
Subject: Re: CUDA version of RANDOMN?

Posted by [wlandsman](#) on Fri, 15 Aug 2008 17:15:32 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Aug 15, 11:28 am, wlandsman <wlands...@gmail.com> wrote:

> On Aug 15, 11:16 am, "hotplainr...@gmail.com" <hotplainr...@gmail.com>
> wrote:

>
>
>

>> On Aug 16, 12:28 am, wlandsman <wlands...@gmail.com> wrote:

>>> On Aug 15, 10:11 am, "hotplainr...@gmail.com" <hotplainr...@gmail.com>
>>> wrote:

>

>>>> Hey guys,

>

>>>> I need to write a kernel to replace the IDL RANDOMN POISSON

>

>>>> for loop

>>>> for loop

>>>> for loop

>>>> c = data[x,y,b]

>>>> if c gt 0.0 then begin

>>>> n = RANDOMN(seedP, POISSON=c)

>>>> endif else begin

>>>> n = 0

>>>> endelse

>>>> data[x,y,b] = n

>>>> endfor

>>>> endfor

>>>> endfor

>

>>>> Could someone point out an example code of how RANDOMN POISSON so that

>>>> I can implement it in CUDA?

>

>>> Your best bet is to probably look at the Poisson generating algorithm

>>> in "Numerical Recipes in C" if you are going to implement it CUDA.

>

>>> I have implemented the "Numerical Recipes in C" algorithm into the IDL

>>> procedure poidev.pro at <http://idlastro.gsfc.nasa.gov/ftp/pro/math/poidev.pro>.

>>> Although poidev.pro is normally slower than calling randomn(POISSON=),

>>> it has advantages for just the problem you describe, which can be

>>> written as simply

>

>>> data = poidev(data)

>

>>> rather than using a triple FOR loop. --Wayne

```

>
>> Thanks for the reply. I was about to use your code until I discovered
>> the problem of achieving this.
>
>>         c = data[x,y,b]
>>         if c gt 0.0 then begin
>>             n = RANDOMN( seedP, POISSON=c )
>>         endif else begin
>>             n = 0
>>         endelse
>
>> I guess the only way is to code a poisson kernel and then do tiling on
>> the data.
>
> Yes, that does mean the code becomes 3 lines instead of 1
>
> g = where( data GT 0, Ng ,complement=g1, Ncomplement=Ng1)
> if Ng GT 0 then data[g] = poidev(data[g])
> if Ng1 GT 0 then data[g1] = 0
>
> --Wayne

```

I still made it too complicated. poidev automatically sets any negative numbers to zero (since the Poisson distribution is not defined for negative numbers). So the original code

```
data = poidev(data)
```

should be fine. --Wayne
