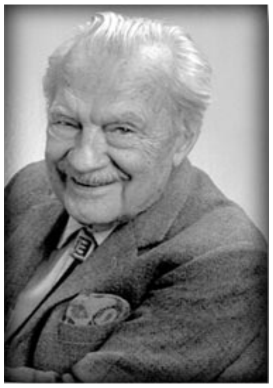
# Monte Carlo: Techniques and Theory

Ronald Kleiss[1]
IMAPP, Radboud University of Nijmegen

version of Sunday 30th August, 2020, 21:34

John von Neumann
the other godfather

Stanislav Ulam
the other godfather


Nick Metropolis
the other godfather

*'We guarantee that each number*
*is random*
*individually, but we don't*
*guarantee that more than one of them*
*is random'*
Unnamed programming consultant quoted in [1]

[1] R.Kleiss@science.ru.nl

# Contents

# List of Algorithms

## The Buffon fragment

What follows is translated from the 'Buffon fragment', clay tablet in Akkadian discovered at the site of *Jebel-i-Qurul*, most likely the library of the temple precinct of the deity Nisaba, which contained a school of the *tupsar enuma Anu Enlil* scribes, who specialized in astronomy and astrology[1]; believed to be based on an earlier Sumerian original, primarily because of the reference to *absu*. Several scholars[2], however, maintain that this fragment is a forgery.

[1] A.L. Oppenheim, *Ancient Mesopotamia* (univ. of Chicago Press, 1977), p242.

[2] see, for instance, von Däniken's reference to the Book of Dzyan (1968).

### The main text

*Master:* Inside the lowly reedstalks! thou mayest find, O my disciple, the secret [of] the temple's column ; yea, verily, the secret of its girth to its width[3], from the river's reeds! To penetrate [the secret], to gain the column's wisdom, thou shalt go to the water's edge to gather reeds and bring them together, yea, even as many as thou canst gather; and [thou] shalt cut [them], so that none shall surpass the others, nor one be less than any[4]; and thou shalt also take clay from the water's edge, even as much as thou canst gather, [and] bring it to the scribe's apprentice. And [the scribe's apprentice] shall shape thereof [a tablet of] two cubits; and [the scribe's apprentice] shall draw many [lines on the tablet] so that none approach nor separate; but they shall be as an army, a well-driven host on the march[5]; and

[3] i.e. the value of $\pi$

[4] *i.e.* cut them to precisely the same length

[5] *i.e.* the lines must be strictly parallel.

the empty space[6] between [the lines] shall be as one reed[stalk], so that it neither crosses [them], nor shall it fall short: but the reed will be like unto a bridge from one line to another[7]. Thou shalt empty thy mind of all [thought], [thou shalt] void thy spirit of all purpose; and thou shalt throw [the stalks] down onto the tablet, yea, and scatter them, even like unto chaff that is scattered by the wind [on the] threshing floor[8], yea like unto the thistle's down spilt by the storm. And the reeds that cross [a line], those thou shalt gather together in thine hands, but the reeds that do not cross thou shalt not [gather together]. And the multitude in the number of the reeds in thine hands[9], thou shalt [take] yet anew[10]. And behold! it is as the column, yea, even as the girth of the column to its width [...]

*Disciple:* O my master, if I [perform] this task, and my brother [performs] this task, and all my brethren [perform] this task, shall [we] not then [approach] closer to the secret of the temple ['s column]?

*Master:* Verily, thou speakest [with] wisdom, O my [disciple]; for as a single stalk leadeth not towards knowledge, and giveth not the secret; so many [stalks] shall reveal much of [the secret]. Yet lo! the secret is revealed ever more slowly to the diligent [...][11]

## Analysis of the prescription

We assume that the stalks are straight

[6]literally, 'absu' i.e. the watery abyss

[7]*i.e.* the parallel lines must be separated by precisely one stalk's length.

[8]*i.e.* throw the reedstalks at random, with no preconcieved pattern

[9]*i.e.* the the total number of stalks divided by the number of retained stalks

[10]*i.e.* multiply by two

[11]A reference to $1/\sqrt{N}$ convergence?

lines of unit length; likwise the lines on the tablet are parallel straight lines with unit separation. Consider a reed-stalk that makes an angle $\phi$ with respect to the parallels. Its projection orthogonal to the parallels then has length $\sin(\phi)$. Supposing that the distribution of the stalks over the tablet is translationally invariant, this gives the probability that the given stalk will intersect one of the lines: note that this relies on the fact that the tablet be large enough to contain all the thrown stalks, hence the reference to 'two cubits'[12]. The procedure requires throwing the stalks with randomly chosen orientation, that is, $\phi$ is a random variable distributed uniformly between 0 and $\pi$ (note that $\phi \to \phi + \pi$ gives the same situation, except for a change in the stalk's orientation which is irrelevant). The expected probability for a given stalk to cross one of the parallels is therefore

$$\langle \phi \rangle = \frac{1}{\pi} \int\limits_0^\pi \sin(\phi)\, d\phi = \frac{2}{\pi} \quad ,$$

so that we arrive at

$$\pi = \frac{2}{\langle \phi \rangle} \quad .$$

The expected probability is measured by using many stalks and estimating $\langle \phi \rangle$ by the value of $x$, where

$$x = \frac{\text{no. of stalks crossing a line}}{\text{total number of stalks}} \quad ,$$

which proves the validity of the algorithm. The estimate for $\langle \phi \rangle$ improves with increasing number of stalks, as suggested by the disciple's question. On the other hand, as indicated by

[12]The mesopotamian cubit is about 51.86 cm, from the specimen discovered by E. Unger at Nippur (*Acta praehistorica et archaeologica* Vol 7. Berliner Gesellschaft für Anthropologie, Ethnologie und Urgeschichte, Hessling Verlag, 1976).

the master, the convergence to the exact answer is only asymptotic. The expected error in the estimate after $N$ stalks have been thrown can be computed to be

$$\begin{aligned} |x - \langle\phi\rangle\,| &\approx \sqrt{\frac{\frac{2}{\pi}\left(1-\frac{2}{\pi}\right)}{N}} \\ &\approx \frac{0.481}{\sqrt{N}} \ . \end{aligned}$$

To obtain an accuracy of 2 decimal digits (*i.e.* to get $\pi \approx 3.14$) one would need about 23,000 stalks, which may explain ‘as many as thou canst gather’. To obtain the next digit would necessitate the use of 2.3 million stalks.

The fragment breaks of in the middle of the disciple’s exclamation: ”Woe is me! The reeds are a heavy [burden], a terrible multitude, an angry host [...]”

## An early scientific application of Monte Carlo

Lord Kelvin reports [2] on an early Monte Carlo method for simulating the motion of a particle in a volume with roughened edges, in order to examine the equidistribution of energy:

‘[$\cdots$] I have evaded the difficulty in a manner thoroughly suitable for thermodynamic application such as the kinetic theory of gases. I arranged to draw lots for 1 out of the 199 points dividing AB into 200 equal parts. This was done by taking 100 cards*, 0, 1 ..... 98, 99, to represent distances from the middle point, and, by the toss of a coin, determining on which side of the middle point it was to be (plus or minus for head or tail, frequently changed to avoid possibility of error by bias). The draw for one of the hundred numbers (0 .... 99) was taken after very thorough shuffling of the cards in each case [$\cdots$]’

The footnote reads:

* ‘I had tried numbered billets (small squares of paper) drawn from a bowl, but found this very unsatisfactory. The best mixing we could make in the bowl seemed to be quite insufficient to secure equal chances for all the billets. Full sized cards like ordinary playing-cards, well shuffled, seemed to give a very fairly equal chance to every card. Even with the full-sized cards, electric attraction sometimes intervenes and causes two of them to stick together. In using one’s fingers to mix dry billets of card, or of paper, in a bowl, very

considerable disturbance may be expected from electrification.’

# 1 Introduction: randomness and probability

## 1.1 Protohistory

Monte Carlo methods are those numerical approaches to any problem in which *at least one* random number is used to obtain an outcome. The idea is not new, see for instance [2]. It came to fruitition with the advent of computers in the 1940's, mainly under the influence of WW2 efforts [3, 4]. One may distinguish Monte Carlo *integration* (of functions) and Monte Carlo *simulation* (of processes). Formally these amount to the same.

## 1.2 Random number streams

### 1.2.1 Random numbers: *circulus in probando*

The concept of what constitutes a set of random numbers is surprisingly tenuous. *Any* given set of numbers can be subjected to exhaustive analysis and so be 'shown' to be not 'fortuitous' but 'determined'[13]. Therefore, a 'true' set of random numbers should be considered rather as a *stream* of numbers, like a tap that can be turned on and, maybe very much later on, be turned off, or left running indefinitely. The idea is that it need never stop,

[13]See, *e.g.* Signor Aglié's discussion of the dimensions of a newspaper stand in U. Eco, *Il Pendolo di Foucault*. Another example is the discovery of Dr. Irving Joshua Matrix that the decimals of $\pi$, correctly interpreted, contain the complete history of the human race (as reported in an interview by Martin Gardner).

and the collection of numbers can in principle grow without limit.

ℵ Notions of randomness as ‘computational complexity’ and ’Kolmogorov complexity’ are essentially defined for finite sets.

### 1.2.2 What is a random stream? Relying on probability

The (to my mind) most operationally useful definition of ‘truly random numbers’ resides in the following description: a stream of numbers is random if, after observing $N$ numbers being produced, you will not be able to arrive at a prediction of the $(N+1)^{\text{th}}$ number to better than that given by its *probability* (for discrete random numbers), or to a prediction of its falling inside a certain interval better than given by its *probability density* (for continuous random numbers). That is, ‘you cannot beat the house’.

ℵ All definitions of random streams follow this approach if you study them closely. It ultimately leads to the frequentist interpretation of probability.

### 1.2.3 What is probability? Relying on a random stream

The (to my mind) most operationally useful definition of ‘probability’ resides in the following description: given a stream of truly random numbers it may be possible, after observing $N$ numbers being produced, to determine *probabilities*, that is, the fraction of numbers that attain a certain value (for discrete random numbers), or the fraction of numbers ending up inside a predetermined interval (for continuous random numbers). That is, 'the house will not beat you', an act of faith that can only be vindicated once $N = \infty$ has been reached and we are all dead.

ℵ This is the *frequentist* interpretation. Among other ones are the *propensity* interpretation, which is untenable without becoming frequentist, and the Bayesian, that rather describes a methodology for obtaining the probabilities.

### 1.2.4 A difference between physics and mathematics

The above notions of randomness and probability are circular and based on an operational picture of computational practice. The *mathematical* branch of probability theory, on the other hand (based on $\sigma$-algebra's and measures), is rigorous, but while it describes what you can do with probabilities, it never asks the question of what probability *means*.

ℵ The meaning of meaning is not mathematics.

## 1.3 Miscellaneous probability items

### 1.3.1 Some notation in these notes

When sums or integrals are given without limits they are understood to run over all applicable real values (from minus to plus infinity).

Probability will always refer to a probability *density*, never to the (cumulative) probability distribution beloved by mathematicians. This is because the notion of density is defined in any dimension, and distribution is not.

The logical step function $\theta(A)$ has for its argument a statement. $\theta(A)$ equals 1 if $A$ is true, and 0 if $A$ is false. If $a$ and $b$ are integers, $\theta(a = b)$ is the Kronecker delta.

The 'falling power' $N^{\underline{k}}$ is defined as $N!/(N-k)! = N(N-1)\cdots(N-k+1)$.

### 1.3.2 Moments and characteristic function

Given a probability density $P(\mathbf{x})$ with support $\Gamma$ we define the expectation value of a function $\varphi(\mathbf{x})$ by

$$\langle\varphi\rangle_P \equiv \int\limits_\Gamma d\mathbf{x}\; P(\mathbf{x})\; \varphi(\mathbf{x}) \quad ; \tag{1}$$

this is exactly what probability means[14]. The subscript $P$ is left out when no confusion can arise. For a *one-dimensional* density $P(x)$, we define the $k^{\text{th}}$ moment as $\left\langle x^k \right\rangle$. Useful notions are

$$\begin{aligned} \text{the mean} \quad &: \quad \langle x\rangle \quad , \\ \text{the variance} \quad &: \quad \sigma(x)^2 \equiv \left\langle x^2\right\rangle - \langle x\rangle^2 \quad , \\ \text{the characteristic function} \quad &: \quad \chi(z) = \chi_P(z) = \langle \exp(izx)\rangle \quad . \end{aligned} \tag{2}$$

[14]In the frequentist sense.

The last is defined if all moments are finite. $\sigma(x)$ is called the *standard deviation.* Obviously, $\chi(0) = 1$, $\chi'(0) = i\langle x \rangle$, $\chi''(0) = -\langle x^2 \rangle$, and

$$P(y) = \frac{1}{2\pi} \int dz \; \chi_P(z) \; e^{-izy} \quad . \tag{3}$$

ℵ In the following we will generally assume that all moments exist.

### 1.3.3 The Chebyshev-Bienaymé inequality

Consider a (one-dimensional) probability density $P(x)$ with finite mean $m$ and finite variance $\sigma^2$. Then, for any $a > 0$,

$$\begin{aligned} \sigma^2 &= \int dx \; P(x) \; (x-m)^2 \;\geq \int\limits_{|x-m|>a} dx \; P(x) \; (x-m)^2 \\ &\geq \int\limits_{|x-m|>a} dx \; P(x) \; a^2 = a^2 \, \mathrm{Prob}\left(|x-m| > a\right) \quad . \end{aligned} \tag{4}$$

We see that the probability for $x$ to fall further from its mean than $k$ times its standard deviation $\sigma$ is always less than $1/k^2$. Since the estimated variance, $E_2$, decreases as $1/N$ this theorem guarantees that Monte Carlo integration converges as long as the finiteness of $\langle E_2 \rangle$ is established with some confidence.

ℵ In the sense of this theorem, the probability density

$$P(x) \propto \theta(|x-m| \geq \sigma) \left( \frac{\sigma}{|x-m|} \right)^{2+\epsilon}$$

where $\epsilon$ is finite but as small as you like, is the widest possible density.

### 1.3.4 The Central Limit Theorem

Let the real numbers $x_{1,2,\ldots,n}$ be iid[15] random with density $P(x)$, and we assume that all the moments are finite. We denote $m = \langle x_j \rangle$, $\sigma = \sqrt{\sigma(x_j)^2}$. Then,

$$\xi = \frac{1}{n} \sum_{j=1}^{n} x_j \tag{5}$$

is also a random variate, with density $P_n(\xi)$. Its characteristic function is

$$\chi_{P_n}(z) = \langle \exp(iz\xi) \rangle = \prod_{j=1}^{n} \langle \exp(izx/n) \rangle = \chi_P(z/n)^n \quad , \tag{6}$$

where $\chi_P(z)$ is the characteristic function of the $x_j$. We can approximate, for large $n$,

$$\log(\chi_{P_n}(z)) \;\; = \;\; n \; \log(\chi_P(z/n))$$

[15]Independent, identically distributed.

$$\begin{aligned} &= n\ \log\left(1 + iz\left\langle x\right\rangle/n - z^2\left\langle x^2\right\rangle/2n^2 + \mathcal{O}\left(1/n^3\right)\right) \\ &= izm - z^2\sigma^2/2n + \mathcal{O}\left(1/n^2\right)\quad . \end{aligned} \tag{7}$$

Thus, for large $n$, we have approximately

$$\begin{aligned} P_n(\xi) &\approx \frac{1}{2\pi}\int dz\ \exp\left(iz(m-\xi) - \frac{z^2\sigma^2}{2n}\right) \\ &= \sqrt{\frac{n}{2\pi\sigma^2}}\ \exp\left(\frac{-n(\xi-m)^2}{2\sigma^2}\right) \end{aligned} \tag{8}$$

This is the simplest version of the Central Limit Theorem: the distribution of the average of $n$ iid random numbers with mean $m$ and standard deviation $\sigma$ approaches a Gaussian with mean $m$ and standard deviation $\sigma/\sqrt{n}$.

$\aleph$ This relies on the finiteness of the moments. A counterexample is the density $P(x) = (1+x^2)^{-1}/\pi$ for which $P_n(\xi) = P(\xi)$ for *any* $n$. Note, however, that in that case even the mean is not well defined, and the variance is not finite.

# 2 Monte Carlo integration

## 2.1 The Monte Carlo idea

### 2.1.1 Point sets and expectation values

In standard Monte Carlo integration[16] the random numbers $\mathbf{x}$ are assumed to be iid with a probability density $P(\mathbf{x})$ that has support $\Gamma$. The integrand is $f(\mathbf{x})$, and the *weight* $w(\mathbf{x})$ is given by

$$w(\mathbf{x}) = f(\mathbf{x})/P(\mathbf{x}) \quad . \tag{9}$$

The weights $w$ are also iid. The expectation values $J_n$ are given by

$$J_n = \langle w \rangle_P = \int\limits_{\Gamma} d\mathbf{x}\; P(\mathbf{x})\; w(\mathbf{x})^n \quad , \tag{10}$$

and therefore $J_0 = 1$ and

$$J_1 = \int\limits_{\Gamma} d\mathbf{x}\; f(\mathbf{x}) \quad , \tag{11}$$

which is the sought-after integral.

[16] As opposed to Quasi-Monte Carlo.

### 2.1.2 Integration as archetype

In a sense *any* Monte Carlo calculation, that is any calculation the outcome of which depends on at least one random number is an integration. Because if the outcome $\mathcal{R}$ of a calculation depends on $N \geq 1$ random numbers, $\mathcal{R} = \mathcal{F}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$, then its expected value is nothing but an integral,

$$\langle \mathcal{R} \rangle = \int d\mathbf{y}_1 \cdots d\mathbf{y}_N \; P(\mathbf{y}_1) \cdots P(\mathbf{y}_N) \; \mathcal{F}(\mathbf{y}_1, \ldots, \mathbf{y}_N) \;\; . \tag{12}$$

It is therefore sensible to concentrate on Monte Carlo integration.

$\aleph$ This is strictly formal; any serious Monte Carlo simulation easily employs many millions of random numbers, and its numerical result is therefore a multimillion-dimensional integral. Nevertheless the above point of view is useful to keep in mind.

### 2.1.3 Point sets, ensembles, and the Leap Of Faith

In the Monte Carlo approach we employ a *point set* $\mathbf{X}$ consisting of $N$ points $\mathbf{x}_j$: $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ where the $\mathbf{x}$'s are sampled from the distribution $P(\mathbf{x})$. An individual point $\mathbf{x}_j$ is called an *event*, and $w_j \equiv w(\mathbf{x}_j)$ is called the corresponding *event weight.* Given $\mathbf{X}$ we can compute

$$S_k = S_k(\mathbf{X}) = \sum_{j=1}^{N} w_j = \sum_{j=1}^{N} w(\mathbf{x}_j)^k \qquad (k = 0, 1, 2, 3, 4) \;\; . \tag{13}$$

Given the notion of a stream of random numbers, we envisage a great number ($\to \infty$) of point sets $\mathbf{X}$: the *ensemble of point sets.* Averaging over the random numbers is averaging over the ensemble, which is defined by the combined probability density it imposes on the points. For instance, for point sets defined on the $d$-dimensional hypercube $I^d = (0,1)^d$ it is simply

$$P(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = 1 \ \ . \tag{14}$$

This immediately shows the uniformity and the iid property. An important thing to remember is that in any calculation we *assume* that $\mathbf{X}$ is a 'typical' member of the ensemble, and that therefore the ensemble averages are meaningful for the given point set. This is the Leap of Faith.

ℵ The Leap of Faith can, and if possible should, be vindicated by repeating a computation with a different point set, obtained by either taking a different part of the generated random number stream, or switching to another random number generator.

## 2.2 Finding estimators

### 2.2.1 Direct sums, unequal-sums, and expectation values

We consider $N$ iid random numbers $w_j$ ($j = 1, 2, \ldots, N$) with expectation values

$$\left\langle {w_j}^k \right\rangle \equiv J_k \ \ . \tag{15}$$

We define the direct sum $S_m$ $(m = 0, 1, 2, \ldots)$ as

$$S_m = \sum_{j=1}^{N} (w_j)^m \quad : \tag{16}$$

these sums are computable in *linear* time, $\mathcal{O}(N)$. $S_0$ is simply equal to $N$. The unequal-sums $S_{m_1,m_2,\ldots,m_k}$ are defined as

$$S_{m_1,m_2,\ldots,m_k} = \sum_{j_1,j_2,\ldots,j_k=1}^{N} (w_{j_1})^{m_1} (w_{j_2})^{m_2} \cdots (w_{j_k})^{m_k} \tag{17}$$

with the constraint that the indices $j_1, \ldots, j_k$ are all *different* from one another. The unequal-sum $S_{m_1,m_2,\ldots,m_k}$ therefore contains $N^{\underline{k}}$ terms. Its straightforward computation takes time $\mathcal{O}\left(N^k\right)$. By the iid assumption we have

$$\langle S_m \rangle = N\, J_m \quad , \quad \langle S_{m_1,m_2,\ldots,m_k} \rangle = N^{\underline{k}}\, J_{m_1} J_{m_2} \cdots J_{m_k} \quad . \tag{18}$$

The falling powers $N^{\underline{k}}$ are discussed in appendix 11.1.1. We can relate products of direct sums to combinations of unequal-sums by the following rule:

$$\begin{aligned} S_{m_1,m_2,\ldots,m_k}\, S_p &= S_{m_1+p,m_2,\ldots,m_k} + S_{m_1,m_2+p,\ldots,m_k} + \cdots \\ &\quad \cdots + S_{m_1,m_2,\ldots,m_k+p} + S_{m_1,m_2,\ldots,m_k,p} \quad . \end{aligned} \tag{19}$$

Explicit relations are given in appendix 11.1.2.

ℵ This provides a way to evaluate unequal-sums in linear time, and a way to find expectation values of nonlinear combinations of sums. Conversely, we can find the combination of direct sums that has a given expectation value.

### 2.2.2 The Monte Carlo estimators

Using the direct sums $S_k$ we define three *Monte Carlo estimators* :

$$\begin{aligned}
E_1 &= \frac{1}{N} S_1 \quad , \\
E_2 &= \frac{1}{N N^{\underline{2}}} \left( N S_2 - {S_1}^2 \right) \quad , \\
E_4 &= \frac{N^{\underline{2}}}{N^3 N^{\underline{4}}} \left( N S_4 - 4 S_3 S_1 + 3 {S_2}^2 \right) \\
&\quad + \frac{1}{N^{\underline{4}}} \left( \frac{2}{N^2 N^{\underline{2}}} - \frac{4}{N^3} \right) \left( N S_2 - {S_1}^2 \right)^2 \quad .
\end{aligned} \tag{20}$$

Since the point set $\mathbf{X}$ is an element of the ensemble of point sets, the numbers $E_{1,2,4}$ are also (one-dimensional) random numbers with their own mean and variance[17]. Using 11.1.2 we can prove

$$\langle E_1 \rangle = J_1 \quad ,$$

[17]In the sense that every new chosen point set $\mathbf{X}$ yields its own values for $E_{1,2,4}$.

$$\begin{aligned}\langle E_2\rangle &= \sigma(E_1)^2 = \frac{1}{N}\left(J_2 - {J_1}^2\right) \quad , \\ \langle E_4\rangle &= \sigma(E_2)^2 = \frac{1}{N^3}\left(J_4 - 4J_3J_1 + 3{J_2}^2\right) \\ &\qquad + \left(\frac{2}{N^2 N^{\underline{2}}} - \frac{4}{N^3}\right)\left(J_2 - {J_1}^2\right)^2 \quad . \end{aligned} \tag{21}$$

We see that (in an ensemble sense) $\langle E_1\rangle$ is the desired integral $J_1$. $E_2$ informs about the ensemble variance in the probability density of $E_1$ values, and it has its own ensemble variance, estimated by $E_4$. Note that $\langle E_2\rangle < \infty$ if the integrand is *quadratically integrable*, but $\langle E_4\rangle < \infty$ only if the integrand is *quartically integrable*.

ℵ There is also an $E_8$ with $\langle E_8\rangle = \sigma(E_4)^2$, and so on. These ever more complicated estimators are not very relevant. $E_1$ and $E_2$ are well known, and $E_4$ *deserves to be.*

### 2.2.3 Positivity of $E_2$ and $E_4$

In the Monte Carlo integral we have $\langle w(\mathbf{x})\rangle = J_1$. Now write $u(\mathbf{x}) = w(\mathbf{x}) - J_1$. Then we have

$$J_2 - {J_1}^2 = \int d\mathbf{x}\; P(\mathbf{x})\; u(\mathbf{x})^2 \quad ,$$
$$J_4 - 4J_3J_1 + 3{J_2}^2 - 4\left(J_2 - {J_1}^2\right)^2 =$$

$$\frac{1}{2}\int d\mathbf{x}\,d\mathbf{y}\;P(\mathbf{x})\,P(\mathbf{y})\;\left(u(\mathbf{x})^2 - u(\mathbf{y})^2\right)^2 \tag{22}$$

so both $E_2$ and $E_4$ have positive expectation value[18]. For a given $\mathbf{X}$, define $u_j = w(\mathbf{x}_j) - J_1$. Then

$$NS_2 - {S_1}^2 = \frac{1}{2}\sum_{j,k=1}^{N} (u_j - u_k)^2 \geq 0 \;\; . \tag{23}$$

So the estimator $E_2$ will always result in a positive number[19]. However, suppose that $w_j$ only takes the values 0 and 1, so that $S_k = Nb$ for all $k > 0$, where $0 < b < 1$. Then, $E_4$ evaluates to a negative number if $b(1-b) > 1/4 - (N-2)/2N(4N-6)$, so $E_4$ cannot be guaranteed to be nonnegative. On the other hand,

$$N^2\left(NS_4 - 4S_3S_1 + 3{S_2}^2\right) - 4\left(NS_2 - {S_1}^2\right)^2 = \frac{N^2}{2}\sum_{j,k=1}^{N}\left({u_j}^2 - {u_k}^2\right)^2 \geq 0 \;\; . \tag{24}$$

This suggests a slight modification of the estimators $E_{2,4}$.

[18]As they should!

[19]Barring numerical accidents.

## 2.3 Estimating in practice

### 2.3.1 Improved estimators

We can use the following 'improved' estimators:

$$
\begin{aligned}
E_1 &= \frac{1}{N} S_1 \quad , \\
\hat{E}_2 &= \frac{1}{N^3}\left(N S_2 - {S_1}^2\right) \quad , \\
\hat{E}_4 &= \frac{1}{N^7}\left(N^2\left(N S_4 - 4 S_3 S_1 + 3 {S_2}^2\right) - 4\left(N S_2 - {S_1}^2\right)^2\right) \quad . \qquad (25)
\end{aligned}
$$

These are equal to the exact ones up to $1/N$ corrections, and are nonnegative by construction.

$\aleph$ Since in any serious calculation $N$ is of the order of at least a few thousand, a $1/N$ correction in uncertainty estimates is not a big deal.

### 2.3.2 Numerical stability; the extended CGL algorithm

Computing $E_{2,4}$ involves large cancellations. Even $E_2$ can almost never be computed using single precision if Eq.(25) is used[20]. The following algorithm avoids this. Suppose $n-1$ random numbers have been used already, and the

[20]This is really true, as *anyone* who tried it seriously will testify.

contribution $w_n$ of the $n^{\text{th}}$ is applied. Define

$$U_k(n) = \frac{1}{n}\sum_{j=1}^{n} {w_j}^k \tag{26}$$

and

$$\begin{aligned}
M(n) &= U_1(n) \ , \\
P(n) &= U_2(n) - U_1(n)^2 \ , \\
Q(n) &= U_3(n) - 3\,U_2(n)U_1(n) + 2\,U_1(n)^3 \ , \\
R(n) &= U_4(n) - 4\,U_3(n)U_1(n) + 3\,U_2(n)^2 - 4\,P(n)^2 \ ,
\end{aligned} \tag{27}$$

and also

$$m = M(n-1) \ , \ p = P(n-1) \ , \ q = Q(n-1) \ , \ r = R(n-1) \ . \tag{28}$$

Let us also define $u = w_n - m$. Then we can update in linear time as follows:

$$\begin{aligned}
M(n) &= m + u/n \ , \\
P(n) &= \frac{n-1}{n}\left(p + \frac{u^2}{n}\right) \ , \\
Q(n) &= \frac{n-1}{n}\left(q + \frac{n-2}{n^2}u^3 - \frac{3p}{n}u\right) \ , \\
R(n) &= \frac{n-1}{n}\left(r + \frac{1}{n}\left(p - \frac{n-2}{n}u^2\right)^2 - 4\left(\frac{q}{n}u - \frac{p}{n^2}u^2)\right)\right) \ .
\end{aligned} \tag{29}$$

This algorithm is useful since (a) it can be used to update in constant time, and (b) it is free of large cancellations.

ℵ The use of $M$ and $P$ in the computation of $\hat{E}_2$ is the original CGL algorithm [5]. The $Q$ and $R$ necessary for $\hat{E}_4$ is discussed in Bakx *et al.* [6].

### 2.3.3 How to *do* Monte Carlo integration

The recommended way to compute a Monte Carlo estimate of the integral

$$J_1 = \int_{\Gamma} d\mathbf{x}\ f(\mathbf{x}) \tag{30}$$

is then as follows: generate a point set $\mathbf{X}$ of iid random numbers $\mathbf{x}_j$ ($j = 1, \ldots, N$) with density $P(\mathbf{x})$. Every time a new point $\mathbf{x}_n$ is added, compute $w(\mathbf{x}_n) = f(\mathbf{x}_n)/P(\mathbf{x}_n)$. Then, update $M, P, Q$, and $R$. After each point, the running ('improved') estimates are given by

$$E_1 = M(n) \ \ , \ \ \hat{E}_2 = P(n)/n \ \ , \ \ \hat{E}_4 = R(n)/n^3 \ \ . \tag{31}$$

Below we give the algorithm in pseudocode

ℵ The updating allows for monitoring how the calculation progresses. This is a very important technique to gauge the quality of the computation, as we shall see.

