Статистические тесты

Основная задача состоит в том, чтобы получить последовательность, которая похожа на случайную.

Если попросить человека написать 100 случайных десятичных цифр, очень мало шансов на то, что он достаточно хорошо может с этим справиться. Люди стремятся избегать комбинаций, кажущихся им неслучайными, таких, как пары одинаковых соседних цифр (хотя примерно каждая из 10 цифр должна совпадать с предыдущей). Поэтому, увидев таблицу действительно случайных чисел, любой человек скорее всего скажет, что они не совсем не случайные, его глаз сразу же отметит некоторые видимые закономерности.

Главная мысль сказанного заключается в том, что мы не можем доверять себе в оценке, случайна или нет данная последовательность чисел. Необходимо использовать какие-то непредвзятые механические тесты.

Статистическая теория даёт нам некоторые количественные критерии случайности.

Рассмотрим один из эмпирических тестов, который применяется для проверки равномерности (проверки частот) целочисленной последовательности. В этом случае проверке подвергается следующая последовательность:

$\langle Y_n \rangle = Y_0, Y_1, Y_2, ...,$

где n — длина последовательности (размер выборки)

Входящие в эту последовательность целые числа распределены равномерно между 0 и $d-1$. Значение d может быть любым. Чтобы тест был представительным, значение d должно быть достаточно большим, но не слишком, чтобы на вычислительной машине это не возникало затруднений во временных рамках. Для каждого целого r, $0 \le r < d$, подсчитать число значений $Y_j = r$ при $0 \le j < n$ и применить критерий χ^2 с числом категорий, равным $k = d$ и вероятностями $p_s = \frac{1}{d}$ — .

Пусть все возможные испытания разделены на k категорий, то при проведении n независимых испытаний исход каждого испытания абсолютно не влияет на исход остальных. Пусть p_s — вероятность того, что результат испытания попадёт в категорию s, и пусть Y_s — число испытаний, тогда можно определить величину V, которая называется статистикой χ^2, соответствующей значениям $Y_1, Y_2, ..., Y_k$, полученным в эксперименте:

$$V = \sum_{1 \le s \le k} \frac{\left(Y_s - np_s\right)^2}{np_s}.$$

Используя тождество $\left(Y_s - np_s\right)^2 = Y_s^2 - 2np_sY_s + n^2p_s^2$ и равенства

$Y_1 + Y_2 + ... + Y_k = n$

$p_1 + p_2 + ... + p_k = 1$

предыдущую формулу для V можно преобразовать к виду

$$V = \frac{1}{n} \sum_{1 \le s \le k} \left(\frac{Y_s^2}{p_s} \right) - n,$$

что в большинстве случаев облегчает вычисления.

Как по значению V определить: является ли равномерной рассматриваемая целочисленная последовательность?

Ответ на этот вопрос даёт таблица, в которой приведено «распределение χ^2 с ν степенями свободы» при разных значениях ν. Следует пользоваться строкой таблицы с $\nu = k-1$; *число «степеней свободы» равно $k-1$, то есть на единицу меньше числа категорий.*

	$p=99\%$	$p=95\%$	$p=75\%$	$p=50\%$	$p=25\%$	$p=5\%$	$p=1\%$
$\nu=1$	0.00016	0.00393	0.1015	0.4549	1.323	3.841	6.635
$\nu=2$	0.00201	0.1026	0.5753	1.386	2.773	5.991	9.210
$\nu=3$	0.1148	0.3518	1.213	2.366	4.108	7.815	11, 34
$\nu=4$	0.2971	0.7107	1.923	3.357	5.385	9.488	13.28
$\nu=5$	0.5543	1.1455	2.675	4.351	6.626	11.07	15.09
$\nu=6$	0.8720	1.635	3.455	5.348	7.841	12.59	16.81
$\nu=7$	1.239	2.167	4.255	6.346	9.037	14.07	18.48
$\nu=8$	1.646	2.733	5.071	7.344	10.22	15.51	20.09
$\nu=9$	2.088	3.325	5.899	8.343	11.39	16.92	21.67
$\nu=10$	2.558	3.940	6.737	9.342	12.55	18.31	23.21
$\nu=11$	3.053	4.575	7.584	10.34	13.70	19.68	24.73
$\nu=12$	3.571	5.226	8.438	11.34	14.84	21.03	26.22
$\nu=15$	5.229	7.261	11.04	14.34	18.25	25.00	30.58
$\nu=20$	8.260	10.85	15.45	19.34	23.83	31.41	37.57
$\nu=30$	14.95	18.49	24.48	29.34	34.80	43.77	50.89
$\nu=50$	29.71	34.76	42.94	49.33	56.33	67.50	76.15
$\nu>30$	*приблизительно* $\nu+2\sqrt{\nu}x_p+\frac{4}{3}x_p^2-\frac{2}{3}$						
$x_p=$	-2.33	-1.64	$-.675$	0.00	0.675	1.64	2.33

