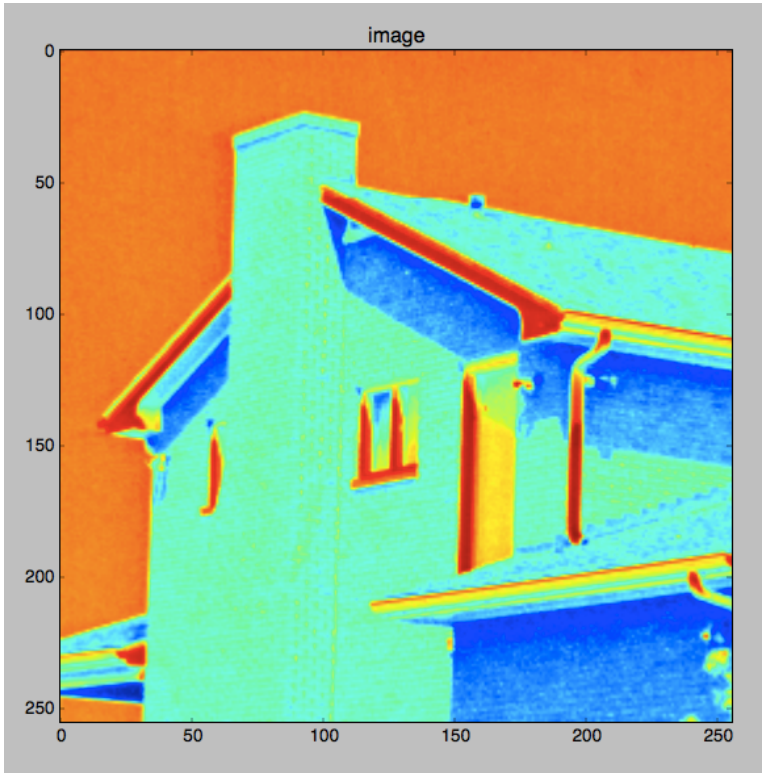
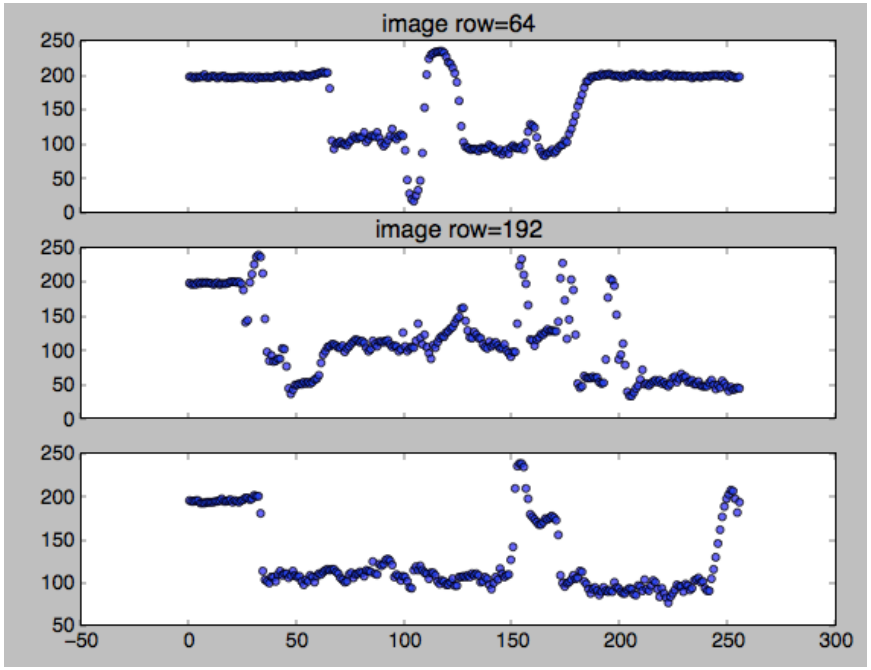
plots from the phase congruency algorithm

followed by a filter in progress, meant for use in calculating corners using curvature (needs 1st and 2nd derives)

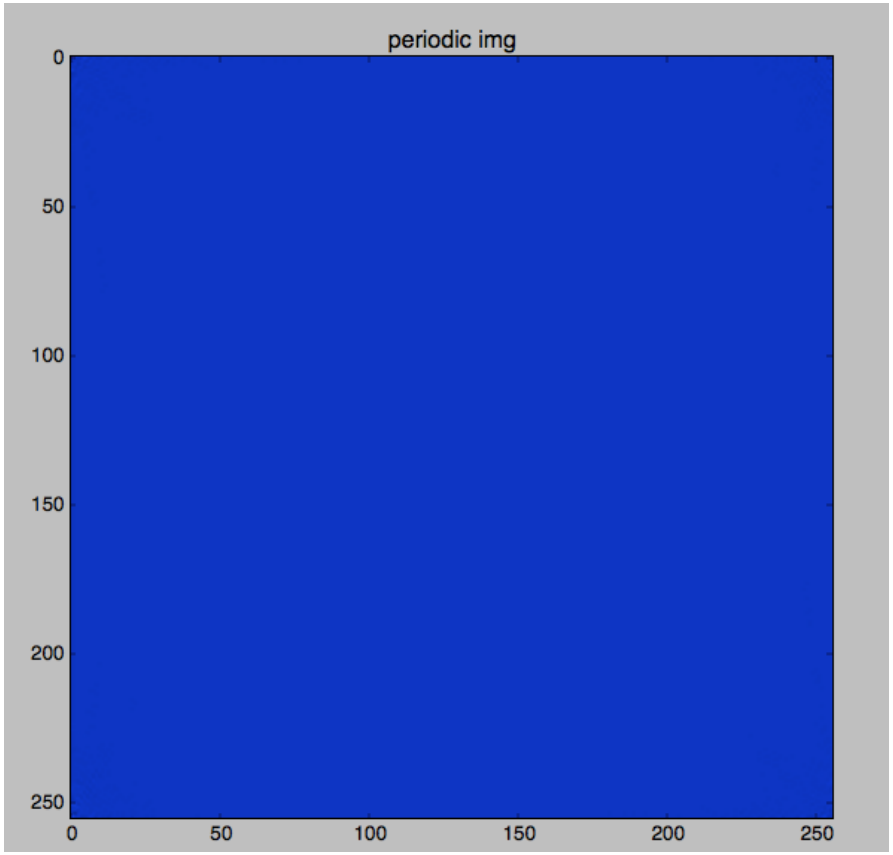
house.gif

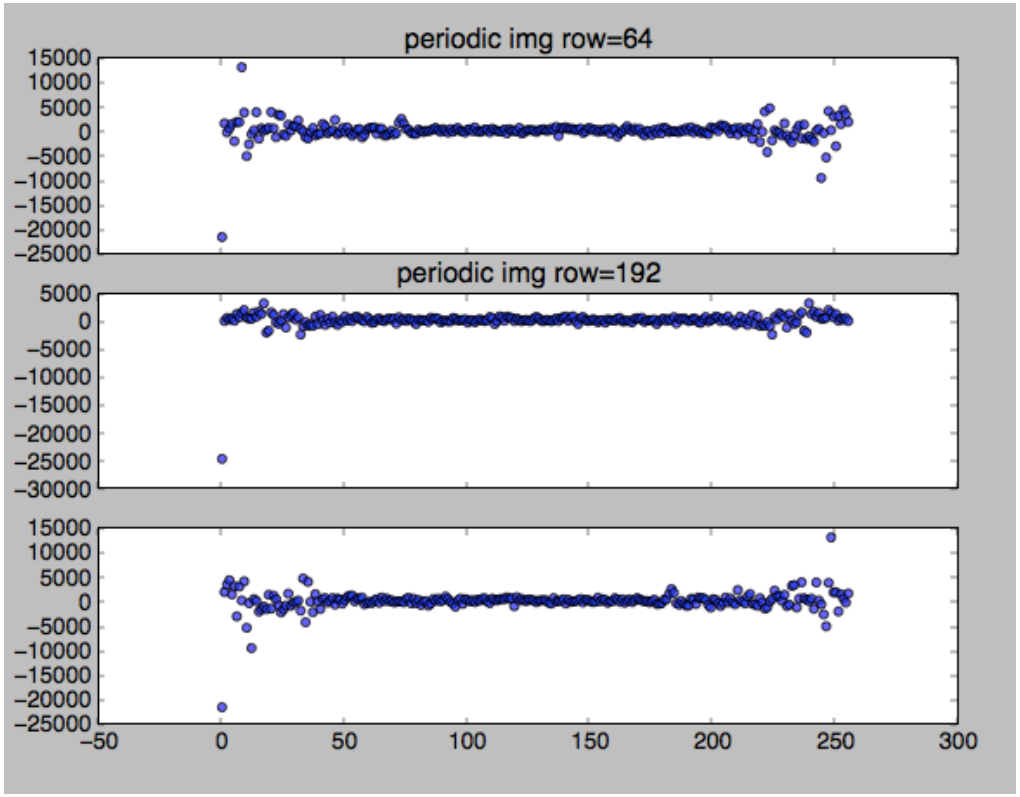




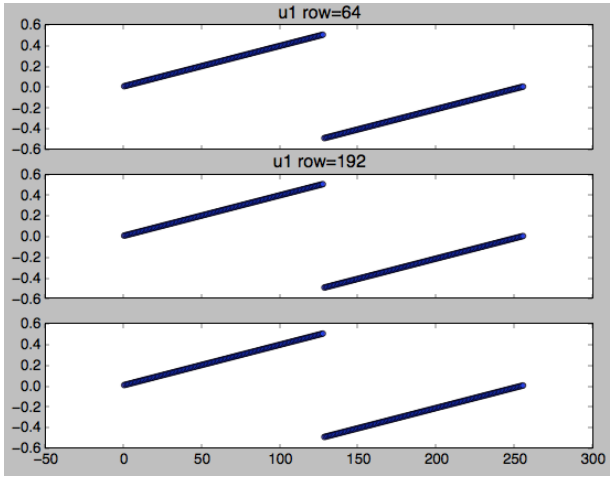
periodic FFT of img

```
_, IM = perfft2(img)
```

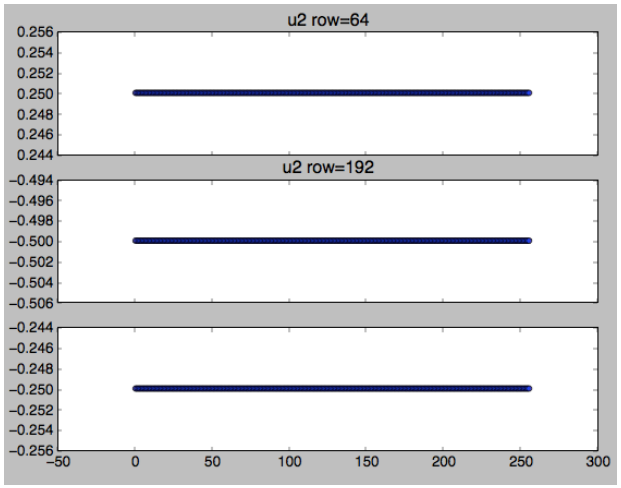




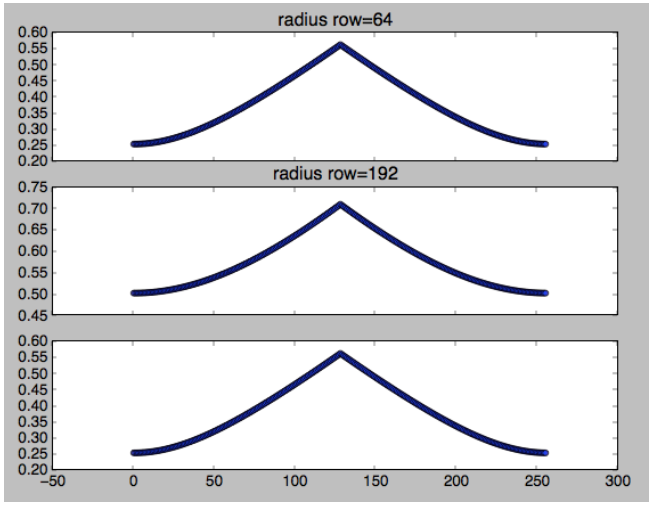
u1

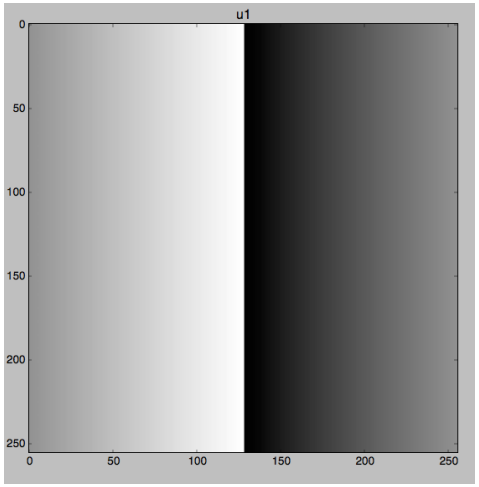


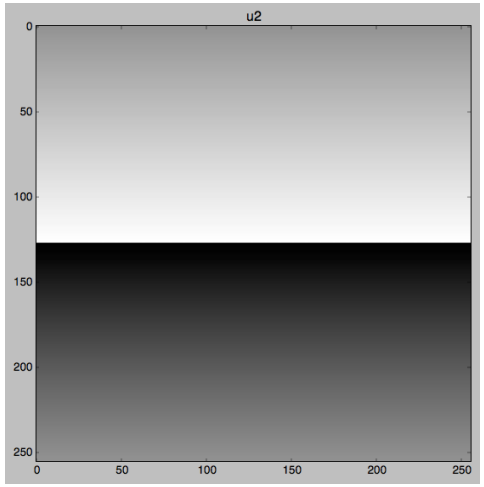
u2

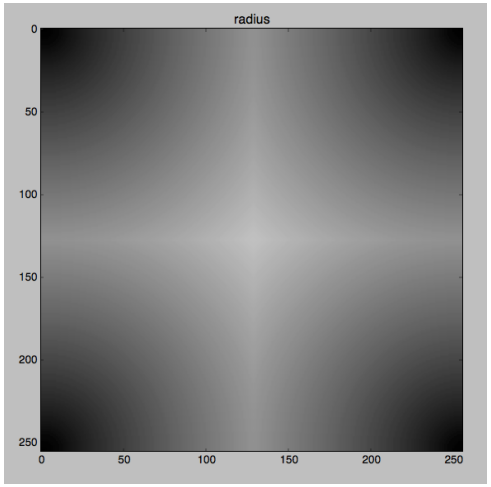


radius

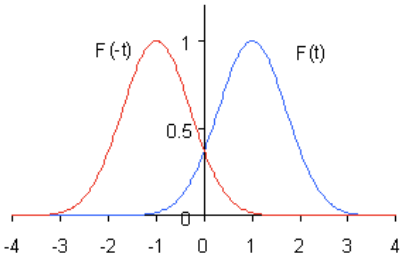


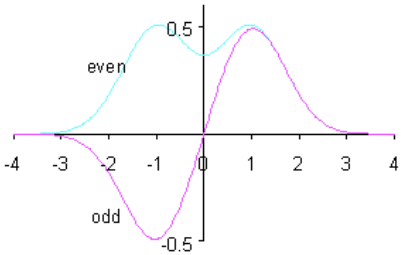






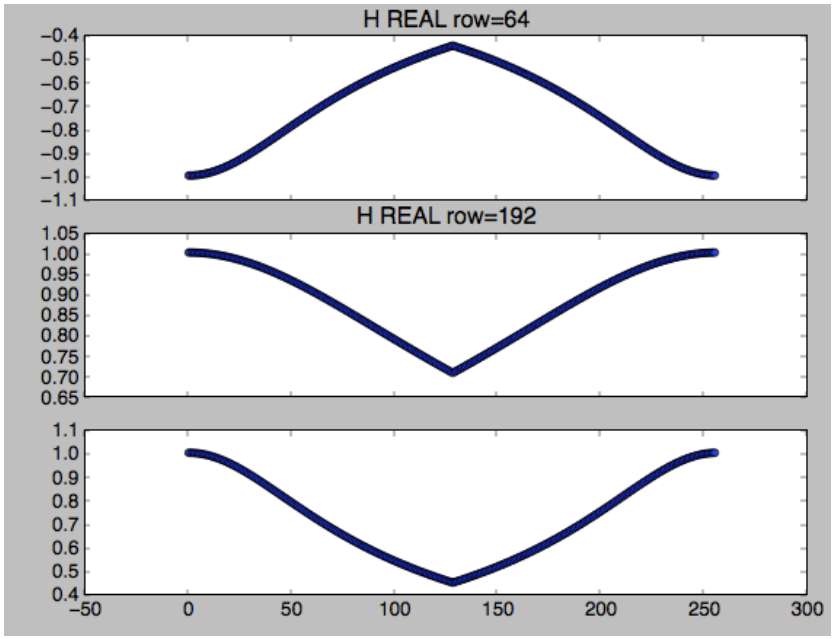
$F(t) = \exp[-(t - 1)^2]$

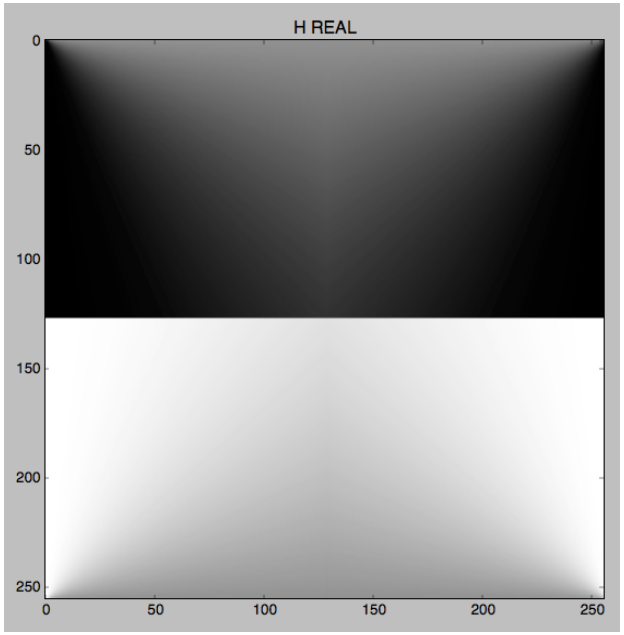




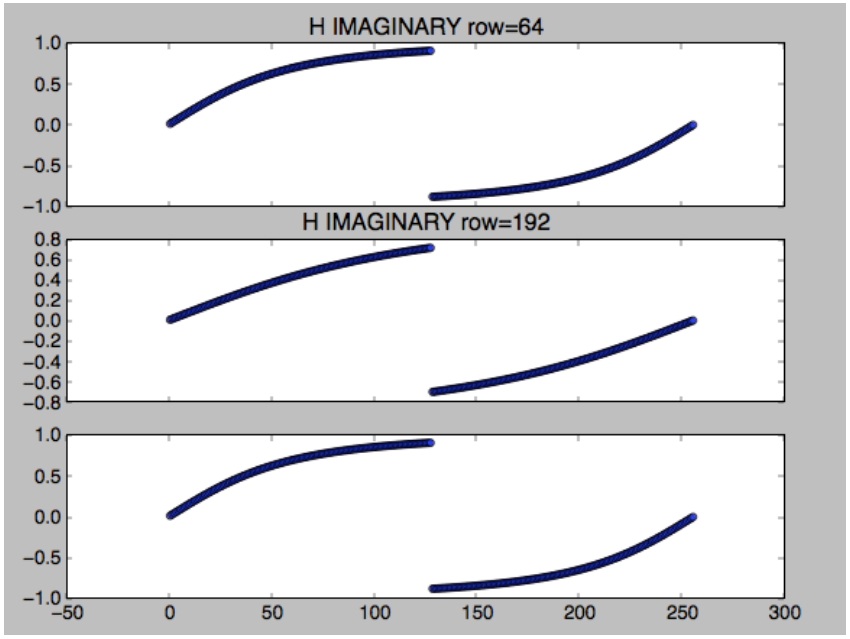
h1 = real(**H**)

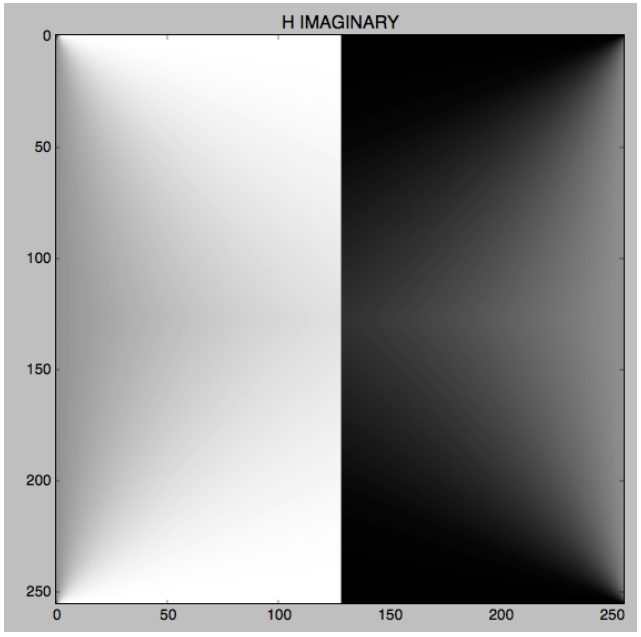
```
H = (1j * u1 - u2) / radius
```



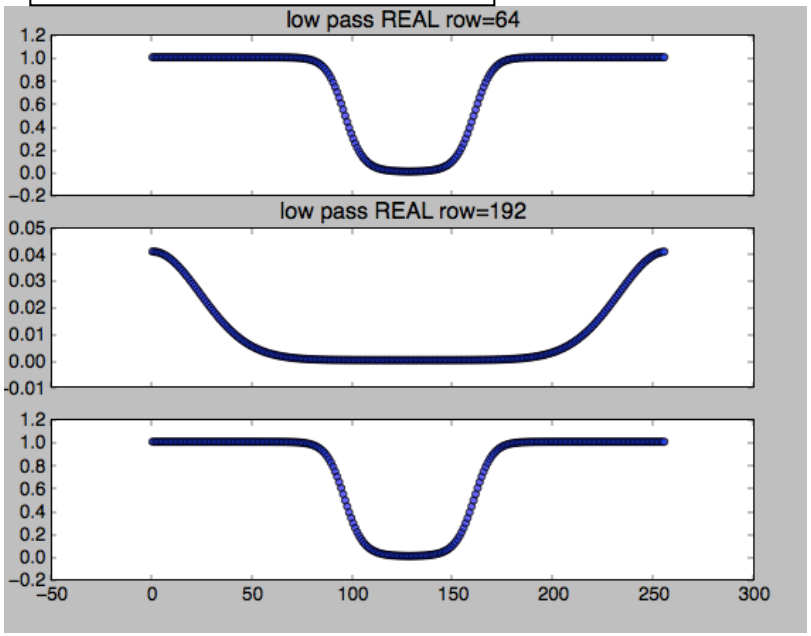


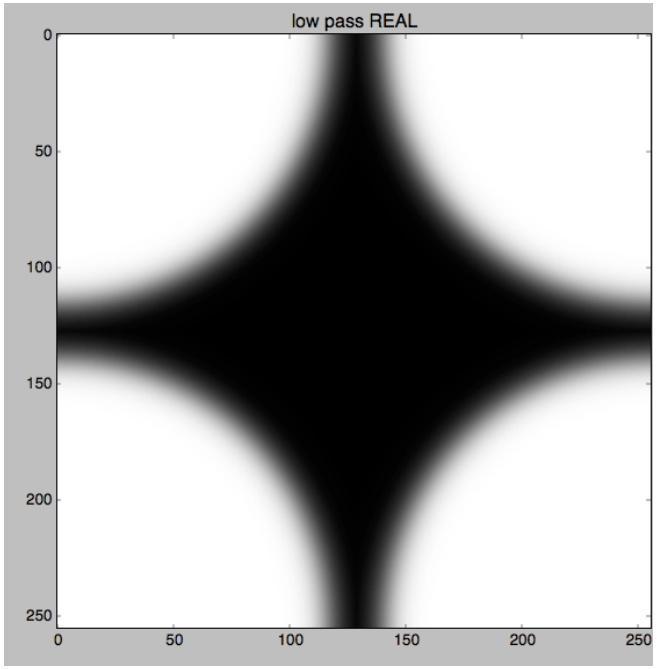
h2 = imaginary(**H**)



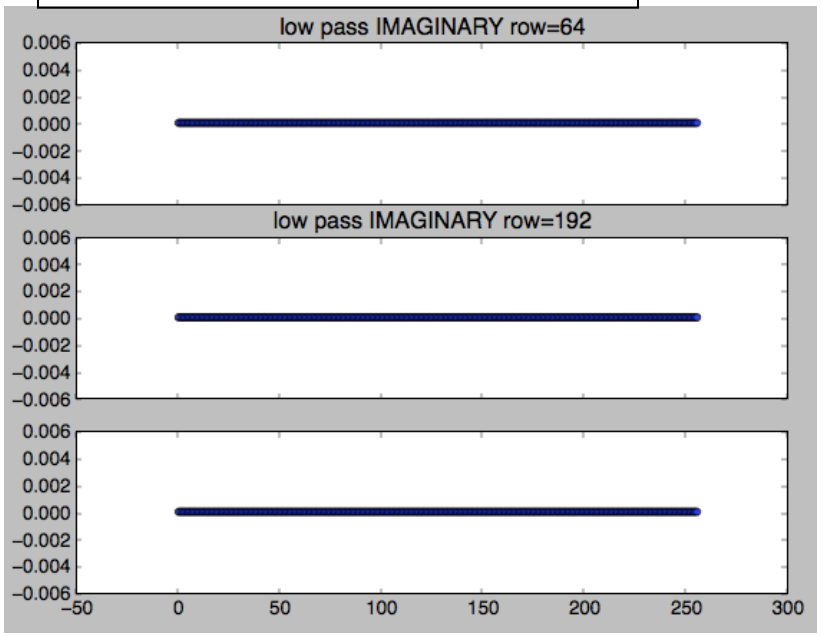


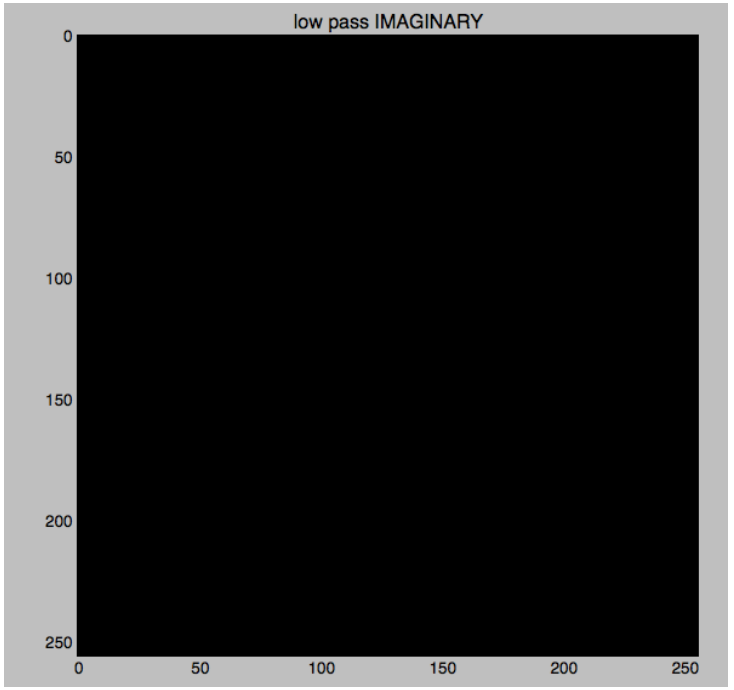
real(**low pass filter**)



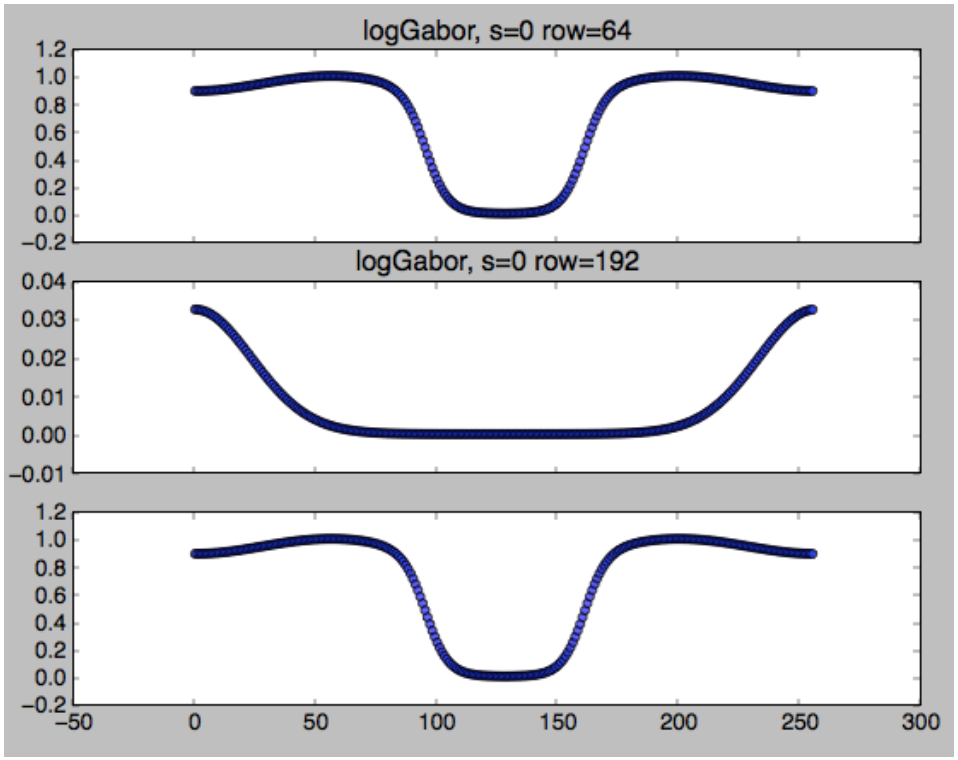


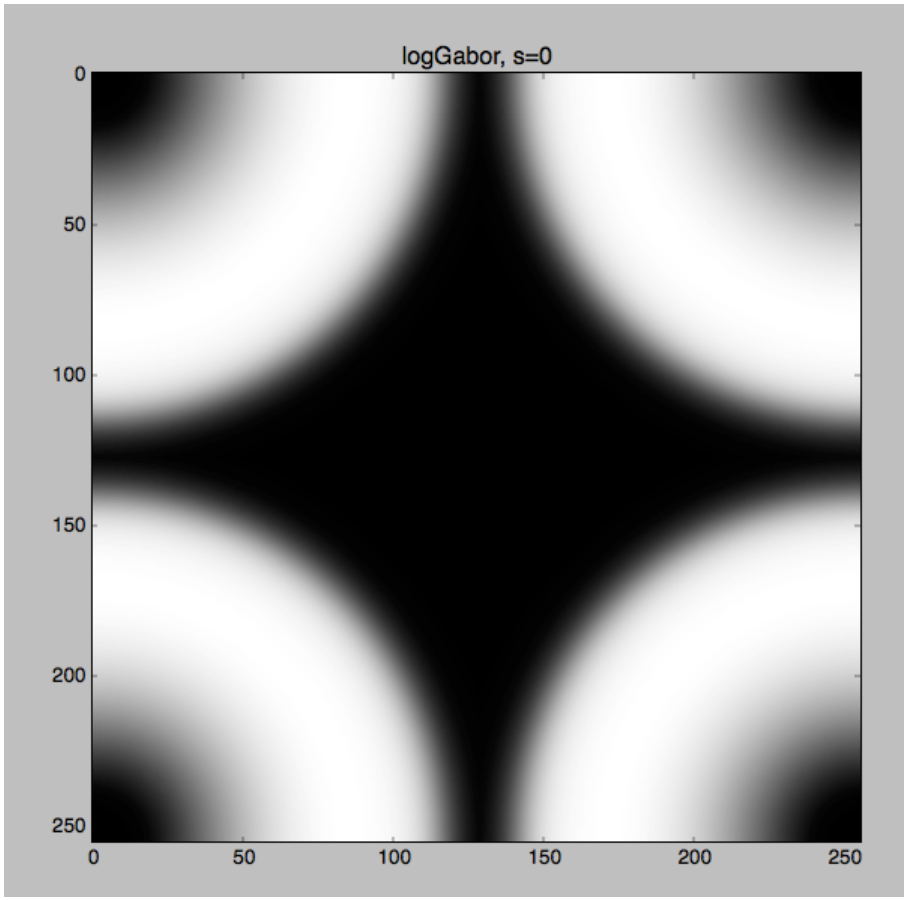
imaginary(**low pass filter**)



