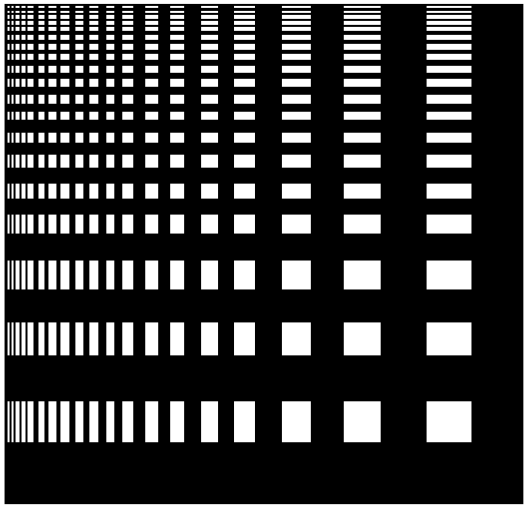
Elif KIR

```
img=imread('testpat2.tif');
imshow(img);
```

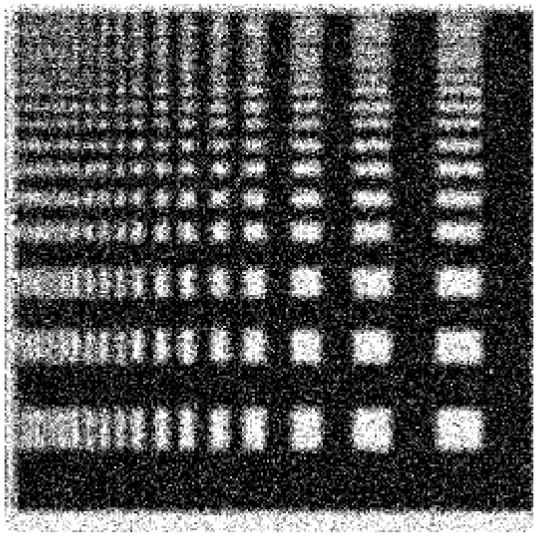


```
size(img)
```

```
ans = 1×2
    256    256
```

```
img=im2double(img);
f=fspecial('average',5);
b_img=imfilter (img, f,'conv',"circular");
bn_img=imnoise(b_img,'gaussian',0, 0.06);
imshow(bn_img)
```

Elif KIR



```
%%%a%%%

[M,N]=size(bn_img)
```

```
M = 256
N = 256
```

```
Hf=fft2(f,M,N) % fft2,
```

```
Hf = 256×256 complex
   1.0000 + 0.0000i   0.9982 - 0.0490i   0.9928 - 0.0978i   0.9838 -
0.1459i …
   0.9982 - 0.0490i   0.9940 - 0.0979i   0.9862 - 0.1463i   0.9749 -
0.1939i
   0.9928 - 0.0978i   0.9862 - 0.1463i   0.9761 - 0.1942i   0.9625 -
0.2411i
   0.9838 - 0.1459i   0.9749 - 0.1939i   0.9625 - 0.2411i   0.9466 -
0.2872i
   0.9714 - 0.1932i   0.9601 - 0.2405i   0.9455 - 0.2868i   0.9274 -
0.3318i
   0.9555 - 0.2393i   0.9420 - 0.2858i   0.9252 - 0.3310i   0.9051 -
0.3749i
   0.9363 - 0.2840i   0.9207 - 0.3294i   0.9018 - 0.3735i   0.8797 -
0.4161i
   0.9140 - 0.3270i   0.8963 - 0.3713i   0.8754 - 0.4140i   0.8515 -
0.4551i
   0.8886 - 0.3681i   0.8690 - 0.4110i   0.8462 - 0.4523i   0.8206 -
0.4918i
   0.8605 - 0.4070i   0.8390 - 0.4484i   0.8145 - 0.4882i   0.7872 -
0.5260i
      ⋮
```

