PSF eigenvalue

March 24, 2020

1 PSF eigenvalue

The objective is to calculate eigenvalue of a PSF matrix.

[1]:
```
using Revise
```

[2]:
```
using PyPlot, LinearAlgebra
pygui(false)
```

[2]:
```
false
```

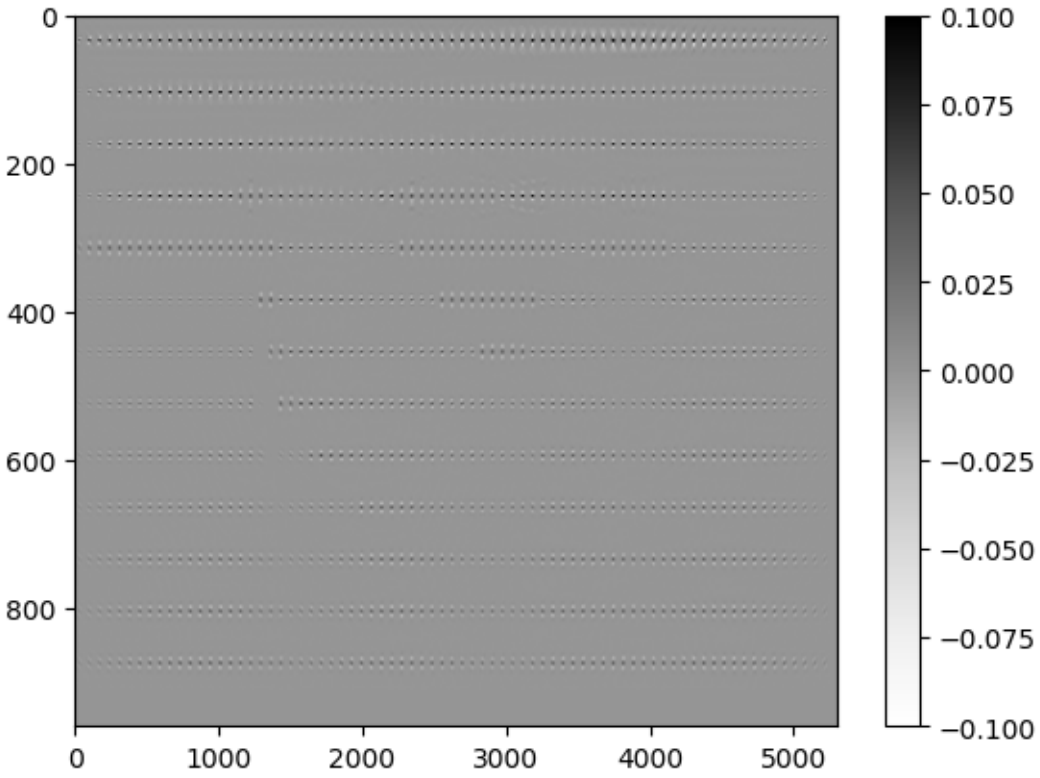
[3]:
```
include("psfeig.jl")
include("utils.jl")
```

[3]:
```
Main.utils
```

1.1 Reading Modeled PSFs

[8]:
```
path = "/home/bruno/Dropbox/Luana_PSF/psfs_su5_size71/
 ↪malha1_psfs_su5_size71_sem_ph_interp.rsf@"
image = utils.read_bin(path,960,5300);
```

[9]:
```
utils.sisshow(image, vmin=-0.1, vmax=0.1)
```



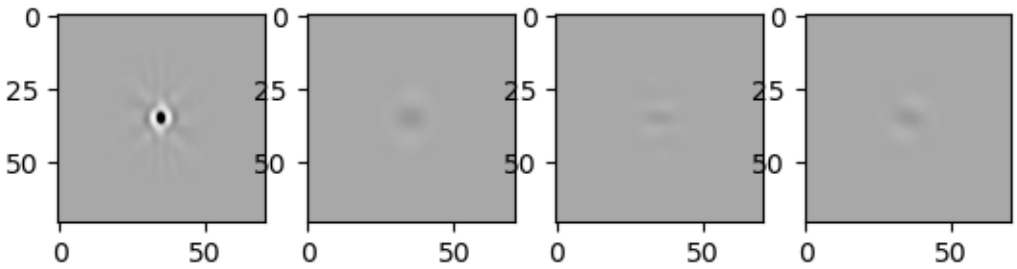
[10]:
```
b1=70; b2=70; k1=36; k2=36; l1=920; l2=5260;
println("psf grid: ($(length(k1:b1:l1)),$(length(k2:b2:l2)))")
psfs = psfeig.psf_chop(image,b1,b2,k1,k2,l1,l2);
```

```
psf grid: (13,75)
```