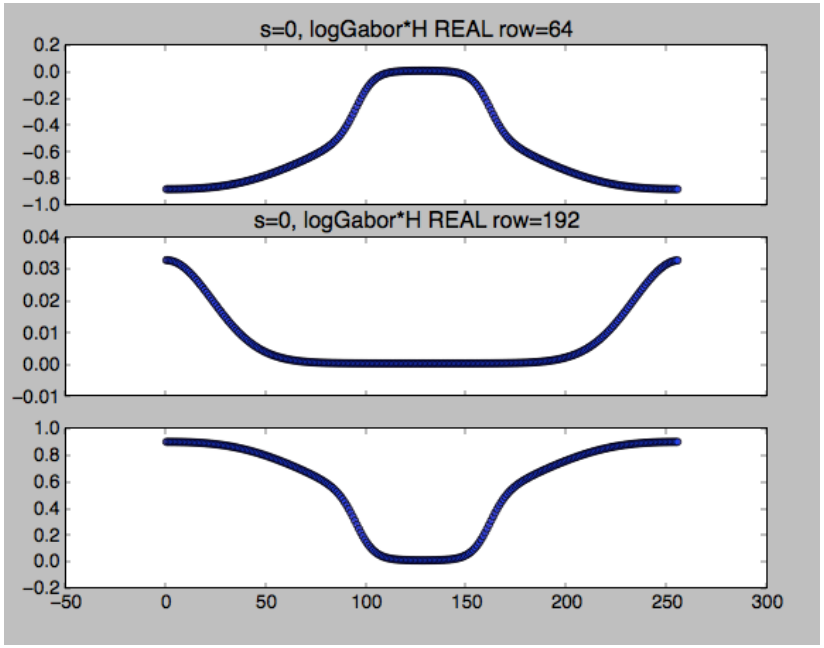


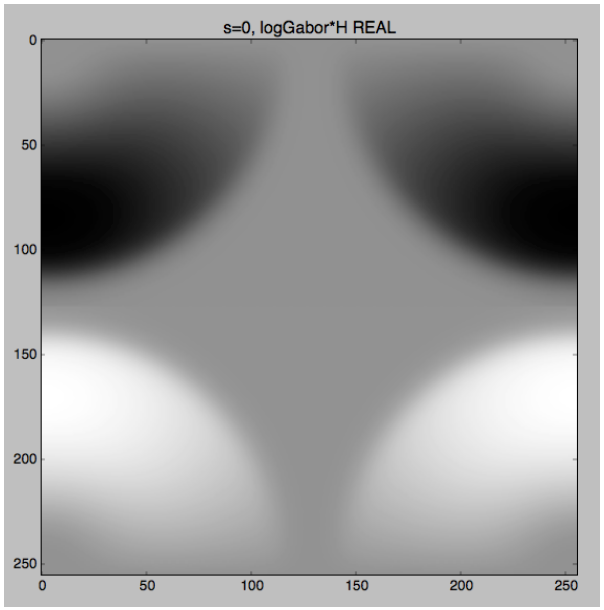
s=0, logGabor * low pass filter

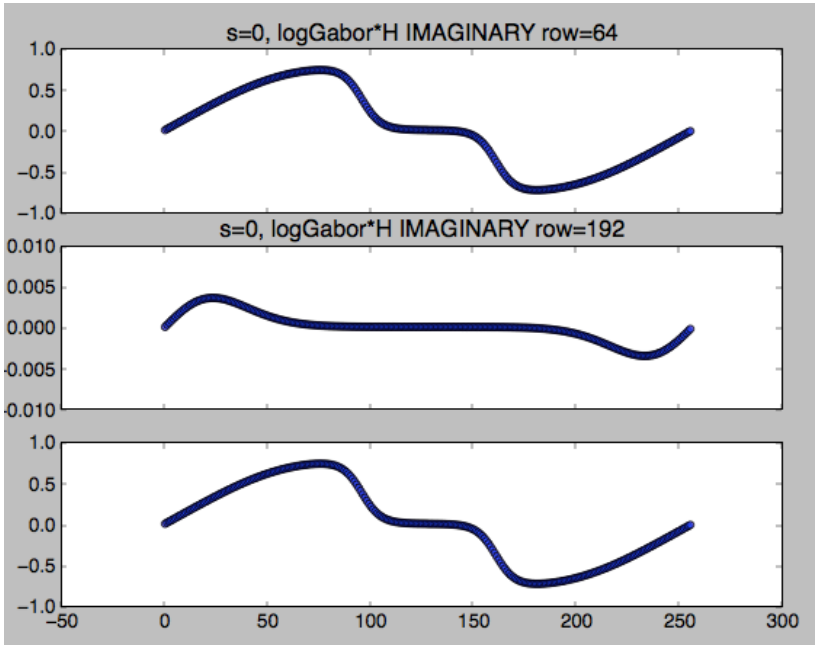


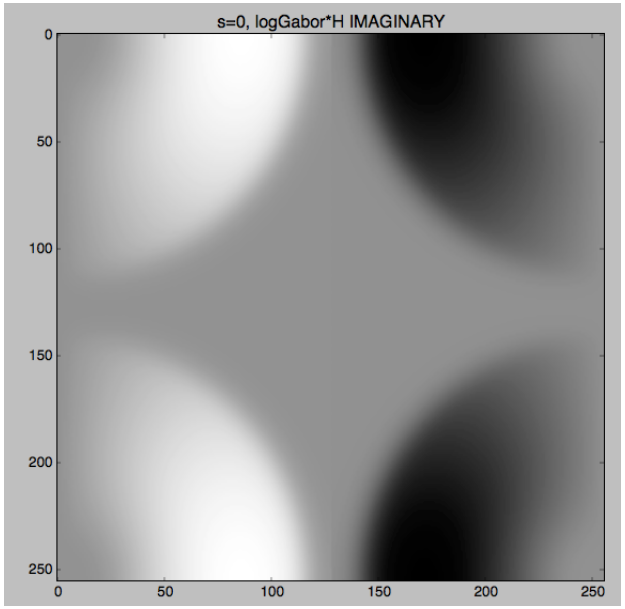


s=0, logGabor * low pass filter * H

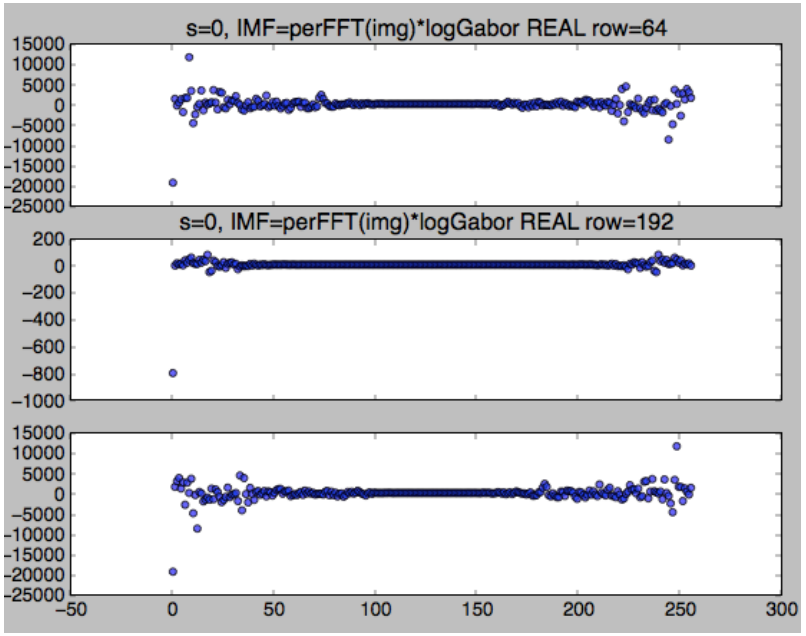


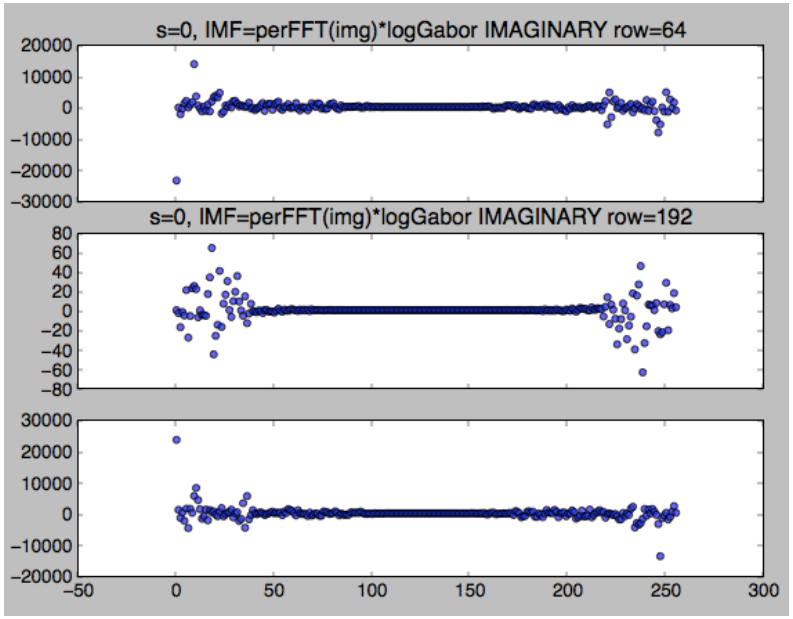


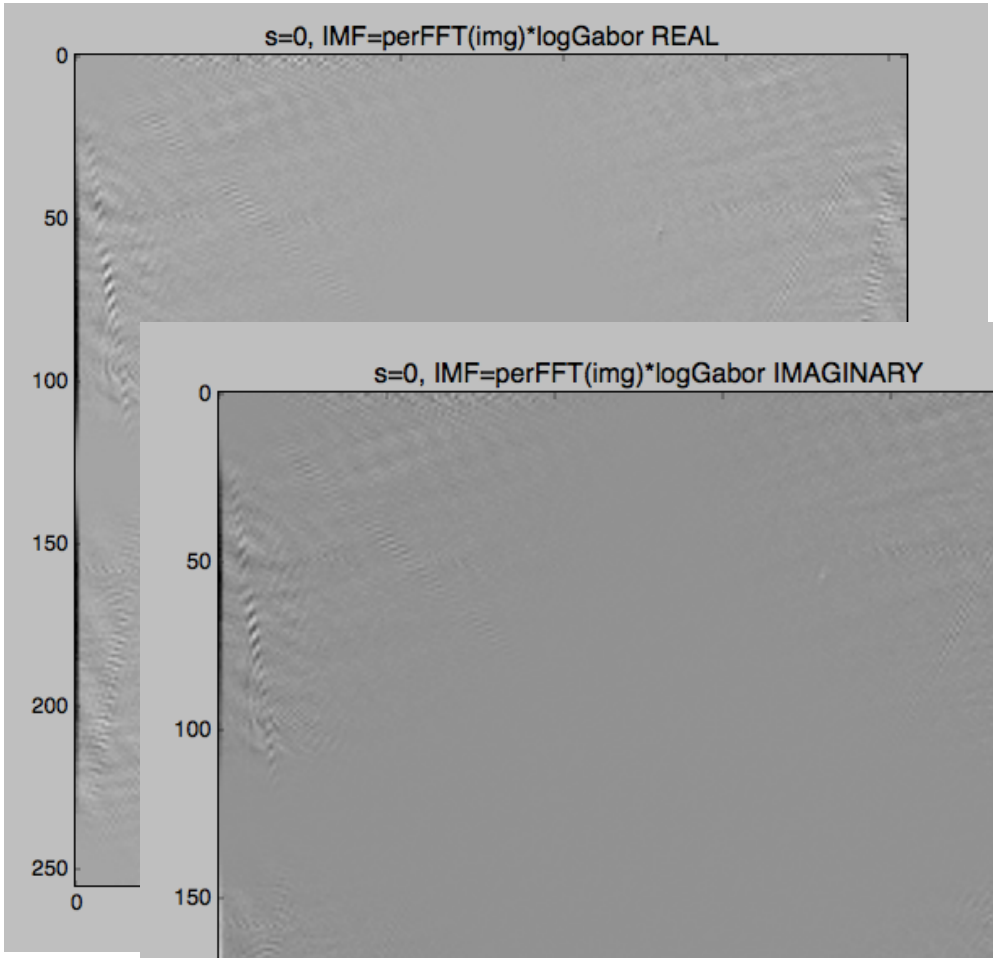


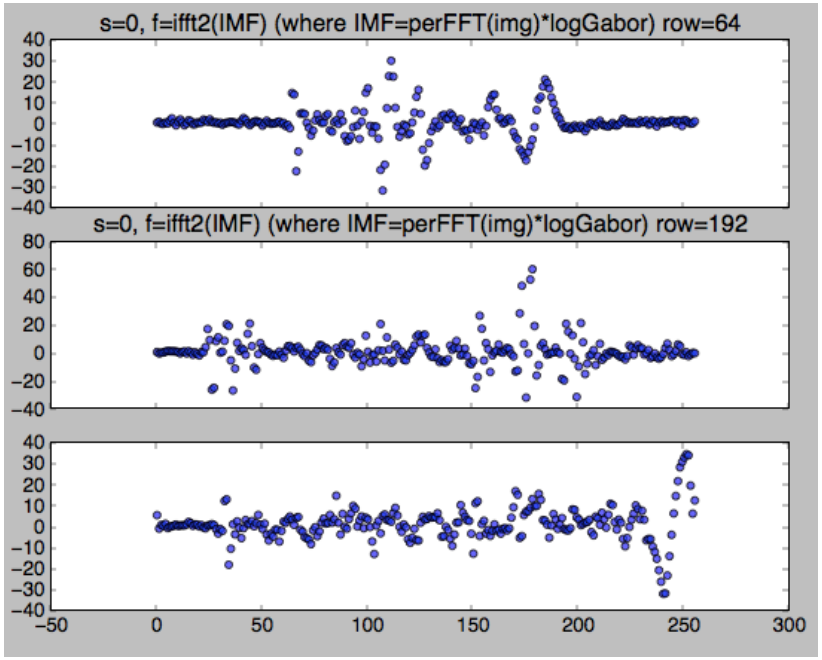


s=0, IMF=perFFT(img)*logGabor

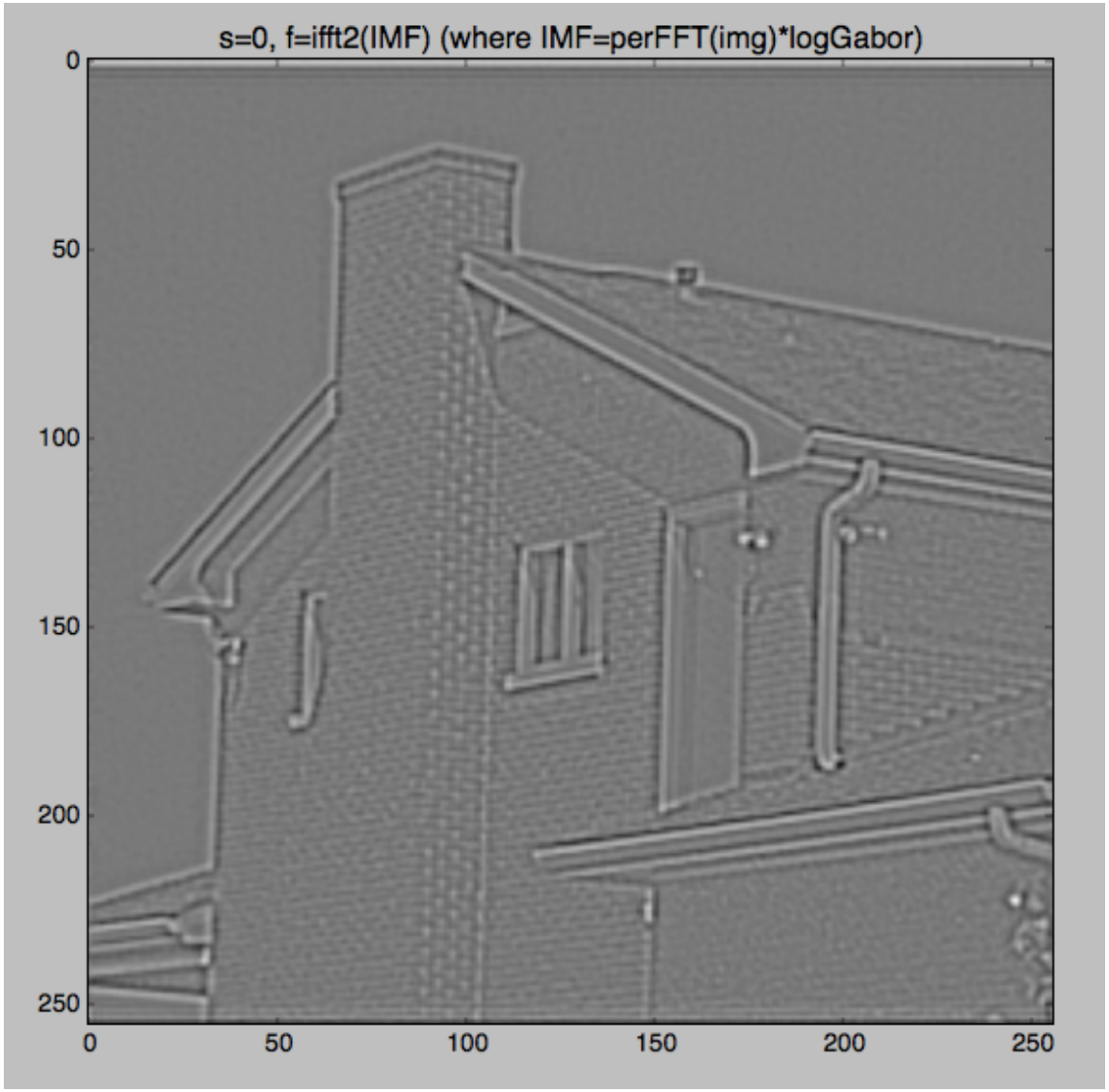


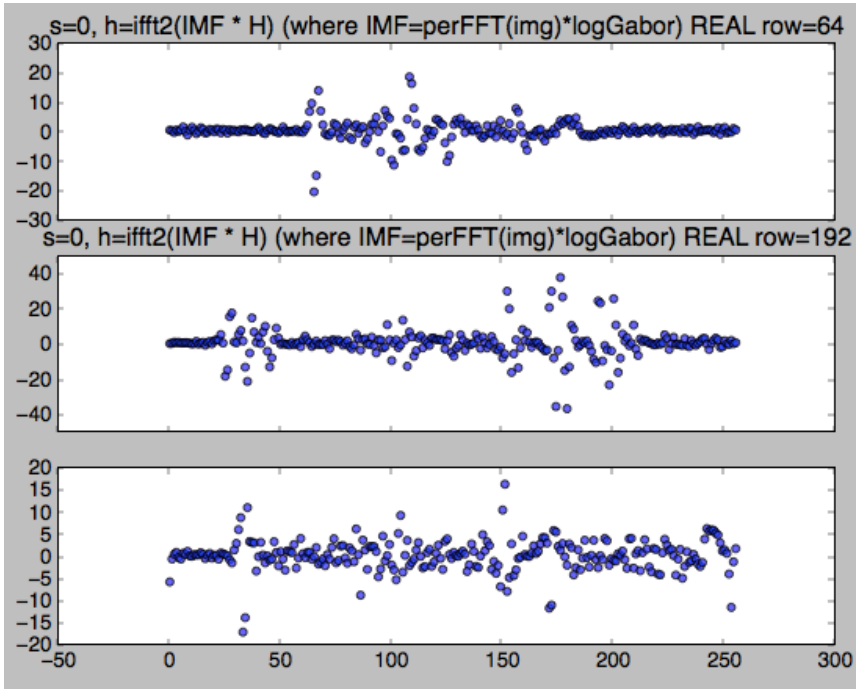




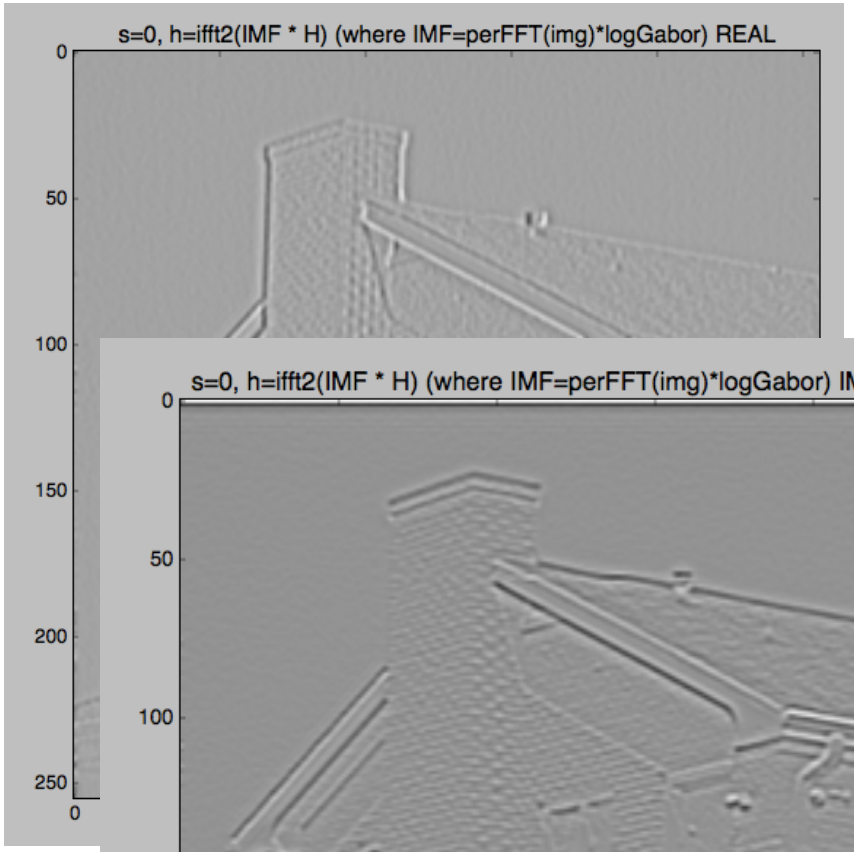


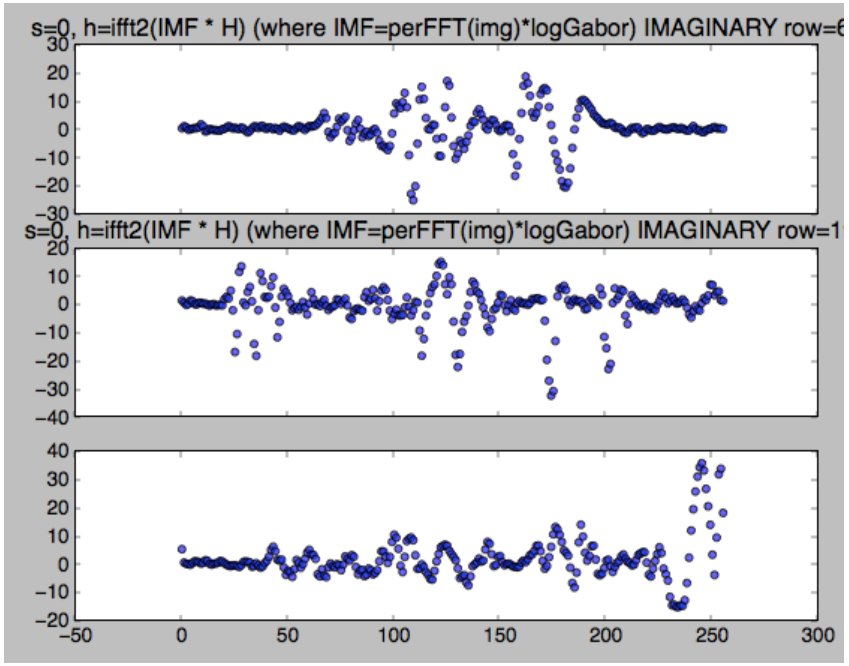

```
s=0, f=ifft2(IMF)
(where IMF=perFFT(img)*logGabor)
```



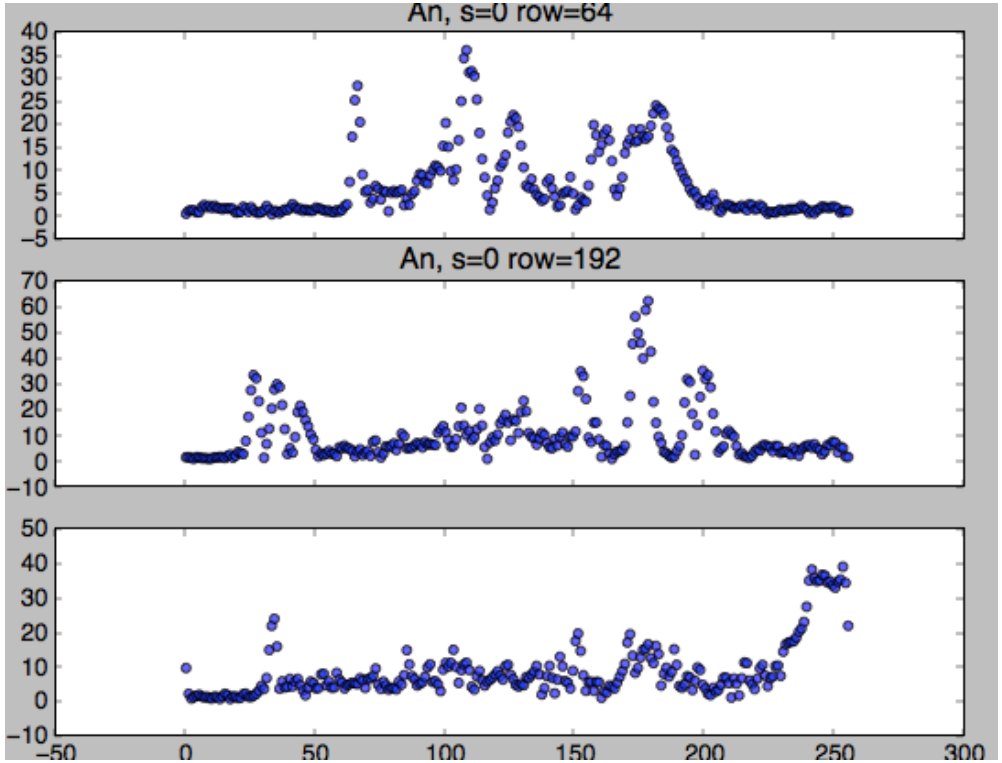


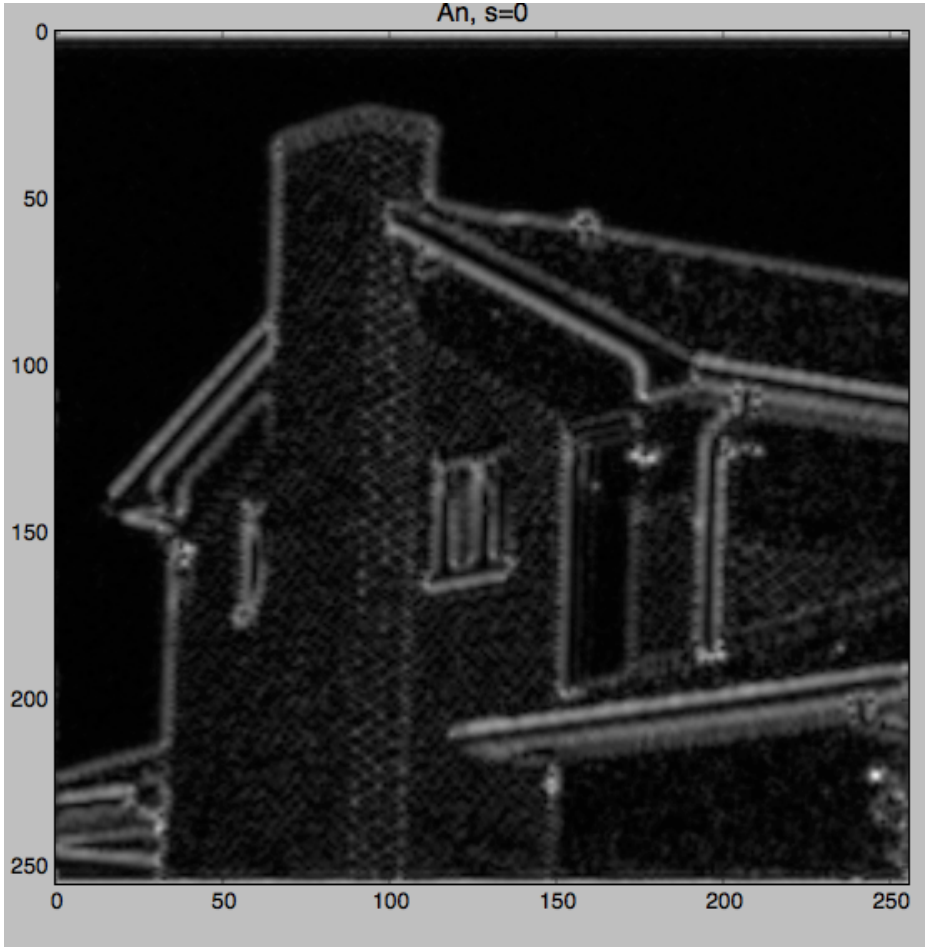
s=0, h=ifft2(IMF * H)
(where IMF=perFFT(img)*logGabor)



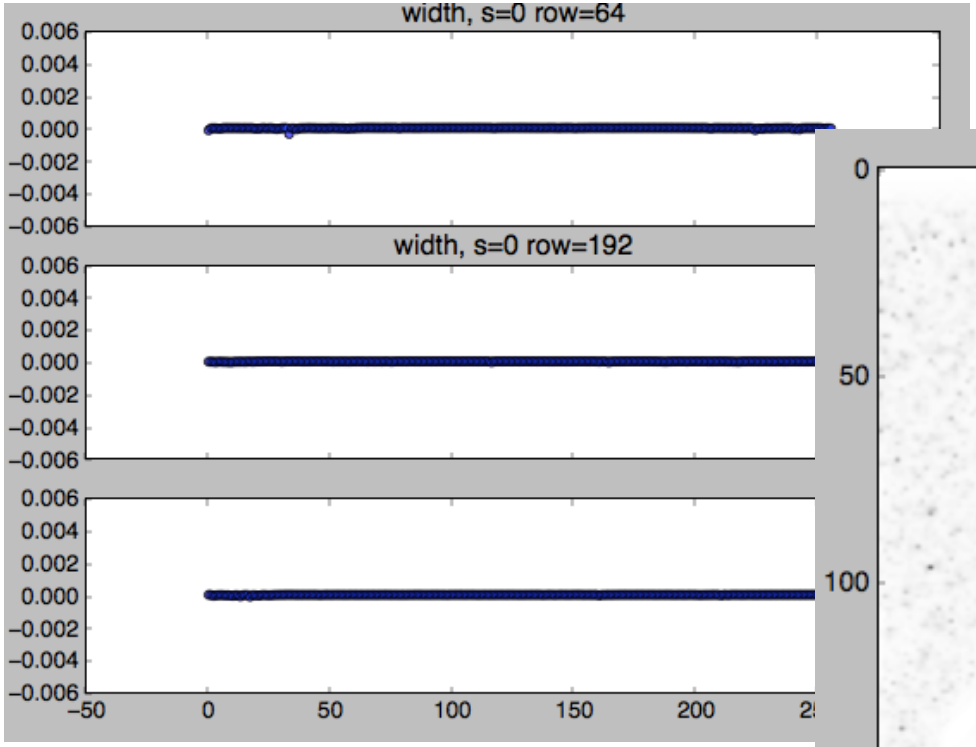


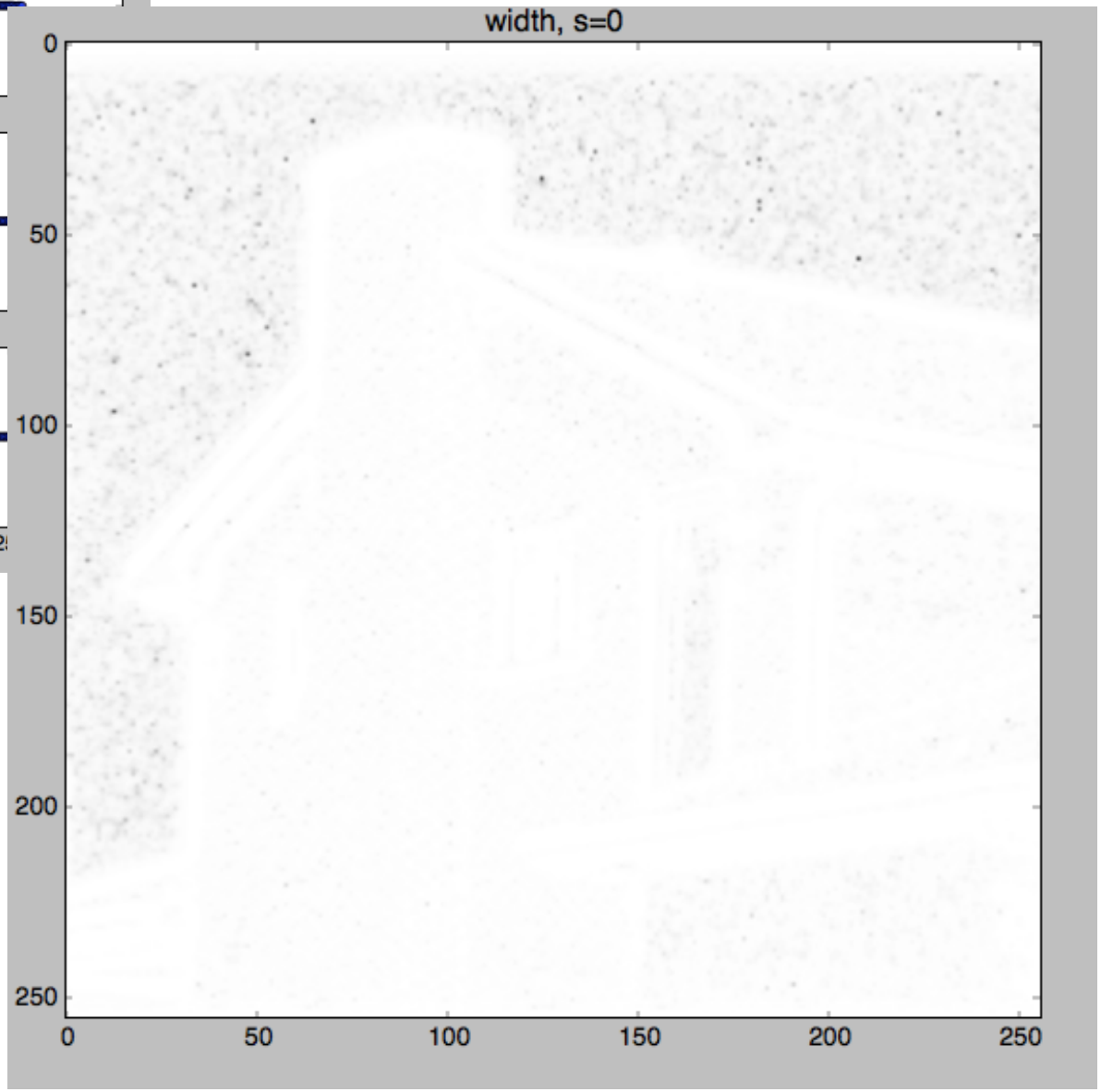
```
s=0, An = sqrt(f * f + h1 * h1 + h2 * h2)
(where f=ifft2(perFFT(img)*logGabor)
   and h=ifft2(perFFT(img)*logGabor * H) )
```



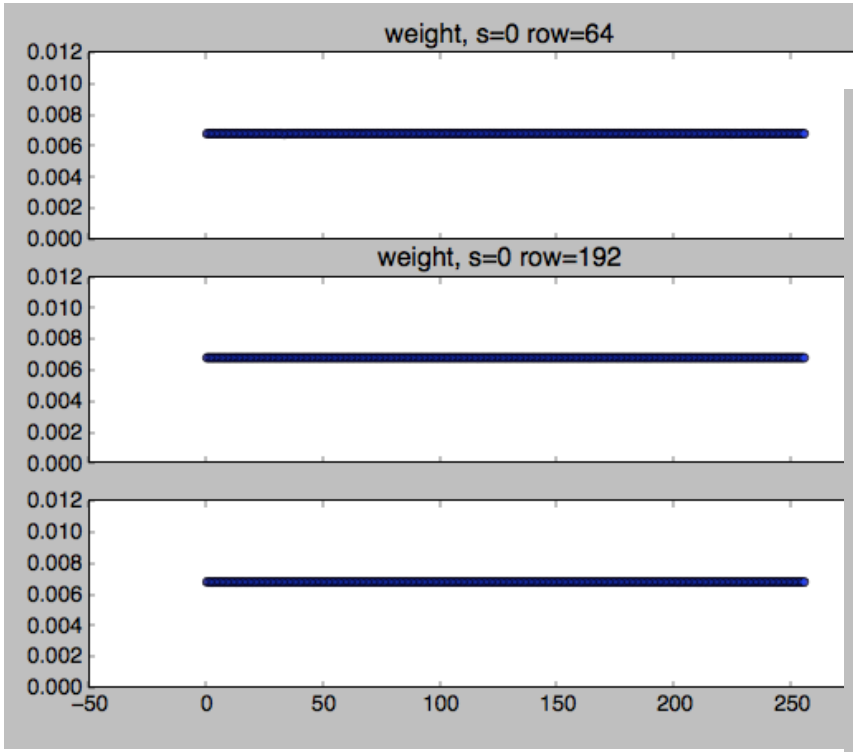


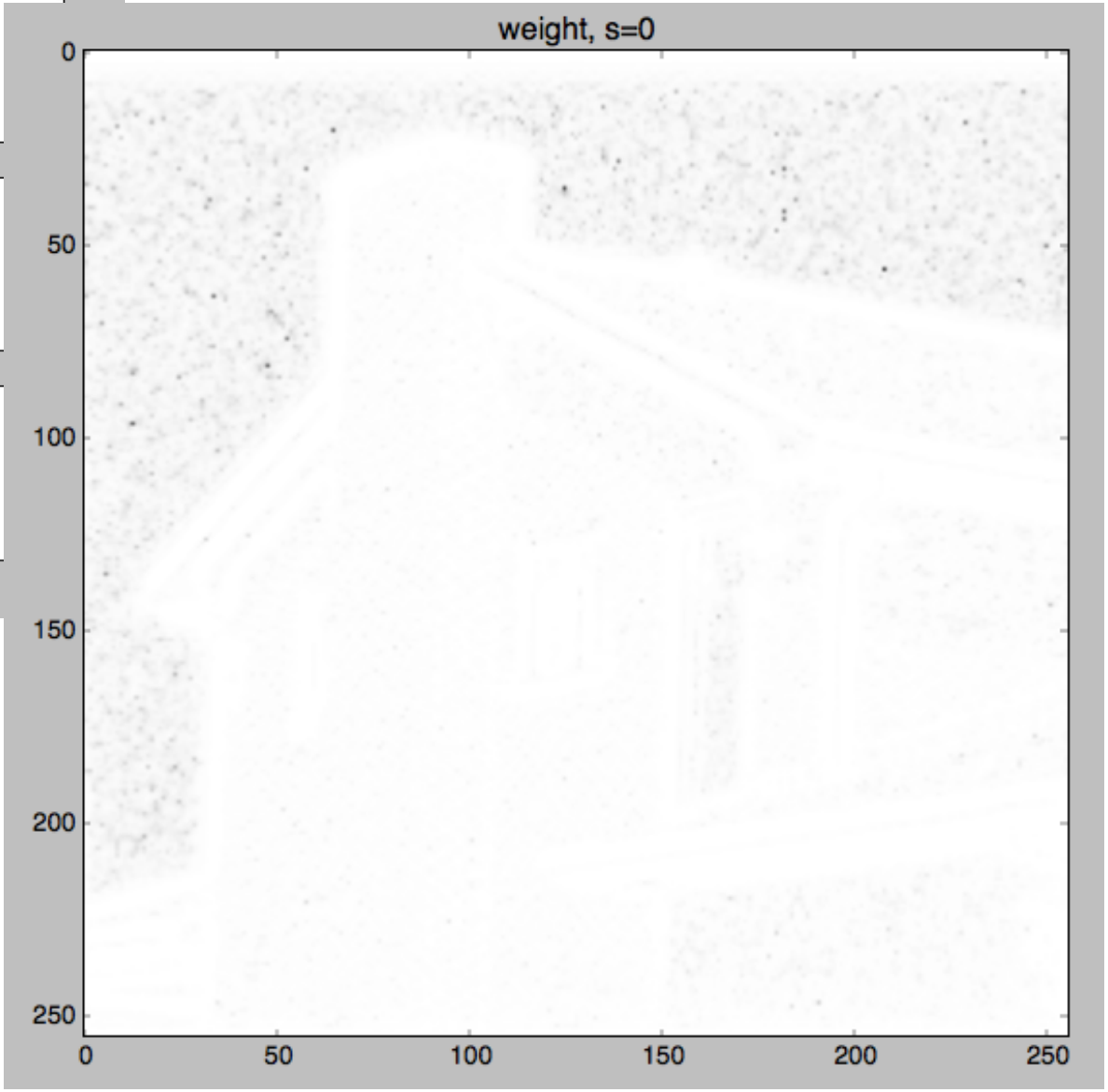
s=0, **width** = (sumAn / (maxAn + epsilon) – 1.) / (nscale – 1)



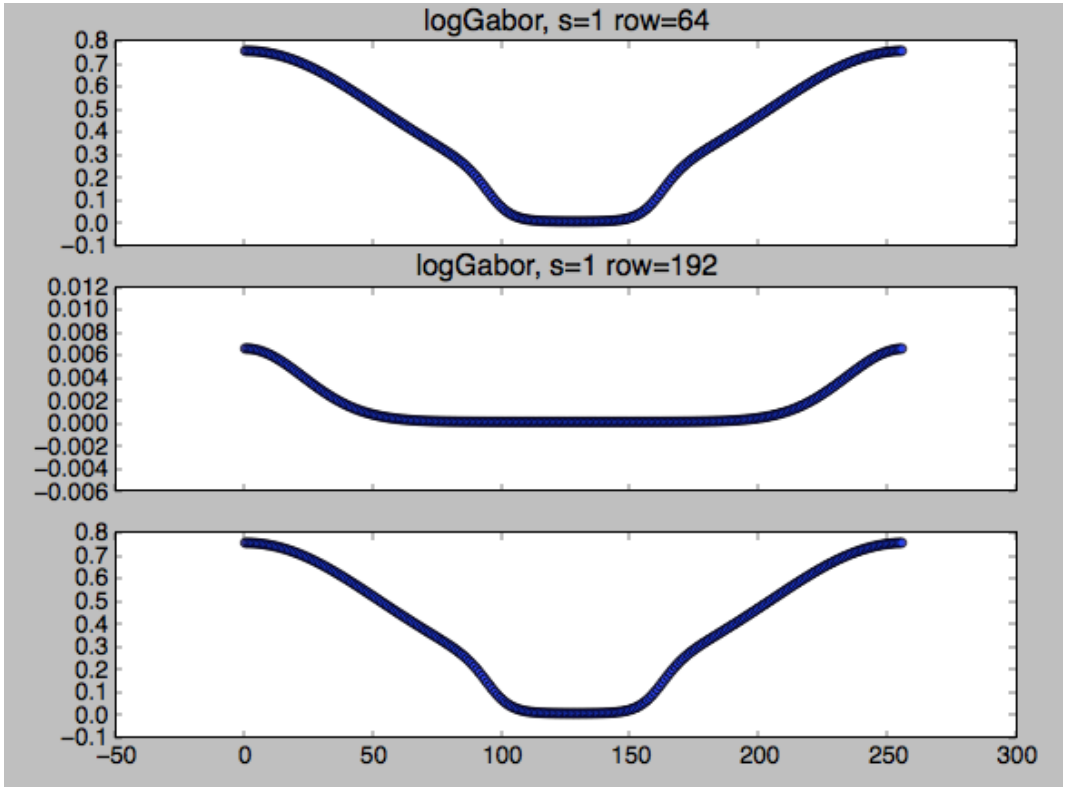


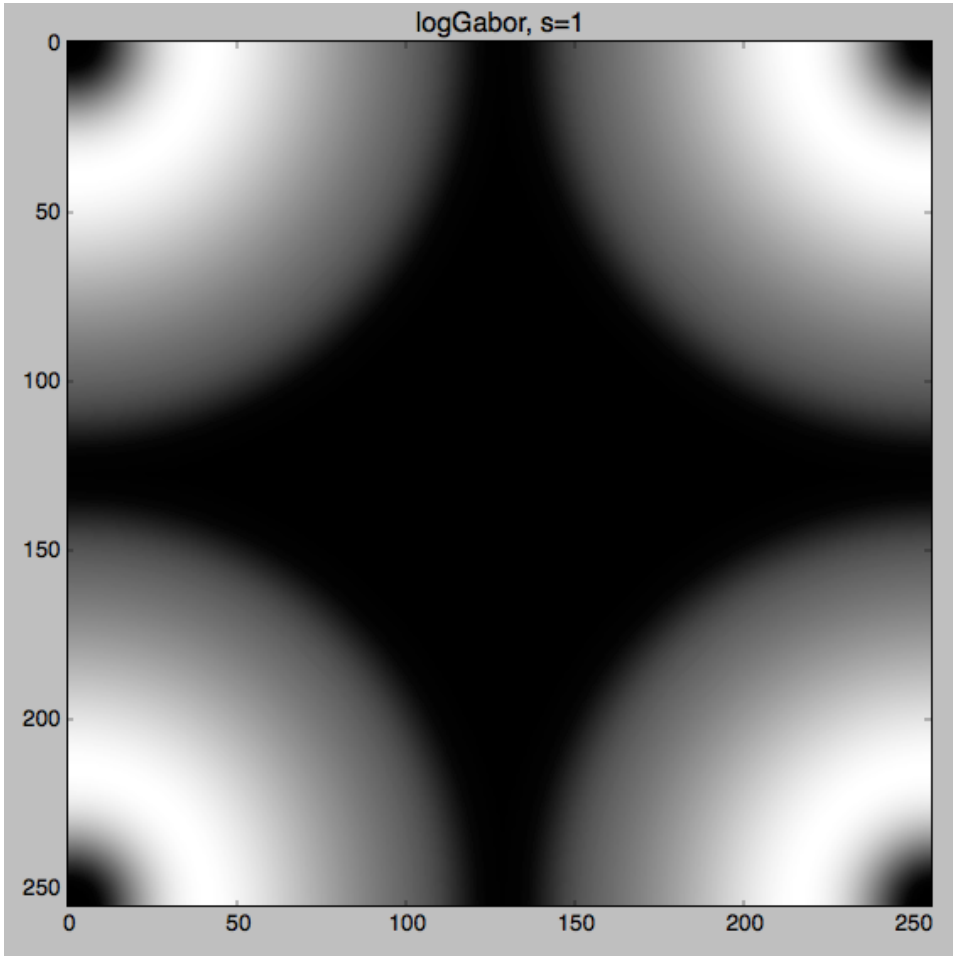
```
s=0, weight = 1. / (1. + exp(g * (cutOff - width)))
    (where cutOff=0.5, g=10.)
```