**Algorithm 1** Extended CGL algorithm for numerically safe updating of running Monte Carlo estimators

```
{ At every call to this algorithm an event weight $w$ is inputted. The
number $n$ is the number of event weights inputted so far. The estimators
$\hat{E}_{1,2,4}$ are the output. The numbers $M, P, Q, R$ are kept internally.}
if $n = 0$ then
    $[M, P, Q, R] \leftarrow [0, 0, 0, 0]$          {Initialization}
end if
$[m, p, q, r] \leftarrow [M, P, Q, R]$        {Variables used in the update}
$n \leftarrow n + 1$
$u \leftarrow w - m$
$M \leftarrow m + u/n$
$P \leftarrow (n-1)(p + u^2/n)/n$
$Q \leftarrow (n-1)(q + (n-2)u^3/n^2 - 3pu/n)/n$
$R \leftarrow (n-1)(r + (p - (n-2)u^2/n)^2/n - 4(qu/n - pu^2/n^2))/n$
$[\hat{E}_1, \hat{E}_2, \hat{E}_4] \leftarrow [M, P/n, R/n^3]$        {The estimators so far}
```

### 2.3.4 How to *report* Monte Carlo integration

After a point set $\mathbf{X}$ of $N$ points has been used to do a Monte Carlo integral, the estimate of the integral is $E_1$. The estimate of its ensemble variance is $\hat{E}_2$, and the estimate of the ensemble variance of $E_2$ is $\hat{E}_4$. The best way to report the result of the computation is

$$J_1 = E_1 \pm \left( {E_2}^{1/2} \pm {E_4}^{1/4} \right) \quad . \tag{32}$$

### 2.3.5 How to *interpret* Monte Carlo integration

The 'Monte Carlo error' is given by $\hat{E}_2^{1/2}$. If the Central Limit Theorem holds, this gives the Gaussian *confidence levels* associated with the answer. The 'error on the error' $\hat{E}_4^{1/4}$ informs about the uncertainty in the confidence levels, which can be quite important.

We have, for a given integrand,

$$\frac{\hat{E}_2^{1/2}}{E_1} \sim \frac{1}{N^{1/2}} \quad . \tag{33}$$

This is the well-known, slow but universal[21] behaviour of Monte Carlo integration. Not so well known but important is the relative uncertainty of the

[21]In particular, independent of the dimensionality of $\Gamma$.

error:

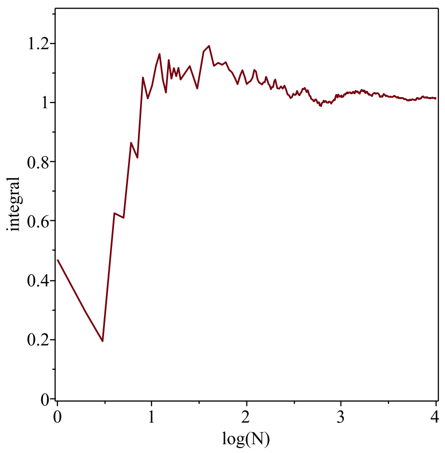
$$\frac{\hat{E}_4^{1/4}}{\hat{E}_2^{1/2}} \sim \frac{1}{N^{1/4}} \ . \tag{34}$$

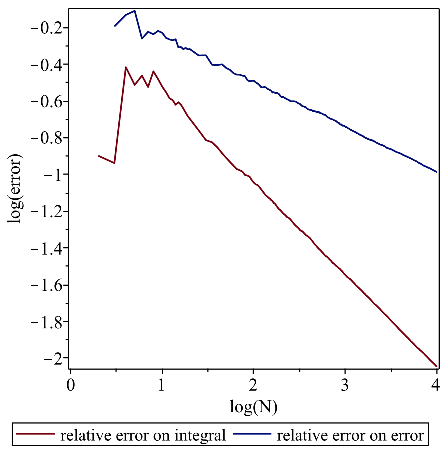
The convergence of the error estimate is much slower that that of the integral. The error can be translated (in the Central Limit Theorem sense) into Gaussian confidence intervals. Here the error on the error becomes important. The $1\sigma$ confidence level (two-sided) is 68.2%; if the relative error on the error is 30 per cent (admittedly quite large) the confidence level can range from 51.6% to 80.6%. It is important to compute $\hat{E}_4$ even if only to confirm that the error on the error is small.

ℵ The error on the error is not widely known, unfortunately. Experience shows that it can be uncomfortably large.

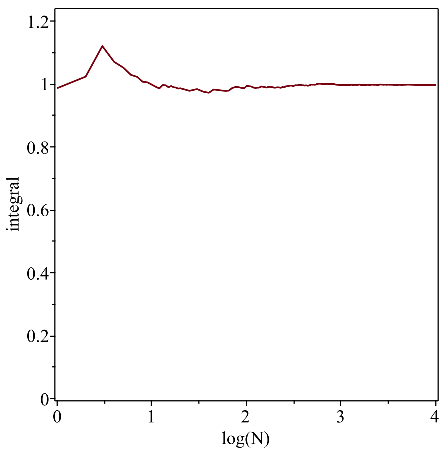
## 2.4 A test case

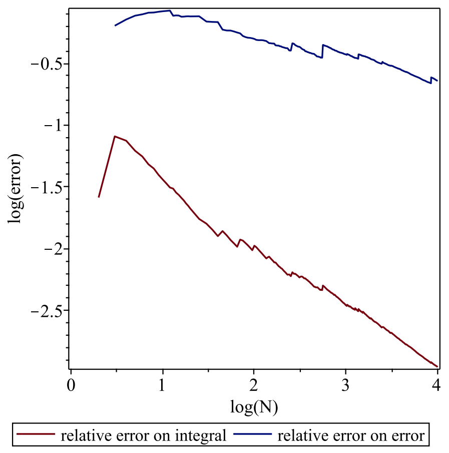
We consider the MC integration of the function $f(a;x) = (1+a)\,x^a\theta(0 < a \leq 1)$, with $a > -1$, for various decreasing values of $a$. The *real* value of the integral is unity. We use the same set of 10000 (pseudo)random numbers for all plots.



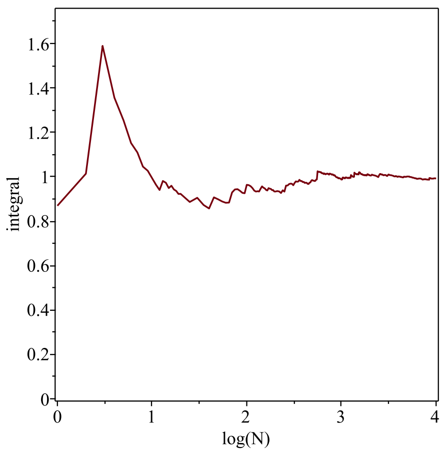


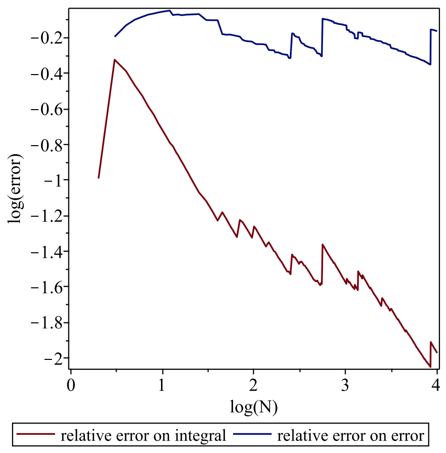
For $a = 2$ the integrand is bounded and perfectly integrable. The estimates of the relative first- and second-order error decrease as $N^{-1/2}$ and $N^{-1/4}$ respectively.



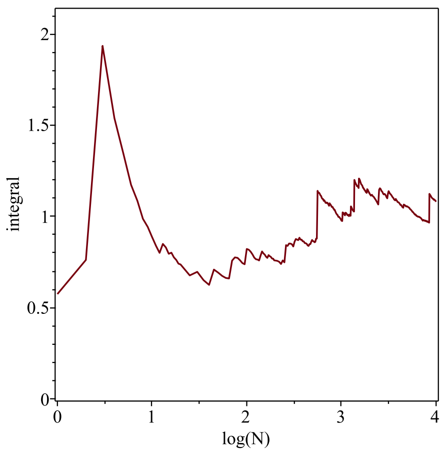


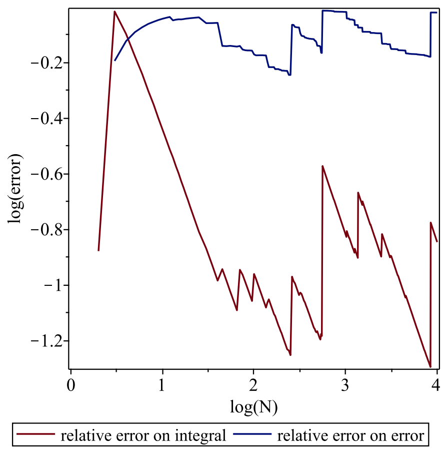
For $a = -0.1$ the integrand is no longer bounded but integrable and also square and quartically integrable. The The first-order error still decreases smoothly, but the second-order error remains quite large (22% at $N = 10000$). The small jumps in the curves arise from MC points close to zero.



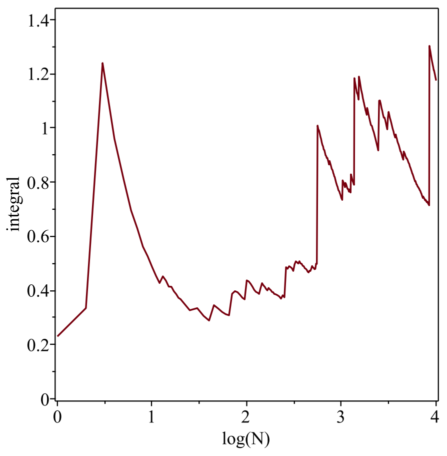


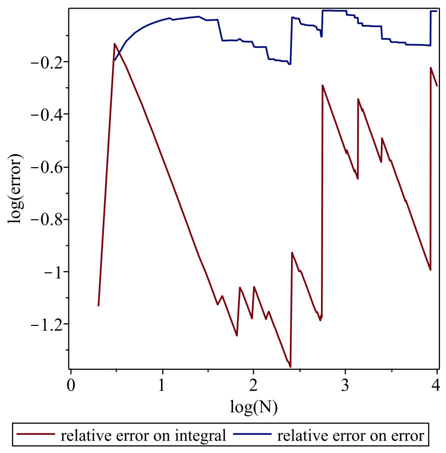
For $a = -0.4$ the integrand is no longer quartically integrable, witness the sizeable jumps in the error curves. The second-order error cannot really be said to decrease at all even though the estimated error at $N = 10000$ is still a reasonable 1%, but its own error is around 60% (relative).





For $a = -0.7$ the integrand is no longer square integrable, and the first-order error estimate cannot be trusted at all, from the fact that here $E_4^{1/2}/E_2$ is around unity.





For $a = -0.9$ integrability is almost completely lost: $E_2$ jumps all over the place and even $E_1$ cannot be said to converge

From these simple tests a number of conclusions can be drawn. In the first place, integration problems like these show, in the development of the estimators, the phenomenon of '*punctuated equilibrium*': the *a priori* behaviour of $E_2$ and $E_4$ is still as $1/N$ and $1/N^3$, respectively, but interspersed with jumps that become more pronounced as the integrand becomes more and more singular[22]. Eventually these jumps counteract the decreasing of the estimators. In the second place, we see that in any MC integration one should not be content with the final numbers alone, but the integration should be monitored while it proceeds. Jumps signal singularities!

## 2.5 Exercises

**Excercise 1 The CGL algorithm**
Write your own Monte Carlo evaluator. This should be a code in which (after initialisation) weights can be inputted one after the other, while a continuous estimate of $\hat{E}_{1,2,4}$ is updated using the CGL algorithm(1). That is, after a number of weights have been inputted, the code should be able to give the Monte Carlo estimators at any moment during the computation.

**Excercise 2 Studying the test case**
Using the random number generator of your choice, perform your own study of the test case of sect.2.4 by using the code of the previous exercise, and playing around with the parameter $a$. You may also want to consider some other function with an adjustable amount of singularity.

**Excercise 3 For diehard combinatoricists only**
Try to find the variance of $\hat{E}_4$, and find the estimator $E_8$ that gives 'the uncertainty in the error estimate on the error'. It involves things like $S_{1,1,1,1,1,1,1,1}$ and $S_{3,2,2,1}\dots$

[22]A similar notion from a quite distant field is found in [7].

# 3 Random number generation

## 3.1 Introduction to random number sources

### 3.1.1 Natural *vs* Pseudo

Random streams can be obtained from natural processes, or by computer. In the last case there *is* an algorithm and the 'next' number is entirely predictable[23]. Such streams aim not at *being* random but at *appearing* (sufficiently) random, and are called *pseudorandom* numbers streams[24]. Both methods have advantages and drawbacks.

1. Natural random number streams: electronic noise, chaotic lasers, air pressure fluctuations, photonics, . . .

    - Advantages: unpredictability
    - Drawbacks: unrepeatability (in all cases), speed (in some cases), unguaranteed probability density (in all cases)

    The picture shows IDQuantique's Quantis-USB-4M Quantum Random Number Generator[25]. It produces 'truly random' bits by observing indi-

[23]By those who are in on the secret, or on the code.
[24]Greek $\psi\epsilon\upsilon\delta\epsilon\iota\nu$, 'to lie'.
[25]https://www.idquantique.com

vidual photons that are or are not reflected by a half-reflecting mirror. I timed (by hand) approximately 21 seconds to generate 80 Mbit and about the same time for $2 \times 10^4$ 10-digit integers or 10-digit floating point numbers in $(0, 1)$. For such scientific tasks as Monte Carlo integration this is typically much too slow. Its main application is cryptographical; there, true unpredictability and unrepeatability are essential.

ℵ *I think that the direct use of a physical supply of random digits is absolutely inacceptable for this reason and for this reason alone.*
John von Neumann on the irreproducibility of natural random numbers [26].

2. Pseudorandom number streams: computer algorithms

   - Advantages: understandability, repeatability

- Drawbacks: predictability, speed (in some cases), nonuniformity (in some cases)

ℵ *Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin.*
John von Neumann on the predictability of pseudo-random numbers [26].

## 3.2 Pseudo-random number streams

### 3.2.1 The structure of pseudo-random number algorithms

Consider algorithms working on the set of integers $a_j \in \{1, 2, \ldots, M\}$, $j = 1, 2, 3, \ldots$. Pseudorandom number generators[26] always generate the next number as a deterministic function of one or more previous ones. We shall first examine the algorithm

$$a_{n+1} = f(a_n) \quad . \tag{35}$$

Given a starting value $a_1$ it generates the stream

$$a_2 = f(a_1) \quad , \quad a_3 = f(a_2) \quad , \quad f(a_4) = f(a_3) \quad , \ldots \tag{36}$$

If $f(a_L) = a_p \in \{a_1, a_2, a_3, \ldots, a_L\}$ (*i.e.* a number comes up that was already generated before) the series $(a_p, a_{p+1}, \ldots, a_L)$ will start repeating itself and

[26]Commonly denoted as PRNGs.

pseudorandomness is lost. $L$ is the *lifetime* of the algorithm $f$ for starting value $a_1$. The series $(a_1, a_2, \ldots, a_{p-1})$ can be called the *runup*, and the series $(a_p, \ldots, a_L)$ the *cycle*.

### 3.2.2 The set of all algorithms

The above algorithm $f$ can be *completely* specified by the list

$$S_f = \Big[ f(1), f(2), f(3), \ldots, f(M-1), f(M) \Big] \quad . \tag{37}$$

Many different-looking algorithms can end up with the same $S_f$. A few conclusions immediately follow:

1. There are precisely $M^M$ possible algorithms of the form (35).
2. The maximum possible lifetime is $M$.
3. The number of algorithms with the maximum lifetime for a *given* starting value is $M!$.
4. The number of algorithms with the maximum lifetime for *any* starting value is $(M-1)!$, the number of permutations with no subcycles. These have no runup.

### 3.2.3 The concept of an arbitrary algorithm

We can pick a 'arbitrarily chosen algorithm' by choosing $S_f$ randomly out of the $M^M$ possibilities, and so arrive at probabilistic statements about 'arbitrary' algorithms[27]. The probability to have $a_2 = f(a_1) \neq a_1$ is $1 - 1/M$; the probability to have $a_3 = f(a_2) \neq a_{1,2}$ is $1 - 2/M$, and so on: the probability of having a lifetime $L$ of *at least* $k$ is therefore[28]

$$\text{Prob}\,(L \geq k) = \frac{M^{\underline{k}}}{M^k} \quad . \tag{38}$$

The probability of having a lifetime of precisely $k$ is

$$\text{Prob}\,(L = k) = \text{Prob}\,(L \geq k) - \text{Prob}\,(L \geq k+1) = \frac{k\,M^{\underline{k}}}{M^{k+1}} \quad . \tag{39}$$

The expected lifetime given a fixed arbitrary starting value $a_1$ is

$$\begin{aligned}\sum_{k=1}^{M} k\text{Prob}\,(L = k) &= \sum_{k\geq 1} k\text{Prob}\,(L \geq k) - \sum_{k \geq 1} k\text{Prob}\,(L \geq k+1) \\ &= \sum_{k\geq 1} \text{Prob}\,(L \geq k) = \sum_{k\geq 1} \frac{M^{\underline{k}}}{M^k}\end{aligned}$$

[27]The phrasing 'random' algorithm may lead to confusion.

[28]Correctly, this gives a probability zero for a lifetime longer than $M$.

$$
\begin{aligned}
&= \int_0^\infty dx\, e^{-x} \sum_{k\geq 1} \frac{x^k\, M^{\underline{k}}}{k!\, M^k} \\
&= \int_0^\infty dx\, e^{-x} \left(-1 + \left(1 + \frac{x}{M}\right)^M\right) \\
&= \sqrt{\frac{\pi M}{2}} - \frac{1}{3} + \sqrt{\frac{\pi}{288\, M}} + \mathcal{O}\left(\frac{1}{M}\right) \quad , \qquad (40)
\end{aligned}
$$

where the last line is derived in appendix 11.2.2.

### 3.2.4 Lessons from random algorithms

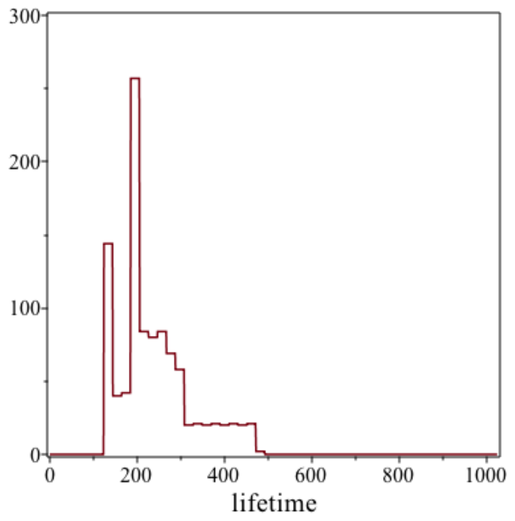
A number of probabilistic remarks hold for randomly chosen algorithms.

- The number $M$ should be very large, since true random number streams correspond to $M \to \infty$.
- The probability of picking an algorithm with lifetime $L = M$ is $M!/M^M \approx \sqrt{2\pi M}\ \exp(-M)$, *i.e.* extremely small.
- The expected lifetime is of order $\sqrt{M}$, so very much smaller than the maximum lifetime.
- The probability of occurrence of at least one *selfie*, a number $a$ such that $f(a) = a$, is *not* small, around 63% (see appendix 11.3.1).

We can draw a number of inferences that are important for practical PRNGs.

- One should aim for long lifetimes. This is of course only the very crudest requirement on a PRNG.
- Good algorithms are rare, and 'lie close to' bad or mediocre algorithms.
- Rounding errors are to be avoided[29]. This is why in many cases the internal arithmetic is done with integer $a_j$, and only at the output stage the floating-point number $a_j/M \in (0,1]$ is returned.
- For an algorithm with maximum lifetime, at the end of the lifetime *all* numbers will have been generated exactly *once*. This does not look random at all. One should avoid coming close to exhausting the whole lifetime. In practice, if you envision using $n$ random numbers, then $L$ should be at least of order $n^2$.

[29]Rounding errors mean that the algorithm does not do *exactly* what it is meant to do, and so deviates from a possible good algorithm into the jungle of bad algorithms.



This plot illustrates the effect of a slight corruption. Here I have taken an algorithm with maximum lifetime 1024 for *all* starting values, and corrupted it by increasing the values of $f(n)$ by 1 at 5 randomly selected places, thus mimicking the effect of rounding errors and such. The resulting algorithm is therefore 'close' to the original one. However, the distributions of the lifetimes for all starting points given here indicates that the corrupted algorithm performs much worse. The average lifetime is about 237, and the maximum is only 473.

### 3.2.5 Shift-register PRNGs

One can let the next number $a_n$ depend not only on $a_{n-1}$ but on a register of size $r$:

$$a_n = f(R_n) \quad , \quad R_n = (a_{n-1}, a_{n-2}, \ldots, a_{n-r+1}, a_{n-r}) \ . \tag{41}$$

The *next* number will be

$$a_{n+1} = f(R_{n+1}) \quad , \quad R_{n+1} = (a_n, a_{n-1}, \ldots, a_{n-r+2}, a_{n-r+1}) \ ; \tag{42}$$

$a_n$ has entered from the left, and $a_{n-r}$ has dropped out on the right : hence the name *shift-register* PRNG. This is in fact just an artifice to enlarge $M$ since we can map the register to integers by

$$R_n \leftrightarrow 1+(a_{n-1}-1)+M(a_{n-2}-1)+M^2(a_{n-3}-1)+\cdots M^{r-1}(a_{n-r}-1) \quad , \tag{43}$$

which is an integer between 1 and $M^r$. The use of the register is just a way of writing very large numbers in base $M$, and only using the leading 'digit' as the random number. The maximum possible lifetime is now $M^r$.

$\aleph$ Shift-register PRNGs can be deceptively simple : an algorithm like $a_n \sim a_{n-s} \pm a_{n-r}$ $(1 \leq s < r)$ is often already quite good.

## 3.3 Bad and good algorithms

### 3.3.1 Bad: the midsquare algorithm

This is one of the oldest attempts at a PRNG algorithm. Of the square of a (not too small) integer, the *last* digit(s) are easily predictable, as are the first few digits. By taking squares of $2d$-digits numbers, and retaining the middle $2d$ digits of the result (adding zeroes if leading digits in the $4d$-digit square are missing), one hopes to achieve a 'difficultly-predictable' sequence. If $K$ is the maximum integer as before it would read, using the 'floor' function,

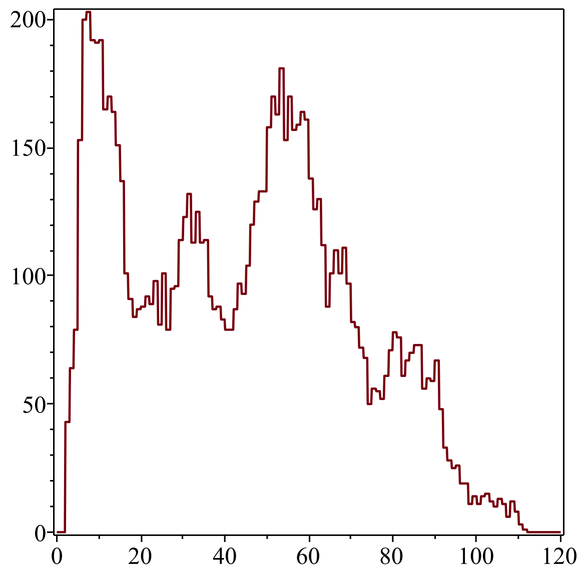
$$a_n = \left\lfloor a_{n-1}^2 \;/\; \sqrt{K} \right\rfloor \mod K \quad . \tag{44}$$

Typically $K$ would have to be a perfect square[30]. For $K = 100$ this would give, for instance the sequence

$$63, 96, 21, 44, 93, 64, 9, 8, 6, 3, 0 \quad .$$

Direct inspection tells us that, for $K = 100$ : (*i*) there are 4 selfies, to wit 00, 10, 50, and 60 ; (*ii*) 61 sequences end in 0 ; (*iii*) the largest lifetime is 15, achieved for the starting value 42 (!) ; (*iv*) the average lifetime is 5.76, way below the 'expected' value of about 12.

[30]This is not strictly necessary, since the floor function erases rounding errors in the $\sqrt{K}$ unless $K$ is really huge.



Here we give the distribution of lifetimes for $K = 10^4$ for all starting values from 1 to 9999. The maximum lifetime is 111 (achieved for the starting value 6239), not even equal to the expected value of around 125. The midsquare method can be considered a typical example of an 'aribitrary' algorithm.

ℵ The midsquare method with $K = 10^{10}$ has been used with broadly satisfactory results in the early 1950's[a] but should be considered totally obsolete for modern applications.

[a]As reported by P.C.Hammer in [4], p33. A lifetime of at least $10^4$ was found for starting value 1111111111.

### 3.3.2 Bad: the chaos approach and the logistic map

The chaotic behaviour of certain dynamical systems would seem to be a source of 'random', or at least unpredictable, sequences. A good example is provided by the fully chaotic *logistic map*. This is based on real numbers rather than integers:

$$x_n = 4\,x_{n-1}\,(1 - x_{n-1}) \quad . \tag{45}$$

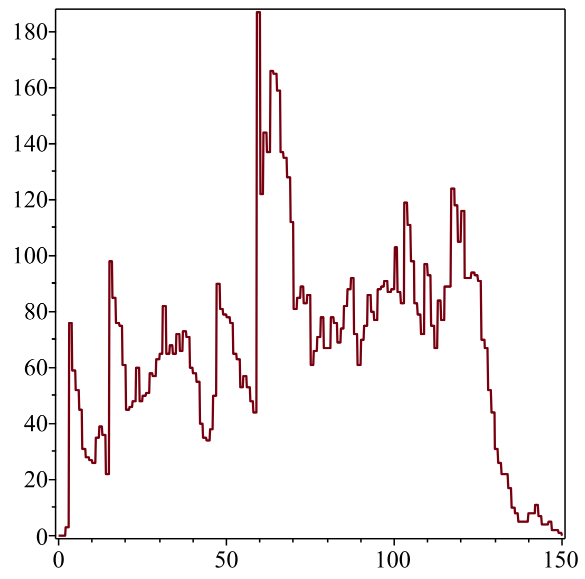
It is easy to see that this will almost certainly yield a sequence with infinite lifetime. If we write

$$x_n = \sin^2\left(\frac{\pi}{2} y_n\right) \quad , \quad 0 \le y_n \le 1 \quad , \tag{46}$$

The map (45) corresponds to

$$y_n = 2y_{n-1} \mod 1 \quad . \tag{47}$$

Now, every non-rational number in $(0, 1)$ has a binary expansional that does not repeat. Therefore, the logistic map (45) will have infinite lifetime — in mathematical principle. However, we can only use finite-precision floating-point numbers. Therefore if $y_n$ is given with (say) 100 binary digits' precision, then iterating (47) would give $y_{n+101} = 0$, a selfie. Of course we do not use the $y$'s but the $x$'s but this means that the selfie $x = 0$ is only avoided because of rounding errors ! Consequently we expect the logistic map (in finite precision) to be an 'arbitrary' algorithm. The same holds for all dynamical systems that are chaotic : the very fact that their behaviour depends extremely sensitively on the initial conditions guarantees that in practice it is driven by rounding errors.



The lifetimes of the logistic-map algorithm for numbers restricted to 4 decimal digits, for all possible starting values. The maximum observed lifetime is 149, a little better than the expectation value 125.

ℵ An even stronger statement about binary (or decimal) expansions is possible. Almost all real numbers (in the sense of Lebesgue measures) are *normal*, that is their binary expansion contains every given block of digits (such as 0, 101, or 1001010110011) with asymptotically the correct frequency. This means that iterating (47) would give a perfect stream of random bits. Unfortunately nobody knows how to construct a normal number. $\pi$ may be normal but that has not been proven yet.

### 3.3.3 Maybe not so bad: linear congruential methods

A very popular and simple (hence analysable!) algorithm is the *linear congruential* method. In terms of integers $x_n$ $(n = 0, 1, 2, \ldots)$ it reads

$$x_{n+1} = \Big(a\, x_n + c\Big) \bmod m \quad , \quad a, c < m \quad . \tag{48}$$

The *modulus* $m$, the *multiplier* $a$, and the *increment* $c$ are integers. One of the advantages is that the period (and other properties) can be determined [8]. The maximum lifetime (in this case, period) is obtained under the so-called Hull-Dobell conditions [9]:

1. $c$ and $m$ are relatively prime;

2. every prime factor of $m$ divides $a - 1$;

3. if $m$ is a multiple of 4, then so must $a-1$ be.

When $c = 0$ the maximum period cannot be achieved[31]. In that case the maximum period can be $m/4$ if $m$ is a power of 2, or $m-1$ if $m$ is a prime, everything depending on the optimal choice of $a$ [10][32]. A final remark: you might think that taking $m$ to be prime is clever. But then note that only $a = 1$ can give the full period. If $m$ is prime there is always a selfie for $a > 1$ (see also appendix 11.3.2); for *some* $a$ a period of length $m-1$ can be achieved. The same holds if $m$ is the product of *distinct* primes. It is therefore advisable to let $m$ be consist of largish powers of primes: obviously $m = 2^p$ is a possibility.