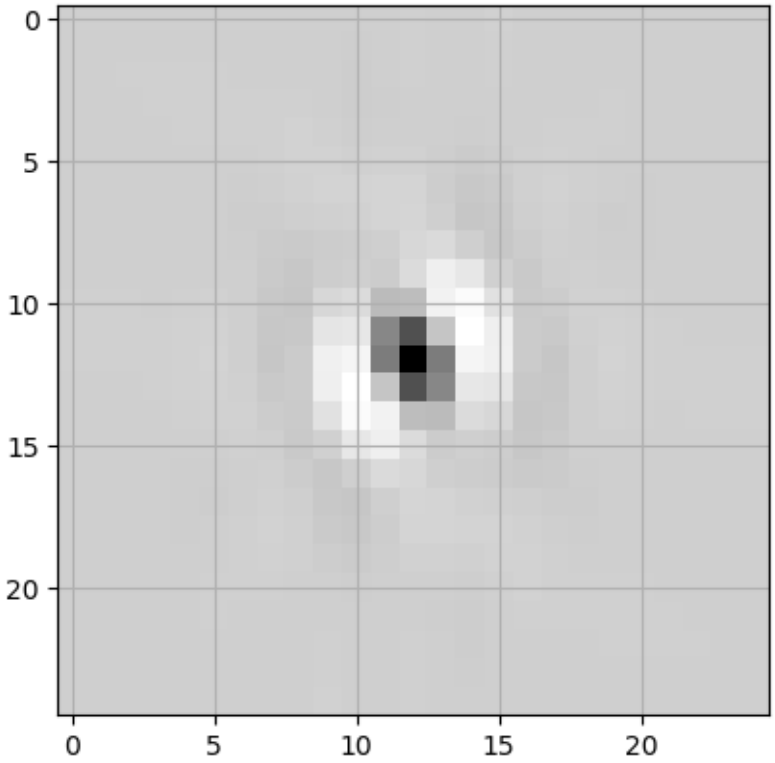
[11]:
```
size(psfs.val,3)
```

[11]: 975

[12]:
```
utils.sisshow(psfs.val[:,:,9*13+1], psfs.val[:,:,9*13+5], psfs.val[:,:,9*13+9],␣
↪psfs.val[:,:,9*13+13], perc=99.9)
```



[13]:
```
o=36; l=12
imshow(psfs.val[(o-l):(o+l),(o-l):(o+l),1], cmap="Greys")
grid()
```



[14]:
```
size(psfs.val[(o-l):(o+l),(o-l):(o+l),1])
```

[14]: (25, 25)

[15]:
```
findmax(psfs.val[(o-l):(o+l),(o-l):(o+l),1])
```

[15]: (3.3668396f0, CartesianIndex(13, 13))

1.2 Eigenvalues of Modeled PSFs

[16]:
```
function plot_psf_with_condition_number(ipsf)
    P = psfs.val[(o-l):(o+l),(o-l):(o+l),ipsf];
    G = psfeig.psf_to_matrix(n,P);
    figure()
```

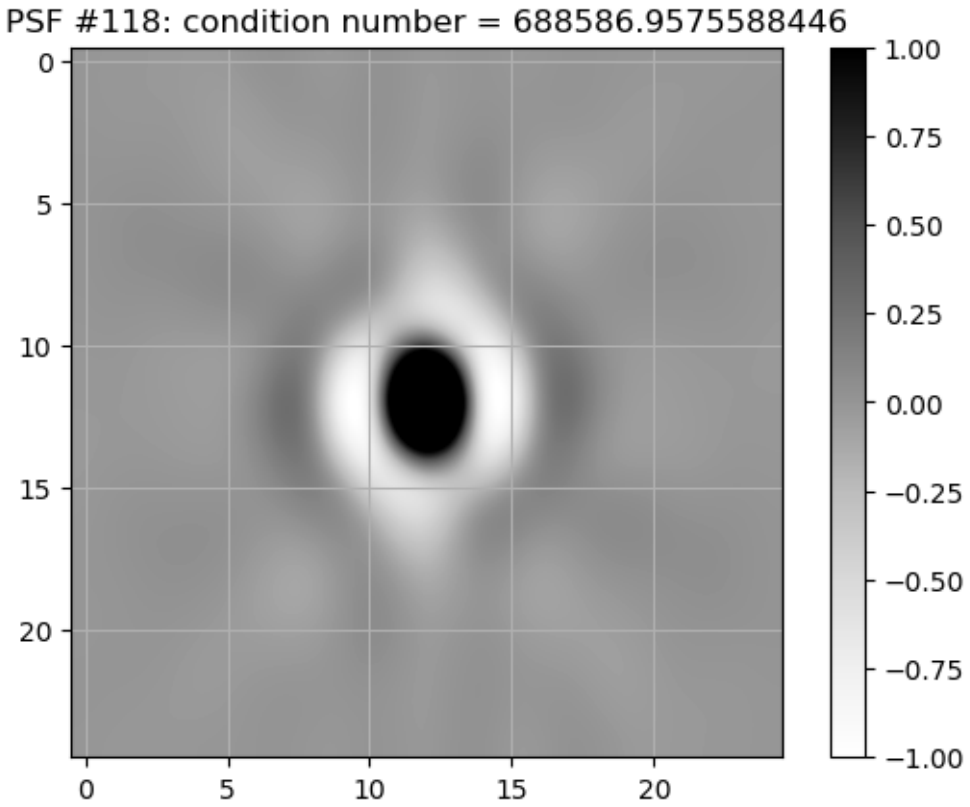
```
    imshow(psfs.val[(o-l):(o+l),(o-l):(o+l),ipsf],
            interpolation="bicubic", cmap="Greys", vmin=-1, vmax=1)
    colorbar()
    grid()
    title("PSF #$(ipsf): condition number = $(cond(G))")
end
```

