Subject: Efficient convolution with irregularly spaced data
Posted by robert.dimeo on Wed, 25 Aug 2004 12:54:29 GMT
View Forum Message <> Reply to Message

Hi,

In data analysis applications that I work on I often need to calculate the mathematical convolution of two vectors. When the data in the kernel and the vector with which it is to be convolved are equally spaced, both CONVOL and FFT work very nicely. However I found that I had to "roll my own" in order to take into account the irregularly spaced data points. I would really rather not use an interpolation or rebinning of the vectors onto a regular grid just so that I could use CONVOL or FFT. Nor do I want to farm this job out to an external routine.

In my implementation, four vectors are passed into the function:

xr: independent variable for the kernel
r: dependent variable defining the kernel
x: independent variable for the vector to be smeared
y: dependent variable defining the vector to be smeared

xr and x do not necessarily need to overlap nor do either of them have to be equally spaced. The routine I wrote, called SUM_CONVOL, is listed below (which I believe is correct numerically).

```
function sum_convol,xr,r,x,y
nxres = n_elements(xr) & nx = n_elements(x)
xmat =  rebin(x,nx,nxres,/sample)-rebin(transpose(xr),nx,nxres,/samp le)
mat = interpol(y,x,temporary(xmat))
 return,total((rebin(transpose(r*deriv(xr)),nx,nxres,/sample) )* $
   ((temporary(mat))),2)
end
```

In writing this function I mustered all of the IDL array mojo I have but it still is not nearly as fast as the (compiled) CONVOL function. I have two questions to the NG: (1) Is there an obvious way to use the FFT or CONVOL routines so that it takes into account irregularly spaced data (other than interpolation)? and, if not, (2) Can anyone see an obvious way to speed up my implementation?

I append an example implementation of the SUM_CONVOL routine at the bottom of this message, called TEST_SUM_CONVOL, so that you can see how I propose to use this function. It uses irregularly spaced vectors for both the kernel and the vector to be smeared.

Many thanks in advance,

Rob

```
; ****************************
function gaussian,x,area,center,fwhm
sig = fwhm/2.354
g = (area/sqrt(2.0*!dpi*sig^2))*exp(-0.5*((x-center)/sig)^2)
return,g
end
; ****************************
pro test_sum_convol
!except = 0
nx = 270
xlo = -3.0 & xhi = 15.0
x = xlo+(xhi-xlo)*randomu(s,nx)
x = x[sort(x)]
xrlo = -3.0 & xrhi = 1.0
xres = xrlo+(xrhi-xrlo)*randomu(s,50)
xres = xres[sort(xres)]
cen1 = 7.0 & cenr = -1.0
fwhm1 = 1.0 & fwhm2 = 1.0
area1 = 2.0 & area2 = 1.0

y = 0.5*(gaussian(x,(3./2.)*area1,cen1,fwhm1)+  $
   gaussian(x,(2./3.)*area1,cen1+2.0,fwhm1))
r = gaussian(xres,area2,cenr,fwhm2)
con = sum_convol(xres,r,x,y)

ymax = max([r,y,con])
xmin = min([x,xres],max = xmax)
plot,x,con,psym = 0,yrange = [0.0,ymax],  $
   xrange = [xmin,xmax],/xsty,/ystyle,thick = 2.0
oplot,xres,r,linestyle = 2,psym = -4
oplot,x,y,linestyle = 1

print,int_tabulated(x,con)
print,(int_tabulated(x,y))*(int_tabulated(xres,r))

end
```