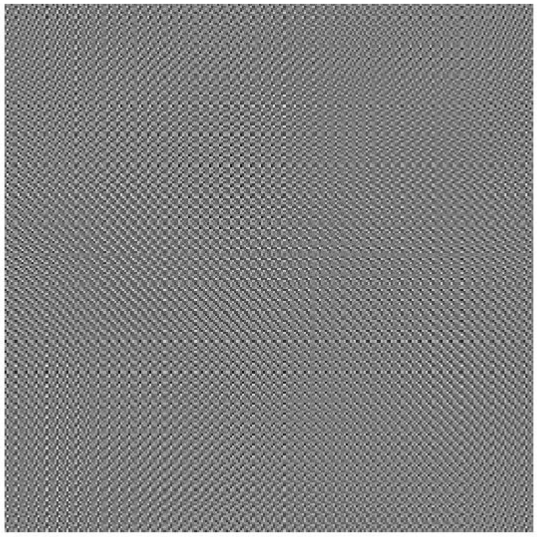
```
Gtemp_INV=1./Hf %inverse  filter
```

```
Gtemp_INV = 256×256 complex
10^4 ×
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i ⋯
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0000i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 +
0.0001i
   0.0001 + 0.0000i   0.0001 + 0.0000i   0.0001 + 0.0001i   0.0001 +
0.0001i
      ⋮
```

```
filtered_img=ifft2(Gtemp_INV.*fft2(bn_img))
```

```
filtered_img = 256×256
    2.2472   97.1903   74.7070 -217.1043   28.6517   54.6962   74.5649 ⋯
 -200.8505   26.8890  -43.4715  123.0277  155.3773 -286.6792   72.9432
 -126.8497  392.3685 -154.3411   -7.2751 -126.6664  -72.2569  368.5315
  118.3704 -160.7283  -19.9927  114.6581  -68.3778   91.2584 -187.9604
  275.9190 -354.4194  120.9347  -19.0963   -8.0109  261.9066 -326.2289
  -29.5253   81.0381  119.5602 -223.3683   33.2805   36.5872   65.2593
 -266.7168   61.7406   -8.3086  102.2282  166.8860 -334.6053   81.8313
  -90.2891  380.3310 -197.4497   -9.6621 -112.9200  -32.8772  358.3066
  122.4698 -126.8205  -62.8618  147.4921  -84.9330   63.3480 -122.1423
  250.9920 -378.6596  151.1284  -15.9332   12.6840  245.7314 -375.4271
      ⋮
```

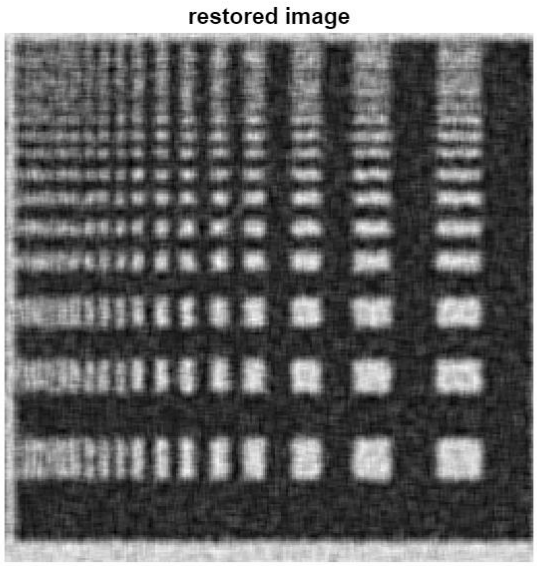
```
figure;imshow(filtered_img,[]);
```



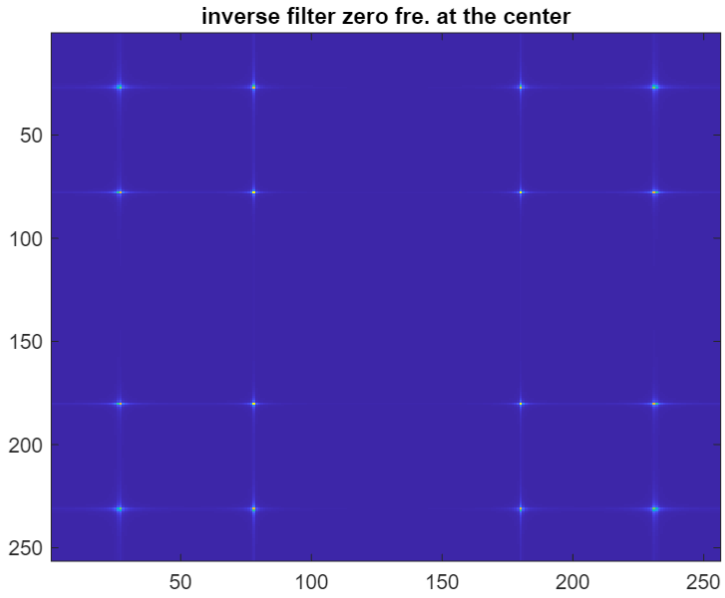
```
%Inverse filter is sensitive to noise.

%%%b%%%
nsr=0.06/var(bn_img(:))
```

```
nsr = 0.5566
```

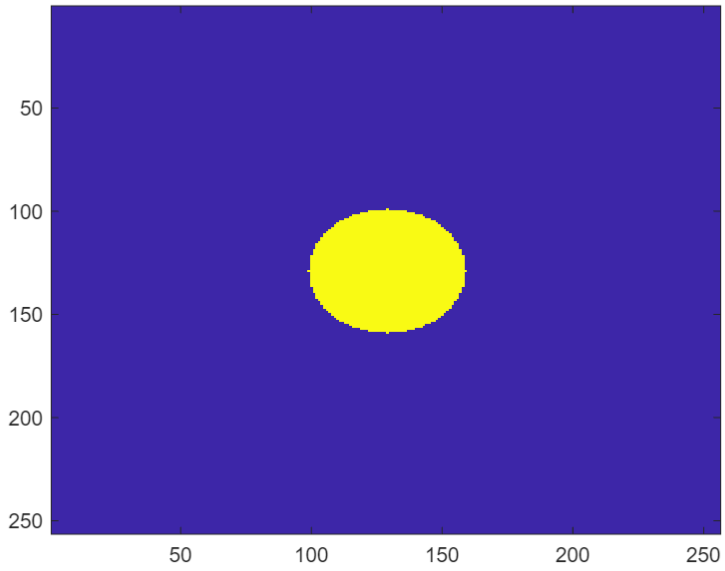
```
Im_rec1=deconvwnr (bn_img,f,nsr);
imshow(Im_rec1,[]);title('restored image')
```



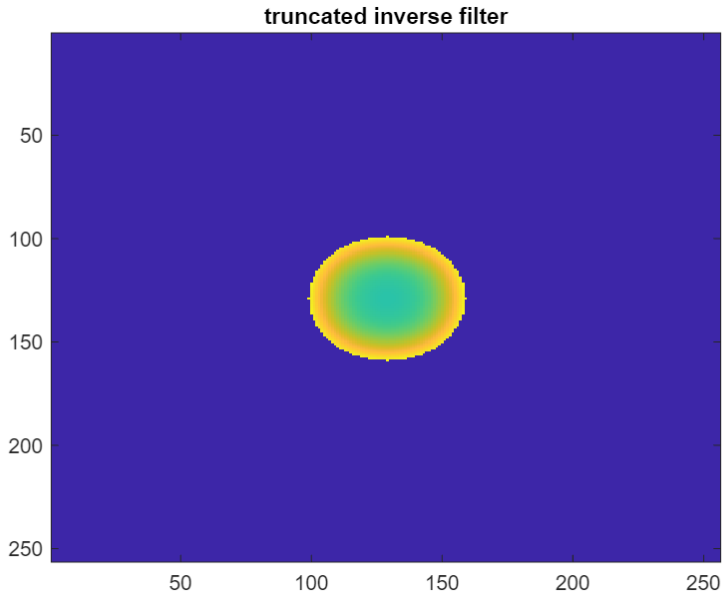
```
%Wiener filter works better when there is noise.

%%%c%%%

%  Truncated inverse filter : Gtemp=1./Hf; %inverse  filter
Gtemp_INV1=fftshift(Gtemp_INV);
figure;imagesc(abs(Gtemp_INV1));title('inverse filter zero fre. at the center')
```




```
[U,V]=meshgrid(-N/2:1:N/2-1,-M/2:1:M/2-1);
D=sqrt(U.^2+V.^2);
fcutoff=30;
D_n=D<=fcutoff;figure;imagesc(D_n);
```

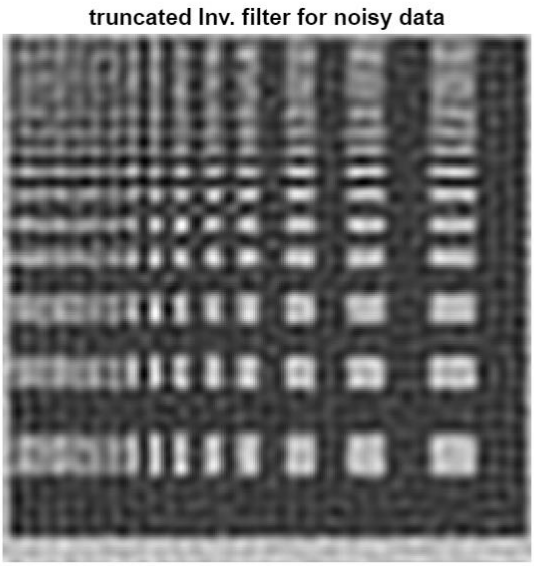



```
Gtemp_INV_T=Gtemp_INV1.*D_n;
figure;imagesc(abs(Gtemp_INV_T));title('truncated inverse filter')
```




```
% shift the low frequency to the corner
Gtemp_INV_T=ifftshift(Gtemp_INV_T);
% apply truncated inverse filter
Im_rec2=ifft2(Gtemp_INV_T.*fft2(bn_img));
figure;imshow(Im_rec2,[]); title('truncated Inv. filter for noisy data')
```

Elif KIR




```
%Radius cutoff filter works good with cutoff 30. But still the best is
%wiener.
```