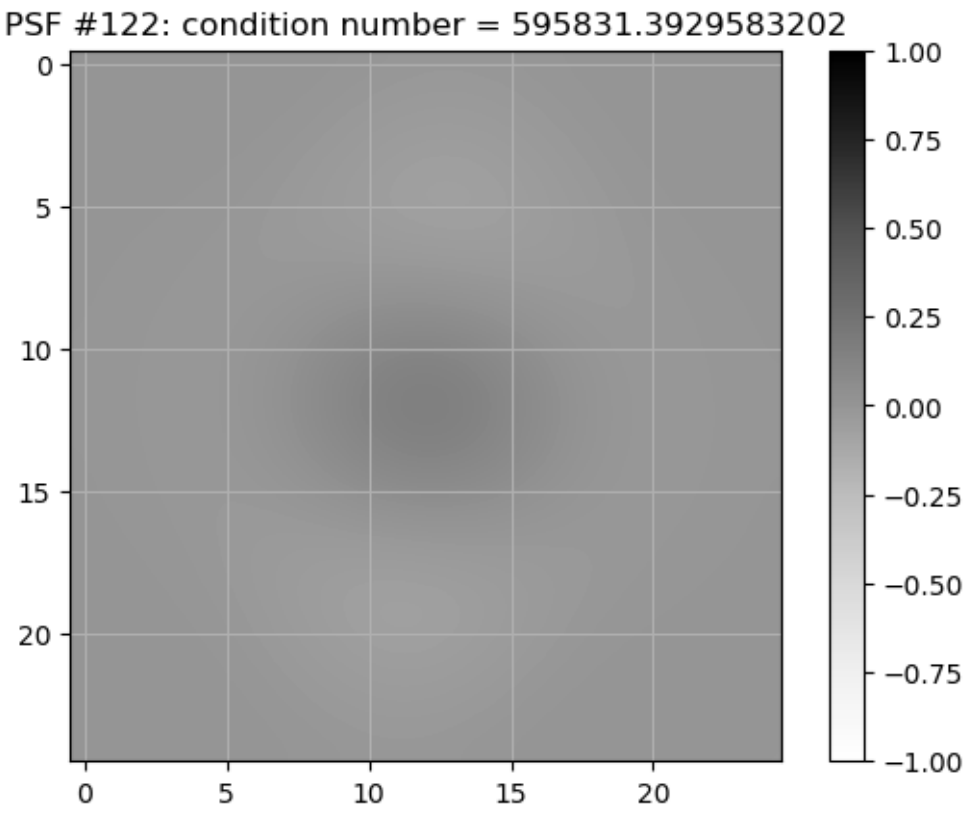
[16]: plot_psf_with_condition_number (generic function with 1 method)