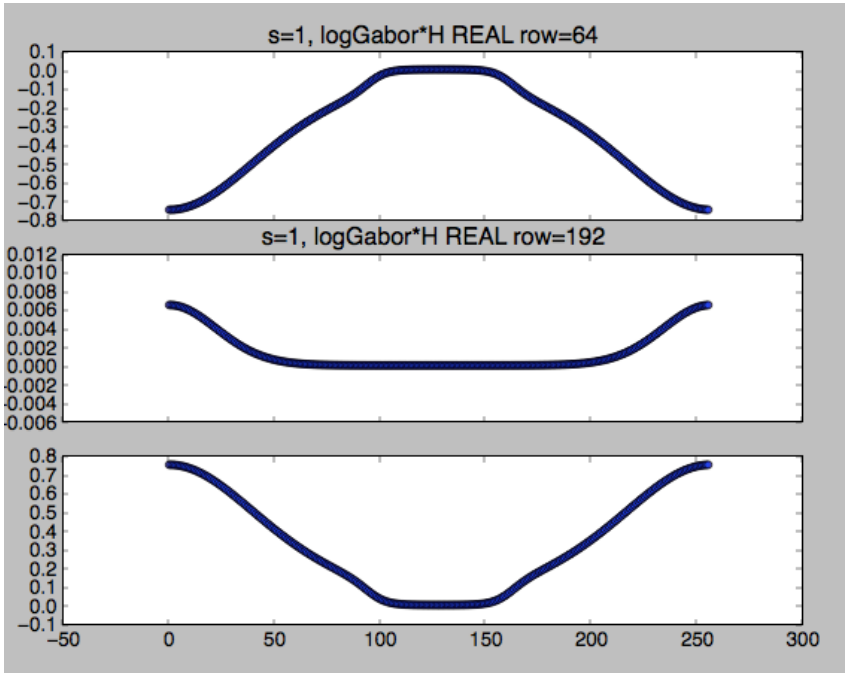


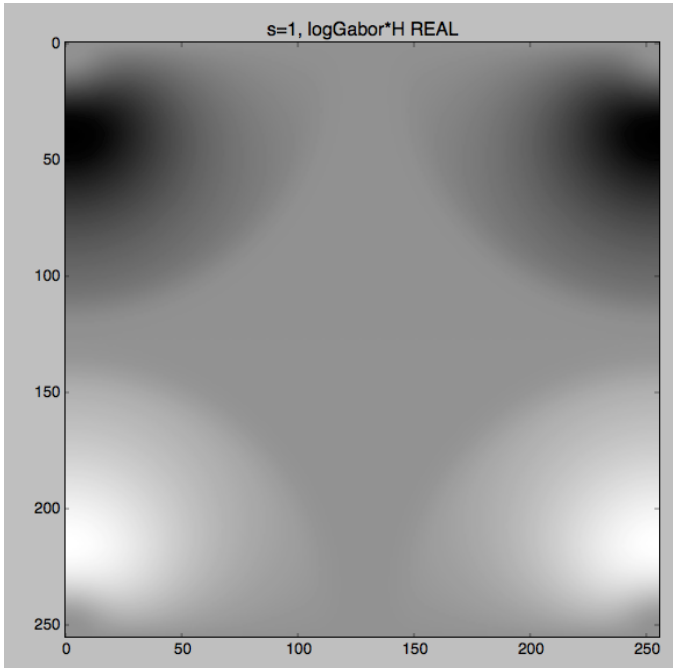
s=1, logGabor * low pass filter

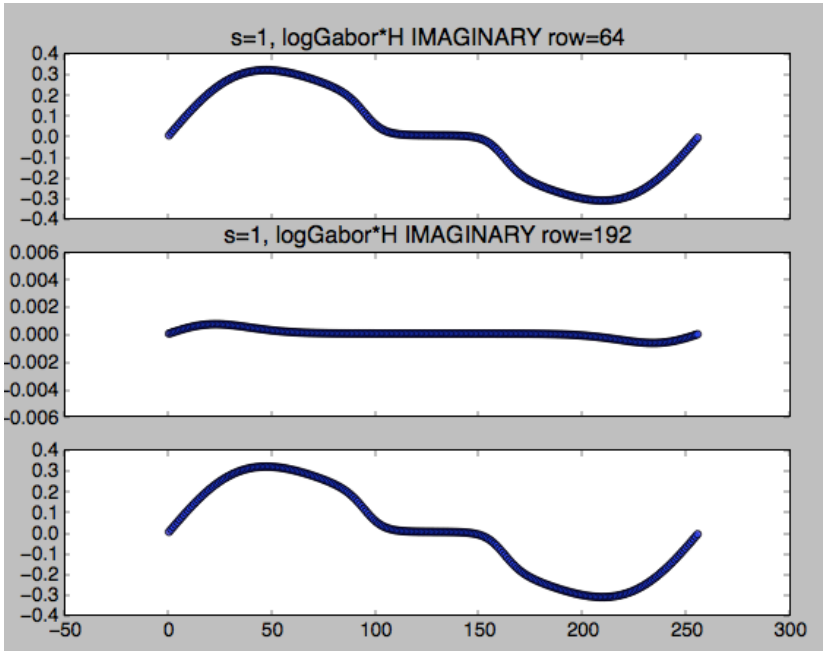


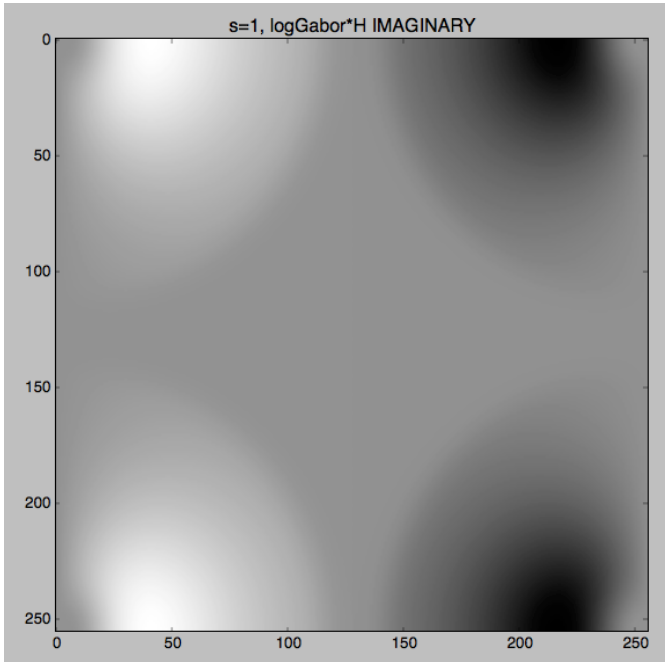


s=1, logGabor * low pass filter * H

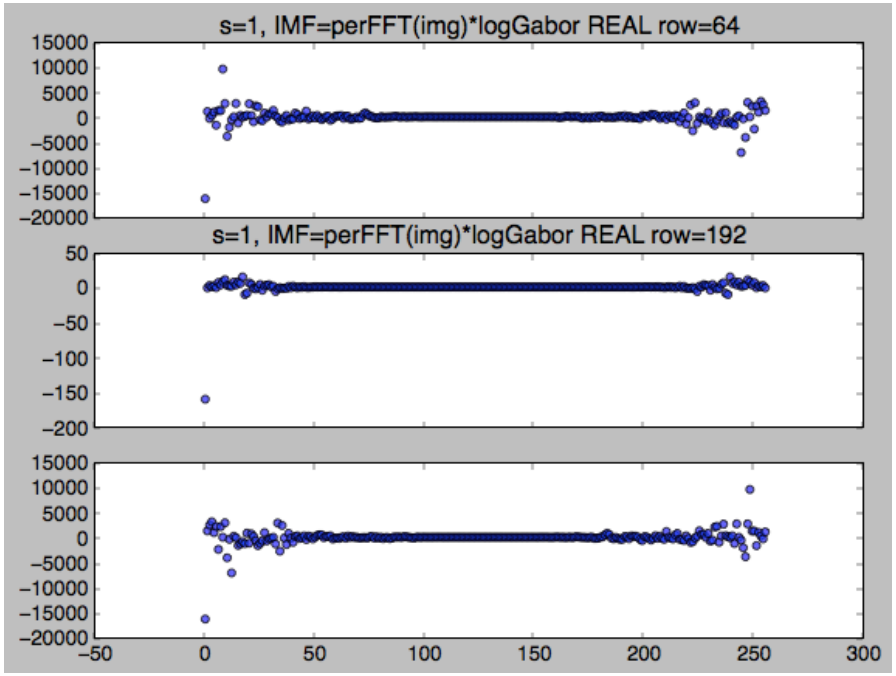


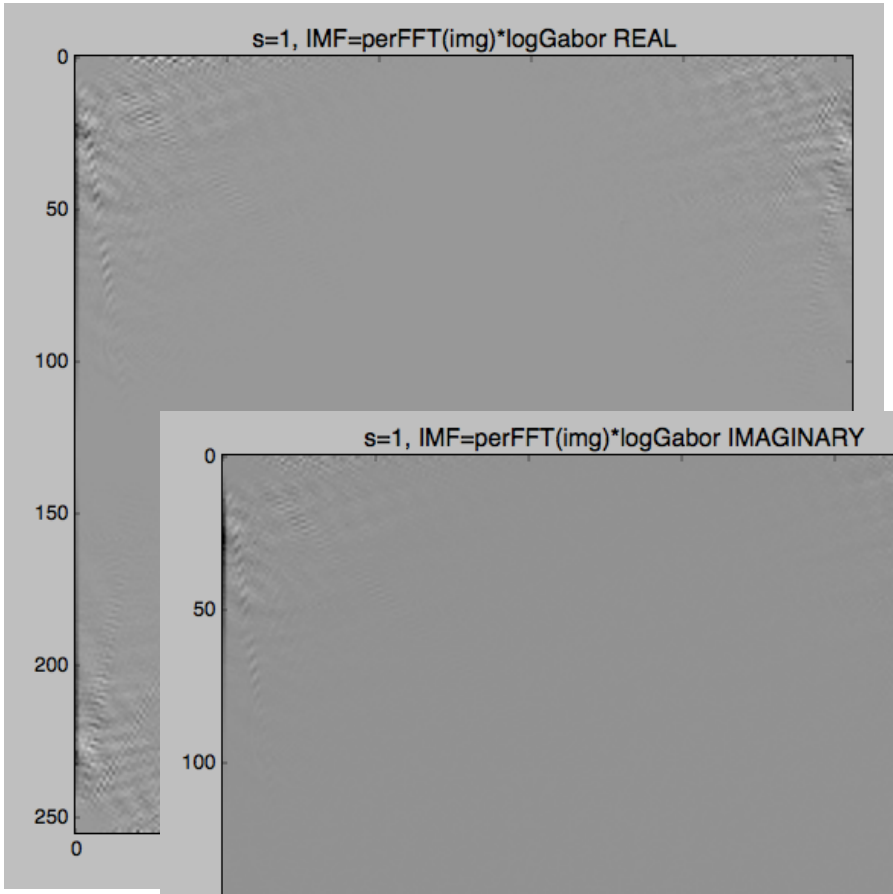


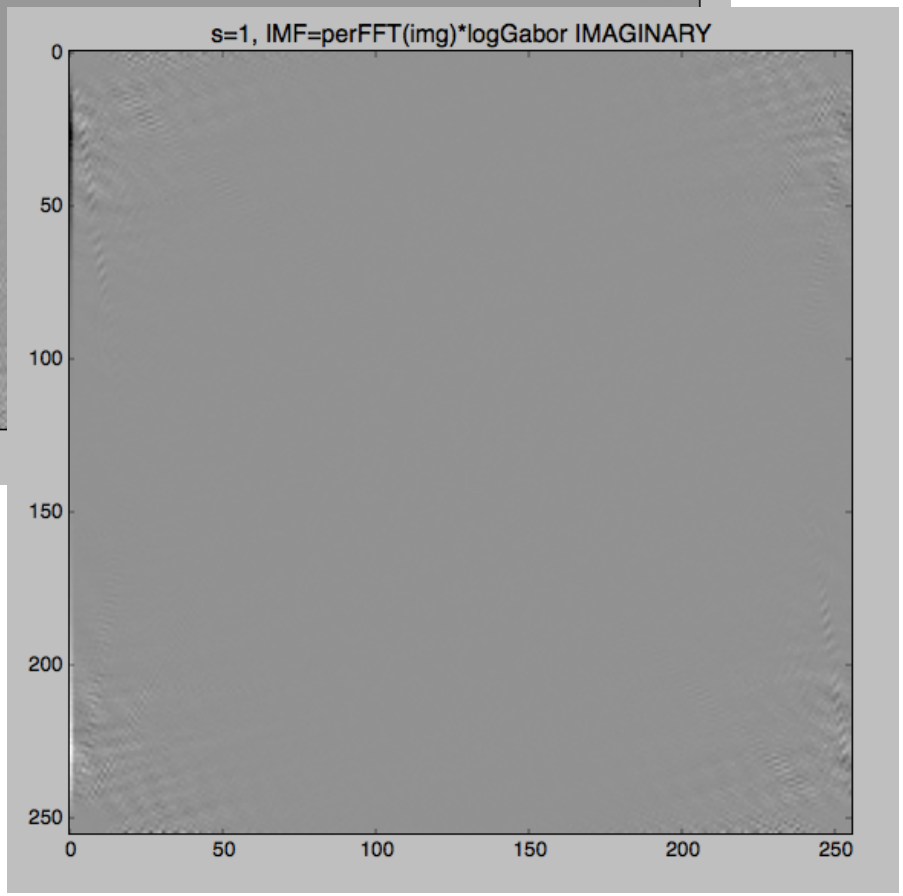


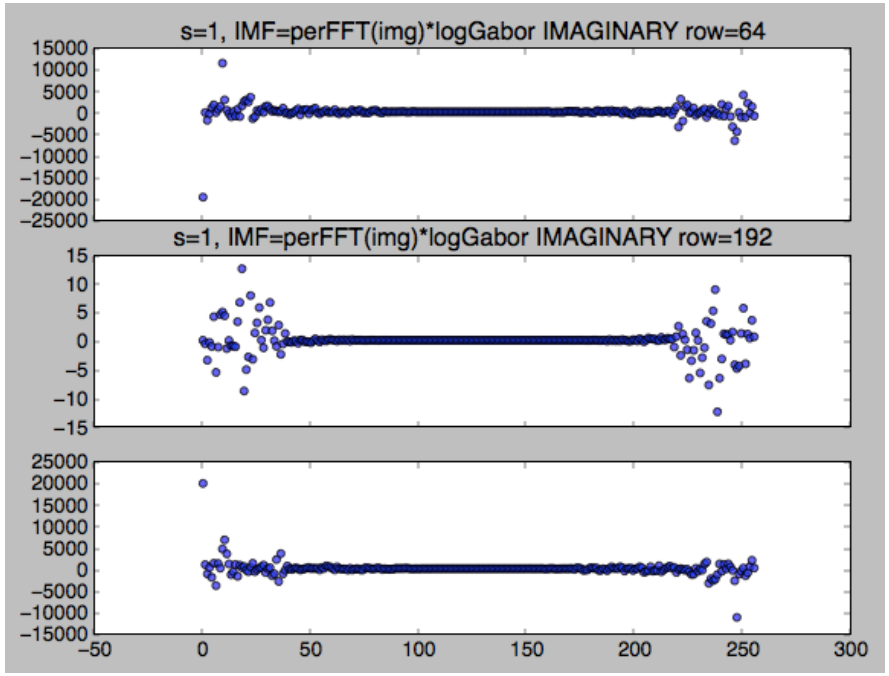


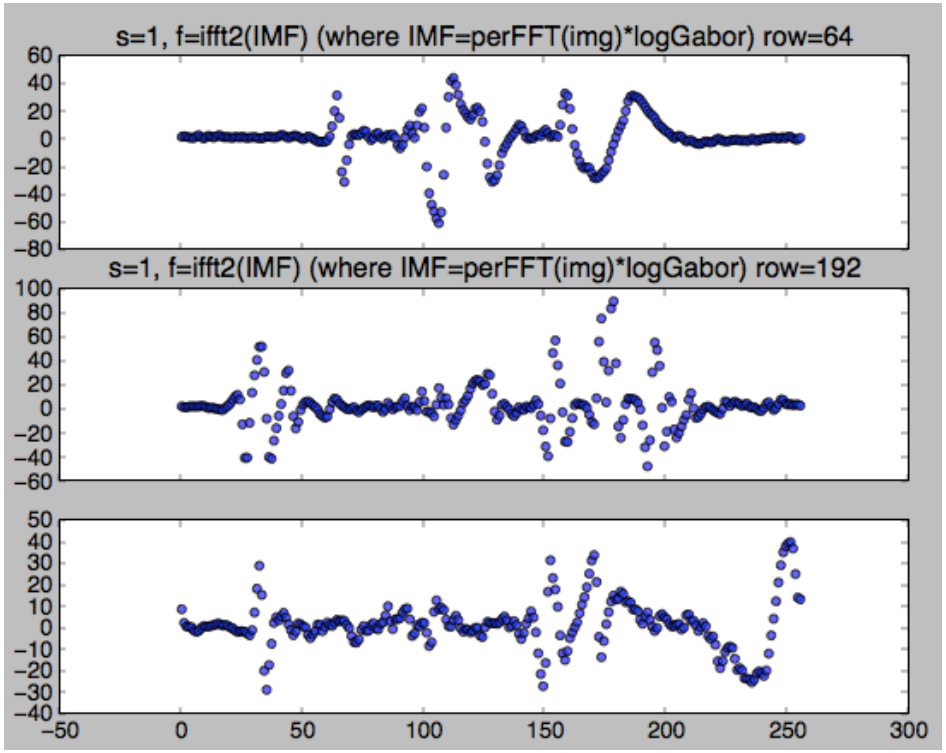
s=1, IMF=perFFT(img)*logGabor



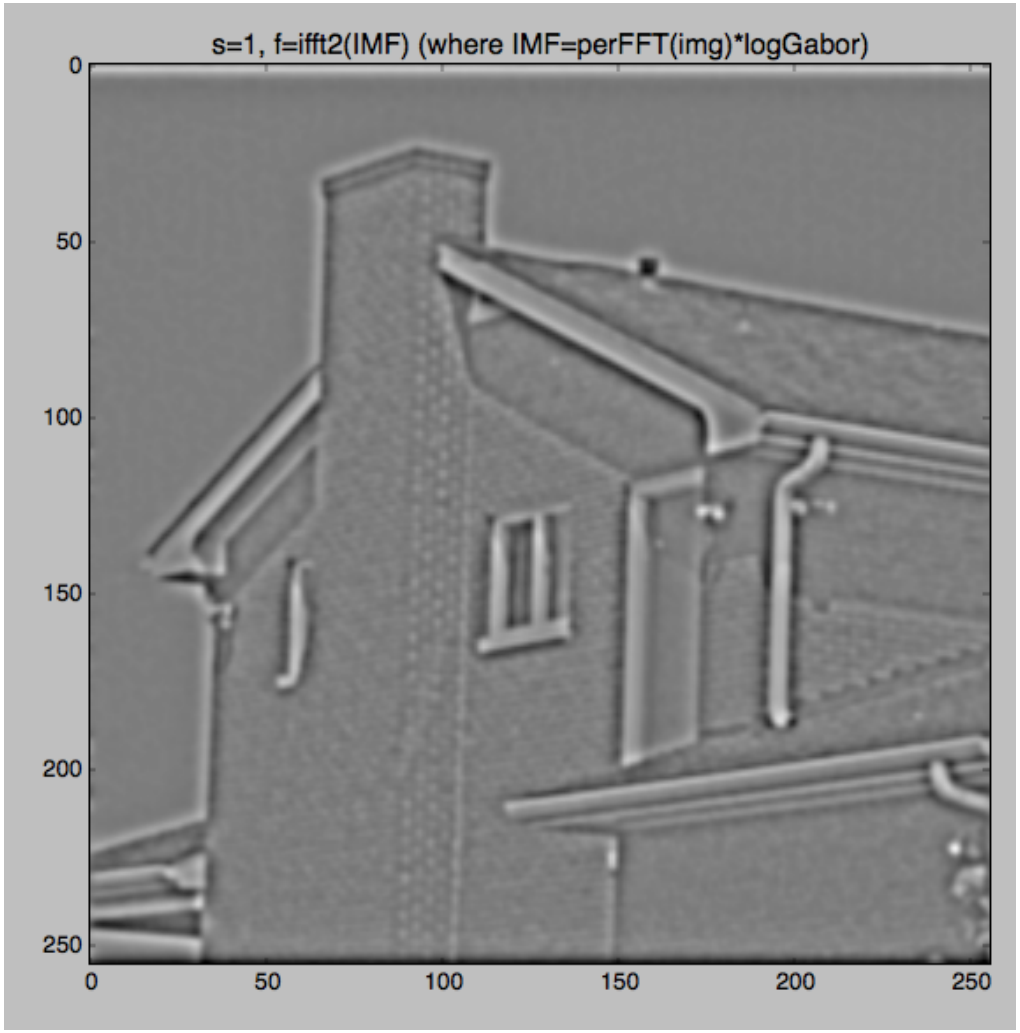


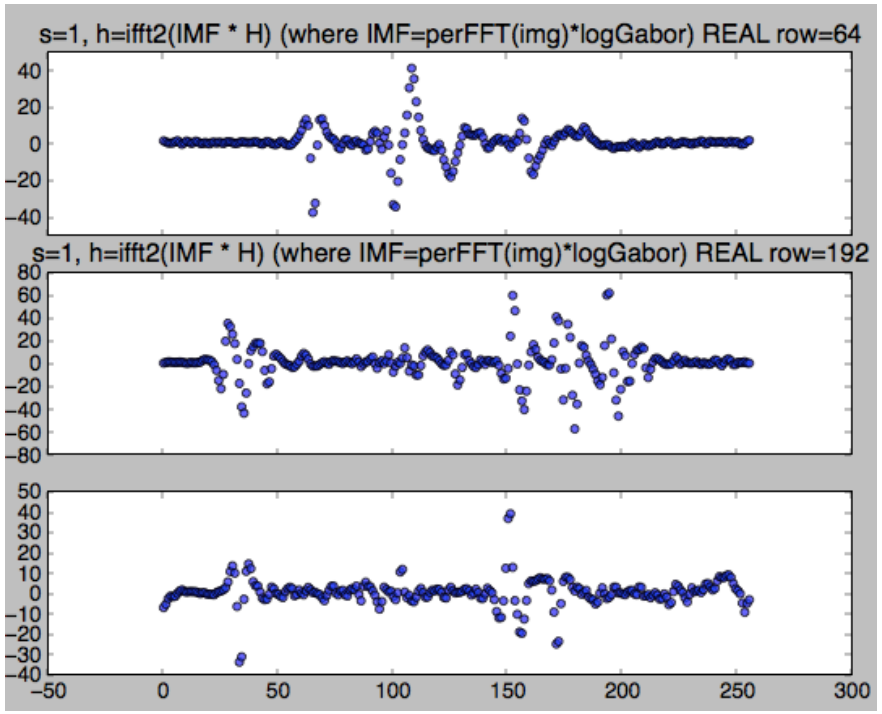


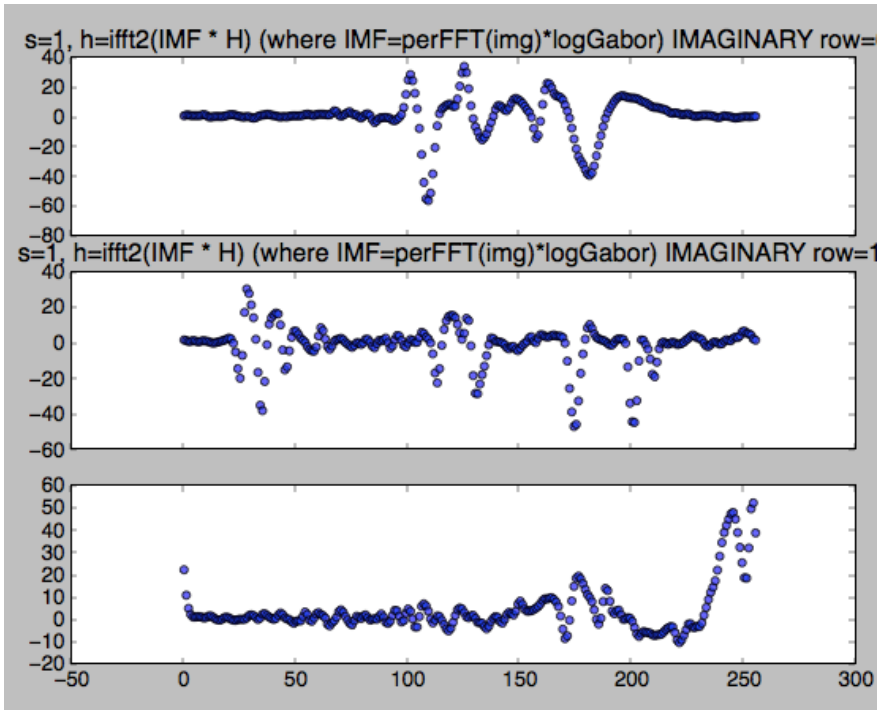




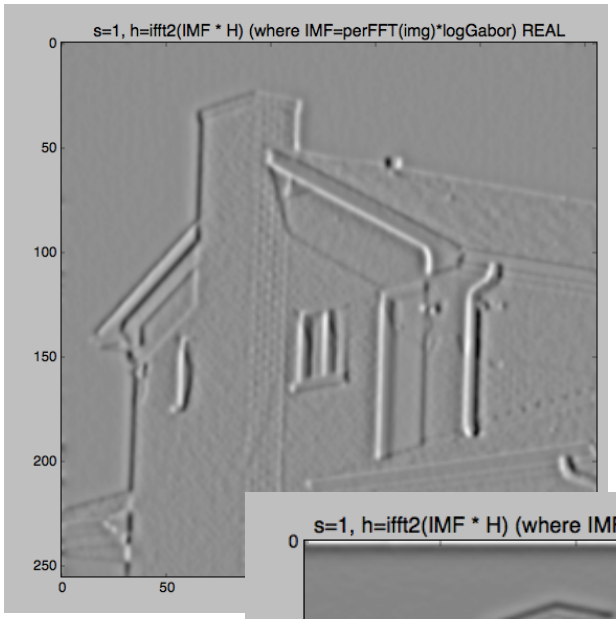
s=1, f=ifft2(IMF)
(where IMF=perFFT(img)*logGabor)

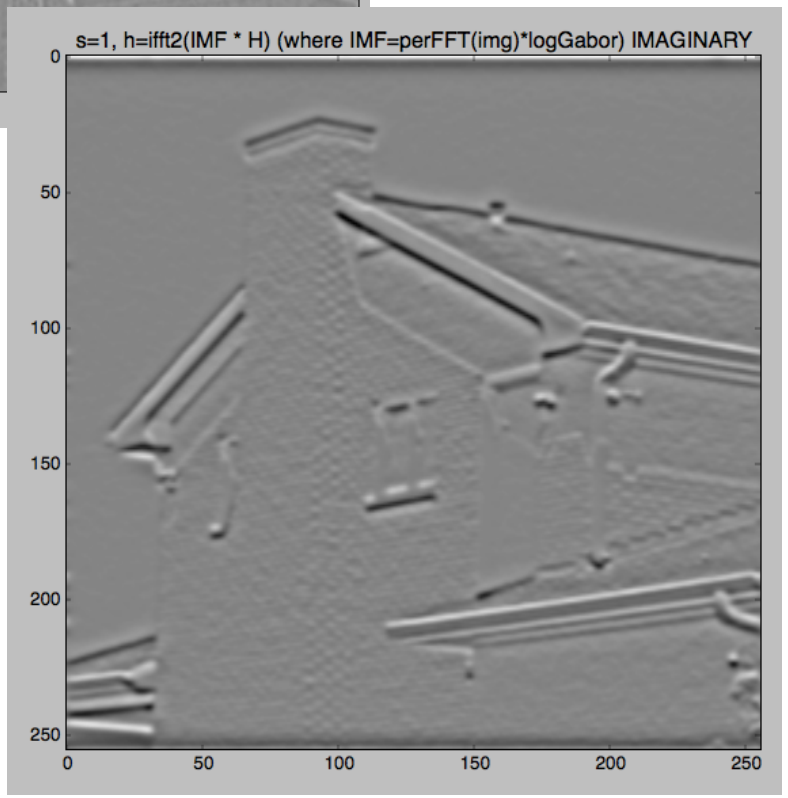




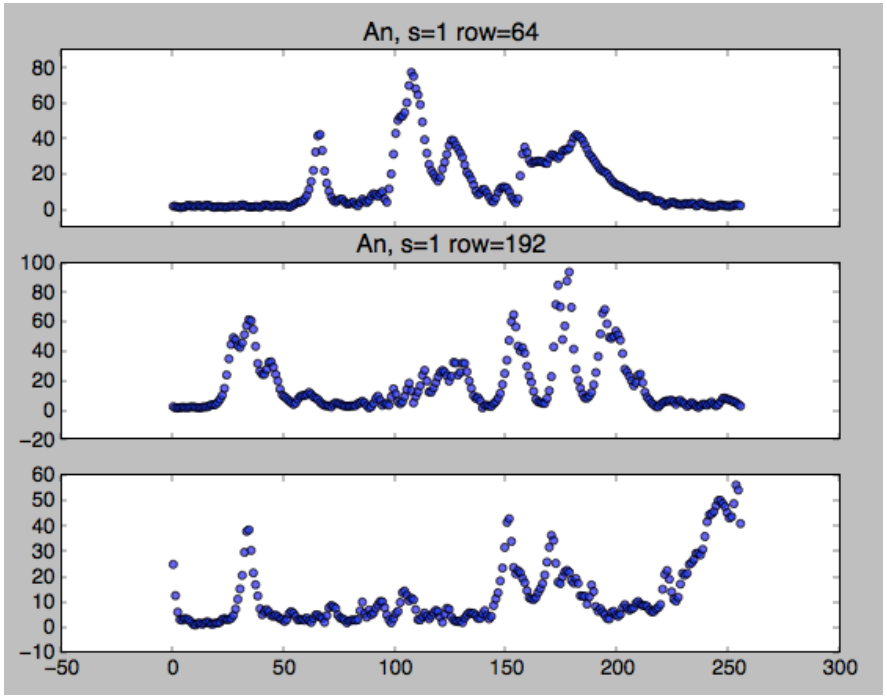


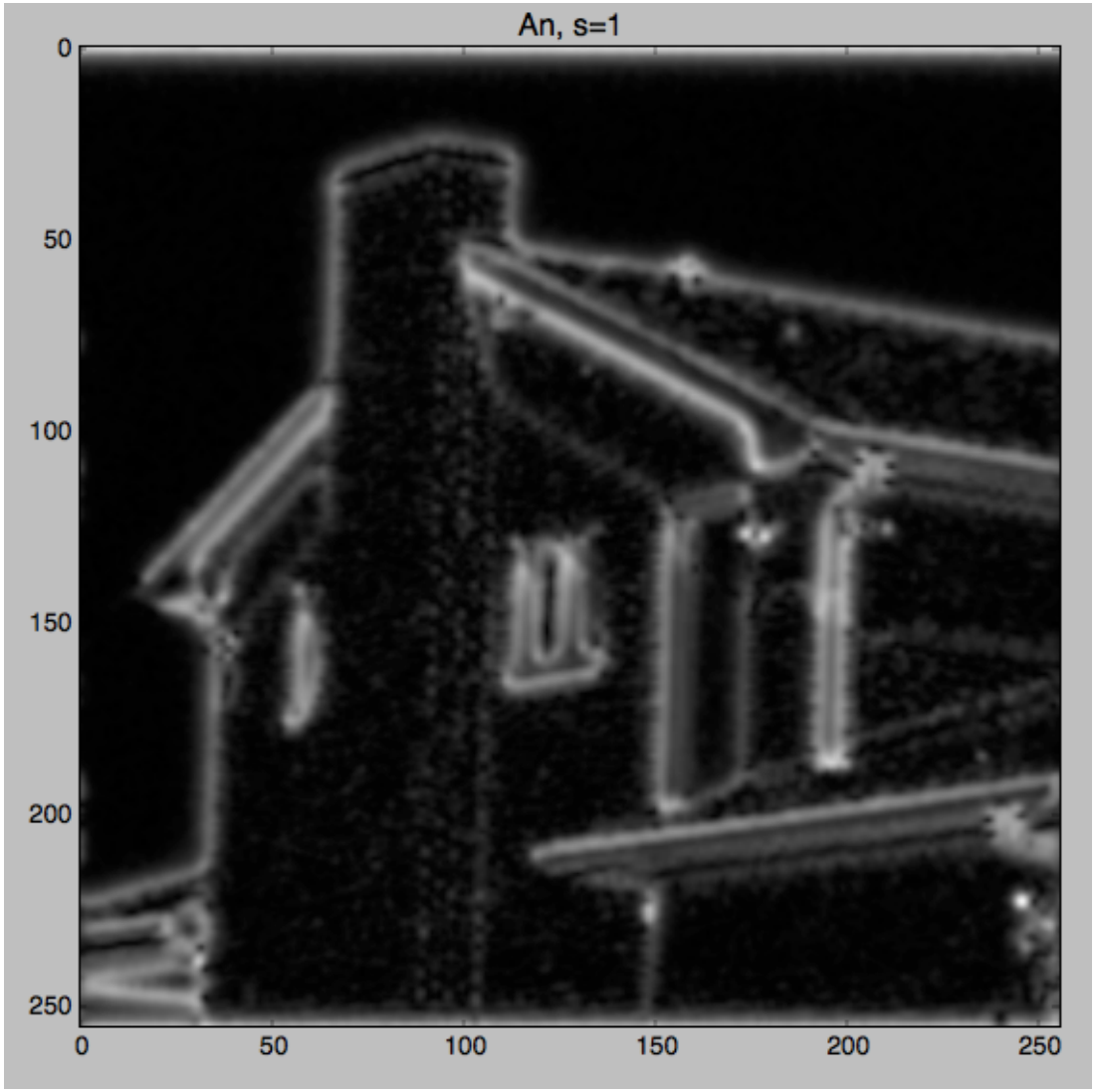

```
s=1, h=ifft2(IMF * H)
(where IMF=perFFT(img)*logGabor)
```



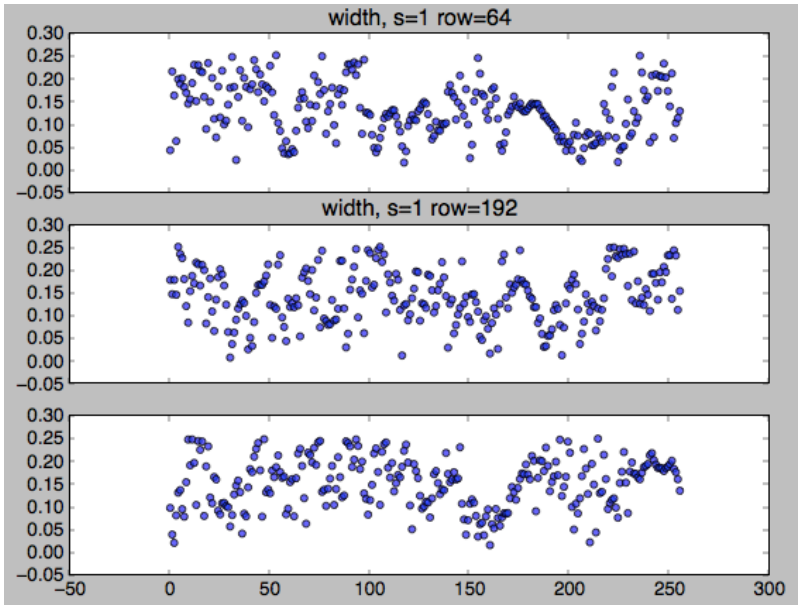


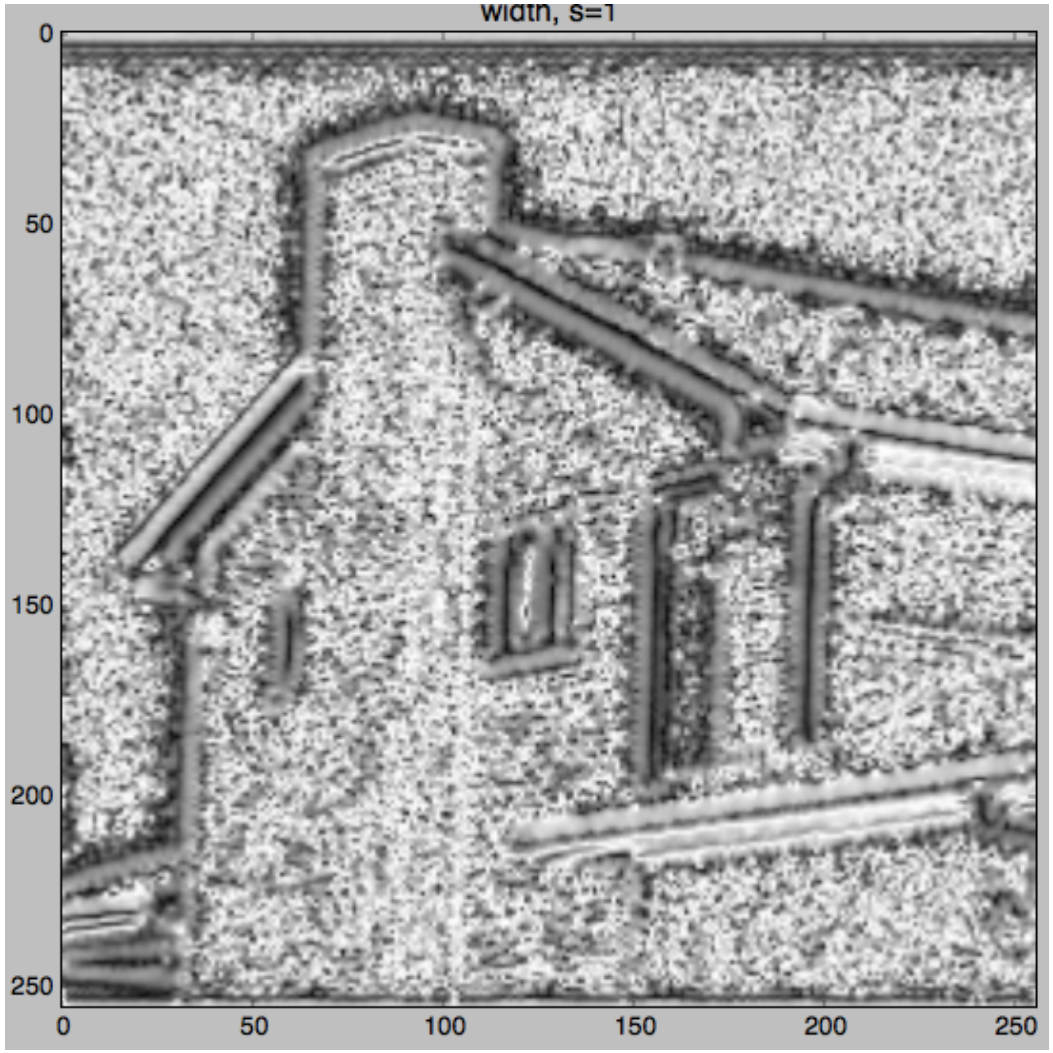
```
s=1, An = sqrt(f * f + h1 * h1 + h2 * h2)
(where f=ifft2(perFFT(img)*logGabor)
   and h=ifft2(perFFT(img)*logGabor * H) )
```



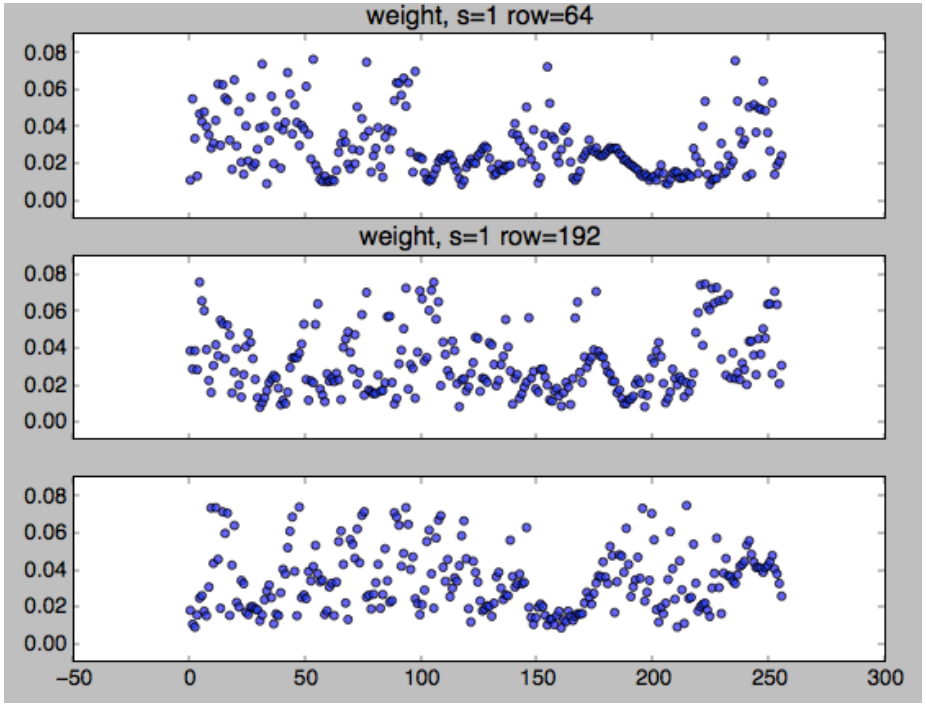


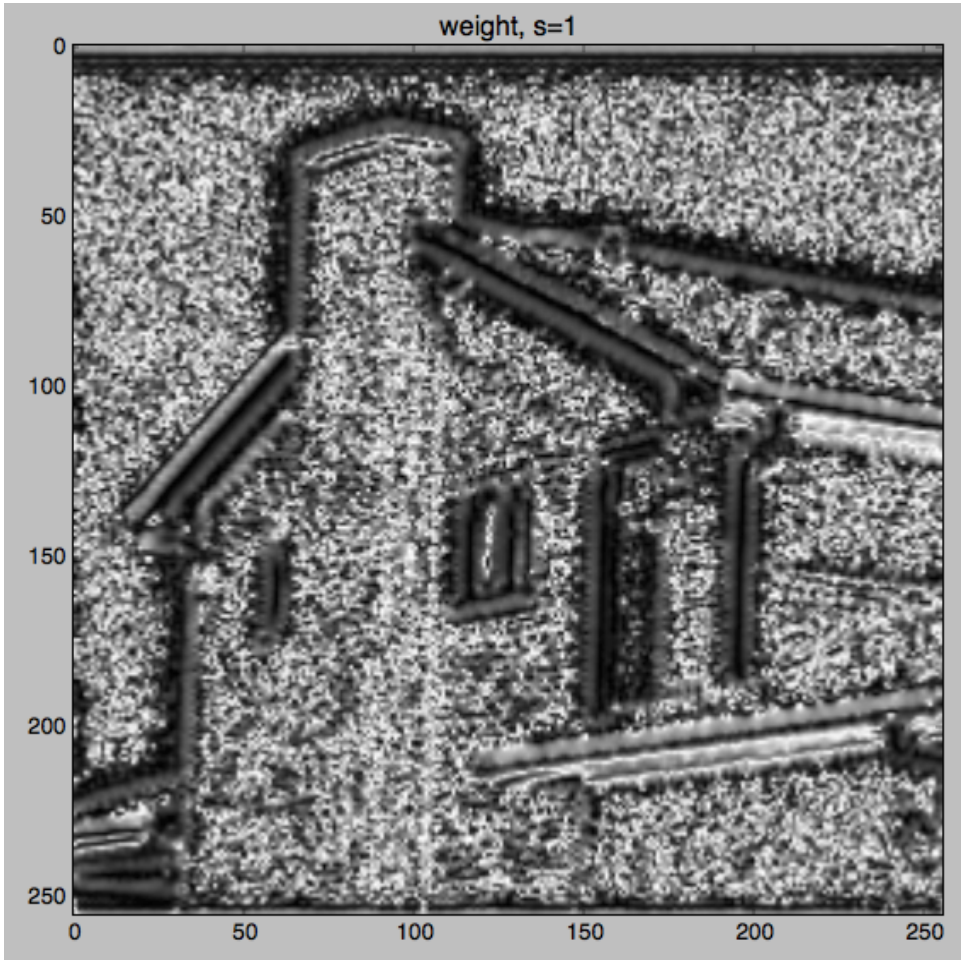
s=1, **width** = (sumAn / (maxAn + epsilon) – 1.) / (nscale – 1)



