Random Number Toolbox

Last time ....

- **Definition** of probability distributions, description of properties, $\int = 1.0000$, etc.
- **Moments** of distributions => mean, variance, skewness, kurtosis …

- **Binomial distribution** for discrete probabilities of outcome of **N** trials with identical probability **ρ** of ‘success’ or ‘failure’.
- Bayesian example of galaxy cluster sample, centrally-dominated or otherwise.

- **Poisson distribution** for many trials with rare possibility of successful outcome, probability of **n** events has mean value **μ**,
- Example – photon arrival statistics, and how signal-to-noise ratio depends on integration time

- **Gaussian (Normal)** distribution, symmetrical, mean **μ**, variance σ^2**,** the asymptotic result of previous two distributions for large Np, μ.

- **Central Limit Theorem => Miraculous!** Gaussian distributions everywhere….
- Examples – truncated exponential; two delta functions

- **Power-law distribution** – the distribution from hell, because it does not conform, and our (tactit / assumed) reliance on Central Limit Theorem is lost. Many pitfalls to navigate regarding indices.
- Example – the number magnitude counts from a deep image of the sky.

Why generate random numbers?

On many occasions in hypothesis-testing and model-fitting we must have a set of numbers distributed how we might guess the data to be.

We may wish

☺ to check error propagation
☺ to test a test to see if it works as advertised;
☺ to test efficiency of tests;
☺ to find how many iterations we require to reach a given level of significance;
☺ to test our code.

We gotta have these random numbers, of two kinds:
- uniformly distributed,
- drawn randomly from a parent population of known frequency distribution.

The pitfalls of random-number generators

Usual form: x=random(idummy)

No excuse for using bad random data.
EXAMPLE: RANDU, the infamous IBM random-number generator.

Cycle length - how long is it before the pseudo-random cycle is repeated?

Important to understand the characteristics of the generator.

Essential to follow the prescribed procedure.

Never forget that the routines generate pseudo-random numbers.

Numerical Recipes presents a number of methods, from single expressions to powerful routines.

Random numbers from a given distribution

How do we draw a set of random numbers following a **given** frequency distribution?

Suppose we have a way of producing random numbers that are uniformly distributed, in say the variable α; and we have a functional form for our frequency distribution $dn/dx = f(x)$. We need a transformation $x = x(\alpha)$ to distort the uniformity of α to follow $f(x)$. But we know that

$$\frac{dn}{dx} = \frac{dn}{d\alpha}\frac{d\alpha}{dx}$$

and as $dn/d\alpha$ is uniform, thus

$$\frac{dn}{dx} = \frac{d\alpha}{dx},$$

and

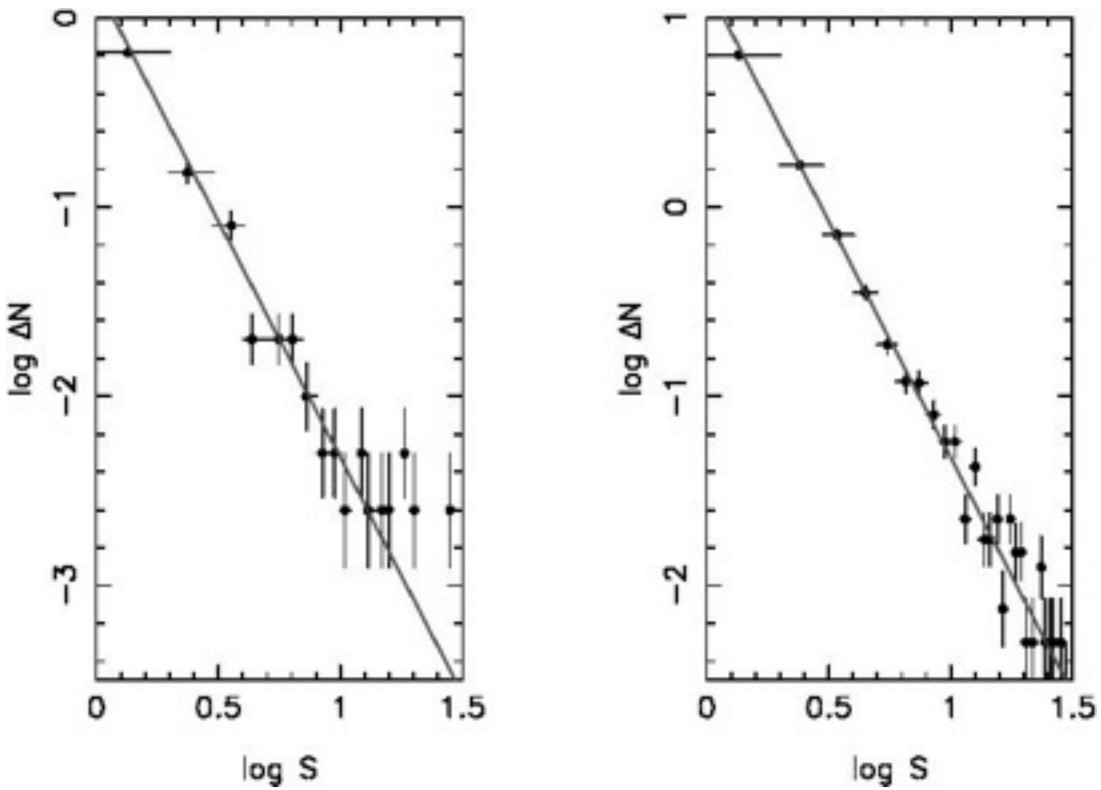
$$\alpha(x) = \int^{x} f(x)dx,$$

so that the required transformation is $x = x(\alpha)$.

We therefore need $x = f^{-1}(\alpha)$, the inverse function of the integral of $f(x)$.

Randoms from a given distribution 2

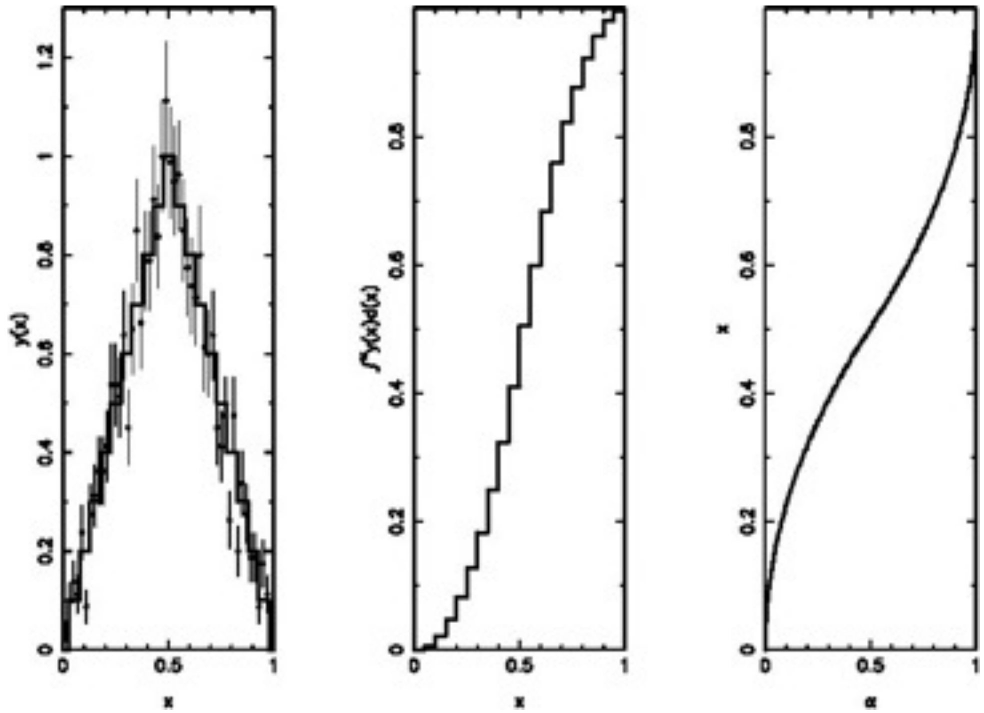
EXAMPLE: A source-count distribution is given by $\mathbf{f(x)dx = -1.5x^{-2.5}dx}$, a `Euclidean' differential source count. Here $\mathbf{d\alpha = -1.5x^{-2.5}dx}$, $\mathbf{\alpha = x^{-1.5}}$, and the transformation is $\mathbf{x = f^{-1}(\alpha) = \alpha^{-1/1.5}}$.



Differential source counts generated via Monte Carlo sampling with an initial uniform deviate, obeying the source-count law $N(>S) = kS^{-1.5}$. The straight line in each shows the anticipated count with slope -2.5. left: k = 1.0, 400 trials, right: k = 10.0, 4000 trials.

Randoms from a given distribution 3

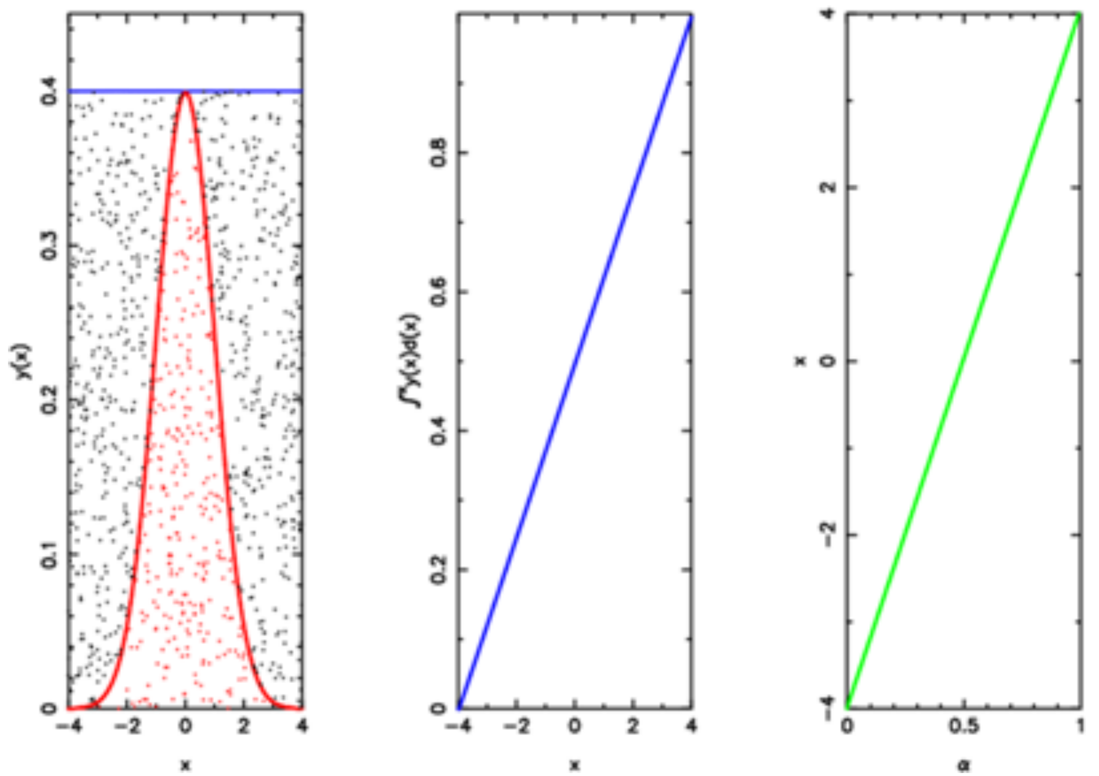
The very same procedure works if we don't have a functional form for f(x)dx.
If this is a histogram, we need simply to calculate the integral version, and perform the reverse function operation as before.
EXAMPLE:



An example of generating a Monte-Carlo distribution following a known histogram. Left: the step-ladder histogram, with points from 2000 trials, produced by a) integrating the function (middle) and b) transforming the axes to produce f^{-1} of the integrated distribution (right). The points with √N error bars in the left diagram are from drawing 2000 uniformly-distributed random numbers and transforming them according to the right diagram.

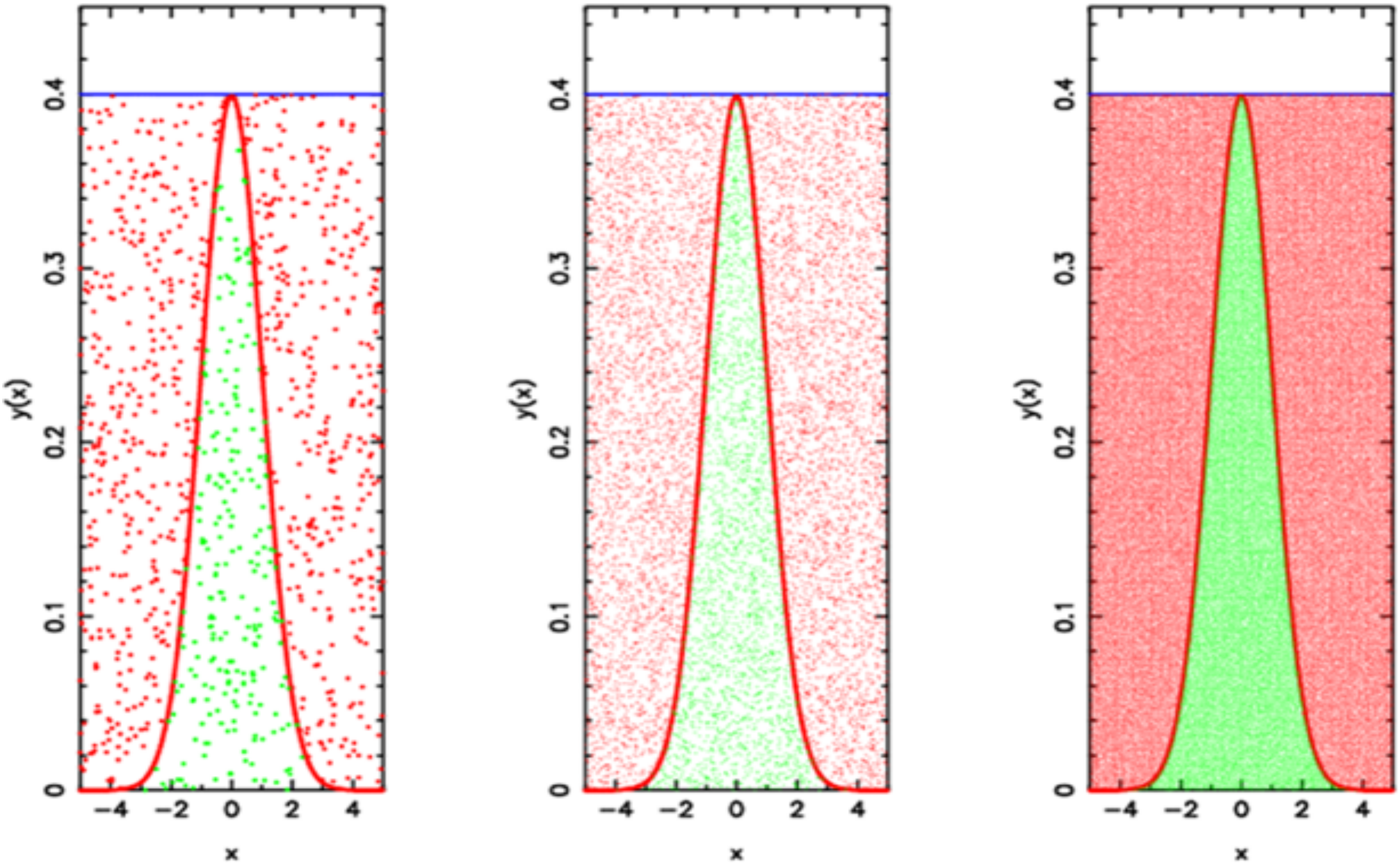
Randoms from a distribution - rejection method

1. Plot the culprit distribution – non integrable, nasty
2. Find a nicely-behaved (integrable) one which looks a bit similar and lives higher.
3. Get the random values of x for this one, via the transform method.
4. Find a random distance up the y-axis for each of these x values
5. Reject the ones which lie outside the require distribution.



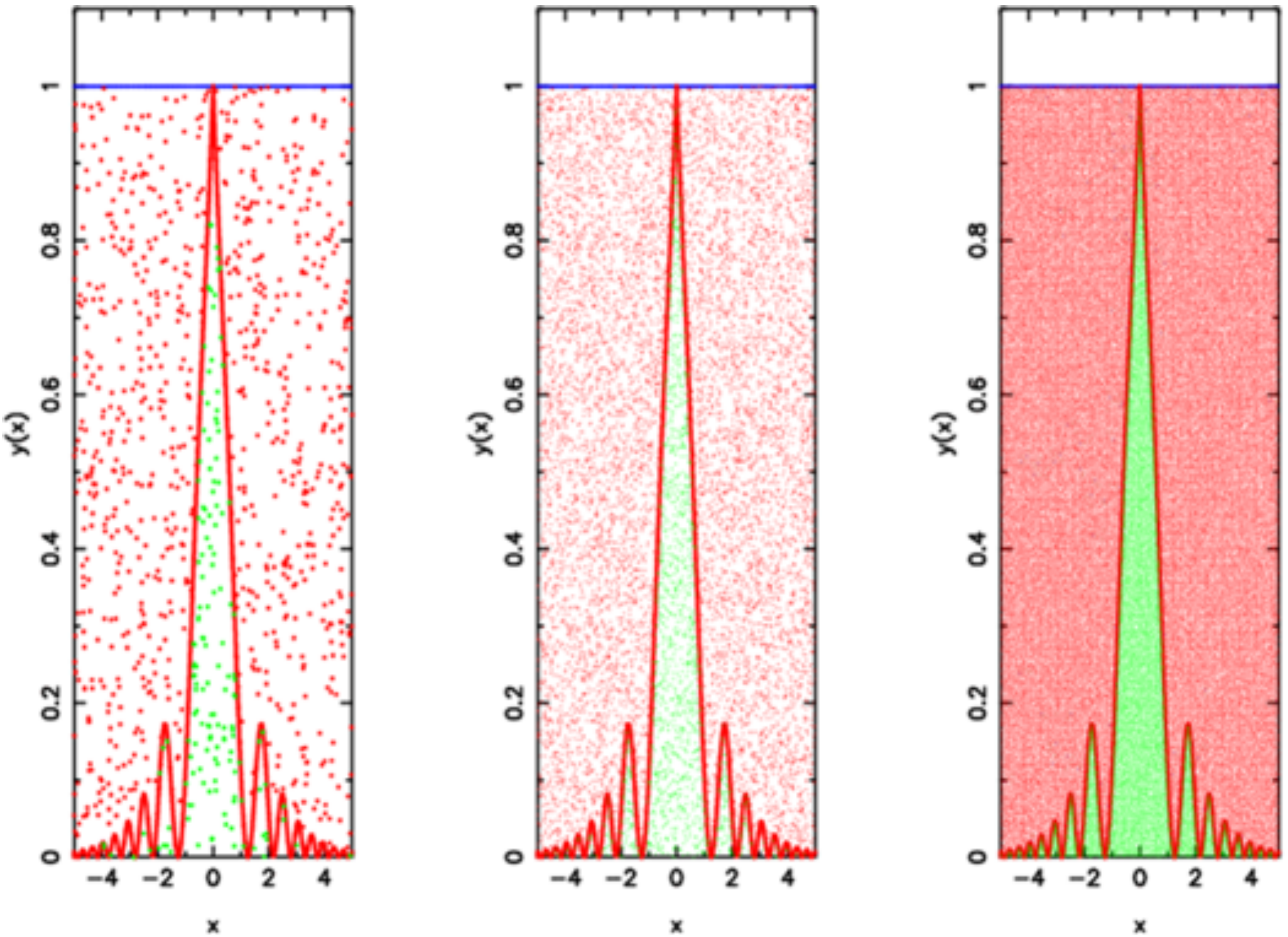
Example 1: Gaussian to +/- 5σ with elevated function (= uniform). Centre, right – transformation of the uniform function to give uniform x (and y) coverage. 1000 trials; 241 points within Gaussian; hence area estimate = (241/1000) x (0.4 x 10.0) = 0.9640.

Rejection method - Example 1 continued



Gaussians out to +/- 5 sigma. Left 1000 points, 241 in zone, area = (241/1000) x (10.*0.4) = 0.9640; Centre 10000 points, 2540 in zone, area = 1.0160; Right 10^6 points, 250830 in zone, area = 1.0015

Randoms via rejection - Example 2

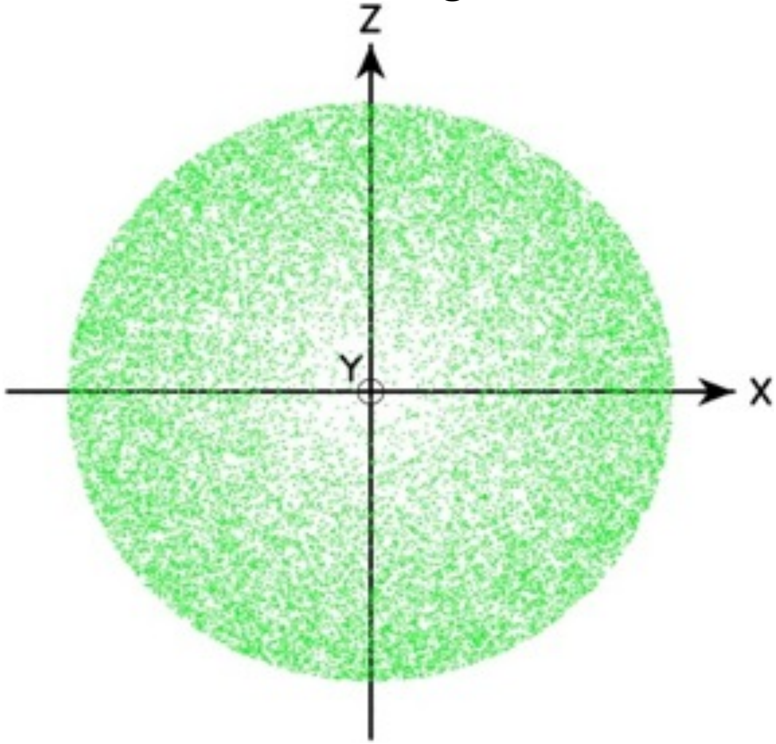


$f(x)=exp^{-|x|}cos(x^2)$. Left, 1000 points, 152 in zone, area = (152/1000) x (10.*1.0) =1.5200; Centre 10000 points, 1444 in zone, area = 1.4440; Right 10^6 points, 140229 in zone, area = 1.4023.

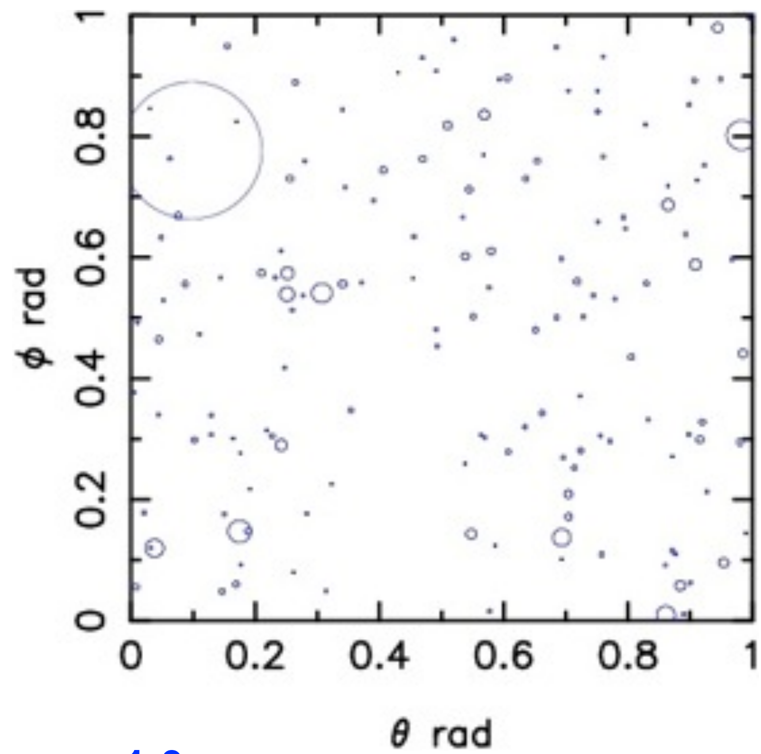
Randoms from a distribution - Example 3

Introducing the toy universe

Thin slice though centre



A view of the "sky"



Assume a flat Euclidean universe with $r_{max} = 1.0$.

Populate this with 10^6 objects randomly but uniformly distributed, i.e. 10^6 values of $(r_i, \theta_i, \varphi_i)$

Assign each a luminosity L = 1.0, so that each produces a flux (at the centre) of $1/r_i^2$

Monte Carlo (random-number) Integration

Numerical integration is a very important use of Monte Carlo.

Highly technical! Outline here only.

Suppose we have a probability distribution **f(x)** defined for $\mathbf{a} < \mathbf{x} \leq \mathbf{b}$.
Draw **N** random numbers **X**, uniformly distributed between **a** and **b**, and calculate the function at these points.
Add these values of the function up, normalize - and this is our answer.

$$\int_a^b f(x)\,dx \simeq \frac{(b-a)}{N} \sum_i f(X_i).$$

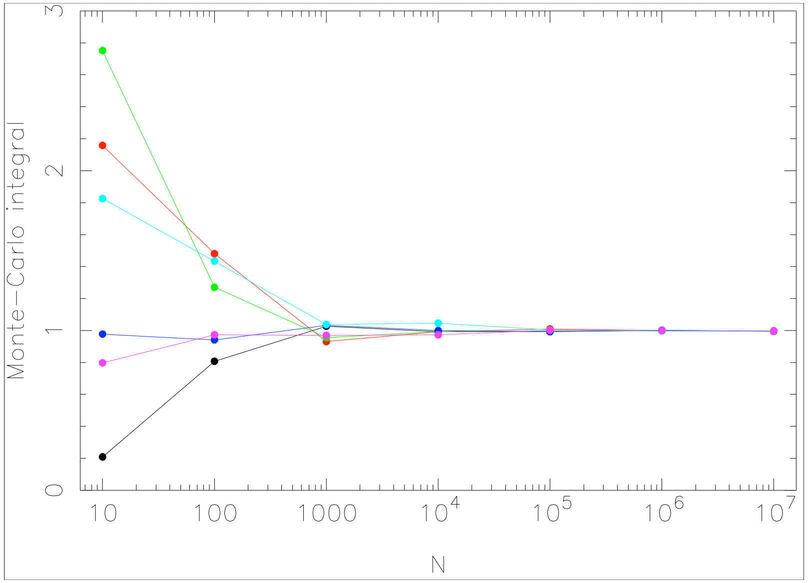
This is **Monte Carlo integration**.

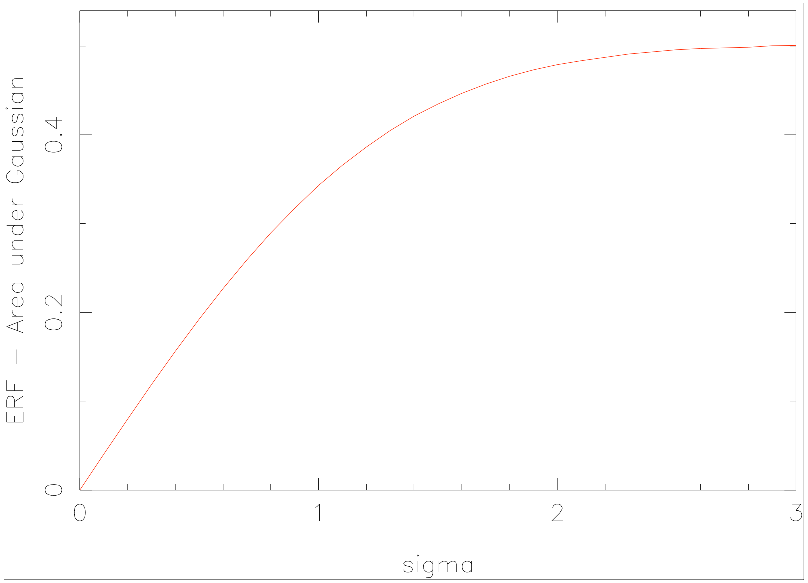
If the $\mathbf{X}_i$ are drawn from the distribution f itself, then they will sample the regions where **f** is large and the integration (for the same number of points) will be far more accurate. This technique is called importance sampling.

M C Integration - Example (Gaussian)

Use a uniform random-number generator such as the function routine *ran1* of *Numerical Recipes*; make N calls to it, scaling the $(0 \rightarrow 1)$ random numbers to the range of σs required, say $k\sigma$. For each resulting value x_i, compute $f(x_i) = \frac{1}{\sigma\sqrt{2\pi}}\exp[-\frac{x_i^2}{2\sigma^2}]$. The integral from 0 to $k\sigma$ is simply

$$\frac{k}{N}\sum_{i=1}^{N} f_i(x).$$





Right – the result, using N=10^6. Left – using +/- 10σ, and varying N. The different curves are the results of different starting indices for the random-number generator. This mindless sum shows how stable MC integration is for well-behaved functions; we have uniformly sampled +/- 10σ, and the function is really a spike between +/- 2σ.

Sorting, indexing, ranking

Note that in most cases, you want to sort non-destructively – unless it's a 1D 'table'. For non-destructive sorting, you'll need to think in terms of 4 arrays:

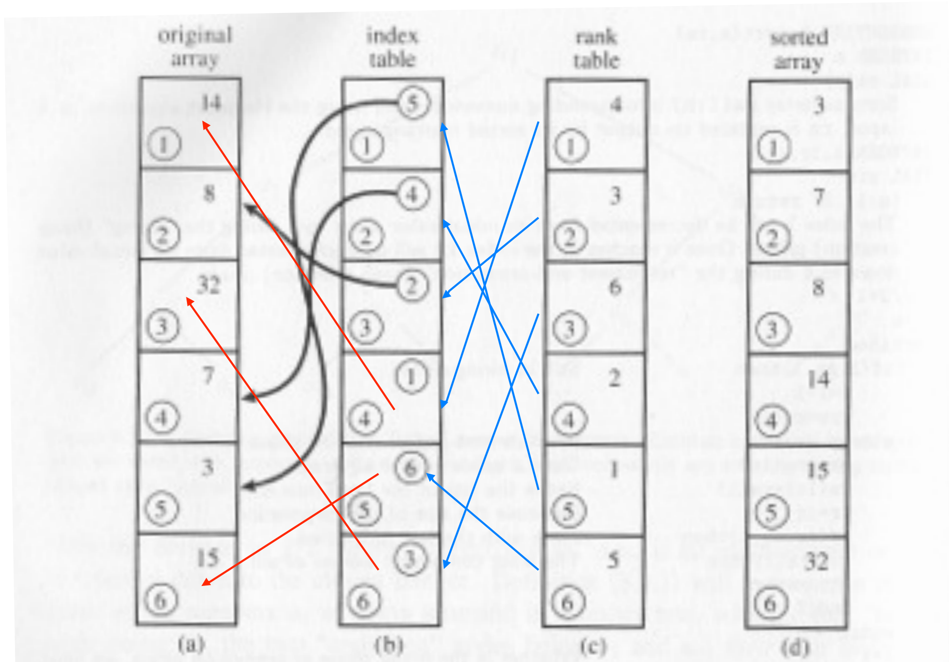


Figure 8.4.1. (a) An unsorted array of six numbers. (b) Index table, whose entries are pointers to the elements of (a) in ascending order. (c) Rank table, whose entries are the ranks of the corresponding elements of (a). (d) Sorted array of the elements in (a).

Borrowed from Numerical Recipes

For general sorting of a 1D array, of course you can do it directly; and it goes as **N!** in speed. Note how much efficiency you can gain by the sweat of others – there are many fast methods. **Mainly you need to look at *Numerical Recipes*!**

Sorting, indexing, ranking - Example

The Francis and Wills (1999) sample of QSOs (quasars)

0947+396	45.66	1.51	3.684	0.23	1.18	3.520	2.08	1.78	0.45	1.24	0.306	0.179	0.143
0953+414	45.83	1.57	3.496	0.25	1.26	3.432	2.19	1.78	0.40	1.24	0.164	0.189	0.093
1114+445	44.99	0.88	3.660	0.20	1.23	3.654	2.27	1.85	0.42	1.48	0.222	0.175	0.092
1115+407	45.41	1.89	3.236	0.54	0.78	3.403	1.90	1.51	0.33	1.14	0.385	0.228	0.134
1116+215	46.00	1.73	3.465	0.47	1.00	3.446	2.14	1.71	0.34	1.20	0.440	0.254	0.126
1202+281	44.77	1.22	3.703	0.29	1.56	3.434	2.72	2.41	0.69	1.87	0.164	0.154	0.098
1216+069	46.03	1.36	3.715	0.20	1.00	3.514	2.12	1.95	0.54	1.20	0.037	0.121	0.056
1226+023	46.74	0.94	3.547	0.57	0.70	3.477	1.64	1.44	0.45	1.00	0.280	0.174	0.018
1309+355	45.55	1.51	3.468	0.28	1.28	3.406	2.01	1.68	0.41	1.15	0.303	0.131	0.064
1322+659	45.42	1.69	3.446	0.59	0.90	3.351	2.19	1.85	0.41	1.30	0.291	0.135	0.097
1352+183	45.34	1.52	3.556	0.46	1.00	3.548	2.14	1.80	0.41	1.29	0.357	0.203	0.116
1402+261	45.74	1.93	3.281	1.23	0.30	3.229	1.91	1.59	0.39	1.09	0.568	0.227	0.161
1415+451	45.08	1.74	3.418	1.25	0.30	3.434	2.32	1.78	0.29	1.40	0.688	0.210	0.142
1427+480	45.54	1.41	3.405	0.36	1.76	3.300	2.03	1.82	0.49	1.21	0.265	0.126	0.117
1440+356	45.23	2.08	3.161	1.19	1.00	3.192	2.14	1.54	0.21	1.05	0.747	0.141	0.092
1444+407	45.92	1.91	3.394	1.45	0.30	3.479	1.99	1.34	0.21	1.06	0.809	0.335	0.164
1512+370	46.04	1.21	3.833	0.16	1.76	3.546	2.02	2.05	0.75	1.28	0.228	0.182	0.050
1626+554	45.48	1.94	3.652	0.32	0.95	3.631	2.14	1.80	0.39	1.36	0.197	0.217	0.118

Sorting example, continued

Sorting on column 3:
Index array: 3 8 17 6 7 14 1 9 11 2 10 5 13 4 16 12 18 15

1114+445	44.99	0.88	3.660	0.20	1.23	3.654	2.27	1.85	0.42	1.48	0.222	0.175	0.092
1226+023	46.74	0.94	3.547	0.57	0.70	3.477	1.64	1.44	0.45	1.00	0.280	0.174	0.018
1512+370	46.04	1.21	3.833	0.16	1.76	3.546	2.02	2.05	0.75	1.28	0.228	0.182	0.050
1202+281	44.77	1.22	3.703	0.29	1.56	3.434	2.72	2.41	0.69	1.87	0.164	0.154	0.098
1216+069	46.03	1.36	3.715	0.20	1.00	3.514	2.12	1.95	0.54	1.20	0.037	0.121	0.056
1427+480	45.54	1.41	3.405	0.36	1.76	3.300	2.03	1.82	0.49	1.21	0.265	0.126	0.117
0947+396	45.66	1.51	3.684	0.23	1.18	3.520	2.08	1.78	0.45	1.24	0.306	0.179	0.143
1309+355	45.55	1.51	3.468	0.28	1.28	3.406	2.01	1.68	0.41	1.15	0.303	0.131	0.064
1352+183	45.34	1.52	3.556	0.46	1.00	3.548	2.14	1.80	0.41	1.29	0.357	0.203	0.116
0953+414	45.83	1.57	3.496	0.25	1.26	3.432	2.19	1.78	0.40	1.24	0.164	0.189	0.093
1322+659	45.42	1.69	3.446	0.59	0.90	3.351	2.19	1.85	0.41	1.30	0.291	0.135	0.097
1116+215	46.00	1.73	3.465	0.47	1.00	3.446	2.14	1.71	0.34	1.20	0.440	0.254	0.126
1415+451	45.08	1.74	3.418	1.25	0.30	3.434	2.32	1.78	0.29	1.40	0.688	0.210	0.142
1115+407	45.41	1.89	3.236	0.54	0.78	3.403	1.90	1.51	0.33	1.14	0.385	0.228	0.134
1444+407	45.92	1.91	3.394	1.45	0.30	3.479	1.99	1.34	0.21	1.06	0.809	0.335	0.164
1402+261	45.74	1.93	3.281	1.23	0.30	3.229	1.91	1.59	0.39	1.09	0.568	0.227	0.161
1626+554	45.48	1.94	3.652	0.32	0.95	3.631	2.14	1.80	0.39	1.36	0.197	0.217	0.118
1440+356	45.23	2.08	3.161	1.19	1.00	3.192	2.14	1.54	0.21	1.05	0.747	0.141	0.092