

Re: random numbers in fortran

Source: <http://coding.derkeiler.com/Archive/Fortran/comp.lang.fortran/2006-12/msg00072.html>

- *From:* Herman D. Knoble <SkipKnobleLESS@xxxxxxxxxxxxxxxxxx>
 - *Date:* Thu, 30 Nov 2006 10:03:14 -0500
-

Gordon: I agree with you about seeking information from statistics and mathematical news groups. I disagree with that the value of 52! has any PRACTICAL bearing on this simulation at hand; that is, it is not necessary to COMPUTE 52! for simulations involving a card shuffling once a "reputable" RNG is available (which they are as outlined in one of my postings). I also disagree that comp.lang.fortran is not the best place to seek such advice.

ANY Monte Carlo simulation is better done with a statistically better quasi-random number generator. This includes this shuffling simulation. Theory is very important, but there does not exist a Monte Carlo simulation (computation) that is perfect theoretically. Yet there are some "good" simulations out there, right?

I agree with your precaution that no simulation is as good as it can be without theory. Knuth Volume 2 shows this very clearly in the introduction (demonstrating that a random process does not yield a random number generator with good statistical properties.)

I'd suggest to the original posters that they do BOTH/AND: Both seek expert statistician's/mathematician's advice, and seek comp.lang.fortran advice.

Cheers.
Skip

On Thu, 30 Nov 2006 14:27:10 GMT, Gordon Sande <g.sande@xxxxxxxxxxxxxxxxxx> wrote:

-|On 2006-11-29 23:30:57 -0400, "Mark Morss" <mfmorss@xxxxxxx> said:
-|
-|> I am sure that the original poster will appreciate this highly
-|> instructive reply. My own interest is not how best to simulate deals
-|> of 52 cards, but how best to simulate discrete distributions with
-|> extremely many possible outcomes.
-|>
-|> What G. Sande and some others seem to have been saying is that, unless
-|> I make sure that I have a generator capable of producing any of the
-|> possible outcomes, and with the correct probability, there is something
-|> fundamentally wrong with my simulation. I admit that I do not fully

Re: random numbers in fortran

- > understand this point, but when I asked for an elucidation and some
- > explanation of its practical implications, G. Sande rather unhelpfully
- > suggested that my question demonstrated so much ignorance that I should
- > consult some "real literature."
- |
- |While it is true that many Fortran users are also experts in other
- |substantive fields it should be self evident that if you want to
- |ask about random number generation then you should try a statistics or
- |mathematics newsgroup.
- |
- |The c.l.f readers are tolerant of such off topic questions but do expect
- |that substantive questions be addressed to the relevant substantive area.
- |Some computer language newsgroups would just treat such questions as
- |off topic and deserving flame targets.
- |
- |Fortran manuals are not real literature on anything but Fortran. There is
- |lots literature on random number generation and you should be prepared to
- |read it if you want serious answers to hard questions. It will help
- |to be able to deal with the relevant mathematics.
- |
- > My first point is, I doubt whether, given the limitations of any
- > particular random number generator, it is possible to know that the
- > underlying distribution is correctly represented even if the cycle
- > length is theoretically sufficient. Secondly, and not alone here, I've
- > argued that so long as the random draws in my simulation are
- > indistinguishable, in very many tries, from an analogous set of draws
- > from the true distribution, it really does not matter whether any other
- > criteria are satisfied. Most particularly, if indistinguishability is
- > satisfied, it does not matter whether there exists a subset of possible
- > outcomes which, starting from a given random seed, are incapable of
- > being produced.
- >
- > Say for example, you have to play Blackjack against two dealers, one
- > shuffling after each deal with an "adequate" generator and one
- > shuffling with an "inadequate" one -- but you do not know which. I
- > hope that some advocates of "adequate" simulation methods will be able
- > to say how, in a less than infinite number of deals, they would
- > discover which was which dealer was which and be able take advantage of
- > it.
- >
- > That's what I meant by practical implication.
- >
- > Ignorant question: I understand also that with a short cycle, the
- > outcomes that can be reached are a function of the seed; with a
- > different seed, a different set of outcomes becomes possible.
- > Generally in practice then, could I not compensate for a short cycle by
- > sometimes changing my seed?
- |
- |A seed is a starting point on the cycle. If you go around the whole cycle
- |then the seed does not matter, up to questions the arise if you are using
- |fixed blocks of values. Trivial example is pairs for normals on even length

Re: random numbers in fortran

-|where the seed being "odd" or "even" would be different. If the short
-|cycle did not permit the type of event you are interested in then you will
-|never find a seed which compensates for its absence.
-|
-|If you are trying to make serious statements about the validity of your
-|experiments than having a simulation that does not have a chance of visiting
-|some events brings into question the seriousness of the simulation. Arguing
-|with Fortan programmers will not get you anywhere useful when your question
-|should be addressed to statisticians and mathematicians. The usual advice
-|will tend to be that you should go consult with a real professional as even
-|newsgroups are of little use for the type of questions you seem to want
-|addressed.
-|
-|> Herman D. Knoble wrote:
-|>> Mark:
-|>>
-|>> What I hear you asking for are some practical solutions to this problem.
-|>>
-|>> In words one solution is to generate random permutations (sampling
-|>> without replacement). For N=52 this will produce a random shuffling
-|>> of a 52-card deck. The code to do this requires two parts:
-|>>
-|>> 1) A Random number generator with "decent" properties.
-|>> The built-in Random_Number intrinsic will do. Or you can use one
-|>> of several good quasi-random number generators. This latter
-|>> strategy would enable one to generate the same quasi-random
-|>> sequence across compilers (and platforms). For example:
-|>>
-|>> <http://users.bigpond.net.au/amiller/random.html>
-|>> (Cycle lengths of these are documented in the leading comments;
-|>> for example TAUS88.F8=90 has a cycle length of about 3E+26).
-|>>
-|>> <http://ftp.cac.psu.edu/pub/ger/fortran/hdk/byterand.f90>
-|>> (Cycle Length of this one is: 6953607871644 > 6.95E+12)
-|>> Also see: <http://www.cs.berkeley.edu/~daw/rnd/>
-|>>
-|>> 2) A Random Permutation code. Knuth provides this. See Subroutine
-|>> RPERM within the code: <http://ftp.cac.psu.edu/pub/ger/fortran/hdk/byterand.f90>
-|>> The sample driver here prompts for N and 3 seeds. Choose N=52 for example.
-|>>
-|>> Unless I'm missing something in this thread, there is no need to
-|>> worry about the number 52!, the number of different permutations of N=52.
-|>>
-|>> Skip Knoble
-|>>
-|>>
-|>> On 28 Nov 2006 12:01:27 -0800, "Mark Morss" <mfmorss@xxxxxxx> wrote:
-|>>
-|>> -|Actually I was not the person who made the original post, but in any
-|>> -|case, my question was, what are the >practical< implications of your
-|>> -|point? This being a forum devoted to Fortran, not combinatorics, I was

Re: random numbers in fortran

-|>> -|hoping for an explanation of how, in Fortran, one would make sure that
-|>> -|this criterion were satisfied.
-|>> -|
-|>> -|How, for example, would one set "cycle length" to be sure that each one
-|>> -|of 52! combinations had the same chance of coming up? My Fortran
-|>> -|reference is Chapman's Fortran 90/95 for Scientists and Engineers,
-|>> -|and in its brief discussion of the RANDOM_SEED and RANDOM_NUMBER
-|>> -|intrinsic, it makes no mention of "cycle length."
-|>> -|
-|>> -|But also on a practical level, given the uncertain randomness of
-|>> -|pseudo-random numbers, how could anyone know that each one of so many
-|>> -|combinations had the same chance of coming up? I would have thought
-|>> -|that if a very large number of simulated deals satisfied apparent
-|>> -|randomness, that would be sufficient for most practical purposes --
-|>> -|even if it were known >a priori< that certain outcomes possible in
-|>> -|reality were impossible in the simulation.
-|>> -|
-|>> -|Gordon Sande wrote:
-|>> -|> On 2006-11-28 10:07:31 -0400, "Mark Morss" <mfmorss@xxxxxxx> said:
-|>> -|>
-|>> -|> You said:
-|>> -|> >
-|>> -|> > "The basic statistical property that you should worry about first is
-|>> -|> > the
-|>> -|> > cycle length. To shuffle cards you want all 52! permutations to have a
-|>> -|> > chance of happening. With a short cycle length that does not happen.
-|>> -|> > A rather critical issue if you are going to pretend that the results
-|>> -|> > have any relevance to real card games."
-|>> -|> >
-|>> -|> > Would you please elucidate? I am not sure of the practical
-|>> -|> > implications of this remark.
-|>> -|>
-|>> -|> You are trying to shuffle cards, or so you said.
-|>> -|>
-|>> -|> There are 52! possible shuffles. Get a high precision calculator and
-|>> -|> do the many precision arithmetic. Or just use an approximation. It is
-|>> -|> more than you can count on your fingers! Or even your fingers and toes.
-|>> -|>
-|>> -|> With a short cycle length the PSEUDO random number generator will not
-|>> -|> be able to represent all those permutations. Even if it had a chance it
-|>> -|> still might not actually have the permutation somewhere on its cycle.
-|>> -|> The first is elementary combinatorics and the second is a more subtle
-|>> -|> issue on the quality of the pseudo random number generator.
-|>> -|>
-|>> -|> If there are permutations missing then anything you do based on
-|>> -|> the "random" permutations will be at best "hap-hazard". That is
-|>> -|> fine for games for young children with short memories but is a bit
-|>> -|> short on quality for most other applications.
-|>> -|>
-|>> -|> You dropped the piece where it said
-|>> -|>

Re: random numbers in fortran

-|>> -|> > It turns out that this "trivial example" is in fact quite demanding of
-|>> -|> > a pseudo random number generator.
-|>> -|>
-|>> -|> which comes as a big surprise to many who do not try to do the simple
-|>> -|> combinatorics. You have lots of company on that.
-|>> -|>
-|>> -|> If all you are doing is a game for young children then fine. Otherwise
-|>> -|> you should read some real literature on the topic which your question
-|>> -|> indicates that you have not yet done.
-|