Если в таблице в строке ν и колонке p находится число x, то это означает, что значение V, определяемое по формуле, будет больше x с вероятностью p.

Проверка с помощью критерия χ^2 заключается в следующем. Проводится n независимых испытаний, где n — достаточно большое число. Подсчитывается число испытаний, результат которых относится к каждой из k категорий, и по формулам вычисляется значение V. Затем V сравнивается с числами из таблицы при $\nu = k-1$. Если V меньше значения, соответствующего $p=99\%$, или больше значения, соответствующего $p=1\%$, то результаты бракуются как недостаточно случайные. Если p лежит между 99 и 95% или между 5 и 1%, то результаты считаются «подозрительными»; при значениях p, полученных интерполяцией по таблице, заключённых между 95 и 90% или 10 и 5%, результаты «слегка подозрительны». Часто с помощью критерия χ^2 проверяют по крайней мере три раза разные части исследуемого ряда чисел, и, если не менее двух раз из трёх результаты оказываются подозрительными, числа отбрасываются как недостаточно случайные. (для проверки равномерности: как недостаточно равномерные).

Доверительные интервалы находятся по следующим формулам

$$p_1 = p - \Phi^{-1}\left(P_{\text{дов}}\right)\sqrt{\frac{p(1-p)}{n}}, \qquad p_2 = p + \Phi^{-1}\left(P_{\text{дов}}\right)\sqrt{\frac{p(1-p)}{n}},$$

где доверительная вероятность $P_{\text{дов}} = 0{,}95$ или $P_{\text{дов}} = 0{,}99$, число испытаний: n = 100000, p – рассчитанная вероятность ошибки, $\Phi^{-1}(x)$ – функция, обратная функции Крампа

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-x}^{x} \mathrm{e}^{-t^2/2}dt.$$

Сделать вывод о соответствии результатов моделирования теоретическим расчётам.

Для нахождения значений функций $Q(x)$ и $\Phi(x)$ воспользуйтесь справочником [4, стр. 76–77], где табулированы значения $\Phi_0(x)$. При этом

$$\Phi_0(x) = \frac{1}{\sqrt{2\pi}}\int_{0}^{x} e^{-t^2/2}dt = \frac{1}{2}\Phi(x) = \frac{1}{2} - Q(x).$$

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}}\int_{0}^{x} e^{-t^2}dx = \Phi\left(\sqrt{x}\right)$$

7.1.26. $\operatorname{erf} x = 1 - (a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5)\, e^{-x^2} + \varepsilon(x)$,

$$t = \frac{1}{1 + px}, \quad |\varepsilon(x)| \leqslant 1.5 \cdot 10^{-7},$$

$p = 0.32759\ 11$, $a_1 = 0.25482\ 9592$,

$a_2 = -0.28449\ 6736$, $a_3 = 1.42141\ 3741$,

$a_4 = -1.45315\ 2027$, $a_5 = 1.06140\ 5429$.