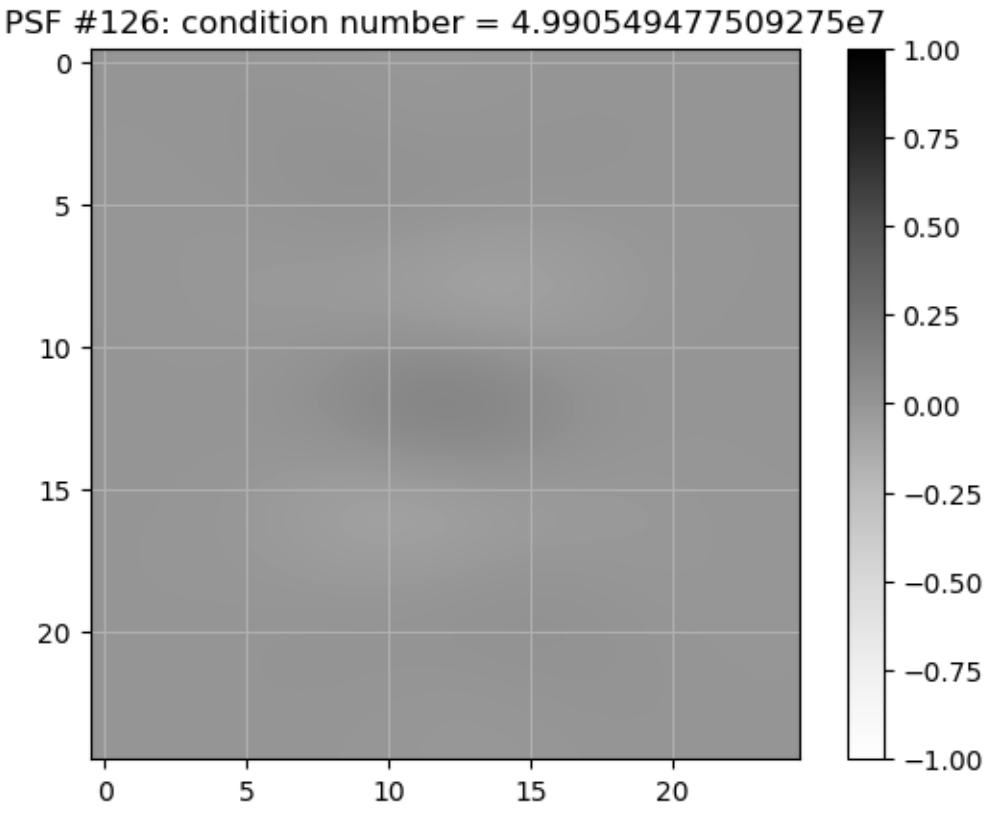
Distict PSFs

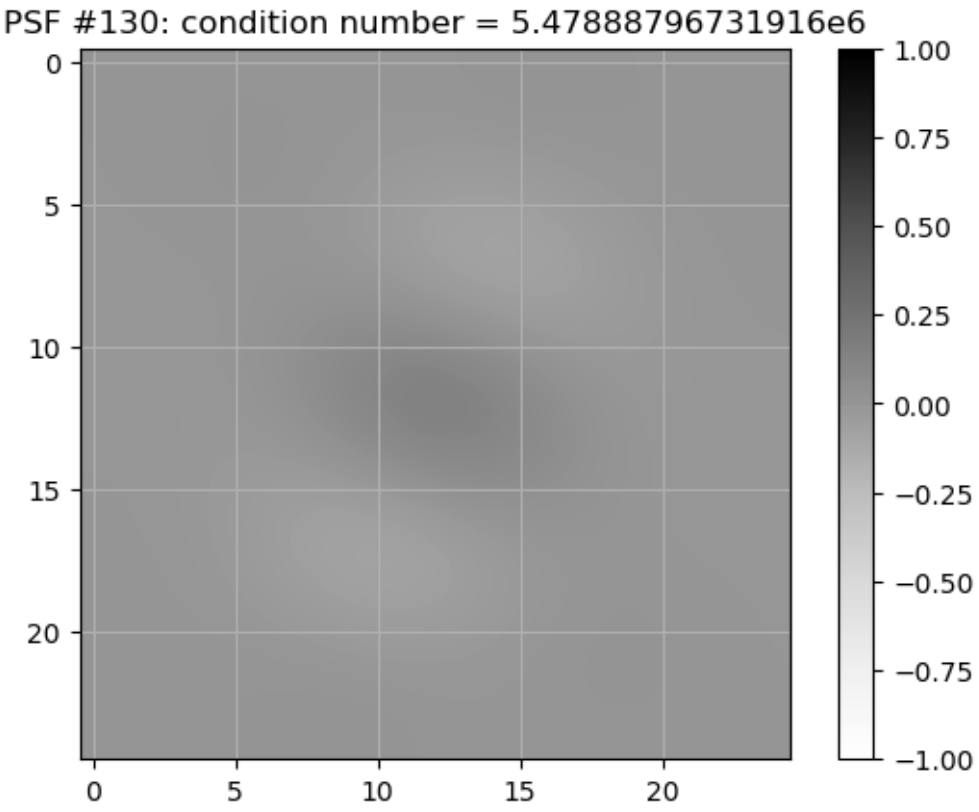
[17]:
```
o=36; l=12; n=25;

for ipsf in [9*13+1, 9*13+5, 9*13+9, 9*13+13]
    plot_psf_with_condition_number(ipsf)
end
```



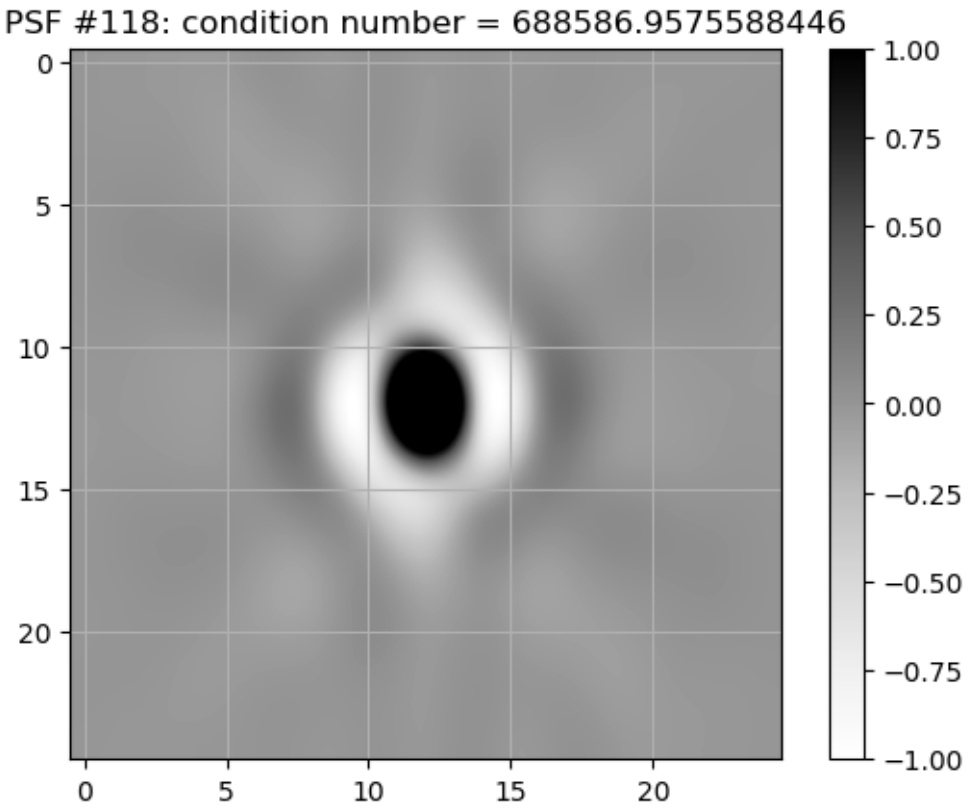

PSF #122: condition number = 595831.3929583202
0
5
10
15
20
0
5
10
15
20
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

