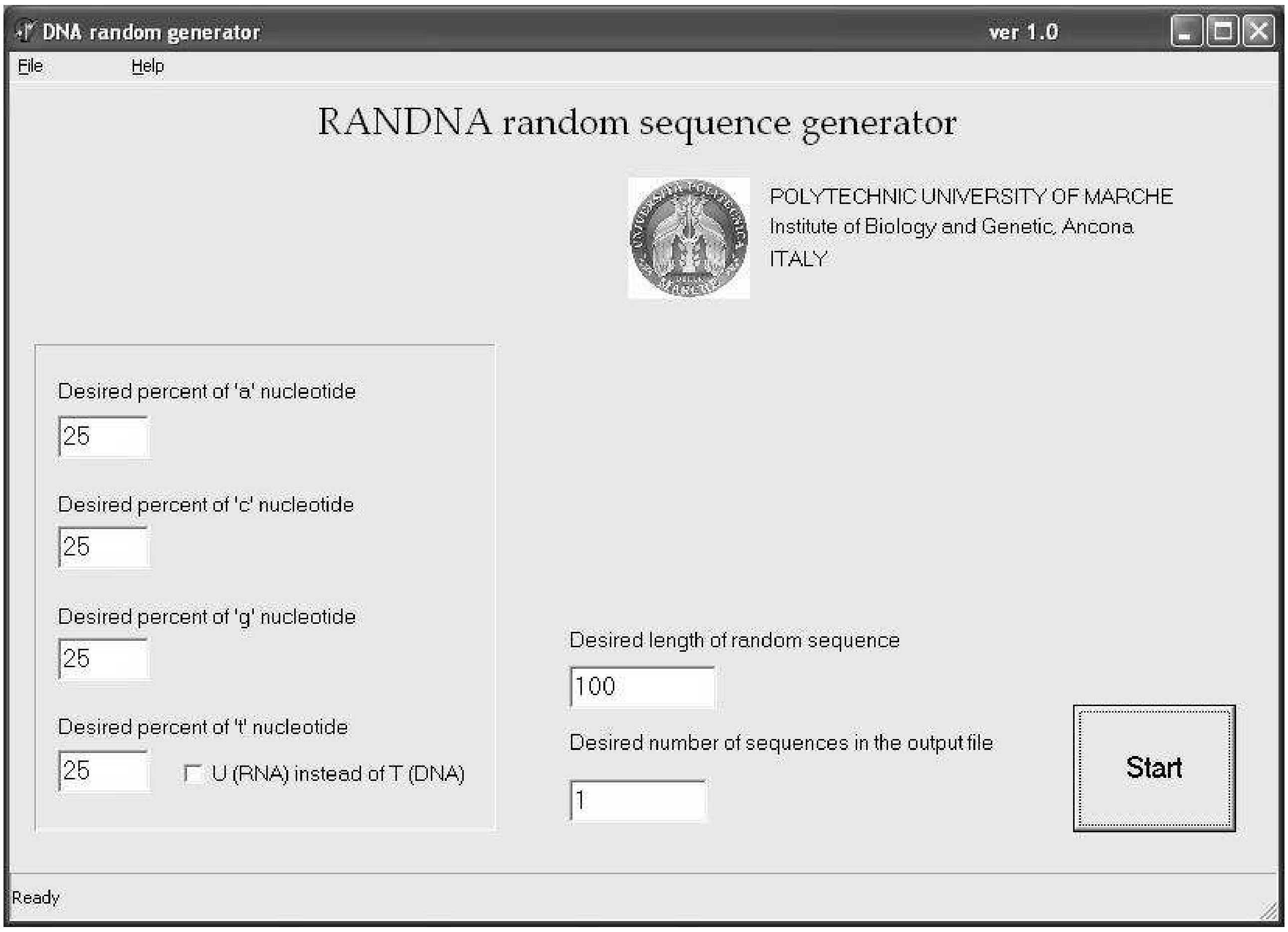
RANDNA

General

RANDNA is a free software which allows to produce random DNA sequences setting both their length and the percentage of nucleotide composition. Random number generator function of Borland Delphi 6 is used, since it guarantees a good randomness, a long cycle length and a high speed. This tool is useful for Monte Carlo simulations, for example to verify the significance of genomic regularities, like the nucleotide correlations or the not uniform distribution of the motifs throughout genomic or mature mRNA sequences. Aligning random sequences (among them) is a good test to estimate the background score of an alignment, that is a threshold below of which the resulting score is merely due to the chance. Its graphic interface allows to easily set the parameters that characterize the sequences being produced and saved in a text format file. It runs on the Windows operative system.