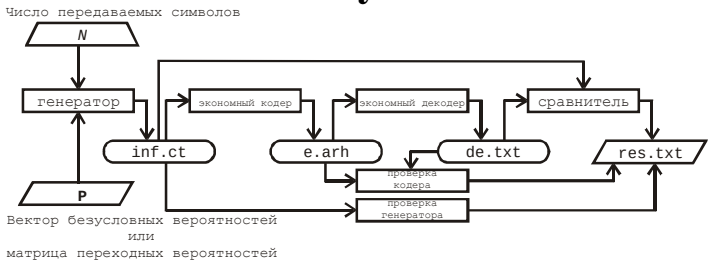
```
#include<fstream.h>
#include<stdlib.h>
unsigned long huge R[32768];
void main(int mn,char* nm[])
{ unsigned long i,N;
double r,xi2;
float f;
if(mn!=3) cerr<<"test_inx.exe in.fb razb\n",exit(1);
ifstream in(nm[1],ios::binary);
if(!in) cerr<<"file \""<<nm[1]<<"\" not open!\n",exit(1);
in.seekg(0,ios::end);N=in.tellg()/4;in.seekg(0,ios::beg);
cout<<"N="<<N<<endl;
unsigned long M=atol(nm[2]);
cout<<"M="<<M<<endl;
for(i=0;i<32768u;R[i++]=0);
for(i=0;i<N;i++) in.read((char*)&f,4),R[int((f+0.5)/(32768./M))]++;
for(i=0;i<M;i++) cout<<R[i]<<'\t'; cout<<endl;
in.close();
r=double(N)/M; cout<<"r="<<r<<endl;
for(xi2=i=0;i<M;i++) xi2+=(R[i]-r)*(R[i]-r); xi2/=r;
cout<<"xi2="<<xi2<<endl;
}

#include<iostream>
#include<fstream>
#include<cstdlib>
#include<cstring>
#include<ctime>
using namespace std;
int main(int mn,char* nm[])
{
long i,N; char h,c; float P,r;
struct timespec z;
clock_gettime(CLOCK_REALTIME,&z);
srand(z.tv_nsec);
if(mn!=4) cerr<<"./gen inf.ct N P(0)\n",exit(1);
if(nm[1][strlen(nm[1])-1]=='t') h=0x30; else h=0;
N=atol(nm[2]); if(N<1) cerr<<"Ошибка N<0!\n",exit(2);
P=atof(nm[3]); if(P>1||P<0) cerr<<"Ошибка P(0)!=(0..1)!\n",exit(3);
r=RAND_MAX*P;
ofstream out(nm[1],ios::binary);
if(!out) cerr<<"Файл \""<<nm[1]<<"\" не открыт!\n",exit(4);
for(i=0;i<N;i++) c=(rand()>=r)+h,out.put(c);
out.close();
return 0;
}
```

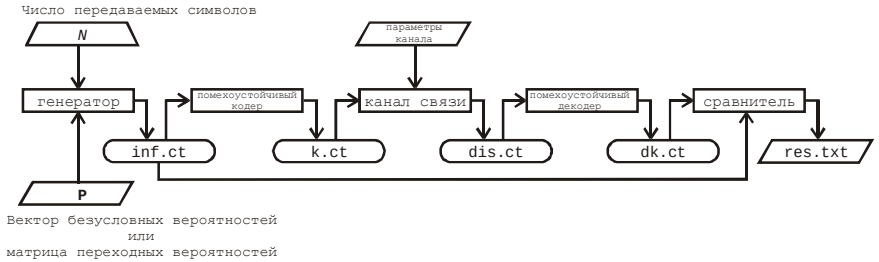
```
#include<iostream>
#include<fstream>
#include<cstdlib>
#include<cstring>
#include<cmath>
using namespace std;
long double fn_q(long double x)
{ long double sc,t; t=1/(1+.2316419*x);
sc=(.254829592+(-.284496736+(1.421413741+(-1.453152027+1.061405429*t)*t)*t)*t)*t;
return(.5*sc*exp(-x*x/2));
}
long double fn_qm1(long double y)
{ long double x0,x=0; if(y<0) y=-y;
while(y<fn_q(x+=5));
for(x0=x-5;;)
 {
 if(fabs(x0-x)<0.000001) break;
 if(y<fn_q((x+x0)/2)) x0=(x+x0)/2; else x=(x+x0)/2;
 }
return((x+x0)/2);
}
int main(int mn,char* nm[])
{ long i,N,s; char h,c; float P,Pd,d,f;
if(mn!=4) cerr<<"./test inf.ct P(0) Pдов\n",exit(1);
if(nm[1][strlen(nm[1])-1]=='t') h=0x30; else h=0;
P=atof(nm[2]); if(P>1||P<0) cerr<<"Ошибка P(0) вне диапазона (0..1)!\n",exit(2);
Pd=atof(nm[3]); if(Pd>1||Pd<0) cerr<<"Ошибка Pдов вне диапазона (0..1)!\n",exit(3);
ifstream in(nm[1],ios::binary);
if(!in) cerr<<"Файл \""<<nm[1]<<"\" не открыт!\n",exit(4);
in.seekg(0,ios::end); N=in.tellg(); in.seekg(0,ios::beg);
cout << "N=" << N;
for(s=i=0;i<N;i++) c=in.get()-h,s+=c;
cout<<"\tQ^-1="<<fn_qm1((1-Pd)/2)<<"\t";
d=fn_qm1((1-Pd)/2)*sqrt(P*(1-P)/N);
s=N-s; f=float(s)/N;
cout << P-d << " < " << f << " < " << P+d;
if(P-d<f&&f+d>P); else cout << " Тест не пройден!", cerr <<"Тест не пройден!\n";
cout << endl;
in.close();
return 0;
}
```

