Fortran version, Jan 28, 199

This version contains Fortran source code. Keep in mind
that these are a melange of patched and blended programs,
parts of which are over thirty years old. They seem to work
when compiled on a (PC) Linux or Sun unix system, and perhaps
they will compile and run for you.
But keep in mind that I throw these in as a convenience in response
to requests from people whose systems were not able to compile
and run the C source code, and please do not blame me for not having
the time to reprocess everything and make it neat, tidy and readable.

Instructions for using
DIEHARD: a battery of tests of randomness.

I hope you will inform me of results, good or bad, of new kinds
of generators you have tested, particularly deterministic
generators, but also the output of physical devices.
(I have found none of the latter that get past DIEHARD,
and would like to learn of any that do.)
Since, in my opinion, there is no true randomness,
collective experience in finding sequences that depart from
the theoretical ideal in a significant way can perhaps lead
to better ways for finding those that do not.

&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

1. How To Use DIEHARD.

This disk contains examples of some tests (15) from the DIEHARD
battery of tests I have developed over many years and found effective.
It also contains some utility files that are described at the end
of this text file. This description is primarily for running
*.exe files under DOS, but should serve as well for UNIX or LINUX
systems with obvious allowance for the absence of .exe labels in unix.

```
 ----------------------------------------------------------|
|The source code is also available in C.  It comes from   |
|an f2c translation from many Fortran files that have     |
|suffered countless patches over their long development,  |
|some going back thirty years.  So while the melange in C|
|might seems to run and  give good results, good luck in  |
|trying to figure out or modify the translated code.      |
|---------------------------------------------------------|
```

To use the disk you need only invoke the diehard.exe file on a PC
using DOS and having 386/486 capabilities.

(For DOS use, you must copy and have in your path the file
DOSXMSF.EXE, included on the disk, before diehard.exe will run.
You will also need files tests.txt, make.txt and operm5d.ata.
To be safe, copy all the files on this disk to your working directory.)

The tests in DIEHARD require that you provide a large binary
file of random integers to be tested. The disk contains the file
makewhat.exe for creating such binary files, as well the file
asc2bin.exe for transforming an ascii file into the kind of file

that DIEHARD expects.

You will no doubt want to copy the contents of this disk to your hard disk and invoke its .exe files from there.

With the exception of the RUNS test, which is a `standard' test and the only one of the standard tests I have found to be very effective, all tests here are those I have developed.

General versions of the tests call for various parameters that specify sample size, bit patterns to be tested, and so on. To avoid complications, I have chosen to fix the parameters of the tests on this disk. For each test, a description is provided, from which interested users may fashion their own versions with variable parameters.

2. Providing Input Files For DIEHARD.

The executable file diehard.exe will prompt you to name the file you want to be tested and ask you to select any or all of the 15 tests. You must provide the binary file that DIEHARD expects---a file of 10 to 11 megabytes, that is, a file of at least 80 million bits. DIEHARD will do the best it can with shorter binary files, then give up on the test under way, give an END OF FILE message and go to the next test.

3. Some Examples.

Imagine, (as is probably the case), you have just got this disk (or diehard.zip via ftp and have unzipped it), have all the files in your current directory and are ready to try DIEHARD. (Having DOSXMSF.EXE in your path.)

If you merely invoked it with the DOS command

```
diehard
```

you would be prompted to give the name of the file you want tested. But of course you have no such file.

So you must first create one. You may use the file makewhat.exe to create files of the form that DIEHARD expects. Suppose you try it and when prompted, choose option 4: make a KISS file. That creates a binary file of integers from the generator KISS, which stands for

```
Keep It Simple Stupid,
```

a generator that combines three simple generators of the form x <-- f(x) for 32 bit integers x. (You will find that KISS, among several others, passes all tests.)

Then the makewhat.exe program will create the binary file kiss.32. After displaying details of the KISS generator, the program will ask you to enter four seed integers, after which it will create a binary file of some 11 megabytes.

