Subject: Re: Convolving speed issue
Posted by pgrigis on Thu, 17 Apr 2008 17:16:05 GMT
View Forum Message <> Reply to Message

I haven't followed exactly what you are doing below,
and I am not familiar with Matlab's conv2, but
couldn't you use the FFT method instead?
That should be much faster.

Paolo

rog...@gmail.com wrote:
> Hi there,
> I have a strange problem with the IDL convolving possibilities. I'd
> like to make a script which convolves 2 matrices - e.g. a and b - in
> the same behavior conv2(a,b,'same') Matlab does. The Problem is not to
> make such a script, the problem is that IDL takes too long or hangs
> when i try to convolve larger matrices. I tried certainly all kinds of
> using the built-in IDL-convol method, but convolving large arrays ends
> always in different results compared to the very fast Matlab conv2.
> Maybe someone could help me. Here's the sample code:
>
> function size_dim, in, direction
> dims = size(in, /dimensions)
> return, dims[direction]
> end
>
> function zeropadding, in,xsize,ysize
> ;only for 2D-arrays
> xsize_in=size_dim(in,0)
> ysize_in=size_dim(in,1)
> shiftx = ceil((xsize-xsize_in)/2)
> shifty = ceil((ysize-ysize_in)/2)
> temp = fltarr(xsize,ysize)
> temp[0:xsize_in-1,0:ysize_in-1] = in
> temp = shift(temp,shiftx,shifty)
> return, temp
> end
>
> function conv2, a,b
> size_a=[size_dim(a,0),size_dim(a,1)]
> size_b=[size_dim(b,0),size_dim(b,1)]
> a=zeropadding(a,size_a[0]+size_b[0], size_a[1]+size_b[1])
> b=zeropadding(b,size_a[0]+size_b[0], size_a[1]+size_b[1])
> c=fltarr(size_a[0]+size_b[0], size_a[1]+size_b[1], /nozero)
> addx = floor(total(size_a)/4)
> endx = ceil(double(total(size_a)/4))
> addy = floor(total(size_b)/4)

```
> endy = ceil(double(total(size_b)/4))
>
> for n1=0,size_a[0]+size_b[0]-1 do begin
>  for n2=0,size_a[1]+size_b[1]-1 do begin
>   temp=0
>   temp2=0
>   for k1=0+addx,size_a[0]+size_b[0]-1-endx do begin
>    for k2=0+addx,size_a[1]+size_b[1]-1-endy do begin
>     if n1-k1 gt-1 and n2-k2 gt-1 then begin
>      temp=a[k1,k2]*b[n1-k1+addx,n2-k2+addy]
>      temp2=temp2+temp
>     endif
>    endfor
>   endfor
>   c[n1,n2]=temp2
>  endfor
> endfor
> temp = shift(c,-2*addx,-2*addy)
> return, temp[0:size_a[0]-1,0:size_b[0]-1]
> end
>
>
> pro conv
> ; sample matrix -> magic(5) in Matlab
> a=  [[17,  24,   1,   8,  15],$
>     [23,   5,   7,  14,  16],$
>      [4,   6,  13,  20,  22],$
>     [10,  12,  19,  21,   3],$
>     [11,  18,  25,   2,   9]]
> b=2*a
> c=a
> d=b
> print, 'Trying own convolution...',string(10b),conv2(a,b), string(10B)
> print, 'Trying built in convolution...',string(10b),$
> shift(convol(zeropadding(c,10,10),zeropadding(d,10,10), center=0,/
> edge_wrap),-4,-4),string(10b)
> end
>
> Maybe there is a solution for using reform in some way? It seems to be
> quicker as for-loops. But I can't imagine how it could work when the
> indices of the multiplying matrices are varying.
>
> Hope on help
>
> Thank you and best regards
>
> Chris
```