RANDNA random sequence generator

POLYTECHNIC UNIVERSITY OF MARCHE
Institute of Biology and Genetic, Ancona
ITALY

Desired percent of 'a' nucleotide: 25

Desired percent of 'c' nucleotide: 25

Desired percent of 'g' nucleotide: 25

Desired percent of 't' nucleotide: 25

☐ U (RNA) instead of T (DNA)

Desired length of random sequence: 100

Desired number of sequences in the output file: 1

Background

Efficiency (or information density) in a language is the ability to transfer or memorize information using the smallest possible number of symbols, whereas redundancy is the loss of efficiency caused by the presence of correlations and different frequencies of symbols or words [1]. According to the information theory, the more erratic the succession of symbols of a language, the greater its efficiency but the language is less robust in terms of the ability to preserve/transfer information in the presence of noise. Natural languages tend to reach a balance between efficiency and robustness; redundancy is therefore a characteristic of natural languages.

The application of information theory to genomic sequences can reveal regularities, that is, the presence of correlations among nucleotides and different frequencies of nucleotides or motifs. To estabilish the presence of such regularities it is important to infer their function, understand the language that specifies them and set up experimental investigations.
In particular the regularities are common in the protein coding regions of eukaryotic genes because they are highly constrained by the presence of at least two languages, one specifying the amino acid by defining the codon and the second regulating the splicing process by defining some codons among their synonyms; this contributes to the formation of enhancer or silencer regulatory elements which allow exons to be recognized as constitutive or alternative [2]. The two languages are able to coexist because the genetic code is degenerate and the splicing language can use bases that are not constrained by the first language.
Coding sequences seem to be overloaded with functions whereas the opposite is true of introns, as they contain few splicing signals while the rest have a weak regulatory role. Moreover, since in higher eukaryotes introns are numerous and often quite large, they are probably less crowded with information than exons, thus obviating the need for overlapping languages in the same nucleotides. For these reasons, the succession of nucleotides is more erratic in introns than in exons. Thus, if intronic sequences were totally erratic, they would indicate the absence of a language and therefore of a function.
The issue is that to exclude the presence of redundancy in sequences, a test should have to check infinite possibilities of linkages among nucleotides and words, an extremely time-consuming procedure. Nevertheless even simple tests can reveal the presence of correlations [3] in sequences but their significance has to be shown.
The Monte Carlo simulation can be used for this aim but it works well if true random sequences are used.

To generate good random sequences is also useful when the score of an alignment has to be evaluated. Indeed aligning random sequences it is possible to estimate a threshold below which the score of the alingment is due to chance.

Only natural physical phenomena like radioactive decay or the arrival of cosmic rays in a detector, represent perfect random number generators.
Artificial random number generators are produced by an algorithm that implements a recursive formula initialized by a random sequence called ‘seed’. Since this function is deterministic, these generators are called ‘pseudo random number generators’ and they do not have the maximum entropy and are periodic. A good algorithm has to have a large amount of seeds from which to start, a good entropy and a very high rollover time, that is, a large period.

We have developed a free software that generates pseudo random DNA sequences using pseudo random generator subroutine of Borland Delphi 6. Is not clear if such an algorithm uses an Intel processor internal function or the time and date of the computer clock to generate the seed, the former method should secure a good randomness but the quality depends on the specific processor [4], the latter could secure nearly the same quality. In Delphi 6 the seed variable is 32 bit long so there are 2 ^32 different seeds and the period of the generator is 2^32 numbers. In the program code we have called ‘Randomize’ function immediatly at the beginning of the instructions so the time that user spends to set the parameters is a further source of randomness.

We have checked the quality of pseudo random number generator of Delphi 6, by means of well-known tests, both by itself and versus a sequence generated by radioactive decas processes, so has to have a genuine random sequence [http://www.fourmilab.ch/hotbits/].
We have used the following test packages : ENT [http://www.fourmilab.ch/random/], DIEHARD [http://www.stat.fsu.edu/pub/diehard/] and RNGTEST [http://www.gapoptic.unige.ch/Prototypes/QRNG/].
Generator used in our software seems very good and we show the results of the tests at www.introni.it/en/software/.

Tests results

Type of test: ENT

The ENT test evaluates a random sequence by means five parameters:

1) Entropy: it is the information density of the sequence, expressed as a number of bits per character. In general higher this value better is the randomness. Sequences we used for the tests were ASCII format, that is, 8 bits per character so the entropy should be as near as possible to 8.
2) The chi-square test gives an absolute number and a percentage which indicates how frequently a truly random sequence would exceed the value calculated. We interpret the percentage as the degree to which the sequence tested is suspected of being non-random.
3) The Arithmetic Mean is the result of summing the all the bytes in the sequence and dividing by the sequence. If the data are close to random, this should be about 127.5
4) Monte Carlo Value for Pi: each successive sequence of six bytes is used as 24 bit X and Y co-ordinates within a square. If the distance of the randomly-generated point is less than the radius of a circle inscribed within the square, the six-byte sequence is considered a "hit". The percentage of hits can be used to calculate the value of Pi. For very large streams the value will approach the correct value of Pi if the sequence is close to random.
5) Serial Correlation Coefficient: this quantity measures the extent to which each byte in the file depends upon the previous byte. For random sequences this value will be close to zero.

Borland Delphi 6 pseudo random number generator:
  Entropy = 7.964234 bits per character.
  Optimum compression would reduce the size of this 9456334 character file by 0 percent.
  Chi square distribution for 9456334 samples is 491674.30, and randomly would exceed this value 0.01 percent of the times.
  Arithmetic mean value of data bytes is 126.5785 (127.5 = random).
  Monte Carlo value for Pi is 3.157229919 (error 0.50 percent).
  Serial correlation coefficient is 0.029911 (totally uncorrelated = 0.0).

Real random number generator (radioactive decay):
  Entropy = 7.963288 bits per character.
  Optimum compression would reduce the size of this 596866 character file by 0 percent.
  Chi square distribution for 596866 samples is 32474.83, and randomly would exceed this value 0.01 percent of the times.
  Arithmetic mean value of data bytes is 126.4953 (127.5 = random).

Monte Carlo value for Pi is 3.159644943 (error 0.43 percent).
Serial correlation coefficient is 0.030643 (totally uncorrelated = 0.0).

Type of test: RNGTEST

for Delphi pseudo random number generator

Autocorrelation Test

The autocorrelation test calcul the normalized correlation function Γ(n).

```
  n   Γ(n)        p-value
---------------------------
  1   0.50189991  1.000
  2   0.49989772  0.275
  3   0.49938911  0.000
  4   0.50069378  1.000
  5   0.50028326  0.951
  6   0.49765173  0.000
  7   0.49706887  0.000
  8   0.49820938  0.000
  9   0.49597694  0.000
 10   0.50179553  1.000
```

Kolmogorov-Smirnov Test on the 10 values

```
D value of the K-S test:    0.500
Probability (D>observed):    1.352%
```

The sequence of bits seems to be random because the probability that D is greater than the expected D is not too close from 0 or 1.

One Level Entropy Test

```
Lenght of Blocks:     4
Nb of Blocks:         2141154

Entropie H:         -1841.339
Expected value:        15.000
Standard deviation σ:   5.477
```

There is too much difference between the expected and the claclutated values of H. The sequences has not be symetrised, or it is not random.

and for Real random numbers

Autocorrelation Test

The autocorrelation test calcul the normalized correlation function Γ(n).

```
  n   Γ(n)        p-value
---------------------------
  1   0.50219373  1.000
  2   0.50001908  0.544
  3   0.49918106  0.000
  4   0.50078243  1.000
  5   0.50010316  0.726
  6   0.49741317  0.000
  7   0.49737078  0.000
  8   0.49873850  0.000
  9   0.49620726  0.000
 10   0.50152082  1.000
```

Kolmogorov-Smirnov Test on the 10 values

```
D value of the K-S test:    0.500
Probability (D>observed):    1.348%
```

The sequence of bits seems to be random because the probability that D is greater than the expected D is not too close from 0 or 1.

One Level Entropy Test

```
Lenght of Blocks:     4
Nb of Blocks:         2122870

Entropie H:         -1761.047
Expected value:        15.000
Standard deviation σ:   5.477
```

There is too much difference between the expected and the claclutated values of H. The sequences has not be symetrised, or it is not random.

Maurer Universal Test

```
 L      Q         K       fTU mean
stand. dev. σ     fTU    Nb. of σ
-------------------------------------
---------------------------------
 1      100    8564516    0.7342
undefined      0.7326   undefined
 2      200    4282108    1.5373
1.718e-004     1.5374    0.5352
 3      400    2854472    2.4020
4.266e-004     2.4016    0.8515
 4      800    2140354    3.3078
5.792e-004     3.3112    5.8713
 5     1600    1711323    4.2552
7.218e-004     4.2534    2.3939
 6     3200    1424236    5.2168
8.508e-004     5.2177    1.0744
 7     6400    1217116    6.1948
9.658e-004     6.1963    1.5532
 8    12800    1057777    7.1489
1.069e-003     7.1837   32.5578
 9    25600     926024    8.1701
1.166e-003     8.1764    5.4041
```

The value of the column "Nb. of σ" must not be greater than 3 because it give the number of standard deviation from the mean.

Run Test

Runs of 0

```
Lenght     number
-----------------
   1     1073921
   2      533421
   3      266312
   4      133055
   5       79737
   6       33585
   7       16740
   8        7309
   9        3094
  10        1305
  11         543
  12         178
  13          73
  14          17
  15           0
>= 16          0
```

Maurer Universal Test

```
 L      Q         K       fTU mean
stand. dev. σ     fTU    Nb. of σ
-------------------------------------
---------------------------------
 1      100    8491380    0.7345
undefined      0.7326   undefined
 2      200    4245540    1.5377
1.726e-004     1.5374    1.6438
 3      400    2830093    2.4024
4.285e-004     2.4016    1.8707
 4      800    2122070    3.3086
5.817e-004     3.3112    4.5606
 5     1600    1696696    4.2560
7.249e-004     4.2534    3.6070
 6     3200    1412046    5.2172
8.545e-004     5.2177    0.5638
 7     6400    1206668    6.1934
9.699e-004     6.1963    2.9021
 8    12800    1048635    7.1493
1.074e-003     7.1837   32.0153
 9    25600     917897    8.1713
1.171e-003     8.1764    4.4103
```

The value of the column "Nb. of σ" must not be greater than 3 because it give the number of standard deviation from the mean.

Run Test

Runs of 0

```
Lenght     number
-----------------
   1     1065337
   2      530464
   3      262997
   4      131902
   5       79209
   6       33095
   7       16685
   8        7251
   9        3109
  10        1309
  11         534
  12         212
  13          69
  14          12
  15           0
>= 16          0
```

Runs of 1

```
Lenght    number
-----------------
   1    1083945
   2     539700
   3     262455
   4     131419
   5      66017
   6      32929
   7      16466
   8       8128
   9       4047
  10       2084
  11       1087
  12        523
  13        259
  14        113
  15         67
>= 16        52
```

Comparaison with the χ^2 distribution with 2L degrees of freedom

```
p-value:  0.00%
The sequence of bits is not random.
The sequence has not been balanced or
it is not random.
```

One level Serial Test

```
Lenght of Blocks:    4
Nb of Blocks:        2141154

V value: 1855.634
p-value:    0.000%
The p-value is too close from 0 or 1.
The sequences has not been balanced
or it is not random.
```

One Level Frequency Test

```
Number of zeros: 4297958 (50.18%)
Number of zeros: 4266658 (49.82%)
Number of σ from 50/50 distribution:
-10.695236
There is too much difference between
the expected and the fequency of 0
and 1. The sequences has not been
balanced or it is not random.
```

Runs of 1

```
Lenght    number
-----------------
   1    1076081
   2     535100
   3     260790
   4     130020
   5      65418
   6      32470
   7      16281
   8       8038
   9       4087
  10       1944
  11        963
  12        498
  13        250
  14        113
  15         65
>= 16        66
```

Comparaison with the χ^2 distribution with 2L degrees of freedom

```
p-value:  0.00%
The sequence of bits is not random.
The sequence has not been balanced or
it is not random.
```

One level Serial Test

```
Lenght of Blocks:    4
Nb of Blocks:        2122870

V value: 1776.431
p-value:    0.000%
The p-value is too close from 0 or 1.
The sequences has not been balanced
or it is not random.
```

One Level Frequency Test

```
Number of zeros: 4262836 (50.20%)
Number of zeros: 4228644 (49.80%)
Number of σ from 50/50 distribution:
-11.733641
There is too much difference between
the expected and the fequency of 0
and 1. The sequences has not been
balanced or it is not random.
```

DIEHARD TEST

Most of the tests in DIEHARD return a p-value, which should be uniform on [0,1) if the input file contains truly independent random bits. Those p-values are obtained by p=F(X), where F is the assumed distribution of the sample random variable X---often normal. But that assumed F is just an asymptotic approximation, for which the fit will be worst in the tails. Thus you should not be surprised with occasional p-values near 0 or 1, such as .0012 or .9983. When a bit stream really FAILS BIG, you will get p's of 0 or 1 to six or more places. By all means, do not, as a Statistician might, think that a p < .025 or p> .975 means that the RNG has "failed the test at the .05 level". Such p's happen among the hundreds that DIEHARD produces, even with good RNG's. So keep in mind that " p happens".

This is the BIRTHDAY SPACINGS TEST Choose m birthdays in a year of n days.List the spacingsbetween the birthdays.If j is the number of values that occur more than once in that list, then j is asymptotically Poisson distributed with mean m^3/(4n).Experience shows n must be quite large, say n>=2^18, for comparing the results to the Poisson distribution with that mean.This test uses n=2^24 and m=2^9,so that the underlying distribution for jis taken to be Poisson with lambda=2^27/(2^26)=2.A sample of 500 j's is taken, and a chi-square goodness of fit testprovides a p value.The first test uses bits 1-24 (countingfrom the left) from integers in the specified file. Then the file is closed and reopened. Next, bits 2-25 are used to provide birthdays, then 3-26 and so on to bits 9-32.Each set of bits provides a p-value, and the nine p-valuesprovide a sample for a KSTEST.

Test for Delphi pseudo random number generator

```
BIRTHDAY SPACINGS TEST, M= 512
N=2**24 LAMBDA=  2.0000
Results for delphi.txt
For a sample of size 500:     mean
delphi.txt     using bits  1 to 24   2.224
  duplicate       number       number
  spacings      observed     expected
       0          49.       67.668
       1         129.       135.335
       2         136.       135.335
       3          94.       90.224
       4          53.       45.112
       5          24.       18.045
  6 to INF      15.        8.282
 Chisquare with  6 d.o.f. =    14.40 p-value=
.974549
```

Test for real random numbers

```
BIRTHDAY SPACINGS TEST, M= 512
N=2**24 LAMBDA=  2.0000
Results for realrand.bin
For a sample of size 500:     mean
realrand.bin     using bits  1 to 24   2.312
  duplicate       number       number
  spacings      observed     expected
       0          56.       67.668
       1         107.       135.335
       2         135.       135.335
       3          98.       90.224
       4          60.       45.112
       5          26.       18.045
  6 to INF      18.        8.282
 Chisquare with  6 d.o.f. =    28.44 p-value=
.999922
```

```
For a sample of size 500:     mean
delphi.txt      using bits  2 to 25   2.194
 duplicate       number       number
 spacings       observed     expected
      0          58.       67.668
      1         105.       135.335
      2         156.       135.335
      3          98.       90.224
      4          49.       45.112
      5          21.       18.045
 6 to INF        13.        8.282
 Chisquare with  6 d.o.f. =    15.51 p-value=
.983383
```

```
For a sample of size 500:     mean
realrand.bin    using bits  2 to 25   2.194
 duplicate       number       number
 spacings       observed     expected
      0          43.       67.668
      1         122.       135.335
      2         146.       135.335
      3         112.        90.224
      4          48.       45.112
      5          21.       18.045
 6 to INF         8.        8.282
 Chisquare with  6 d.o.f. =    17.08 p-value=
.991011
```

```
For a sample of size 500:     mean
delphi.txt      using bits  3 to 26   2.174
 duplicate       number       number
 spacings       observed     expected
      0          49.       67.668
      1         120.       135.335
      2         149.       135.335
      3         108.        90.224
      4          42.       45.112
      5          22.       18.045
 6 to INF        10.        8.282
 Chisquare with  6 d.o.f. =    13.21 p-value=
.960148
```

```
For a sample of size 500:     mean
realrand.bin    using bits  3 to 26   2.086
 duplicate       number       number
 spacings       observed     expected
      0          66.       67.668
      1         115.       135.335
      2         144.       135.335
      3          94.       90.224
      4          54.       45.112
      5          22.       18.045
 6 to INF         5.        8.282
 Chisquare with  6 d.o.f. =     7.73 p-value=
.741296
```

```
For a sample of size 500:     mean
delphi.txt      using bits  4 to 27   2.032
 duplicate       number       number
 spacings       observed     expected
      0          61.       67.668
      1         126.       135.335
      2         152.       135.335
      3          91.       90.224
      4          44.       45.112
      5          20.       18.045
 6 to INF         6.        8.282
 Chisquare with  6 d.o.f. =     4.23 p-value=
.354095
```

```
For a sample of size 500:     mean
realrand.bin    using bits  4 to 27   2.094
 duplicate       number       number
 spacings       observed     expected
      0          62.       67.668
      1         126.       135.335
      2         133.       135.335
      3          97.       90.224
      4          53.       45.112
      5          24.       18.045
 6 to INF         5.        8.282
 Chisquare with  6 d.o.f. =     6.31 p-value=
.610959
```

```
For a sample of size 500:     mean
delphi.txt      using bits  5 to 28   1.992
 duplicate       number       number
 spacings       observed     expected
      0          86.       67.668
      1         127.       135.335
      2         114.       135.335
      3          84.       90.224
      4          63.       45.112
      5          20.       18.045
 6 to INF        6.         8.282
 Chisquare with  6 d.o.f. =    17.21 p-value=
.991446
```

```
For a sample of size 500:     mean
delphi.txt      using bits  6 to 29   2.076
 duplicate       number       number
 spacings       observed     expected
      0          67.       67.668
      1         130.       135.335
      2         132.       135.335
      3          78.       90.224
      4          65.       45.112
      5          20.       18.045
 6 to INF        8.         8.282
 Chisquare with  6 d.o.f. =    10.94 p-value=
.909899
```

```
For a sample of size 500:     mean
delphi.txt      using bits  7 to 30   2.262
 duplicate       number       number
 spacings       observed     expected
      0          47.       67.668
      1         115.       135.335
      2         151.       135.335
      3          89.       90.224
      4          58.       45.112
      5          31.       18.045
 6 to INF        9.         8.282
 Chisquare with  6 d.o.f. =    24.24 p-value=
.999529
```

```
For a sample of size 500:     mean
realrand.bin    using bits  5 to 28   2.134
 duplicate       number       number
 spacings       observed     expected
      0          54.       67.668
      1         131.       135.335
      2         137.       135.335
      3          95.       90.224
      4          51.       45.112
      5          21.       18.045
 6 to INF       11.         8.282
 Chisquare with  6 d.o.f. =     5.32 p-value=
.496224
```

```
For a sample of size 500:     mean
realrand.bin    using bits  6 to 29   2.068
 duplicate       number       number
 spacings       observed     expected
      0          51.       67.668
      1         146.       135.335
      2         131.       135.335
      3         101.        90.224
      4          42.       45.112
      5          20.       18.045
 6 to INF        9.         8.282
 Chisquare with  6 d.o.f. =     6.86 p-value=
.666083
```

```
For a sample of size 500:     mean
realrand.bin    using bits  7 to 30   2.218
 duplicate       number       number
 spacings       observed     expected
      0          52.       67.668
      1         129.       135.335
      2         127.       135.335
      3         100.        90.224
      4          49.       45.112
      5          32.       18.045
 6 to INF       11.         8.282
 Chisquare with  6 d.o.f. =    17.52 p-value=
.992440
```

```
For a sample of size 500:     mean
delphi.txt      using bits  8 to 31   2.176
 duplicate       number       number
 spacings       observed     expected
      0          53.       67.668
      1         127.       135.335
      2         139.       135.335
      3          85.       90.224
      4          62.       45.112
      5          28.       18.045
 6 to INF        6.         8.282
 Chisquare with  6 d.o.f. =    16.54 p-value=
.988859

For a sample of size 500:     mean
delphi.txt      using bits  9 to 32   2.250
 duplicate       number       number
 spacings       observed     expected
      0          50.       67.668
      1         115.       135.335
      2         137.       135.335
      3         104.        90.224
      4          66.       45.112
      5          17.       18.045
 6 to INF       11.         8.282
 Chisquare with  6 d.o.f. =    20.42 p-value=
.997667

   The 9 p-values were
.974549   .983383   .960148   .354095
.991446   .909899   .999529   .988859
.997667

A KSTEST for the 9 p-values yields 1.000000
```

```
For a sample of size 500:     mean
realrand.bin     using bits  8 to 31   2.278
 duplicate       number       number
 spacings       observed     expected
      0          49.       67.668
      1         115.       135.335
      2         141.       135.335
      3          96.       90.224
      4          57.       45.112
      5          30.       18.045
 6 to INF       12.         8.282
 Chisquare with  6 d.o.f. =    21.54 p-value=
.998531

For a sample of size 500:     mean
realrand.bin     using bits  9 to 32   2.290
 duplicate       number       number
 spacings       observed     expected
      0          64.       67.668
      1         105.       135.335
      2         130.       135.335
      3          96.       90.224
      4          65.       45.112
      5          18.       18.045
 6 to INF       22.         8.282
 Chisquare with  6 d.o.f. =    39.07 p-value=
.999999

   The 9 p-values were
.999922   .991011   .741296   .610959
.496224   .666083   .992440   .998531
.999999

A KSTEST for the 9 p-values yields 1.000000
```

THE 3DSPHERES TEST

Choose4000 random points in a cube of edge 1000.At each point, center a sphere large enough to reach the next closest point. Then the volume of the smallest such sphere is (very close to) exponentially distributed with mean 120pi/3.Thusthe radius cubed is exponential with mean 30. (The mean isobtained by extensive simulation).The 3DSPHERES test gener- ates 4000 such spheres 20 times.Each min radius cubed leads to a uniform variable by means of 1-exp(-r^3/30.), then aKSTEST is done on the 20 p-values.

```
The 3DSPHERES test for file delphi.txt

sample no:1   r^3=   8.022     p-value= .23464
sample no:2   r^3=  26.784     p-value= .59050
sample no:3   r^3=  20.590     p-value= .49659
sample no:4   r^3=  33.189     p-value= .66922
sample no:5   r^3=  12.174     p-value= .33356
sample no:6   r^3=   3.705     p-value= .11619
sample no:7   r^3= 116.391     p-value= .97934
sample no:8   r^3=  37.957     p-value= .71783
sample no:9   r^3=  35.460     p-value= .69333
sample no:10  r^3=  13.289     p-value= .35786
sample no:11  r^3=    .407     p-value= .01348
sample no:12  r^3=   7.040     p-value= .20918
sample no:13  r^3=  15.316     p-value= .39983
sample no:14  r^3=  34.110     p-value= .67922
sample no:15  r^3=  17.221     p-value= .43675
sample no:16  r^3=    .889     p-value= .02920
sample no:17  r^3=  22.402     p-value= .52608
sample no:18  r^3=  40.772     p-value= .74310
sample no:19  r^3=   2.790     p-value= .08882
sample no:20  r^3=  49.152     p-value= .80571

A KS test is applied to those 20 p-values.
------------------------------------------------------
3DSPHERES test for file delphi.txt
p-value= .265109
```

```
The 3DSPHERES test for file realrand.bin

sample no:1   r^3=   1.204     p-value= .03935
sample no:2   r^3=  46.825     p-value= .79004
sample no:3   r^3=   4.493     p-value= .13910
sample no:4   r^3=  97.689     p-value= .96147
sample no:5   r^3=  64.896     p-value= .88504
sample no:6   r^3=    .469     p-value= .01550
sample no:7   r^3=   1.927     p-value= .06222
sample no:8   r^3=  44.777     p-value= .77521
sample no:9   r^3=   2.567     p-value= .08201
sample no:10  r^3=   3.135     p-value= .09922
sample no:11  r^3=  27.322     p-value= .59777
sample no:12  r^3=   9.392     p-value= .26879
sample no:13  r^3=   4.255     p-value= .13222
sample no:14  r^3=   4.168     p-value= .12972
sample no:15  r^3=   1.827     p-value= .05909
sample no:16  r^3=  28.502     p-value= .61329
sample no:17  r^3=  48.746     p-value= .80306
sample no:18  r^3=  54.916     p-value= .83967
sample no:19  r^3=  74.422     p-value= .91632
sample no:20  r^3=   9.816     p-value= .27905

A KS test is applied to those 20 p-values.
------------------------------------------------------
3DSPHERES test for file realrand.bin
p-value= .943276
```

Systems

RANDNA is written in Borland Delphi v.6 and runs on ix86 compatible processors under Microsoft Windows as well as on Apple Macintosh, Linux and Unix-based platforms using Windows emulator software with one of the required Microsoft Windows versions.

Installation

RANDNA is compound from the following files:
randna.exe : Executable file
randna.chm, randna.hhp : Help files
output.txt : Example of an output text format file
randna_test.txt :output of the test of the Borland Delphi 6 pseudo random number generator

On the web there are both zip archive and separete files. To install the software you should download and uncompress the zip files into a directory. Alternatively you can download files separately and at least you should run the executable file.

Program use

The user can choose to obtain a uniform distribution of nucleotides by setting all the frequencies at 25% or have a different distribution leaving from the equiprobability. Setting the flag ‘U instead of T’ the software generates RNA instead of DNA sequences. Maximum length of the sequence being produced can be choosed up to two thousand millions of nucleotides, maximum number of sequences generated is up to 255. After processing, an output text format file is produced.

Output

Example of output file:

```
>1
ggggccgttcgtgggtacattatcaccgccacagcaataggaatagcatgccggtaatatacctag
tgatcgaaaaaggctgatataggagatcaaacactgctagcattatgacccgccgggaaagtacaa
accaagttctactaccccttacctgcagataggctcaaagctgtagtttagttctacctactatgg
ga

>2
ggtcggggtatgatatacagtgccattgtgacggcgccgtgaggcgttcggactcacccacgtaac
cgcgggagtatagctccggggtagcctaatcgcgcgggataattcgtacttacagctgtggcgata
ctatagggaccgcacttgtatccttgcgctttagacacggcagaaaagtaaaattcacgccctata
cg

>3
aaatacatccctggaaccccctaagggagacggggaaccccgaggaaagtaaacgatgcttcctaaa
gttgaggcacatccagtcgatgctgggattttgcagggagtgcgttgacactaagctcacacacga
gtggaccgagtacgaacactattcggtgtacttaattaaccgtttctctttacattcgagtaactg
tc

>4
ttgggtatgattgggtctaagcagggatcataaagccttgcgtataagacgcccccgaaaaccgga
acttaccgctagaccacagttaccaattgaccttcgctgtgtagattacttcacatatttgccccc
gccgcagacgtaatcatcgtctttcctatgtagttatagggggttgtcccgtgctgcagtggttgcg
cg

>5
acatgtcggtgtgaagggaggtgtacaggtgatgcagacgtcttgcaccggccagctgtccgcttc
tcaaggcggtttaccggatatctcaatcataaataacgtacttccaaacgacacaggcgaatcaga
aagaggcttctctgcagatgaagagattctttatactgggattaaaggagacttctcaggtgatct
tg
```

Glossary

Correlation: Link between nucleotides. There is a correlation between nucleotides when the presence of a particular base (implier) implies or promotes the presence of another particular (implied) base in a different position. If the distance between correlated nucleotides is relatively short (about less than 10), they can have a functional role, for example be involved in a binding site. Long distance correlations in transcription sequences can be involved in mRNA secondary structure.

Information Theory: Efficiency in a language is the ability to transfer or memorize information using the smallest possible number of symbols, whereas redundancy is the loss of efficiency caused by the presence of correlations and different frequencies of symbols or words. According to the information theory, the more chaotic the succession of symbols of a language, the greater its efficiency but the less robust the language in terms of the ability to preserve/transfer information in the presence of noise. Natural languages tend to reach a balance between efficiency and robustness; redundancy is therefore a characteristic of natural languages.

Language: The protein coding regions of genes are highly constrained by the presence of at least two languages, one specifying the amino acid by defining the codon (coding function) and the second regulating the splicing process by defining some codons among their synonyms; this contributes to the formation of enhancer or silencer regulatory elements which allow exons to be recognized as constitutive or alternative. The two languages are able to coexist because the genetic code is degenerate and the splicing language can use bases that are not constrained by the first language. Since not all exonic synonymous mutations affect splicing efficiency, the splicing language obviously does not specify all synonymous codons and can drive splicing by creating alternative signals with equivalent function. Consequently, other constraints or languages can be present in gene coding sequences. In fact, there is evidence that synonymous codon usage is partly constrained by the isochore composition of the region in which the gene lies. Some codons create motifs to bind proteins involved in transcription, in the export of mRNA towards the cytoplasm, and in translation. In addition, coding sequences seem to be overloaded with functions whereas the opposite is true of introns, as they contain few splicing signals while the rest have a weak regulatory role.

References

1. Shannon, CE. (1948) A Mathematical Theory of Communication. The Bell System Technical Journal, **27**: 379–423, 623–656

2. Pagani, F., Raponi, M., and Baralle, FE. (2005) Synonymous mutations in CFTR exon 12 affect splicing and are not neutral in evolution. Proc Natl Acad Sci U S A. **102**:6368-6372.

3. Piva, F. and Principato, G. (2005) CORRELATION FINDER. In Silico Biol. **5**:0042

4. Huang, F., and Shen, H. (2004) Intel random number generator-based true random number generator. Di Yi Jun Yi Da Xue Xue Bao. **24**:1091-1095

Authors:

Francesco Piva, Giovanni Principato
Institute of Biology and Genetics
Polytechnic University of Marche
Via Brecce Bianche, Monte D'Ago
60131 Ancona
ITALY
f.piva@univpm.it

Acknowledgments:

We would like to thank Matteo Giulietti for the help in collecting the genuine random sequences.