Subject: Re: random musings
Posted by news.qwest.net on Mon, 19 Mar 2007 20:37:44 GMT
View Forum Message <> Reply to Message

"Christopher Thom" <cthom@oddjob.uchicago.edu> wrote in message
news:Pine.SOC.4.64.0703191516310.28339@oddjob.uchicago.edu.. .
> Hi all,
>
> this morning I started writing some monte carlo code, in which I'm
> generating random arrays, and scaling/shifting them to fit the (gaussian)
> distribution parameters I need. One thing is not clear in the online help:
> Which random function is "better"? Should I be using randomn() or
> randomu()? The help is explicit about the algorithms used, but there is
> comparison between the randomness of the two functions. And since both
> functions accept a /normal flag, it's unclear as to which is the better
> choice.
>
> Is there any common wisdom that makes this choice obvious?
>
> cheers
> chris

That's a good question. I have always used randomn for my purposes.
If you really need a good RNG, check out:
http://sprng.cs.fsu.edu/Version2.0/users-guide.html
You can read more about RNG and Monte Carlo and the
tests they perform in the user's guide. They have a PC linux port
available.

Cheers,
bob

Subject: Re: random musings
Posted by news.verizon.net on Mon, 19 Mar 2007 20:58:47 GMT
View Forum Message <> Reply to Message

On Mar 19, 4:30 pm, Christopher Thom <c...@oddjob.uchicago.edu> wrote:
> Hi all,
>
> this morning I started writing some monte carlo code, in which I'm
> generating random arrays, and scaling/shifting them to fit the (gaussian)
> distribution parameters I need. One thing is not clear in the online help:
> Which random function is "better"? Should I be using randomn() or
> randomu()?

I believe these functions are identical. In the ancient days, IDL
had two types of randomn number generators: RANDOMU, for a uniform

distribution, and RANDOMN for a normal distribution.   When addition
distributions (e.g. binomial, Poisson, Gamma) were added (in V5.0 I
believe), RSI chose against adding additional RANDOM* functions, and
instead made all the distributions available from either RANDOMU or
RANDOMN.

This is one of those case where IDL suffers from having a long
history.

--Wayne

Subject: Re: random musings
Posted by Christopher Thom on Tue, 20 Mar 2007 04:41:40 GMT

View Forum Message <> Reply to Message

Quoth Wayne Landsman:

> On Mar 19, 4:30 pm, Christopher Thom <c...@oddjob.uchicago.edu> wrote:
>> Hi all,
>>
>> this morning I started writing some monte carlo code, in which I'm
>> generating random arrays, and scaling/shifting them to fit the (gaussian)
>> distribution parameters I need. One thing is not clear in the online help:
>> Which random function is "better"? Should I be using randomn() or
>> randomu()?
>
> I believe these functions are identical. In the ancient days, IDL had
> two types of randomn number generators: RANDOMU, for a uniform
> distribution, and RANDOMN for a normal distribution. When addition
> distributions (e.g. binomial, Poisson, Gamma) were added (in V5.0 I
> believe), RSI chose against adding additional RANDOM* functions, and
> instead made all the distributions available from either RANDOMU or
> RANDOMN.

Hmmmm. If this is true, then the documentation is definitely in need of an
update. Randomn() supposedly uses the "box muller method", while randomu()
uses one from an ACM paper that is cited, with a "bays durham shuffle
added", giving something "similar to ran1()" from NR in C section 7.1.
If they are the same thing, it is certainly not clear from the docs. :-(

OTOH, the interfaces seem very similar...they all provide the same flags
(/gamma, /normal etc), which certainly points towards a common algorithm.

Oh. I just did the obvious thing -- tried both with the same seed:

```
IDL> x=randomn(180479L, 5,/normal)
IDL> y=randomu(180479L, 5,/normal)
IDL> print,x,y
     2.01146     -1.55186   -0.0893551    -0.627972    -0.780082
     2.01146     -1.55186   -0.0893551    -0.627972    -0.780082
```

Surely different algorithms would return different randomness for the same seed.

Problem solved...

cheers
chris

Subject: Re: random musings
Posted by Paolo Grigis on Tue, 20 Mar 2007 08:53:15 GMT

View Forum Message <> Reply to Message

Christopher Thom wrote:

```
> Quoth Wayne Landsman:
>
>> On Mar 19, 4:30 pm, Christopher Thom <c...@oddjob.uchicago.edu> wrote:
>>> Hi all,
>>>
>>> this morning I started writing some monte carlo code, in which I'm
>>> generating random arrays, and scaling/shifting them to fit the (gaussian)
>>> distribution parameters I need. One thing is not clear in the online help:
>>> Which random function is "better"? Should I be using randomn() or
>>> randomu()?
>> I believe these functions are identical.  In the ancient days, IDL had
>> two types of randomn number generators:  RANDOMU, for a uniform
>> distribution, and RANDOMN for a normal distribution.  When addition
>> distributions (e.g. binomial, Poisson, Gamma) were added (in V5.0 I
>> believe), RSI chose against adding additional RANDOM* functions, and
>> instead made all the distributions available from either RANDOMU or
>> RANDOMN.
>
> Hmmmm. If this is true, then the documentation is definitely in need of an
> update. Randomn() supposedly uses the "box muller method", while randomu()
> uses one from an ACM paper that is cited, with a "bays durham shuffle
> added", giving something "similar to ran1()" from NR in C section 7.1.
> If they are the same thing, it is certainly not clear from the docs. :-(
```

In most implementations, uniformly distributed random numbers are generated first, and then some trick is used to transform these into the distribution you want (e.g. normal in this case). Therefore, "ran1()" (or whatever flavor thereof they use internally) produces the uniform samples, an then the box muller transformation is applied to these values to get the normally distributed samples.

Ciao,
Paolo

>
> OTOH, the interfaces seem very similar...they all provide the same flags
> (/gamma, /normal etc), which certainly points towards a common algorithm.
>
> Oh. I just did the obvious thing -- tried both with the same seed:
>
> ```
> IDL> x=randomn(180479L, 5,/normal)
> IDL> y=randomu(180479L, 5,/normal)
> IDL> print,x,y
>       2.01146    -1.55186   -0.0893551    -0.627972    -0.780082
>       2.01146    -1.55186   -0.0893551    -0.627972    -0.780082
> ```
>
> Surely different algorithms would return different randomness for the same
> seed.
>
> Problem solved...
>
> cheers
> chris