It is interesting to note the following. Suppose $m = 2^n$, and $a = 2^p + q$, with $p > n/2$. Then

$$x_n = \left(2q\, x_{n-1} - q^2\, x_{n-2}\right) \bmod m \quad , \tag{49}$$

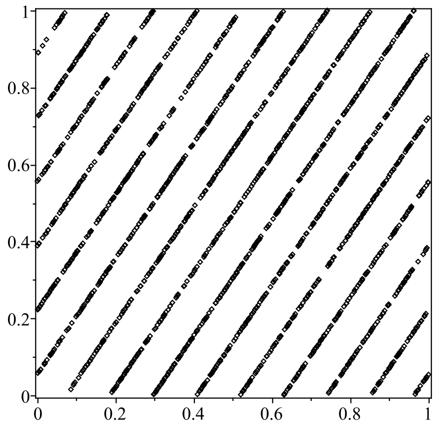
so this algorithm can be viewed as a shift-register PRNG as well. It also means that the triples $[x_{3k-2}, x_{3k-1}, x_{3k}]$ all lie on a 2-dimensional plane. Using the $\bmod m$ this plane intersects the 3-dimensional $m \times m \times m$ cube a number of times, so forming a collection of planes on which all generated triples must fall. This is easily extended to larger multiplets of points in hypercubes. Such structures of multiples of points are common to all multiplicative congruential PRNGs [11], the *maximum* number of planes in a

[31]Since then $x_n = 0$ is a selfie.

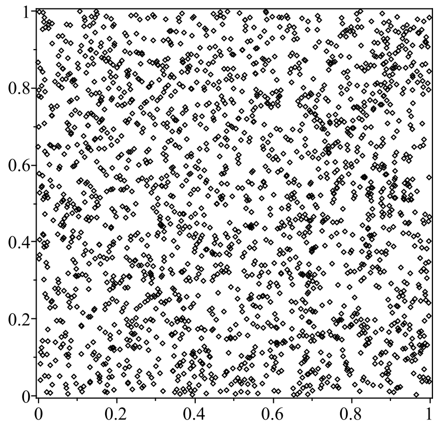
[32]The indispensable reference here is [8].

$d$-dimensional hypercube being $(d!m)^{1/d}$. This limits the usefulness of multiplicative congruential generators, even when the period length $m$ is acceptable.

### 3.3.4 Horrible: `RANDU`, and a possible redemption



Cube slice for `RANDU`



Cube slice for $a = 69069$.

For many years the PRNG called `RANDU` was popular[12]. It has $m = 2^{31}$, $c = 0$ and $a = 2^{16} + 3 = 65539$. Typically, $x_1$ is chosen to be 1. These values are mainly inspired by simplicity of implementation on a 32-bit machine. The choice of $a$ has turned out to be about the worst possible; the number of planes (as discussed above) is very small in low dimensions. We can visualise this by, say, taking a slice of the cube with $0.32 \le x_{3k} \le 0.34$. `RANDU` gives only 15 planes! Notice that for such features to become obvious it was necessary for easy 2-d plotting to become available. A much better choice appears to be $a = 69069$ [13][33].

### 3.3.5 Good: `RCARRY` and `RANLUX`

An example of a shift-register PRNG that does better than a multiplicative congruential one, with a larger register, is `RCARRY`[14]. Its parameters are the modulus $B$, and two index parameters $s$ and $r$, the latter being the register length. We use $B = 2^{24}$, $s = 10$ and $r = 24$. In addition there is a so-called *carry bit* $c_n$ associated with the pseudorandom integer $x_n$. The complete register therefore reads

$$[x_{n-1}, \ldots, x_{n-s}, \ldots, x_{n-r}; c_{n-1}]$$

and the algorithm is

$$x_n = (x_{n-s} - x_{n-r} - c_{n-1}) \mod B \tag{50}$$

where $c_n = 0$ or $c_n = 1$ according to whether or not the $\mod B$ was necessary. In order to avoid having to build the register anew every time, it is best to view it as cyclic with the starting point at the back.

[33]Viewed as shift-register PRNGs we have for `RANDU` $x_n = 6x_{n-1} - 9x_{n-2}$. For the redemption value $a = 69069$ we have $x_n = 7066x_{n-1} - 12482089x_{n-2}$. The coefficients are far larger, but the register length is still only 2.

**Algorithm 2** The `RCARRY` algorithm using a cyclic register.

```
{The register is $[a_1, a_2, a_3, \ldots, a_r]$, and its *last entry* is currently $a_j$. The current value of the carry bit is $c$. The new pseudorandom number is $x/B$.}
$p \leftarrow s + j$, if $p > r$ then replace $p \leftarrow p - r$        {cyclicity}
$q \leftarrow r + j$, if $q > r$ then replace $q \leftarrow q - r$        {cyclicity}
$y \leftarrow a_p - a_q - c$
**if** $y \geq 0$ **then**
  $x \leftarrow y$ and $c \leftarrow 0$
**else**
  $x \leftarrow y + B$ and $c \leftarrow 1$
**end if**
replace $a_j$ by $x$
$j \leftarrow j - 1$; if $j = 0$ then replace $j \leftarrow r$        {cyclicity}
**return** $x/B$
```

As usual, the register has to be initialised. Best practice is to fill it with large 'arbitrary' numbers, of $\mathcal{O}(B)$, and take $c = 0$. If the starting values are small, such algorithms may take a long time before the output starts to 'look random'.

There is an interesting observation here. If we define

$$z_n = \sum_{k=0}^{r-1} x_{n-r+k} B^k - \sum_{k=0}^{s-1} x_{n-s+k} B^k + c_{n-1} \quad , \tag{51}$$

then the `RCARRY` algorithm is seen to be

$$z_n = (a\, z_{n-1}) \bmod m \quad , \quad m = B^r - B^s + 1 \quad , \quad a = m - (m-1)/B \quad . \tag{52}$$

It is a multiplicative congruential PRNG, acting on integers that have $B$ digits in base $B$; The carry bit is essential to have the multiplications work out correctly[34]. The number $m \approx 2.47 \times 10^{173}$ is prime[35], the period is $(m-1)/48$, about $5 \times 10^{171}$. In [14] it is recommended to generate numbers in a batch of 24, then skipping the next 223, then another batch of 24, and so on. This is the `RANLUX` generator.

[34] Just like the 'carry' operation is necessary in long multiplication.

[35] For the record:
$m$ = 247330401473104534060502521019647190035131349101211839914063056092897225106531867170316401061243044987830824361237755009768067533563832694140062258226274209795000570856079361, and
$a$ = 247330386731063812101356613826074608049705993956988322662342632748341364772062482825984947599810524762601263757689206714403985091753014167166773356178267065685142904661606401.

ℵ A nice feature is that this algorithm can be implemented in floating-point immediately since rounding errors can be avoided.

### 3.3.6 Good: the Mersenne Twister

Abbreviated by MT, this is likely the most popular PRNG nowadays [15]. It is most convenient to represent the integers $x_n$ in binary form, and then all the operations are in $F_2$ (that is, addition is binary `XOR` and multiplication is bitwise `AND`). We describe here the 32-bit case (hence $w = 32$ below). The algorithm is officially a TGFSRPRNG[36], called Mersenne Twister because its period, $2^{19937} - 1$, is a Mersenne prime. The core of the MT algorithm is

$$x_n = x_{n-q} + x_{n-p+1} \begin{pmatrix} 0 & 0 \\ 0 & I_m \end{pmatrix} M + x_{n-p} \begin{pmatrix} I_{w-m} & 0 \\ 0 & 0 \end{pmatrix} M \quad , \tag{53}$$

where $I_k$ is the $k \times k$ unit matrix, and $M$ a carefully chosen $w \times w$ matrix. Note that the binary numbers are multiplied from the left since they are 'row vectors'. The binary number $x_n$ is then multiplied into a *tempering matrix* $T$:

$$z_n = x_n T \quad , \tag{54}$$

[36]Twisted Generalised Feedback Shift Register Pseudo-Random Number Generator.

and $z_n$ is the next pseudorandom integer. The matrix $M$ has a special form:

$$M = \left( \begin{array}{c|c} 0 & \\ 0 & I_{w-1} \\ \vdots & \\ \hline \multicolumn{2}{c}{a} \end{array} \right) \tag{55}$$

where $a$ is another $w$-bit integer. For the tempering step, we can realize that the matrices

$$R = \begin{pmatrix} 0 & \\ 0 & I_{w-1} \\ \vdots & \\ 0 & \cdots 0 \end{pmatrix} \quad , \quad L = \begin{pmatrix} 0 \cdots & 0 \\ & \vdots \\ I_{w-1} & 0 \\ & 0 \end{pmatrix} \tag{56}$$

shift the numbers $x$ to the right and to the left, respectively, by one bit. If we furthermore define the diagonal matrix $D(y) = \mathrm{diag}(y_1, y_2, \ldots, y_w)$ for the $w$-bit number $y = (y_1, y_2, \ldots, y_w)$, the tempering matrix $T$ can be written as follows:

$$T = (R^u \cdot D(d) + 1) \cdot (L^s \cdot D(b) + 1) \cdot (L^t \cdot D(c) + 1) \cdot (R^l + 1) \ \ . \tag{57}$$

Here '1' stands for the $w \times w$ unit matrix. In the table we give the values of the various MT parameters.

| parameter | decimal | binary |
|---|---|---|
| $w$ | 32 | |
| $p$ | 624 | |
| $q$ | 227 | |
| $m$ | 31 | |
| $a$ | 2567483615 | 10011001000010001011000011011111 |
| $u$ | 11 | |
| $d$ | 4294967295 | 11111111111111111111111111111111 |
| $s$ | 7 | |
| $b$ | 2636928640 | 10011101001011000101011010000000 |
| $t$ | 15 | |
| $c$ | 4022730752 | 11101111110001100000000000000000 |
| $l$ | 18 | |

The period of MT is $2^{wp-m} - 1$ which is huge. Moreover, we define the property of $k$-distribution to $v$ bits accuracy as follows: taking $y_n$ to consist of the leading $v$ bits of $x_n$, then the concatenation

$$Y_n = (y_{kn+1}, y_{kn+2}, \dots, y_{kn+k-1})$$

runs over all $2^{kn}$ possibilities as we progress through the series over its whole period[37]. Viewed differently, the $k$-dimensional vectors $\vec{Y_n}$ form a complete

[37]Except the case where all the $y_n$ are zero: if the register consists of zeroes only, we have a selfie.

**Algorithm 3** The two major steps of the `MT` algorithm.

```
{The register is $[x_{n-1}, x_{n-2}, \dots, x_{n-p}]$. Operations: $+$ stands for bitwise XOR, and $\times$ stands for bitwise AND, $>>k$ stands for 'shift right by $k$ bits, $<<$ stands for 'shift left by $k$ bits'.}
———————— Twisted FSR operation ————————
$y \leftarrow$ (upper $w - r$ bits of $x_{n-p+1}$) $+$ (lower $r$ bits of $x_{n-p}$)
if last bit of $y$ is 0 then
    $y \leftarrow y >> 1$
else
    $y \leftarrow (y >> 1) + a$
end if
$x_n \leftarrow x_{n-q} + y$
———————— Tempering operation ————————
$z \leftarrow x_n + ((x_n >> u) \times d)$
$z \leftarrow z + ((z << s) \times b)$
$z \leftarrow z + ((z << t) \times c)$
$z \leftarrow z + (z >> l)$
return $z/2^{32}$ as a floating-point number
```

and perfect hypercubic lattice[38]. The MT is $k$-distributed to 32 bits accuracy for $k$ up to 623, where multiplicative congruential generators typically are no better than 5-distributed.

ℵ However, since in any realistic case the whole period is very far from being used up, the '623-distributed' property is of little use. Indeed the algorithm is known to fail some randomness tests. As one expert states 'there is no one-size-fits-all PRNG'.

## 3.4 Exercises

**Excercise 4 Randomly chosen algorithms**
Perform your own simulation of 'arbitrary algorithms'. Take $M$ to be largish, and determine the lifetime for a fixed starting value (1, say). Note that you can do this by simply generating random integers in $(1, M)$ and stopping when you pick some integer for the second time. Do this a great number of times and produce a histogram of the lifetimes, and compute the average.

**Excercise 5 PRNG playing**
Program a linear congruential PRNG using $m$, $a$ and $c$. Make a 2-d plot of (say) the first 1000 pairs $(x_{2n+1}, x_{2n})$. Try to find plots that look random, it is harder than you might think!

[38]This is either a very good thing or a very bad thing, see also sec.**??**.

**Excercise 6** `RCARRY`
Program your own version of the `RCARRY` algorithm. Make plots of 1000 pairs as in exercise 5. Experiment with various seeds.

# 4 Testing PRNGs

## 4.1 Empirical testing strategies and doubts

Let us consider strings of $n$ bits. Of the $2^n$ possibilities, some strings, $11111111111\cdots$, say, will not 'look random', while $1010101010101010\cdots$ 'looks slightly more random'. A string like $011011100101110111100010001\cdots$ may 'look quite random'[39]. In order to decide whether or not to trust a sequence as 'acceptably random' it is customary to perform empirical tests on the sequence. That is, one computes *some* number using the sequence, and compares that to what would be expected from an equally long sequence of truly random numbers. Then some criterion is applied to decide whether the string has passed this test. For instance, for $n = 10^4$ we can simply count the number of 1's. If this is falls between, say, 4900 and 5100, we accept the string. Two sobering observations must be made. In the first place, a string of truly random bits will fail this test in about 30 per cent of the cases. We might enlarge our window of acceptance to run from 4800 to 5200 and reject only about 5 per cent of truly random strings, but then we will also accept more nonrandom ones. In the second place, one usually performs several or even many tests. A good string, one that looks appreciable random, will fail about 30 per cent of the tests at the one-$\sigma$ level: what do you conclude? What if one of the failed tests is actually the integration that you want to perform? When does one stop testing?

[39]Actually, it is not! I just wrote $0123456789\cdots$ in binary.

#### 4.1.1 The Leeb Conundrum: too much of a good thing

We can look at the procedure of testing $n$-bit strings in a more abstract manner, adapted from [16]. Let us denote by $U$ the set of all $2^n$ possible strings. Taking averages for truly random $n$-bit strings is taking averages over $U$. A test will single out strings that pass it. We say that the test has *strictness* $1-s$ if a fraction $s$ of the $2^n$ strings pass, and all other ones fail. The following sounds trite but is not: *a string $x$ passes test $T$ if it is an elements of the subset $U_T$ of $U$ of all strings that pass test $T$*. In other words, *any* test $T$, no matter how elaborately formulated, is completely described by $U_T$. If test $T$ has strictness $1-s$, then $U_T$ contains $s \times 2^n$ elements. This gives us a handle on the concept of *all tests of strictness* $1-s$: it is the set of all subsets of $U$ of size $s \times 2^n$. Each subset contains precisely $s \times 2^n$ strings, and all strings must occur precisely equally often. The probability that string $x$ passes test $T$ of strictness $1-s$ is therefore given by

$$\binom{2^n}{s\,2^n} \times (s\,2^n) \times \frac{1}{2^n} = \frac{s\,(2^n)!}{(s\,2^n)!((1-s)2^n)!} \ , \tag{58}$$

*independently of $x$*. That is, if we perform *all possible tests* of a given strictness, *every* string will perform equally well (or badly) as every other string! Too much testing is not good. In practice, of course, not all tests of a given strictness are considered equivalent, but this only goes to show that in deciding whether or not you like a sequence as 'random', art and taste are involved as much as objective testing.

### 4.1.2 Any test is a uniformity test

One of the most basic requirements for a PRNG that claims to generate iid uniform random numbers in (0,1), say, is that they indeed come out *uniformly*; and therefore one of the most basic tests is to make a histogram (*i.e.* a *binning*) of a sample of the numbers and decide whether or not it is accpetable. For this we can use a $\chi^2$ test, dicussed below. But that leaves the *iid* property to be tested, which can only be done by using the points as coordinates in a more-dimensional hypercube. That way we can test for uniformity in $2, 3, 4, \ldots$ dimensions (where for instance `RANDU` will fail in 3 dimensions), by binning them. There are, however, many more 'traditional' or 'popular' tests that can be performed. I maintain that *all* these tests are in fact also uniformity tests, simply with differently defined bins. As an example, the *poker test* takes 5-tuples of PRNG numbers and assigns playing-card values to them. A poker hand results: let us say it happens to be full house. The *probability* for obtaining a full house is known, and this must simply be the probability for our 5 numbers to fall into that (weirdly shaped, non-connected) bin in a 5-dimensional hypercube that corresponds to 'full house'. Testing whether the various poker hands occur with the right frequencies is therefore a uniformity test in 5 dimensions, just with complicated bins. One might argue that the dimenionality of the hypercube could be undetermined. For instance in the *runs* test, we look in a sequence $x_1, x_2, x_3, \ldots$ for *runs* of ascending or descending numbers. For instance, the configuration $x_n > x_{n+1} < x_{n+2} < x_{n+3} < \cdots < x_{n+\ell} > x_{n+\ell+1}$ consitutes a run of length $\ell$. The distribution of the run lengths for truly random numbers is known. For very long runs the probability is very small, and therefore one would opt for lumping run lengths of more than 19, say, together in a bin labelled ‘very long runs’. That means that we actually work in a 20-dimensional hypercube.

### 4.1.3 The $\chi^2$ characteristic

This quantity is defined for any collection of objects that can be gathered into a finite number of *bins* numbered $1, \ldots, B$. Let us suppose that we have some hypothesis (that of equidistribution, for instance) that predicts the *expected* occupancy[40] of bin $j$ to be $e_j > 0$. We can compare this to the actual *observed* occupancy of that bin, $n_j$. The $\chi^2$ statistic tests the observation against the hypothesis as follows:

$$\chi^2 = \sum_{j=1}^{B} \frac{(n_j - e_j)^2}{e_j} \quad . \tag{59}$$

The Bernoulli distribution assigns the probability of occupancy $(n_1, n_2, \ldots, n_B)$ if the objects are independently distributed with probability $p_j$ to end up in bin $j$:

$$P(n_1, n_2, \ldots, n_B) = \frac{N!}{n_1!\, n_2! \cdots n_B!}\, {p_1}^{n_1} {p_2}^{n_2} \cdots {p_B}^{n_B} \quad . \tag{60}$$

[40] *i.e.* how many objects fall into the bin.

Here $N = \sum_j n_j$ is the total number of objects. From

$$\frac{N!}{n_1!\, n_2! \cdots n_j! \cdots n_B!}\, n_j{}^{\underline{k}} = N^{\underline{k}}\, \frac{(N-k)!}{n_1!\, n_2! \cdots (n_j-k)! \cdots n_B!} \tag{61}$$

we find the expectations

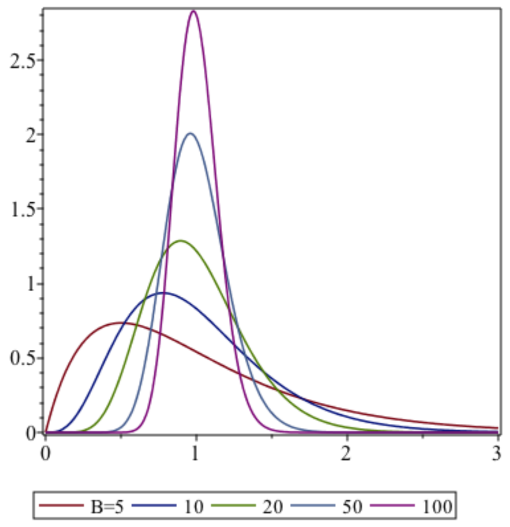
$$\langle n_j{}^{\underline{m}} \rangle = N^{\underline{m}}\, p_j{}^m \quad , \quad \langle n_j{}^{\underline{m_j}} n_k{}^{\underline{m_k}} \rangle = N^{\underline{m_j + m_k}}\, p_j{}^{m_j} p_k{}^{m_k} \quad . \tag{62}$$

Some algebra then leads to

$$\begin{aligned} \left\langle \chi^2 \right\rangle &= B - 1 \ , \\ \sigma(\chi^2)^2 &= 2(B-1)\left(1 - \frac{1}{N}\right) + \frac{1}{N}\left(\sum_{j=1}^{B} \frac{1}{p_j} - B^2\right) \quad . \end{aligned} \tag{63}$$

In the variance the second term is nonnegative, vanishing if every $p_j = 1/B$. For large $N$ the density of $\chi^2$ is a $\Gamma$-density (*cf* Eq.(148)):

$$P_{\chi^2}(t) = \frac{1}{2\,\Gamma\left((B-1)/2\right)}\,(t/2)^{(B-3)/2}\, \exp(-t/2) \tag{64}$$



for $\chi^2$ to attain the value $t$. The confidence levels (although less simple than those for the normal distribution) are therefore known. The plot shows the renormalised distributions $(B-1)P_{\chi^2}((B-1)t)$ for various values of $B$. As the number of bins increases the density approaches a Gaussian centered around 1 (*i.e.* the value of $\chi^2$ becomes centered around $B-1$. The nice thing about $\chi^2$ is that for large values of $N$ it only depends on the *number* of bins $B$. This holds if, say, the second term in Eq.(63) can be neglected. A widely used rule of thumb says that we should make sure that every $e_j$ is at least 5-10 or so. At any rate the *expected* $\chi^2$ is strictly equal to $B-1$, and therefore a quick look at the '$\chi^2$ per degree of freedom' is usually already a good indicator of a dubious result: it ought to differ not too much from one.

## 4.2 Theoretical testing strategies

A number of tests can be performed on the sequence as a whole. An example is the property of $k$-distribution mentionied in sect. 3.3.6. Several theoretical tests have been constructed for linear congruential PRNGs

### 4.2.1 The number-to-number correlation

### 4.2.2 The spectral test

ℵ Theoretical tests deserve reservation since they (usually) apply to the *whole* period rather than to the much smaller sequences one is likely to actually use. They must not be used to 'certify' the goodness of a PRNG, but they *can* help identify a bad one.

## 4.3 Exercises

**Excercise 7 Equidistribution**
Divide the unit interval (0,1) into 100 equal-sized bins. Write code to evaluate the $\chi^2$ of a point set against the hypothesis that it is equidistributed. Do the test for a 'good' PRNG (`RCARRY` or otherwise, or maby a linear congruential PRNG that you like), and perform the $\chi^2$ a number of times. Do this also for a PRNG that you known is 'bad'. If you feel up to it, repeat the exercise using 2 or even 3 dimensions.

**Excercise 8 Trouble with empty bins**
Suppose that we have a $\chi^2$ test with $B$ bins, all expected to be equally filled. Now furthermore suppose that *actually* $M$ of the bins cannot be filled for some reason (an example is given below, in exercise 9). Show that in that

case the $\chi^2$ for $N$ points will behave as $NM/(B - M)$ under the assumption that the non-empty bins are equally filled. Therefore $\chi^2$ will in this case run away as $N$ increases.

**Excercise 9 Permutation tests**

1. Take a PRNG of your choice, and perform the *permutation test*: divide your generated point set in triples $(x_{3n-2}, x_{3n-1}, x_{3n})$ and determine which of the 6 different possible orderings in magnitude they belong. Then perform the $\chi^2$ test (with $B = 6$ here).

2. (From [8]) Consider the *Fibonacci algorithm*, a shift-register generator using
$$x_k = (x_{k-1} + x_{k-2}) \bmod 1$$
Prove that the permutations $x_{k-2} < x_k < x_{k-1}$ and $x_{k-2} > x_k > x_{k-1}$ can never occur, and verify this. Also see how the $\chi^2$ of the previous item will deteriorate as the number of points $N$ increases.

3. Show that a similar problem holds for $x_k = (x_{k-24} + x_{k-55}) \bmod 1$. Argue that *in this case* the problem is much less severe for any reasonable $N$.

# 5 Quasi-Monte Carlo

## 5.1 Generalities of QMC

### 5.1.1 The New Leap of Faith

In Monte Carlo integration, the relatively slow $1/\sqrt{N}$ behaviour of the error estimate is due to the fact that the point sets $\mathbf{X}$ are chosen from the ensemble of random iid uniform point sets. We may consider doing better by choosing our point set to be 'more uniform' (in some sense to be defined) than a random point set is expected to be. Such point sets are called *superunuiform*, or *quasirandom*. Doing Monte Carlo with them is therefore called *Quasi-Monte Carlo* (QMC). The difference with regular Monte Carlo is the fact that the point sets $\mathbf{X}$ used are *not* typical members of the ensemble of random iid uniform point sets[41]. But that means that the *basis* on which the various error estimates $E_{1,2,4}$ of sect. 2.2.2 are constructed is now invalid! We shall have to define a new ensemble, and come to a New Leap of Faith that says that the particular superuniform point set that we use is typical for that ensemble.

It would seem reasonable to insist that, since we take trouble to use point sets with a low measure of nonuniformity, the new superuniform ensemble should be characterised by that measure. For the rest there does not seem to be any reason not to stay as close to the random iid ensemble as possible. Let us assume that there is a function $T(\mathbf{X}) > 0$ that is a measure of the

[41]Although they are in there, of course; but they form a tiny minority.

nonuniformity of the point set $\mathbf{X}$[42]. We also assume that $T(\mathbf{X})$ is invariant under any permutation of the points inside $\mathbf{X}$. We shall then use the following superuniform ensemble density (restricting ourselves to integration over the unit hypercube $I^d$):

$$\begin{aligned} P_S(t; \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) &= \frac{1}{P(t)}\, \delta(T(\mathbf{X}) - t) \ , \\ P(t) \ &= \ \int d\mathbf{x}_1\, d\mathbf{x}_2 \cdots d\mathbf{x}_N \ \delta(T(\mathbf{X}) - t) \ . \end{aligned} \tag{65}$$

$t$ is the value of $T$ for all point sets in the superuniform ensemble. The normalisation $P(t)$ is the probability density to find $T(\mathbf{X}) = t$ for a point set $\mathbf{X}$ in the uniform iid ensemble. The density $P(t)$ is computed, for various definitions of nonuniformity $T$, in section 6. The absence of the iid property is obvious[43].

ℵ The New Leap of Faith appears to be ignored by almost everyone in the QMC field, with serious consequences (see below).

[42]Hence the smaller $T(\mathbf{X})$, the more uniform the point set $\mathbf{X}$ is.

[43]This ensemble may be called *microcanonical*: the corresponding canonical one would let $T(\mathbf{X})$ fluctuate around the value of $t$, but not much seems to be gained by going over to this ensemble.

### 5.1.2 The mechanism of error improvement

In the superuniform ensemble, the points are not iid as we have seen. On the other hand, by the definition of $T$ and since the superuniform ensemble is a subset of the uniform one, the density $P_S(t; \mathbf{x}_1, \ldots, \mathbf{x}_N)$ is symmetric in the point coordinates. Let us consider the situation where we disregard all points except two of them:

$$\frac{1}{P(t)} \int d\mathbf{x}_3\, d\mathbf{x}_4 \cdots d\mathbf{x}_N P_S(t; \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_N) \equiv 1 - \frac{1}{N} F_2(t; \mathbf{x}_1, \mathbf{x}_2) \ . \tag{66}$$

This way of introducing the two-point function $F_2$ is always possible, although we anticipate a factor $1/N$. Moreover we assume that each point in $\mathbf{X}$ is distributed uniformly when considered individually:

$$\int d\mathbf{y}\ F_2(t; \mathbf{x}, \mathbf{y}) = 0 \ . \tag{67}$$

If we take the Monte Carlo estimator $E_1$ to do QMC as well:

$$E_1 = \frac{1}{N} \sum_j w(\mathbf{x}_j) \ , \tag{68}$$

we see that the superuniform ensemble average

$$\left\langle E_1 \right\rangle_S = J_1 \tag{69}$$

again gives us an unbiased integral estimate. The real issue is in the variance:

$$\begin{aligned}\left\langle {E_1}^2\right\rangle_S &= \frac{1}{N^2}\left\langle\sum_{j,k} w(\mathbf{x}_j)\,w(\mathbf{x}_k)\right\rangle_S \\ &= \frac{1}{N^2}\left(N\int d\mathbf{x}\;w(\mathbf{x})^2 + N^{\underline{2}}\left(\int d\mathbf{x}\;w(\mathbf{x})\right)^2\right. \\ &\quad \left. -\frac{N^{\underline{2}}}{N}\int d\mathbf{x}\,d\mathbf{y}\;F_2(t;\mathbf{x},\mathbf{y})\,w(\mathbf{x})\,w(\mathbf{y})\right) \quad , \end{aligned} \tag{70}$$

so that the variance now reads

$$\sigma(E_1)^2_S = \frac{1}{N}\left(J_2 - {J_1}^2 - \left(1-\frac{1}{N}\right)\int d\mathbf{x}\,d\mathbf{y}\;F_2(t;\mathbf{x},\mathbf{y})\,w(\mathbf{x})\,w(\mathbf{y})\right) \quad . \tag{71}$$

Two important conclusions follow:

1. A small, $\mathcal{O}\left(1/N\right)$ deviation from iid uniformity can have a large effect on the expected integration error.

2. We may expect that the integrand values $w(\mathbf{x})$ and $w(\mathbf{y})$ are correlated when $\mathbf{x}$ and $\mathbf{y}$ are close to one another[44]; in particular they will tend to have the same sign. For QMC to be an improvement, we therefore want $F_2(t;\mathbf{x},\mathbf{y})$ to become positive when $\mathbf{x}$ and $\mathbf{y}$ approach one another: in a sense, the points in $\mathbf{X}$ must feel a mutual *repulsion*.

[44]In some reasonable sense.

Another way to write Eq.(71) is

$$\sigma(E_1)^2_S = \frac{1}{2N}\int d\mathbf{x}\,d\mathbf{y}\left(1+\frac{N-1}{N}F_2(t;\mathbf{x},\mathbf{y})\right)\Big(w(\mathbf{x})-w(\mathbf{y})\Big)^2 \quad . \tag{72}$$

The ideally best possible error estimate that is valid for any integrand $w$ is therefore reached (for $N \to \infty$) when

$$F_2(t;\mathbf{x},\mathbf{y}) = -1 + \delta^d(\mathbf{x}-\mathbf{y}) \;\; . \tag{73}$$

As we shall see, this is precisely what we obtain for $T(\mathbf{X}) \to 0$, a beautiful but admittedly unreachable situation.

We of course have to somehow compute $P(t)$ and $F_2$ for a given nonunuiformity $T$, and show that $F_2$ integrates to zero. This is dealt with in section 6.

$\aleph$ Like the New Leap of Faith, the mechanism behind the error improvement of QMC over MC seems to be virtually unknown.

## 5.2 Error estimators

### 5.2.1 The first-order estimate

### 5.2.2 The second-order estimate

### 5.2.3 Payback time: Lack of Leap of Faith is Punished

# 6 Nonuniformity of point sets

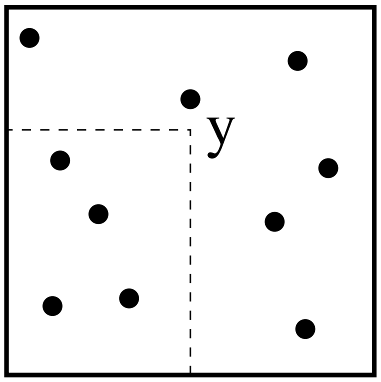
## 6.1 Measures of nonuniformity: Discrepancy

### 6.1.1 The star discrepancy

Consider *any* point set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ in the $d$-dimensional unit hypercube $I^d$. With $\theta(\mathbf{x} < \mathbf{y})$ we shall mean the function that is 1 if $\mathbf{x}^\mu < \mathbf{y}^\mu$ for all $\mu = 1, 2, \ldots, d$, otherwise zero[45]. We define the *local discrepancy* $g(\mathbf{y})$ as

$$\begin{aligned} g(\mathbf{y}) &= \frac{1}{N} \sum_j h(\mathbf{x}_j; \mathbf{y}) \ , \\ h(\mathbf{x}_j; \mathbf{y}) &= \theta(\mathbf{x}_j < \mathbf{y}) - \mathrm{vol}(\mathbf{y}) \quad , \quad \mathrm{vol}(\mathbf{y}) = \prod_{\mu=1}^{d} \mathbf{y}^\mu \ . \end{aligned} \tag{74}$$

[45]That is, $\mathbf{x}$ finds itself inside the rectangle spanned by $\mathbf{y}$ and the origin.



The local discrepancy compares the fraction of points 'below' $\mathbf{y}$ with what that fraction would be if the points were ideally uniformly distributed. In the plot we give an example for $d = 2$. The point $\mathbf{y}$ has coordinates $(1/2, 2/3)$ and its rectangle contains 4 points out of 10, hence $g(\mathbf{y}) = 1/15$. At every corner, the local discrepancy vanishes. Measures of *global* discrepancy can be defined: the most important are the *extreme* discrepancy[46] (the *Kolmogorov-Smirnov* statistic):

$$L^*_\infty = \sup_{\mathbf{y} \in I^d} |g(\mathbf{y})| \quad , \tag{75}$$

and the *quadratic* discrepancy (the *Kramér-von Mises* statistic):

$$L^*_2 = \int\limits_{I^d} d\mathbf{y}\; g(\mathbf{y})^2 \quad . \tag{76}$$

The extreme discrepancy is beloved of mathematicians, but the quadratic one is easier to handle in computations. At any rate, if one is small the other

[46]The asterisk refers to the fact that the hyper-rectangles are attached to the origin.

will be small as well, since $L_2^* < (L_\infty^*)^2$ and the function $g(\mathbf{y})$ is piecewise linear in the components $\mathbf{y}^\mu$ [47]. The $L_2^*$ discrepancy can be formulated as a function of $\mathbf{X}$ only:

$$\begin{aligned} L_2^* &= \frac{1}{N^2}\sum_{i,j=1}^{N}\prod_{\mu=1}^{d}\left(1-\max(\mathbf{x}_i^\mu,\mathbf{x}_j^\mu)\right) - \frac{2^{1-d}}{N}\sum_{i=1}^{N}\prod_{\mu=1}^{d}\left(1-(\mathbf{x}_i^\mu)^2\right) + 3^{-d} \\ &= \frac{1}{N^2}\sum_{i,j=1}^{N}\mathfrak{b}(\mathbf{x}_i,\mathbf{x}_j)\ \ , \end{aligned} \tag{77}$$

with the 'two-point function' defined as

$$\mathfrak{b}(\mathbf{x}_i,\mathbf{x}_j) = \prod_{\mu=1}^{d}\left(1-\max(\mathbf{x}_i^\mu,\mathbf{x}_j^\mu)\right) - 2\prod_{\mu=1}^{d}\frac{(1-(\mathbf{x}_i^\mu)^2)}{2} + 3^{-d}\ \ . \tag{78}$$

For point sets $\mathbf{X}$ taken from the random iid ensemble, we have the expectation

$$\langle L_2^*\rangle = \frac{1}{N}\left(2^{-d}-3^{-d}\right)\ \ . \tag{79}$$

It is instructive to compare this for the 'hypercubic' lattice with $N = M^d$ points. This has

$$x_j^\mu = \frac{2k_\mu - 1}{2M}\quad,\quad j = 1+\sum_{\mu=1}^{D}(k_\mu-1)M^{\mu-1}\quad,\quad k_\mu\ \in\ \{1,2,\ldots,M\}\ \ , \tag{80}$$

[47]With jumps of magnitude $1/N$ whenever $\mathbf{y}^\mu = \mathbf{x}_j^\mu$ for some $\mu$ and $j$.

and the quadratic discrepancy evaluates to

$$L_2^* = \left(\frac{1}{3}\right)^d \left[\left(1+\frac{1}{2M^2}\right)^d - 2\left(1+\frac{1}{8M^2}\right)^d + 1\right] \;\approx\; \frac{d}{4N^{2/d}\,3^d} \;\;, \qquad (81)$$

where the approximation holds for large $M$. For $d > 4$ the 'random' point sets are, by this definition, *more uniform* than the regular hypercubic ones.

ℵ For given $N$, the hypercubic lattice will win out as $d$ increases. However, the usual situation is that $d$ is fixed (by the integration problem itself), and $N$ is the free parameter that tells us how much computing resources can be spent.

### 6.1.2 Random *vs* Regular: Translation *vs* Rotation

You may wonder how a random point set can be more uniform than a nice, beautiful hypercubic lattice. The answer[48] must be the following. In low dimensions ($d = 1, 2$ or so) the translational invariance (by steps of $1/M$) of the hypercubic lattice is an obvious advantage. Indeed, in one dimension the regularly-spaced point set is the most uniform one by any standard. However, in more dimensions we also have to consider the *rotational* properties of the point set. A random collection of points does not look very different when rotated, whereas for the hypercubic lattice some directions contain

[48]At least, the answer that satisfies me.

many points (especially if we look parallel to the axes), while some directions contain hardly any points at all. As the dimensionality increases, there are more and more directions to choose from, and the rotational invariance becomes the more dominant property.

### 6.1.3 The Roth bound

It is obvious that the ideal finite point set, with $L_2^* = L_\infty^* = 0$, does not exist. In fact the discrepancies have lower bounds. In [17][49] the so-called *Roth bound* is proven: for any point set $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ in $I^d$ $(d \geq 2)$ there is a constant $c_d > 0$ (depending only on $d$) such that

$$L_2^*(\mathbf{X}) > c_d \, \frac{\log(N)^{d-1}}{N^2} \quad . \tag{82}$$

We see that the discrepancy can be quite a bit smaller than the 'expected' $(2^{-d} - 3^{-d})/N$ for random point sets; the question, of course, is how to find such low-discrepancy point sets, and what $c_d$ is. In [17] it is proven that Eq.(82) holds with

$$c_d = 2^{-8d} \left((d-1)\log(2)\right)^{1-d} \quad , \tag{83}$$

but I believe the 'real' $c_d$ is (considerably) larger.

[49]This book is an absolute must for anyone interested in low-discrepancy point sets and related matters. I find it interesting to see that the emphasis is on $L_\infty^*$ rather than on $L_2^*$, perhaps reflecting the background of the authors as mathematicians rather than physicists. Section 2.2, Lemma 2.5 is especially relevant.

### 6.1.4 The Koksma-Hlawka inequality

### 6.1.5 The Wiener measure and the Woźniakowski Lemma

Let us consider functions $f(\mathbf{x})$ on the d-dimensional unit hypercube $I^d$. For $d = 1$, the *Wiener measure* $W$ describes an ensemble of functions that is defined by its expectation values:

$$\langle f(x)\rangle_W = 0 \quad , \quad \langle f'(x)\, f'(y)\rangle_W = \delta(x-y) \ . \tag{84}$$

By integration we then find

$$\langle f(x)\, f(y)\rangle_W = \min(x,y) \ . \tag{85}$$

Let us now consider integrating such a function using Monte Carlo. The integration error,

$$\eta = \frac{1}{N}\sum_j f(x_j) - \int\limits_0^1 dx\, f(x) \ , \tag{86}$$

(where the sum runs from $j = 1$ to $j = N$) has a squared average over $W$, called the *complexity*:

$$\left\langle \eta^2 \right\rangle_W = \frac{1}{N^2}\sum_{j,k}\min(x_j,x_k) - \frac{2}{N}\sum_j (x_j - x_j^2/2) + \frac{1}{3} \ . \tag{87}$$

The more-dimensional variant is called the *Wiener sheet measure*:

$$\langle \varphi(\mathbf{x})\rangle_W = 0 \quad , \quad \left\langle \frac{\partial^n}{\partial x^1\cdots\partial x^d} f(\mathbf{x})\, \frac{\partial^n}{\partial y^1\cdots\partial y^d} f(\mathbf{y}) \right\rangle_W = \delta^d(\mathbf{x}-\mathbf{y}) \ , \tag{88}$$

and

$$\langle f(\mathbf{x})\, f(\mathbf{y})\rangle_W = \prod_{\mu=1}^{d} \min(\mathbf{x}^\mu, \mathbf{x}^\mu) \quad . \tag{89}$$

The corresponding complexity is

$$\langle \eta^2 \rangle_W = \frac{1}{N^2} \sum_{j,k} \prod_{\mu} \min(\mathbf{x}_j^\mu, \mathbf{x}_k^\mu) - \frac{2}{N} \sum_{j} \left( \mathbf{x}_j^\mu - \frac{1}{2}(\mathbf{x}_j^\mu)^2 \right) + \left(\frac{1}{3}\right)^d \quad . \tag{90}$$

By replacing every $\mathbf{x}^\mu$ by $1 - \mathbf{x}^\mu$ this is seen to be nothing but the quadratic discrepancy $L_2^*$, with every (hyper)rectangle anchored not to the origin point $(0, 0, \ldots, 0)$ but the point $(1, 1, \ldots, 1)$. This is the Woźniakowski lemma: for functions drawn from the Wiener (sheet) measure the complexity is given by the discrepancy of the point set [18]. The central notion here is that of a *problem class*: the ensemble of functions that we assume our particular integrand is drawn from.

ℵ The type of function typical of the Wiener (sheet) measure is hardly what you would encounter in high-energy phenomenology: functions with a fractal structure, continuous but nowhere differentiable. Our usual integrand, in contrast, has discontinuities but is differentiable elsewhere. Nevertheless the above derivation shows that a conjectured ensemble of integrands dictates a measure of nonuniformity. In what follows we try to make this notion more workable.

## 6.2 Measures of nonuniformity: Diaphony

### 6.2.1 Fourier problem classes

We consider the set of all functions $f$ periodic on the $d$-dimensional unit hypercube that can be written using Fourier modes,

$$f(\mathbf{x}) = \sum_{\mathbf{n}}^{*} a_{\mathbf{n}} \; \exp(2i\pi \, \mathbf{n} \cdot \mathbf{x}) \;\; , \tag{91}$$

where the asterisk means that the sum is to be taken over all $d$-dimensional vectors $\mathbf{n}$ with integer components, except the zero vector $\mathbf{n} = (0, 0, \ldots, 0)$. Since constant functions are integrated with zero error, the zero modes are irrelevant to our discussion[50]. We take the $a_{\mathbf{n}}$ to be real numbers. The Fourier ensemble measure is then defined by the measure on the set of coefficients $a_{\mathbf{n}}$, which we take to be Gaussian as follows:

$$\mathcal{D}f = \prod_{\mathbf{n}}^{*} da_{\mathbf{n}} \; \exp\left(-\frac{a_{\mathbf{n}}^2}{2\sigma_{\mathbf{n}}^2}\right) \frac{1}{\sqrt{2\pi\sigma_{\mathbf{n}}^2}} \tag{92}$$

This implies the following ensemble averages:

$$\langle a_{\mathbf{n}} \rangle_F = 0 \quad , \quad \langle a_{\mathbf{n}} \, a_{\mathbf{n}'} \rangle_F = \sigma_{\mathbf{n}}^2 \; \theta(\mathbf{n} = \mathbf{n}') \;\; . \tag{93}$$

[50]Indeed, you may argue that any integration is nothing but an attempt to strip an integrand of all its *non*zero modes.

The real quantity $\sigma_{\mathbf{n}}^2$ is called the *strength* of the mode $\mathbf{n}$. We shall assume that $\sigma_{\mathbf{n}}^2$ depends on the components $\mathbf{n}^\mu$ ($\mu = 1, 2, \ldots, d$) only through their absolute values. Moreover we shall insist that the total strength is finite, in fact by convention

$$\sum_{\mathbf{n}}^{*} \sigma_{\mathbf{n}}^2 = 1 \quad . \tag{94}$$

ℵ The functions $f$ are real *on average*. The Fourier problem class contains the real-valued functions but is actually even larger.

### 6.2.2 Fourier diaphony

We envisage integrating $f$ using a point set $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$, so that the integration error is the estimated integral:

$$\eta = \frac{1}{N} \sum_{j=1}^{N} f(\mathbf{x}_j) \tag{95}$$

The squared error, averaged over the Fourier problem class, is then

$$\begin{aligned} \left\langle |\eta|^2 \right\rangle_F &= \frac{1}{N^2} \sum_{j,k=1}^{N} \sum_{\mathbf{n},\mathbf{n}'}^{*} \langle a_{\mathbf{n}}\, a_{\mathbf{n}'} \rangle \exp\left( 2i\pi(\mathbf{n}\cdot\mathbf{x}_j - \mathbf{n}'\cdot\mathbf{x}_k) \right) \\ &= \frac{1}{N^2} \sum_{\mathbf{n}}^{*} \sigma_{\mathbf{n}}^2 \left| \sum_{j=1}^{N} \exp(2i\pi\, \mathbf{n}\cdot\mathbf{x}_j) \right|^2 \quad . \end{aligned} \tag{96}$$

The better the point set is at integrating the various Fourier modes, the smaller the error will be. This, then, leads us to define the *diaphony* of the point set as

$$\begin{aligned} T(\mathbf{X}) &= \frac{1}{N}\sum_{\mathbf{n}}^{*}\sigma_{\mathbf{n}}^2\left|\sum_{j=1}^{N}\exp(2i\pi\,\mathbf{n}\cdot\mathbf{x}_j)\right|^2 = \frac{1}{N}\sum_{j,k=1}^{N}\beta(\mathbf{x}_j-\mathbf{x}_k)\ \ , \\ \beta(\mathbf{z}) &= \sum_{\mathbf{n}}^{*}\sigma_{\mathbf{n}}^2\,\exp(2i\pi\,\mathbf{n}\cdot\mathbf{z})\ \ . \end{aligned} \tag{97}$$

The *bare two-point function* $\beta(\mathbf{z})$ is periodic in each of the components $\mathbf{z}^\mu$, and is invariant under $\mathbf{z}^\mu \to -\mathbf{z}^\mu$ for $\mu = 1, 2, \ldots, \mu$. In addition we have the following important properties:

$$\beta(0) = 1\ \ ,\quad \int \beta(\mathbf{z})\,d\mathbf{z} = 0\ \ . \tag{98}$$

The factor in front of the definition of $T(\mathbf{X})$ reads $1/N$ (not $1/N^2$) by convention. The choice of the strengths $\sigma_{\mathbf{n}}^2$ specifies the particular diaphony. By construction, $T(\mathbf{X})$ is nonnegative, and for the 'most nonuniform' point set, where all $\mathbf{x}_j$ coincide, $T(\mathbf{X}) = N$.

$\aleph$ To have a diaphony that we can realistically compute for a point set, it is important to have $\beta(\mathbf{z})$ in some kind of closed form.

### 6.2.3 Choose your strength: examples of diaphony

A few examples of diaphony may be given. These are mainly geared towards allowing results in closed form, plus the 'physical' intuition that modes with small $|\mathbf{n}|$ are 'naturally' more prominent than those with large $|\mathbf{n}|$.

1. Euler diaphony[51] $T_E$ for $d = 1$:

$$\begin{aligned} \sigma_n^2 &= \frac{3}{\pi^2}\frac{1}{n^2} \;, \\ \beta_E(x) &= 1 - 6|x|(1-|x|) \end{aligned} \tag{99}$$

   For the computation of $\beta_E(x)$ see sec.11.4.1.

2. Euler diaphony $T_E$ for $d > 1$:

$$\begin{aligned} \sigma_{\mathbf{n}}^2 &= \frac{1}{2^d-1}\prod_{\mu=1}^{d}\left(\theta(\mathbf{n}^\mu = 0) + \frac{3}{\pi^2(\mathbf{n}^\mu)^2}\theta(\mathbf{n}^\mu \neq 0)\right) \;, \\ \beta(\mathbf{x}) &= \frac{1}{2^d-1}\left(-1 + \prod_{\mu=1}^{d}(1+\beta_E(\mathbf{x}^\mu))\right) \end{aligned} \tag{100}$$

3. Gulliver diaphony[52] $T_G$:

$$\sigma_{\mathbf{n}}^2 = \left(\left(\frac{1+s}{1-s}\right)^d - 1\right)^{-1} s^{-|n^1|-|n^2|-\cdots-|n^d|} \quad , \quad 0 < s < 1 \;,$$

[51] Named after Euler's formula $\sum_{n\geq 1} n^{-2} = \pi^2/6$.

[52] After Gulliver de Boer, the student who suggested it.

$$\beta_G(\mathbf{x}) \;=\; \left(\left(\frac{1+s}{1-s}\right)^d - 1\right)^{-1} \left(-1 + \prod_{\mu=1}^{d} \frac{1-s^2}{1-2s\cos(2\pi\mathbf{x}^\mu)+s^2}\right) \tag{101}$$

4. Block diaphony[53] $T_B$:

$$\begin{aligned}
\sigma_{\mathbf{n}}^2 &= \frac{1}{2p}\prod_{\mu=1}^{d}\theta(-c \le \mathbf{n}^\mu \le c) \quad , \quad 2p = (2c+1)^d - 1, \\
\beta_B(\mathbf{x}) &= \frac{1}{2p}\left(-1 + \prod_{\mu=1}^{d}\frac{\sin((2c+1)\pi\,\mathbf{x}^\mu)}{\sin(\pi\,\mathbf{x}^\mu)}\right)
\end{aligned} \tag{102}$$

5. Jacobi diaphony[54] $T_J$:

$$\begin{aligned}
\sigma_{\mathbf{n}}^2 &= \frac{1}{K(\lambda;0)^d - 1}\,\exp(-\lambda\,|\mathbf{n}|^2) \ , \\
\beta_J(\mathbf{x}) &= \frac{1}{K(\lambda;0)^d - 1}\left(-1 + \prod_{\mu=1}^{d} K(\lambda;\mathbf{x}^\mu)\right) \\
K(\lambda;z) &= \sum_n \exp(-\lambda n^2 + 2i\pi n z)
\end{aligned}$$

[53] Since the contributing modes are treated *en bloc.*

[54] Beause of the emergence of Jacobi theta functions with real-valued nome.

$$= \sqrt{\frac{\pi}{\lambda}} \sum_n \exp\left(-\frac{\pi^2(n+z)^2}{\lambda}\right) \quad . \tag{103}$$

In the last line the Poisson summation formula has been used (*cf* sec.11.2.1).

A major drawback of diaphonies is that lattice vectors **n** that have the same norm but are oriented differently can have very different strengths, especially the more-dimensional Euler diaphony. The Gulliver diaphony aims at repairing this somewhat while still having a closed form for the two-point function. The Jacobi diaphony comes closest to full rotational invariance, while the sums can usually be restricted to a manageable number of terms.

$\aleph$ For the Euler diaphony in two dimensions, the lattice vectors $\mathbf{n} = (13, 0)$ and $\mathbf{n} = (12, 5)$ make an angle of only 21 degrees, but their strengths differ by a factor 21 as well.

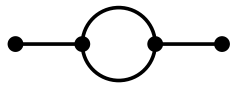
## 6.3 QFT for diaphony

### 6.3.1 The distribution of diaphony

A given point set may have a diaphony of, say, 0.8: is this good, or bad, or what? The natural yardstick is of course what you would expect for a *random* point set. It is therefore interesting to see what can be said about the distribution of $T(\mathbf{X})$ if the point set $\mathbf{X}$ is taken from the ensemble of random point sets discussed in section 2.1.3. In particular, we will be interested in $\langle T^\ell \rangle$ ($\ell = 1, 2, \ldots$) where the average is over the ensemble of random point sets, that is, we integrate over the points in the point set, assuming them to be iid uniform in the hypercube. Eventually, we want to be able to say something about the generating function $\Omega(z) \equiv \langle \exp(zT) \rangle$.

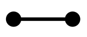
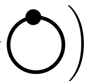
### 6.3.2 Feynman rules for diaphony in the large-$N$ limit

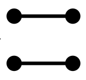
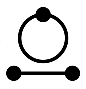
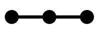
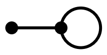
It is useful to formulate the computation of the various moments of $T$ in terms of Feynman diagrams. The bare two-point function $\beta(\mathbf{x}_j - \mathbf{x}_k)$ is the bare propagator, and the points themselves are the vertices of the diagrams. The lattice vectors $\mathbf{n}$ are the momenta, and momentum is conserved at each vertex. An example is

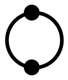
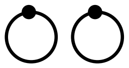
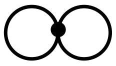


$$= \int d\mathbf{x}_1\, d\mathbf{x}_2\, d\mathbf{x}_3\, d\mathbf{x}_4\, \beta(\mathbf{x}_1 - \mathbf{x}_2)\beta(\mathbf{x}_2 - \mathbf{x}_3)^2\beta(\mathbf{x}_3 - \mathbf{x}_4)$$

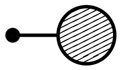
Diagrams can be disconnected. If a diagram contains $k$ vertices then it picks up a factor $N^{\underline{k}}$ since we then have to sum over $k$ *distinct* points. In addition there is a combinatorial factor, the number of ways that diagram can be formed. The first moments of the $T$ distribution are now

$$\langle T \rangle = \frac{1}{N}\Big( N^{\underline{2}}$$  $$+ N^{\underline{1}}$$  $$\Big) \quad ,$$

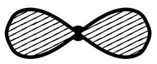
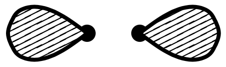
$$\langle T^2 \rangle = \frac{1}{N^2}\Big( N^{\underline{4}}$$  $$+ 2N^{\underline{3}}$$  $$+ 4N^{\underline{3}}$$  $$+ 4N^{\underline{2}}$$ 

$+2N^{\underline{2}}$  $+ N^{\underline{2}}$  $+ N^{\underline{1}}$  $\Big)$ (104)

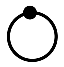
A tremendous simplification arises from the properties of $\beta(\mathbf{z})$. In the first place, all tadpoles vanish:

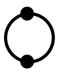
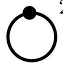
 $= \;\; 0 \;\; .$ (105)

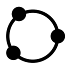
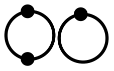
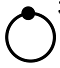
Secondly, we have *one-vertex reducibility*: any two pieces of a connected diagram that are connected by a single vertex factor can be separated[55]:

 $=$  . (106)

This is due to the fact that no momentum can flow between the pieces through the connecting vertex. All this means that only one-particle irreducible vacuum diagrams are nonzero. In addition, we are not really interested in values of $N$ smaller than $10^3$ or so. We can therefore take the large-$N$ limit, so that in $\langle T^p\rangle$ only those diagrams survive that carry a prefactor $N^{\underline{p}} \approx N^p$:

$\langle T\rangle \;=$  ,

$\left\langle T^2\right\rangle \;=\; 2$  $+$  $^2$ ,

$\left\langle T^3\right\rangle \;=\; 8$  $+\,6$  $+$  $^3$ , (107)

[55]Without changing the $N^{\underline{k}}$ in front, of course.

and so on[56]. These diagrams are sums of products of *bracelets*. A $k$-bead bracelet $B_k$ evaluates to

$$B_k \equiv \int d\mathbf{x}_1\, d\mathbf{x}_2 \cdots d\mathbf{x}_k\; \beta(\mathbf{x}_1 - \mathbf{x}_2)\beta(\mathbf{x}_2 - \mathbf{x}_3)\cdots\beta(\mathbf{x}_k - \mathbf{x}_1) = \sideset{}{^*}\sum_{\mathbf{n}} \sigma_{\mathbf{n}}^{2k} \quad . \tag{108}$$

Owing to our choice of diaphony[57] the diaphony distribution becomes $N$-independent for large $N$. In other terms in the $1/N$ expansion we may encounter non-bracelet diagrams such as the last diagram in Eq.(104). In general, a $L$-loop connected diagram carries a factor $1/N^{L-1}$, and we see that $1/N$ plays the rôle of Planck's constant.

ℵ The star discrepancy $NL_2^*$ of course has its own propagator, defined as

$$\beta(\mathbf{x}_1, \mathbf{x}_2) = \int d\mathbf{y}\; h(\mathbf{x}_1;\mathbf{y})h(\mathbf{x}_2;\mathbf{y})$$

But since this propagator is not translationally invariant nor integrates to zero, its analysis in terms of Feynman diagrams is much more complicated. Still, some results have been derived [19].

[56]The sum of the coefficients in $\langle T^m\rangle$ is $(2m)!/2^m m!$.

[57]In particular the use of the factor $1/N$ rather than $1/N^2$.

### 6.3.3 Collecting bracelets

We can immediately find the expectation value[58] and variance of $T$:

$$\langle T\rangle = \sum_{\mathbf{n}}^{*} \sigma_{\mathbf{n}}^2 = 1 \quad , \quad \sigma(T)^2 = \left\langle T^2\right\rangle - 1 = 2\sum_{\mathbf{n}} \sigma_{\mathbf{n}}^4 \ . \tag{109}$$

But we can do more. Let us consider a contribution to $\langle \exp(zT)\rangle$ that contains $n_1$ one-bead bracelets, $n_2$ two-bead bracelets, $n_3$ three-bead bracelets, and so on. The total number of propagators involved is then

$$n = n_1 + 2n_2 + 3n_3 + \cdots \tag{110}$$

The number of ways to divide $n$ propagators into $n_1$ groups of one, $n_2$ groups of two, $n_3$ groups of three, and so on, is

$$R(n_1, n_2, n_3, \ldots) = \frac{n!}{n_1!\, n_2!\, n_3! \cdots (1!)^{n_1}\, (2!)^{n_2}\, (3!)^{n_3} \cdots} \ , \tag{111}$$

where we also take into account the indistinguishability of the propagators in each group. Now, a $k$-bead bracelet can be built from $k$ propagators in

$$2(k-1)\, 2(k-2)\, 2(k-3) \cdots 2 = \frac{2^k k!}{2k}$$

[58]This was the reason for insisting on the normalisation of the total strength.

ways. The contribution under consideration has, in addition, a factor $z^n/n!$. Putting everything together and summing over all values of $n_1, n_2, n_3, \ldots$ then gives us

$$\begin{aligned}
\Omega(z) &= \sum_{n_{1,2,3,\ldots}\geq 0} \frac{1}{n_1!}\left(\frac{2z}{2}B_1\right)^{n_1} \frac{1}{n_2!}\left(\frac{(2z)^2}{4}B_2\right)^{n_2} \frac{1}{n_3!}\left(\frac{(2z)^3}{2}B_3\right)^{n_3}\cdots \\
&= \exp\left(\sum_{k\geq 1}\frac{(2z)^k}{2k}B_k\right) = \exp\left(-\frac{1}{2}\sum_{\mathbf{n}}^{*}\log(1-2z\sigma_{\mathbf{n}}^2)\right) \\
&= \left(\prod_{\mathbf{n}}^{*}(1-2z\sigma_{\mathbf{n}}^2)\right)^{-1/2} \quad . \qquad (112)
\end{aligned}$$

In this derivation, the factor $1/(2k)$ is precisely the symmetry factor of the $k$-bead bracelet, and the occurrence of the 2 in the $(2z)^k$ is due to the bosonic nature of the propagators, by which they can always be connected in two ways in any bracelet. For the block diaphony we have

$$\Omega_B(z) = \frac{1}{(1-z/p)^p} \quad , \qquad (113)$$

and for the one-dimensional Euler diaphony

$$\Omega_E(z) = \prod_{n\geq 1}\left(1-\frac{6z}{\pi^2 n^2}\right)^{-1} = \frac{\sqrt{6z}}{\sin(\sqrt{6z})} \quad . \qquad (114)$$

The one-dimensional Gulliver diaphony has

$$\Omega_G(z) = \prod_{n\geq 1}(1-s^n x)^{-1} \quad , \quad x = \frac{1-s}{s}z \quad . \tag{115}$$

### 6.3.4 The diaphony distribution for large $N$

The actual probability density $P(t) = \mathrm{Prob}\,(T(\mathbf{X}) = t)$ is given by the inverse Laplace transform

$$\begin{aligned} P(t) &= \frac{1}{2\pi i}\int\limits_\Gamma dz\; \exp(-zt)\,\Omega(z) \\ &= \frac{1}{2\pi}\int\limits_\Gamma dz\;\; \exp\left(-zt - \frac{1}{2}\sum_{\mathbf{n}}^{*}\log(1-2z\sigma_{\mathbf{n}}^2)\right) \quad , \end{aligned} \tag{116}$$

where the integration contour $\Gamma$ runs from $-i\infty$ to $+i\infty$, passing to the left of every singularity of $\Omega(z)$, that is it crosses the real axis below $1/(2\max_{\mathbf{n}}\sigma_{\mathbf{n}}^2)$. For some diaphonies we can write it in (almost) closed form: for the block diaphony

$$P_B(t) = \frac{p^p}{(p-1)!}t^{p-1}\;\exp(-tp) \quad , \quad p = ((2c+1)^d - 1)/2 \quad , \tag{117}$$

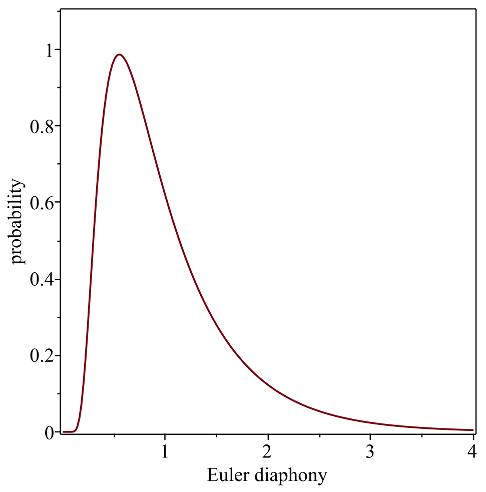
and for the Euler diaphony (again employing Poisson's formula)

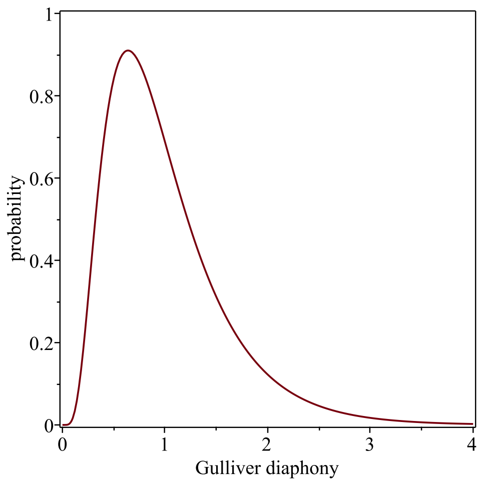
$$P_E(t) \;=\; \sum_{n\geq 1}(-)^{n-1}\frac{n^2\pi^2}{3}\;\exp\left(-\frac{n^2\pi^2 t}{6}\right)$$

$$= \sqrt{\frac{3}{2\pi t}} \sum_{n=-\infty}^{\infty} \left( \frac{3(2n+1)^2}{t^2} - \frac{1}{t} \right) \exp\left( -\frac{3(2n+1)^2}{2t} \right) \quad . \tag{118}$$

Finally, for the one-dimensional Gulliver diaphony we find

$$\begin{aligned} P_G(t) &= \sum_{m\geq 1} \exp\left( \frac{-t}{(1-s)s^{m-1}} \right) R_m(s) \quad , \\ R_1(s) &= \frac{1}{(1-s)} \prod_{n\geq 1} (1-s^n)^{-1} \quad , \\ R_m(s) &= -\frac{s^{m-2}}{1-s^{m-1}} R_{m-1}(s) \quad , \quad m \geq 2 \quad . \end{aligned} \tag{119}$$





The plots show the large-$N$ probability density of the one-dimensional Euler and Gulliver ($s = 0.5$) diaphonies. For large $t$ the distribution decays exponentially. The maximum probability is attained for $t$ values appreciably lower than the mean value 1. At $t = 0$, the Euler diaphony distribution $P_E(t)$ has an essential singularity. For all diaphonies that have an infinite number of nonzero strengths the distribution $P(t)$ must have the form

$$P(t) = \sum_n c_n \exp(-a_n t) \quad , \quad \sum_n c_n = 0 \quad ,$$

where the $c_n$ asymptotically go to zero while the $a_n$ increase without bound[59].

[59]Otherwise the total strength would not be finite.

As soon as $\Re(t) < 0$ the exponentials will explode and $P(t)$ is no longer finite[60]. We therefore conjecture that $t = 0$ is a singular point of $P(t)$ if the number of nonzero strengths is infinite.

### 6.3.5 The saddle-point approximation

In the computation of $P(t)$ we may use a saddle-point approximation, as follows. We can write

$$P(t) = \frac{1}{2\pi i}\int\limits_{\Gamma} dz\ \exp(\phi_t(z)) \quad , \quad \phi_t(z) = -zt - \frac{1}{2}\sum_{\mathbf{n}}^{*}\log(1-2z\sigma_{\mathbf{n}}^2) \ . \tag{120}$$

We can look for the extremal point $z_0$ of $\phi_t(z)$:

$$\phi_t'(z_0) = \sum_{\mathbf{n}}^{*}\frac{\sigma_{\mathbf{n}}^2}{1-2z_0\sigma_{\mathbf{n}}^2} - t = 0 \quad , \quad z_0 < 1/(2\max_{\mathbf{n}}\sigma_{\mathbf{n}}^2) \ . \tag{121}$$

Then we can write the saddle-point approximation as

$$\begin{aligned} \phi_t(z) &\approx \phi_t(z_0) + \frac{1}{2}(z-z_0)^2\phi_t''(z_0) \quad , \quad \phi_t''(z_0) = \sum_{\mathbf{n}}^{*}\frac{2\sigma_{\mathbf{n}}^4}{(1-2z_0\sigma_{\mathbf{n}}^2)^2} \ , \\ P(t) &\approx \sqrt{\frac{2\pi}{\phi_t''(z_0)}}\ \exp(\phi_t(z_0)) \ . \end{aligned} \tag{122}$$

[60]I have checked this numerically for the Gulliver diaphony.

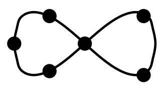
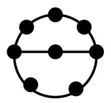
For the block diaphony this is simply the Stirling approximation for the prefactor $p^p/(p-1)!$, good to better than one percent even at $p \approx 10$. For the Euler and Gulliver diaphonies the approximation is also excellent. In practice, to obtain the saddle-point results it is best to take $z_0$ as the independent variable, and compute both $t$ and $P(t)$ as a function of $z_0$ between $-\infty$ and the first singularity. Note that $t \to \infty$ corresponds to having $z_0$ sidling up te the first singularity. The saddle-point value $z_0 = 0$ corresponds, very properly, to the expectation value $t = 1$. The limit $t \downarrow 0$ is reflected in $z_0 \to -\infty$. For the Block diaphony we have the exact relation $z_0 = p(1 - 1/t)$.

### 6.3.6 $1/N$ corrections to the diaphony distribution

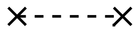
So far we have discussed only the $N \to \infty$ limit for $P(t)$. Io order to obtain the leading correction in the $1/N$ expansion we must take into account the following two effects. In the first place, concomitant with every factor $z^q$ the product of all bracelets contains the factor $N^{\underline{q}}/N^q \approx 1 - q(q-1)/2N$. This $1/N$ effect can be represented by taking

$$\left(1 - \frac{z^2}{2N}\frac{\partial^2}{(\partial z)^2}\right)\Omega(z) =$$
$$\Omega(z)\left[1 - \frac{z^2}{2N}\left(\left(\sum_{\mathbf{n}}^{*}\frac{\sigma_{\mathbf{n}}^2}{1-2z\sigma_{\mathbf{n}}^2}\right)^2 + \sum_{\mathbf{n}}^{*}\frac{2\sigma_{\mathbf{n}}^4}{1-2z\sigma_{\mathbf{n}}^2}\right)\right] \quad . \tag{123}$$

In the second place, we must include diagrams that have one vertex less than the number of its propagators[61]: these are of the forms

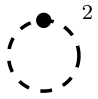
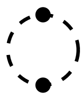
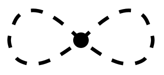
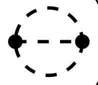

and


with any number of vertices added onto the various propagators. It is seen that $p$-point vertices effectively carry a coupling constant $N^{-(p-2)/2}$. It is useful to introduced the *dressed propagator*



$$\equiv \tilde{\beta}(z;\mathbf{x}) = \sum_{\mathbf{n}}^{*} \frac{2z\sigma_{\mathbf{n}}^2}{1-2z\sigma_{\mathbf{n}}^2} \exp(2i\pi\mathbf{n}\cdot\mathbf{x}) \quad , \tag{124}$$

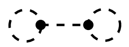
denoted by a dashed line, in terms of which we can write the generating function including its $1/N$ corrections as

$$\left\langle e^{zT} \right\rangle = \Omega(z) \Bigg( 1 - \frac{1}{8N}$$

$$- \frac{1}{4N}$$

$$+ \frac{1}{8N}$$

$$+ \frac{1}{12N}$$

$$\Bigg) \tag{125}$$

Note that all these diagrams carry their 'natural' symmetry factor. The first and third of the four diagrams cancel one another and we find

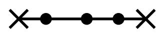
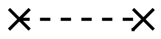
$$\left\langle e^{zT} \right\rangle = \Omega(z) \left( 1 - \frac{1}{4N} \int d\mathbf{x}\, \tilde{\beta}(z;\mathbf{x})^2 + \frac{1}{12N} \int d\mathbf{x}\, \tilde{\beta}(z;\mathbf{x})^3 \right) \quad . \tag{126}$$

[61]But of these diagrams we need to take only the first power.

ℵ In Eq.(125), there could appear the single remaining connected two-loop vacuum diagram  , with its own symmetry factor 1/8. Since it is one-particle irreducible, it vanishes for diaphonies; not, however, for the $\chi^2$ discrepancy that we shall discuss below (*cf* sect.6.4).

### 6.3.7 The two-point function

As mentioned in sect. 5.1.2 we still have to find the two-point correlation $F_2(t;\mathbf{x}_1,\mathbf{x}_2)$. This can also be done diagrammatically, by computing the averages of $T^k$ keeping $\mathbf{x}_1$ and $\mathbf{x}_2$ fixed and integrating over the $N-2$ other points. The only nonvanishing extra diagrams are of the form

 $\rightarrow$  $= \tilde{\beta}(z;\mathbf{x}_1-\mathbf{x}_2)$ , (127)

that again carry an effective factor $1/N$ because these diagrams also have one vertex less than they have bare propagators. We can therefore write

$$\left\langle e^{zT}\right\rangle_{\mathbf{x}_{1,2}\ \mathrm{fixed}} = \Omega(z)\left(1+\frac{1}{N}\,\tilde{\beta}(z;\mathbf{x}_1-\mathbf{x}_2)\right) \tag{128}$$

and then

$$F_2(t;\mathbf{x}_1,\mathbf{x}_2) = -\left(\int dz\, e^{-tz}\,\Omega(z)\,\tilde{\beta}(z;\mathbf{x}_1,\mathbf{x}_2)\right)\left(\int dz\, e^{-tz}\,\Omega(z)\right)^{-1} \quad . \tag{129}$$

In the saddle-point approximation this become quite simple:

$$F_2(t;\mathbf{x}_1,\mathbf{x}_2) = -\tilde{\beta}(z_0(t);\mathbf{x}_1,\mathbf{x}_2) \quad . \tag{130}$$

For the one-dimensional Euler diaphony we find

$$\tilde{\beta}_E(z;x) = \sum_{n\neq 0} \frac{6z}{\pi^2 n^2 - 6z} \exp(2i\pi n x) \tag{131}$$

which satisfies the differential equation for $0 < x < 1$:

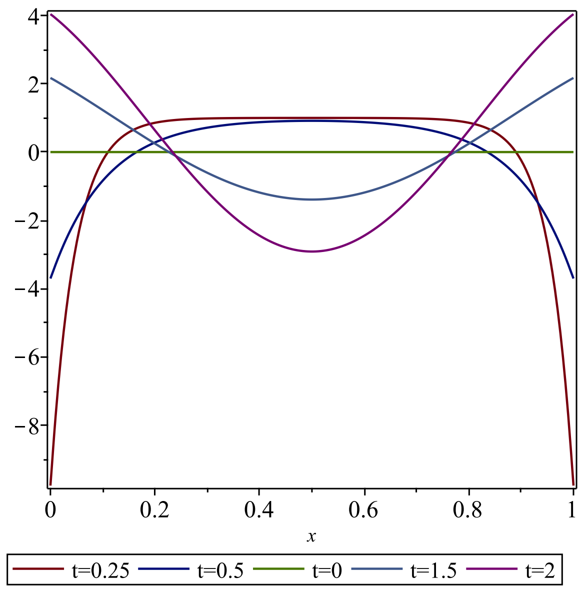
$$\frac{\partial^2}{(\partial x)^2}\tilde{\beta}_E(z;x) + 24z^2\,\tilde{\beta} - E(z;x) = 24z^2 \quad . \tag{132}$$

It has the solutions[62]

$$\tilde{\beta}_E(z;x) = \begin{cases} 1 - \frac{y}{2}\left(e^{-yx} + e^{-y(1-x)}\right)/\left(1 - e^{-y}\right) & , \quad z < 0 \\ 1 - \frac{y}{2}\left(\sin(yx) + \sin(y(1-x))\right)/(1 - \cos(y)) & , \quad z > 0 \end{cases} \quad , \tag{133}$$

[62]For the boundary conditions, see appendix 11.4.1.

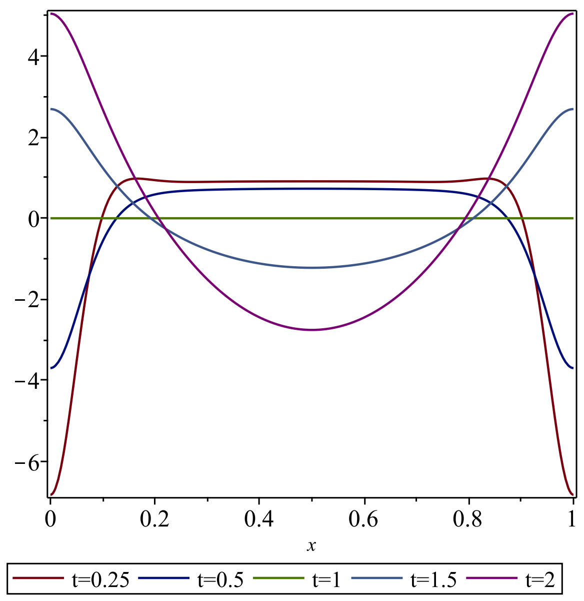
where $y = \sqrt{24|z|}$.



Here we plot the function $\tilde{\beta}_E(z_0(t);x)$ for various values of $t$. In the saddle-point approximation, $\tilde{\beta}(z(t);x_1-x_2)$ is equal to $-F_2(t,x_1,x_2)$. For small $t$ the 'repulsion effect' is evident. For large $t$, in contrast, there is 'attraction' and the points tend to cluster.

For the Block diaphony, where we have the saddle-point approximation $t = p/(p-z(t))$,

$$\tilde{\beta}_B(z_0(t);\mathbf{x}) = (t-1)\left(-1+\prod_{\mu=1}^{d}\frac{\sin((2c+1)\pi\,\mathbf{x}^\mu)}{\sin(\pi\,\mathbf{x}^\mu)}\right) \quad . \qquad (134)$$



Here we plot $\tilde{\beta}_G(z_0(t);x)$ in the saddle-point approximation for the one-dimensional Gulliver diaphony,with $s = 0.5$. This two-point function is not easily obtained in closed form: I have simply summed numerically to large enough $n$. Qualitatively it is quite similar to $\tilde{\beta}_E(z_0(t);x)$.

### 6.3.8 Testing too much: the Dirac limit

The various diaphonies can be considered test of equidistribution. As an illustrative example, the block diaphony $T_B$ tests how well Fourier modes with $|n^\mu| \le c$ are integrated. Surely, if we include more and more modes, the test will become more stringent? We have

$$\sigma(T_B)^2 = \left\langle T_B^2 \right\rangle - \left\langle T_B \right\rangle^2 = 2 \sum_{\mathbf{n}}^{*} \sigma_{\mathbf{n}}^4 = \frac{2}{p} \quad , \tag{135}$$

so that for $c$ very large the variance of the $T_B$ distribution vanishes: the distribution becomes a Dirac delta, and *every* point set[63] ends up with $T_B \approx 1$. For the Gulliver and Euler diaphonies we find, similarly:

$$\begin{aligned} \sigma(T_G)^2 &\approx \left(\frac{1-s}{4}\right)^d \quad \text{for } s \text{ approaching } 1 \quad , \\ \sigma(T_E)^2 &\approx (21/40)^d \quad \text{for } d \text{ very large} \quad . \end{aligned} \tag{136}$$

The result holds generally: if $\sum_{\mathbf{n}}^{*} \sigma_{\mathbf{n}}^2 = 1$, then $\sum_{\mathbf{n}}^{*} \sigma_{\mathbf{n}}^4$ will tend to zero unless a *finite* number of strengths completely out-dominate all the other ones [20]. The 'ultimate test' is no test at all; the message is that one should not test 'ad infinitum', but *when* to stop testing is not clear. In this respect, as in others, Monte Carlo is an art rather than a prescription.

ℵ The limit of 'large number of modes' has its own Central Limit theorem: the sums $\sum_{\mathbf{n}}^{*} \sigma_{\mathbf{n}}^{2k}$ approach zero ever faster for increasing $k$, and therefore the density $P(t)$ takes on a Gaussian form [20].

[63]Because every point set finds itself, eventually, in the ensemble of random iid point sets.

## 6.4 Measures of nonuniformity: $\chi^2$

### 6.4.1 The $\chi^2$ as a discrepancy

The well-known $\chi^2$ density can also be cast in the language of problem classes. In this case we divide the $I^d$ hypercube into $B$ non-overlapping, but not necessarily simply connected or even connected, regions ('*bins*'). We define

$$\theta_n(\mathbf{x}) = \begin{cases} 1 & , \quad \mathbf{x} \text{ inside bin } n \\ 0 & , \quad \mathbf{x} \text{ outside bin } n \end{cases} \quad . \tag{137}$$

We have

$$\theta_n(\mathbf{x})\,\theta_m(\mathbf{x}) = \delta_{mn}\,\theta_n(\mathbf{x}) \quad , \quad \sum_{n=1}^{B}\theta_n(\mathbf{x}) = 1 \quad . \tag{138}$$

The volume of the bins is given by

$$v_n = \int d\mathbf{x}\;\theta_n(\mathbf{x}) \quad , \quad \sum_{n=1}^{B} v_n = 1 \quad . \tag{139}$$

The 'Lego' problem class[64] now consists of functions that are piecewise constant over the bins:

$$f(\mathbf{x}) = \sum_n \;\alpha_n\,\theta_n(\mathbf{x}) \quad . \tag{140}$$

[64]Because the functions look like the 'Lego plots' common in experimental analysis.

The ensemble measure is again Gaussian, with

$$\langle\alpha_n\rangle_L = 0 \quad , \quad \langle\alpha_m\,\alpha_n\rangle_L = \delta_{mn}\,\frac{1}{v_n} \quad . \tag{141}$$

It is this choice that singles out the $\chi^2$ distribution out of all possible 'Lego'-like discrepancies. The integration error,

$$\eta = \frac{1}{N}\sum_{j=1}^{N} f(\mathbf{x}_j) - \int d\mathbf{x}\, f(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{B}\alpha_n \sum_{j=1}^{N}(\theta_n(\mathbf{x}_j) - v_n) \quad , \tag{142}$$

has expected square

$$\left\langle\eta^2\right\rangle_L = \frac{1}{N^2}\sum_{n=1}^{B}\frac{1}{v_n}\left(\sum_j(\theta_n(\mathbf{x}_j) - v_n)\right)^2 \quad , \tag{143}$$

and this leads us to propose the measure of nonuniformity to be

$$T_L(\mathbf{X}) = \frac{1}{N}\sum_{n=1}^{B}\frac{1}{v_n}\left(\sum_j(\theta_n(\mathbf{x}_j) - v_n)\right)^2 \quad . \tag{144}$$

This is exactly the $\chi^2$ of the point set $\mathbf{X}$ tested against the hypothesis of uniform distribution over $I^d$.

ℵ The fact that the bins may have any shape, or consist of disconnected parts, is the reason for considering (almost) any empirical test of a PRNG as a $\chi^2$ test of uniformity in a possibly many-dimensional space with weird-looking bins (*e.g.* the poker test).

### 6.4.2 Large-$N$ results for $\chi^2$

The bare propagator for $T_L$ is

$$\beta_L(\mathbf{x},\mathbf{y}) = -1 + \sum_n \theta_n(\mathbf{x})\theta_n(\mathbf{y})/v_n \quad . \tag{145}$$

It is again tadpole-free, $\int \beta(\mathbf{x},\mathbf{y})\,d\mathbf{y} = 0$; but it is not translation invariant and hence not one-vertex reducible except when all volumes are equal, $v_n = 1/M$. On the other hand, we have the nice property

$$\int d\mathbf{z}\;\beta_L(\mathbf{x},\mathbf{z})\,\beta_L(\mathbf{z},\mathbf{y}) = \beta_L(\mathbf{x},\mathbf{y}) \quad . \tag{146}$$

This means that all bracelets come out the same:

$$B_k = B_1 = \int d\mathbf{x}\,\beta(\mathbf{x},\mathbf{x}) = M - 1 \quad . \tag{147}$$

We find immediately that, for $N \to \infty$[65],

$$\begin{aligned} \Omega_L(z) &= (1-2z)^{-(B-1)/2} \quad , \\ P_L(t) &= \frac{1}{2\Gamma\left(\frac{B-1}{2}\right)}\left(\frac{t}{2}\right)^{(B-3)/2} \exp\left(-\frac{t}{2}\right) \quad . \end{aligned} \tag{148}$$

[65]We take $B$ to be odd for simplicity here.

The saddle-point is reached for

$$z_0(t) = \frac{1}{2}\left(1 - \frac{B-1}{t}\right) \quad . \tag{149}$$

The number $B-1$ is called the *number of degrees of freedom*. Note that for $B=1$ we have only the single bin $I^d$ itself with trivially $T_L(\mathbf{X}) = 0$ for all point sets, hence no degrees of freedom.

ℵ The $\chi^2$ discrepancy has expectation value $B-1$ rather than 1. If we renormalize by scaling $t$ to $t/(B-1)$ the probability density is exactly that for the Block diaphony with $p=(B-1)/2$; a curious result since the two notions of nonuniformity are defined in totally different ways!

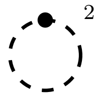
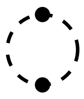
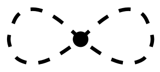
### 6.4.3 Two-point function and $1/N$ corrections for $\chi^2$

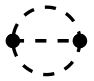
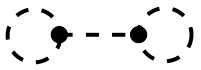
Because of property (146) we can immediately find the dressed two-point function:

$$\begin{aligned} \times\text{-----}\times \quad &= \quad \tilde{\beta}_L(z;\mathbf{x},\mathbf{y}) = \frac{2z}{1-2z}\,\beta_L(\mathbf{x},\mathbf{y}) \\ &= \quad \frac{2z}{1-2z}\left(-1 + \sum_n \frac{\theta_n(x)\theta_n(y)}{v_n}\right) \quad . \end{aligned} \tag{150}$$

As before, we find repulsion for small $t$[66]. The $1/N$ correction terms now include an extra diagram: we have

$$\Omega_L(z) = (1-2z)^{-(M-1)/2}\left(1+\frac{\mathcal{A}}{N}\right) \quad ,$$

$\mathcal{A} = -\frac{1}{8}$  $^2 - \frac{1}{4}$  $+\frac{1}{8}$ 

$+\frac{1}{12}$  $+\frac{1}{8}$ 

$$= \frac{z^2}{2(1-2z)^2}\left(\sum_n \frac{1}{v_n} - M^2 - 2M + 2\right)$$
$$+\frac{z^3}{3(1-2z)^3}\left(5\sum_n \frac{1}{v_n} - 3M^2 - 6M + 4\right) \quad . \qquad (151)$$

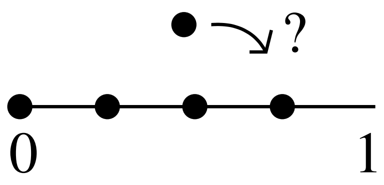
[66]In the saddle-point approximation.

# 7 Superuniform point sets

Point sets with a discrepancy/diaphony (considerably) lower than that expected for truly random ones are called superuniform.

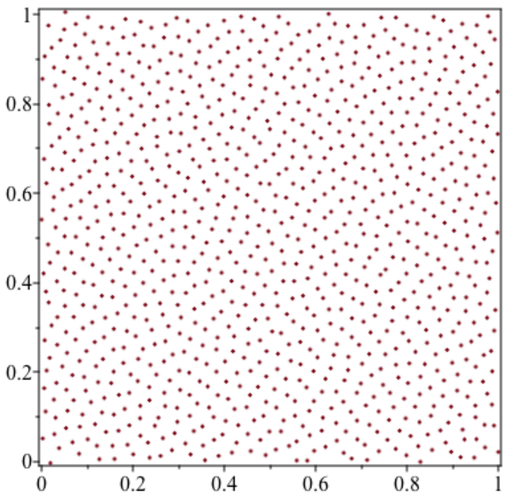
## 7.1 Fixed point sets *vs* streams

We shall discuss several approaches to the construction of point sets with low discrepancy/diaphony. Here it becomes important to distinguish between fixed-size point sets and streams. If a point set of $n$ points has very low diaphony, then adding an $(n+1)^{\text{th}}$ one will be problematic from the uniformity point of view. This point set of 4 points at 0,1/4,1/2,3/4 has the smallest diaphony possible. Where can we put the 5th one without increasing the diaphony (note that points at 0 and at 1 coincide)? A point set of 5 points with minimal diaphony has its points at 0,1/5,2/5,3/5,4/5.



### 7.1.1 Diaphony minimisation

For given $N$ there exists the point set with minimal discrepancy/diaphony. *Finding* this point set is (in more than one dimension) not feasible. In general the best we can hope for is to obtain point sets with very low diaphony. Two strategies can be envisaged: either minimising the diaphony by shifting points around, or invoking some rule. Numerical minimisation typically relies on (a) descending methods that use expressions for the *gradient* of the diaphony[67], or (2) the Metropolis algorithm (*cf* sect.9.4.1). Both methods are very slow.



The plot shows a two-dimensional low-discrepancy point set with $N = 1000$. It was obtained by student T.Blank in about a week's computing time, by descending the Gulliver diaphony with $s = 0.5$. The obtained diaphony is $7.2\ 10^{-8}$. It is clear that simply running a PRNG many times and selecting the 'best' point set is a hopeless strategy.

[67]This cannot be done for discrepancy since the local discrepancy is by construction not differentiable where it sounts.

### 7.1.2 Korobov sequences: good lattice points

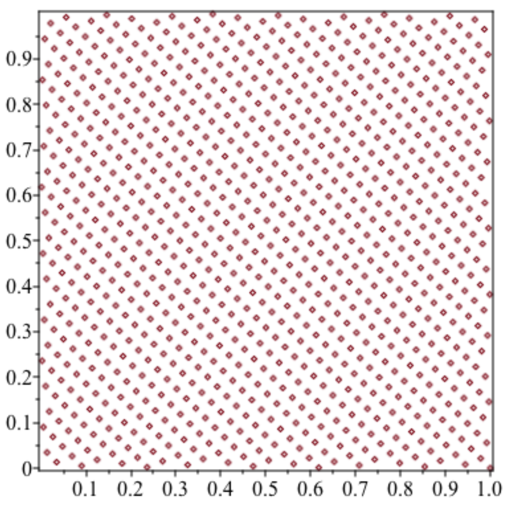
A widely used strategy is that of Korobov sequences, or the method of *good lattice points*. For a $d$-dimensional $N$-point set this consists of identifying a 'good' lattice vector with natural coefficients

$$\vec{g} = (g^1, g^2, \ldots, g^d) \ , \tag{152}$$

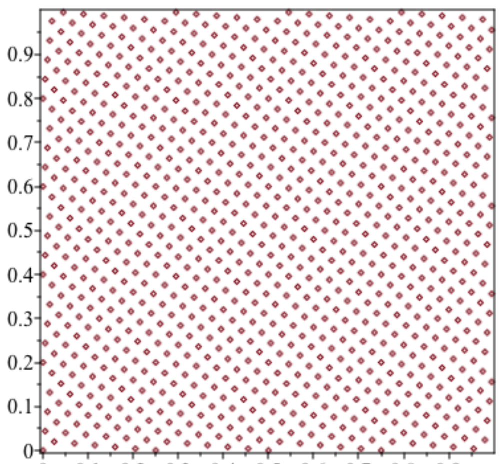
and then the points are defined by

$$\mathbf{x}_k = \left( \left\{ \frac{k\, g^1}{N} \right\}, \left\{ \frac{k\, g^2}{N} \right\}, \ldots, \left\{ \frac{k\, g^d}{N} \right\} \right) \quad , \quad k = 1, 2, \ldots, N \ . \tag{153}$$

We can take $g^1 = 1$ without loss of generality. The other components $g^j$ should at least be relatively prime to $N$ and to one another. One possibility



for $d = 2$ is this: if $N$ equals the $n^{\text{th}}$ Fibonacci number $F_n$ we can take $\vec{g} = (1, F_{n-1})$. The plot shows the result for $F_{16} = 987$, $F_{15} = 610$. The uniformity is obvious, but so is the lack of rotational symmetry, especially when compared to the plot in sect.7.1.1. The second plot has 1000 points, using the lattice vector $\vec{g} = (85/1000, 948/1000)$, found by student G. van Bergen. Its Gulliver diaphony ($s = 0.5$) is



lattice vector = (85,948)/1000

$2.22\ 10^{-11}$, quite a bit smaller than that of the 1000-point plot in sec.7.1.1. Essentially, such point sets have the same advantages and drawbacks as 'hypercubic' lattices. The low value of diaphony is, in some way, a result of specialising the lattice vector to *that particular definition* of nonuniformity. In my opinion fixed-$N$ point sets are of limited use since for a realistic nontrivial integration problem it is not known *a priori* what $N$ ought to be. An important insight, however, is the following: in the complete point set, the set of the $j^{\text{th}}$ coordinates of the points, $(x_1^j, x_2^j, \ldots, x_N^j)$, are precisely the minimal-diaphony, equidistant point sets in one dimension, thus explaining in some qualitative way the low diaphony of the full $d$-dimensional point set[68].

[68]However, the vector $\vec{g} = (1, 1, \ldots, 1)$ would give the same projections but an unacceptable more-dimensional set. Hence the requirement that the components of $\vec{g}$ be mutually prime.

## 7.2 QRNG algorithms

In view of the above, the more attractive idea is to search for low-diaphony *stream* algorithms, or QRNG: *Quasi-random number generators.*

### 7.2.1 Richtmeyer-Kronecker streams

Korobov sequences eventually 'run out of steam' since if we continue the rule (153) beyond $k = N$ the points will start to repeat. We can therefore envisage to let $N$ approach infinity, and of course then the $g^j$ have to approach infinity as well: the rational numbers $g^j/N$ have to become *irrational*. This is the idea of Richtmeyer sequences: rather than identifying a lattice vector $\vec{g}$ we search for 'irrational vectors'

$$\vec{g} = (\theta^1, \theta^2, \ldots, \theta^d) \ , \tag{154}$$

where the numbers $\theta^j$ are all irrational numbers that are also mutually irrational[69]. The quasi-random numbers $\mathbf{x}_k$ are then given by

$$\mathbf{x}_k = (\{k\,\theta^1\}, \{k\,\theta^2\} \ldots, \{k\,\theta^d\}) \ . \tag{155}$$

The suitability of the irrationals $\theta^j$ can be investigated using their continued-fraction representation, that we shall now discuss.

[69]That is, every ratio $\theta^i/\theta^j$ is also irrational.

### 7.2.2 Excursion into fractions (cont'd)

Let $\theta$ be a number in $(0,1)$. Then

$$\frac{1}{\theta} = a + \theta' \quad , \quad a = \left\lfloor \frac{1}{\theta} \right\rfloor \tag{156}$$

so that $a$ is an natural number and $\theta'$ is a number in $[0,1)$. We can therefore repeat this procedure to arrive at the *continued-fraction representation* of $\theta$:

$$\theta = \cfrac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4 + \cdots}}}} \equiv [a_1, a_2, a_3, a_4, \ldots] \ . \tag{157}$$

If $\theta$ is a rational number, $a_j$ will become infinite for some $j$ and the fraction stops there; we can then write $\theta = [a_1, a_2, \ldots, a_{j-1}]$. For irrational $\theta$ the continued-fraction representation continues forever. Some examples:

$$\begin{aligned}
\pi - 3 &= [7,15,1,292,1,1,1,2,1,3,1,14,2,1,1,2,2,2,2,\ldots] \ , \\
(\sqrt{5}-1)/2 &= [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,\ldots] \ , \\
\sqrt{26}-5 &= [10,10,10,10,10,10,10,10,10,10,10,10,10,10,\ldots] \ , \\
\sqrt{3}-1 &= [1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,\ldots] \ , \\
\sin(1) &= [1,5,3,4,19,2,2,2,2,2,7,2,2,1,136,3,20,3,1,3,\ldots] \ , \\
2^{1/3}-1 &= [3,1,5,1,1,4,1,1,8,1,14,1,10,2,1,4,12,2,3,\ldots] \ , \\
e-2 &= [1,2,1,1,4,1,1,6,1,1,8,1,1,10,1,1,12,1,1,14,\ldots] \ .
\end{aligned} \tag{158}$$

If the continued-fraction expansion is periodic, then $\theta$ will be the solution of a quadratic equation with integer coefficients, a *quadratic irrational.* Therefore all other irrational numbers, like $2^{1/3}$, have continued-fraction expansions that are *aperiodic.*

$\aleph$ The fact that the continued-fraction coefficients for most irrational numbers do not form a periodic pattern might lead one to propose these as a source of 'truly random' integers. This is a bad idea, since (a) the irrational number would have to be known to many millions of digits, and (b) non-periodicity does not imply randomness (*cf* Eq.(158) for $e-2$), or even a *known* distribution of integers.

### 7.2.3 Rational approximations to irrationals

Truncating the continued-fraction representation gives us a method to approximate numbers by rational numbers:

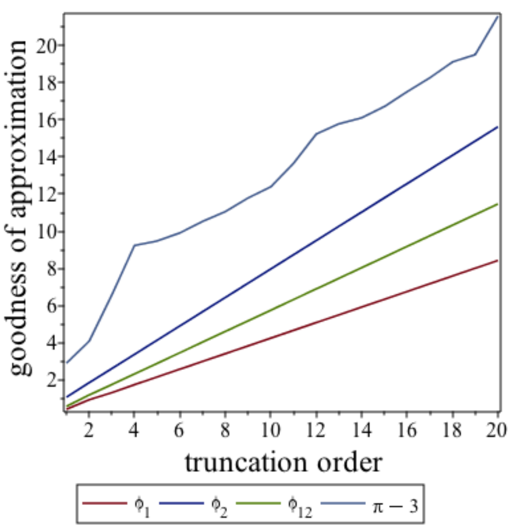
$$\begin{aligned} \theta \quad \approx \quad & \theta_n = \frac{p_n}{q_n} \ , \\ & p_0 = 0 \ , \quad q_0 = 1 \ , \\ & p_1 = 1 \ , \quad q_1 = a_1 \ , \\ & p_n = a_n p_{n-1} + p_{n-2} \ , \quad q_n = a_n q_{n-1} + q_{n-2} \ . \end{aligned} \tag{159}$$

We see that if $a_j$ becomes infinite, then $\theta$ is the rational number $p_{j-1}/q_{j-1}$. In some sense, therefore, the number $\pi$ is *almost* rational since $a_4 = 292$ is so large. In fact, replacing 292 by infinity gives the Chinese approximation $\pi \sim 355/113$, which gets 6 decimals correct[70]. We can also claim to know the *most irrational number in the universe*: it is that number $\phi_1$ for which all the coefficients $a_j$ are 1, and we find[71]

$$\phi_1 = \frac{1}{1+\frac{1}{1+\frac{1}{1+\cdots}}} = \frac{1}{1+\phi_1} \quad \Rightarrow \quad \phi_1 = \frac{1}{2}\left(\sqrt{5}-1\right) \sim 0.618\cdots \tag{160}$$

[70]This fraction is called the *Milü*, established by Zu Chongzhi (429-500 AD).

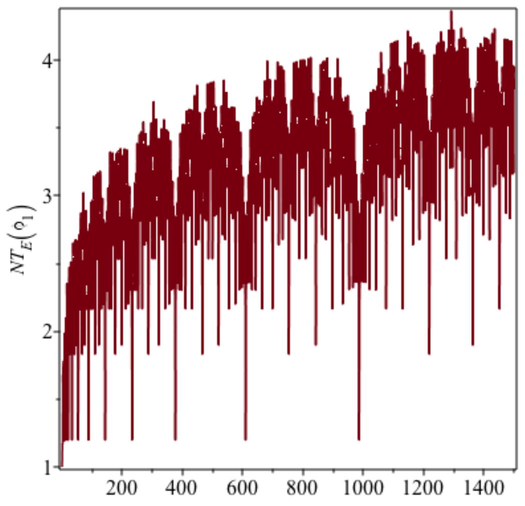
[71]Note how, in Eq.(160), the infinite tail of continued-fraction coefficients is itself replaced by $\phi_1$: a form of *telescoping*. We will meet a similar situation later on, in section 9.2.3.



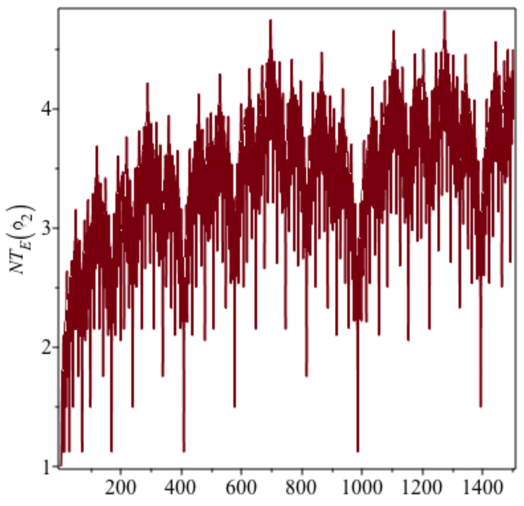
*i.e.* the golden ratio. This plot shows the goodness of the rational approximation (the number of correct decimal digits) for $\phi_1$, and also for $\phi_2 = [2, 2, 2, 2, \ldots] = -1 + \sqrt{2}$, $\phi_{12} = [1, 2, 1, 2, 1, 2, \ldots] = -1 + \sqrt{3}$, and $\pi - 3$. The almost-rationality of $\pi$ is seen from the jump in goodness from $n = 3$ to $n = 4$. The smaller the continued-fraction coefficients, the worse the rational approximation: no curve exists below that for $\phi_1$.

### 7.2.4 Almost-equidistancy for Richtmeyer sequences

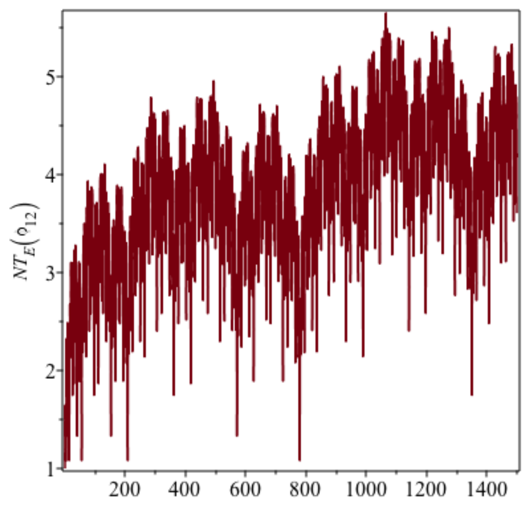
Let us consider Richtmeyer sets $x_k$, $k = 1, \ldots, N$, where $x_k = \{k\,\phi_1\}$, $\phi_1 = [1, 1, 1, 1, \ldots]$ being the golden ratio. We plot the running value of $NT_E(\phi_1)$,



using the Euler diaphony $T_E$. The number $\phi_1$ is approximated by ratios of Fibonacci numbers, $F_{n-1}/F_n$, the approximation improving for increasing $n$. Every time $N = F_m$ for some $m$, the distribution of points is *almost* equidistant with $x_k \sim q(k)/F_n$, where $q(k) \in [1, F_n]$ is some function of $k \in [1, F_n]$. That is, when $N = F_n$ the distribution of points has essentially the smalles possible diaphony. This is the source of superuniformity: in between these 'optimal' values the diaphony cannot grow too much before coming down again. From appendix 11.5.1 we see that the Fibonacci numbers approximate $F_n \sim (1.618)^{-n}$ so that the diaphony moves out further and further as $n$ increases. This is the reason



for the logarithmic term in the Roth bound. A similar phenomenon is observed for $NT_E(\phi_2)$ with $\phi_2 = [2, 2, 2, 2, \ldots] = -1 + \sqrt{2}$, except that the returns to almost-equidistancy are now further apart, since $1/\phi_2 \sim 2.414$. As in the previous case a fractal pattern is evident, corresponding to a rational approximation to the irrational that is less than optimal. The quality of the diaphony is seen to depend on how close together the 'optimal' values are. An intermediate case is that of



a mixture of coefficients 1 and 2, for instance $\phi_{12} = [1, 2, 1, 2, \ldots] = -1 + \sqrt{3}$ which is plotted here. The 'optimal' values (see appendix 11.5.1) are spaced $\sim (1.932)^n$, and the diaphony is minimal for odd $n$, next-to-minimal for even $n$. The minima are closer together than for $\phi_2$. We observe similar behaviour for other irrationals such as $[1, 1, 2, 1, 1, 2, 1, 1, 2, 1, 1, \ldots] = -1 + \sqrt{5/2}$.

### 7.2.5 van der Corput streams

In the above Richtmeyer sequences, the (one-dimensional) distribution of points is *almost* equidistant at the 'optimal' values of $N$. We can improve on that using the following approach. The *van der Corput transform* $\phi_b(n)$ of an integer $n \geq 0$ in base $b$ is defined as follows: if $n$ has the $b$-ary expansion

$$n = n_0 + n_1 b + n_2 b^2 + n_3 b^3 + \cdots \tag{161}$$

then

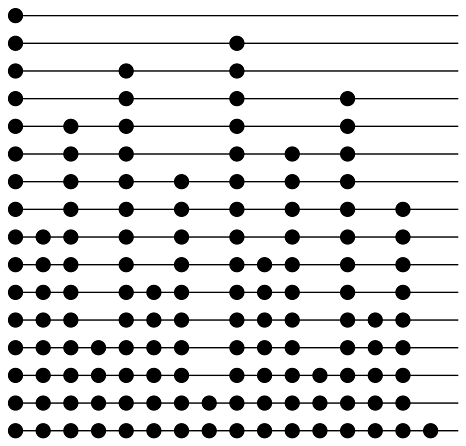
$$\phi_b(n) = n_0 b^{-1} + n_1 b^{-2} + n_2 b^{-3} + n_4 b^{-4} + \cdots \quad . \tag{162}$$

An explicit algorithm is given here: for $n$ given, it runs in time $\mathcal{O}(\log(n))$.

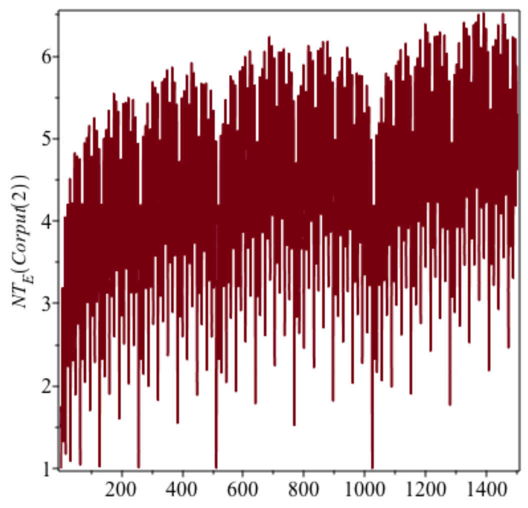
**Algorithm 4** The van der Corput transform $\phi_b(n)$

```
{The base is $b$, the input number is $n > 0$. }
$m \leftarrow n$   {this avoids changes in $n$}
$k \leftarrow 0$
while $m > 0$ do
    $k \leftarrow k + 1$ , $c_k \leftarrow m \bmod b$   {get the next digit}
    $m \leftarrow (m - c_k)/b$
end while
$j \leftarrow k$ ; $\phi \leftarrow 0$   {start at the least significant digit; $\phi$ is the output}
while $j > 0$ do
    $j \leftarrow j - 1$ , $\phi \leftarrow (c_j + \phi)/b$
end while
return $\phi$
```

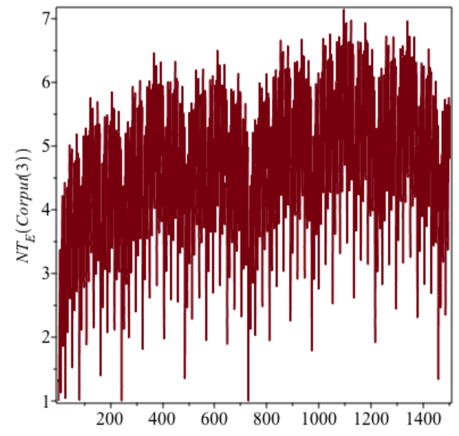
In one dimension, the point set $\{\phi_b(1), \phi_b(2), \ldots, \phi_b(N)\}$ will be exactly equidistant if $N = b^p$ for some power $p$: we return to the 'best' case as

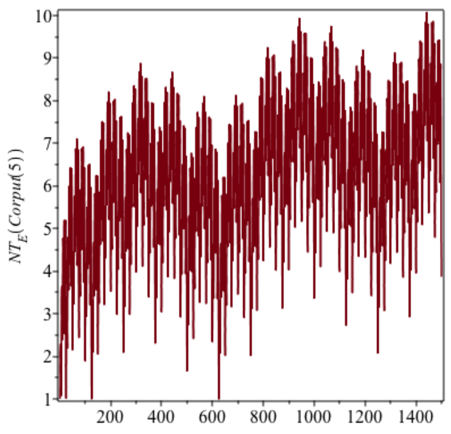


often as possible for given $b$. Here we represent how the first van der Corput numbers $\phi_2(n)$ $(= 0, 2, \ldots, 15)$ are filling the unit interval $[0, 1)$. At line 1 $(n = 0)$ the discrepancy is automatically minimal. In line 2 the 2 points are spaced equidistantly, as they are in line 4, 8, and 16. Every time $2^k$ points have been filled in the discrepancy returns to its minimal value. In between, at line



6, say, the discrepancy is 'almost optimal'. This is evidenced by looking at $NT_E(\mathbf{X})$ as before. The evolution of the Euler diaphony shows an even more explicitly fractal pattern, and the optimal values at $N = 2^p$ have the absolute minimal diaphony. Inbetween, for $N = 2^p + 2^{p-1}$, say, the diaphony is slightly higher, and for $N = 2^p + 2^{p-2} \pm 2^{p-2}$ it is higher again. For larger values of the base $b$ the minima are necessarily spread further apart. Below we give similar results for $b = 3$ and $b = 5$.



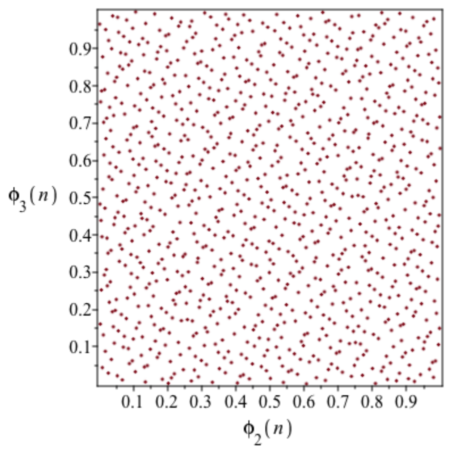


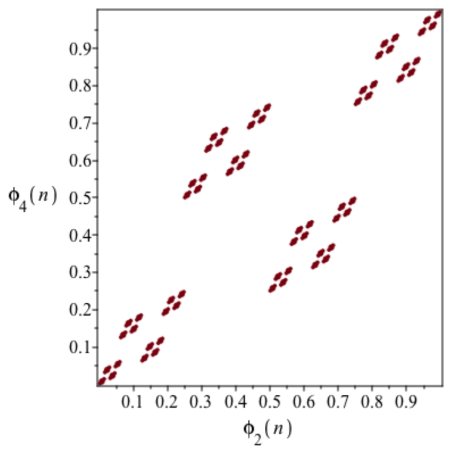
### 7.2.6 Van der Corput sequences in more dimensions

In two dimensions we may consider the sequences

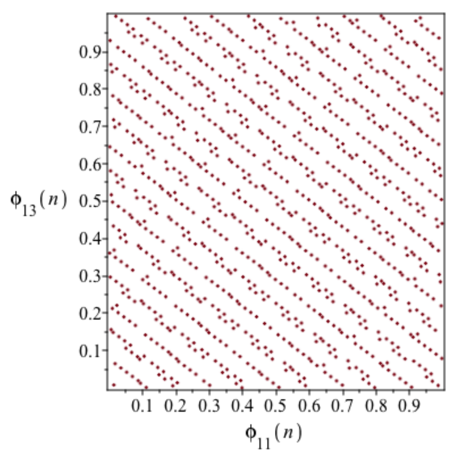
$$\mathbf{x}_n = (\phi_2(n), \phi_3(n)) \quad \text{and} \quad \mathbf{x}_n = (\phi_2(n), \phi_4(n)) \quad . \tag{163}$$

The first 1000 points are displayed below.





Whereas the choice $b = 2, 3$ is quite acceptable for a superuniform point set, the choice $b = 2, 4$ is extremely unlucky: the pattern is fractal, with



diamonds built up from smaller diamonds and so on. It is clear that in more dimensions the bases must be relatively prime. But that implies that with increasing dimension $d$ the bases must increase *at least as fast* as the lowest $d$ primes. Going back to the one-dimensional projections of the point set, that implies that some 'optimal' values are going to be very far apart for appreciable $d$. As an example, we plot the point set for $b = 11, 13$, which is a two-dimensional projection for allvan der Corput sequences for $d \geq 6$. This shows that the discrepancy/diaphony can become large. In [21] the following result is given:

$$L_2^* < \frac{(\log N)^{2d}}{N^2} K_d \quad , \quad K_d = \prod_{j=1}^{d} \frac{(b_j - 1)^2}{\log(b_j)} \quad . \tag{164}$$

so we can determine the value of $N_d$ for which the van der Corput sequence

| $d$ | $K_d$ | $N_d$ |
|---|---|---|
| 2 | 5.253 | $0.1560\ 10^7$ |
| 3 | 52.24 | $0.1807\ 10^{12}$ |
| 4 | 966.4 | $0.1079\ 10^{18}$ |
| 5 | $0.4030\ 10^5$ | $0.3192\ 10^{24}$ |
| 6 | $0.2263\ 10^7$ | $0.2144\ 10^{31}$ |
| 7 | $0.2045\ 10^9$ | $0.3649\ 10^{38}$ |
| 8 | $0.2252\ 10^{11}$ | $0.1087\ 10^{46}$ |

is *guaranteed* to have a discrepancy smaller than the 'random value' $(2^-d - 3^-d)/N$. This is given in the table. Since we only consider an *upper limit* on $L_2^*$ here, the picture is not really so bleak. Nevertheless, the fact that *asymptotically* van der Corput sequences are superuniform does not guarantee useful behaviour for moderate $N$.

### 7.2.7 Niederreiter streams

If the van der Corput sequences deteriorate in higher dimensions because the bases $b$ become large, we may decide to keep the same base $b$ in all $d$ dimensions but change the ordering of the points. This is the idea behind the *Niederreiter sequences*. For given base $b$ (think of $b = 2$) we find functions $p_j(k)$, $(j = 1, 2, \dots, b)$ with the following property: for all $m$, if $k$ runs from 1 to $b^m$ then $p_j(k)$ takes on all values between 1 and $b^m$ as well. In other words, $p_j(k)$ is a permutation of the numbers $(1, 2, \dots, b^m)$. Having found these, the Niederreiter sequence is defined by

$$\mathbf{x}_k = \Big(\phi_b(p_1(k)), \phi_b(p_2(k)), \dots, \phi_b(p_d(k))\Big) \quad . \tag{165}$$

The trick is of course in finding the permutation functions $p_j$. A method of doing so is described in [23] and implemented independently in [24]. The (asymptotic!) bounds on the discrepancy are very much better that those for van der Corput sequences.

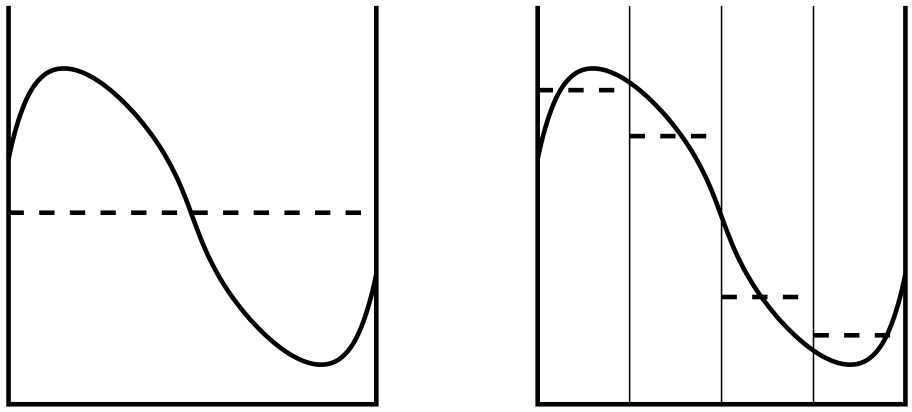
# 8 Variance reduction

Rather than attempting to lower integration errors by using quasi-random point sets we may try to 'reformulate' the integration problem in such a way that the integrand's variance (the $J_2 - {J_1}^2$) becomes smaller: this is called *variance reduction*. A number of strategies to achieve this exist.

## 8.1 Stratified sampling

### 8.1.1 General strategy

It can easily be checked that if a function $f(\mathbf{x})$ has a certain variance, then the function $f(\mathbf{x}) + c$, where $c$ is a constant, has the same variance: the constant piece is integrated with zero error, and it pays to make it as large as possible. To this end we can divide the integration region $\Gamma$ into *strata*, or bins, that do not overlap. The we assign a given number of integration points to each stratum, and perform the Monte Carlo estimate of the integral in each, summing the results for our estimators. Such an approach is called *stratified sampling.*



In the above picture we compare the 'constant' parts of the same integrand with one or four (equally large) strata. The non-constant part of the integrand is smaller for more strata. How to choose the strata, and how many points to choose in each of them, is the nontrivial part.

### 8.1.2 Uniform stratification

Let us, for simplicity, consider integration of $f(\mathbf{x})$ over the unit hypercube $\Omega$ , using uniform iid random numbers. We divide the hypercube into $m$ strata

$\Omega_k$ $(k = 1, 2, \ldots, m)$, each with volume

$$\omega_k = \int\limits_{\Omega_k} d\mathbf{x}\, f(\mathbf{x}) \quad . \tag{166}$$

In each stratum $k$ we choose $N_k$ points, so the total number of integration points is $N = \sum_k N_k$. The *density* of these point in each stratum $k$ is therefore $\theta(\mathbf{x} \in \Omega_k)/\omega_k$, and the expected error is therefore given by

$$\langle\eta\rangle^2_{\text{strat}} = \sum_{k=1}^{m} \frac{V_k}{N_k} \quad , \quad V_k = \int\limits_{\Omega_k} d\mathbf{x}\, \omega_k\, f(\mathbf{x})^2 - \left(\int\limits_{\Omega_k} d\mathbf{x}\, f(\mathbf{x})\right)^2 \quad . \tag{167}$$

We must compare this to the unstratified expected error

$$\left\langle\eta^2\right\rangle = \frac{V_0}{N} \quad , \quad V_= \int\limits_{\Omega} d\mathbf{x}\, f(\mathbf{x})^2 - \left(\int\limits_{\Omega} d\mathbf{x}\, f(\mathbf{x})\right)^2 \quad . \tag{168}$$

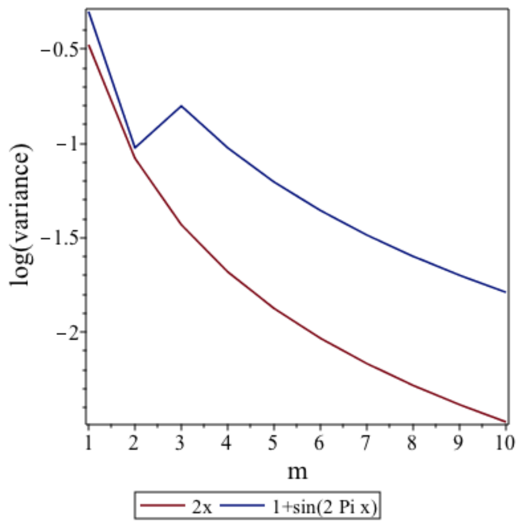
The improvement in the expected error is therefore given by

$$\begin{aligned} \left\langle\eta^2\right\rangle - \left\langle\eta^2\right\rangle_{\text{strat}} &= \sum_{k=1}^{m} \left(\frac{1}{N} - \frac{\omega_k}{N_k}\right) \int\limits_{\Omega_k} d\mathbf{x}\, f(\mathbf{x})^2 \\ &+ \sum_{k=1}^{m} \frac{1}{N_k} \left(\int\limits_{\Omega_k} d\mathbf{x}\, f(\mathbf{x})\right)^2 - \frac{1}{N} \left(\sum_{k=1}^{m} \int\limits_{\Omega_k} d\mathbf{x}\, f(\mathbf{x})\right)^2 \quad . \end{aligned} \tag{169}$$

The last line of the error improvement (169) can be written as

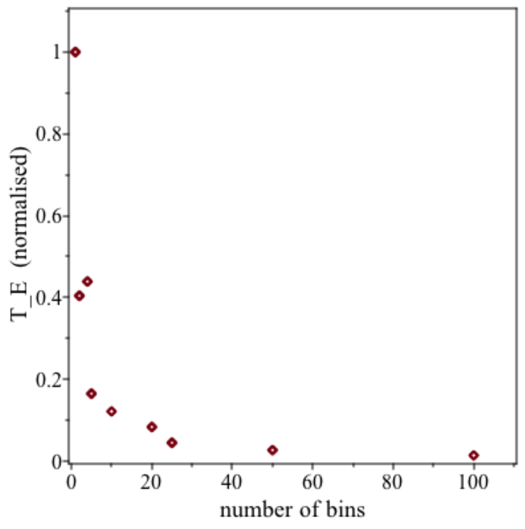
$$\frac{1}{N}\sum_{k\neq\ell}\left(\sqrt{\frac{n_\ell}{n_k}}\int\limits_{\Gamma_k} d\mathbf{x}\, f(\mathbf{x}) - \sqrt{\frac{n_k}{n_\ell}}\int\limits_{\Gamma_\ell} d\mathbf{x}\, f(\mathbf{x})\right)^2$$

and is never negative. The first line can be put to zero by choosing $N_k = N\omega_k$, so that each stratum gets its 'proper share' of the points. This is called *uniform stratification*, and we can see that it never hurts. The picture shows the logarithm of the variance (that is, $N$ times the expected squared error) for the two functions $2x$ and $1 + sin(2\pi x)$. The number of strata runs from 1 to 10, and the lines are to guide the eye. It is clear that stratification can improve the estimated error considerably; on the other hand from comparing the results for 1+sin($2\pi x$) when going from 2 to 3 strata we also see that more strata are not automatically better!



### 8.1.3 Stratification as quasification

A way to understand the effect of (uniform) stratification is to realise that it imposes a certain uniformity on the point set used. The plot shows this.



Starting with a pseudorandom point set $\mathbf{X} = (x_1, \ldots, x_N)$ $(N = 100)$ iid on the unit interval (0,1), we construct $m$ bins,where $m$ is a divisor of $N$. We then redistribute the points into another point set $\mathbf{Y} = (y_1, \ldots, y_N)$ by setting $y_{kN/m+j} \leftarrow x_{kN/m+j} + k/m$, where $k = 0, \ldots, m-1$ and $j = 1, \ldots, N/m$. The resulting Euler diaphony is normalised to the unbinned case $m = 1$. We observe that with increasing $m$ the diaphony of the 'evened-out' point set $\mathbf{Y}$ *generally* decreases. The decrease is (very roughly) as $1/m$: in this example, for $m = N = 100$ the diaphony has decreased by a factor 0.014.

### 8.1.4 An example: `VEGAS`

### 8.1.5 An example: `PARNI`

## 8.2 Importance sampling

### 8.2.1 General strategy

The general idea behind importance sampling is to concentrate integration points where it counts. In straightforward Monte Carlo we have, as we have seen,

$$J_1 = \int\limits_{\Gamma} d\mathbf{x}\, g(\mathbf{x})\, w(\mathbf{x}) \quad , \quad w(\mathbf{x}) = \frac{f(\mathbf{x})}{g(\mathbf{x})} \quad , \tag{170}$$

where de density $g(\mathbf{x})$ should not vanish for any $\mathbf{x}$ for which $f(\mathbf{x}) \neq 0$. Now, if we can arrange for the weights $w(\mathbf{x})$ to fluctuate as little as possible, then we may expect a good error. In the ideal case where $f(\mathbf{x}) = c\, g(\mathbf{x})$ with constant $c$, the error will be zero, and a single evaluation to find $c$ is sufficient. On the other hand, since $g(\mathbf{x})$ is a density it has to be properly normalized, so that we actually have performed the integration analytically up to a factor! Ideal importance sampling is no importance sampling. In practice, we search for a $g$ that 'looks like' $f$, Note that where $f(\mathbf{x}) < 0$ the density $g$ can never look good since it must be positive. In the usual practice of particle phenomenology, $f(\mathbf{x})$ is nonnegative but exhibits strong peaks in certain subspaces of phase space; this is the typical situation that these lecture noes have in mind. Note that $f(\mathbf{x}) = 0$ while $g(\mathbf{x}) > 0$ is perfectly allowed[72].

In order to do importance sampling, we need an algorithm to generate pseudorandom points $\mathbf{x}$ not uniformly but with a prescribed, non-uniform density $g(\mathbf{x})$. The field of constructing such algorithms is vast, and will be the subject of section 9.

### 8.2.2 Multichanneling

Not all densities can be easily and efficiently generated, as we shall see[73]. But consider the following situation [22]: suppose that the density $g(\mathbf{x})$ can be written as a linear combination of (sub)densities $g_j(\mathbf{x})$:

$$g(\mathbf{x}) = \sum_{j=1}^{m} \alpha_j \, g_j(\mathbf{x}) \quad , \quad \sum_{j=1}^{m} \alpha_j = 1 \quad , \tag{171}$$

where each of the $g_j(\mathbf{x})$ *can* be generated easily or efficiently. In a continuous variation on this, we may write

$$g(\mathbf{x}) = \int dj \; \alpha(j) \; g(j,\mathbf{x}) \quad , \quad \int dj \; \alpha(j) = 1 \quad . \tag{172}$$

The subdensities are called *channels*. The total $g$ can then be generated by first picking a channel $j$ with probability $\alpha_j$ (or $\alpha(j)$), and then generating $\mathbf{x}$

[72]Although if this happens too often the integration becomes inefficient.

[73]For instance, the different peaking structures in $f(\mathbf{x})$ are perhaps best described by *different* sets of variables. This happens very often in particle phenomenology.

from $g_j(\mathbf{x})$ (or from $g(j,\mathbf{x})$). We shall restrict ourselves to the discrete case. Note that during the integration we have to compute the full $g(\mathbf{x})$ every time, so also those $g_k(\mathbf{x})$ must be computed whose channel $k$ was not picked!

Importance sampling is useful if we can make the quantity

$$W = J_2 = \int\limits_{\Gamma} d\mathbf{x}\, g(\mathbf{x})\; w(\mathbf{x})^2 = \int\limits_{\Gamma} d\mathbf{x}\, \frac{f(\mathbf{x})^2}{g(\mathbf{x})} \tag{173}$$

small. Using Lagrange multipliers to impose the restriction on the $\alpha_j$, the minimum is given by the conditions

$$\frac{\partial}{\partial \alpha_n}\left( W + \lambda \sum_j \alpha_j \right) = 0 \quad , \quad n = 1, \ldots, m \;\; , \tag{174}$$

which boils down to the condition

$$W_n = \int\limits_{\Gamma} d\mathbf{x}\, g_n(\mathbf{x})\; w(\mathbf{x})^2 = W \quad , \quad n = 1, \ldots, m \;\; : \tag{175}$$

each channel 'contributes the same amount' to the total error. We may try to achieve this goal in an adaptive way, where we first start by initialising the $\alpha$'s (if we have no information whatsoever, we might take them all equal). We then perform Monte Carlo integration until the $W_n$ have been reasonably determined[74]. Next, we update the $\alpha$'s, for instance by the rule

$$\alpha_n \quad \to \quad \alpha'_n = \frac{\beta_n}{\sum\limits_j \beta_j} \quad , \quad \beta_n = \alpha_n \left(\frac{W_n}{W}\right)^r \quad , \tag{176}$$

where $r$ is a number that regulates the eagerness of the algorithm, in order to avoid overshooting. This updating rule has at least the virtue that $W_n = W$ is a fixed point, but other strategies are imaginable[75].

ℵ Adaptive strategies for this problem have not been studied very much. There is certainly room for improvement!

[74]Note that at every integration point we have to calculate all the $g_n(\mathbf{x})$ anyway, so this is no computational burden.

[75]It feels a bit counter-intuitive that if a channel contributes al lot to $W$ we would need to *increase* its $\alpha$. But if we give more integration points to that channel, its error will of course *decrease.*

# 9 Non-uniform PRNGs

Often (and especially in the case of importance sampling) we are required to generate *non*-uniform pseudorandom numbers, and for many given densities a great number of strategies and tricks have been developed.

## 9.1 The Art of Transforming, Rejecting, and Being Smart

In generating nonuniform random numbers one invariably starts out with a source of iid pseudorandom numbers uniform in $(0, 1)$[76]. These are then subjected to transformations, decisions and combinations. These can be broadly classified under

- Inversions (mappings) where a function of a variate is computed;
- Rejections where variates are kept or rejected according to some criterium;
- Cleverly combining several variates into new ones;
- The building of a repertoire of tricks as a basis for new tricks;

[76]Some care has to be taken here since quite often the special values 0 or 1 can give rise to numerical problems. We shall assume that they are never generated by your favourite PRNG.

- Random-walk methods that employ ergodicity.

We shall discuss examples of all these. *The* reference text is the book by Devroye [25].

## 9.2 The UA formalism

### 9.2.1 Unitary algorithms as words and as pseudocode

Experience shows that, when we perform all kind of manipulations on random variates, it quickly becomes unclear what is precisely the density of the resulting numbers. Here it becomes useful to adhere to the *Unitary Algorithm* (UA) formalism. This is actually nothing more than integration statements with the number 1 on the left-hand side of the equation. These statements can be put in words as well, and describe immediately (pseudo)code for software implementation of algorithms. The simplest example is

$$1 = \int\limits_0^1 d\rho \quad . \tag{177}$$

In words this reads 'there exists an algorithm for generating numbers $\rho$ uniformly between 0 and 1'. And this is true, since we assume that we have a good PRNG at hand. in pseudocode, the UA statement reads

```
$\rho \leftarrow$ prng
```

which just means ‘get a number out of your PRNG’. The second ingredient of the UA formalism is the multiplication by unity in the form of saturated Dirac deltas. For instance we may extend Eq.(177) as follows:

$$\begin{aligned} 1 &= \int_0^1 d\rho \;\; \int dx\; \delta(x-\rho^2) \\ &= \int_0^1 d\rho\; \int dx\; \frac{1}{2\rho}\,\delta(\rho-\sqrt{x}) \\ &= \int dx\; \frac{1}{2\sqrt{x}}\,\theta(0<\sqrt{x}<1) \\ &= \int_0^1 dx\; \frac{1}{2\sqrt{x}} \quad . \end{aligned} \tag{178}$$

In words, we now have the statement ‘there is an algorithm for generating the density $1/(2x^{1/2})$ between 0 and 1’, and indeed there is: in pseudocode it reads

```
$\rho \leftarrow$ **prng**
$x \leftarrow \rho^2$
```

In the UA description, the 1 on the left is to ensure that we are dealing with properly normalised densities. For instance, $\theta(0 < x < 1/2)$ is not a density, but $2\theta(0 < x < 1/2)$ is. Furthermore, in the step $\int dx\; \delta(x-\rho^2)$

the boundaries on $x$ are $\pm\infty$ so that there *always* is some $x$ for any $\rho$: this ensures that the algorithm actually finishes and yields a result. The final boundaries 0 and 1 on $x$ arise from the fact that when we eliminate $\rho$ from the Dirac delta we must make sure that $\rho$ actually runs from 0 to 1.

### 9.2.2 Inversion of variates in UA

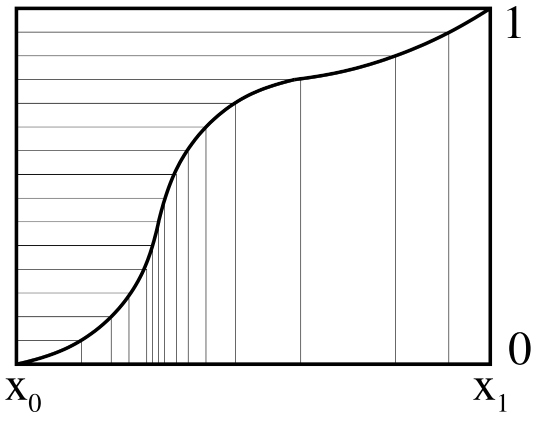
One of the basic tools of the trade is the following. Suppose that we want to generate $x$ according to some (properly normalised) density $g(x)$, between $x_0$ and $x_1$. Suppose, furthermore, that we can find the primitive of $g(x)$:

$$G(x) = \int\limits_{x_0}^{x} dy\; g(y) \quad \to \quad G(x_0) = 0\;,\; G(x_1) = 1\;\;. \tag{179}$$

And suppose, *in addition*, that we are able to invert $G$ somehow, so that $G^{-1}(z)$ can be computed. We then put $x$ equal to $G^{-1}(\rho)$, in UA speak:

$$\begin{aligned} 1 &= \int\limits_0^1 d\rho \int dx\; \delta(x - G^{-1}(\rho)) \\ &= \int dx \int\limits_0^1 d\rho\; \delta(\rho - G(x))\; \frac{1}{(G^{-1})'(\rho)} \\ &= \int dx\; \frac{1}{(G^{-1})'(G(x))}\; \theta(0 < G(x) < 1) \end{aligned}$$

$$= \int_{x_0}^{x_1} dx\ g(x) \qquad (180)$$



An illustration of the transformation method. We plot a function $G(x)$ and intersect it with regularly spaced horizontal lines. The resulting $x$ values are on the lower axis. Obviously, the *density* of $x$ values must be proportional to the *slope* of $G(x)$.

This method has the advantage that it uses only one **prng** per call, and it is very elegant. On the other hand, computing $G^{-1}$ may not be possible analytically, or be very time-consuming; moreover it is really suited only to one-dimensional densities so that for more-dimensional distributions one has to be able to successively integrate the density over its variables, and then also be able to invert the primitives. This can quickly become unfeasible. But for often-occurring, simple distributions this method works like a dream. Some examples:

**Algorithm 5** Generating the exponential distribution $\exp(-x)$ on $[0, \infty)$

```
$\rho \leftarrow$ **prng**
$x \leftarrow -\log(\rho)$
```

**Algorithm 6** Generating the Cauchy distribution $\sim (x^2+1)^{-1}$ on $(-\infty, \infty)$

```
$\rho \leftarrow$ **prng**
$x \leftarrow \tan(\pi(\rho - 1/2))$
```

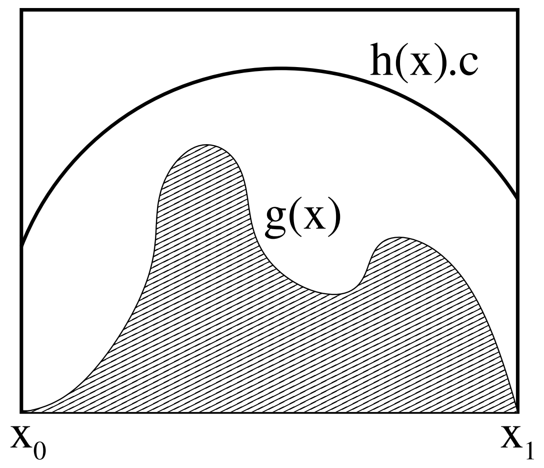
### 9.2.3 Rejection of variates in UA

This method works for more-dimensional densities as well as for one-dimensional ones. We want to generate a difficult target density $g(\mathbf{x})$ in some region $\Gamma$. 'Difficult' here means that we cannot use inversion, in fact we may not even know the normalisation of $g(\mathbf{x})$. Suppose that we *are* able to generate $\mathbf{x}$ according to *another* density $h(\mathbf{x})$, and that $g(\mathbf{x}) < c\,h(\mathbf{x})$ in $\Gamma$ for some known number[77] $c$. We then use the following algorithm, in pseudocode:

[77]Larger than 1 if both $g$ and $h$ are properly normalised.

**Algorithm 7** Rejection method to generate $g(x)$ by rejecting from $h(x)$

```
loop
  generate $\mathbf{x}$ according to $h(\mathbf{x})$
  $\rho \leftarrow$ prng
  if $\rho < g(\mathbf{x})/(c\,h(\mathbf{x}))$ then
    return {The value $\mathbf{x}$ is accepted}
  end if
end loop
```



This illustrates the rejection method in one dimension. We want to fill the histogram of $g(\mathbf{x})$, the shaded area, uniformly. The algorithm relies on filling the area below $c\,h(\mathbf{x})$ uniformly, and cutting away, on a probabilistic basis, the points above the shaded area. That is, points are accepted with a probability $g(\mathbf{x})/(c\,h(\mathbf{x}))$.

We can analyse this algorithm the UA way as well, in spite of the fact that we may have to go through the loop an unbounded number of times.

$$
\begin{aligned}
1 &= \int\limits_{\Gamma} d\mathbf{x}\; P(\mathbf{x}) \;\;, \\
P(\mathbf{x}) &= \int\limits_{\Gamma} d\mathbf{y}_1\, h(\mathbf{y}_1) \int\limits_0^1 d\rho_1 \left[\theta\left(\rho_1 < \frac{g(\mathbf{y}_1)}{c\, h(\mathbf{y}_1)}\right) \delta(\mathbf{x}-\mathbf{y}_1)\right. \\
&\quad +\theta\left(\rho_1 > \frac{g(\mathbf{y}_1)}{c\, h(\mathbf{y}_1)}\right) \int\limits_{\Gamma} d\mathbf{y}_2\, h(\mathbf{y}_2) \int\limits_0^1 d\rho_2 \;\Big[ \\
&\quad \theta\left(\rho_2 < \frac{g(\mathbf{y}_2)}{c\, h(\mathbf{y}_2)}\right) \delta(\mathbf{x}-\mathbf{y}_2) + \\
&\quad \left.\theta\left(\rho_2 > \frac{g(\mathbf{y}_2)}{c\, h(\mathbf{y}_2)}\right) \int\limits_{\Gamma} d\mathbf{y}_3\, h(\mathbf{y}_3) \int\limits_0^1 d\rho_3 \cdots \Big]\right]
\end{aligned} \tag{181}
$$

The expression for $P(\mathbf{x})$ is potentially infinite, but we can *telescope* it[78]:

$$
\begin{aligned}
P(\mathbf{x}) &= \int\limits_{\Gamma} d\mathbf{y}_1\, h(\mathbf{y}_1) \int\limits_0^1 d\rho_1 \left[\theta\left(\rho_1 < \frac{g(\mathbf{y}_1)}{c\, h(\mathbf{y}_1)}\right) \delta(\mathbf{x}-\mathbf{y}_1)\right. \\
&\quad \left. + \,\theta\left(\rho_1 > \frac{g(\mathbf{y}_1)}{c\, h(\mathbf{y}_1)}\right) P(\mathbf{x})\right]
\end{aligned}
$$

[78]Remember the continued fractions?

$$
\begin{aligned}
&= \int\limits_{\Gamma} d\mathbf{y}_1\, h(\mathbf{y}_1)\,\left[\frac{g(\mathbf{y}_1)}{c\,h(\mathbf{y}_1)}\delta(\mathbf{x}-\mathbf{y}_1) + \left(1 - \frac{g(\mathbf{y}_1)}{c\,h(\mathbf{y}_1)}\right) P(\mathbf{x})\right] \\
&= \frac{g(\mathbf{x})}{c} + P(\mathbf{x}) - P(\mathbf{x}) \int\limits_{\Gamma} d\mathbf{y}_1\, \frac{g(\mathbf{y}_1)}{c} \quad . \qquad (182)
\end{aligned}
$$

And we finally arrive at

$$
P(\mathbf{x}) = g(\mathbf{x}) \left( \int\limits_{\Gamma} d\mathbf{y}\; g(\mathbf{y}) \right)^{-1} \quad , \qquad (183)
$$

so that the rejection algorithm is *self-normalising*. Also note that it is really necessary that $g(\mathbf{x}) < c\,h(\mathbf{x})$ everywhere. The rejection algorithm is conceptually very simple but also has its drawbacks. It uses at least two calls to **prng** to obain a single variate. It is common to use a uniform distribution for $h(\mathbf{x})$ but that will not work if $\Gamma$ is infinitely large. Finding a good value for $c$ may not be easy, or impossible if $g(\mathbf{x})$ goes to infinity somewhere. Of course the algorithm will work for any sufficiently large $c$ but its efficiency will become low if $c$ is very large. Rejection is typically used if we have *almost* the distribution we want, and has only to be massaged a bit. The canonical reference to the method is to von Neumann [26], but surely the method must be older.

## 9.3 Repertoire and the Rule of Nifty

An important rôle in the generation of nonuniform variates is played by the buildup of a repertoire of algorithms, and by application of the following Rule of Nifty: *a clever way of constructively computing the normalisation of a density will usually lead to an efficient algorithm for generating the density.*

### 9.3.1 Building up a repertoire

It pays to keep in mind tricks that work. We give an example here. Suppose that we have an algorithm for generating $g(x)\theta(x > 0)$. Then, multiplying with a **prng** results in

$$
\begin{aligned}
1 &= \int_0^\infty dy\; g(y) \;\int_0^1 d\rho \;\int dx\; \delta(x - \rho y) \\
&= \int dx \;\int_0^\infty dy\, \frac{g(y)}{y} \;\int_0^1 d\rho\, \delta\left(\rho - \frac{x}{y}\right) \\
&= \int dx \;\int_0^\infty dy\, \frac{g(y)}{y}\, \theta(x < y) \\
&= \int_0^\infty dx\; P(x) \quad , \quad P(x) = \int_x^\infty dy\, g(y)/y \quad .
\end{aligned}
\tag{184}
$$

By repeated applications, we can arrive at surprisingly simple algorithms for surprisingly ugly densities, as algorithms 8 to 11 show. The functions $E_1(x)$ and $K_0(x)$ are discussed in [27]. You are not very likely to ever have to generate these densities, but it is nice to know how to do it even so.

**Algorithm 8** Generating $\log(1/x)^k/k!$ over $(0,1])$

```
  $\rho_{1,2,\ldots,k+1} \leftarrow$ **prng**
  $x \leftarrow \rho_1\rho_2\cdots\rho_k\rho_{k+1}$
```

**Algorithm 9** Generating $x^k\exp(-x)/k!$ over $[0,\infty)$

```
  $\rho_{1,2,\ldots,k+1} \leftarrow$ **prng**
  $x \leftarrow -\log(\rho_1\rho_2\cdots\rho_k\rho_{k+1})$
```

**Algorithm 10** Generating the exponential-integral density $E_1(x)$ over $[0,\infty)$

```
  $\rho_{1,2} \leftarrow$ **prng**
  $x \leftarrow -\rho_1 \log(\rho_2)$
```

**Algorithm 11** Generating the Bessel density $2K_0(2\sqrt{x})$ over $[0,\infty)$

```
$\rho_{1,2} \leftarrow$ **prng**
$x \leftarrow \log(\rho_1)\log(\rho_2)$
```

### 9.3.2 The normal distribution: the Box-Müller algorithm

As an example of the Rule of Nifty we can consider the Guassian, or normal density:

$$N(x) = \frac{1}{K}\exp(-x^2) \quad , \quad K = \int\limits_{-\infty}^{\infty} dx\;\exp(-x^2) \quad . \tag{185}$$

Computing $K$ by integrating the density in a straightforward way involves the error function, so generating normal variates by transformation calls for the i*nverse error function*, which is horrible. Straightforward rejection from a uniform distribution is also impossible on the interval $(-\infty,\infty)$. However, there exists the 'doubling trick', where polar coordinates are used:

$$\begin{aligned} K^2 &= \int dx\,dy\;\exp(-x^2-y^2) \\ &= \int\limits_0^{2\pi} d\phi \int\limits_{-}^{\infty} dr\;r\;\exp(-r^2) = \pi\int\limits_0^{\infty} ds\;\exp(-s) = \pi \quad . \end{aligned} \tag{186}$$

We already know how to generate the density $\exp(-s)$ (see sect.9.2.2). This method is known as the Box-Müller algorithm [28]. There are faster methods, but (to my mind) none so elegant.

**Algorithm 12** The Box-Müller algorithm

```
$\rho_{1,2} \leftarrow$ **prng**
$r^2 \leftarrow -\log(\rho_1)$
$\phi \leftarrow 2\pi\rho_2$
$x \leftarrow r\cos(\phi)$
$y \leftarrow r\sin(\phi)$ {$x$ and $y$ are independent normal variates}
```

### 9.3.3 The Euler algorithm

Quite often we are asked to generate numbers satisfying some constraint. An example of the Rule of Nifty is the *generalised Euler distribution*, the $n$-dimensional probability $P(x_1, x_2, \ldots, x_n)$ given by

$$P(x_1, x_2, \ldots, x_n) \;\sim\; {x_1}^{p_1}\, {x_2}^{p_2} \cdots {x_n}^{p_n}\; \delta\left(\sum_{j=1}^{n} x_j - 1\right) \quad , \tag{187}$$

where $p_j \geq 0$ are integers. The normalisation factor is here unknown: we must also compute it. It might be tempting to generate the $x_j$ in order, with $x_j$ between 0 and $1 - x_1 - \cdots - x_{j-1}$, but we can see straightaway that, first, the distribution of $x_{1,2,\ldots,n-1}$ is independent of $n$ (which is surely wrong) and, second, the value of $x_n$ is completely fixed by $x_{1,2,\ldots,n-1}$ so that its density cannot play a rôle. The better algorithm is given here, further on we shall

**Algorithm 13** The Euler density with parameters $p_1, p_2, \ldots, p_n$

```
generate $y_j$ in $(0,\infty)$ according to $\sim\ y^{p_j}\ \exp(-y)$ for $j = 1, 2, \ldots, n$
$s \leftarrow y_1 + y_2 + \cdots y_n$
$x_j \leftarrow y_j/s$ for $j = 1, 2, \ldots, n$
```

discuss its various aspects.

Let us write this out in the UA formalism. It reads

$$\begin{aligned}
1 \quad = \quad & \frac{1}{p_1!p_2!\cdots p_n!} \int\limits_0^\infty dy_1 \cdots dy_n \; {y_1}^{p_1} e^{-y_1} \cdots {y_n}^{p_n} e^{-y_n} \\
& \int ds\; \delta(y_1 + \cdots + y_n - s) \\
& \int dx_1 \cdots dx_n \; \delta\left(x_1 - \frac{y_1}{s}\right) \cdots \delta\left(x_n - \frac{y_n}{s}\right) \quad . \qquad (188)
\end{aligned}$$

First, we eliminate the $y$'s in favor of the $x$'s, taking care to correctly handle the factors of $s$ coming from the Dirac deltas, and then we do the integral over $s$:

$$1 \quad = \quad \frac{1}{p_1!p_2!\cdots p_n!} \int\limits_0^\infty dx_1 \cdots dx_n \; {x_1}^{p_1} \cdots {x_n}^{p_n} \;\; \exp(-s(x_1 + \cdots + x_n))$$

$$
\begin{aligned}
&\int\limits_0^\infty ds\; s^{p_1+\cdots+p_n+n}\;\delta(s(x_1+\cdots+x_n)s-s)\\
&= \quad \frac{1}{p_1!p_2!\cdots p_n!}\int\limits_0^\infty dx_1\cdots dx_n\; {x_1}^{p_1}\cdots {x_n}^{p_n}\;\delta(x_1+\cdots+x_n-1)\\
&\int\limits_0^\infty ds\; s^{p_1+p_n+n-1}\,e^{-s}\\
&= \quad \frac{\Gamma(p_1+\cdots+p_n+n)}{\Gamma(p_1+1)\cdots\Gamma(p_n+1)}\int\limits_0^\infty dx_1\cdots dx_n\; {x_1}^{p_1}\cdots {x_n}^{p_n}\;\delta\left(\sum_j x_j-1\right)\quad .
\end{aligned}
\tag{189}
$$

We see that the density is precisely correct, and we obtain the normalisation into the bargain. The 'original' distribution of the $y$'s contains a factor $\exp(-y)$. Such a damping factor is necessary if we want to allow for arbitrarily large $y$ values: these *must* be allowed since $(x_1, x_2, \ldots, x_n) = (1, 0, \ldots, 0)$ must be possible. The damping factor might also have been $\exp(-y^2)$, say, but our choice is seen to be the better one since it leads to a uniform sampling.

### 9.3.4 The Kinderman-Monahan algorithm

Let us generate two variates $v, u$ iid uniformly, $0 < u < 1$ and $-1 < v < 1$, and consider their ratio:

$$\begin{aligned} 1 &= \frac{1}{2}\int\limits_0^1 du \int\limits_{-1}^1 dv \int dx\ \delta\left(x - \frac{v}{u}\right) \\ &= \frac{1}{2}\int dx \int\limits_0^1 du\, u \int\limits_{-1}^1 \delta(v - xu) \\ &= \frac{1}{4}\int dx \int\limits_0^1 d(u^2)\ \theta(x^2u^2 < 1) \quad . \end{aligned} \tag{190}$$

The step functions can be translated as

$$\begin{aligned} &\theta(x^2u^2 < 1)\theta(0 < u < 1) = \\ &\qquad \theta(|x| < 1)\theta(0 < u < 1) + \theta(|x| > 1)\theta(0 < u < 1/|x|) \quad , \end{aligned} \tag{191}$$

and so we get

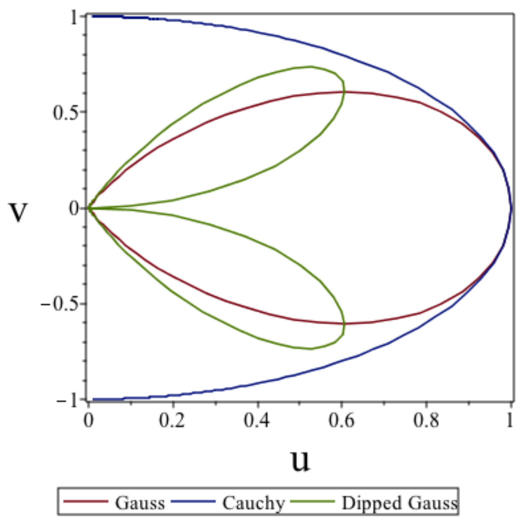
$$1 = \int dx\ h(x) \quad , \quad h(x) = \frac{1}{4}\left[\theta(|x| < 1) + \frac{1}{x^2}\theta(|x| > 1)\right] \quad . \tag{192}$$

Thus we have a method to generate a density $h(x)$ that has a bump around zero and tails that fall off as $1/x^2$. We may use this one to generate other distributions by rejection[79]: The Kinderman-Monahan, or *ratio of uniforms* algorithm[29] . Let us generate $u$ and $v$ and keep only points that fall inside a certain region defined by a function $f$:

$$\begin{aligned}
&\int\limits_0^1 du \int\limits_{-1}^1 dv\; \theta\left(u < \sqrt{f(v/u)}\right)\; \int dx\; \delta(x - v/u) \\
&= \int dx \int\limits_0^1 du\, u\; \theta(u < \sqrt{f(x)})\; \theta(-1 < ux < 1) \\
&\sim \int dx\; f(x) \quad , \qquad (193)
\end{aligned}$$

and this proves that $x$ is generated proportional to $f(x)$. Note that we need $f(x) < 1$ as well as the fact that $\sqrt{f(x)} < 1/|x|$ in order for the last step function to be irrelevant. The Kinderman-Monahan algorithm essentially asks only for the determination that the points $(u, v)$ are inside a certain region, and this can be made quite efficient. The region of acceptance is

[79]Since rejection is self-normalising we can afford to be somewhat sloppy with the overall factors.



bounded by a curve $(u(\tau), v(\tau))$ where $v(\tau) = \tau\, u(\tau)$, $u(\tau) = \sqrt{f(\tau)}$. Here we give three examples: the Gauss density $\sim \exp(-x^2)$, the Cauchy density $\sim 1/(1+x^2)$, and a 'dipped Gauss' density $\sim x^2 \exp(-x^2)$. We may relax the constraints $0 < u, |v| < 1$, and simply look for the maxima of $u(\tau)$ and $|v(\tau)|$ in order to decide on the rectangle in which to sample the points uniformly. For the Cauchy distribution, the region of acceptance is a half circle, and this gives us an even simpler method of generating it, which avoids taking an tangent at the cost of more random numbers.