Having finished creating the file that will test KISS, enter

```
     diehard
  and you will be prompted to enter the name of the file to be tested.
  You will of course enter the name of your only file so far, say
     kiss.32
  You will then be asked to enter the name of an output file, since
  DIEHARD displays results both to the screen and to a specified output
  file.  You might, as I do, choose
     kiss.out
  for the name of your output file.

  You will then be asked, with a crude menu, to choose the tests
  you want run.  This is done with a sequence of 15 1's or 0's.
  For example: 111111111111111 selects all the tests, while
  010101010101010 selects only the even numbered menu items.

   For a fast PC, DIEHARD may take perhaps 10 minutes for the entire
   battery of tests (but an early 386 I tested it on took 203 minutes).

   You will see the output on the screen and, when the tests are
   completed, you may examine results in detail by displaying the
   file kiss.out.

   Next, you may want to test some other generators. The makewhat.exe
   program provides for creating binary files for generic

     Congruential Generators
     Shift Register  Generators
     Lagged Fibonacci  Generators
     Multiply-with-carry Generators on 32 bits: x(n)=ax(n-1)+carry mod
2^32

   There are also choices for specific generators:
     A version of the mother-of-all RNG's
     KISS
     Two multiply-with-carry on 16 bits
     A simple but good combination generator COMBO
     The super-duper generator
     A subtract-with-borrow  generator
     The system generator in Microsoft or Sun Fortran
     An extended congruential generator
     The ran2 generator from Numerical Recipes
     An inverse congruential generator

   There is also a file that will combine, by addition mod 2^32
   or exclusive-or,  any two binary files that have already been created.
   The file is MERGE.EXE, and of course by repeated invocations you
   can create a mixture of as many binary files as you wish.
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
:

   OK, those are some of the ways you can create and test files
   with programs on this disk.

   But what if you want to test your own---or some other---generator?
```

And it is only available through some program or procedure
in Fortran, C, Pascal etc.?

There are at least two ways:
1) Write a main program in the language of your procedure, in which you open a binary file and send your random integers to it.

But if your language can't---or you don't know how to---create a binary file of the form required by DIEHARD, then

2) Write a main program that sends the (hex) output of your random number procedure to an ordinary ascii file. Then invoke the file on this disk, ASC2BIN.EXE that will prompt for the names of your input and output files and do the conversion.

Your random number generator should produce 32-bit integers. (If 31 bits, left justify by shift-left-one, as some of the tests in DIEHARD favor leading bits.) You should send them to your ascii file in hex form, 8 hex 'digits' per integer, 10 integers per line, no intervening spaces. The ascii file will be twice as big as the binary file it is reduced to.

You can see that DIEHARD requires large test files. But they need only be created one at a time, then deleted.

The CDROM I will soon be distributing contains some 60 ten-meg binary files of random numbers for use in serious Monte Carlo calculations as well as for input to DIEHARD. Most of those files combine several of the best deterministic generators with white noise and black noise, (the latter from a CD of rap music). Some of the files contain only random bits from physical devices. They fail many of the tests in DIEHARD.

When you absolutely, positively have to have an unassailable source of random bits, you should, in my opinion, use a combination of the output from a physical device(s) with some of the best deterministic random number generators. As you can see from applying DIEHARD, many of the latter may well be suitable by themselves.

George Marsaglia
geo@stat.fsu.edu

Addendum: You will also find a postscript file mwc1.ps that, when processed, describes what I think are the RNG's of the future: multiply-with-carry. You will find they pass all tests, have very long periods and are particularly well suited for computer implementation.
I predict a version with base b=2^32 will be the generator of choice in future systems. That file mwc1.ps is extracted from a summary description of RNG's, tests and the theory behind them im a TeX file that I may or may not ever complete.
I have also included the postscript file keynote.ps. It is taken from

the Keynote Address of a national meeting 11 years ago, and describes some of the ideas behind the tests. Many have reported difficulty in finding Proceedings of the Meeting in their libraries.

Another potentially useful file is getrnor.exe. It will transform a binary file of random bits into a (binary) file of real standard normal random variables using my ziggurat method.

Files on this version of DIEHARD:

diehard.doc	
diehard.exe	The diehard exec file
diequick.exe	A terse version of diehard
makewhat.exe	The exec file for making random number files
tests.txt	Descriptions of the tests. Read by diehard.exe
make.txt	Descriptions of RNG's. Read by makewhat.exe
asc2bin.exe	Converts an ascii integer file to a binary file
getrnor.exe	Converts a random bit file to normal random variables
merge.exe	Makes a new binary bit file by merging two others
mwc1.ps	A postscript file describing multiply-with-carry RNG's
monkey.ps	Describes the difficult-to-pass monkey tests.
cdmake.ps	Early version of postscript file for the CDROM
operm5d.ata	The covariance matrix read by the OPERM5 test
dosxmsf.exe	A file necessary for running diehard.exe under DOS

Versions for UNIV or LINUX will not have that last DOS-required file, and the executable files have no .exe extensions, being simply
diehard, diequick, makewhat, asc2bin, getrnor, and merge
They will also have a different version of the makewhat and make.txt files, calling for different choices for tests of the system generator.