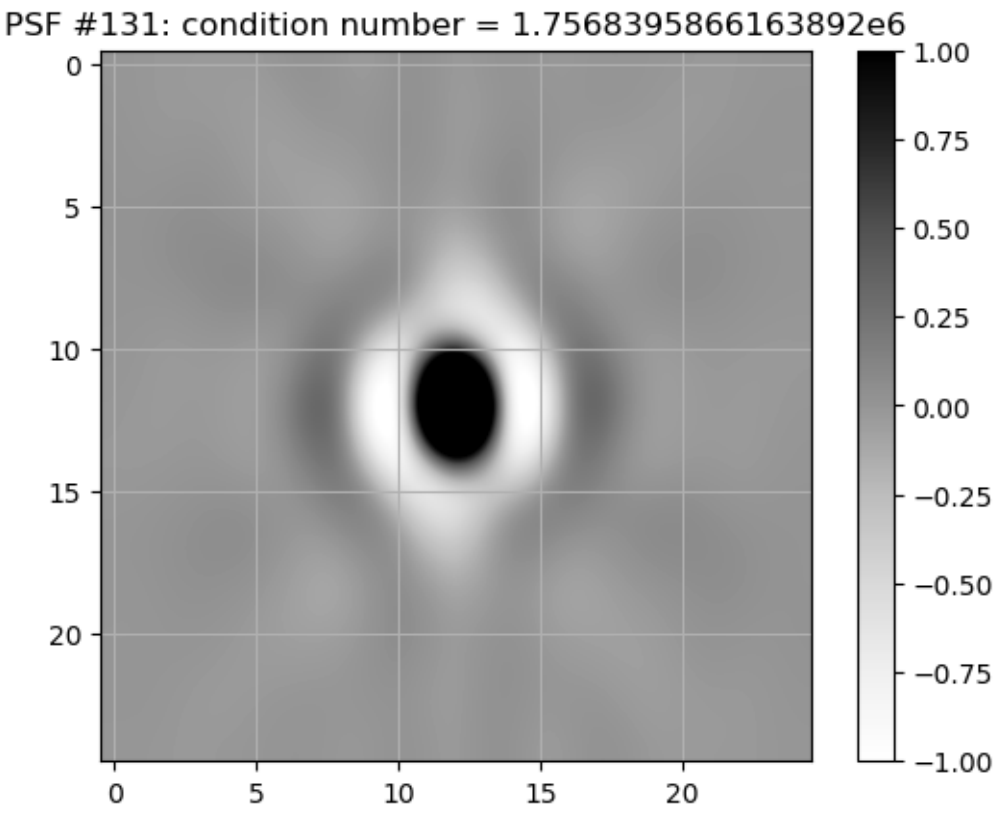

PSF #126: condition number = 4.990549477509275e7
0
5
10
15
20
0
5
10
15
20
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

