
Subject: Re: Convolution of Stick Spectra
Posted by [bjackel](#) on Sun, 10 Sep 2000 00:44:53 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Todd

It helps a bit to pre-calculate your variance term.
Cuts execution time from 27 to 18 seconds on my PC.

```
variance_term= ( ( (.12*sqrt(energy/1000))/1.6651)*1000)^2
FOR indx=0L,nstick-1 DO BEGIN
    result = result + intensity[indx]* $
        exp(-((energy_scale - energy[indx])^2)/
variance_term[indx] )
ENDFOR
```

Nothing else springs to mind. Your energy dependent width is a real problem ie. can't use single FFT for convolution.

If precision isn't a major issue, you could pre-calculate Gaussians for each energy bin (2048 of them) and just multiply by the appropriate intensities. Only useful if you'll be reading in lots of different spectra in each run.

It'll be interesting to see what others come up with...

Brian Jackel

Todd Clements wrote:

```
>
> Hi all...
>
> In my on-going effort to speed up the code in our lab, I have another
> 'challenge' for you (I have to put 'challenge' in quotes because it seems
> that no matter what I ask, someone knows the answer off (or at least
> nearly off) the top of their head!)
>
> We have a situation where we need to convolute (with energy dependent
> gaussians) a number of stick spectra on a well-defined energy axis. The
> stick spectra are read in from another program as a 2-dimensional array,
> using dddread. The [0,*] elements are the energies of the sticks, and the
> [1,*] values are the intensities. These have no inherent spacing, they are
> just calculated intensities at whatever energy the calculation returns.
```

Subject: Re: Convolution of Stick Spectra
Posted by [mole6e23](#) on Thu, 14 Sep 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

bjackel@phys.ucalgary.ca wrote:

```
> It helps a bit to pre-calculate your variance term.  
> Cuts execution time from 27 to 18 seconds on my PC.  
>  
> variance_term= ( ( (.12*sqrt(energy/1000))/1.6651)*1000)^2  
> FOR indx=0L,nstick-1 DO BEGIN  
>   result = result + intensity[indx]* $  
>       exp(-((energy_scale - energy[indx])^2)/  
> variance_term[indx] )  
> ENDFOR  
>
```

Actually, I was surprised to find that didn't change the timing any on my system (alpha OSF unix 5.3 Nov 11 1999) to any appreciable degree. With the code I first posted (attached below as a procedure), it took about 10.5 seconds either way.

```
IDL> test  
time other:      10.706325  
time precalc:    10.545039
```

Jumping up to 20,000 elements, the timing was still pretty much identical (42.9 vs. 42.7 seconds).

Todd

```
pro test  
convoluted = fltarr( 2, 2048 )  
convoluted[0,*] = findgen( 2048 ) / 2047. * 4.
```

```
;; Let's fake a stick spectrum, we usually have at least this many elements  
stick = abs(randomn( systime(1), 2, 5000 ))  
stick[0,*] = stick[0,*] * 4.  
stick[1,*] = stick[1,*] * 1000.
```

```
time = systime(1)
```

```
variance = (((.12*sqrt(stick[0,*]/1000))/1.6651)*1000)^2  
for i=0L, n_elements( stick ) / 2 - 1 do $  
    convoluted[1,*] = stick[1,i]*exp(-((convoluted[0,*] - stick[0,i])^2)/ $  
        variance[i]) $  
    + convoluted[1,*]
```

```
time2 = systime(1)

for i=0L, n_elements( stick ) / 2 -1 do $
    convoluted[1,*] = stick[1,i]*exp(-((convoluted[0,*] - stick[0,i])^2)/ $
        (((.12*sqrt(stick[0,i]/1000))/1.6651)*1000)^2) $
        + convoluted[1,*]

print, 'time other: ', systime(1) - time2
print, 'time precalc: ', time2 - time

end
```