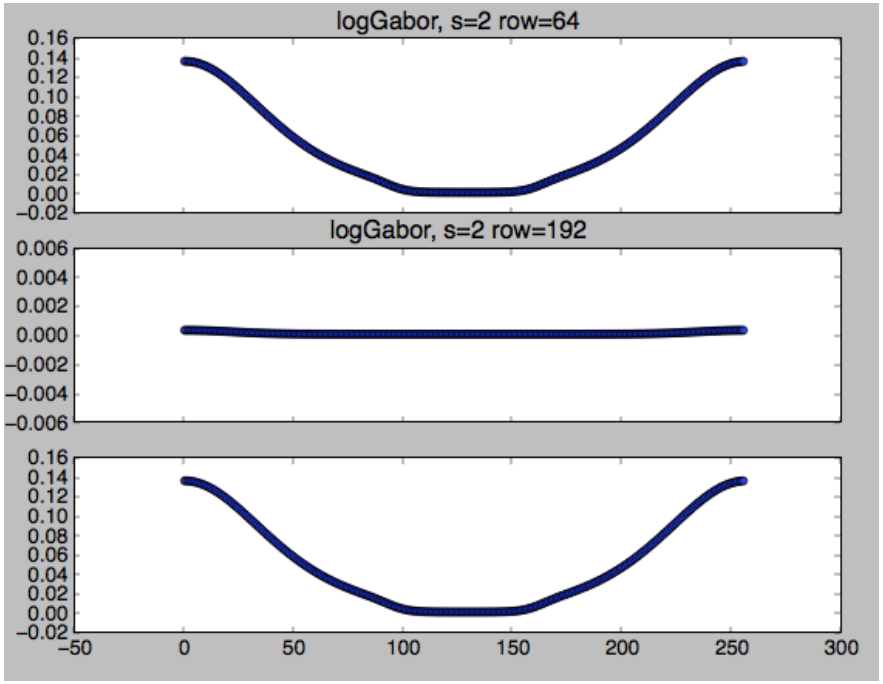


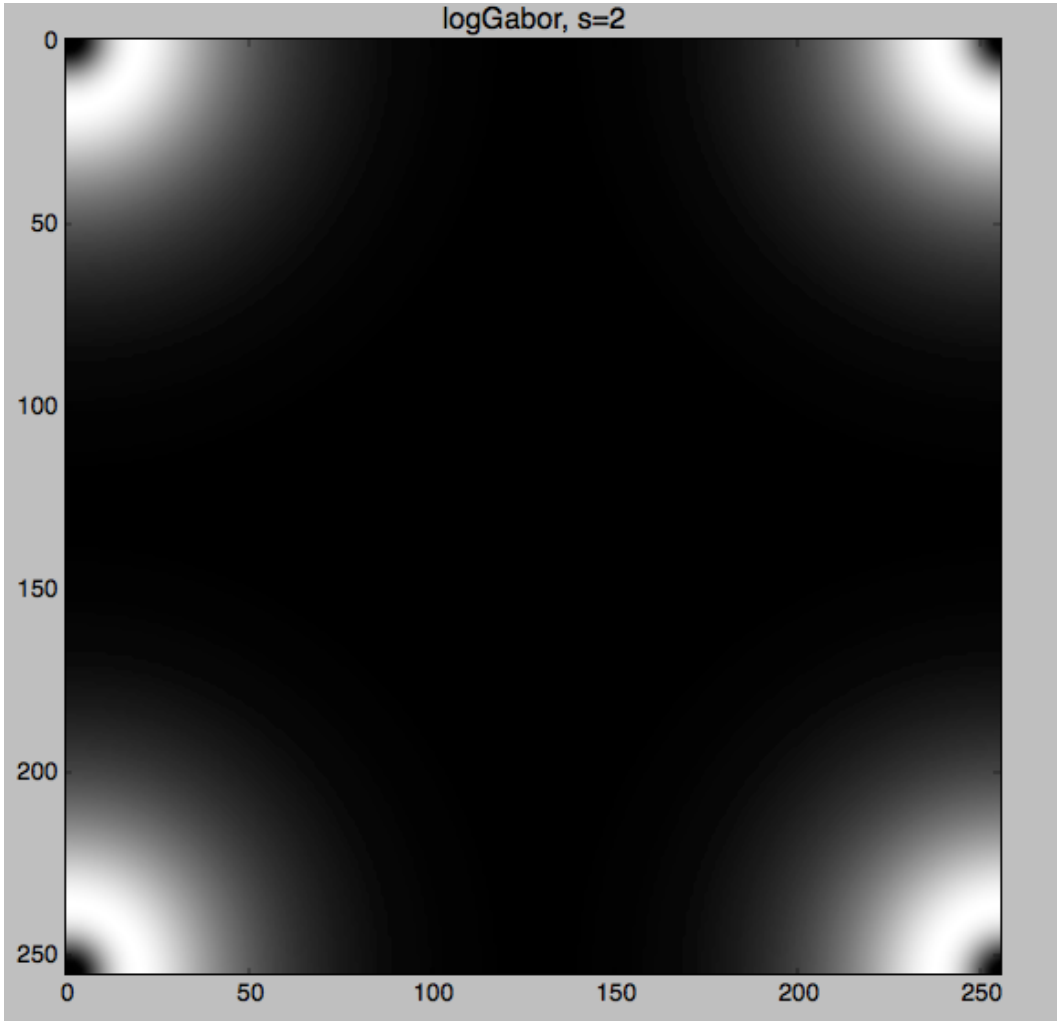
```
s=1, weight = 1. / (1. + exp(g * (cutOff - width)))
    (where cutOff=0.5, g=10.)
```



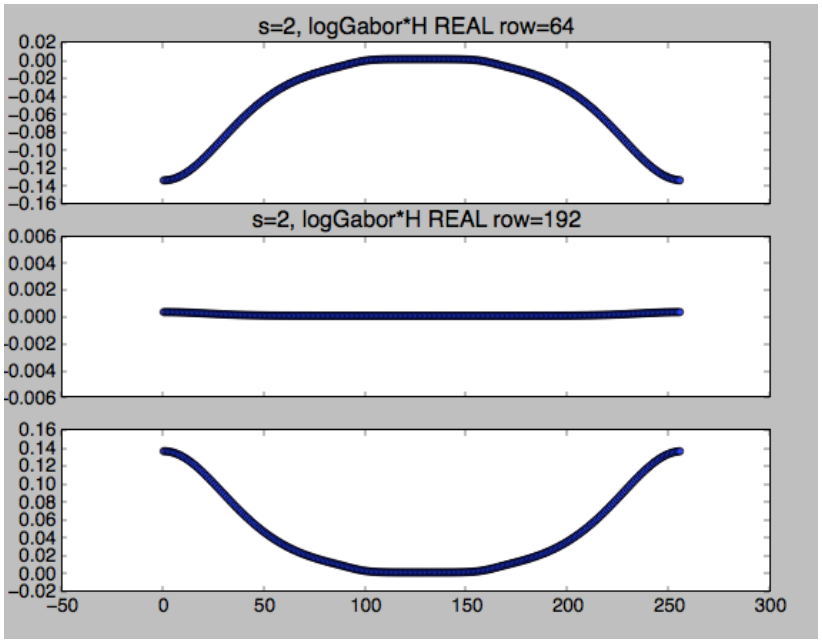


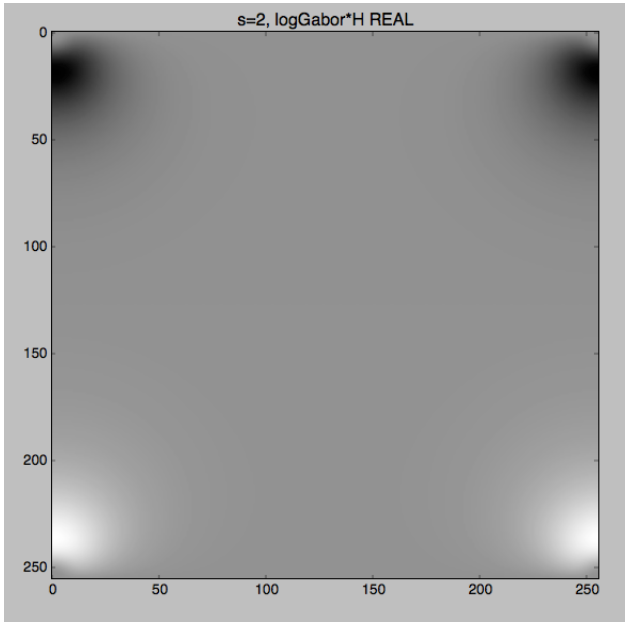
s=2, logGabor * low pass filter

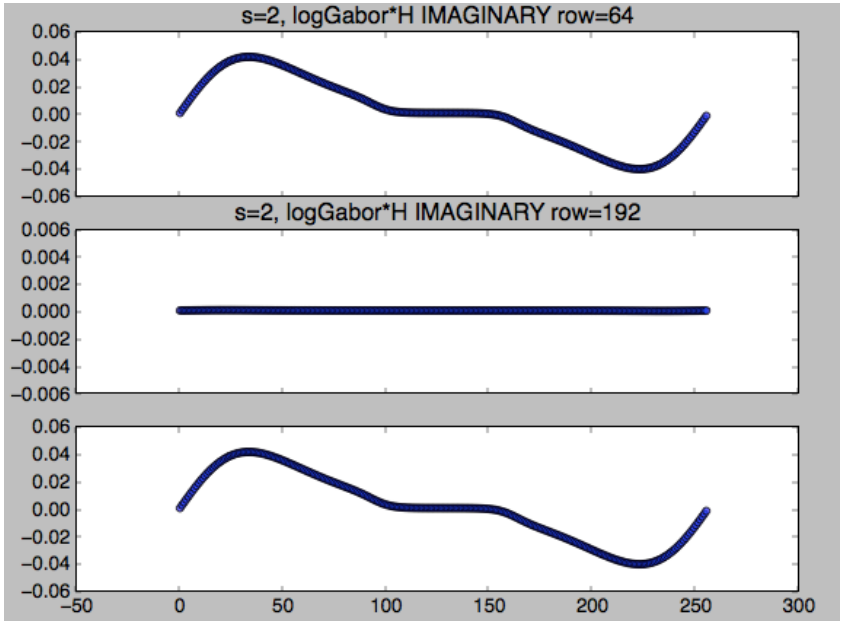


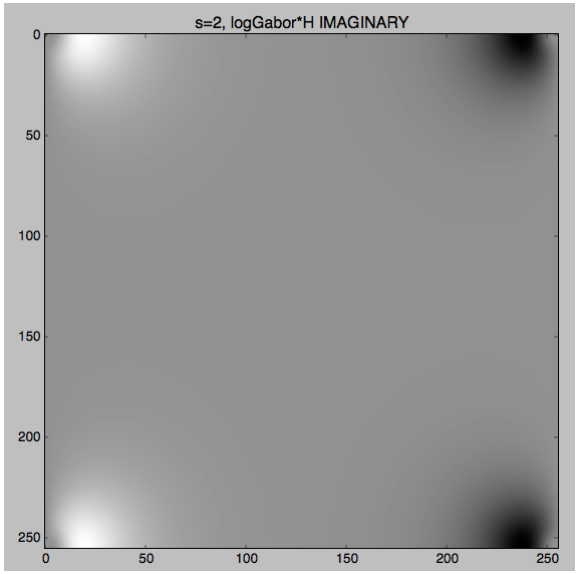


s=2, logGabor * low pass filter * H

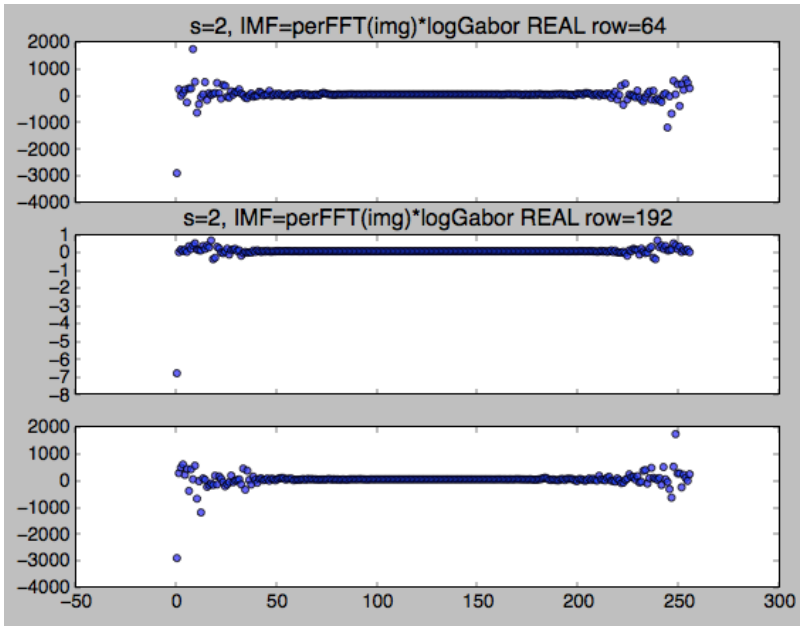


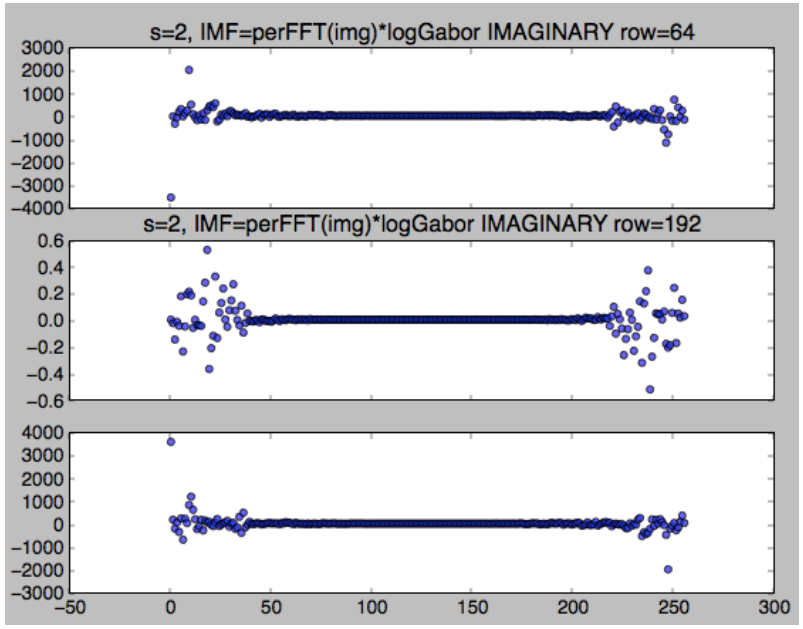


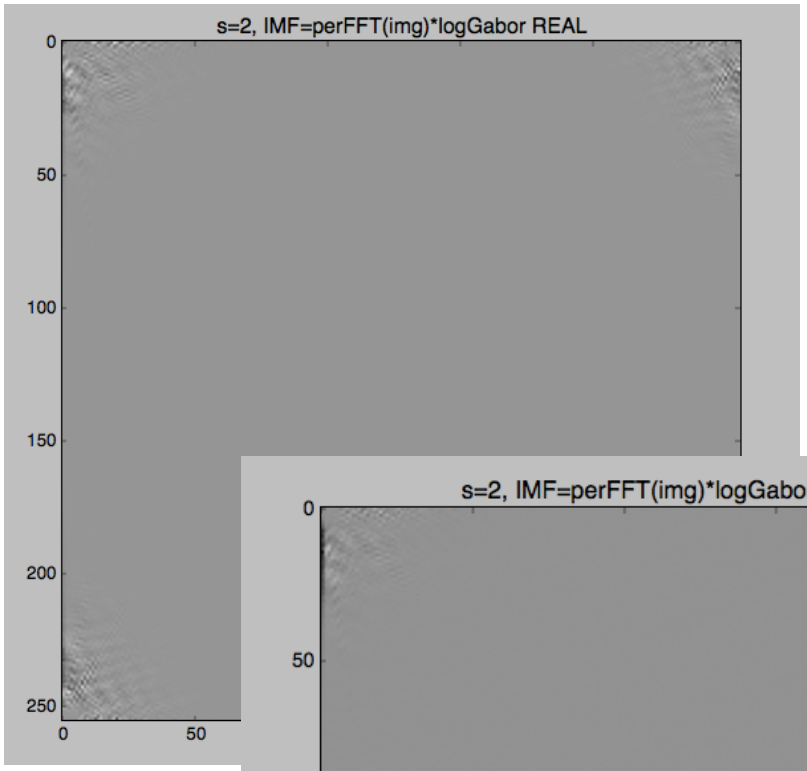


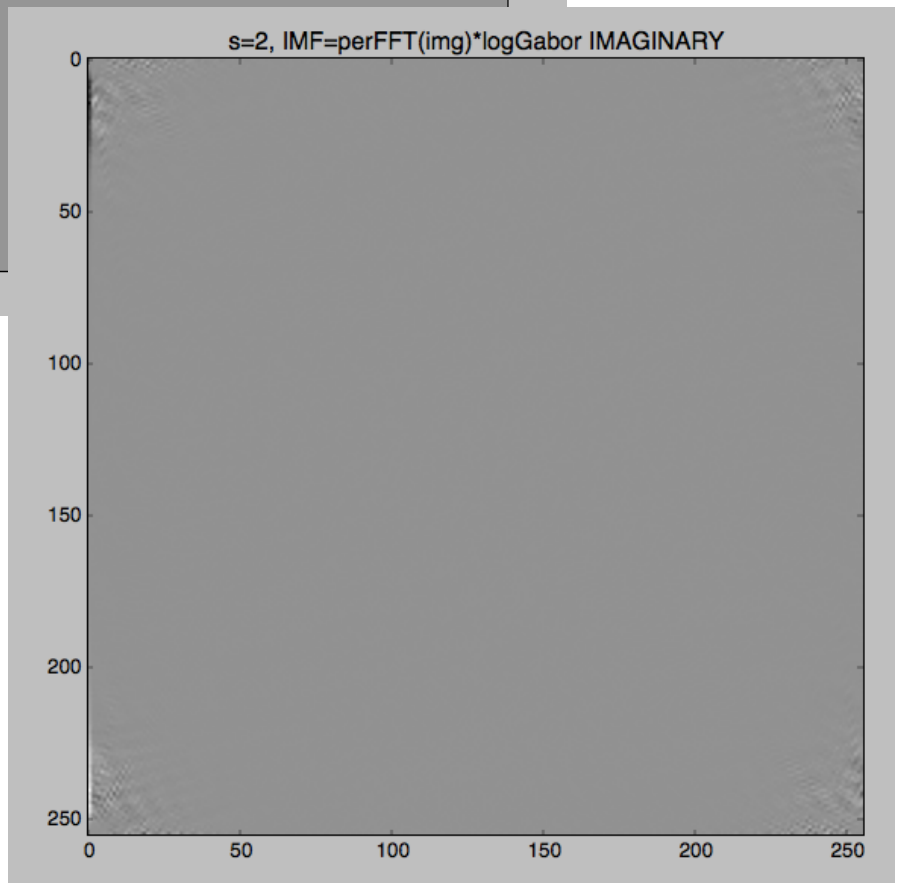


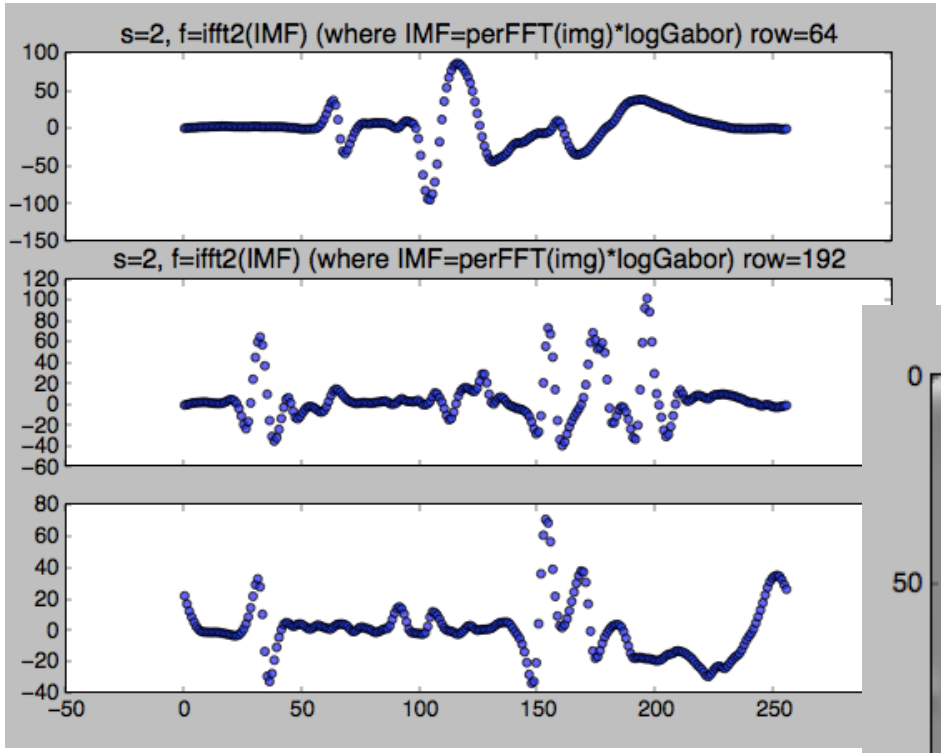
s=2, IMF=perFFT(img)*logGabor



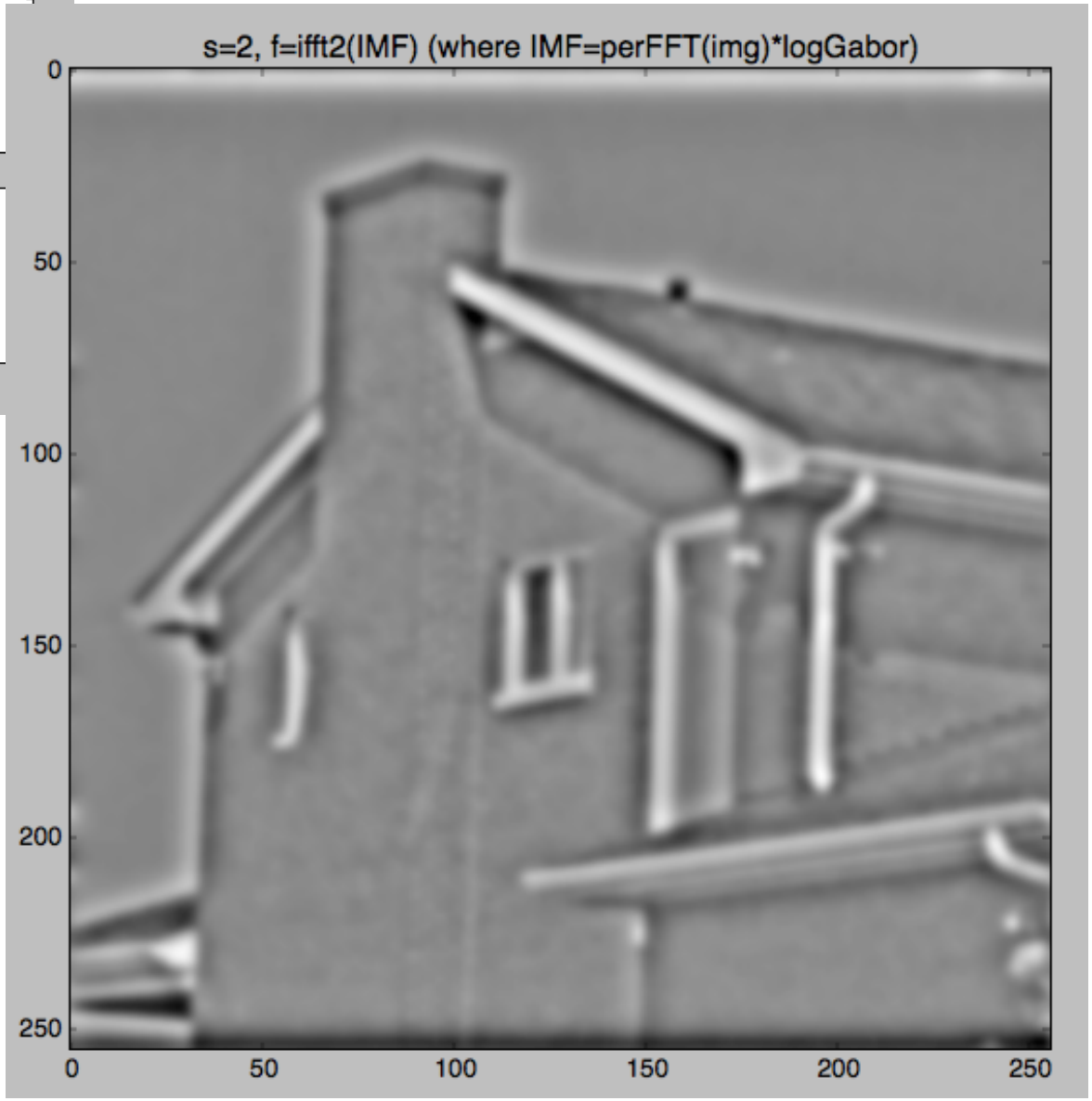




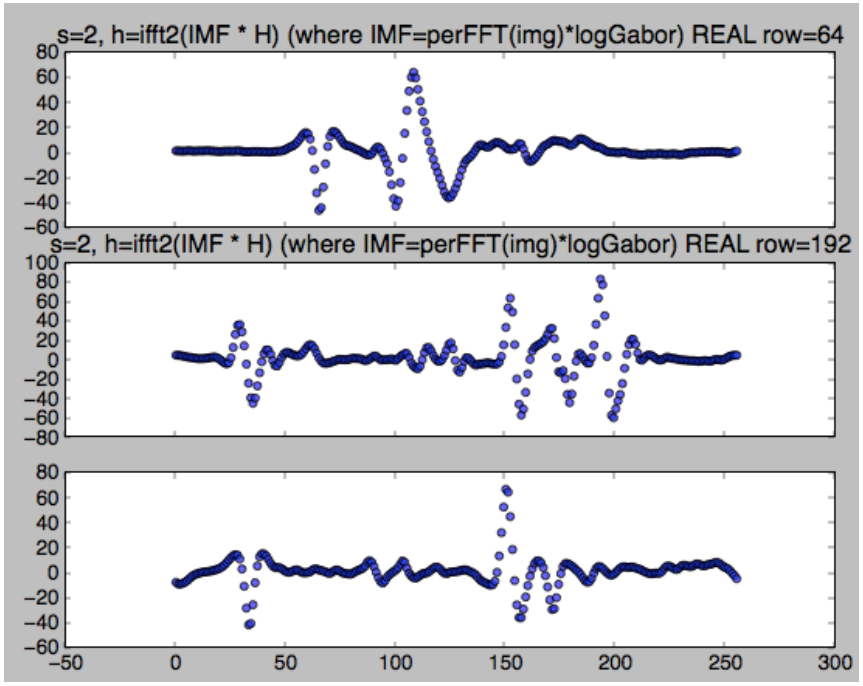


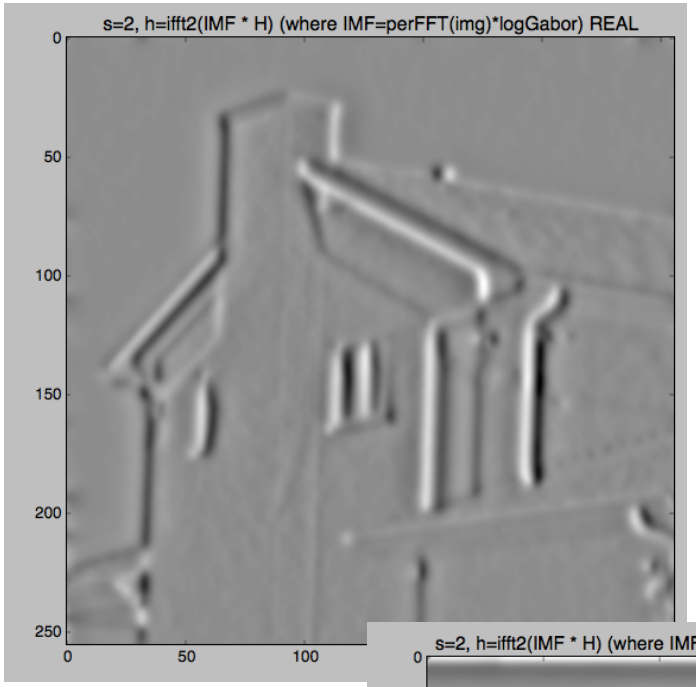


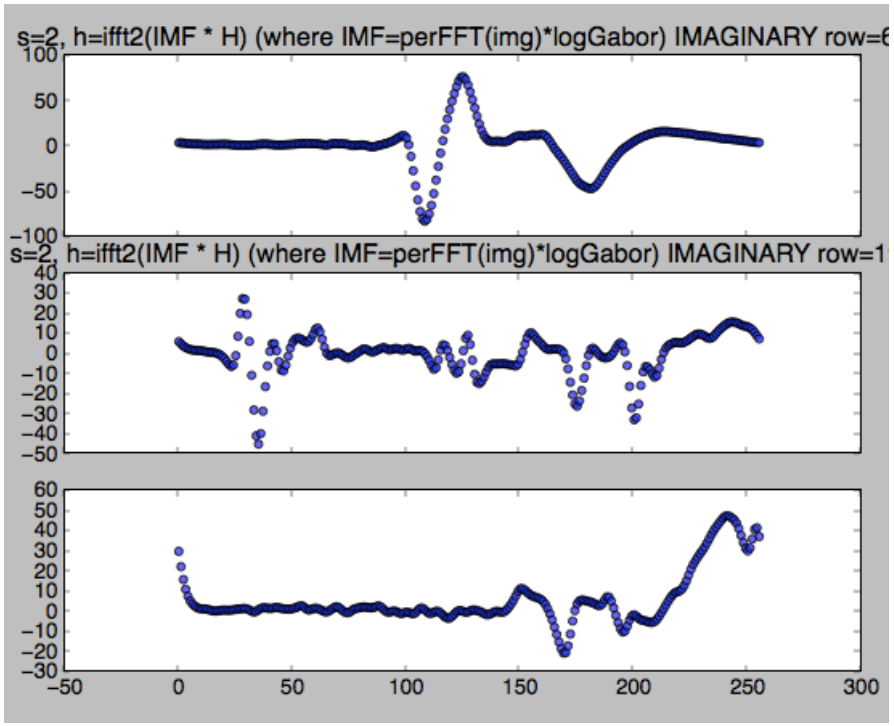
s=2, f=ifft2(IMF)
(where IMF=perFFT(img)*logGabor)

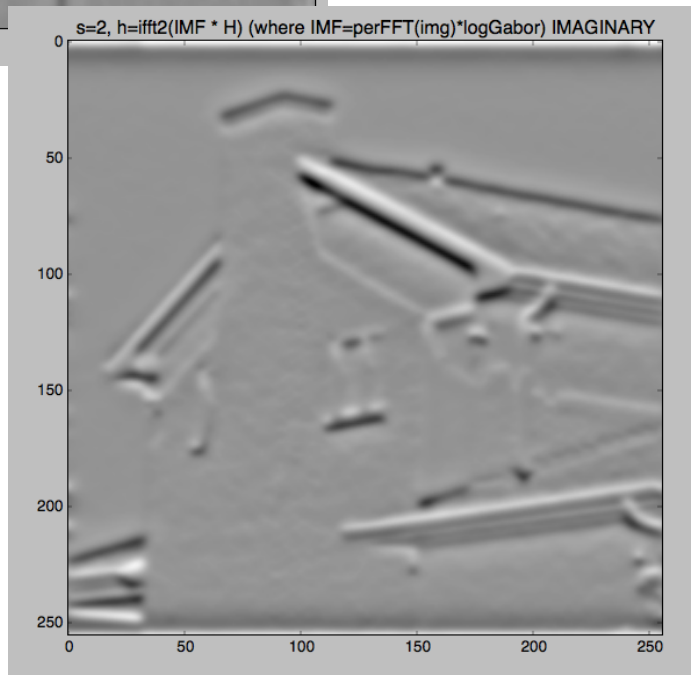


s=2, h=ifft2(IMF * H)
(where IMF=perFFT(img)*logGabor)

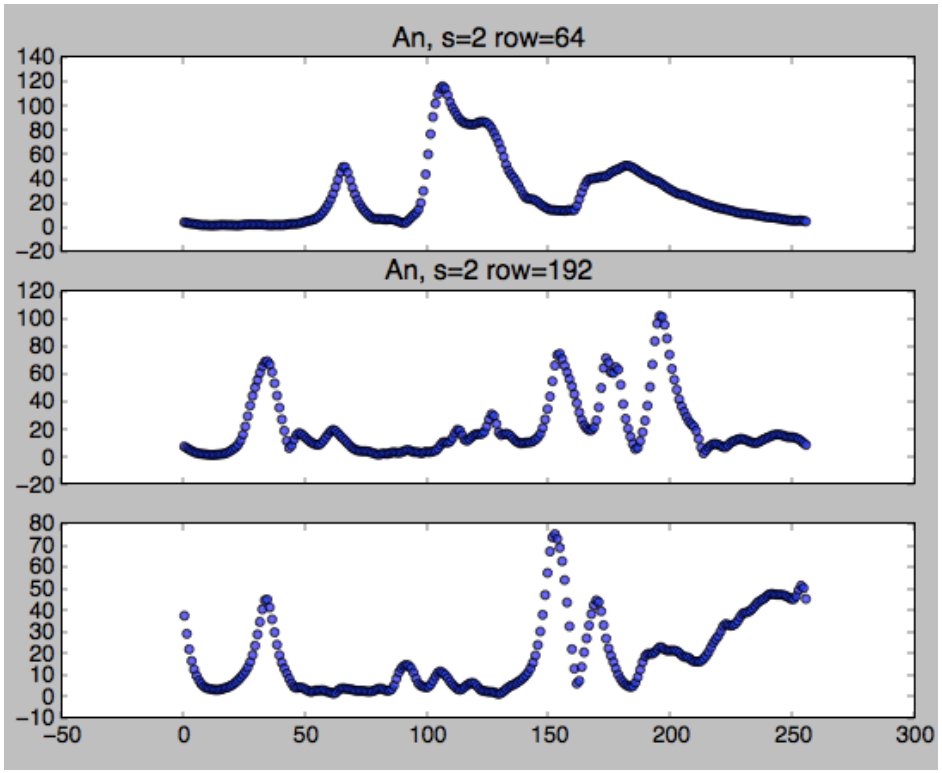


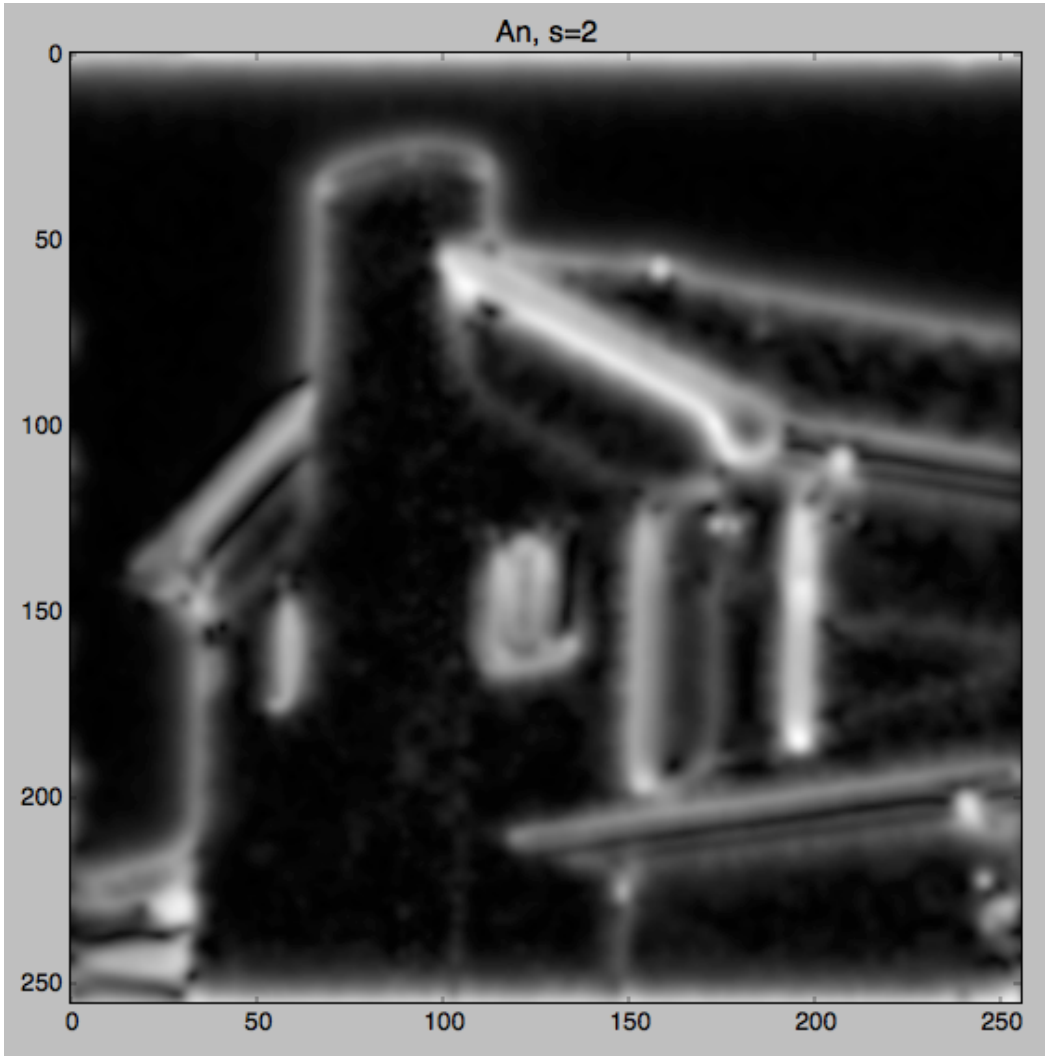




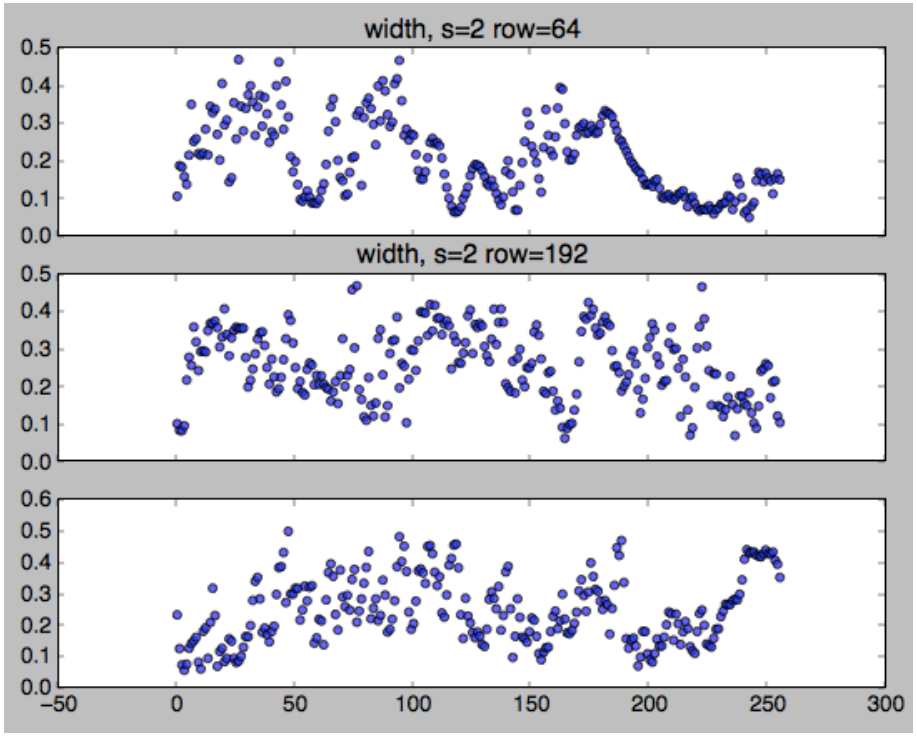


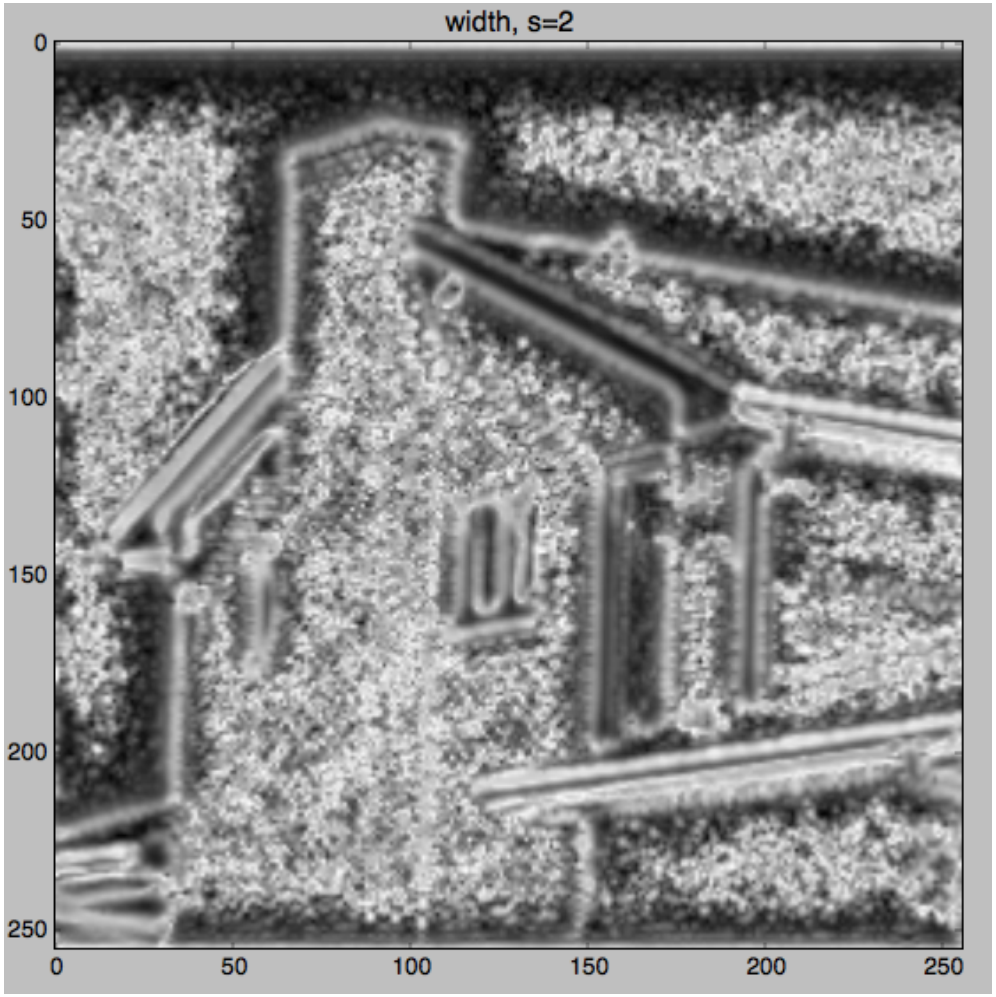
```
s=2, An = sqrt(f * f + h1 * h1 + h2 * h2)
(where f=ifft2(perFFT(img)*logGabor)
   and h=ifft2(perFFT(img)*logGabor * H) )
```