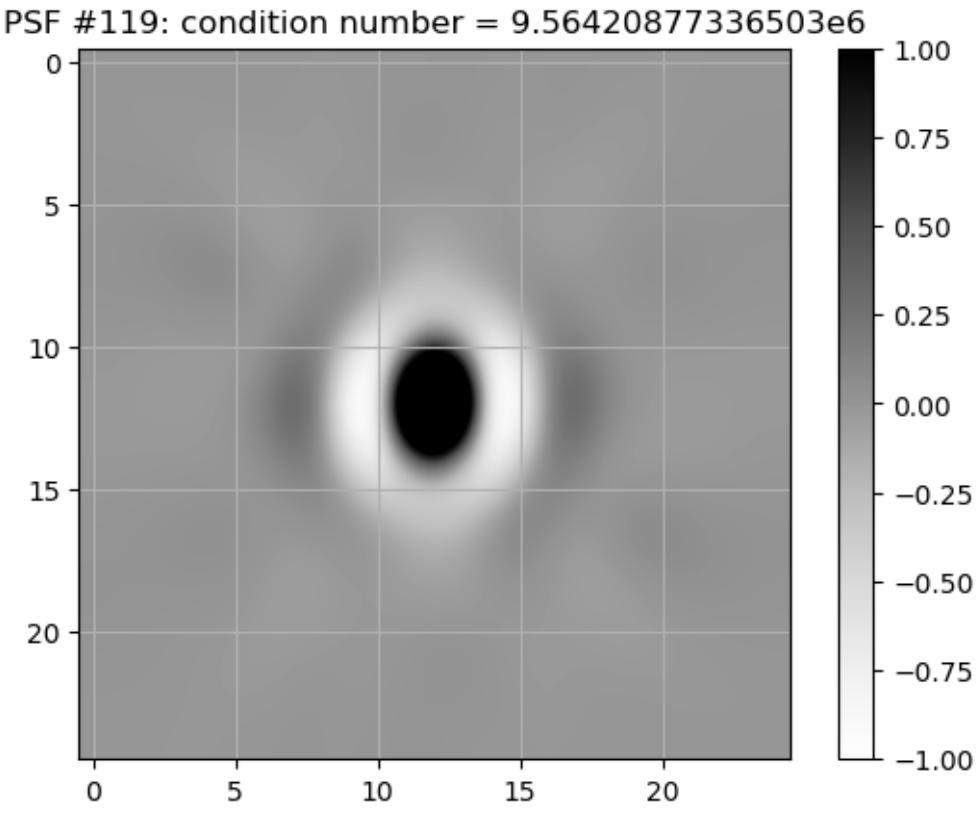


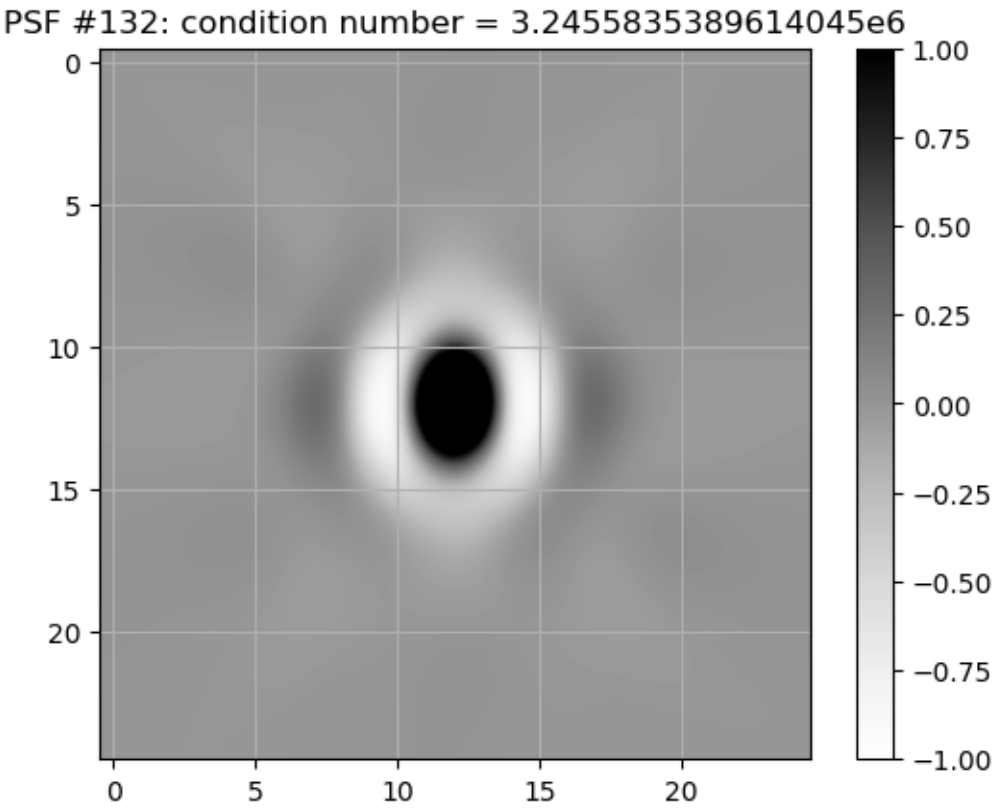
[18]:
```
o=36; l=12; n=25;

for ipsf in [9*13+1, 10*13+1, 9*13+2, 10*13+2]
    plot_psf_with_condition_number(ipsf)
end
```


PSF #118: condition number = 688586.9575588446
0
5
10
15
20
0
5
10
15
20
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


PSF #131: condition number = 1.7568395866163892e6
0
5
10
15
20
0
5
10
15
20
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


PSF #119: condition number = 9.564208773365O3e6
0
5
10
15
20
0
5
10
15
20
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00



1.3 Eigenvalues of 4x4 PSFs

Distinct PSFs

[25]:
```
o=36; l=12; n = 100

P1 = psfs.val[(o-l):(o+l),(o-l):(o+l), 9*13+1];
P2 = psfs.val[(o-l):(o+l),(o-l):(o+l), 9*13+5];
P3 = psfs.val[(o-l):(o+l),(o-l):(o+l), 9*13+9];
P4 = psfs.val[(o-l):(o+l),(o-l):(o+l), 9*13+13];
```

[26]:
```
G = psfeig.psf_to_matrix(n,P1,P2,P3,P4); varinfo(r"G")
```

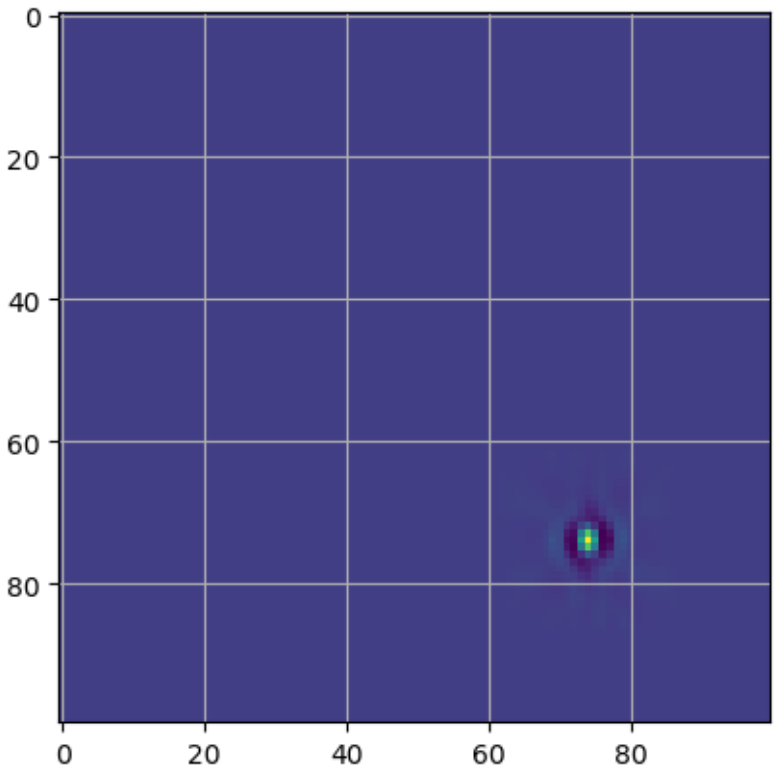
[26]:

name	size	summary
G	762.939 MiB	10000×10000 Array{Float64,2}

[27]:
```
cond(G)
```

[27]:
```
3.1523892698240 61e8
```

[28]:
```
imshow(reshape(G[(75-1)*100+75,:],100,100)); grid()
```



Neighbour PSFs

[29]:
```
o=36; l=12; n = 100

P1 = psfs.val[(o-l):(o+l),(o-l):(o+l),  9*13+1];
P2 = psfs.val[(o-l):(o+l),(o-l):(o+l), 10*13+1];
P3 = psfs.val[(o-l):(o+l),(o-l):(o+l),  9*13+2];
P4 = psfs.val[(o-l):(o+l),(o-l):(o+l), 10*13+2];
```

[30]:
```
G = psfeig.psf_to_matrix(n,P1,P2,P3,P4); varinfo(r"G")
```

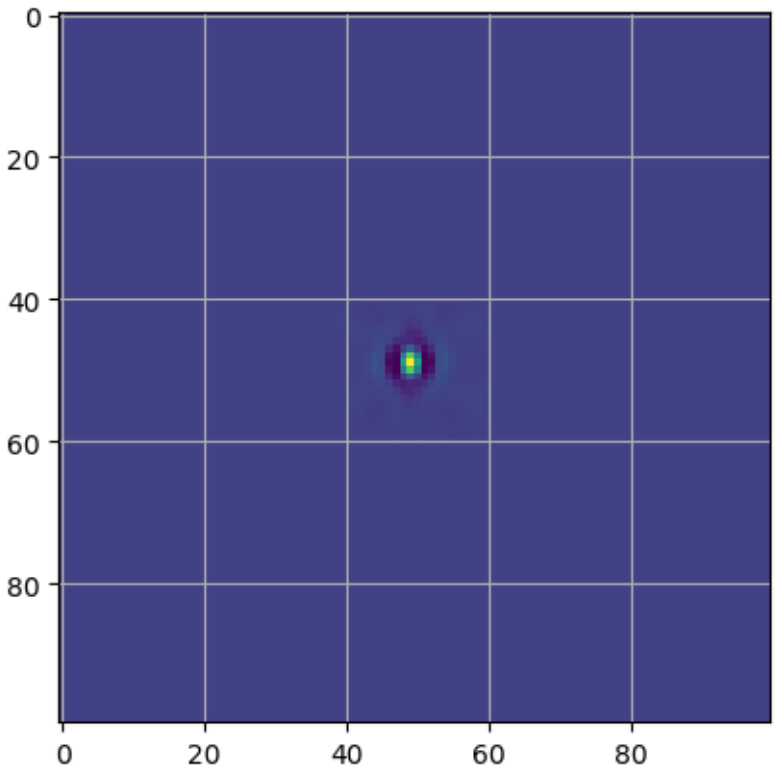
[30]:

name	size	summary
G	762.939 MiB	10000×10000 Array{Float64,2}

[31]:
```
cond(G)
```

[31]: 1.8344483898310825e7

[32]:
```
imshow(reshape(G[(50-1)*100+50,:],100,100)); grid()
```



1.4 PSF Eigenvalue Developmment

[39]:
```
n = 25;
P = psfeig.bluker(12,1.0);
```

[107]:
```
G = psfeig.psf_to_matrix(n,P); varinfo(r"G")
G2 = psfeig.psf_to_matrix(n,P,P,P,P); varinfo(r"G2")
G3 = psfeig.psf_to_matrix(n,P,0.5*P,0.2*P,0.1*P); varinfo(r"G3")
```

[107]:

