Subject: Re: Generating N random numbers that add to a TOTAL
Posted by Russell Ryan on Fri, 08 Aug 2014 19:32:47 GMT
View Forum Message <> Reply to Message

Hi Mike,
I might be interested in GPULib. I see it's quite pricey (at least for my budget) and Tech-X offers a free trial. But before I go through the trouble of even getting the free trial, what else can you tell me about GPULib?

Specifically, I was wondering about what hardware/software do I need to use GPULib? I use Mac OSX 10.8.5 and IDL 8.2.3 at present, and that sounded sufficient. But anything else I should be aware of? Such as GPU cards, RAM, etc.?

Can you give any examples of the code usage? Like what will my IDL code now look like?

What about if I need to port the code to another workstation?

Anything else a GPU newbie (but seasoned IDLer) should know or should ask?

THanks,
Russell

On Friday, August 8, 2014 12:54:19 PM UTC-4, Mike Galloy wrote:
> On 8/6/14, 9:52 PM, Gianguido Cianci wrote:
>
>> Hi all,
>
>>
>
>> I am wondering if anybody has suggestions on how to improve the function below. It seems ok for floating precision numbers.
>
>>
>
>> For integers it's a different story:
>
>> It works great if N<<TOTAL. When N approaches TOTAL I get a few numbers and then a bunch of zeros... Also, setting /DIFFERENT makes it run for ever if N is large. Also, the sum of res adds up TOTAL=/-1, not always to TOTAL exactly...
>
>>
>
>> Suggestions?
>
>>
>
>> Thanks,

```
>
>> Gianguido
>
>>
>
>>
>
>>
>
>> FUNCTION nrndaddto,  n,  total,  integers = integers,  different = different
>
>>
>
>> compile_opt idl2
>
>>
>
>> res = dblarr(n)
>
>> res[0] = randomu(seed, 1,  /double)*(total)
>
>>
>
>> FOR i = 1,  n-2 DO BEGIN
>
>>     res[i] = randomu(seed, 1,  /double)*(total-total(res[0:i-1],  /double))
>
>> ENDFOR
>
>> res[n-1] = total-total(res[0:n-2],  /double)
>
>>
>
>> IF ~keyword_set(integers) THEN integers = 0
>
>>
>
>> IF keyword_set(integers) THEN res = round(res)
>
>> IF keyword_set(different) THEN BEGIN
>
>>     IF n_elements(res) NE n_elements(unique(res,  /sort)) THEN res = $
>
>>     nrndaddto(n, total, integers = integers, different = 1)
>
>> ENDIF
>
>>
```

>
>>
>
>> RETURN, res
>
>> END
>
>>
>
>
>
> What about just FLOOR the normalized float values and then just
>
> increment the required number of values with the largest remainders?
>
>
>
> function mg_random_to_total, n, sum, seed=seed
>
> compile_opt strictarr
>
>
>
> x = randomu(seed, n)
>
> x *= sum / total(x, /preserve_type)
>
> int_x = long(floor(x))
>
> dec_x = x - int_x
>
>
>
> int_total = total(int_x, /preserve_type)
>
> ind = sort(dec_x)
>
> int_x[ind[0:(sum - int_total - 1)]]++
>
>
>
> return, int_x
>
> end
>
>
>
> Mike

>
> --
>
> Michael Galloy
>
> www.michaelgalloy.com
>
> Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)
>
> Research Mathematician
>
> Tech-X Corporation