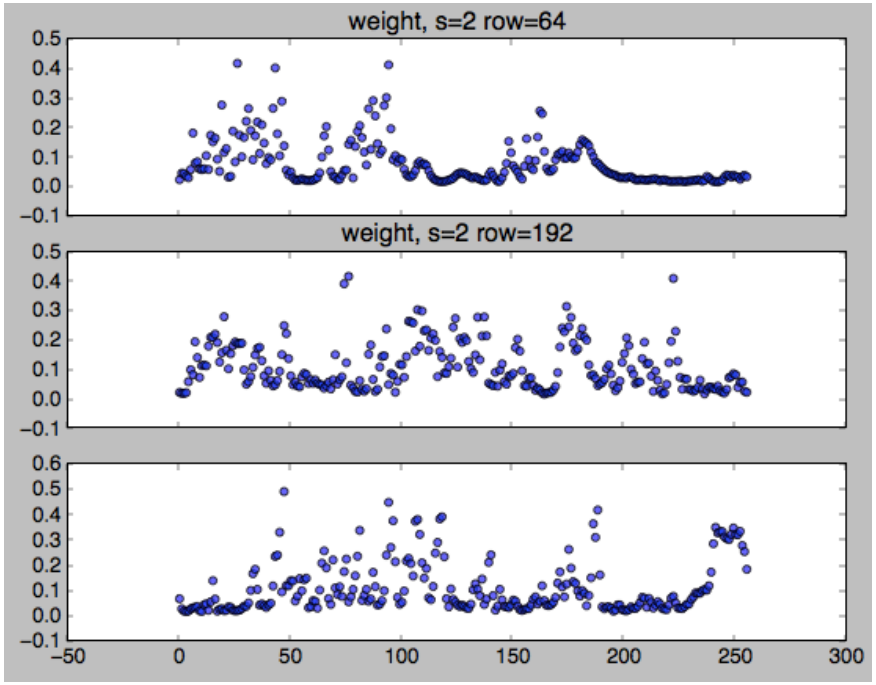


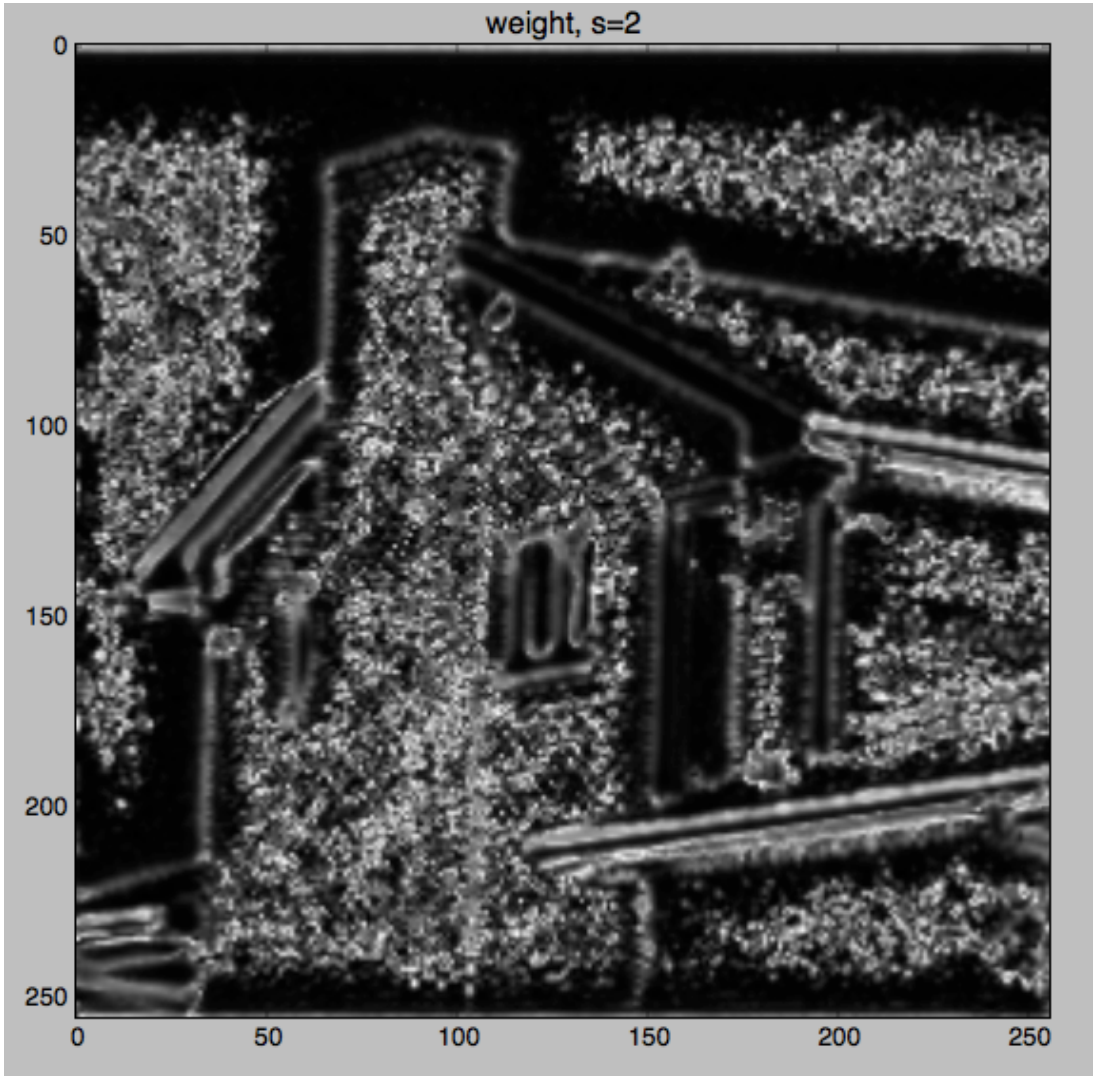
s=2, **width** = (sumAn / (maxAn + epsilon) – 1.) / (nscale – 1)



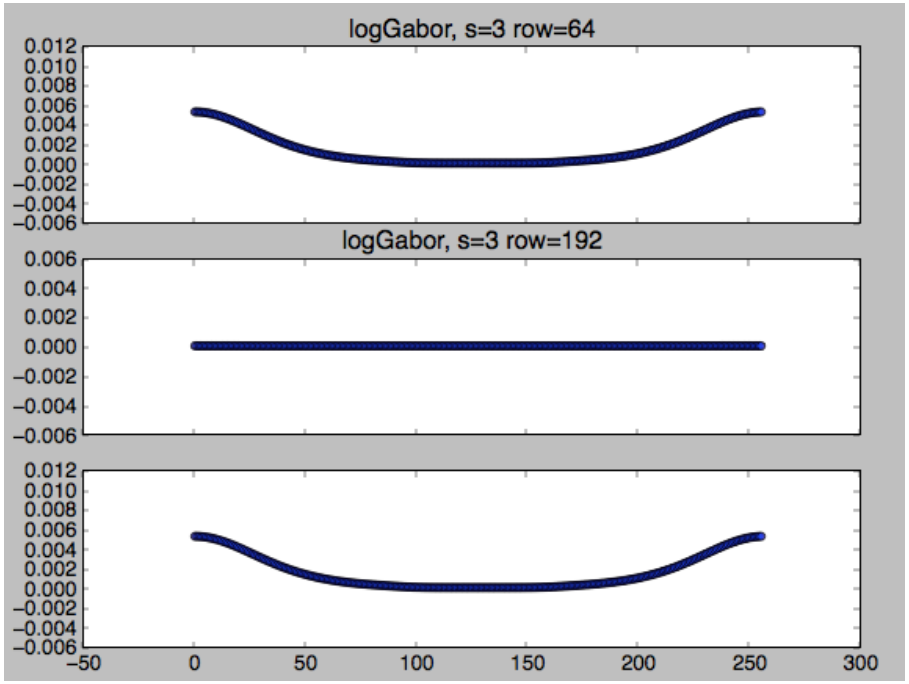


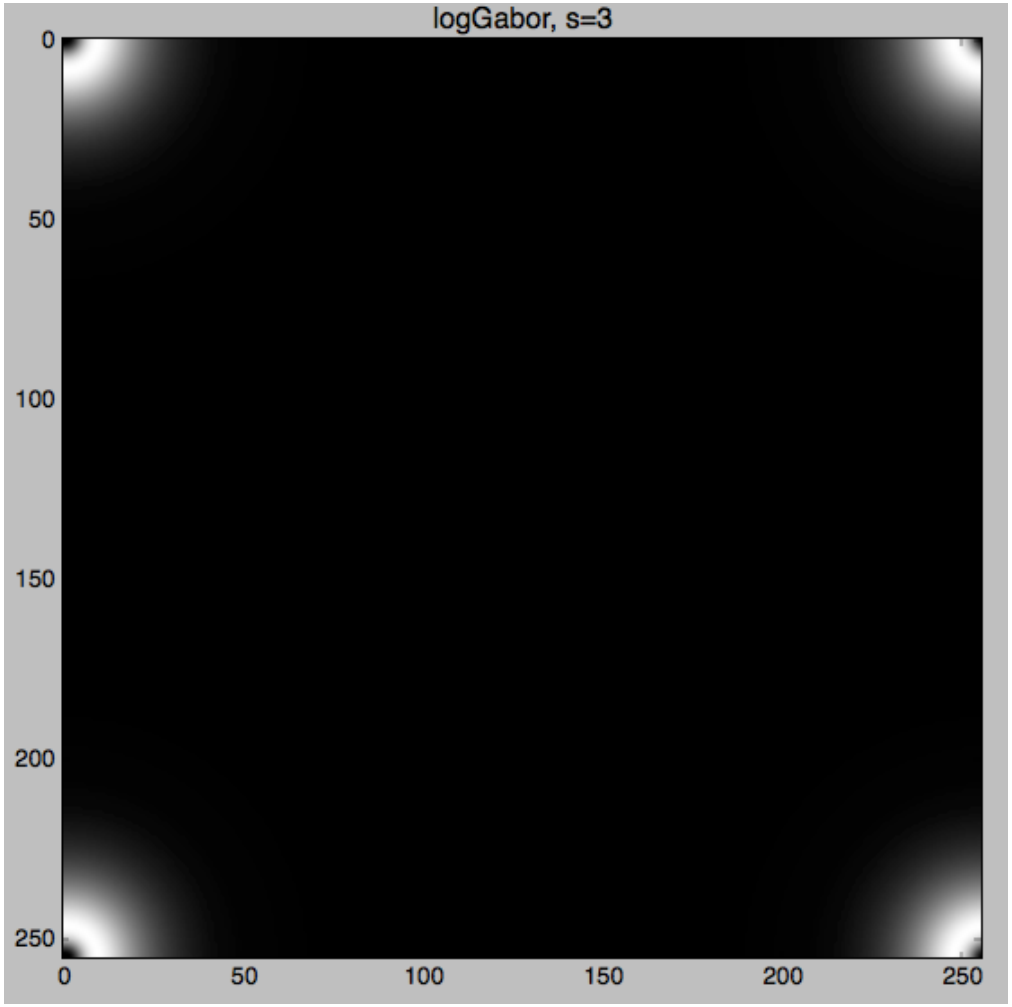
s=2, **weight** = 1. / (1. + exp(g * (cutOff – width)))
(where cutOff=0.5, g=10.)



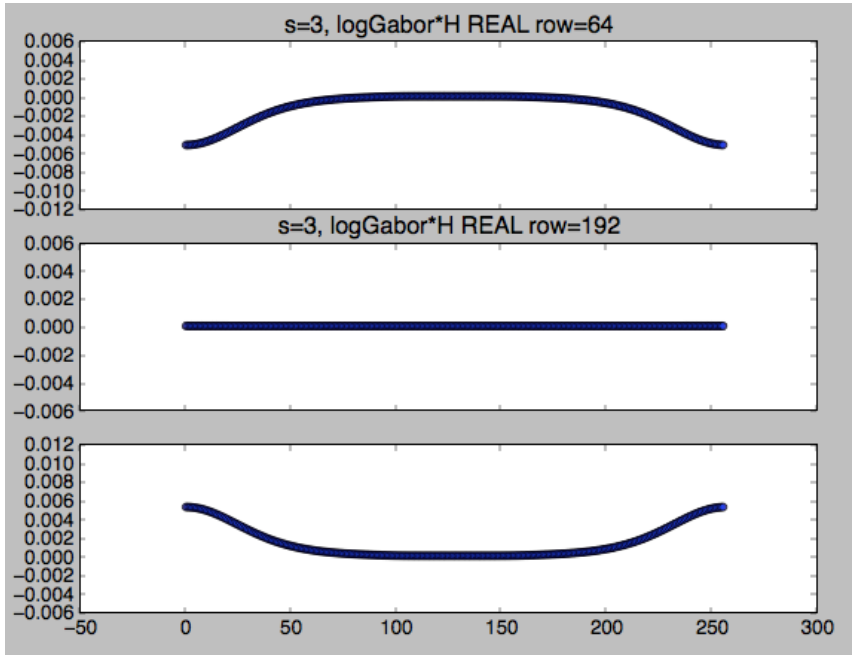


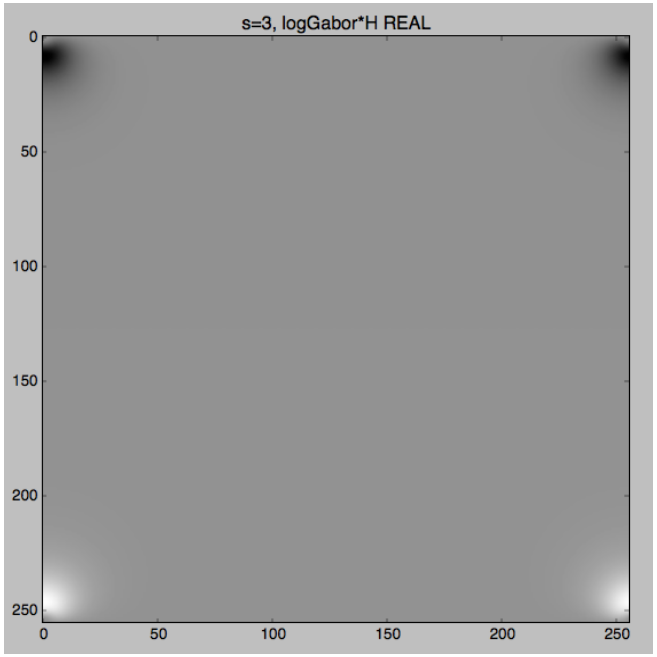
s=3, logGabor * low pass filter

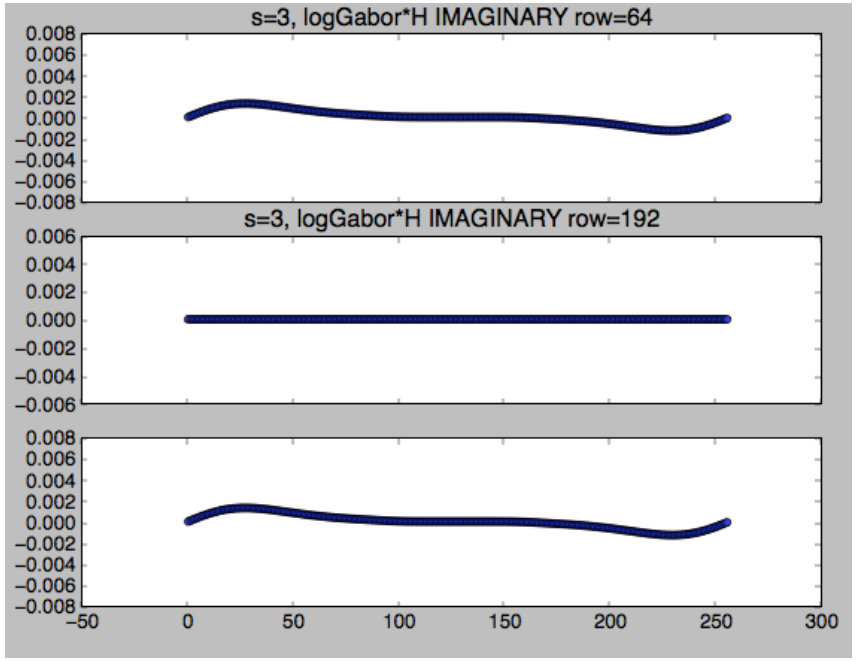


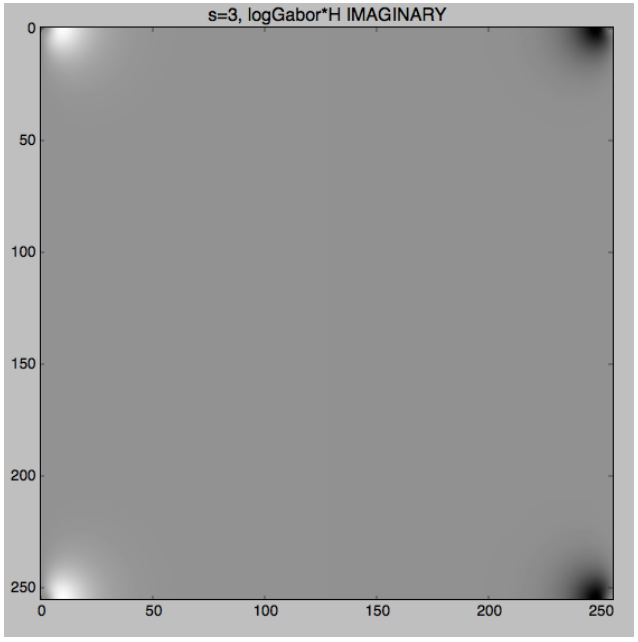


s=3, logGabor * low pass filter * H

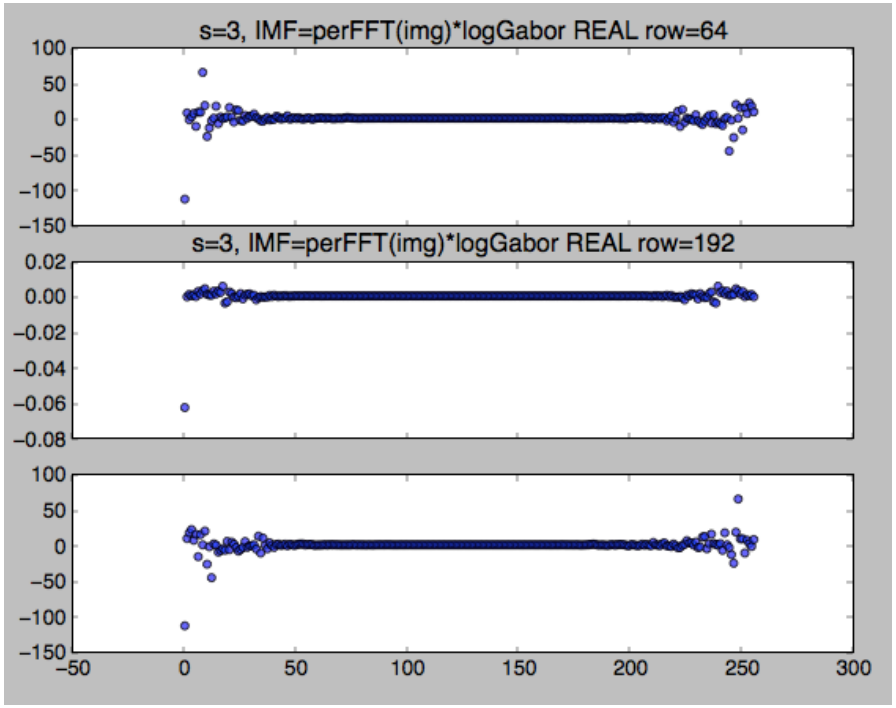


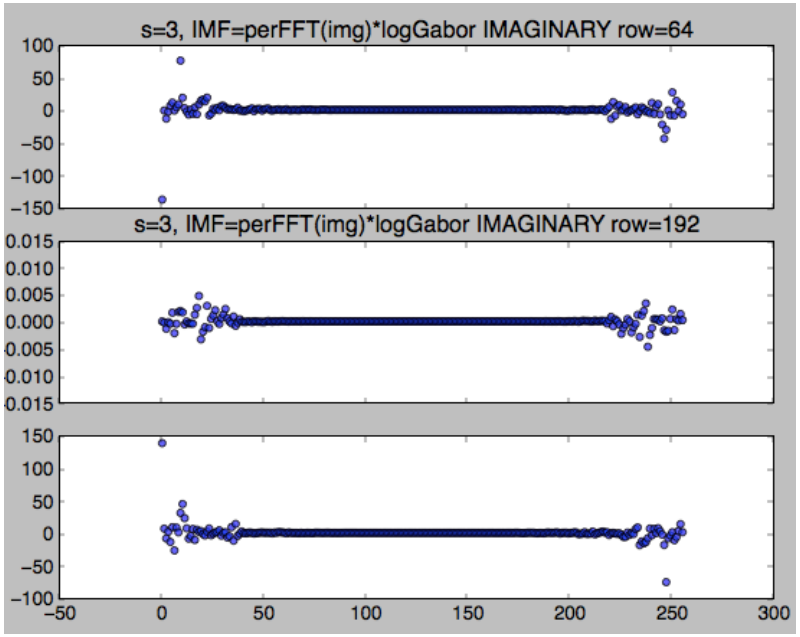


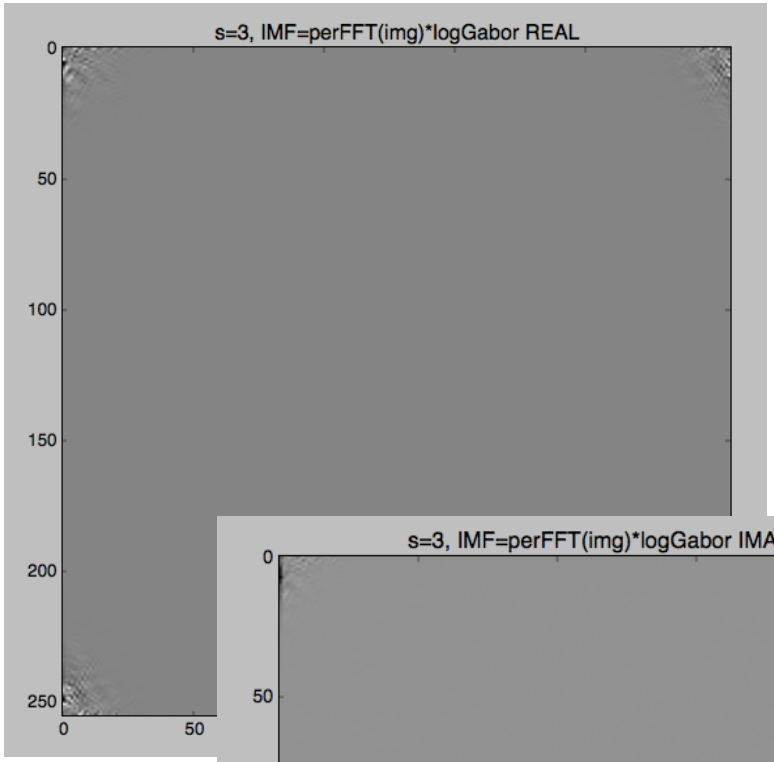


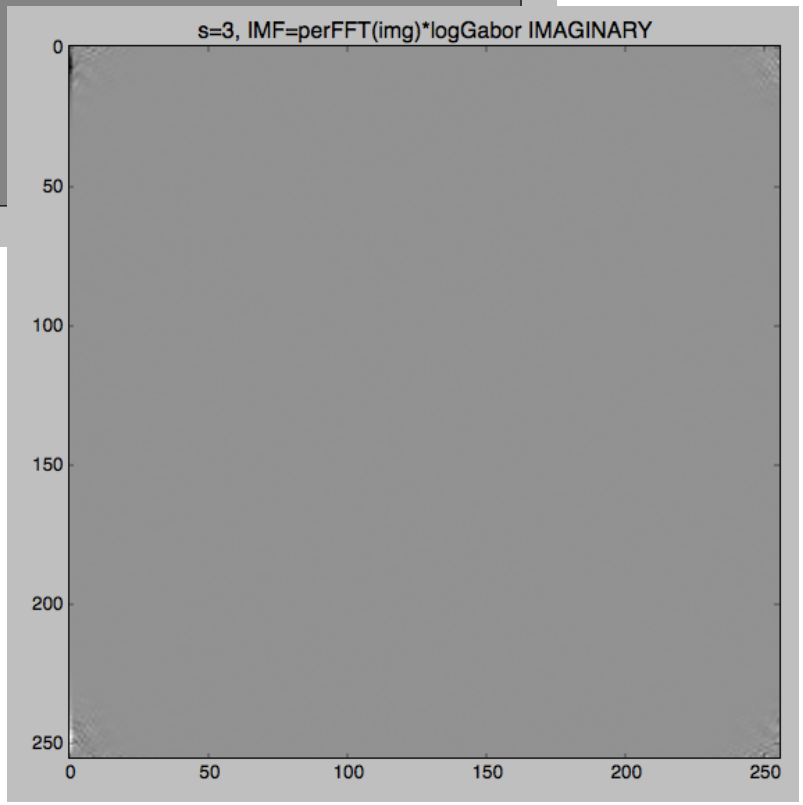


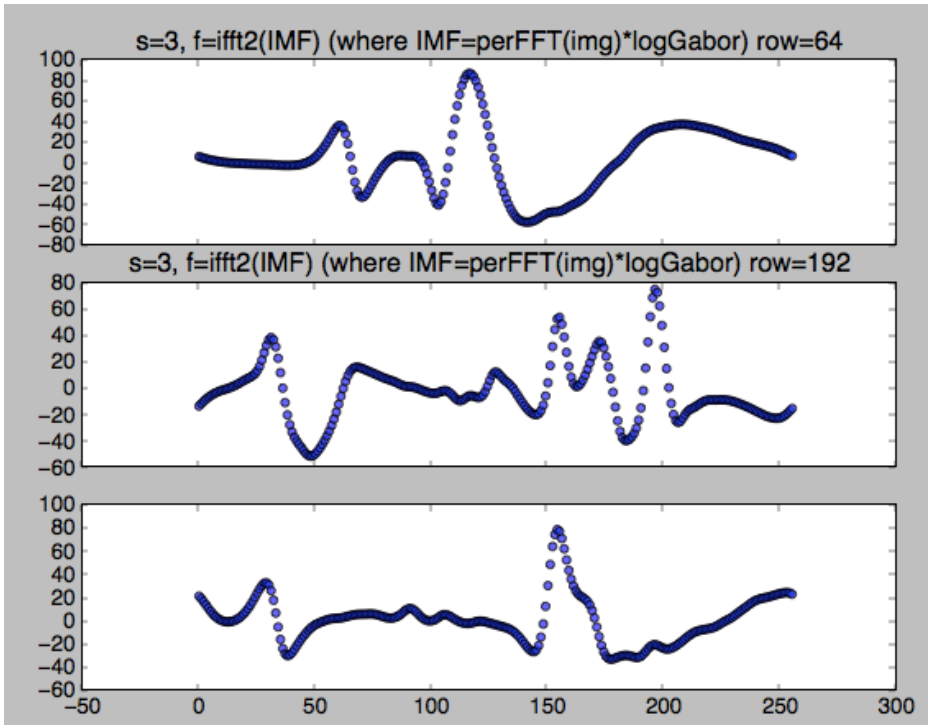
s=3, IMF=perFFT(img)*logGabor



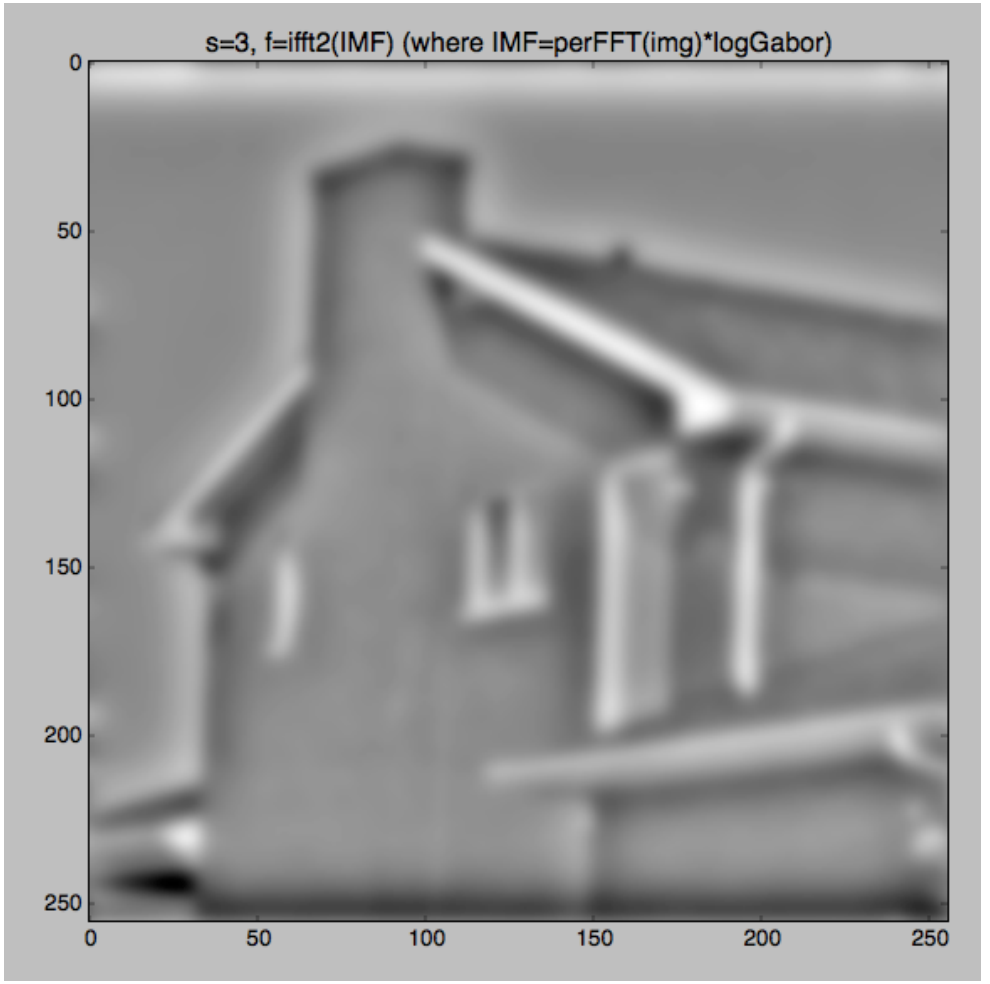




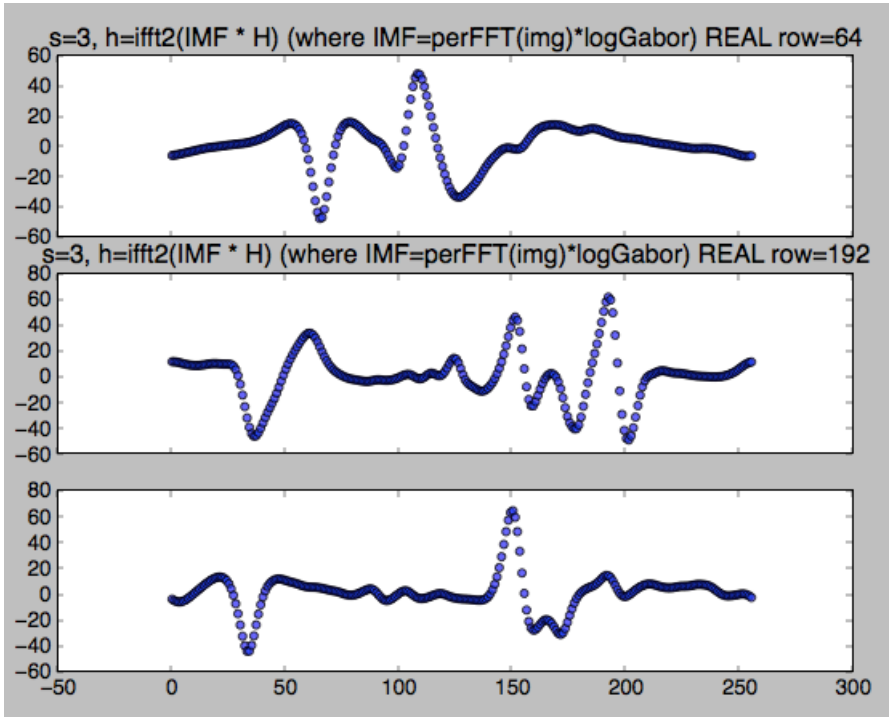


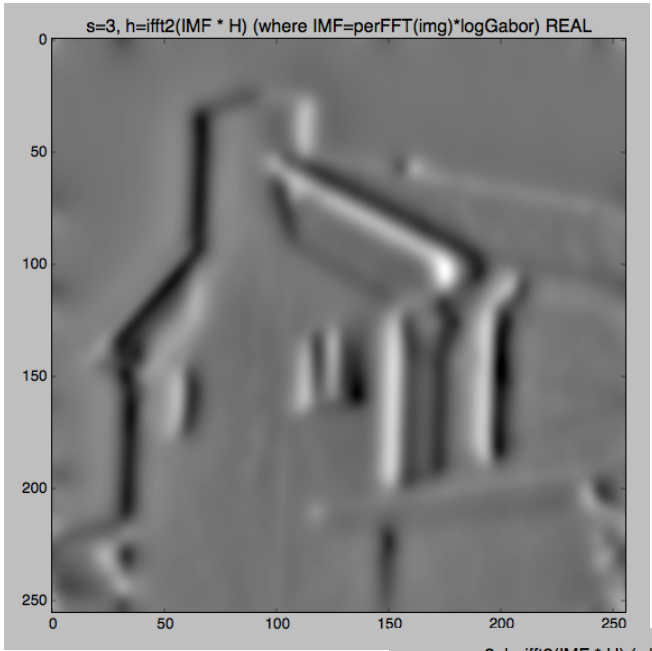


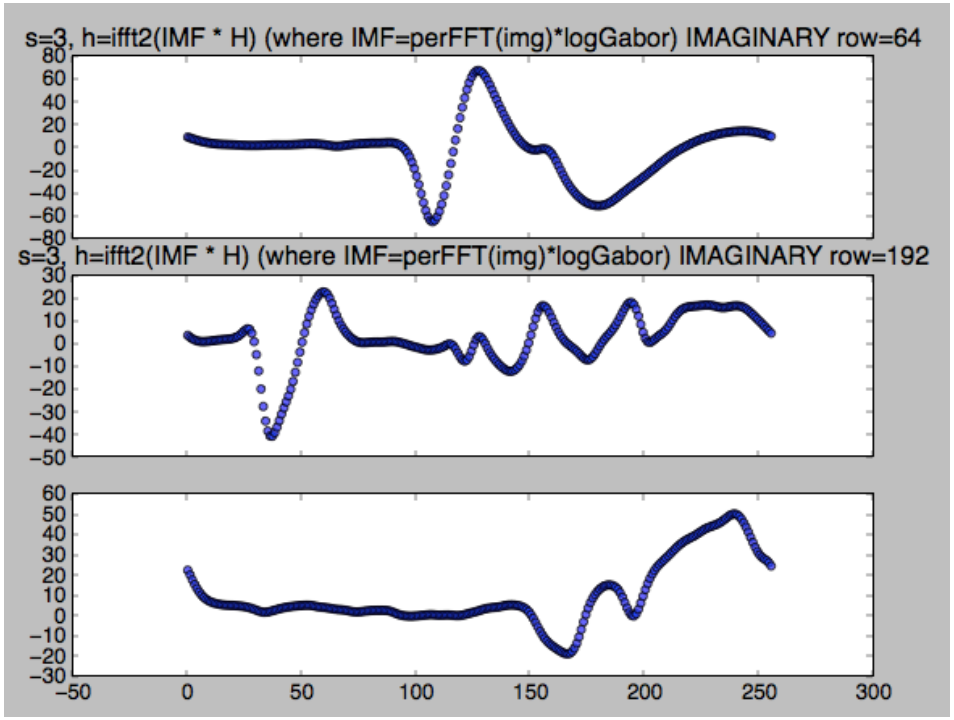
s=3, f=ifft2(IMF)
(where IMF=perFFT(img)∗logGabor)

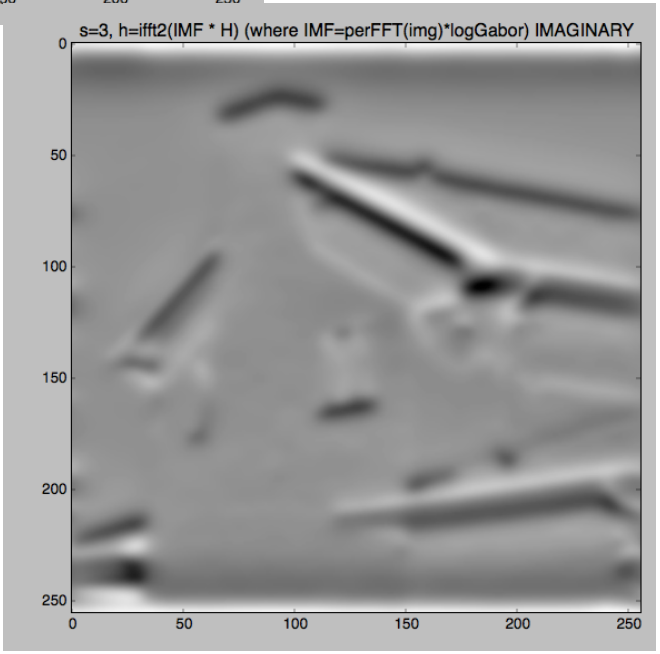


s=3, h=ifft2(IMF * H)
(where IMF=perFFT(img)*logGabor)

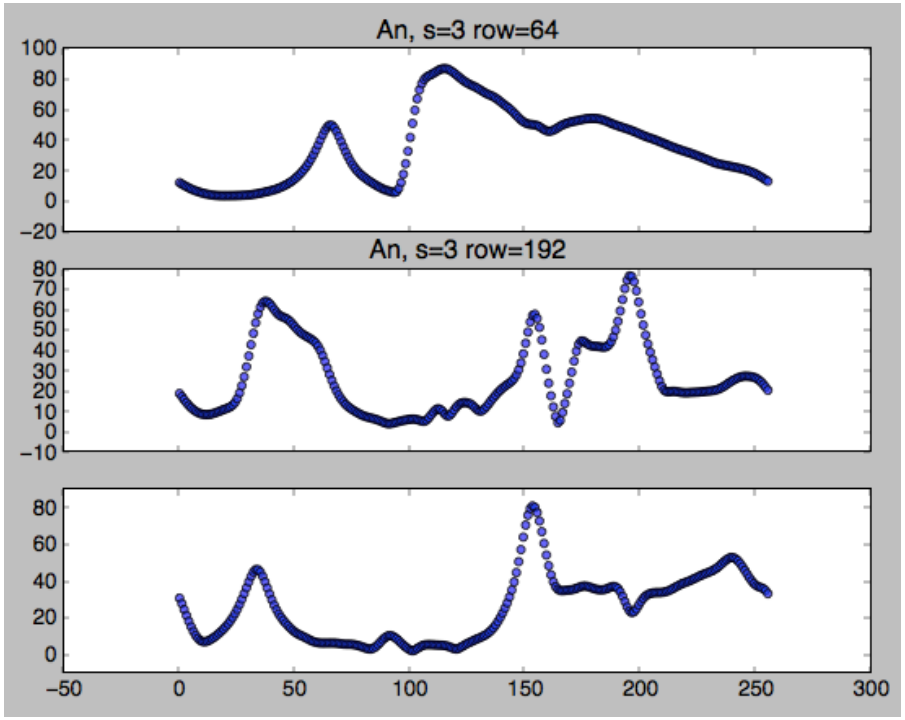


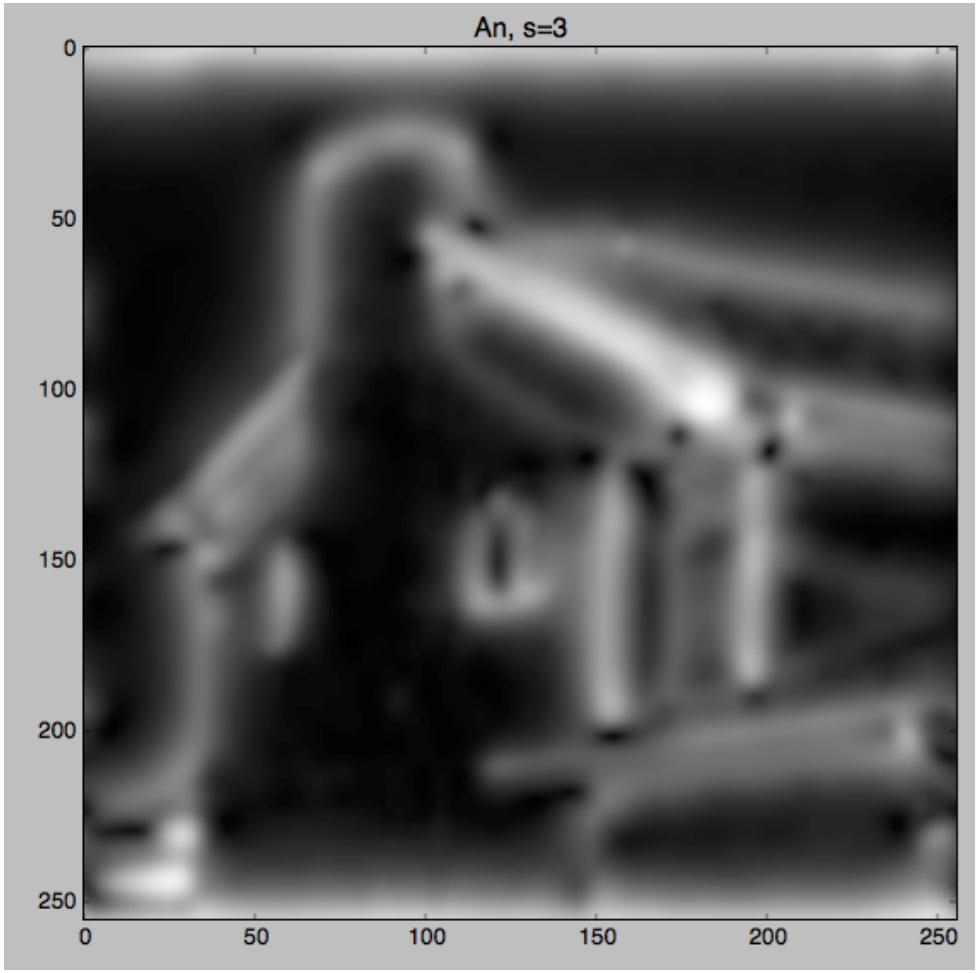




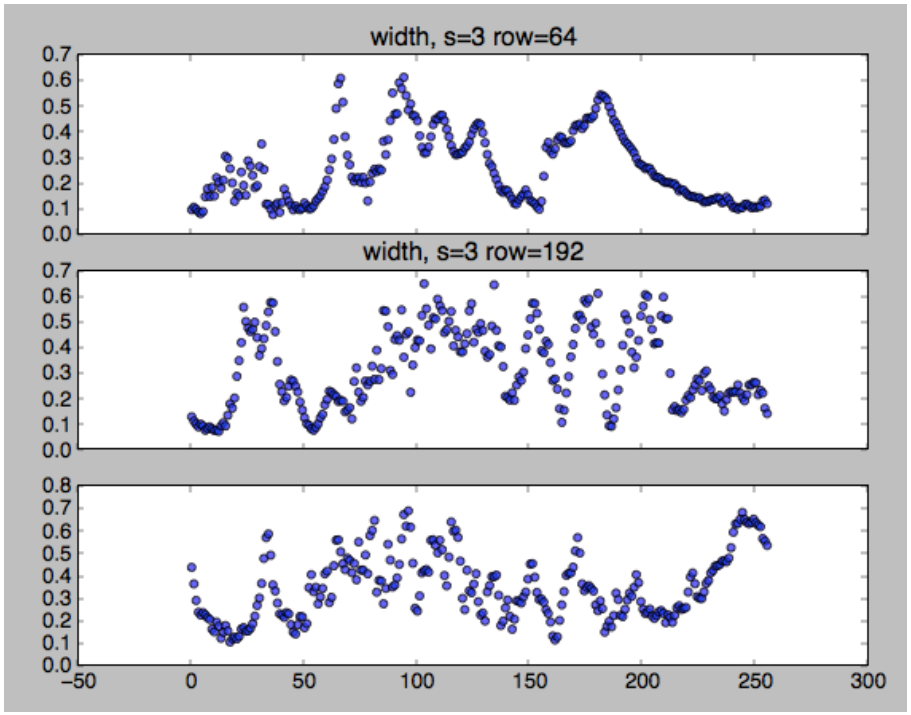


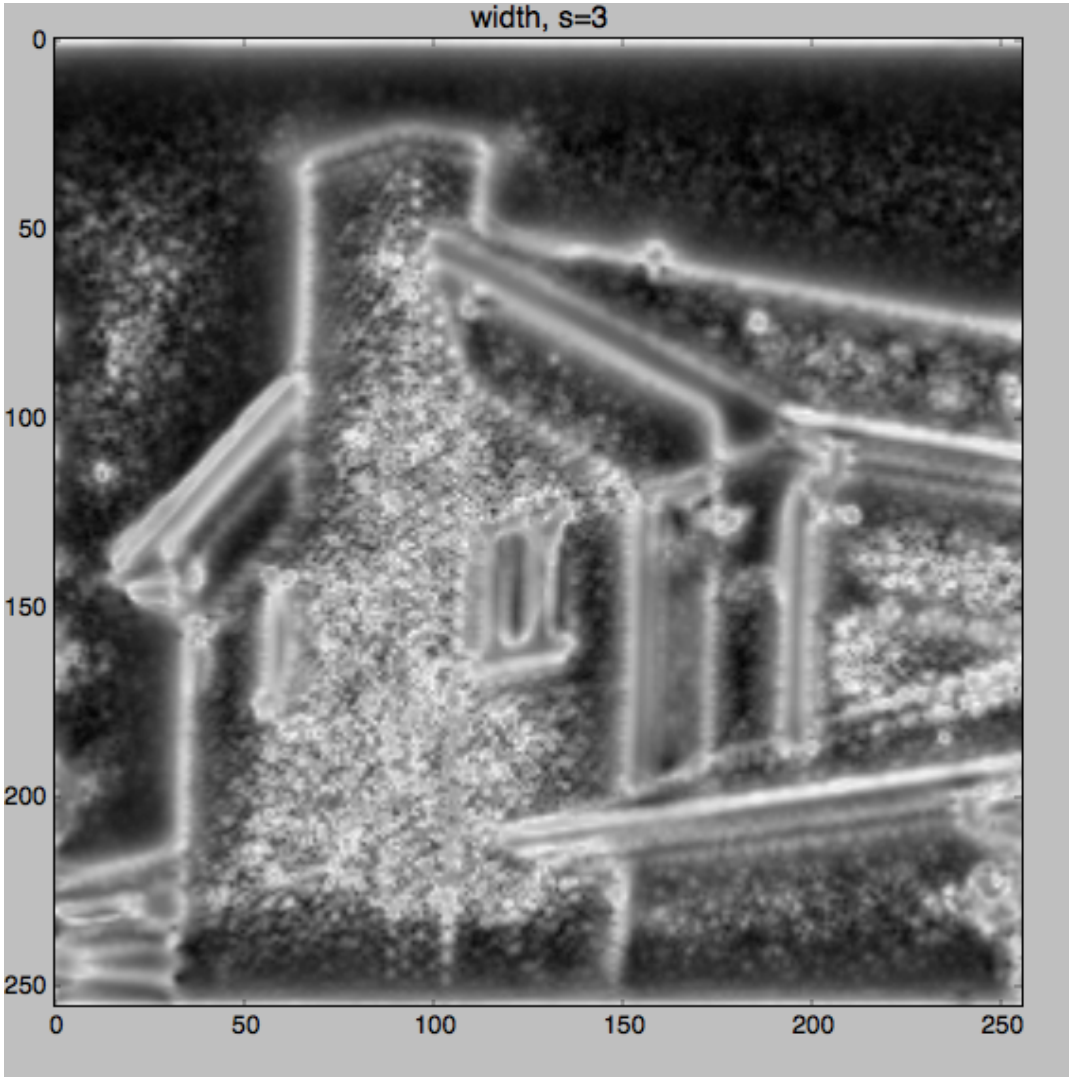
```
s=3, An = sqrt(f * f + h1 * h1 + h2 * h2)
(where f=ifft2(perFFT(img)*logGabor)
   and h=ifft2(perFFT(img)*logGabor * H) )
```



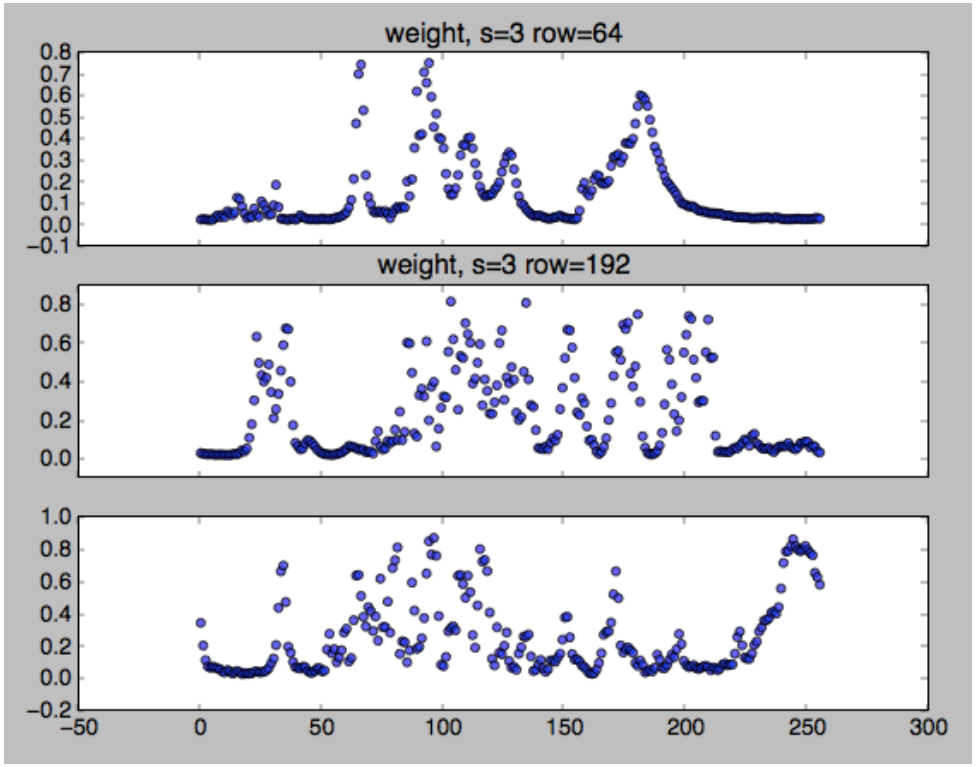


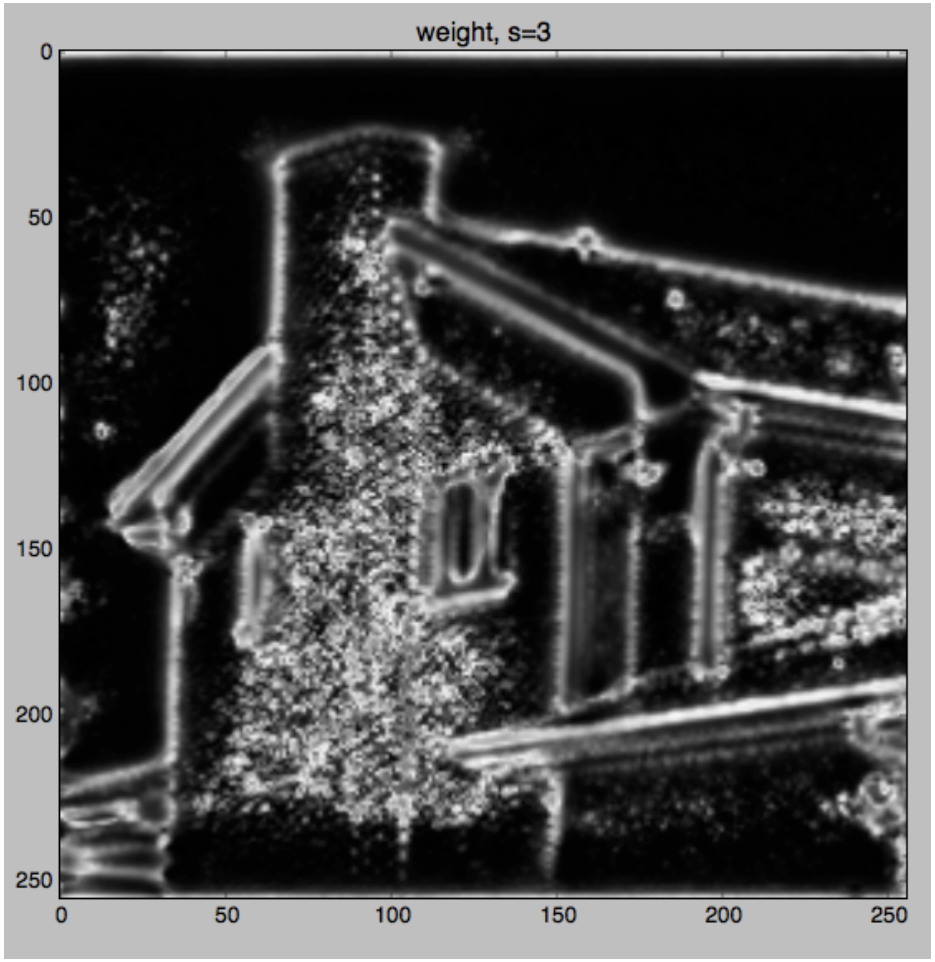
s=3, **width** = (sumAn / (maxAn + epsilon) – 1.) / (nscale – 1)



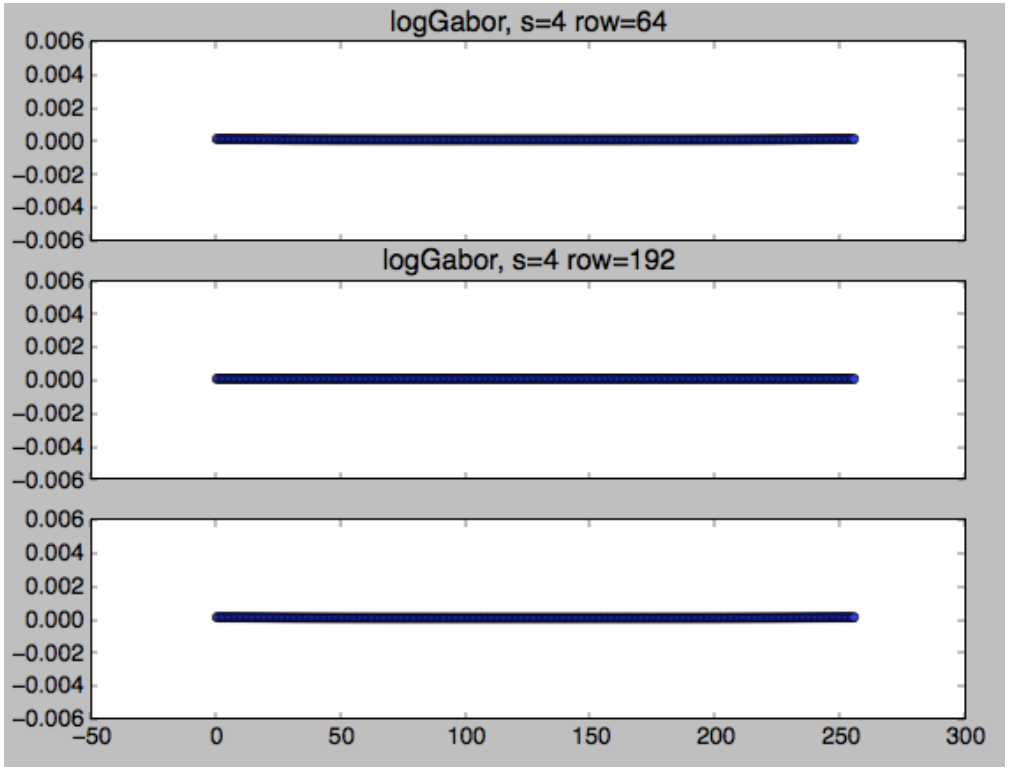


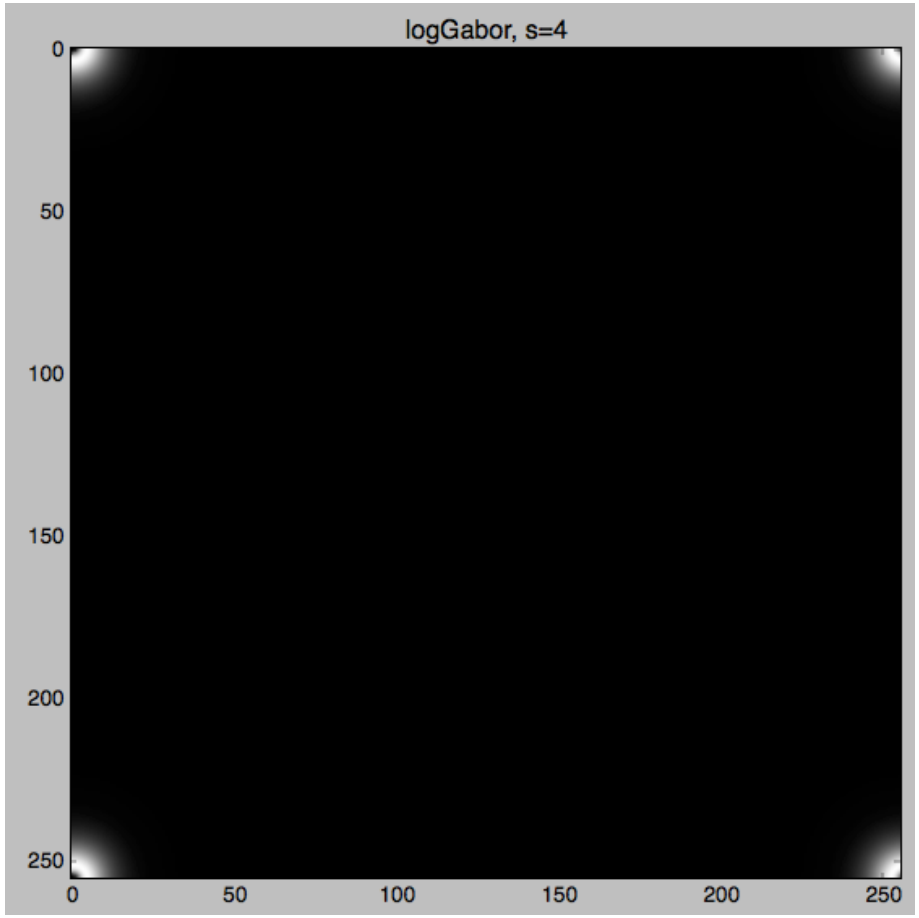
s=3, **weight** = 1. / (1. + exp(g * (cutOff – width)))
(where cutOff=0.5, g=10.)



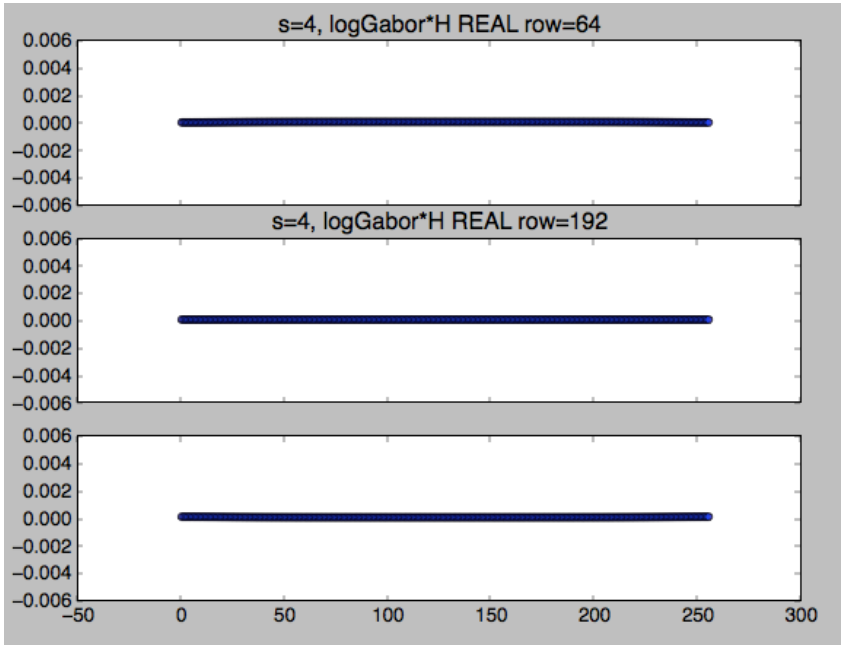


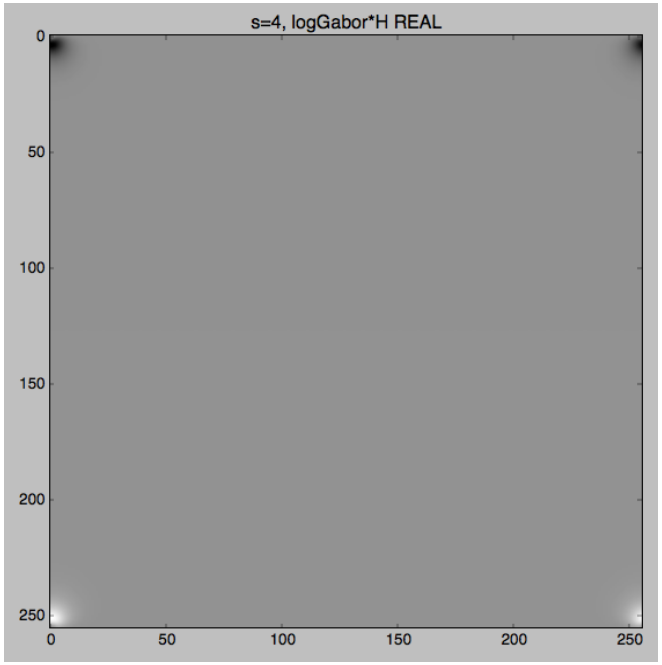
s=4, logGabor * low pass filter

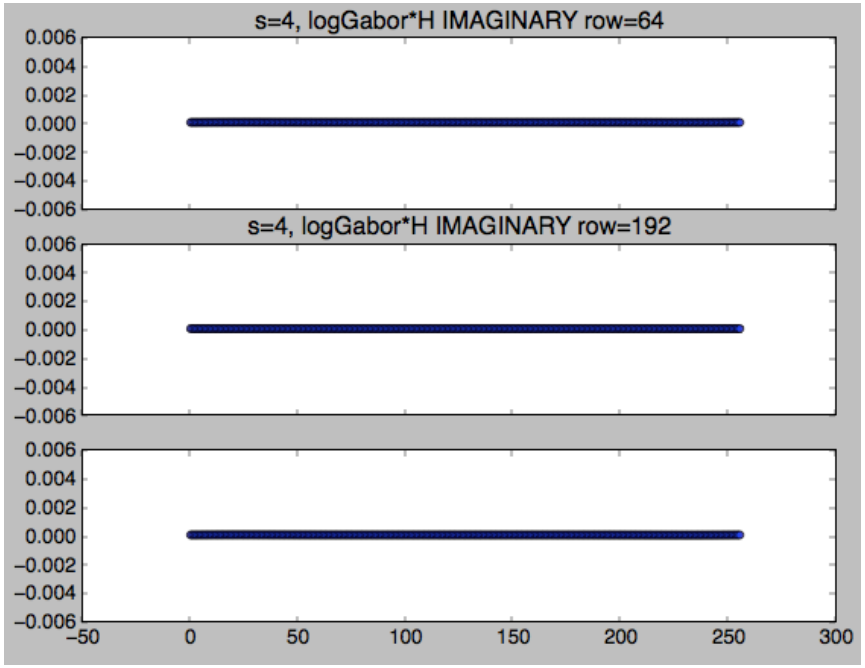


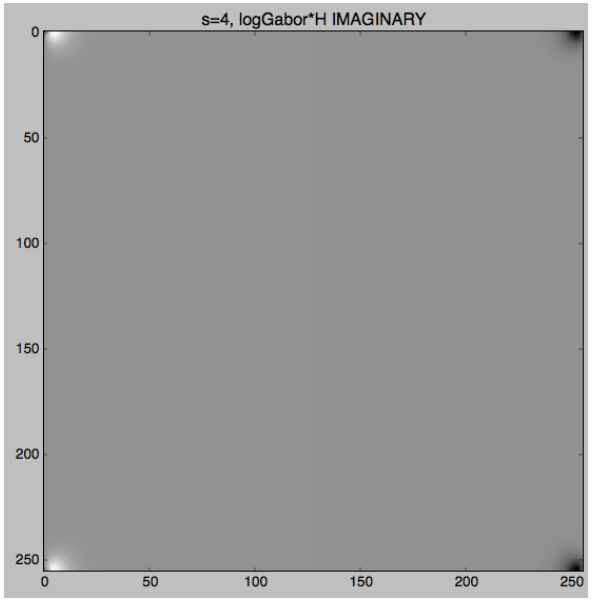


s=4, logGabor * low pass filter * H

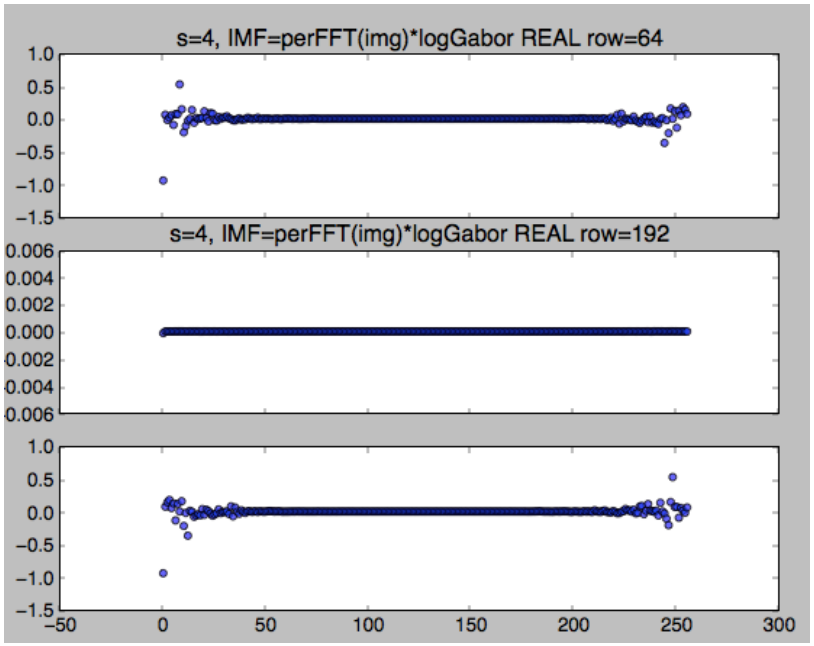


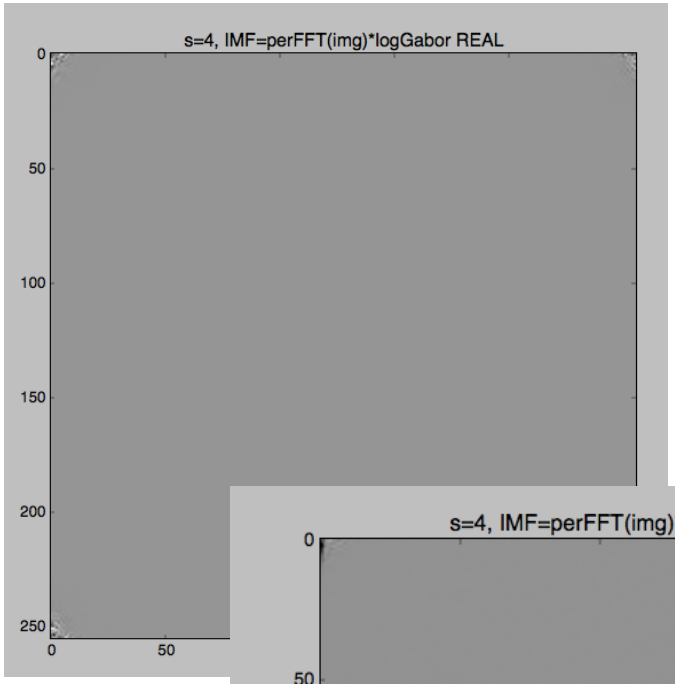


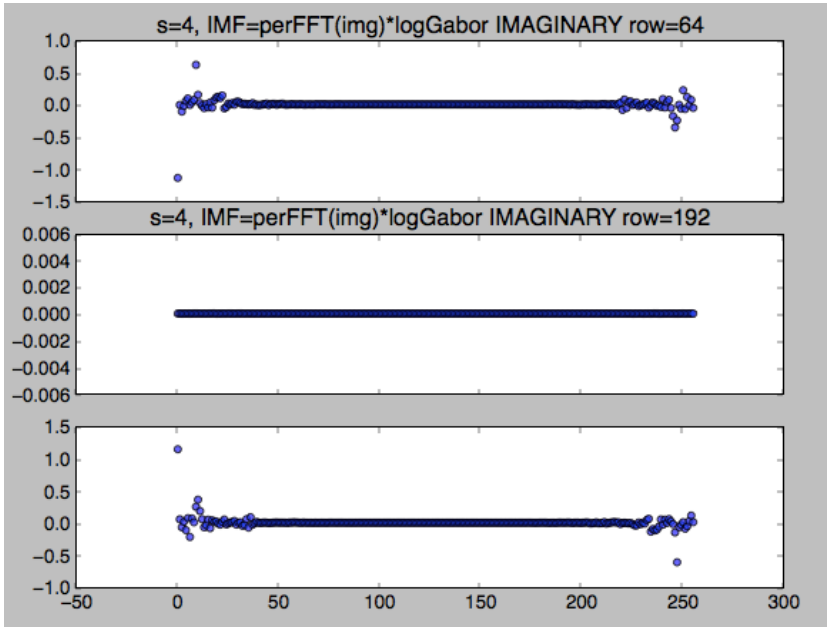


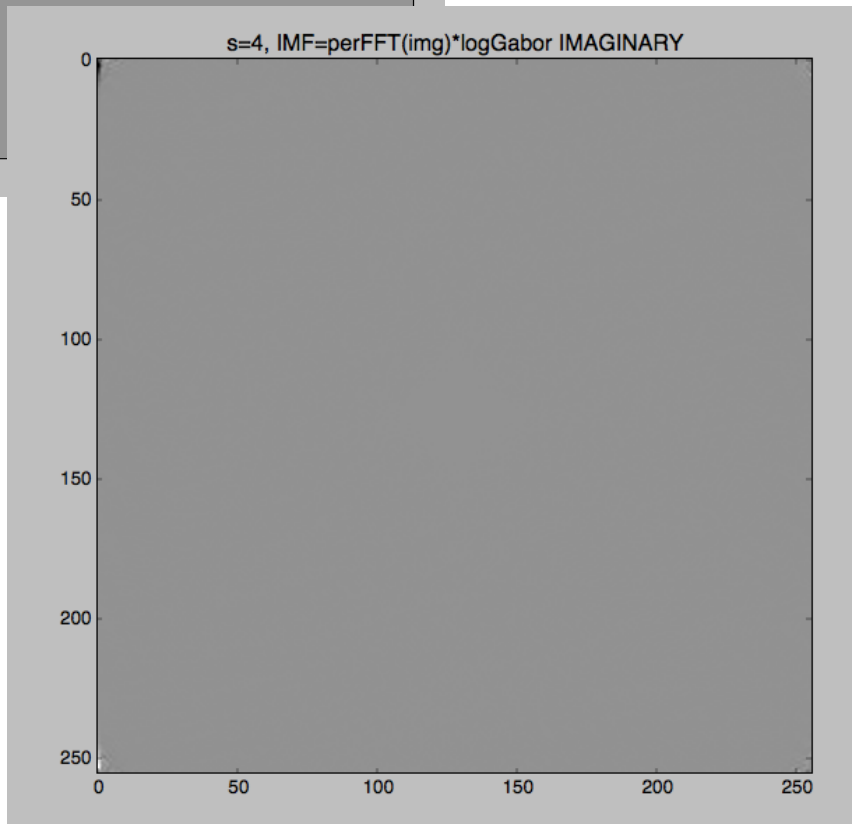


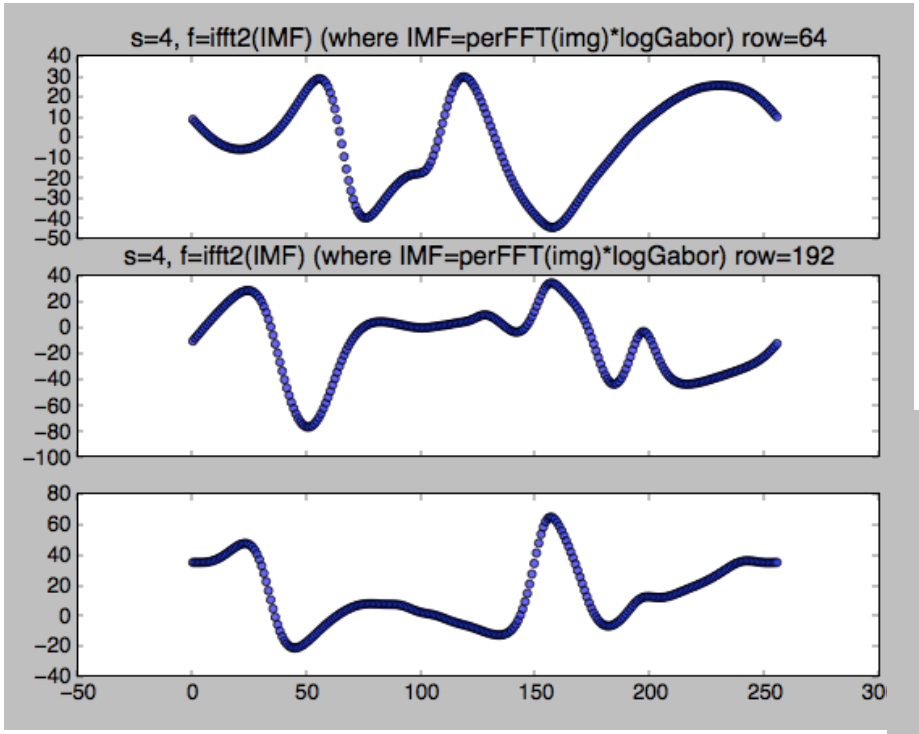
s=4, IMF=perFFT(img)*logGabor



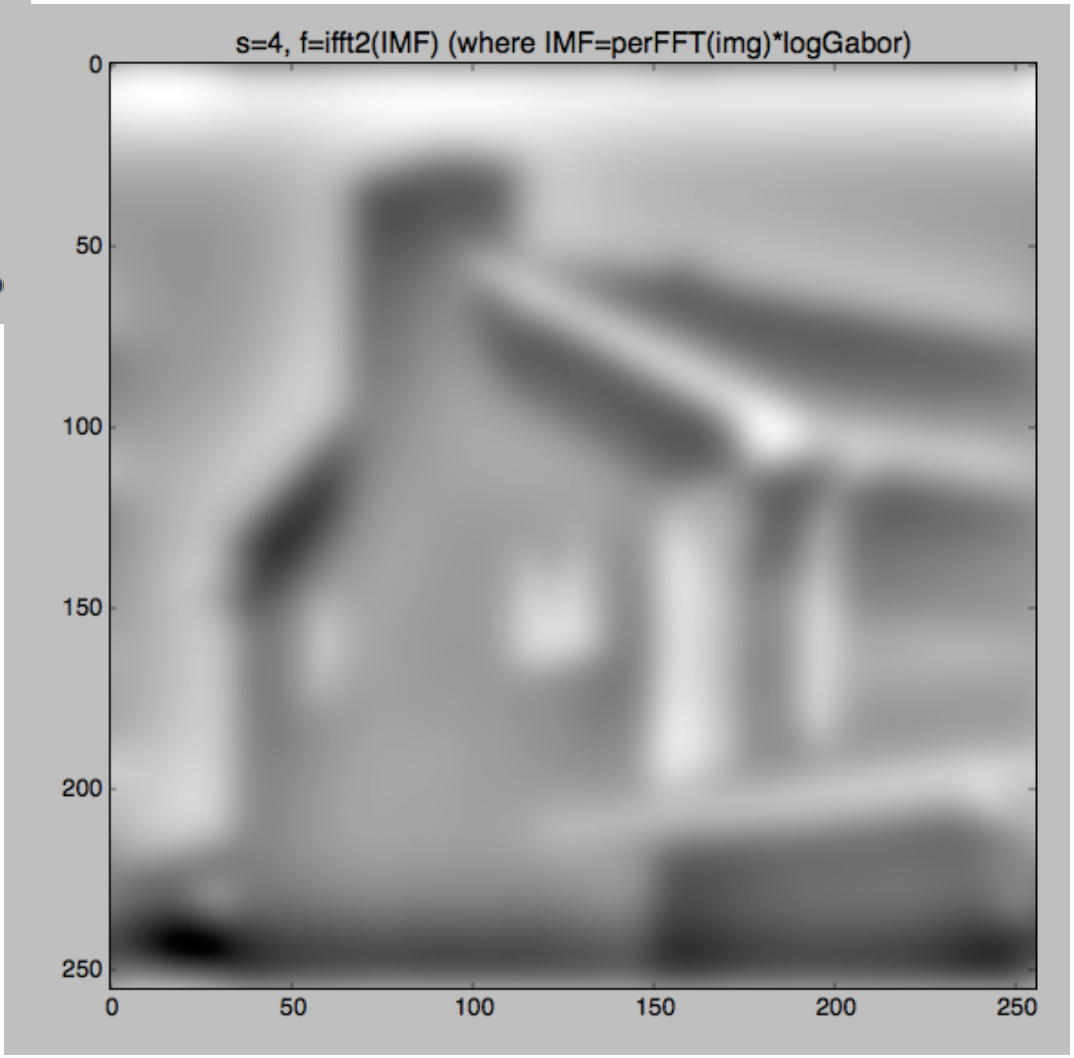




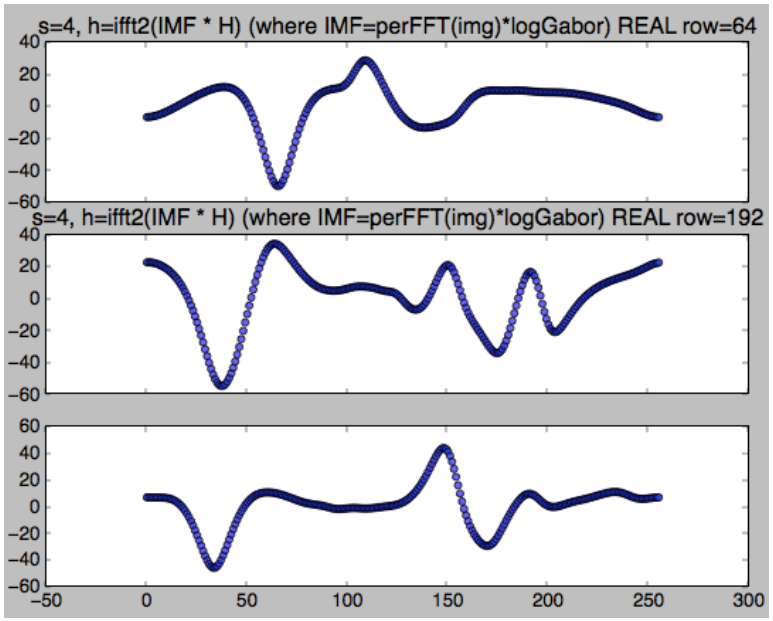


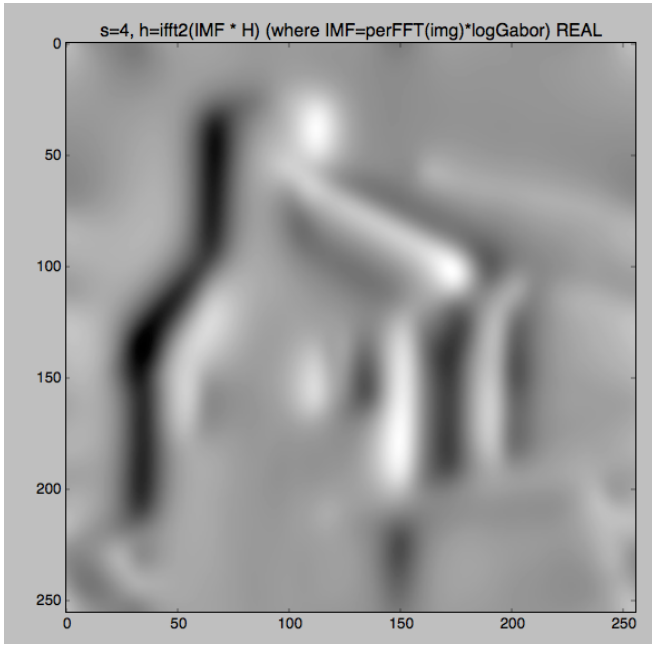


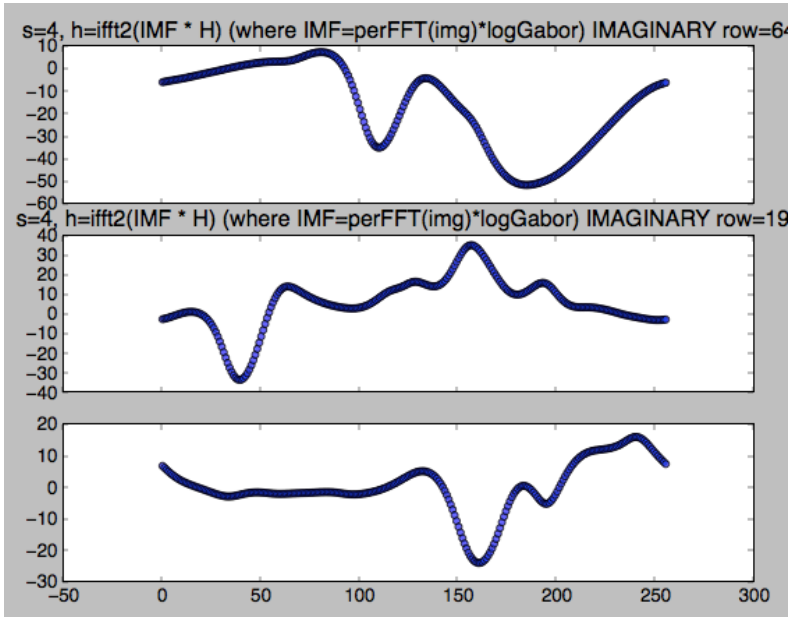
s=4, f=ifft2(IMF)
(where IMF=perFFT(img)*logGabor)

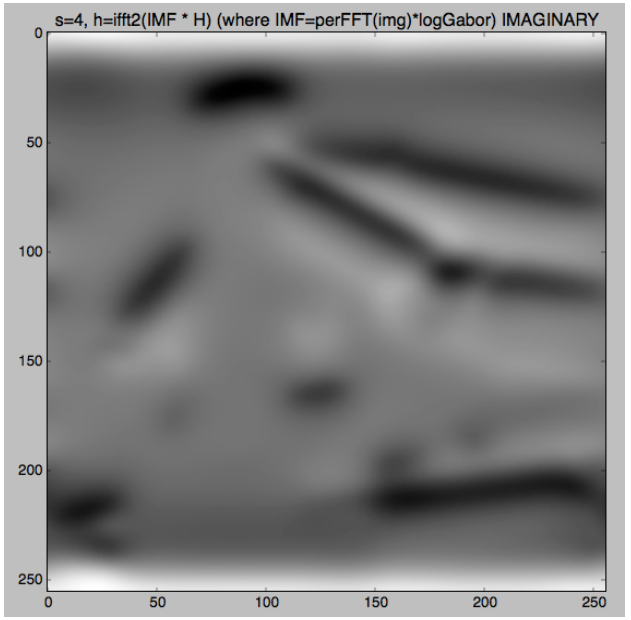


s=4, h=ifft2(IMF * H)
(where IMF=perFFT(img)*logGabor)

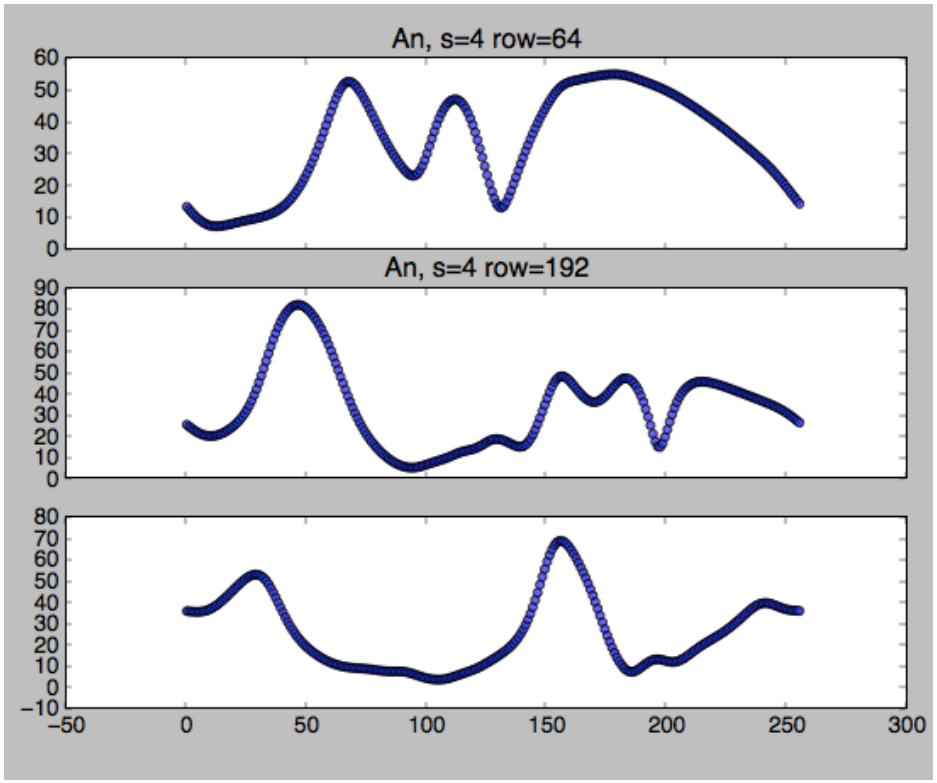


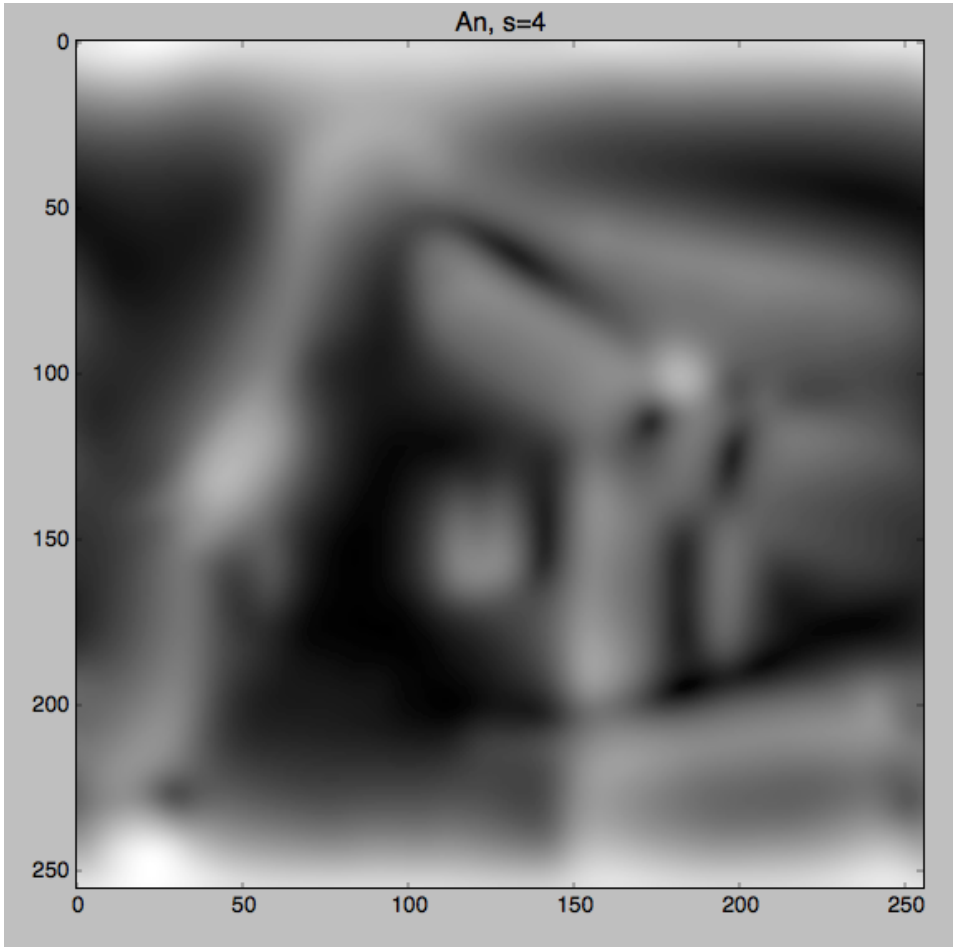




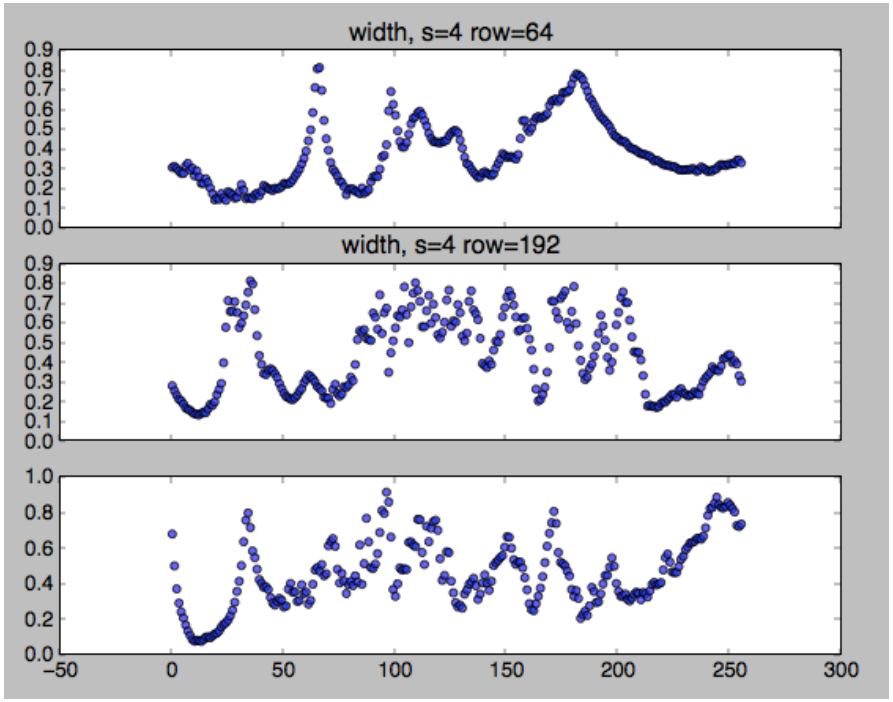


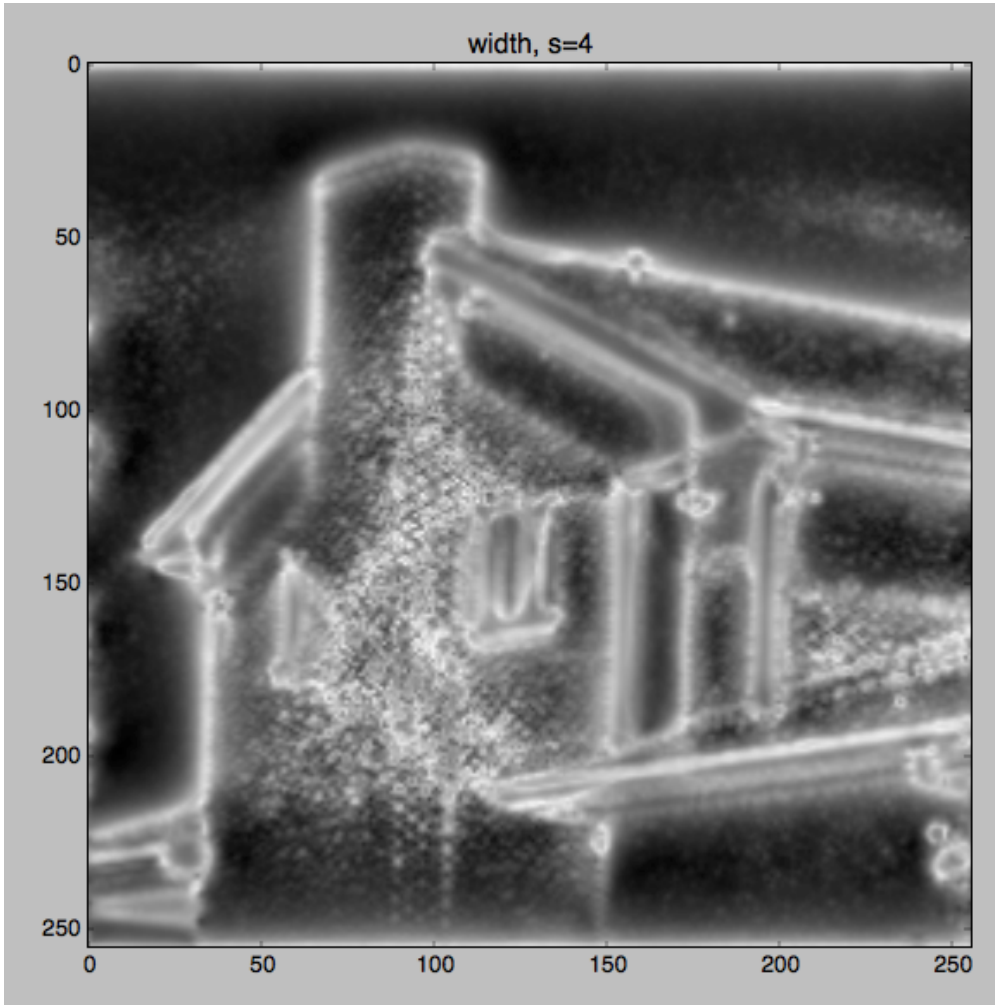
```
s=4, An = sqrt(f * f + h1 * h1 + h2 * h2)
(where f=ifft2(perFFT(img)*logGabor)
   and h=ifft2(perFFT(img)*logGabor * H) )
```



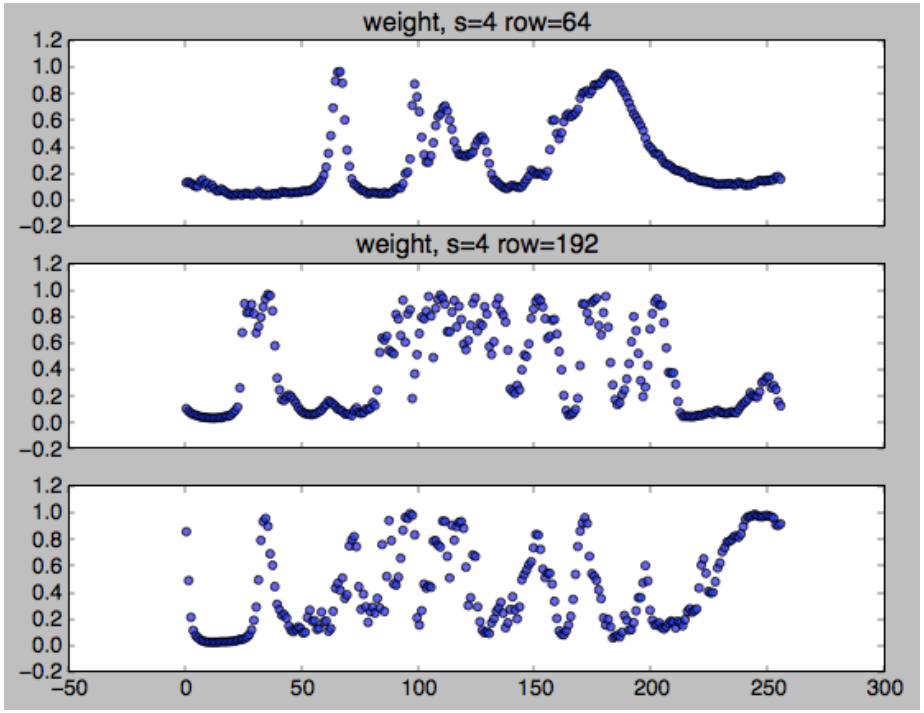


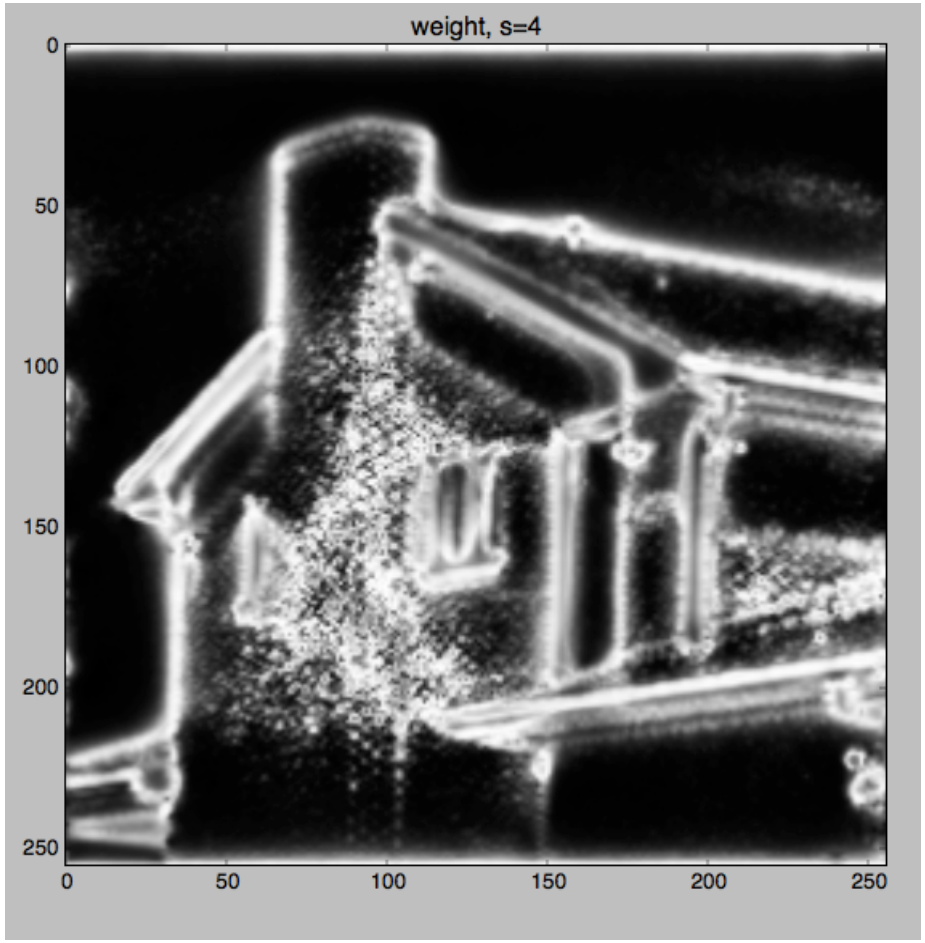
s=4, **width** = (sumAn / (maxAn + epsilon) – 1.) / (nscale – 1)



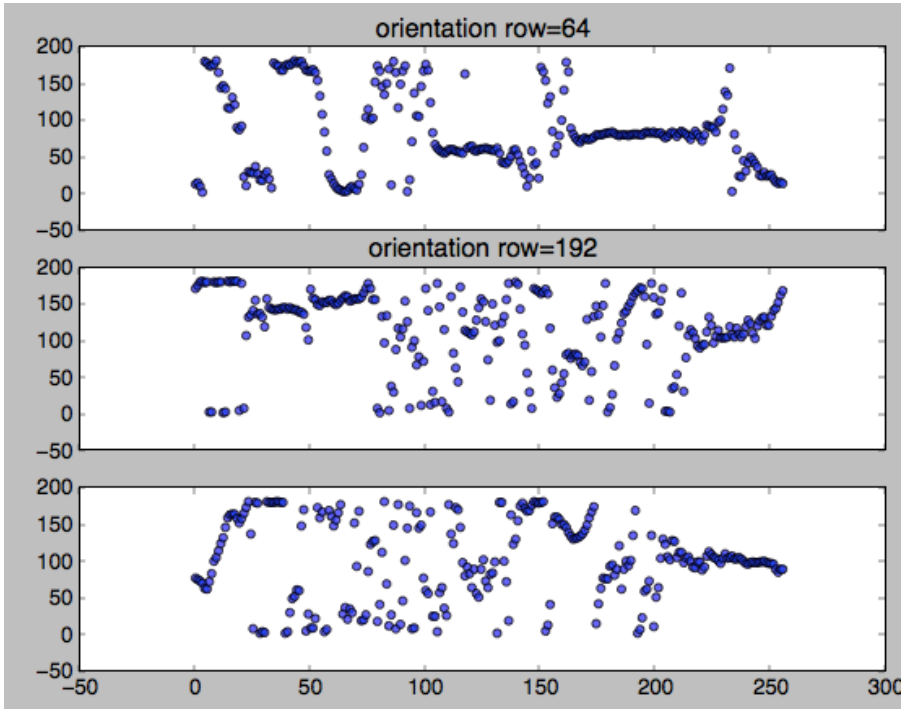


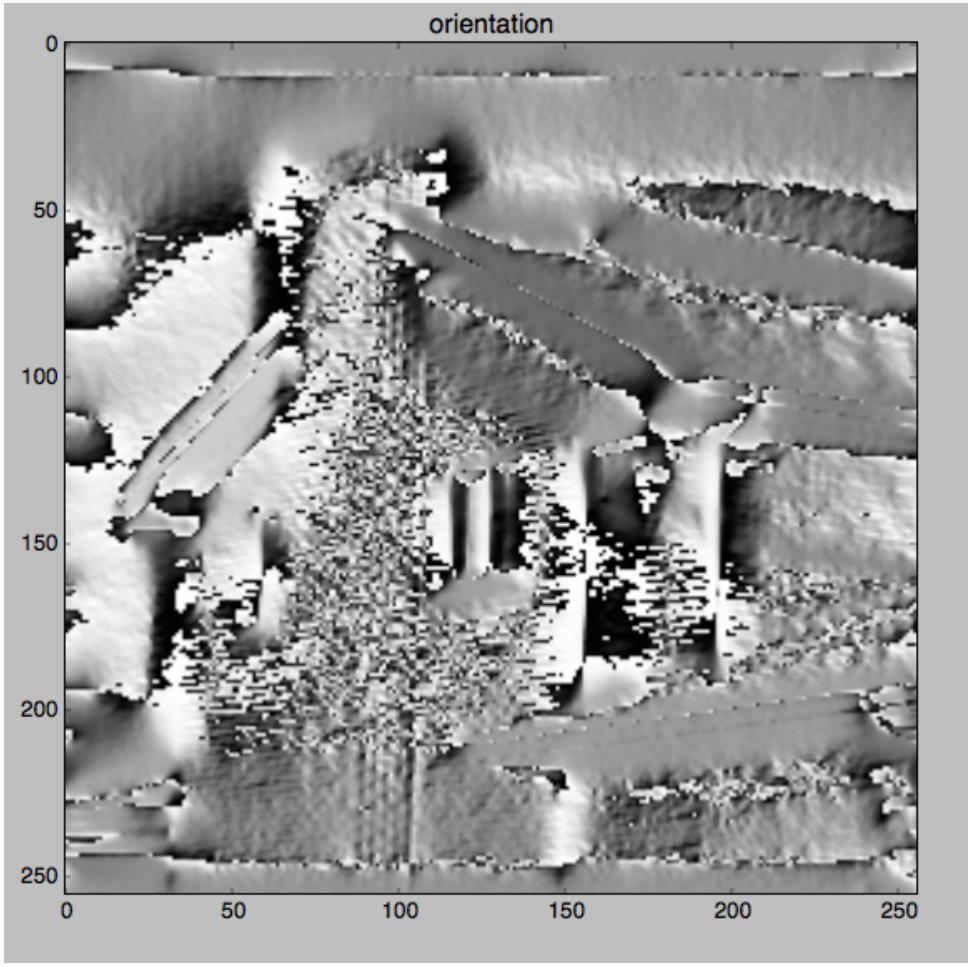
```
s=4, weight = 1. / (1. + exp(g * (cutOff - width)))
    (where cutOff=0.5, g=10.)
```



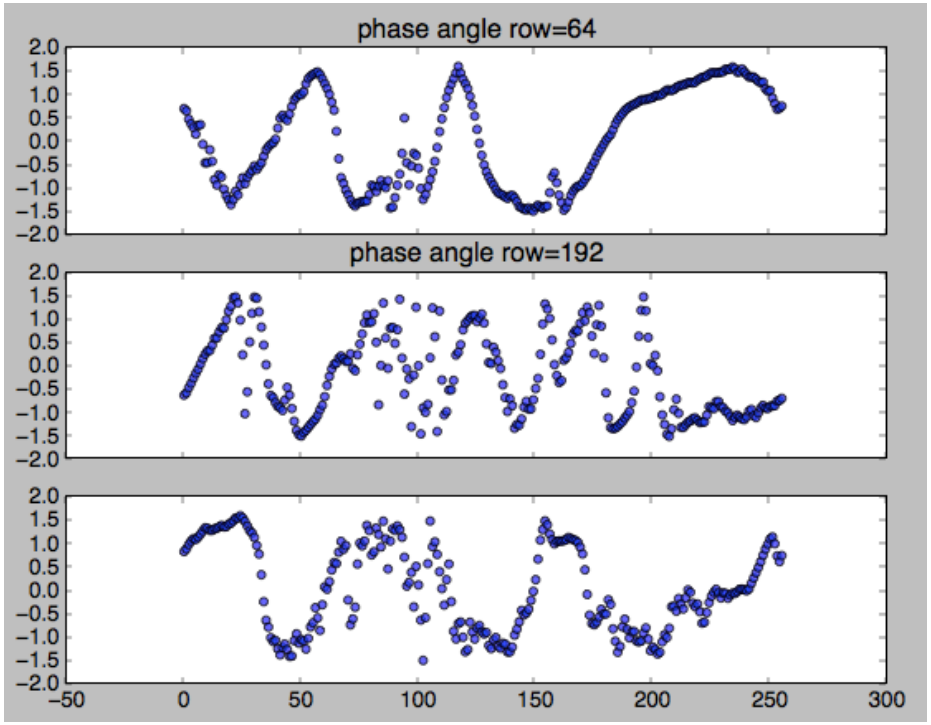


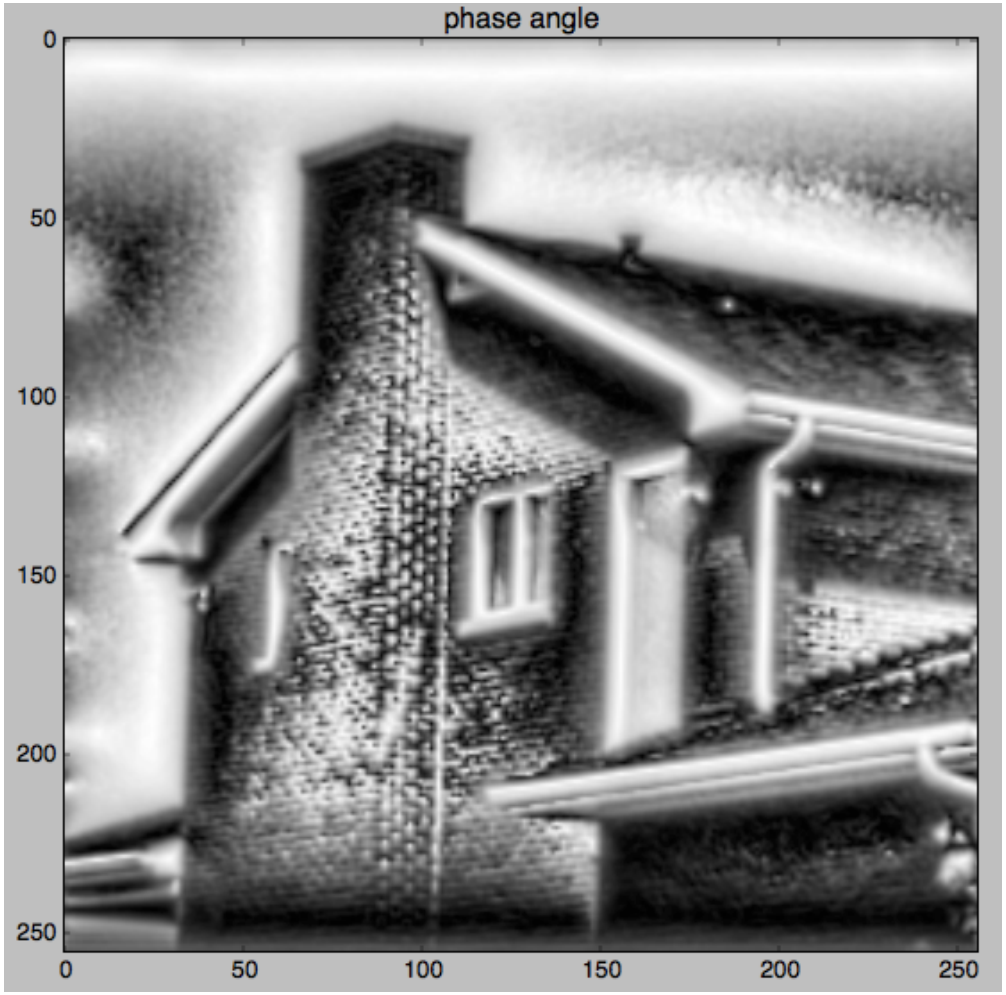
orientation



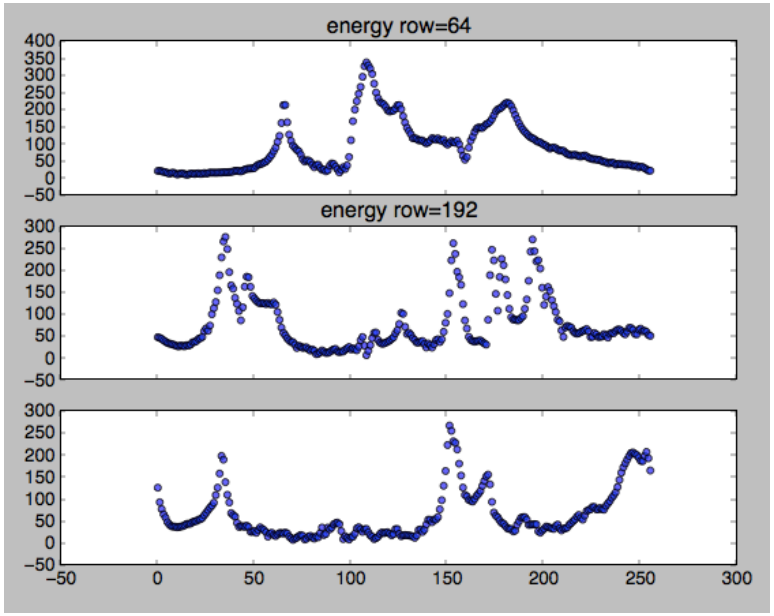


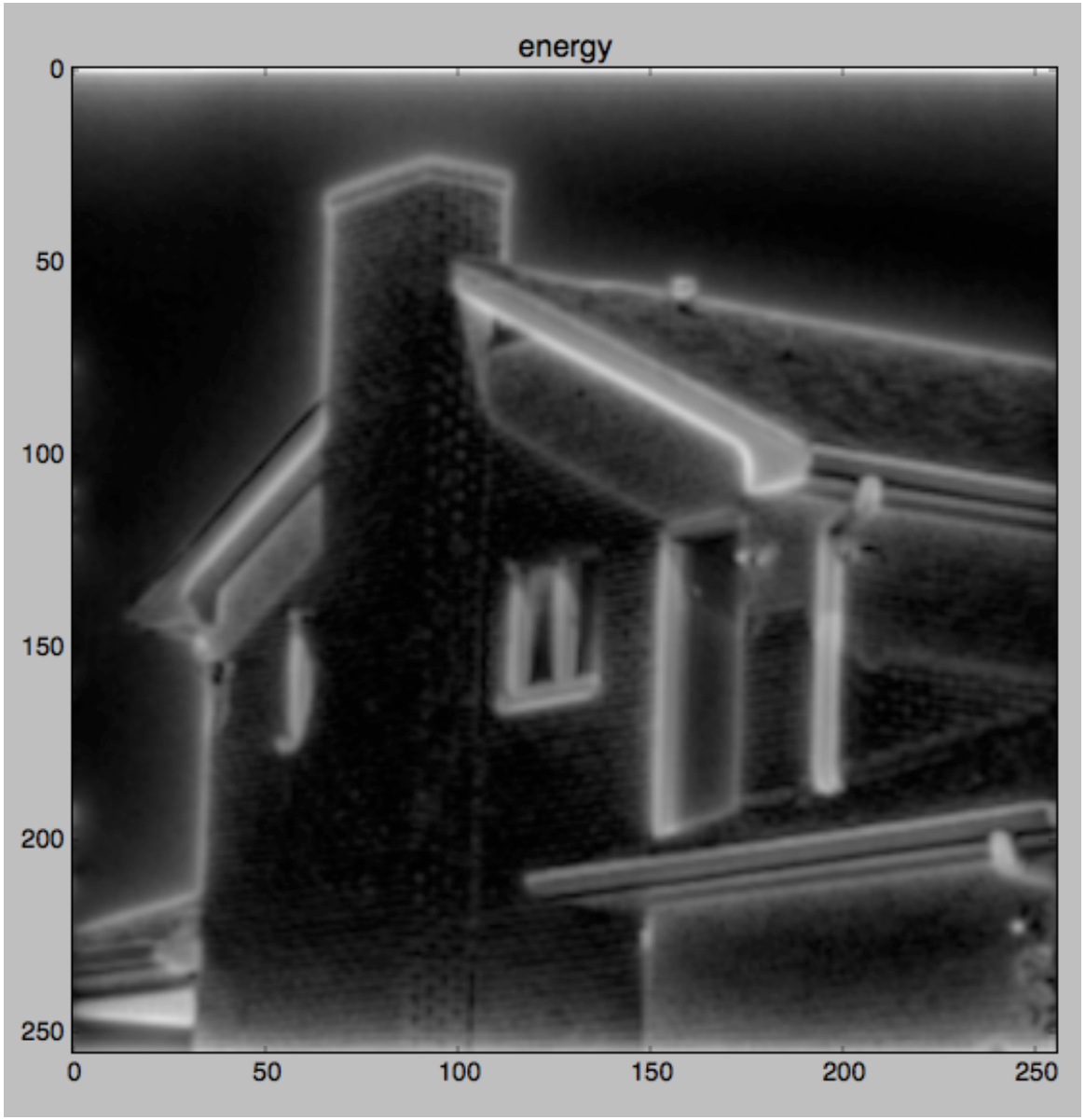
phase angle

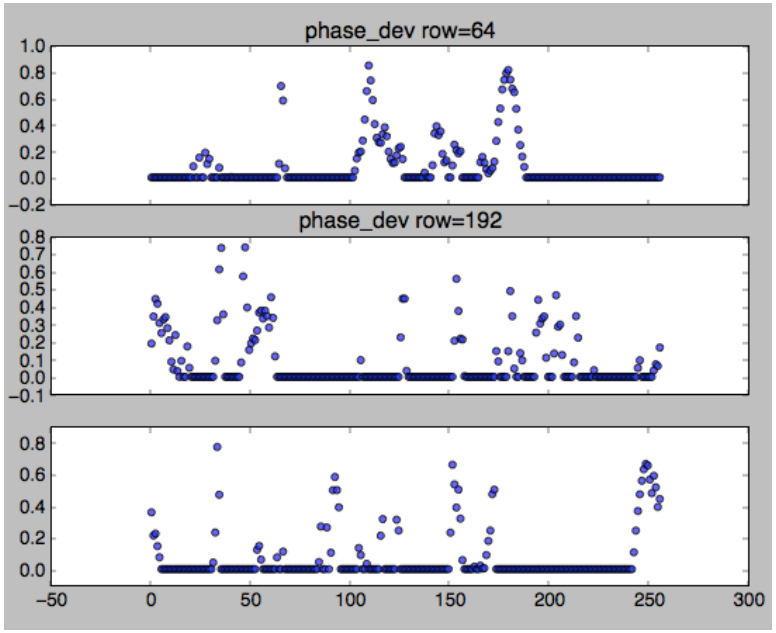

phase angle row=64
phase angle row=192
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−50
0
50
100
150
200
250
300


phase angle
0
50
100
150
200
250

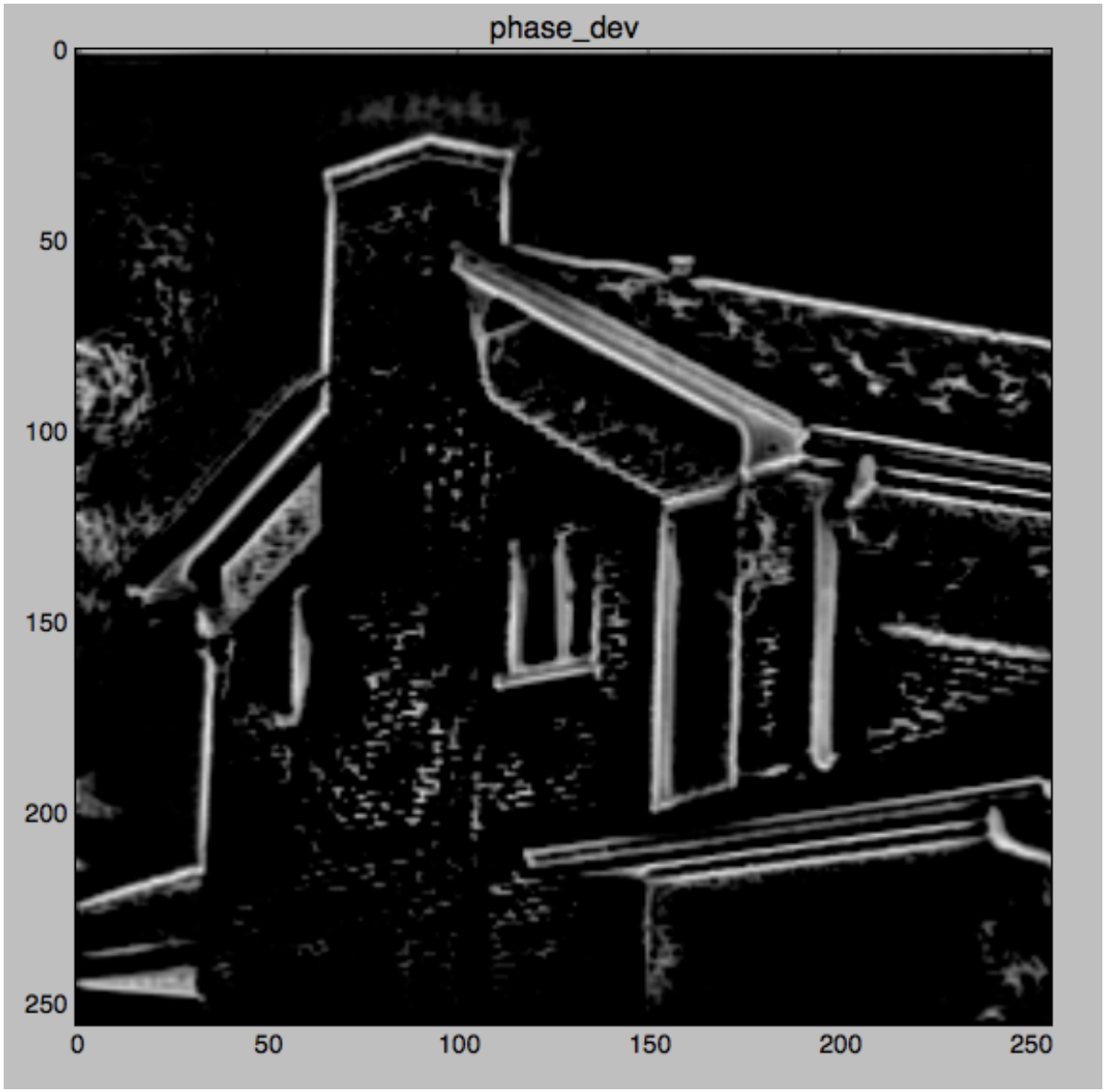
energy

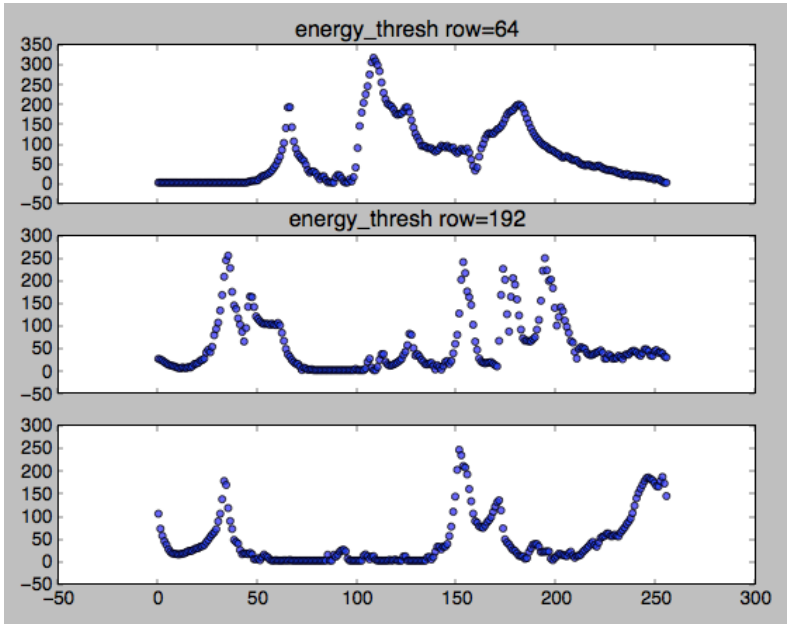




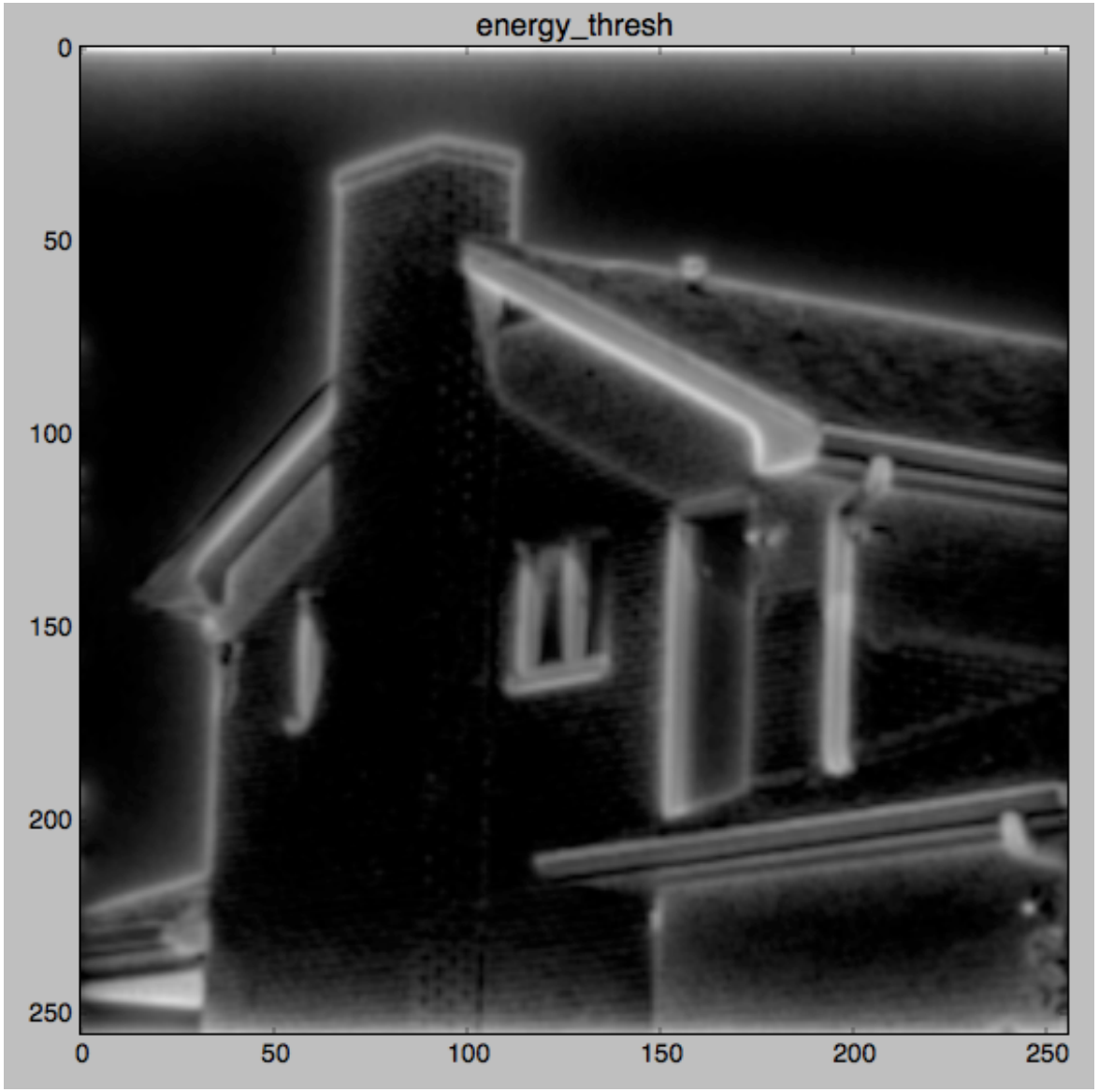


phase_dev

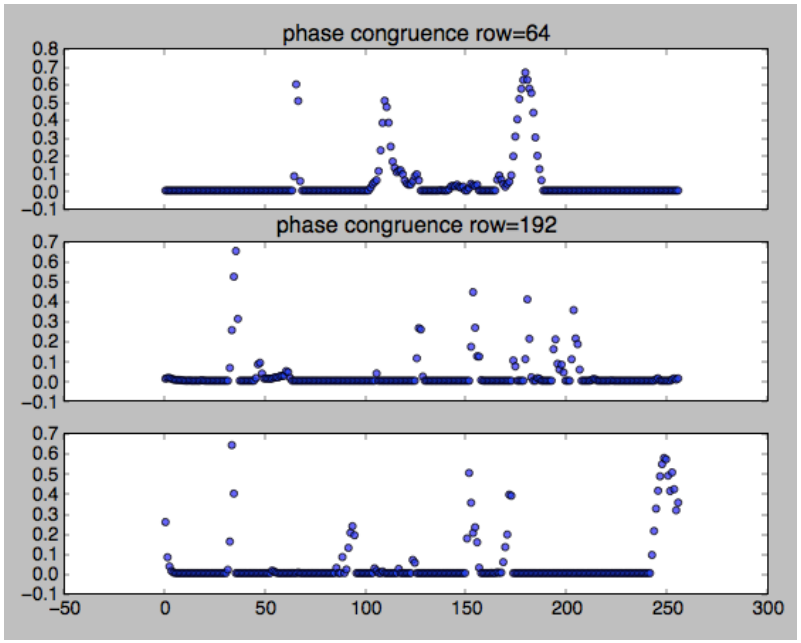


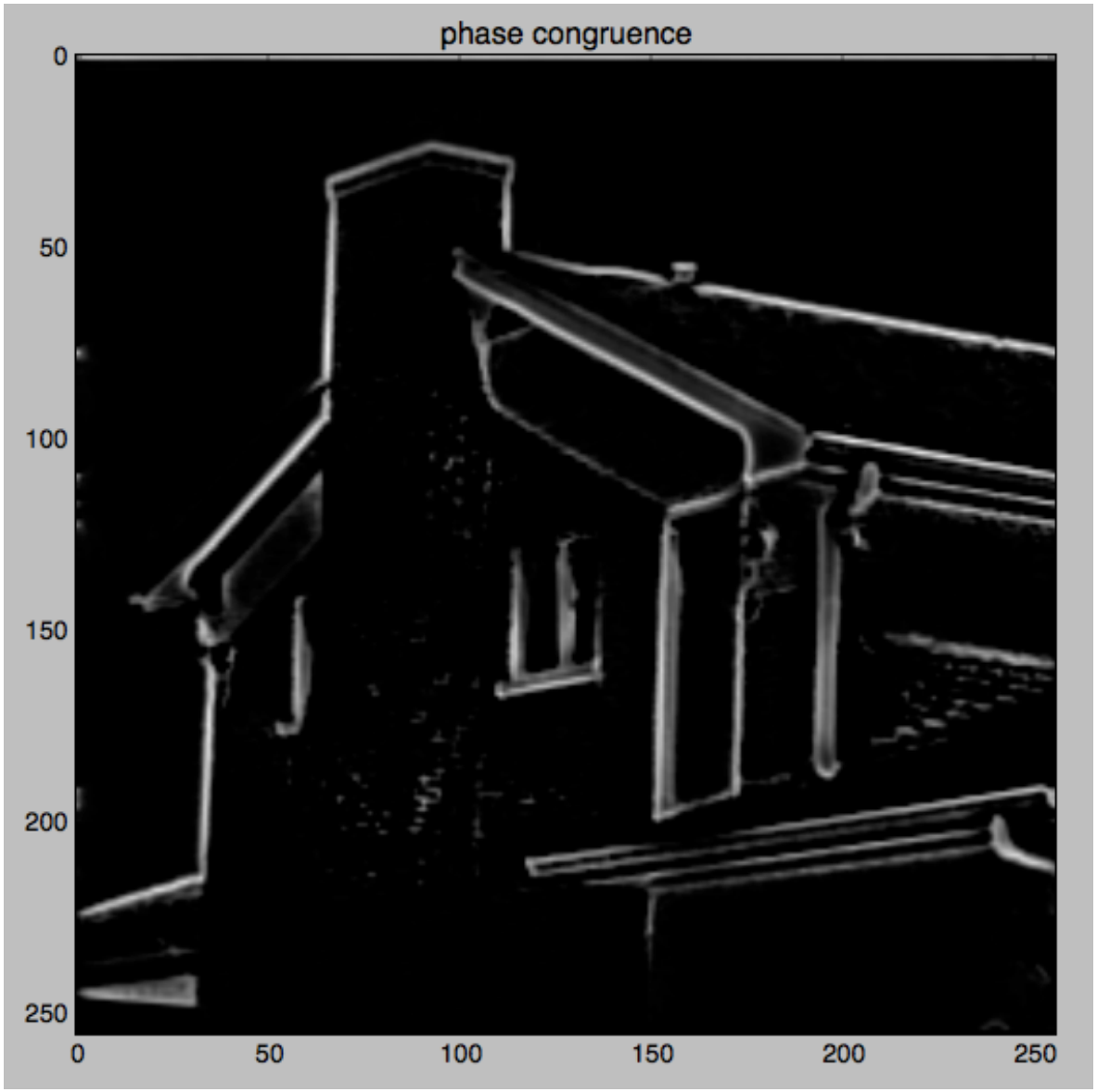


energy_thresh



```
phase congruence =
    maximum(
        1. - deviationGain * arccos(energy / (sumAn + epsilon)),
        0)
```



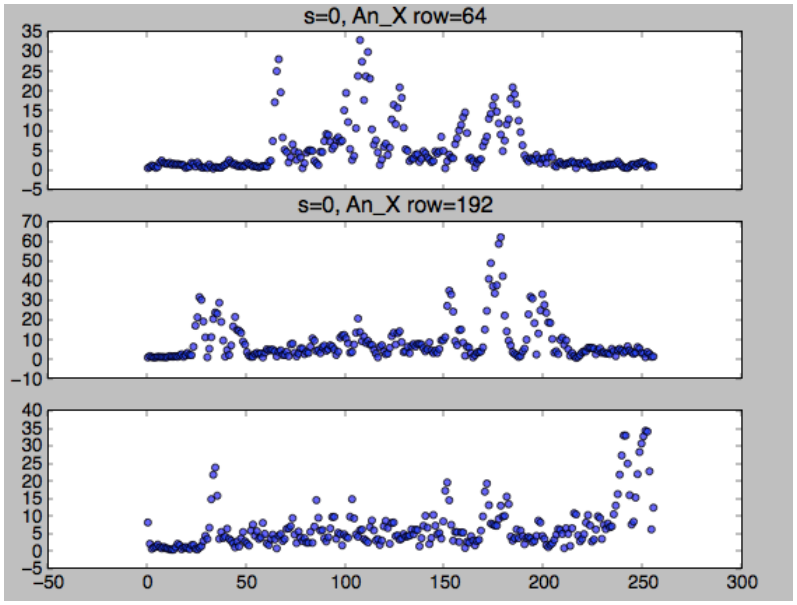


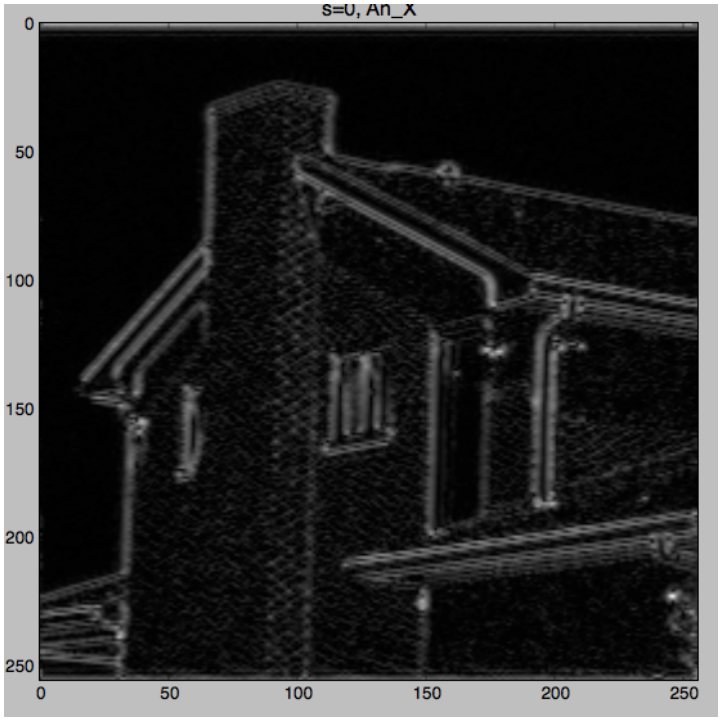
Looking at the x and y components of An

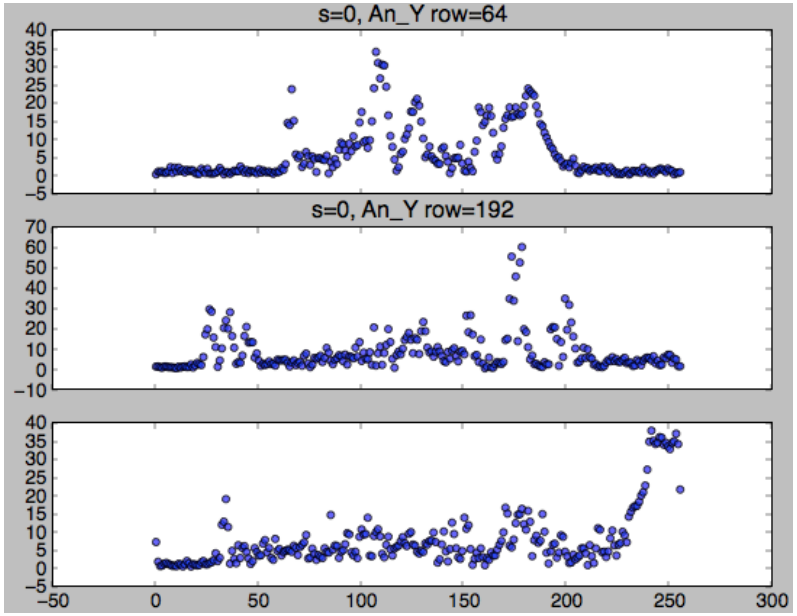
And afterwards, attempting to make a 2nd derive filter.

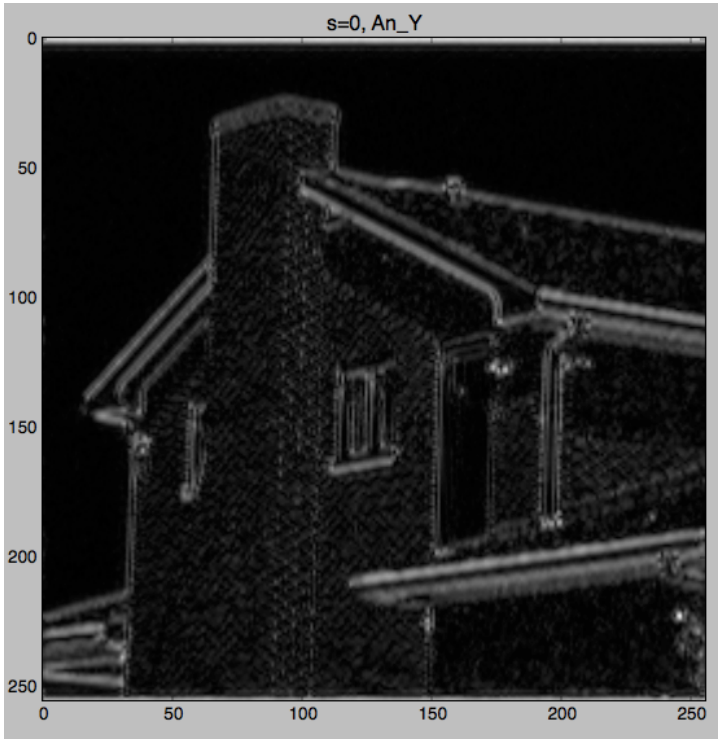
corner curvature uses the first and second derivatives in x and y

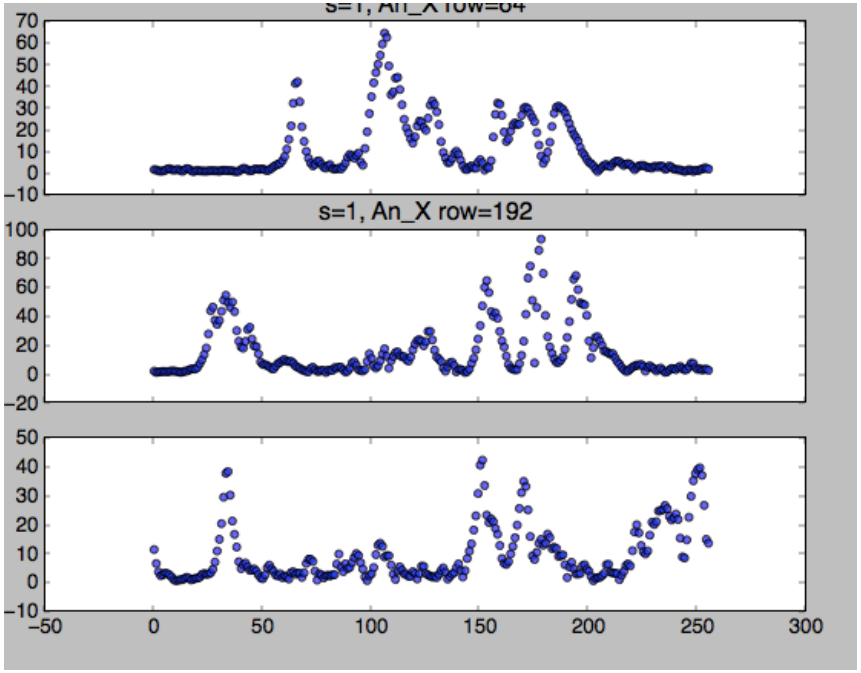
$$k(t,o\sim) = \frac{X\_dot(t,o\sim) * Y\_dot\_dot(t,o\sim) - Y\_dot(t,o\sim) * X\_dot\_dot(t,o\sim)}{(X\_dot^2(t,o\sim) + Y\_dot^2(t,o\sim))^1.5}$$

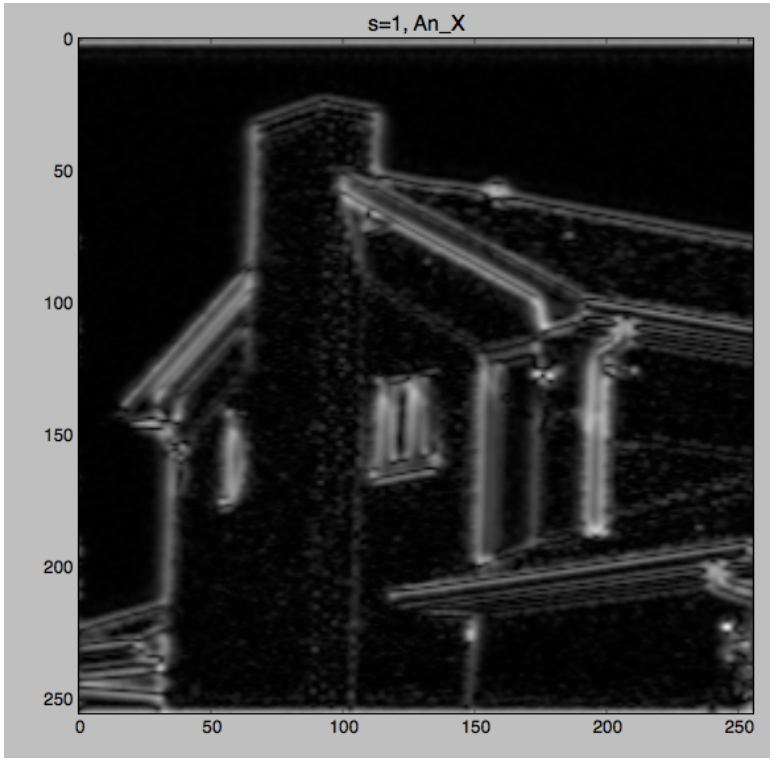

s=0, An_X row=64
s=0, An_X row=192

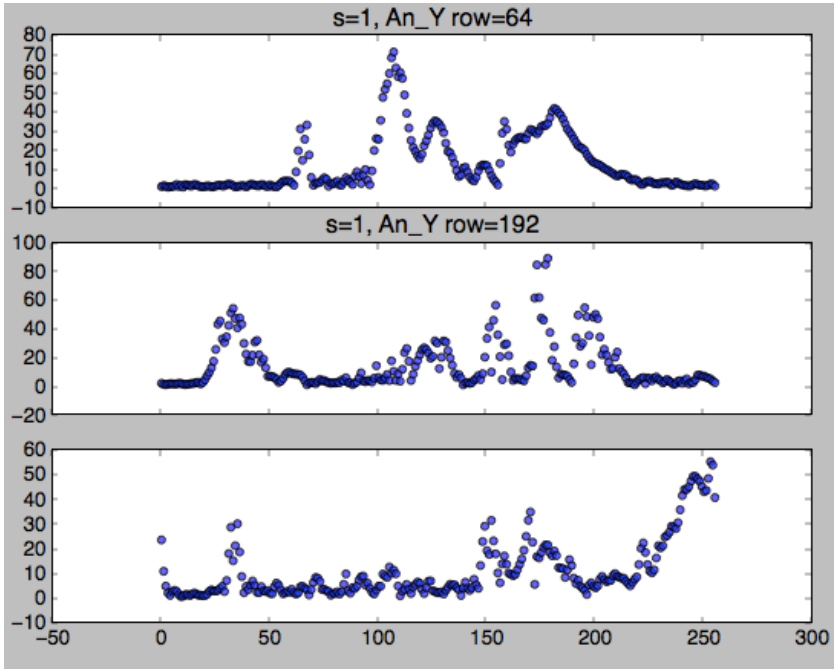

s=0, An_X

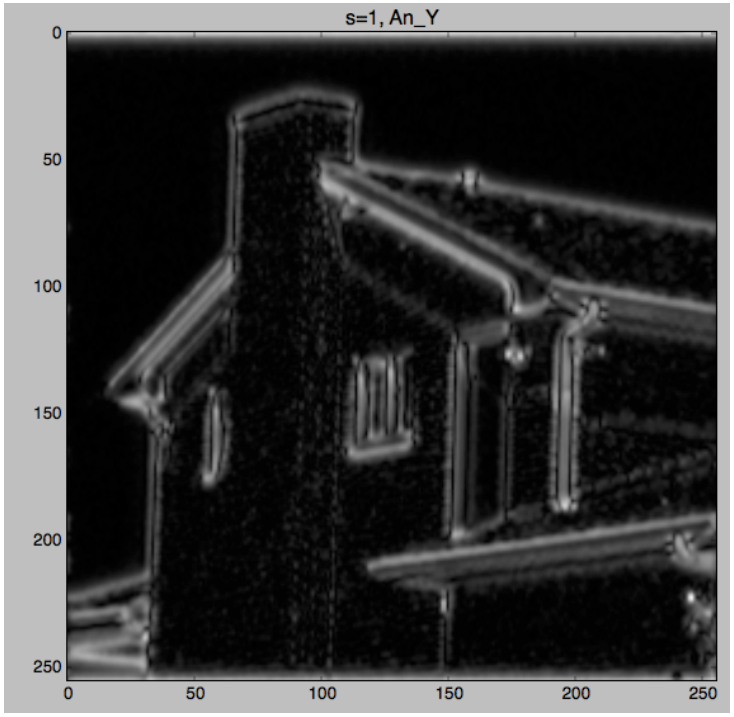

s=0, An_Y row=64
s=0, An_Y row=192

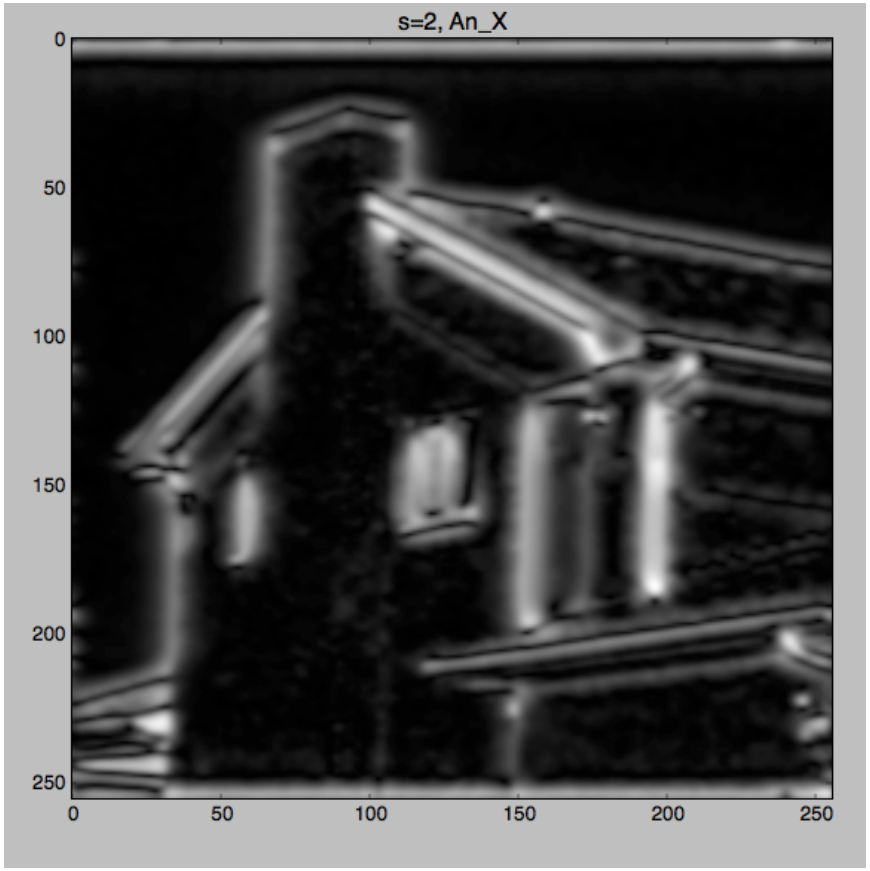

s=0, An_Y

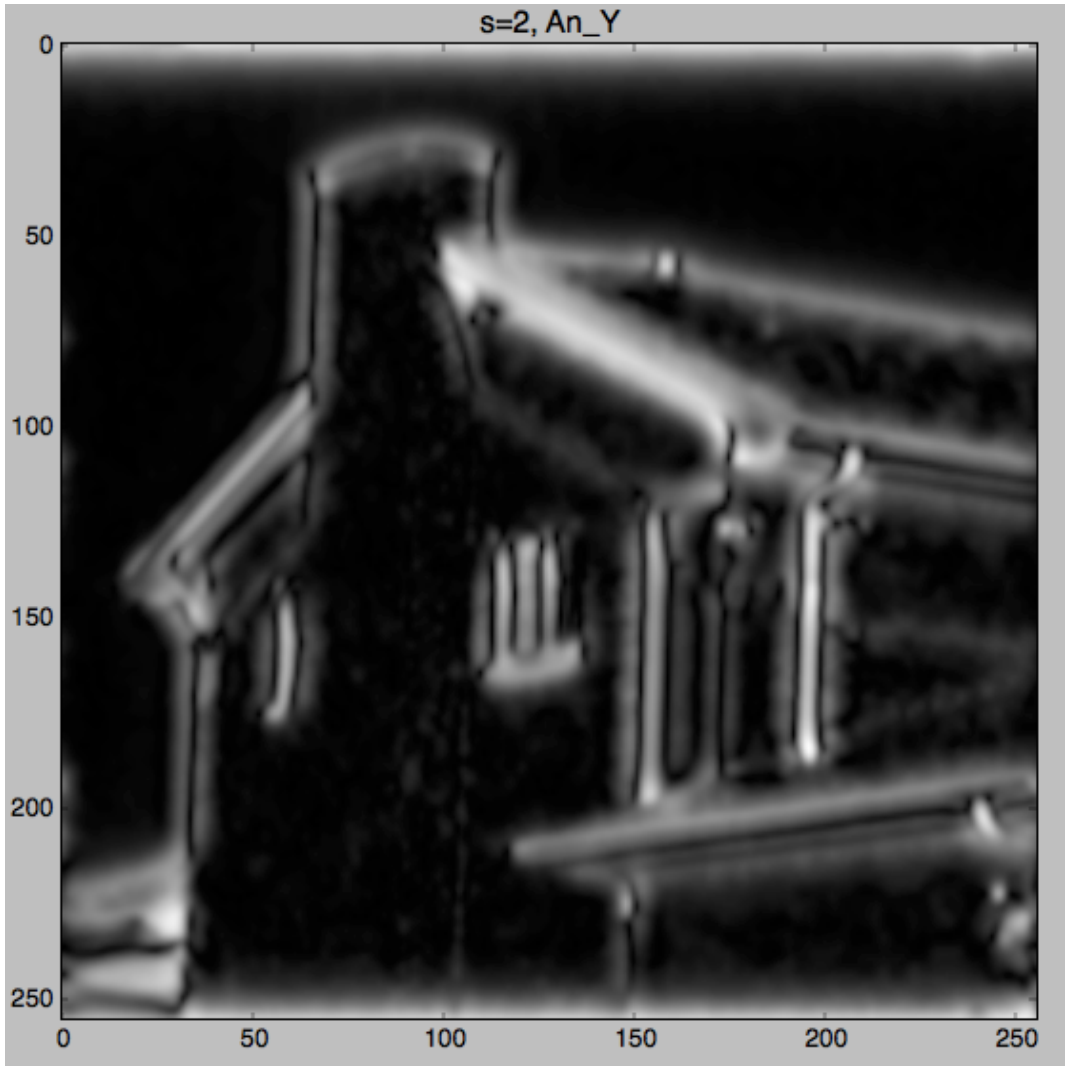

s=1, An_X row=64
s=1, An_X row=192

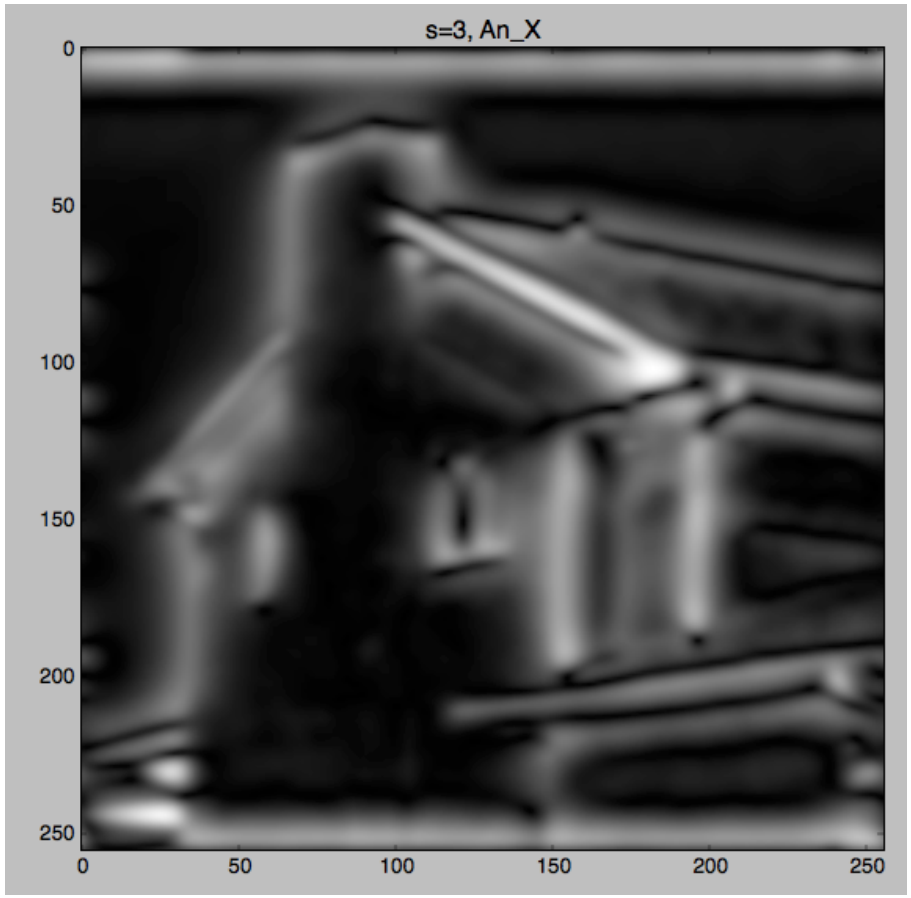

s=1, An_X

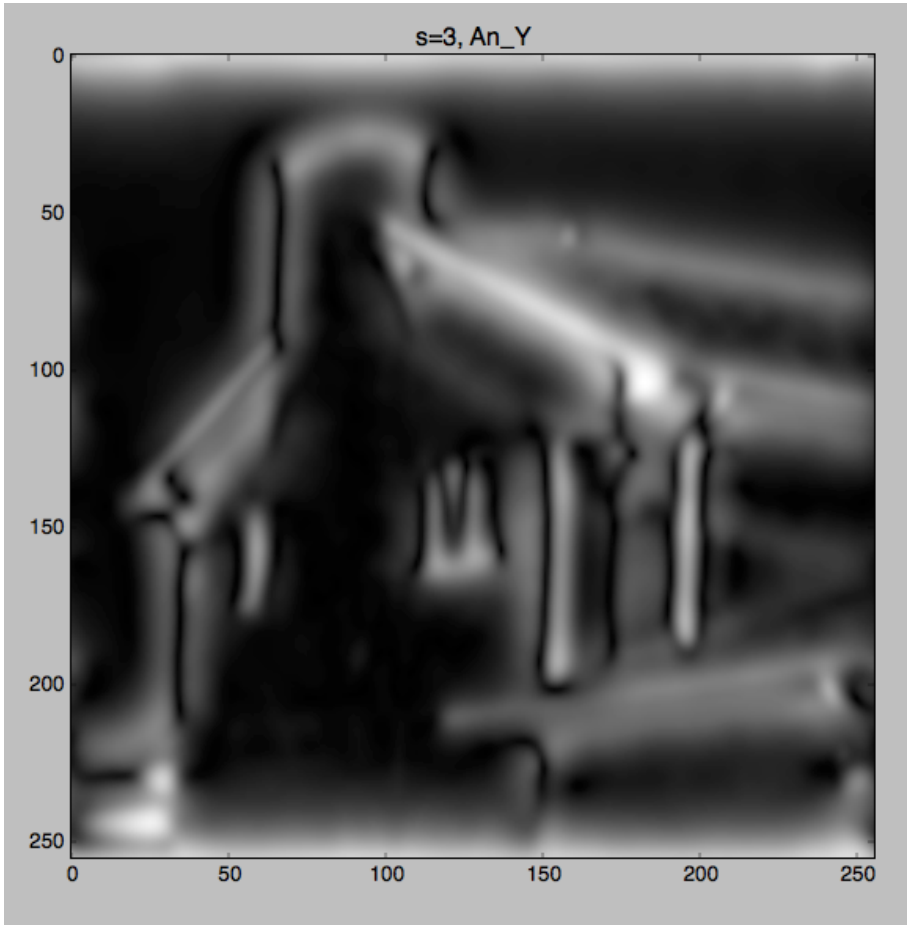

s=1, An_Y row=64
s=1, An_Y row=192

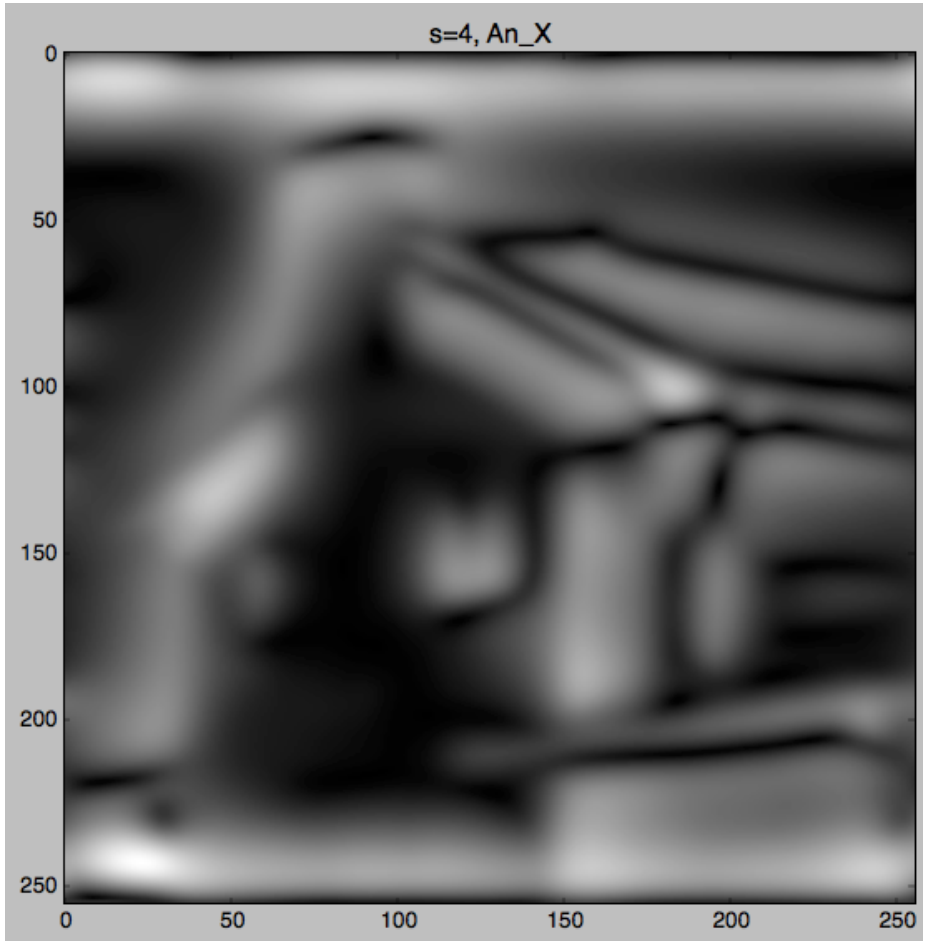

s=1, An_Y

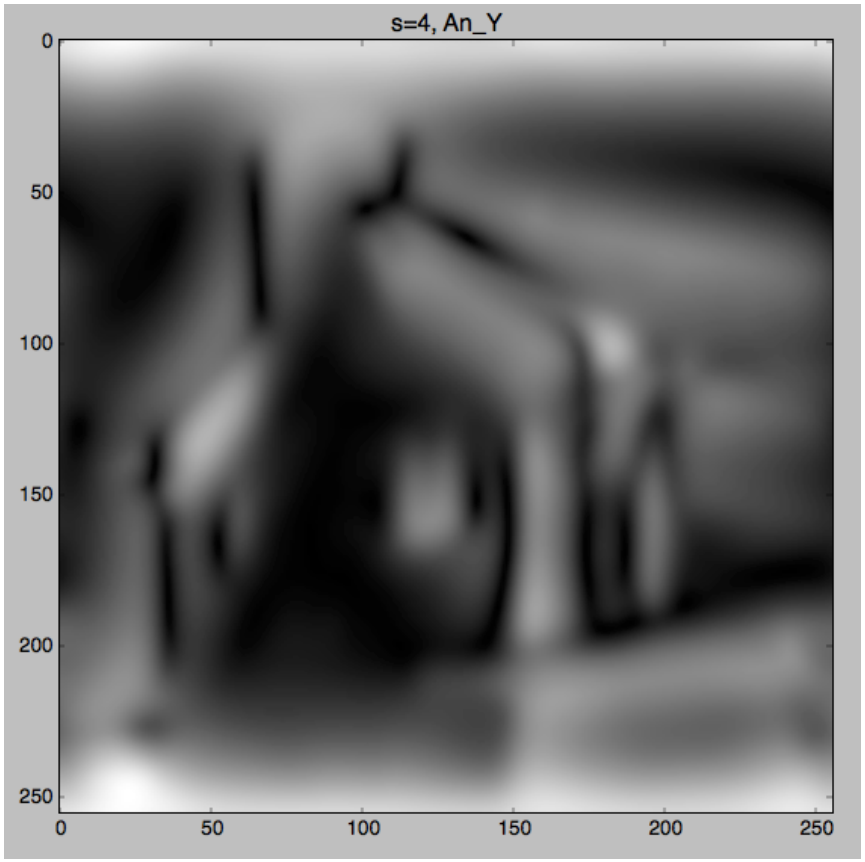

s=2, An_X
0
50
100
150
200
250
0
50
100
150
200
250


s=2, An_Y
0
50
100
150
200
250
0
50
100
150
200
250


s=3, An_X
0
50
100
150
200
250
0
50
100
150
200
250


s=3, An_Y
0
50
100
150
200
250
0
50
100
150
200
250


s=4, An_X
0
50
100
150
200
250
0
50
100
150
200
250


s=4, An_Y
0
50
100
150
200
250
0
50
100
150
200
250