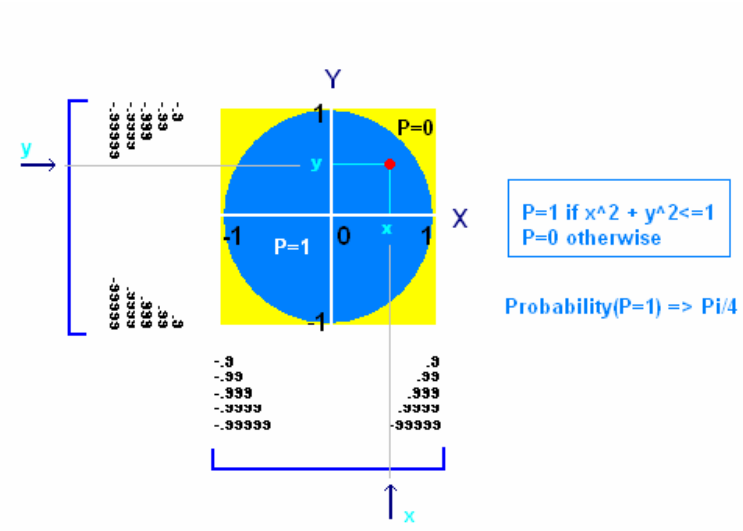
Random and Randomness

Juan Chamero, juan.chamero@intag.org, lectures
Discrete Structures & Algorithms, ds_005, as of June 2006



Monte Carlo Approach to compute the Pi number

Introduction

Most nature processes are at random. From an infinite packaged order -the Big Bang- the Universe is relentlessly going to chaos, total absence of life, of movement. However randomness in lines, planes and volumes at a certain scale, seen from “upwards” may look like ordered, neat, with well defined lines surfaces and volumes. Dead tend to make all things the same. However, following a sequence of infinite “however”, we suppose that our Universe is causal and as such everything that happens in a system, either open or closed, is supposed to be “thermodynamically” deterministic, that is nature processes would be at last “pseudo random”, with all events being “at large” (at God scale perhaps?) predictable.

Pseudo Random Numbers: Randomness could be emulated by a computer program, for instance to generate (if binary) a sequence of (2^128 -1) strings of 128 bits each, all different but “at large” cycling in a predetermined sequence, from a given “seed” to go back to the same seed after (2^128-1) steps. They are predictable in the sense that given one string you may “guess” the string m steps ahead via a mathematic transformation, that is they are pseudo random generated by a virtual machines with (2^n – 1) different states S(j) at “time-steps” j’s. All those states are predictable from a given “seed” S(0) by an expression of the following type: S(j) = (T^j).S(0) where T^j is the “j power” of a unitary transformation T0 that obtain S(j+1) from S(j).

Random numbers are used for games, operations research, simulation, scientific experimenting and for the creation of crypto keys. In these cases, by obvious reasons, numbers should be unpredictable or at least extremely difficult to predict/unveil!. Ideally crypto keys should be created by **True Random Numbers** generators. True random numbers are those generated by nature process, for instance the position of a gel particle within a Brownian movement. Most nature processes are “entropy sources” in the sense that entropy is a state function.

Each state, for instance an ice cube, meanwhile its temperature is T has certain **entropy** that depends only of the temperature, no matter how the ice cube was formed. In the other extreme of complexity we may imagine a man of today with entropy that would depends on his “state” as a man alive, no matter how he achieves this state. Leaving this ice cube in an environment at a higher temperature, let’s say 20 Centigrade degrees, it will experiment a “degradation” becoming water, dying as ice.

If we isolate the ice cube and the room where the ice cube is on a table and consider both as a system we may measure that the air around the ice will frozen a little and we note that the ice cube start to melt becoming water that slides along the table wasting some energy: thermal energy transforms in kinetic energy. When a new equilibrium state is attained many things changed, no more ice cube being the most substantial change!. Globally we intuit that the order of the Universe, even though infinitesimally, has been degraded. This degradation is measured by the increase of its entropy, the more entropy the more the disorder is. In some extent entropy is also considered the “arrow of time” because as we go forward in time the entropy of an isolated system can only increase or remains the same, never decrease!. *Entropy is in this concern the clock of the Universe.*

Our concern is particularly related to **Information Theory** where entropy means how much randomness (or uncertainty) an event has. If we have to guess a number (or any object) out of a sequence of 16 our uncertainty ⇔ “degree of ignorance” has a value of 16 if all numbers are equally probable. Now we are going to program a sort of “guessing game” with the aid of an informant that will provide us information -in the most economic way imagined- in order to reduce our ignorance gradually (we were tempted to say “bitwise”). First of all to impress the “public” we have to organize a little the scenario: the objects (letters) will be presented along a sequence in positions 0 to 15, representing pairs [position object] as depicted below:

Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Letter	z	%	4	a	h	w	c	*	#	1	&	5	9	@	+	=

Let’s program now the game supposing that we ignore the positional binary system. At most we were trained to know that a 0 means that something is in the lower half of a sequence and that a 1 means the contrary. To make things easier our game will deal with sets having 2^n objects, 2, 4, 8, 16, 32… 2^n, being n a natural number.

a) Someone of the public selects an object and informs our assistant either the object or the position chosen.
b) Our assistant must provide us the minimum but sufficient information to “guess” the chosen object, of course avoiding any unfair form.

Be the object chosen the number 11 (or the symbol &). Our assistant must provide us pieces of information to reduce our “ignorance” progressively; let’s say along a binary progression of the following type: 16 => 8 => 4 => 2 => 1. If now our assistant whisper the binary sequence [1010] in our ears to inform us that the right answer is the symbol located in position 10 in decimal we may say that we passed from an uncertainty measured by 16 to the certainty, namely from 16 to 1. The same information could be obtained progressively if previously we agreed with our assistant the procedure (as discussed above) : 1 meaning that the answer is in the upper half of the uncertainty range. So the sequence [1010] will mean:

Step 1: [1]⇔ the number is in the upper half [9 10 11 12 13 14 15 16] being 8 the level of our uncertainty after receiving the first “bit” of information.
Step2: [0] ⇔ Once received the second bit as 0, according to the agreed procedure, our uncertainty is now reduced to 4, being certain that our number will be in the lower half, among numbers [9 10 11 12];
Step 3: [1] ⇔ Our informant whisper now the third bit as a 1 and we know then that our number will be one of two, namely [11 12];
Step 4: [0] ⇔ finally our informant whisper the fourth bit that permit us to be certain that the right answer is 11!.

Shannon defined **Information Entropy** as a variable related to the probabilities of events, outcomes of a random process like for instance flipping a coin. If all the symbol outcomes are equally likely then increasing the number of symbols should the entropy increases.

Information - Formal definition

Claude E. Shannon in its Mathematical Theory of Communications (a reprint from The Bell System Technical Journal) defines entropy H(x) in terms of a discrete random event *x*, with n possible states or outcomes as:

$$H(x) = \sum_{i=1}^{n} p(i) \log_2 \left(\frac{1}{p(i)}\right) = -\sum_{i=1}^{n} p(i) \log_2 p(i).$$

Where p(i) are probabilities of outcomes i's. In our last examples n1=16, p1(i) = 1/16 for i=1, 2, 3,…, 16, if all outcomes were equally probable resulting H1=4, and if n2=2, (flipping a coin) p2(0-Head)= ½ , p2(1-Tail) = ½ , giving H1=1, being H in both cases the number of bits necessary to represent precisely the uncertainty of the guessing. Effectively with four bits we may represent 16 equally different outcomes, from 0000 to 1111 and with one bit the two flipping of coin outcomes, 0 for Head and 1 for Tail.

True Random Numbers

There are many Internet services providing True Random Numbers, using natural sources like radiation, Brownian movement, and quantum mechanics effects. Below we list some of them:

1. HotBits is a site from Switzerland at FermiLab. They said textually about themselves: *HotBits is an Internet resource that brings* genuine *random numbers, generated by a process fundamentally governed by the inherent uncertainty in the quantum mechanical laws of nature, directly to your computer in a variety of forms. HotBits are generated by timing successive pairs of radioactive decays detected by a Geiger-Müller tube interfaced to a computer.*

The HotBits generation hardware produces data at a modest rate (about 30 bytes per second). Once the random bytes are delivered, they are immediately discarded--the same data will never be sent to any other user and no records are kept of the data at this or any other site.

A really good source of entropy is a radioactive source. The points in time at which a radioactive source decays are completely unpredictable, and can be sampled and fed into a computer, avoiding any buffering mechanisms in the operating system. In fact, this is what the HotBits people at Fermilab in Switzerland are doing.

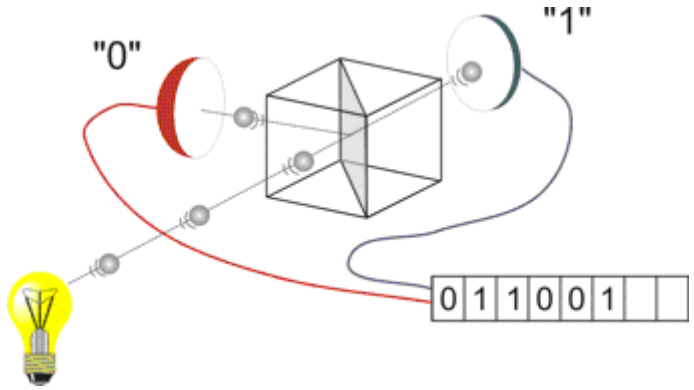
Another sources of entropy could be atmospheric noise from a radio, like that used here at random.org, or even just background noise from an office or laboratory. The lavarand people at Silicon Graphics have been clever enough to use lava lamps to generate random numbers, so their entropy source not only gives them entropy, it also looks good! The latest random number generator to come online (both lavarand and HotBits precede random.org) is Damon Hart-Davis' Java EntropyPool which gathers random bits from a variety of sources including HotBits and random.org, but also from web page hits received by the EntropyPool's web server.

2. The LavaRND people at Silicon Graphics maintain a site that provides random numbers from lava lamps (kidding!). They proudly announce that LavaRND is a cryptographically sound random number generator. At its heart, they use not lava but some small thermal noise sources like some chips easy to find in Web cams like CCD.

3. Random.org. They use atmospheric noise to generate random numbers. They proudly announce themselves as the only service which offers a large (16K) block of numbers at once. They tuned a radio into a frequency where nobody is transmitting.

The atmospheric noise picked up by the receiver is fed into a workstation through the microphone port where it is sampled by a program at a given rate and the upper seven bits of each sample are discarded. The remaining bits are turned into a stream of bits with supposedly has a high entropy value. **Skew Correction** is performed on the bit stream, in order to ensure that there is an approximately even distribution of 0s and 1s.

4. ID Quantique. Photons - light particles - are sent one by one onto a semi-transparent mirror and detected. The exclusive events (reflection - transmission) are associated to "0" - "1" bit values. The operation of Quantis is continuously monitored to ensure immediate detection of a failure and disabling of the random bit stream. A High bit rate of 4Mbits/sec (up to 16Mbits/sec for PCI card) could be obtained.



The Quantum random bits generation process. See its justification

.

Random Numbers Products and Manufacturers

The Marsaglia CD-ROM

One of the best sources of random numbers is the George Marsaglia CD-ROM containing 600 megabytes of random numbers. These were produced by many sources (for example rap music). Some suspected key ideas were used on it such as “if we have two collections of all possible bytes intentionally unordered (as octets) XOR them may kill all possible remaining order”. Marsaglia used three hardware sources identified by region origin: Canada, Germany and California.

Note: Some randomness gurus have found serious failures to this generator. One source of error could be in programs that manage data because some tricky details. For instance one of these experts found that Canadian and German generators failed when tested and inspecting the series they discover the following pattern: each byte containing a 10 was preceded by a byte containing a 13!. What could have happened?. One explanation was: when you write a program to write bytes to disk, you must open it as binary. If not the program thinks you are trying to write an end-of-line and the convention on a PC is to represent this by a carriage return (13) followed by a line feed (10).

Here are some manufacturers’ web sites:

- Colorado http://www.comscire.com/
- Holland http://valley.interact.nl/av/com/orion/home.html
- Canada http://www.tundra.com/ (find the data encryption section, then look under RBG1210. My device is an NM810 which is 2?8? RBG1210s on a PC card)
- Protego from Sweden, http://www.protego.se
- ID Quantique, http://www.idquantique.com/products/quantis.htm

Skew Correction

Recommended source: Request for Comments, RFC 1750

We are going to describe here a De-Skew technique, originally due to von Neumann. Suppose the probability of getting a 0 is equal to ½ plus a given error e and that all bits are independent. To eliminate skew, John von Neumann suggested to following algorithm:

- Read the bits two at a time.
- Skip 00's and 11's pairs.
- For 01 and 10 outputs the first bit.

Meaning to exam a bit stream as a sequence of non-overlapping pairs. We could then discard any 00 or 11 pairs found, interpret 01 as a 0 and 10 as a 1. Assume the probability of a 1 is 0.5 + e and the probability of a 0 is 0.5 - e where e is the eccentricity of the source. Then the probability of each pair is as follows:

pair	probability
00	(0.5 – e)^2 = 0.25 – e – e^2
01	(0.5 – e)*(0.5 + e) = 0.25 – e^2
10	(0.5 + e)*(0.5 - e) = 0.25 – e^2
11	(0.5 + e)^2 = 0.25 + e + e^2

So the probability of a 0 in the de-skewed sequence is the same as for a 1!. This technique will completely eliminate any bias but at the expense of taking an indeterminate number of input bits for any particular desired number of output bits. The probability of any particular pair being discarded is 0.5 + 2e^2 so the expected number of input bits to produce X output bits is X/(0.25 - e^2).

Defining Randomness

Sequences at random

Let's consider infinite sequences of x's pertaining to a GF2

.

Definition 1: Density D(x, n)

$$D(x, n) = 1/n. \sum (x(i)), \text{ for } i<n$$

Where the ∑ operator stands for Summa extended over i. This is like the "average" of ones. The limiting density of x's is Lim(n)D(x, n) if that limit exists.

A true random sequence should have a limiting density ½ (though one might wonder why the limit should actually exist in the strict sense of convergence in mathematic analysis sense).
In fact, one would look for a certain degree of convergence: the D(x, n) should tend to ½ even though not too rapidly.

Definition 2: An infinite sequence of x's pertaining to a GF2 is Mises random if the limiting density of any subsequence (xij) is ½ where the subsequence is selected by a rule named *Auswahlregel*.
For example, all the following sequences should have limiting density ½.

x0, x1, x2, . . . , xn, . . .
x0, x2, x4, . . . , x2n, . . .
x1, x4, x7, . . . , x3n+1, . . .
x0, x1, x4, . . . , xn2, . . .

Auswahlregeln Rule

This rule must be defined as a function N => N without any knowledge of x: otherwise we can simply pick a subsequence of all 0's. Now suppose we have a countable system of Auswahlregeln and our sequence passes all these tests. In other words, we have accounted for many increasing functions f : N => N and for each of these: lim(n)D((xf(n)), n) = ½ apply. Then we can think the sequence of x's as being true random.

Some random numbers tables

1000 numbers ranged between 1 and 10^6 - First Half

402731	355255	154123	584556	751892	903723	25205	760565	897040	213990
441393	677505	513222	477979	575753	153682	927392	462215	326299	154300
7113	538794	659094	940086	739828	164042	345218	924871	276389	877453
112344	699253	160452	245731	935663	131555	97529	453863	237865	446778
582918	659913	579583	267023	236868	776264	579878	158904	245705	314755
242246	644370	29066	55101	727869	10294	713041	664440	47941	581262
106445	963934	422305	389524	858656	312743	463920	143510	626996	75941
44558	322874	665393	370611	884583	700677	644060	841021	455419	285948
852984	651258	181086	601009	311878	348734	364168	817977	278107	638529
503425	222469	879648	542972	36080	72762	220882	229886	511935	113689
820528	65764	771115	868606	297175	886905	822500	895128	103563	82383
454158	540267	436079	20744	10822	967651	544366	804219	451866	861483
973190	671425	239898	457862	271177	301561	594764	467726	624052	513146
651157	26436	969007	730913	564395	402834	569136	716368	195069	982773
941901	502563	94654	766071	433906	695429	168568	957576	728272	148213
564964	731952	555924	755902	244080	250173	889551	96071	421689	313064
244314	167124	910755	109523	879437	869055	141570	221321	885127	792362
234977	366080	467802	234934	971705	389998	312822	502094	970228	508835
58146	707416	176557	411654	84764	988424	125248	106819	229198	302001
635963	851857	436919	875096	116619	170512	131842	205997	955607	786409
798168	616861	920186	747493	371193	89771	99944	701803	337795	321898
335517	629836	76756	32623	561271	765490	573580	903601	416345	585323
942989	541129	110975	10323	514031	491242	34238	374835	334675	627431
651885	743818	304934	663833	998674	577361	313107	399227	64378	271688
956407	831546	109956	433681	280359	728984	831667	482074	247478	748427
80184	406763	5193	530165	841365	43432	689502	968111	767587	244725
663521	171253	27660	684025	882715	759566	26623	768598	19665	484674
339912	355548	535437	673437	868915	529508	436557	12535	386206	817160
165361	204483	393140	773686	392295	201825	760986	137368	435325	743865
465678	440884	605953	584710	851201	253401	966675	200924	985525	439910
929779	823883	874702	661457	274790	748398	957099	196545	409026	273619
13936	861360	427853	193261	469031	690423	643138	682794	573367	153231
208994	937108	844273	821546	162671	705495	506970	386180	778256	813997
907508	665302	583885	352004	66542	879896	553498	960091	119094	363836
370235	818862	141234	392104	96156	385213	963452	264686	773125	607811
340552	859759	819024	599492	199309	527948	33991	471319	561111	914422
760311	77303	789400	604991	655287	459663	205745	322275	448294	512560
365314	848266	689923	409228	798852	78319	542320	642772	24522	525587
825543	426495	91718	764801	642175	871543	756602	501042	619124	59665
9357	757087	854838	862245	162107	602905	531981	336645	48774	665606
636871	656134	512035	171716	988495	794742	145965	665021	879351	51008
562383	304588	704141	22833	384694	758538	425808	778008	991437	855172
128354	68051	661493	103890	205699	307198	862970	671014	229046	209262
74240	927183	465301	327361	839239	916390	425635	359064	538934	650876
134409	408439	374379	876828	439344	226248	185475	578841	432027	719581
298798	749018	792228	92700	431475	660083	705749	387091	894808	523139
362575	268843	709018	863779	64615	340339	928097	255587	27048	278296
838362	625096	197433	440369	167165	962867	778557	26203	904885	467099
682499	68030	793724	204422	666646	941242	793207	987888	123860	799254
399158	333009	163327	168446	612715	554773	350460	953904	543950	639771

1000 numbers ranged between 1 and 10^6 - Second Half

507368	741560	267648	60201	686812	549492	942751	605533	633070	696625
96195	817291	244691	343187	740342	847998	150521	154055	113408	55556
753021	600817	996556	206266	529291	556184	233396	282657	177960	510283
588077	106376	474659	989934	399735	163972	373956	350341	807149	170869
563289	420543	917312	383036	704262	160937	666785	181754	227104	309281
851083	507075	17199	441110	412477	997084	786149	749333	912248	238208
480289	560399	640650	662733	121869	141432	494298	540186	626572	504770
607672	63544	612575	544909	557693	817951	94969	910596	174155	637526
770918	108629	309371	551619	695373	723873	360181	593923	293386	81866
763764	210197	499000	239312	907660	215696	266324	820661	690647	125907
240812	261508	748336	438925	406423	858922	236916	495915	928165	74811
102363	659287	64973	958067	365198	864384	857272	184884	1515	350970
606544	888663	802780	113272	335680	705320	20085	127327	880542	992949
691119	169669	2256	762437	841847	843387	670241	12729	788545	416216
866910	603139	222251	336233	227213	393772	894677	778420	724251	256433
313627	419089	86153	138042	652751	438991	389899	623023	773061	383
427148	221092	512927	194856	720270	873615	721935	279893	588572	414098
913787	394235	123153	241451	570025	788980	970076	569283	115794	169025
421442	586507	269760	339576	638478	398400	707585	163898	170094	271350
74685	255993	580046	714423	514854	23866	326097	729762	885200	378001
107956	316428	998855	587125	413658	868643	980244	148746	98687	530640
969331	899957	279348	107830	444774	115092	518064	803026	468996	137469
628251	623816	830101	922496	676894	187753	678496	456730	433268	342736
460017	67891	781569	989942	613858	651343	937955	204282	421223	506955
551492	27154	203805	983230	672484	96096	215460	268737	428982	186315
272166	874152	462995	612945	939274	607422	10587	67834	899126	847122
168761	70969	36479	971218	890343	526252	323322	375821	300350	259904
320130	154035	841271	314972	118815	791875	257867	493601	41166	468622
807177	101938	94352	583628	424882	714102	118249	214981	225805	1337
565596	97764	681091	971430	626475	292899	177735	973600	415421	113407
948023	669948	788894	285744	876513	113451	648163	766619	791165	310145
828782	124910	876774	969	989787	671817	733136	203128	120576	522427
373614	56491	371052	843224	780484	330051	117301	941224	911782	278482
644946	534257	234257	579261	422763	350480	576118	528989	255457	167680
182345	106639	490194	284026	545174	820313	369547	342332	399341	404579
87283	608957	891362	745380	410440	161179	335042	997210	46436	679168
252228	960905	195084	727861	222337	282351	972626	941294	519452	501394
791025	744013	975885	426401	484236	116913	934294	190242	362931	923620
244690	32545	605965	607058	148153	513295	80594	797572	959889	631540
252375	322443	944438	490524	181308	450773	706513	454344	741861	982455
51763	130881	205573	619091	245597	7673	491550	129109	543035	165523
593634	197807	614668	275675	146747	673127	382458	272429	198392	771932
615020	36851	694004	270003	660497	689770	845168	800741	616901	89884
524483	173943	564949	721992	470836	729368	525571	27529	488340	465165
567928	757253	196049	61459	109223	605898	89911	641598	657670	105019
323666	653545	408192	879387	101907	725919	425115	65179	404303	211800
151786	360788	610547	616733	499082	336837	797667	523431	111795	460528
478648	208509	78451	535893	837780	123702	388973	116954	897058	73235
700775	311922	390914	594050	843244	98180	70525	423444	119363	361505
511949	559334	383043	982303	250125	551059	450925	152187	815219	803037

256 HX pairs

79	41	e6	44	17	de	d9	f8	3b	b2	4f	45	c8	be	7c	ca
17	09	6c	ee	17	fb	a3	39	bd	c0	12	e6	50	24	d6	18
61	69	35	c3	80	10	a8	9a	4a	e8	b6	0c	6f	06	2c	5f
09	2e	a7	5b	c2	01	ae	98	8a	77	aa	89	b4	de	83	ef
f9	2a	af	cd	3f	51	8d	43	d0	01	d4	dd	aa	43	dc	87
9e	9d	c8	ac	ae	48	c7	46	99	24	f0	cc	1b	95	0c	c0
d9	2f	a2	e2	8f	47	70	97	f3	69	55	4e	35	a8	31	c7
12	ec	68	e3	d4	74	4c	5a	99	2d	8d	e2	3d	08	42	68
5c	bc	6b	56	06	26	7d	68	8e	bb	1e	db	59	df	11	5d
b8	85	f9	9c	67	03	08	9a	c8	63	d8	8f	ea	f1	11	74
4f	e4	29	81	15	9b	58	68	ed	d7	bf	ec	e5	e9	12	5e
e0	3d	69	29	f1	e7	ae	7b	a3	5e	7c	a2	02	7f	28	b3
0e	6b	f8	83	76	1b	04	cd	70	9e	e6	37	83	b9	f6	ef
01	9b	75	a7	b8	26	91	ab	ee	62	68	82	80	74	26	cc
95	ac	82	d7	20	e0	57	be	06	5f	78	84	60	f3	19	80
aa	8b	ce	8e	fa	dc	49	08	56	c0	c8	ff	52	d7	26	44

256 octal

010	250	262	242	366	012	113	002	360	376	236	347	263	317	225	132
240	301	102	373	366	212	220	261	370	312	066	317	161	071	321	305
064	367	106	140	356	107	103	335	120	226	306	170	030	172	053	202
306	133	255	366	105	130	006	026	037	115	105	212	212	037	324	345
337	315	001	032	044	275	226	164	313	163	345	253	342	327	355	227
150	331	047	101	243	025	124	145	146	327	067	240	321	016	132	222
271	301	315	272	342	346	253	270	054	233	243	137	016	373	122	273
304	313	052	366	276	024	047	361	157	106	141	115	376	332	221	342
073	156	133	014	343	161	356	244	026	251	073	213	342	332	252	020
333	132	057	350	354	173	167	202	214	154	273	135	201	105	336	142
250	024	052	366	042	151	045	373	007	200	267	265	122	201	212	025
375	036	223	356	257	174	220	314	214	325	122	151	124	204	375	063
025	352	110	117	035	376	025	016	032	305	176	343	366	304	065	367
177	160	265	316	175	141	171	100	345	375	306	115	061	060	176	031
213	254	063	211	046	074	240	353	060	341	240	144	125	274	334	266
221	053	144	065	355	313	017	034	332	304	333	167	267	116	164	147

Monte Carlo Techniques

Source: Department of Physics, Drexel University, Philadelphia.

This technique is fundamentally used to simulate statistic events and it derives from Integral Calculus. Let's suppose we have to compute H by integrating function f over a given n-space as ∫follows

$$H = \int\int\int\int \ldots f(x(1), x(2), x(3), \ldots, x(n).\ dx(1).dx(2).dx(3).\ldots.dx(n)$$

Where ∫'s stand for "Integrals" defined over domains in an n-dimensional space of the elementary computation (f.dv) where f stands for an n-dimensional function and dv for the infinitesimal hypercube defined by dv = dx(1).dx(2).dx(3).….dx(n). In its discrete approximation It would mean m^n elementary computations of the core algorithm in a Cartesian space, namely m^n infinitesimal calculations of the form H <= H + f.dv along n loops of range m, where m stands for the number of sample points along each dimension supposedly all equal. If on the contrary for each dimension we have a particular m(i) the Cartesian space will have m(1).m(2).m(3).…m(n) points instead.

Note: That's too much for m and n values easy to find in simulations, for example being n=6 and m=1000 for a 1000^6 = 10^18 grid for some weather forecast models!. We may navigate throughout this space systematically by "layers" at "brute force" mode (throughout all points) or following random walks. Monte Carlo approach estimates H via random samples within the Cartesian m^n space.

One dimensional case 1-D

For a given f defined over an interval [a b], the H computation will in pseudo code have the following form:

```
H=0, i=0
Do Core from i=0 till i=m, i++
Core: H = H + f(i)
H= delta.H
```

Where:

delta is the interval length divided in m parts, a practical infinitesimal.
f(i) is an array of m values along [a b]

Then the content of H at the end of computation will be

$$H = delta.\ \sum (i)\ f(i).,$$

That reads delta multiplied by the ∑ of all array values. It's the same as

$$H = delta.m.<f>,$$

Where <f> is the average of f along the interval [a b]. As delta.m is the length L of interval [a b] it results

$$H = L.<f>$$

So instead of computing H via "brute force" along m elementary computations, the Monte Carlo technique try to get a "good enough" statistical estimator of <f>. The economy rests on choosing a step larger than "delta" step. Let's suppose that m = 10^8 points; applying the brute force method would imply to make 10^8 calculations an to keep someplace at hand equal number of f(i) values. What would be the resulting "error" in H if we use 1000 points instead of 10^8?: Perhaps not too much for our purpose. We may then take 1000 points spaced regularly along the interval [a b], equal number of corresponding f values and obtain <f> and its corresponding "variance" by the expression

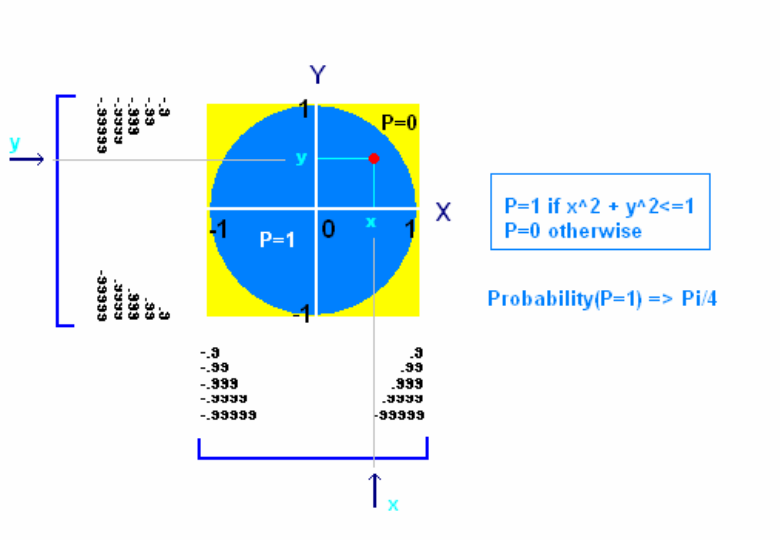
$$\sigma^2 = \frac{1}{N}\left[< f^2 > - < f >^2\right]$$

Where <f^2> means the average of squares.

Pi Estimation by Monte Carlo

One of the classical Monte Carlo demos is the determination of number "pi" at a certain precision level. It's based on the "quadrature" paradox of the antique world (Babylonians knew Pi with a reasonable precision as 25/8 = 3.125 that compared with Pi = 3.1416 gives us an error of 0.5%). If we integrate with a very elementary f (for instance f=1 within the X[-1, 1], Y[-1, 1] domain and zero outside it) along a circle of radio equal to 1, enclosed with a square of size equal to 2 we may find the following expression:

Area of the circle = (Pi)x1^2 = Pi
Area of the square = 4



Then area of the circle/area of the square = Pi/4, is a parameter that could be approached statistically by Monte Carlo techniques as follows. The idea is to “throw”/”pick” a zero dimension point chosen at random within the interval [-1, 1] in both dimensions X and Y of a Cartesian space. The probability p of these points to “fall” in either figure, the circle or the square is in relation to their areas

p(circle) = Pi/4
p(square) = 1

We may then generate N random pairs [x, y] of coordinates, for example between the range [-0.99999, 0.99999] and simply compute a Boolean variable P telling us if falling is inside or outside the circle. The pseudo code could be something like the one depicted below:

```
H=0
.....
x <= Random [-1 1]
y <= Random[-1 1]
If (x^2 + y^2>=1) P=1, and P=0 otherwise
H=H+P
.....
```

..

At the end of computation the estimation of Pi is given by four times H/N. As this is a pseudo Bernoulli process it will follow the Binomial Distribution that gives:

E(H) = E(<f>) = N.p
Variance = N.p.(1-p)

In numbers if we generate 10,000 pairs [x, y],

N=10,000,
E(H) = 10,000x3,1416/4 = 10,000x0.7854 = 7854

Totaling 7854 points within the circle with an expected standard deviation given by:

Variance = 10.000x(0.7854)x(0,2146) = 1685 => standard deviation = (1685)^ ½ = 41.

Statistics and Tests

NIST, http://csrc.nist.gov/rng/

The quality of random numbers can be measured in a variety of ways. One common method is to compute the **Information Density**, *or* entropy, in a series of numbers. The higher the entropy in a series of numbers is, the more difficult it is to predict a given number on the basis of the preceding numbers in the series. A sequence of good random numbers will have a high level of entropy, although a high level of entropy does not guarantee randomness. Paradoxically files compressed have a high level of entropy but as data is highly structured its randomness is not guaranteed being in fact considered not at random. Hence, for a thorough test of a random number generator, computing the level of entropy in the numbers alone is not enough.

Guide to Statistical Tests

http://csrc.nist.gov/rng/rng9.html

A total of sixteen statistical tests were developed, implemented and evaluated. The following describes each of the tests.

- Frequency (Monobits) Test
    - The focus of the test is the proportion of zeroes and ones for the entire sequence. The purpose of this test is to determine whether that number of ones and zeros in a sequence are approximately the same as would be expected for a truly random sequence. The test assesses the closeness of the fraction of ones to ½, that is, the number of ones and zeroes in a sequence should be about the same.
- Test for Frequency within a Block
    - The focus of the test is the proportion of zeroes and ones within M-bit blocks. The purpose of this test is to determine whether the frequency of ones is an M-bit block is approximately M/2.
- Runs Test
    - The focus of this test is the total number of zero and one runs in the entire sequence, where a run is an uninterrupted sequence of identical bits. A run of length k means that a run consists of exactly k identical bits and is bounded before and after with a bit of the opposite value. The purpose of the runs test is to determine whether the number of runs of ones and zeros of various lengths is as expected for a random sequence. In particular, this test determines whether the oscillation between such substrings is too fast or too slow.
- Test for the Longest Run of Ones in a Block
    - The focus of the test is the longest run of ones within M-bit blocks. The purpose of this test is to determine whether the length of the longest run of ones within the tested sequence is consistent with the length of the longest run of ones that would be expected in a random sequence. Note that an irregularity in the expected length of the longest run of ones implies that there is also an irregularity in the expected length of the longest run of zeroes. Long runs of zeroes were not evaluated separately due to a concern about statistical independence among the tests.
- Random Binary Matrix Rank Test
    - Description: The focus of the test is the rank of disjoint sub-matrices of the entire sequence. The purpose of this test is to check for linear dependence among fixed length substrings of the original sequence.

- Discrete Fourier Transform (Spectral) Test

    - Description: The focus of this test is the peak heights in the discrete Fast Fourier Transform. The purpose of this test is to detect periodic features (i.e., repetitive patterns that are near each other) in the tested sequence that would indicate a deviation from the assumption of randomness.

- Non-overlapping (Aperiodic) Template Matching Test

    - Description: The focus of this test is the number of occurrences of pre-defined target substrings. The purpose of this test is to reject sequences that exhibit too many occurrences of a given non-periodic (aperiodic) pattern. For this test and for the Overlapping Template Matching test, an m-bit window is used to search for a specific m-bit pattern. If the pattern is not found, the window slides one bit position. For this test, when the pattern is found, the window is reset to the bit after the found pattern, and the search resumes.

- Overlapping (Periodic) Template Matching Test

    - Description: The focus of this test is the number of pre-defined target substrings. The purpose of this test is to reject sequences that show deviations from the expected number of runs of ones of a given length. Note that when there is a deviation from the expected number of ones of a given length, there is also a deviation in the runs of zeroes. Runs of zeroes were not evaluated separately due to a concern about statistical independence among the tests. For this test and for the Non-overlapping Template Matching test, an m-bit window is used to search for a specific m-bit pattern. If the pattern is not found, the window slides one bit position. For this test, when the pattern is found, the window again slides one bit, and the search is resumed.

- Maurer's Universal Statistical Test

    - Description: The focus of this test is the number of bits between matching patterns. The purpose of the test is to detect whether or not the sequence can be significantly compressed without loss of information. An overly compressible sequence is considered to be non-random.

- Lempel-Ziv Complexity Test

    - Description: The focus of this test is the number of cumulatively distinct patterns (words) in the sequence. The purpose of the test is to determine how far the tested sequence can be compressed. The sequence is considered to be non-random if it can be significantly compressed. A random sequence will have a characteristic number of distinct patterns.

- Linear Complexity Test

    - Description: The focus of this test is the length of a generating feedback register. The purpose of this test is to determine whether or not the sequence is complex enough to be considered random. Random sequences are characterized by a longer feedback register. A short feedback register implies non-randomness.

- Serial Test

    Description: The focus of this test is the frequency of each and every overlapping m-bit pattern across the entire sequence. The purpose of this test is to determine whether the number of occurrences of the 2m m-bit overlapping patterns is approximately the same as would be expected for a random sequence. The pattern can overlap.

- Approximate Entropy Test

    - Description: The focus of this test is the frequency of each and every overlapping m-bit pattern. The purpose of the test is to compare the frequency of overlapping blocks of two consecutive/adjacent lengths (m and m+1) against the expected result for a random sequence.

- Cumulative Sum (Cusum) Test
    - Description: The focus of this test is the maximal excursion (from zero) of the random walk defined by the cumulative sum of adjusted (-1, +1) digits in the sequence. The purpose of the test is to determine whether the cumulative sum of the partial sequences occurring in the tested sequence is too large or too small relative to the expected behavior of that cumulative sum for random sequences. This cumulative sum may be considered as a random walk. For a random sequence, the random walk should be near zero. For non-random sequences, the excursions of this random walk away from zero will be too large.
- Random Excursions Test
    - Description: The focus of this test is the number of cycles having exactly K visits in a cumulative sum random walk. The cumulative sum random walk is found if partial sums of the (0,1) sequence are adjusted to (-1, +1). A random excursion of a random walk consists of a sequence of n steps of unit length taken at random that begin at and return to the origin. The purpose of this test is to determine if the number of visits to a state within a random walk exceeds what one would expect for a random sequence.
- Random Excursions Variant Test
    - Description: The focus of this test is the number of times that a particular state occurs in a cumulative sum random walk. The purpose of this test is to detect deviations from the expected number of occurrences of various states in the random walk.