Subject: Re: fast convolving question
Posted by Chris[5] on Thu, 29 May 2008 09:41:59 GMT
View Forum Message <> Reply to Message

On May 28, 5:16 am, rog...@googlemail.com wrote:
> Dear all,
>
> the following code is able to convolve 2 matrices (e.g. 100x100 with
> 100x100) very fast by using 3 different approaches.
> The first one is fft. The second one is based on pre-computing indices
> (only ~6 times slower than fft by the given size) and multiplying
> indexed vectors with 1 for-loop and the third one is without any loop
> (~100 times faster than fft by the given size).
>
> My aim i.e the task is to convolve always without the use of fft, so i
> made this script. Unfortunately something is going wrong with the
> third method and I can't find where the error is.
>
> Please, help me!
>
> Thanks and king regards
>
> Christian
>
> Here it is:
>
> Function convolve, a,b,method,loop=loop
>
> if method eq 'fft' then begin
>
> s1=size(a)
> s2=size(b)
> nx1=s1[1]
> ny1=s1[2]
> nx2=s2[1]
> ny2=s2[2]
> aa=fltarr(nx1+nx2-1,ny1+ny2-1)
> bb=fltarr(nx1+nx2-1,ny1+ny2-1)
> aa[0,0]=a
> bb[nx1-1,ny1-1]=b
> conv=double(shift(fft(fft(aa,-1)*fft(bb,-1),
> 1)*n_elements(aa),nx2,ny2))
> return, conv
> endif
>
> if method eq 'discrete' then begin
>
> kernel = a

```
>  matrix  =       b
>
>  siz_k   =       size(kernel,/dimensions)
>  sx_k    =       siz_k[0]
>  sy_k    =       siz_k[1]
>  sk              =       sx_k*sy_k
>
>  siz_m   =       size(matrix,/dimensions)
>  sx_m    =       siz_m[0]
>  sy_m    =       siz_m[1]
>  sm              =       sx_m*sy_m
>
>  conv    =       fltarr(sm,/nozero)
>  mat             =       fltarr(sx_k+sx_m-1,sy_k+sy_m-1)
>  ;padding matrix with zeros
>   mat[((sx_k-1)/2):((sx_k-1)/2+sx_m-1),((sy_k-1)/2):((sy_k-1)/ 2+sy_m-1)]
>  =       matrix
>  ;compute indices
>  indarray2       =       make_array(sx_m+sx_k-1,sy_m+sy_k-1,/index)
>  indarray        =       transpose(indarray2)
>  ind                     =       reform(indarray[0:sx_m-1,0:sy_m-1],sm,/overwrite)
>  indsmall        =       (indarray[0:sx_k-1,0:sy_k-1])(reverse(indarray2[0:sk-1]))
>  kernel          =       reform(kernel,sk,1,/overwrite)
>
>  ;convolve by multiplying vectors
>  if keyword_set(loop) then $
>          for i=0,sm-1 do (conv)[i]=kernel##mat( indsmall+ind(i) ) $
>
>          else (conv)[(i=indarray2[0:sm-1])]=kernel##mat( indsmall+ind(i) )
>
>  conv            =       reform(conv,sx_m,sy_m,/overwrite)
>  return, conv
>  endif
>  end
>
>  pro test_method_conv
>
>  par     =       100
>  a       =       dist(par)
>  b       =       dist(par)
>
>  t0=systime(1)
>  c1=convolve(a,b,'fft')
>  print,'Convolve with fft: ',systime(1)-t0,' seconds'
>  Window,1,xsize=200,ysize=200,title='Convolve with fft' &
>  TVSCL,congrid(c1,100,100)
>
>  t0=systime(1)
```

```
>  c2=convolve(a,b,'discrete')
>  print,'Convolve discrete without loop: ',systime(1)-t0,' seconds'
>  Window,2,xsize=200,ysize=200,title='Convolve discrete without loop' &
>  TVSCL,congrid(c2,100,100)
>
>  t0=systime(1)
>  c3=convolve(a,b,'discrete',/loop)
>  print,'Convolve discrete with loop: ',systime(1)-t0,' seconds'
>  Window,3,xsize=200,ysize=200,title='Convolve discrete with loop' &
>  TVSCL,congrid(c3,100,100)
>  end
```

Hmm- why don't you just use the build in IDL function convol?