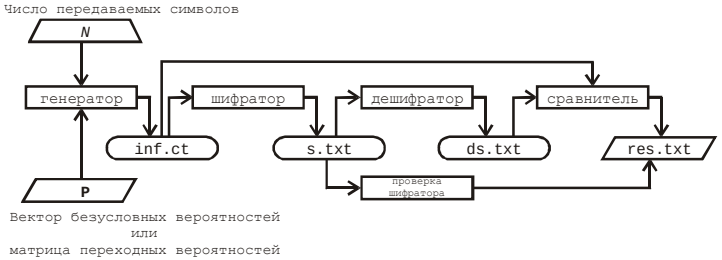
командная строка

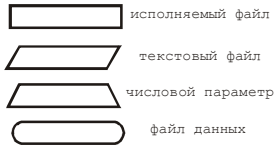
```
rm res.txt
rm err.txt
for((i=0;i<100;i++)); do ./gen i.ct 1000000 0.1; ./test i.ct 0.1 0.95 >> res.txt 2>> err.txt; done
```

Реализовать следующие схемы









Задание для генератора символов

Вектор вероятностей для генератора символов с размером алфавита M: $\mathbf{P} = \left\| P(a_0)\, P(a_1) \ldots P(a_{M-1}) \right\|$.

Матрица переходных вероятностей для генератора символов с размером алфавита M:

$$\mathbf{P} = \left\| \begin{matrix} P(a_0|a_0) & P(a_1|a_0) & \ldots & P(a_{M-1}|a_0) \\ P(a_0|a_1) & P(a_1|a_1) & \ldots & P(a_{M-1}|a_1) \\ \ldots & \ldots & \ldots & \ldots \\ P(a_0|a_{M-1}) & P(a_1|a_{M-1}) & \ldots & P(a_{M-1}|a_{M-1}) \end{matrix} \right\|, \quad \sum_{i=0}^{M-1} P(a_i|a_j) = 1, \quad \forall j = \overline{0, M-1}.$$

Размер генерируемых символов: 1 байт. Коды генерируемых символов: произвольные. Объём выборки: $\overline{1, 2^{32}-1}$.

Проверить генератор, используя выбранный статистический тест.

Задание для экономного кодирования

Экономный кодер и соответствующий ему декодер

0. Статистический алгоритм Хаффмана
1. Адаптивный алгоритм Хаффмана
2. Алгоритм Шеннона Фэно
3. Алгоритм LZ77
4. Алгоритм LZSS
5. Алгоритм LZ78
6. Алгоритм LZW
7. Адаптивный арифметический алгоритм

Номер варианта = номер бригады % 8

Вычислить энтропию для источника без памяти

Вычислить энтропию для марковского источника

Объём выборки сообщения: $N = 1000000$.

1. Синтезировать сообщение для источника без памяти (размер алфавита и вероятности задаются преподавателем).

Определить возможность сокращения сообщения.

Сжать сообщение заданным по варианту кодом. Вычислить среднюю длину сжатого сообщения.

Сжать сообщение используемым архиватором (например, 7z). Вычислить среднюю длину сжатого сообщения.

Сравнить энтропии и средние длины сообщений, а также возможность сокращения сообщения и результат сжатия.

2. Синтезировать сообщение для источника с памятью (размер алфавита и матрица переходных вероятностей задаётся преподавателем).

Определить возможность сокращения сообщения для данного источника и в предположении, что источник без памяти.

Сжать сообщение заданным по варианту кодом. Вычислить среднюю длину сжатого сообщения.

Сжать сообщение используемым архиватором (например, 7z). Вычислить среднюю длину сжатого сообщения.

Сравнить энтропии и средние длины сообщений, а также возможность сокращения сообщения и результат сжатия.

3. Показать работоспособность экономного кодера и декодера, сравнением исходного и результирующего сообщения.