name	size	summary
G3	2.980 MiB	625×625 Array{Float64,2}

[109]:
```
cond(G)
```

[109]: 4167.911018197733

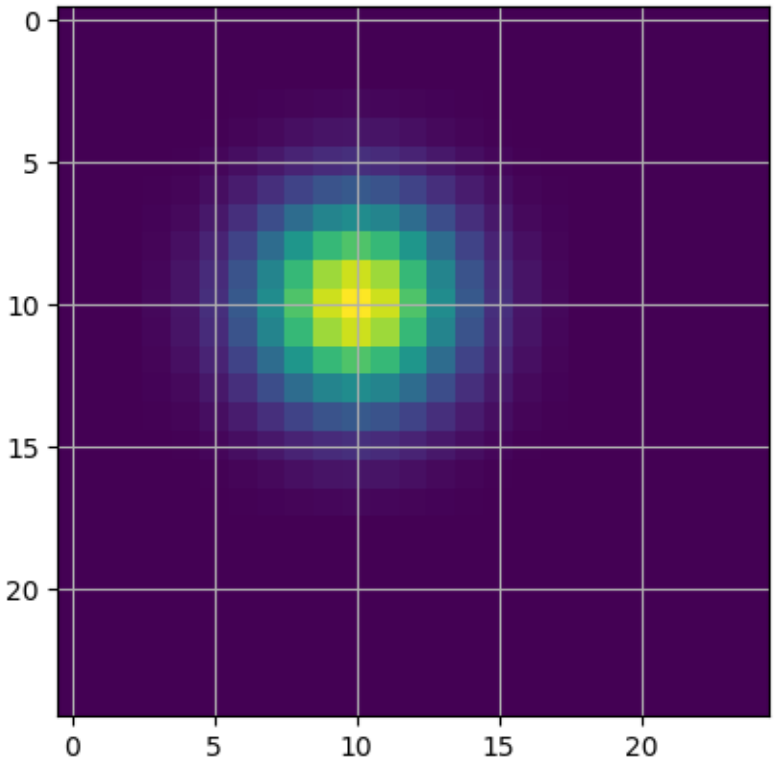
[110]:
```
cond(G2)
```

[110]: 4167.911018197979

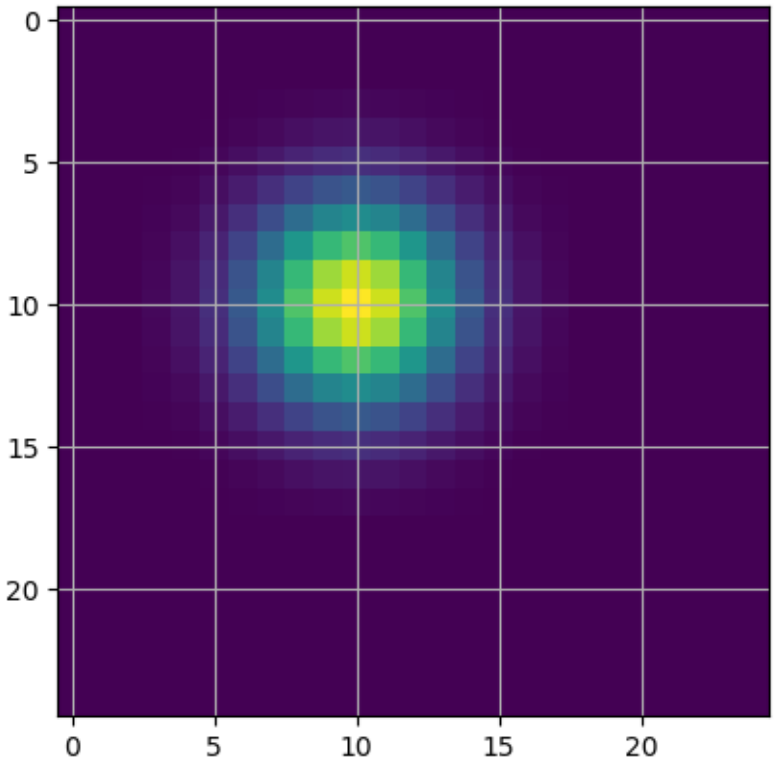
[111]:
```
cond(G3)
```

[111]: 11380.202021198646

[105]:
```
imshow(reshape(G[(11-1)*25+11,:],25,25)); grid()
```



[103]:
```
imshow(reshape(G4[(11-1)*25+11,:],25,25)); grid()
```



[40]:
```
imshow(P)
```

[40]: PyObject <matplotlib.image.AxesImage object at 0x7fbe251ea410>

[41]:
```
findmax(P)
```

[41]: (0.15915494309189535, CartesianIndex(13, 13))

[42]:
```
size(P)
```

[42]: (25, 25)

1.5 Eigenvalue Timing

n	Time	Size
64	27s	128Mb
96	162s	324Mb

[36]:
```
n=100; x = rand(Float32, n*n,n*n); varinfo(r"x")
```

[36]:

name	size	summary
x	381.470 MiB	10000×10000 Array{Float32,2}

[83]:
```
@time v = eigvals(x);
```

```
165.460979 seconds (899.31 k allocations: 428.171 MiB, 0.01% gc time)
```

[84]:
```
x = nothing
```

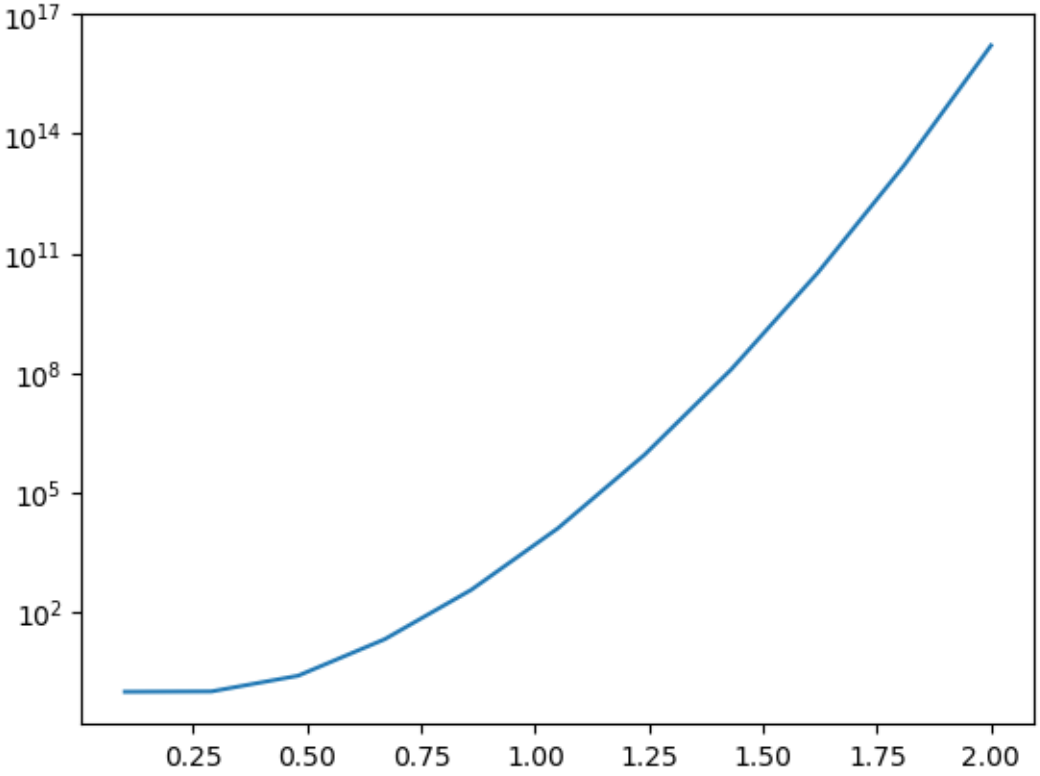
[85]:
```
GC.gc()
```

1.6 Evaluate the condition number of blur matrix

[35]:
```
N = 50;
k = 12;

nn = 11;
c = zeros(nn);
sigma = range(0.1, stop=2.0, length=nn);
@time for i = 1:nn
    P = psfeig.bluker(k,sigma[i]);
    G = psfeig.psf_to_matrix(N,P);
    c[i] = cond(G);
end

semilogy(sigma,c);
```




```
47.223515 seconds (81.13 k allocations: 1.044 GiB, 0.16% gc time)
```

[ ]: