Subject: Scale the psf on images.
Posted by anes.tziamtzis on Mon, 12 Jan 2015 08:15:20 GMT
View Forum Message <> Reply to Message

Hi

I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.

Which technique you think is the best? I tried with the convolve script but the result is not good.

imconv = convolve(image1, image2, FT_PSF = psf)

image 1 the science frame and image is the psf frame that I created through iraf.

Any suggestions are appreciated

Thanks a lot.

Subject: Re: Scale the psf on images.
Posted by rryan%stsci.edu on Mon, 12 Jan 2015 09:17:21 GMT
View Forum Message <> Reply to Message

I think you might be missing something... Or maybe I am. But let's run down the procedure to make sure we're clear with each other.

(1) take first image of sky.
(2) compute PSF from the first image.

(3) take second image of sky
(4) compute PSF from the second image

Now you want to subtract the second image from the first to look for something that changed? But you want to correct for the PSF? Is that what you're doing? If so, then I think you've misunderstood a few things. What you need to do is compute the *transformation kernel* between the first and second image. In principle that kernel is the ratio of the FFTs of the PSFs. But in practice that is very noisy for a host of reasons, and I'm fairly sure is not what is normally done (at least not with empirical PSFs). But for a quick and dirty thing, this is probably the best you're going to get... So in pseudo code you're looking at something like this:

im1 = image 1 of the target
psf1 = psf of image 1
im2 = image 2 of the target
psf2 = psf of image 2

Now compute the kernel.

p1 = Fourier transform of PSF1
p2= Fourier transform of psf2

kernel= inverse fourier transform of (p1/p2)

im2_prime = convolve(img2, kernel)

diff = im1 - img2_prime

now inspect the diff image. YOu can short cut a few things by not using convolve and working with the FFTs yourself, but either way there are all sorts of gotchas along this path, so be careful.

Good luck
Russell

On Monday, January 12, 2015 at 9:15:23 AM UTC+1, anes.tz...@gmail.com wrote:
> Hi
>
> I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.
>
> Which technique you think is the best? I tried with the convolve script but the result is not good.
>
> imconv = convolve(image1, image2, FT_PSF = psf)
>
> image 1 the science frame and image is the psf frame that I created through iraf.
>
> Any suggestions are appreciated
>
> Thanks a lot.

Subject: Re: Scale the psf on images.
Posted by on Mon, 12 Jan 2015 10:42:27 GMT
View Forum Message <> Reply to Message

Den måndag 12 januari 2015 kl. 09:15:23 UTC+1 skrev anes.tz...@gmail.com:
> Hi
>

> I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.
>
> Which technique you think is the best? I tried with the convolve script but the result is not good.
>
> imconv = convolve(image1, image2, FT_PSF = psf)
>
> image 1 the science frame and image is the psf frame that I created through iraf.
>
> Any suggestions are appreciated
>
> Thanks a lot.

A dirty trick you could try:

Compute the Fourier transform of both images, calculate phases and amplitudes from the real and imaginary parts of the transforms. Then substitute the amplitude in one of the images with that from the other. For the one you changed: go back to real and imaginary form, then compute the inverse transform.

The reason this might work is that the Fourier amplitude has most of the PSF information, while the phase has most of the information about the scene.

Subject: Re: Scale the psf on images.
Posted by anes.tziamtzis on Mon, 12 Jan 2015 12:38:26 GMT

View Forum Message <> Reply to Message

On Monday, January 12, 2015 at 5:17:23 PM UTC+8, rryan%s...@gtempaccount.com wrote:
> I think you might be missing something... Or maybe I am. But let's run down the procedure to make sure we're clear with each other.
>
> (1) take first image of sky.
> (2) compute PSF from the first image.
>
> (3) take second image of sky
> (4) compute PSF from the second image
>
> Now you want to subtract the second image from the first to look for something that changed? But you want to correct for the PSF? Is that what you're doing? If so, then I think you've misunderstood a few things. What you need to do is compute the *transformation kernel* between the first and second image. In principle that kernel is the ratio of the FFTs of the PSFs. But in practice that is very noisy for a host of reasons, and I'm fairly sure is not what is normally done (at least not with empirical PSFs). But for a quick and dirty thing, this is probably the best you're going to get... So in pseudo code you're looking at something like this:

>
> im1 = image 1 of the target
> psf1 = psf of image 1
> im2 = image 2 of the target
> psf2 = psf of image 2
>
> Now compute the kernel.
>
> p1 = Fourier transform of PSF1
> p2= Fourier transform of psf2
>
> kernel= inverse fourier transform of (p1/p2)
>
> im2_prime = convolve(img2, kernel)
>
> diff = im1 - img2_prime
>
> now inspect the diff image. YOu can short cut a few things by not using convolve and working with the FFTs yourself, but either way there are all sorts of gotchas along this path, so be careful.
>
> Good luck
> Russell
>
>
>
>
> On Monday, January 12, 2015 at 9:15:23 AM UTC+1, anes.tz...@gmail.com wrote:
>> Hi
>>
>> I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.
>>
>> Which technique you think is the best? I tried with the convolve script but the result is not good.
>>
>> imconv = convolve(image1, image2, FT_PSF = psf)
>>
>> image 1 the science frame and image is the psf frame that I created through iraf.
>>
>> Any suggestions are appreciated
>>
>> Thanks a lot.

Hi Russell

Well this is what I want to do. Do you know if there is a more efficient/better way for doing such a thing? I cannot think of something else.

I would like to thank you for your time.

Subject: Re: Scale the psf on images.
Posted by rryan%stsci.edu on Mon, 12 Jan 2015 12:57:32 GMT
View Forum Message <> Reply to Message

Yes and no... You've stumbled onto a fairly tough problem and if my simple thing from before doesn't work, then you're unlikely to find anything that you'll get in place for this upcoming run. Again, the simple thing I outlined is mathematically correct and is exactly what you're after in the case of continuous functions without noise (and probably some other conditions as well). Well, you don't have that, so don't expect much success (though it does work in certain cases, like high signal-to-noise, well-sampled PSFs).

If you've got a lot of time and/or are very worried about this, you can look into the code which forms the basis for many ground-based pipelines (I think LSST uses a variant of this code):

http://www.astro.washington.edu/users/becker/v2.0/hotpants.h tml

But every time I try this, I run into problems. So, good luck --- you'll need it. Another trick you can exploit, is to remember the definition of these functions and not work in fourier space, but work in image space. I've never tried anything like Mats' suggestion, but give it a go...

But, just out of curiosity... how different are the PSFs? Can you represent them by analytical functions (such as Gaussian or Moffat)?

R

On Monday, January 12, 2015 at 9:15:23 AM UTC+1, anes.tz...@gmail.com wrote:
> Hi
>
> I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.
>
> Which technique you think is the best? I tried with the convolve script but the result is not good.
>
> imconv = convolve(image1, image2, FT_PSF = psf)
>
> image 1 the science frame and image is the psf frame that I created through iraf.
>

> Any suggestions are appreciated
>
> Thanks a lot.

Subject: Re: Scale the psf on images.
Posted by bstecklu on Tue, 13 Jan 2015 08:59:22 GMT
View Forum Message <> Reply to Message

On 2015-01-12 09:15, anes.tziamtzis@gmail.com wrote:
> Hi
>
> I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.
>
> Which technique you think is the best? I tried with the convolve script but the result is not good.
>
> imconv = convolve(image1, image2, FT_PSF = psf)
>
> image 1 the science frame and image is the psf frame that I created through iraf.
>
> Any suggestions are appreciated
>
> Thanks a lot.
>

There are a couple of packages around for difference imaging or difference image
analysis (DIA), e.g. ISIS http://www2.iap.fr/users/alard/package.html. When it
comes to IDL you might want to have a look at Dan Bramich's page
http://www.danidl.co.uk/danidl.features.shtml which provides both papers and
code (DanIDL v1.1) for this purpose.

Subject: Re: Scale the psf on images.
Posted by anes.tziamtzis on Wed, 14 Jan 2015 06:50:17 GMT
View Forum Message <> Reply to Message

Hi Russel,

Thanks a lot for your tip. I need to ask sometihng extra. I used the following:
fits_read, 'im1.fits',image1,header1
fits_read, 'im2.fits',image2,header2
fits_read, 'im1.fits.psf.1.fits',image3,header3
fits_read, 'im2.fits.psf.1.fits',image4,header4

```
;Compute the difference kernel
psf1=fft(image3)
psf2=fft(image4)

kernel = REAL_PART(FFT(psf1/psf2, /INVERSE))

image2_prime = convolve(image2, kernel)
diff = image1 - image2_prime

WRITEFITS,'im_conv.fits',diff,header1


end
```

No stars are seen in the resulting image, but the background has crazy values. The range is from -9e-8 to 9e8. What i have done wrong here?

Subject: Re: Scale the psf on images.
Posted by on Wed, 14 Jan 2015 08:43:37 GMT

View Forum Message <> Reply to Message

Den onsdag 14 januari 2015 kl. 07:50:20 UTC+1 skrev anes.tz...@gmail.com:

```
> Hi Russel,
>
> Thanks a lot for your tip. I need to ask sometihng extra. I used the following:
> fits_read, 'im1.fits',image1,header1
> fits_read, 'im2.fits',image2,header2
> fits_read, 'im1.fits.psf.1.fits',image3,header3
> fits_read, 'im2.fits.psf.1.fits',image4,header4
>
> ;Compute the difference kernel
> psf1=fft(image3)
> psf2=fft(image4)
>
> kernel = REAL_PART(FFT(psf1/psf2, /INVERSE))
>
> image2_prime = convolve(image2, kernel)
> diff = image1 - image2_prime
>
> WRITEFITS,'im_conv.fits',diff,header1
>
>
> end
>
> No stars are seen in the resulting image, but the background has crazy values. The range is
from -9e-8 to 9e8. What i have done wrong here?
```

Have a look at image2_prime, most likely it is dominated by amplified noise. If that is the case, you need to include a noise filter in your convolution operation.

You could inspect the Fourier transform of the kernel, psf1/psf2 in your notation (although I would have used otf1 and otf2 for the Fourier transforms of the psfs, otf = optical transfer function).

Look for spatial frequencies where the absolute values are > 1 (assuming both otfs are normalized to unity at the origin). Those are frequencies where you amplify image2. If image2 is dominated by noise at those frequencies (as is usually the case at high spatial frequencies), then the resulting image will be dominated by that. Zeroing the otf ratio in these frequencies in principle deals with this but may cause other problems, so you want a soft transition between zeroed frequencies and frequencies that are not zeroed.

How to deal with this is not an IDL question, it is more of a maths issue. Read up on low pass filters, in particular Wiener filters, in some image processing text book.

Subject: Re: Scale the psf on images.
Posted by anes.tziamtzis on Thu, 15 Jan 2015 07:13:16 GMT

View Forum Message <> Reply to Message

Well perhaps i will get the award for the dumbest person in the world for this.

On the following link there is an fft example
http://www.exelisvis.com/docs/fftreducebackgroundnoise.html

As you said Mats they do something to reduce the noise. The problem is by using this method, again the final image doesnt make sense. There is no structure actually and the pixel values vary from -1e7 to 1e7. This is the code

```
fits_read, 'Image1.fits',image1,header1
fits_read, 'Image1.fits.psf.1.fits',image2,header2
fits_read, 'Image2.fits',image3,header3
fits_read, 'Image2.fits.psf.1.fits',image4,header4


;Compute the difference kernel
p1=fft(image2)
p2=fft(image4)

; Compute the power spectrum of the transform and apply a log scale.

powerSpectrum = ABS(p1)^2
scaledPowerSpect = ALOG10(powerSpectrum)

powerSpectrum2 = ABS(p2)^2
scaledPowerSpect2 = ALOG10(powerSpectrum2)
```

```
; Scale the power spectrum to make its maximum value equal to 0.
scaledPS = scaledPowerSpect - MAX(scaledPowerSpect)
scaledPS2 = scaledPowerSpect2 - MAX(scaledPowerSpect2)

surface,powerSpectrum2

mask = REAL_PART(scaledPowerSpect) GT -5
psf1_trans = p1*mask
mask2 = REAL_PART(scaledPowerSpect2) GT -5
psf2_trans = p2*mask2
kernel = REAL_PART(FFT(psf1_trans/psf2_trans, /INVERSE, /CENTER))


image3_prime = convol(image3, kernel)
diff = image1 - image3_prime


WRITEFITS,'im_conv.fits',diff,header3
end
```

Subject: Re: Scale the psf on images.
Posted by on Thu, 15 Jan 2015 07:58:15 GMT

View Forum Message <> Reply to Message

Den torsdag 15 januari 2015 kl. 08:13:19 UTC+1 skrev anes.tz...@gmail.com:
> Well perhaps i will get the award for the dumbest person in the world for this.
>
> On the following link there is an fft example
> http://www.exelisvis.com/docs/fftreducebackgroundnoise.html
>
> As you said Mats they do something to reduce the noise. The problem is by using this method, again the final image doesnt make sense. There is no structure actually and the pixel values vary from -1e7 to 1e7. This is the code
>
> fits_read, 'Image1.fits',image1,header1
> fits_read, 'Image1.fits.psf.1.fits',image2,header2
> fits_read, 'Image2.fits',image3,header3
> fits_read, 'Image2.fits.psf.1.fits',image4,header4
>
>
> ;Compute the difference kernel
> p1=fft(image2)
> p2=fft(image4)
>
> ; Compute the power spectrum of the transform and apply a log scale.
>

```
>  powerSpectrum = ABS(p1)^2
>  scaledPowerSpect = ALOG10(powerSpectrum)
>
>  powerSpectrum2 = ABS(p2)^2
>  scaledPowerSpect2 = ALOG10(powerSpectrum2)
>
>
>  ; Scale the power spectrum to make its maximum value equal to 0.
>  scaledPS = scaledPowerSpect - MAX(scaledPowerSpect)
>  scaledPS2 = scaledPowerSpect2 - MAX(scaledPowerSpect2)
>
>  surface,powerSpectrum2
>
>  mask = REAL_PART(scaledPowerSpect) GT -5
>  psf1_trans = p1*mask
>  mask2 = REAL_PART(scaledPowerSpect2) GT -5
>  psf2_trans = p2*mask2
>  kernel = REAL_PART(FFT(psf1_trans/psf2_trans, /INVERSE, /CENTER))
>
>
>  image3_prime = convol(image3, kernel)
>  diff = image1 - image3_prime
>
>
>  WRITEFITS,'im_conv.fits',diff,header3
>  end
```

You are aware that you are trying to construct noise filters based on the power spectra of the PSFs?

Subject: Re: Scale the psf on images.
Posted by anes.tziamtzis on Thu, 15 Jan 2015 08:04:16 GMT

View Forum Message <> Reply to Message

On Thursday, January 15, 2015 at 3:58:18 PM UTC+8, Mats Löfdahl wrote:
> Den torsdag 15 januari 2015 kl. 08:13:19 UTC+1 skrev anes.tz...@gmail.com:
>> Well perhaps i will get the award for the dumbest person in the world for this.
>>
>> On the following link there is an fft example
>> http://www.exelisvis.com/docs/fftreducebackgroundnoise.html
>>
>> As you said Mats they do something to reduce the noise. The problem is by using this method, again the final image doesnt make sense. There is no structure actually and the pixel values vary from -1e7 to 1e7. This is the code
>>
>> fits_read, 'Image1.fits',image1,header1
>> fits_read, 'Image1.fits.psf.1.fits',image2,header2

```
>>  fits_read, 'Image2.fits',image3,header3
>>  fits_read, 'Image2.fits.psf.1.fits',image4,header4
>>
>>
>>  ;Compute the difference kernel
>>  p1=fft(image2)
>>  p2=fft(image4)
>>
>>  ; Compute the power spectrum of the transform and apply a log scale.
>>
>>  powerSpectrum = ABS(p1)^2
>>  scaledPowerSpect = ALOG10(powerSpectrum)
>>
>>  powerSpectrum2 = ABS(p2)^2
>>  scaledPowerSpect2 = ALOG10(powerSpectrum2)
>>
>>
>>  ; Scale the power spectrum to make its maximum value equal to 0.
>>  scaledPS = scaledPowerSpect - MAX(scaledPowerSpect)
>>  scaledPS2 = scaledPowerSpect2 - MAX(scaledPowerSpect2)
>>
>>  surface,powerSpectrum2
>>
>>  mask = REAL_PART(scaledPowerSpect) GT -5
>>  psf1_trans = p1*mask
>>  mask2 = REAL_PART(scaledPowerSpect2) GT -5
>>  psf2_trans = p2*mask2
>>  kernel = REAL_PART(FFT(psf1_trans/psf2_trans, /INVERSE, /CENTER))
>>
>>
>>  image3_prime = convol(image3, kernel)
>>  diff = image1 - image3_prime
>>
>>
>>  WRITEFITS,'im_conv.fits',diff,header3
>>  end
>
```

> You are aware that you are trying to construct noise filters based on the power spectra of the PSFs?

To be honest I am totally lost with it. This is the first time I am doing something like this, so actually I have no clue on what each command does. A bit more light is appreciated. Thanks a lot for your time.

Subject: Re: Scale the psf on images.
Posted by   on Thu, 15 Jan 2015 10:19:59 GMT

Den torsdag 15 januari 2015 kl. 09:04:18 UTC+1 skrev anes.tz...@gmail.com:
> On Thursday, January 15, 2015 at 3:58:18 PM UTC+8, Mats Löfdahl wrote:
>> Den torsdag 15 januari 2015 kl. 08:13:19 UTC+1 skrev anes.tz...@gmail.com:
>>> Well perhaps i will get the award for the dumbest person in the world for this.
>>>
>>> On the following link there is an fft example
>>> http://www.exelisvis.com/docs/fftreducebackgroundnoise.html
>>>
>>> As you said Mats they do something to reduce the noise. The problem is by using this method, again the final image doesnt make sense. There is no structure actually and the pixel values vary from -1e7 to 1e7. This is the code
>>>
>>> fits_read, 'Image1.fits',image1,header1
>>> fits_read, 'Image1.fits.psf.1.fits',image2,header2
>>> fits_read, 'Image2.fits',image3,header3
>>> fits_read, 'Image2.fits.psf.1.fits',image4,header4
>>>
>>>
>>> ;Compute the difference kernel
>>> p1=fft(image2)
>>> p2=fft(image4)
>>>
>>> ; Compute the power spectrum of the transform and apply a log scale.
>>>
>>> powerSpectrum = ABS(p1)^2
>>> scaledPowerSpect = ALOG10(powerSpectrum)
>>>
>>> powerSpectrum2 = ABS(p2)^2
>>> scaledPowerSpect2 = ALOG10(powerSpectrum2)
>>>
>>>
>>> ; Scale the power spectrum to make its maximum value equal to 0.
>>> scaledPS = scaledPowerSpect - MAX(scaledPowerSpect)
>>> scaledPS2 = scaledPowerSpect2 - MAX(scaledPowerSpect2)
>>>
>>> surface,powerSpectrum2
>>>
>>> mask = REAL_PART(scaledPowerSpect) GT -5
>>> psf1_trans = p1*mask
>>> mask2 = REAL_PART(scaledPowerSpect2) GT -5
>>> psf2_trans = p2*mask2
>>> kernel = REAL_PART(FFT(psf1_trans/psf2_trans, /INVERSE, /CENTER))
>>>
>>>
>>> image3_prime = convol(image3, kernel)
>>> diff = image1 - image3_prime
>>>

>>>
>>> WRITEFITS,'im_conv.fits',diff,header3
>>> end
>>
>> You are aware that you are trying to construct noise filters based on the power spectra of the PSFs?
>
> To be honest I am totally lost with it. This is the first time I am doing something like this, so actually I have no clue on what each command does. A bit more light is appreciated. Thanks a lot for your time.

I think you need to think of this as a mathematics problem and not only as a coding problem. If you don't understand what the commands do and what to expect from the data, there is no way you are going to be able to adapt code snippets you find to your own problem.

For example, the code you found at exelisvis only does filtering but there are no PSFs or kernels involved so you can't apply it blindly to your problem.

A maths thing you missed: The misnamed variables psf1_trans and psf2_trans (they are not psfs, they are the Fourier transforms of psfs, i.e., otfs) have zeroes where mask1 and mask2 have zeroes, so when you do psf1_trans/psf2_trans you divide by zero where mask2 has zeroes.

So this is not what you want to do, you want a single low-pass filter (or mask, as you name it) that you use to get rid of data in frequencies where you would otherwise amplify noise rather than signal in (the Fourier transform of) the image you are modifying, image2 in this case (going by the file name, you variable is called image3). So base your filter on where image2 is noise dominated.

Understanding what quantities you are dealing with and naming the variables properly and consistently might help some. You could start your program like this:

```
; Read image 1 and its psf
fits_read, 'Image1.fits', image1, image_header1
fits_read, 'Image1.fits.psf.1.fits',psf1, psf_header1

; Read image 2 and its psf
fits_read, 'Image2.fits', image2, image_header2
fits_read, 'Image2.fits.psf.1.fits',psf2, psf_header2

;Compute the optical transfer functions
otf1 = fft(psf1)
otf2 = fft(psf2)
```

Just so you don't confuse yourself with what is what later in the code.

And then do read up on image processing, what to expect the Fourier transform of an image to look like (where to find signal and where to find noise), and how to construct a proper filter. Make sure you understand Fourier transforms enough that your convolution operation, including the low-pass filtering, can be expressed as multiplication and division in the Fourier domain. Saves you the call to convol() and makes it easier to see where you are dividing by zero or almost zero.

Or try the dirty trick I suggested in my first reply. The good thing about it is that since you don't do any convolutions that might amplify noise, you don't have to bother with noise filters. The assumptions behind it is that the otfs are mainly real-valued and that the images are similar enough that the differences you are looking for do not change the power spectrum significantly.

And you need the two images to be already registered and have the same spatial image scale, of course. But that goes for what you are trying to do now as well.

Subject: Re: Scale the psf on images.
Posted by anes.tziamtzis on Thu, 15 Jan 2015 13:33:33 GMT

View Forum Message <> Reply to Message

On Monday, January 12, 2015 at 4:15:23 PM UTC+8, anes.tz...@gmail.com wrote:
> Hi
>
> I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.
>
> Which technique you think is the best? I tried with the convolve script but the result is not good.
>
> imconv = convolve(image1, image2, FT_PSF = psf)
>
> image 1 the science frame and image is the psf frame that I created through iraf.
>
> Any suggestions are appreciated
>
> Thanks a lot.

Well the observations are on Fermi sources at optical bands. We want to take a series of frames for every source and check if there is variability. The variation of the psf will be hopefully small, but since the expected variation is low as well (0.1-0.2) magnitudes, I want the best possible accuracy on the image subtraction.

In terms of amplitudes and phases I found a simple example that deals with a function.

```
 t=findgen(1024)*10*!pi/1024
 S=t/(2*!pi)-fix(t/(2*!pi)) ; sawtooth function
 plot,S & wait,1
 c=fft(S,1)
plot,abs(c)
```

```
 !p.multi=[0,1,2]
 plot_io, abs(c[0:99]),title='Amplitudes (log scale)'
phase=atan(imaginary(c),float(c))
 plot,phase(0:99),title='Phases (radians)'
```

I guess i want the equivalents of abs(c[0:99]) and phase=atan(imaginary(c),float(c)) in the psf images and then do the thing you said on your first comment?

Subject: Re: Scale the psf on images.
Posted by on Thu, 15 Jan 2015 15:08:45 GMT

View Forum Message <> Reply to Message

Den torsdag 15 januari 2015 kl. 14:33:34 UTC+1 skrev anes.tz...@gmail.com:
> On Monday, January 12, 2015 at 4:15:23 PM UTC+8, anes.tz...@gmail.com wrote:
>> Hi
>>
>> I would like to perform a live test during an upcoming observing run. I want to take a series of frames on various sources and check for variability on them. I would like to make the image subtraction as accurate as possible, thus I think that image subtraction should take place after the psf of the frames is matched.
>>
>> Which technique you think is the best? I tried with the convolve script but the result is not good.
>>
>> imconv = convolve(image1, image2, FT_PSF = psf)
>>
>> image 1 the science frame and image is the psf frame that I created through iraf.
>>
>> Any suggestions are appreciated
>>
>> Thanks a lot.
>
> Well the observations are on Fermi sources at optical bands. We want to take a series of frames for every source and check if there is variability. The variation of the psf will be hopefully small, but since the expected variation is low as well (0.1-0.2) magnitudes, I want the best possible accuracy on the image subtraction.

I can't of course promise you that this is going to give the best accuracy or even that it will work. Depends on your data. I just suggested it again because if it does what you need, it would relieve you from learning enough to do the proper convolution and filter thing.

So try it if you are in hurry. Disregard it if you'd rather do it properly and know what you are doing. The danidl stuff that bstecklu linked to looks interesting, I might want to look into that myself at some point.

> I guess i want the equivalents of abs(c[0:99]) and phase=atan(imaginary(c),float(c)) in the psf

images and then do the thing you said on your first comment?

Yes and no. It is those operations but with that dirty trick you operate on the images only, you don't involve the psfs. You make image2 look more like image1 by replacing the Fourier spectrum of image2 with the one that belongs to image1.

Subject: Re: Scale the psf on images.
Posted by   on Thu, 15 Jan 2015 15:12:22 GMT
View Forum Message <> Reply to Message

Den torsdag 15 januari 2015 kl. 16:08:47 UTC+1 skrev Mats Löfdahl:
> Fourier spectrum

Just so I don't introduce any extra confusion: by "Fourier spectrum" I'm referring to the same thing as when I wrote "Fourier amplitude" before.
