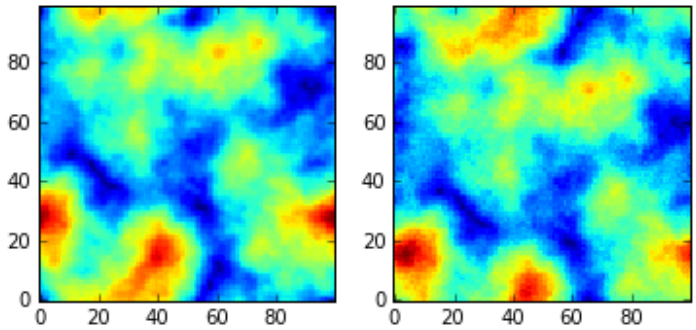
The Cross-Correlation package is available on github: https://github.com/keflavich/image_registration.

The goal is to determine the offset between two images with primarily extended structure.

In [1]:
```
# import statement (with warnings silenced).
with warnings.catch_warnings():
    warnings.filterwarnings("ignore")
    import image_registration
errmsgs = np.seterr(all='ignore') # silence warning messages about div-by-zero
```

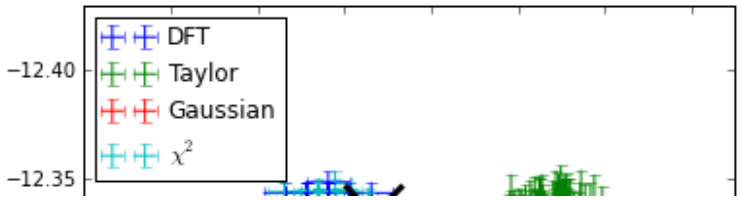
In [8]:
```
# create a simulated image by randomly sampling from a power-law power spectrum with
im1 = image_registration.tests.make_extended(100)
# create an offset version corrupted by noise
im2 = image_registration.tests.make_offset_extended(im1, 4.76666, -12.333333333333333
subplot(121); img1=imshow(im1)
subplot(122); img2=imshow(im2)
```

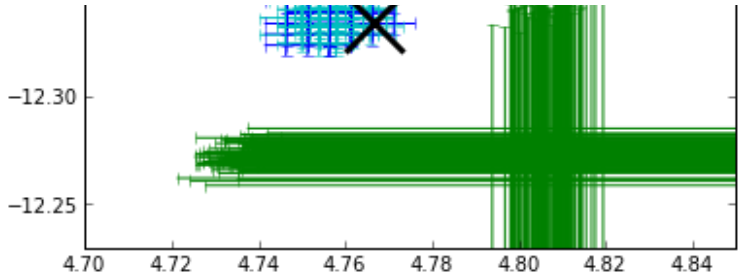


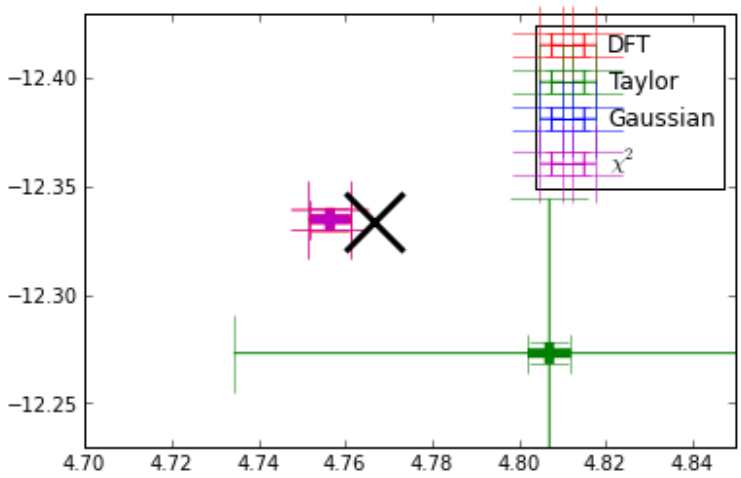
In [3]:
```
# Run the registration methods 100 times each (and hide the output)
offsets_n1,eoffsets_n1 = image_registration.tests.compare_methods(im1,im2,noise=0.1)
```

In [11]:
```
# plot the simulation data
# (note that the "gaussian" approach is hidden; it was problematic)
image_registration.tests.plot_compare_methods(offsets_n1,eoffsets_n1,dx=4.76666666,dy
figure(2); ax=axis([4.7,4.85,-12.23,-12.43])
figure(1); ax=axis([4.7,4.85,-12.23,-12.43])
# the outputs below show the x,y standard deviations (i.e., the "simulated error"),
# the means of the reported errors (i.e., the measured errors)
# and the ratio of the measured error to the simulated error - should be ~1 if correc
# the black X is the correct answer
```

```
Standard Deviations:  [ 0.00471562  0.00529775  0.00503669  0.00506778  0.
0.
  0.00464351  0.00482529]
Error Means:  [ 0.00497512  0.00497512  0.0727985   0.07114768  0.          0.
  0.00490234  0.00476562]
emeans/stds:  [  1.05503078   0.93910169  14.45364199  14.03922113          nan
          nan   1.05574073   0.98763573]
```

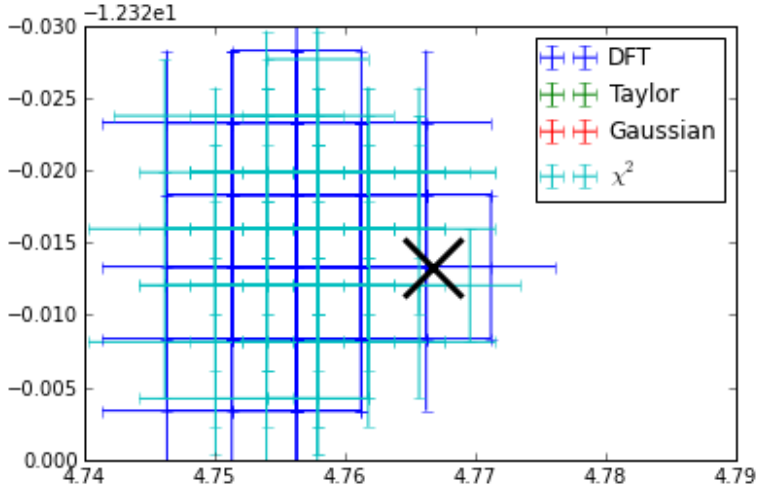


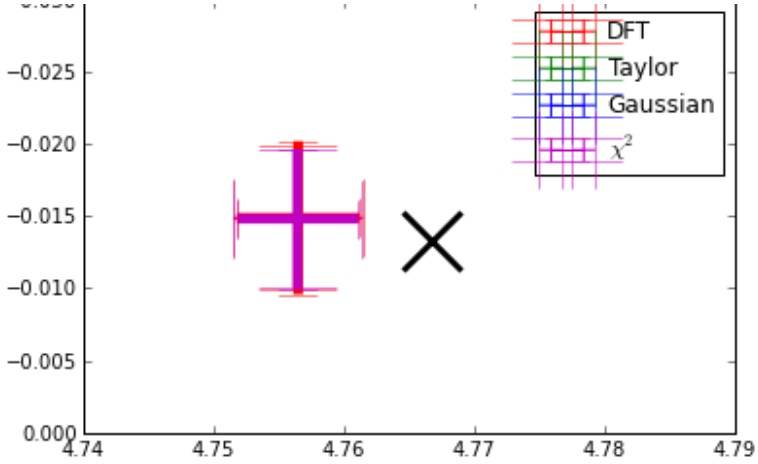




In [10]:
```
# plot the simulation data but zoomed in more (same as above otherwise)
# (note that the "gaussian" approach is hidden; it was problematic)
image_registration.tests.plot_compare_methods(offsets_n1,eoffsets_n1,dx=4.76666666,dy
figure(2); ax=axis([4.74,4.79,-12.32,-12.35])
figure(1); ax=axis([4.74,4.79,-12.32,-12.35])
# the outputs below show the x,y standard deviations (i.e., the "simulated error"),
# the means of the reported errors (i.e., the measured errors)
# and the ratio of the measured error to the simulated error - should be ~1 if correc
# the black X is the correct answer
```

```
Standard Deviations:  [ 0.00471562  0.00529775  0.00503669  0.00506778  0.
0.
  0.00464351  0.00482529]
Error Means:  [ 0.00497512  0.00497512  0.0727985   0.07114768  0.          0.
  0.00490234  0.00476562]
emeans/stds:  [  1.05503078   0.93910169  14.45364199  14.03922113          nan
          nan   1.05574073   0.98763573]
```




−0.025
−0.020
−0.015
−0.010
−0.005
0.000
4.74
4.75
4.76
4.77
4.78
4.79
DFT
Taylor
Gaussian
χ^2

In [5]: