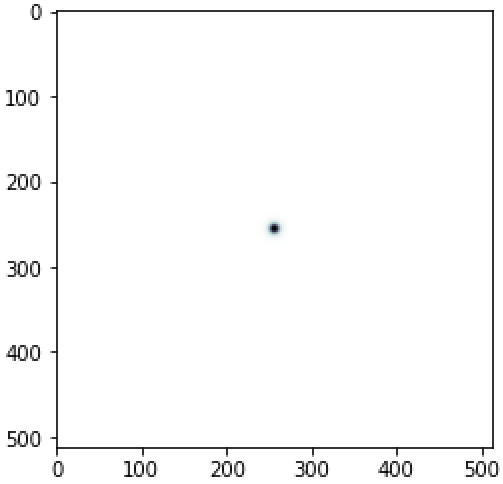
```
D0 = 5
```

In [432]:

```
for u in range(imgr):
    for v in range(imgc):
        if (u == imgr/2 and v == imgc/2):
            bhpf[u][v] = 0
        else:
            D_uv = Euclidean(imgr,imgc,u,v)
            H_uv = BHPFTransFunc(D0,D_uv,n)
            bhpf[u][v] = H_uv
```

In [433]:

```
ms = cv2.magnitude(bhpf[:,:,0],bhpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

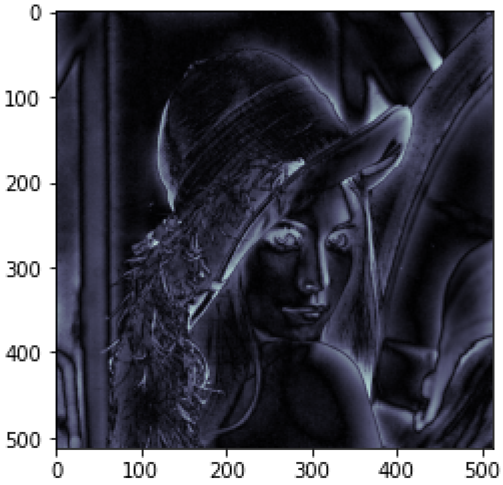


In [434]:

```
fshift = imageft*bhpf
f_ishift = np.fft.ifftshift(fshift)
imgbhpf = cv2.idft(f_ishift)
imgbhpf = cv2.magnitude(imgbhpf[:,:,0],imgbhpf[:,:,1])
```

In [435]:

```
plt.imshow(imgbhpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 15

In [436]:

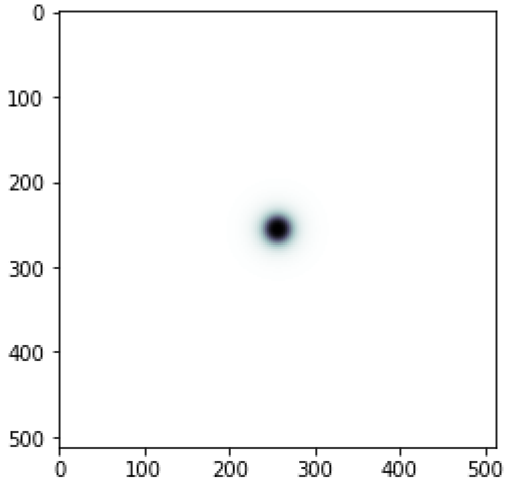
```
D0 = 15
```

In [437]:

```
for u in range(imgr):
    for v in range(imgc):
        if (u == imgr/2 and v == imgc/2):
            bhpf[u][v] = 0
        else:
            D_uv = Euclidean(imgr,imgc,u,v)
            H_uv = BHPFTransFunc(D0,D_uv,n)
            bhpf[u][v] = H_uv
```

In [438]:

```
ms = cv2.magnitude(bhpf[:,:,0],bhpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

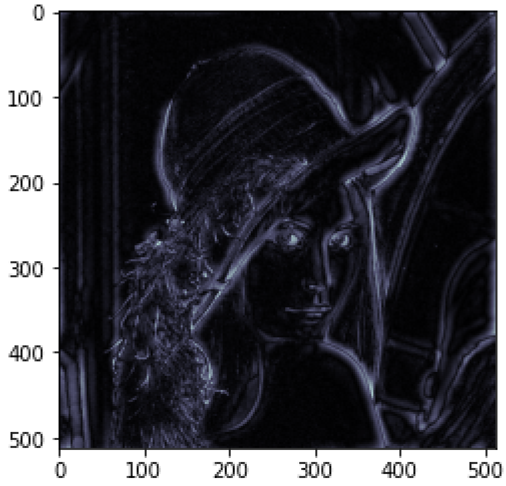


In [439]:

```
fshift = imageft*bhpf
f_ishift = np.fft.ifftshift(fshift)
imgbhpf = cv2.idft(f_ishift)
imgbhpf = cv2.magnitude(imgbhpf[:,:,0],imgbhpf[:,:,1])
```

In [440]:

```
plt.imshow(imgbhpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 30

In [441]:

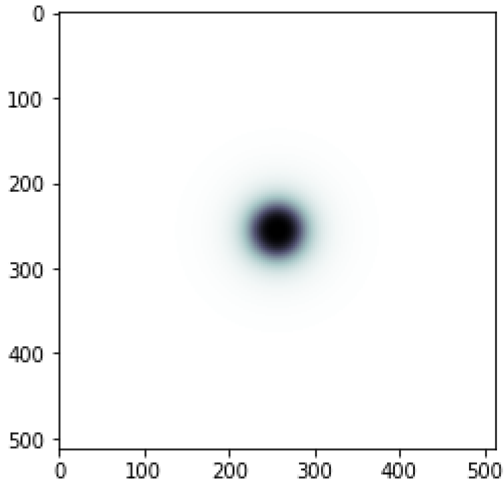
```
D0 = 30
```

In [442]:

```
for u in range(imgr):
    for v in range(imgc):
        if (u == imgr/2 and v == imgc/2):
            bhpf[u][v] = 0
        else:
            D_uv = Euclidean(imgr,imgc,u,v)
            H_uv = BHPFTransFunc(D0,D_uv,n)
            bhpf[u][v] = H_uv
```

In [443]:

```
ms = cv2.magnitude(bhpf[:,:,0],bhpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

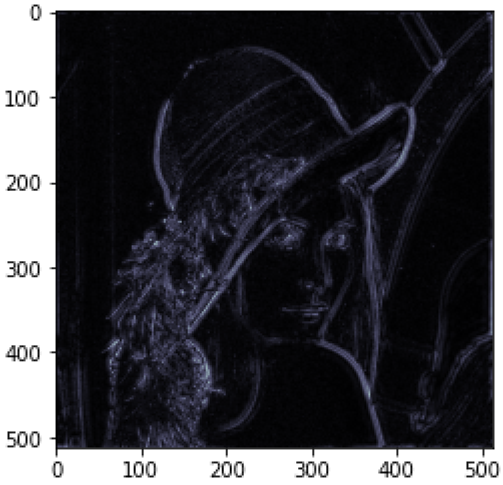


In [444]:

```
fshift = imageft*bhpf
f_ishift = np.fft.ifftshift(fshift)
imgbhpf = cv2.idft(f_ishift)
imgbhpf = cv2.magnitude(imgbhpf[:,:,0],imgbhpf[:,:,1])
```

In [445]:

```
plt.imshow(imgbhpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 90

In [446]:

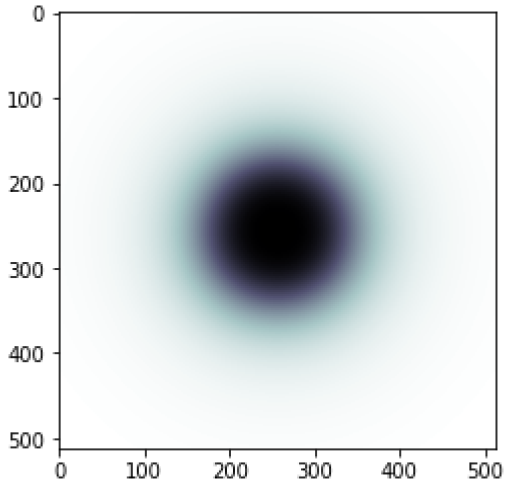
```
D0 = 90
```

In [447]:

```
for u in range(imgr):
    for v in range(imgc):
        if (u == imgr/2 and v == imgc/2):
            bhpf[u][v] = 0
        else:
            D_uv = Euclidean(imgr,imgc,u,v)
            H_uv = BHPFTransFunc(D0,D_uv,n)
            bhpf[u][v] = H_uv
```

In [448]:

```
ms = cv2.magnitude(bhpf[:,:,0],bhpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

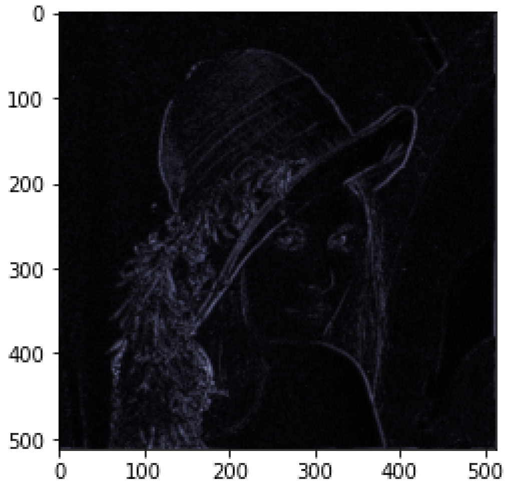


In [449]:

```
fshift = imageft*bhpf
f_ishift = np.fft.ifftshift(fshift)
imgbhpf = cv2.idft(f_ishift)
imgbhpf = cv2.magnitude(imgbhpf[:,:,0],imgbhpf[:,:,1])
```

In [450]:

```
plt.imshow(imgbhpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 120

In [451]:

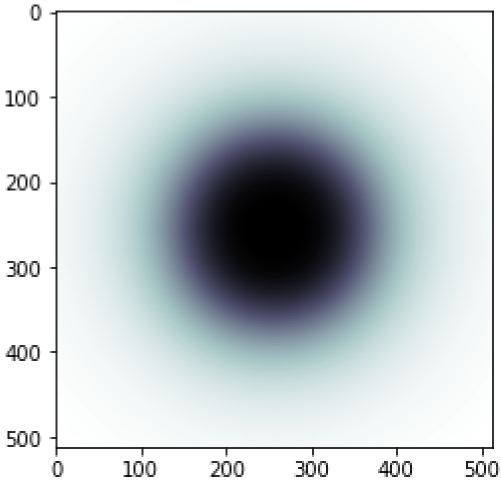
```
D0 = 120
```

In [452]:

```
for u in range(imgr):
    for v in range(imgc):
        if (u == imgr/2 and v == imgc/2):
            bhpf[u][v] = 0
        else:
            D_uv = Euclidean(imgr,imgc,u,v)
            H_uv = BHPFTransFunc(D0,D_uv,n)
            bhpf[u][v] = H_uv
```

In [453]:

```
ms = cv2.magnitude(bhpf[:,:,0],bhpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

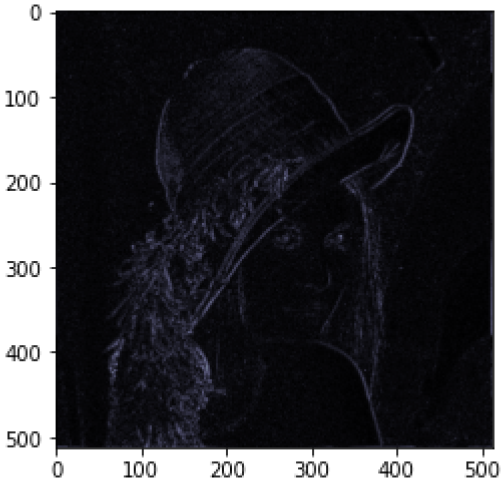


In [454]:

```
fshift = imageft*bhpf
f_ishift = np.fft.ifftshift(fshift)
imgbhpf = cv2.idft(f_ishift)
imgbhpf = cv2.magnitude(imgbhpf[:,:,0],imgbhpf[:,:,1])
```

In [455]:

```
plt.imshow(imgbhpf, cmap=plt.cm.bone)
plt.show()
```



Gaussian High Pass Filters for different cut-off frequencies

For D0 = 5

In [456]:

```
D0 = 5
```

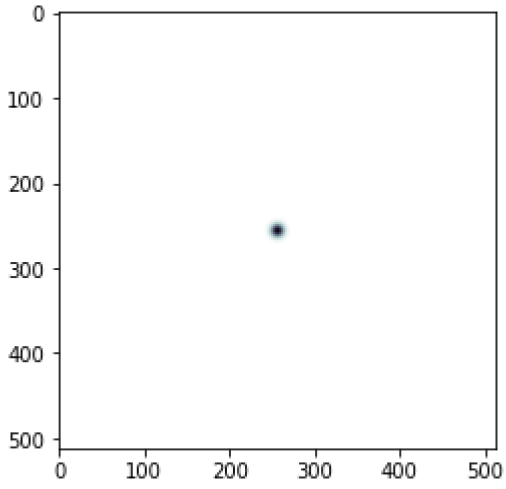
In [457]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GHPFTransFunc(D0,D_uv)

        ghpf[u][v] = H_uv
```

In [458]:

```
ms = cv2.magnitude(ghpf[:,:,0],ghpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

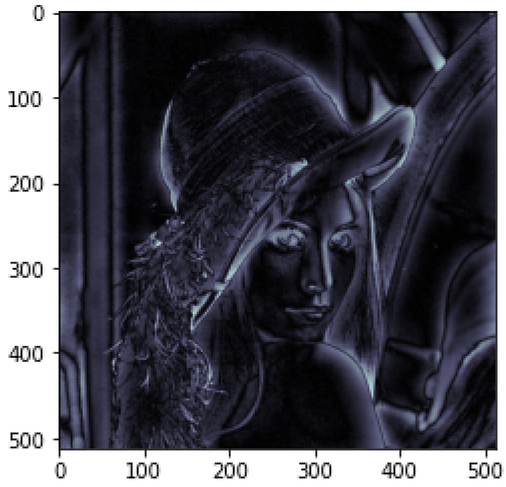


In [459]:

```
fshift = imageft*ghpf
f_ishift = np.fft.ifftshift(fshift)
imgghpf = cv2.idft(f_ishift)
imgghpf = cv2.magnitude(imgghpf[:,:,0],imgghpf[:,:,1])
```

In [460]:

```
plt.imshow(imgghpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 15

In [461]:

```
D0 = 15
```

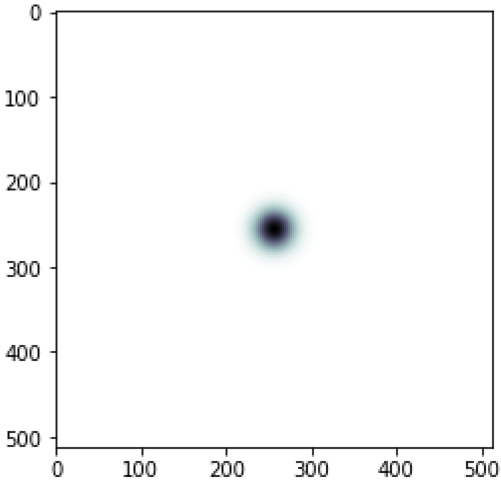
In [462]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GHPFTransFunc(D0,D_uv)

        ghpf[u][v] = H_uv
```

In [463]:

```
ms = cv2.magnitude(ghpf[:,:,0],ghpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

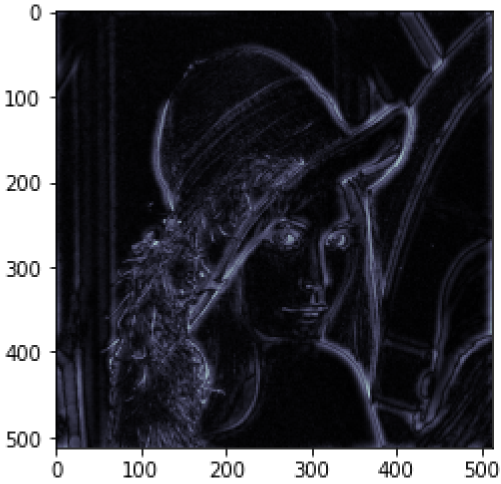


In [464]:

```
fshift = imageft*ghpf
f_ishift = np.fft.ifftshift(fshift)
imgghpf = cv2.idft(f_ishift)
imgghpf = cv2.magnitude(imgghpf[:,:,0],imgghpf[:,:,1])
```

In [465]:

```
plt.imshow(imgghpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 30

In [466]:

```
D0 = 30
```

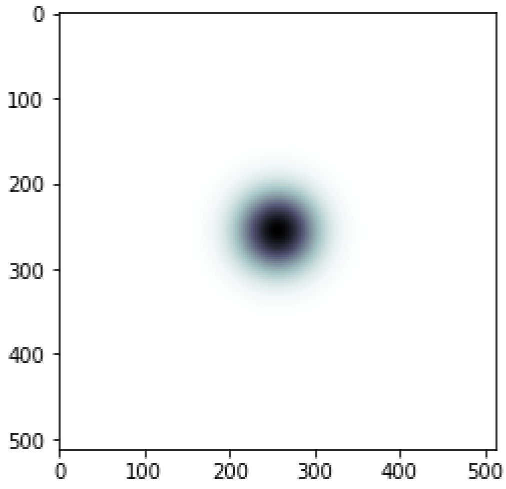
In [467]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GHPFTransFunc(D0,D_uv)

        ghpf[u][v] = H_uv
```

In [468]:

```
ms = cv2.magnitude(ghpf[:,:,0],ghpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

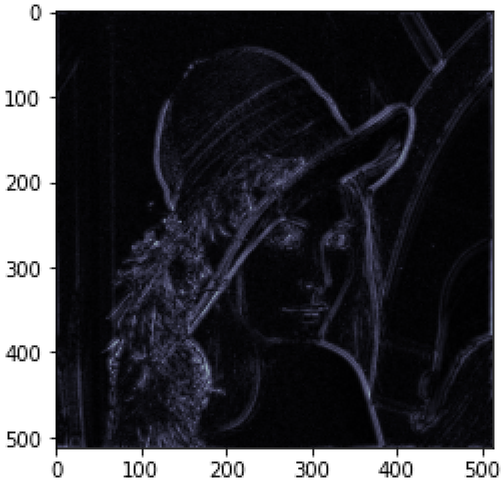


In [469]:

```
fshift = imageft*ghpf
f_ishift = np.fft.ifftshift(fshift)
imgghpf = cv2.idft(f_ishift)
imgghpf = cv2.magnitude(imgghpf[:,:,0],imgghpf[:,:,1])
```

In [470]:

```
plt.imshow(imgghpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 90

In [310]:

```
D0 = 90
```

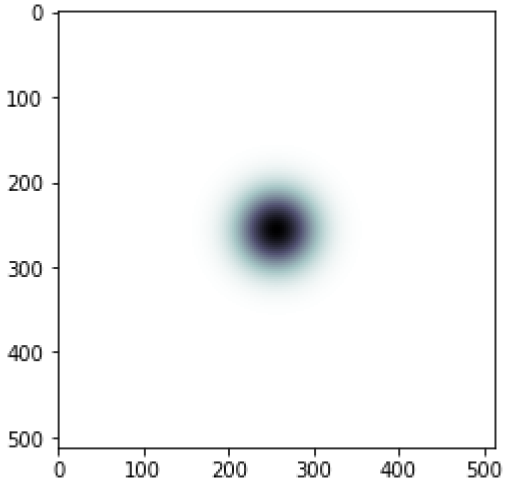
In [471]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GHPFTransFunc(D0,D_uv)

        ghpf[u][v] = H_uv
```

In [472]:

```
ms = cv2.magnitude(ghpf[:,:,0],ghpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

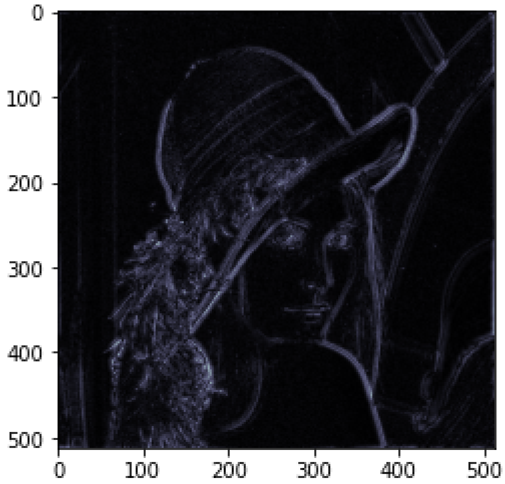


In [473]:

```
fshift = imageft*ghpf
f_ishift = np.fft.ifftshift(fshift)
imgghpf = cv2.idft(f_ishift)
imgghpf = cv2.magnitude(imgghpf[:,:,0],imgghpf[:,:,1])
```

In [474]:

```
plt.imshow(imgghpf, cmap=plt.cm.bone)
plt.show()
```



For D0 = 120

In [475]:

```
D0 = 120
```

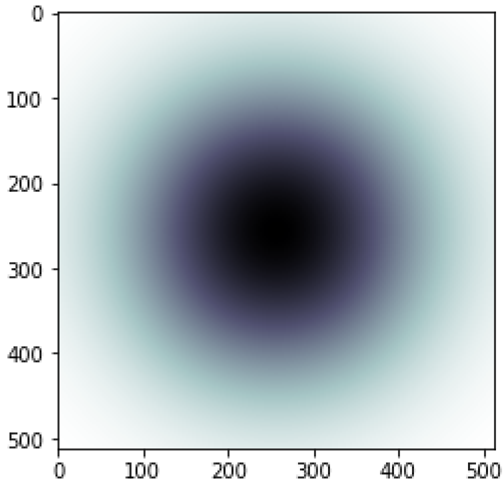
In [476]:

```
for u in range(imgr):
    for v in range(imgc):
        D_uv = Euclidean(imgr,imgc,u,v)
        H_uv = GHPFTransFunc(D0,D_uv)

        ghpf[u][v] = H_uv
```

In [477]:

```
ms = cv2.magnitude(ghpf[:,:,0],ghpf[:,:,1])
plt.imshow(ms, cmap=plt.cm.bone)
plt.show()
```

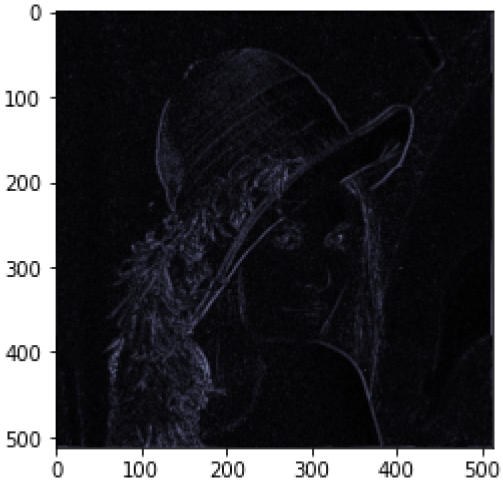


In [478]:

```
fshift = imageft*ghpf
f_ishift = np.fft.ifftshift(fshift)
imgghpf = cv2.idft(f_ishift)
imgghpf = cv2.magnitude(imgghpf[:,:,0],imgghpf[:,:,1])
```

In [479]:

```
plt.imshow(imgghpf, cmap=plt.cm.bone)
plt.show()
```



In [ ]:

```
```