## 9.4 Random-walk algorithms

The algorithms discussed so far return iid variates. There is another strategy that is widely used, where new variates are determined from the previous ones. In the space of events the algorithm therefore performs a 'random walk' of which one must ensure that (a) the points visited have the desired density, also called the target density, (b) eventually the whole space is visited

**Algorithm 14** Generating the Cauchy density by ratio of uniforms

```
loop
    $\rho_{1,2} \leftarrow$ **prng**
    $r \leftarrow 2\rho_1 - 1$
    if $r^2 + \rho_2^2 < 1$ then
        return $r/\rho_2$
    end if
end loop
```

(that is, ergodicity holds), and (c) that the subsequent points can be argued to be *essentially* independent. Such algorithms also go under the name of Markov-Chain Monte Carlo (MCMC).

### 9.4.1 The Metropolis algorithm

This is nowadays also called the Metropolis-Hastings method [30]. The algorithm is actually quite simple. Suppose we are at a point $\mathbf{x}_n$. Then, using *some* prescription, we generate a *candidate* new point $\mathbf{y}$ using a probability density $g(\mathbf{x}_n; \mathbf{y})$. We then compare the target densities $P(\mathbf{x})$ and $P(\mathbf{y})$:

$$R = \frac{P(\mathbf{y})}{P(\mathbf{x})} \quad . \tag{194}$$

If $R > 1$ then we accept the candidate point $\mathbf{x}_{n+1} = \mathbf{y}$; this is reasonable since the candidate point is more probable than the old one. However, if

$R < 1$ we *may* accept $\mathbf{x}_{n+1} = \mathbf{y}$, with probability $R$. If not, then we stick to the old point, and have $\mathbf{x}_{n+1} = \mathbf{x}_n$. The fact that we allow for a reduction in probability is what allows us to wander all over the space; otherwise we would simple be working our way towards a point of (locally) maximal probability. This method will work *provided* that we have *detailed balance*: we insist that

$$g(\mathbf{x};\mathbf{y}) = g(\mathbf{y};\mathbf{x}) \ . \tag{195}$$

The convergence of the Metropolis algorithm can best be pictured as follows. Suppose our space is populated by many random walkers, that at the start are distributed with some density $p_1(\mathbf{x})$. These walkers then each take one Metropolis step; some of them will remain where they are, others will be displaced. Their density after this first step is then $p_2(\mathbf{x})$, presumably a different one. This continues, so that as the algorithm proceeds we have a succession of densities $p_1(\mathbf{x}), p_2(\mathbf{x}), p_3(\mathbf{x}), \ldots$. The idea is that $\lim_{n\to\infty} p_n(\mathbf{x}) = P(\mathbf{x})$. The UA formulation is

$$\begin{aligned}
1 &= \int d\mathbf{x}\, p_n(\mathbf{x}) \ , \\
p_n(\mathbf{x}) &= \int d\mathbf{z}\, p_{n-1}(\mathbf{z}) \int d\mathbf{y}\ g(\mathbf{z};\mathbf{y}) \\
&\quad \times \Big[ \ \theta(P(\mathbf{y}) > P(\mathbf{z}))\, \delta(\mathbf{x}-\mathbf{y}) \\
&\qquad + \theta(P(\mathbf{y}) < P(\mathbf{z}))\, \frac{P(\mathbf{y})}{P(\mathbf{z})}\, \delta(\mathbf{x}-\mathbf{y})
\end{aligned}$$

$$+ \theta(P(\mathbf{y}) < P(\mathbf{z})) \left(1 - \frac{P(\mathbf{y})}{P(\mathbf{z})}\right) \delta(\mathbf{x} - \mathbf{z})\Bigg] \quad . \qquad (196)$$

Let us now suppose that at some moment the walkers are actually distributed according to the target density: $p_{n-1}(\mathbf{x}) = P(\mathbf{x})$. After stepping, their density becomes

$$\begin{aligned} p_n(\mathbf{x}) &= \int d\mathbf{z}\; P(\mathbf{z})\, g(\mathbf{z};\mathbf{x})\; \theta(P(\mathbf{x}) > P(\mathbf{z})) \\ &+ \int d\mathbf{z}\; P(\mathbf{x})\, g(\mathbf{z};\mathbf{x})\; \theta(P(\mathbf{x}) < P(\mathbf{z})) \\ &+ P(\mathbf{x}) \int d\mathbf{y}\; g(\mathbf{x};\mathbf{y})\; \theta(P(\mathbf{y}) < P(\mathbf{x})) \\ &- \int d\mathbf{y}\; g(\mathbf{x};\mathbf{y})\, P(\mathbf{y})\; \theta(P(\mathbf{x}) > P(\mathbf{y})) \quad . \qquad (197) \end{aligned}$$

Under detailed balance and renaming the integration variables, the first and fourth lines cancel and we are left with

$$\begin{aligned} p_n(\mathbf{x}) &= P(\mathbf{x}) \int d\mathbf{z}\; g(\mathbf{x};\mathbf{z}) \; \Big[\theta(P(\mathbf{x}) < P(\mathbf{z})) + \theta(P(\mathbf{z}) < P(\mathbf{x}))\Big] \\ &= P(\mathbf{x}) \int d\mathbf{z}\; g(\mathbf{x};\mathbf{z}) \;=\; P(\mathbf{x}) \quad . \qquad (198) \end{aligned}$$

We see that the target density $P(\mathbf{x})$ is a *fixed point* of the Metropolis algorithm, so that we can feel more or less confident that we shall approach our goal. Note the essential rôle played by the detailed balance requirement! For the rest we are completely free in the choice of $g$: we may change it at will at any moment. The choice of $g$ *does* influence the performance of the algorithm, though. If we take 'small' steps the probability of accepting a candidate point is probably high, but it will take a (very) long time to cover the space. If the steps are 'large' then we move over the space quickly, but we may expect that not many of such proposed steps are accepted[80]. A good rule of thumb seems to be that about half of the candidates should be accepted.

A drawback is the fact that the various points are *not* independent. This is typically overcome (hopefully) by taking a number of Metropolis steps before claiming the new point. How many steps are necessary and sufficient depends on the case: here, as everywhere, Monte Carlo is partly an art.

### 9.4.2 An elementary case study for Metropolis

We can completely analyse the Metropolis algorithm in a very simple case. This is the generation of the density $P(x) = 2x\,\theta(0 < x < 1)$. We shall use the step recipe $g(x;y) = \theta(0 < x, y < 1)$ which certainly gives detailed

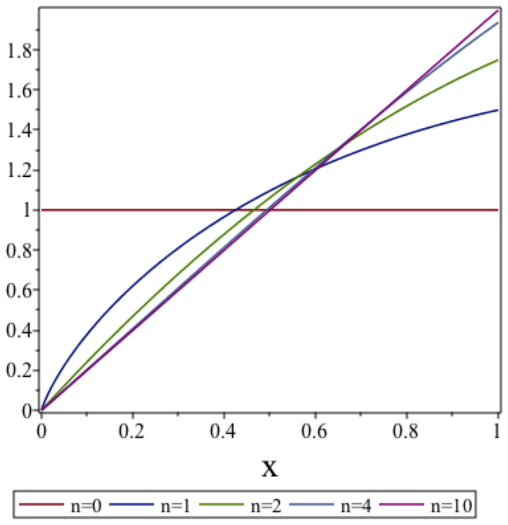
[80]Especially if there are several probability maxima the chance of moving from the neighbourhood of one maximum to that of another can be very small.

balance. The Metropolis step of Eq.(196) then takes the form

$$p_n(x) = \frac{x}{2}\,p_{n-1}(x) + \int\limits_0^x dx\; p_{n-1}(z) + x\int\limits_x^1 dz\;\frac{p_{n-1}(z)}{z} \quad . \tag{199}$$

We start with $p_0(x) = \theta(0 < x < 1)$. The subsequent densities are

$$\begin{aligned} p_1(x) &= \frac{3}{2}\,x - x\log(x) \quad , \\ p_n(x) &= \left(2 + \frac{1}{2^{n-1}(n-1)}\right)x - \frac{n+1}{2^n(n-1)}x^n \quad , \quad n \geq 2 \quad . \end{aligned} \tag{200}$$



We plot the shapes of $p_n(x)$ for $n = 0, 1, 2, 4$, and 10. The density $p_{10}(x)$ is essentially indistinguishable from the target density $P(x) = 2x$. As we see from Eq.(200), the approach to $P$ is exponentially fast. This way of generating $P(x)$ is of course much clumsier than the simple inversion rule $x \leftarrow \sqrt{\mathbf{prng}}$ but it avoids taking a square root. Two-thirds of the candidates are accepted in this case.

An interesting observation is the following. Suppose that you would need to generate the distribution $x(3 - \log(x))/2$. Inversion seems pretty hopeless in this case, and rejection from a uniform density demands the computation of the logarithm. We see that an alternative method is to generate a random variable uniformly in $(0, 1)$ and then perform precisely *one* step of the Metropolis algorithm!

### 9.4.3 Applications of the Metropolis algorithm

The Metropolis algorithm finds a very natural (and, in fact, its original) application in statistical mechanics. For a system in the canonical ensemble its microconfigurations $\mathbf{X}$ are distributed with a density

$$P(\mathbf{X}) \sim \exp\left(-\frac{1}{kT}\,H(\mathbf{X})\right) \quad , \tag{201}$$

where $k$ is Boltzmann's constant, $T$ is the themperature, and $H(\mathbf{X})$ the Hamiltonian function of the system. Thermodynamic quantities are computed as averages over a sample of microconfigurations with this density. Calculating $H$ is usually very cumbersome by itself, but in the Metropolis algorithm it is not necessary since what we are really interested in is the *ratio* of densities:

$$P(\mathbf{Y})/P(\mathbf{X}) = \exp\left(\frac{1}{kT}\left(H(\mathbf{X}) - H(\mathbf{Y})\right)\right) \quad . \tag{202}$$

We therefore only have to compute the *change* in the Hamiltonian when stepping from $\mathbf{X}$ to $\mathbf{Y}$. Especially for simple steps, the flipping of a single spin in an Ising system, say, this can be done very quickly[81].

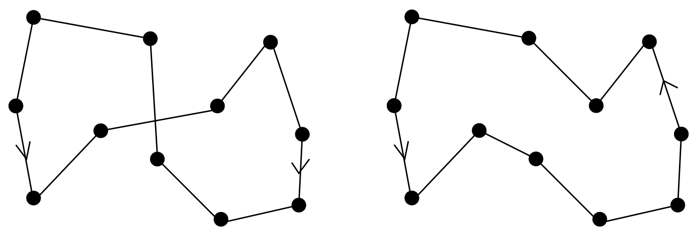
Another interesting application is that of *simulated annealing*, best exemplified by the Travelling Salesman problem: given a set of $N$ 'cities' with known distances between them, find the shortest route that visits all the cities. For simplicity we demand that we come back to the starting city, and the direction of travel is unimportant. This gives us $(N-1)!/2$ different routes, and a direct enumeration to find the optimal route becomes unpractical for $N > 13$ or so. What we can do instead is to assign a probability density to each route $R$:

$$P(R) \sim \exp\left(-\frac{1}{kT}\,L(R)\right) \quad , \tag{203}$$

where $kT$ is as before, and $L(R)$ is the total length of route $R$. The 'temperature' is of course a totally fictional one. Using the Metropolis algorithm we can sample this density; and by starting at high $T$ and then gradually and carefully lowering it towards zero we may hope to end up in a maximum of the probability, *i.e.* a very short (or even *the* shortest) route. Typical steps

[81]Flipping a single spin should not be the *only* possible step, however, especially close to an ordering transition. Occasionally, whole *blocks* of spins can be flipped as well. As we have already pointed out, choosing between different steps depending on the circumstances is perfectly allowed as long as detailed balance is maintained.

in this game are the interchange of two cities, and (somewhat surprisingly at first sight) the reversal of a whole sequence of cities. Because of the triangle inequality a route that crosses itself is always shortened by undoing the crossing, as indicated here. But that implies that the cities on on side of the crossing must be travelled in the *opposite* order!



### 9.4.4 Gibbs sampling

Another example of a random-walk method is Gibbs sampling, that aims at replacing the generation of a multidimensional density by a succession of one-dimensional ones. It is best described by an example. Suppose that we want to generate a three-dimensional density $P(\mathbf{x}) = P(x^1, x^2, x^3)$. This may not be possible directly, but suppose that we *can* generate $x^1$ for fixed $x^{2,3}$, and $x^2$ for fixed $x^{1,3}$, and also $x^3$ for $x^{1,2}$ fixed. Since these are one-dimensional densities, that is usually easier. Note that we have to normalize properly: for the generation of $x^j$ the appropriate density is not $P(\mathbf{x})$ itself but $P_j$ where

(in the $n$-dimensional case)

$$P_j(x^1,\ldots,x^{j-1},x^j,x^{j+1},\ldots,x^n) =$$
$$P(\mathbf{x})\ \left[\int d\xi\ P(x^1,\ldots,x^{j-1},\xi,x^{j+1},\ldots,x^n)\right]^{-1}\quad . \tag{204}$$

Now assume that at some step $n$ in the Gibbs algorithm we have generated a point $\mathbf{x}$. A Gibbs step to the new point $\mathbf{y}$ is then performed by first generating $y^1$ using $P(y^1,x^2,x^3)$, then generating $y^2$ according to $P_2(y^1,y^2,x^3)$, and finally generating $y^3$ from $P_3(y^1,y^2,y^3)$. At each step one new coordinate replaces an old one. Again considering a distribution of random walkers on our space, with density $p_n(\mathbf{x})$, we can then UA towards the density after one Gibbs step:

$$\begin{aligned}
1 &= \int d^3\mathbf{x}\ p_n(\mathbf{x}) \\
&= \int dx^1\, dx^2\, dx^3\ p_n(x^1,x^2,x^3) \\
&\quad \times\ dy^1\ P_1(y_1,x^2,x^3)\ dy^2\ P_2(y^1,y^2,x^3)\ dy^3\ P_3(y^1,y^2,y^3) \\
&= \int d\mathbf{y}\ P(\mathbf{y})\ R_n(\mathbf{y})\quad ,
\end{aligned} \tag{205}$$

where after reshuffling of factors we have

$$R_n(\mathbf{y}) = \int dx^1\ \frac{p_n(x^1,x^2,x^3)}{\int d\xi\ P(\xi,x^2,x^3)}\ dx^2\ \frac{P_1(y^1,x^2,x^3)}{\int d\xi P(y_1,\xi,x^3)}\ dx^3\ \frac{P(y^1,x^2,x^3)}{\int d\xi\ P(y^1,y^2,\xi)}\quad . \tag{206}$$

The new density $p_{n+1}(\mathbf{y})$ is therefore equal to $P(\mathbf{y})R_n(\mathbf{y})$. Now, assume that at step $n$ the $\mathbf{x}$'s are actually distributed with the target distribution: then $p_n(\mathbf{x}) = P(\mathbf{x})$. It immediately follows that $R_n(\mathbf{y}) = 1$ and $p_{n+1}(\mathbf{y}) = P(\mathbf{y})$: we have thus proven that the target density is a fixed point of this algorithm.

### 9.4.5 An elementary case study for Gibbs

A very simple analysable case for Gibbs sampling is the two-dimensional density

$$P(\mathbf{x}) = x^1 + x^2 \quad . \tag{207}$$

If at step $n$ the density of walkers is given by $p_n(\mathbf{x})$ then after one Gibbs step we shall have (*cf* Eq.(205))

$$p_{n+1}(\mathbf{y}) = P(\mathbf{y}) \int dx^1\, dx^2\, \frac{p_n(\mathbf{x})\,(y^1 + x^2)}{(x^2 + 1/2)(y^1 + 1/2)} \tag{208}$$

Let us start with a uniform density of walkers: $p_1(\mathbf{x}) = \theta(0 < x^{1,2} < 1)$. After the first Gibbs step, Eq.(208) yields

$$p_1(\mathbf{x}) = P(\mathbf{x})\, \frac{2 - \tau + 2\tau\, x^1}{1 + 2x^1} \quad , \quad \tau = \log(3) \quad . \tag{209}$$

By direct computation we find that, in general

$$p_n(\mathbf{x}) = P(\mathbf{x})\, \frac{a_n + b_n x^1}{1 + 2x^1} \quad , \tag{210}$$

with a double recursion relation:

$$\begin{pmatrix} a_n \\ b_n \end{pmatrix} = \mathcal{M} \begin{pmatrix} a_{n-1} \\ b_{n-1} \end{pmatrix} \quad , \quad \begin{pmatrix} a_1 \\ b_1 \end{pmatrix} = \begin{pmatrix} 2-\tau \\ 2\tau \end{pmatrix} \quad , \tag{211}$$

with

$$\mathcal{M} = \begin{pmatrix} 1-\kappa & \kappa/2 \\ 2\kappa & 1-\kappa \end{pmatrix} \quad , \quad \kappa = \tau - \tau^2/2 = 0.49514\cdots \tag{212}$$

The matrix $\mathcal{M}$ has eigenvalues 1 and $1 - 2\kappa = 0.0097\cdots$, for eigenvectors $v_1 = (1, 2)$ and $v_2 = (1, -2)$, respectively. Starting with the vector $(2 - \tau, 2\tau) = v_1 + (1 - \tau)\, v_2$ and repeatedly applying $\mathcal{M}$, we shall suppress the $v_2$ component exponentially fast, and find that $\lim_{n\to\infty}(a_n, b_n) = (1, 2)$, in other words,

$$\lim_{n\to\infty} p_n(\mathbf{x}) = P(\mathbf{x}) \quad . \tag{213}$$

Since $1 - 2\kappa$ is so small, the approach to the target density is very fast in this case.

## 9.5 UA for the von Neumann exponential generator

In [26], John von Neumann describes an interesting algorithm for generating the density $\sim \exp(-x)$ between 0 and 1:

"The normal procedure is to pick $T$ from a uniform distribution on $(0, 1)$ and compute $X = -\log(1-T)$, but $[\cdots]$ it seems objectionable to compute a transcedental function of a random number $[\cdots]$ We select random numbers $Y$ from a uniform distribution on $(0,1)$ as follows. Pick $Y_1$ and $Y_2$. If $Y_1 \leq Y_2$, we stop. If $Y_1 > Y_2$, we select $Y_3$. If $Y_2 \leq Y_3$, we stop. Otherwise $Y_1 > Y_2 > Y_3$, and we select $Y_4$, etc. With probability one there will be a least $n$ for which

$$Y_1 > Y_2 > \cdots > Y_n, \text{ but } Y_n \leq Y_{n+1}.$$

Now is $n$ is odd, we accept $Y_1$ as $X$, but if $n$ is even we reject all the $Y_i$ $[\cdots]$"

It is interesting to see how this algorithm can be analysed using the unitary-algorithm formalism[82]. By 'rejecting all the $Y_i$' we must understand 'starting all over again', otherwise the algorithm could not possibly be unitary[83]. If $p(x)$ stands for the generated distribution, we can write the algorithm as

$$\begin{aligned} p(x) &= \int\limits_0^1 dt \left[ (1-t)\,\delta(x-t) + \int\limits_0^t d\rho\, A(\rho) \right] \;, \qquad (214) \\ A(\rho) &= (1-\rho)\,p(x) + \int\limits_0^\rho d\sigma\, B(\sigma) \;, \end{aligned}$$

[82]In [26] a proof is also given, which basically relies on enumerating all the possible cases. It is simple enough but does not lend itself easily to generalisations.

[83]Because it would mean simply giving up and not generating an $X$ value at all!

$$B(\rho) \;=\; (1-\rho)\,\delta(x-t) + \int\limits_0^\rho d\sigma\, A(\sigma) \;\; . \qquad (215)$$

This can easily be checked by writing out the first few interations. Note the difference between $A$ and $B$ that of course reflects the even/odd distinction above. Eq.(215) implies, for $A$ and $B$:

$$\begin{aligned} &A'(\rho) = B(\rho) - p(x) \quad , \quad B'(\rho) = A(\rho) - \delta(x-t) \;\; , \\ &A(0) = p(x) \quad , \quad B(0) = \delta(x-t) \;\; , \\ &A(1) = \int\limits_0^1 d\sigma\, B(\sigma) \quad , \quad B(1) = \int\limits_0^1 d\sigma\, A(\sigma) \;\; . \end{aligned} \qquad (216)$$

These linear equations are easily solved, and we find

$$\begin{aligned} A(\rho) &= \delta(x-t) \;+\; e^{-\rho}\left(p(x) - \delta(x-t)\right) \;\; , \\ B(\rho) &= p(x) \;+\; e^{-\rho}\left(p(x) - \delta(x-t)\right) \;\; . \end{aligned} \qquad (217)$$

We can then evaluate Eq.(214) by integrating the right-hand side:

$$p(x) = \left(p(x)/e + \exp(-x)\right)\,\theta(0<x<1) \;\; , \qquad (218)$$

in other words,

$$p(x) = \frac{1}{1-1/e}\, e^{-x}\, \theta(0 < x < 1) \quad , \tag{219}$$

as advertised. Another twist is possible: we may decide, after each rejection, to increase the finally accepted $x$ by unity. This means that the obtained density is not simply $p(x)$ but rather $p_0(x)$, where

$$\begin{aligned}
p_n(x) &= \int\limits_0^1 dt \left[ (1-t)\,\delta(t+n-x) + \int\limits_0^t d\rho\, A_n(\rho) \right] \quad , \\
A_n(\rho) &= (1-\rho)\, p_{n+1}(x) + \int\limits_0^\rho d\sigma\, B_n(\sigma) \quad , \\
B_n(\rho) &= (1-\rho)\, \delta(t+n-x) + \int\limits_0^\rho d\sigma\, A_n(\sigma) \quad .
\end{aligned} \tag{220}$$

By following exactly the same steps as above we can find

$$A_n(\rho) = \delta(t+n-x)\, \left(1 - e^{-\rho}\right) + p_{n+1}(x)\, e^{-\rho} \quad , \tag{221}$$

and so find a recursion relation:

$$p_n(x) = e^{n-x}\theta(n < x < n+1) + p_{n+1}(x)/e \quad . \tag{222}$$

We can now simply iterate this to obtain

$$\begin{aligned} p_0(x) &= e^{-x}\theta(0 < x < 1) + e^{-x}\theta(1 < x < 2) + p_2(x)/e^2 \\ &= e^{-x}\theta(0 < x < 1) + e^{-x}\theta(1 < x < 2) + e^{-x}\theta(2 < x < 3) + p_3(x)/e^3 \quad = \quad \cdots \\ &= \exp(-x)\,\theta(x > 0) \quad : \end{aligned} \tag{223}$$

and we have now found a way of generating the full exponential distribution using only comparisons between (pseudo)random numbers. Von Neumann ends on a sobering note: "$\cdots$ It is a sad fact of life, however, that under the particular conditions of the ENIAC it was slightly quicker to use a truncated power series for $\log(1-T)$ than to carry out all the discriminations $\cdots$"

ℵ The above analysis is really meant as an illustration rather than as a serious algorithm proposal. We should keep in mind that this way of generating exponential variates relies much more on the properties of the PRNG since it is required to be well-behaved in more dimensions, rather than just in one dimension if we follow algorithm 5. In fact, the goodness of the generated density is a nice test of your PRNG: the values of $\exp(-x)$ should be *uniformly* distributed in the unit interval!

.

# 10 Phase space algorithms for particle physics

In particle physics, a recurrent integration problem is that of a differential cross section over the collection of allowed final states. The determination of the differential cross section is outside the scope of these lectures, rather we are interested in the phase space integration itself.

## 10.1 The uniform phase space problem in particle phenomenology

In particle phenomenology (*not* in statistical physics), *phase space* denotes the collection of all possible final-state *momenta.*, including the constraints posed by the on-shell consitions on the individual momenta as well as the overall restriction posed by conservation of energy and momentum. Denoting the masses of the particles by $m_j$ and the four-momenta by $p_j^\mu$ $(j = 1, 2, \ldots, n)$, and the *total* four-momentum by $P^\mu$, the (relativistic) phase space integration element reads[84]

$$dV_n(P; p_1, \ldots, p_n) = \prod_{j=1}^{n} \left( d^4 p_j \, \delta(p_j^2 - m_j^2) \right) \, \delta^4 \left( P - \sum_{j=1}^{n} p_j \right) \quad . \tag{224}$$

It defines a $(3n-4)$-dimensional subspace of the $4n$-dimensional space of all momentum components. Because of the mass-shell conditions, this subspace

[84]There is an additional constraint that the energies of all momenta be *positive*. This is usually left to be understood.

is highly nonlinear. *The* ‘phase space problem’ is that of designing algorithms to generate phase space points (‘events’) with uniform probability.

ℵ That this is far from trivial can be seen from the fact that the *volume* of phase space is in general unknown. The only closed-form results I am aware of are those for $n = 2$ with arbitrary masses, for $n = 3$ with two or three zero masses, and for general $n$ the *ultra-relativistic limit* with $m_j = 0$, and the *nonrelativistic limit* where $P^2 \approx (m_1 + \cdots + m_n)^2$. For the case of a single nonzero mass and several vanishing masses, see appendix 11.6.1.

## 10.2 Two-body phase space

### 10.2.1 The two-body algorithm

The simplest phase space is the two-body one: by standard methods we have

$$\begin{aligned} dV_2(P;p_1,p_2) &= d^4p_1\,\delta(p_1^2 - m_1^2)\,d^4p_2\,\delta(p_2^2 - m_2^2)\,\delta^4(P - p_1 - p_2) \\ &= \frac{1}{8}\mathcal{F}\left(\frac{m_1^2}{s}, \frac{m_2^2}{s}\right)\,d\Omega \quad , \end{aligned} \tag{225}$$

where $s = P^2$, $\Omega$ is the solid angle of $\mathbf{p}_1$ in the rest frame of $P^\mu$, and

$$\mathcal{F}\,(x,y) = \Big((1 - x - y)^2 - 4xy\Big)^{1/2} \quad . \tag{226}$$

The algorithm is given below; the momenta $p_{1,2}$ are generated in the rest frame of $P$, and then boosted to the frame in which $P^\mu$ was given. The

**Algorithm 15** Two-body phase space with masses $m_{1,2}$ and total invariant energy $\sqrt{s} > m_1 + m_2$

```
$p_1^0 \leftarrow (s + m_1^2 - m_2^2)/2\sqrt{s}$
$p_2^0 \leftarrow \sqrt{s} - p_1^0$   ,   $q \leftarrow \sqrt{(p_1^0)^2 - m_1^2}$  {compute energies and momenta}
$\cos\theta \leftarrow -1 + 2\rho$   ,   $\phi \leftarrow 2\pi\rho$ {the solid angle}
$p_1^1 \leftarrow q \sin\theta \cos\phi$   ,   $p_1^2 \leftarrow q \sin\theta \sin\phi$   ,   $p_1^3 \leftarrow q \cos\theta$
$\mathbf{p}_2 \leftarrow -\mathbf{p}_1$ {construct the momenta in the $P$ rest frame}
boost $p_{1,2}^\mu$ to the actual frame of $P^\mu$ (see algorithm 16)
```

Lorentz boost is that which takes the vector $(\sqrt{s}, \vec{0})$ over into $(P^0, \mathbf{P})$. We give it here separately.

**Algorithm 16** Lorentz boost from $P^\mu$, with $P^2 = s$, at rest to given form, applied on vector $p^\mu$. The resultant vector is $q^\mu$.

```
$q^0 \leftarrow (p^0 P^0 + p^1 P^1 + p^2 P^2 + p^3 P^3)/\sqrt{s}$ {Note signs!}
$\vec{q} \leftarrow \vec{p} + \vec{P}\,(p^0 + q^0)/(\sqrt{s} + P^0)$
```

### 10.2.2 Two-body reduction

Let us manipulate Eq.(224) by multiplying it with a clever choice of unity:

$$
\begin{aligned}
dV(P;p_1,p_2,\ldots,p_n) = \prod_{j=1}^{n}\left(d^4p_j\,\delta(p_j^2-m_j^2)\right)\ \delta^4\left(P-\sum_{j=1}^{n}\right) \\
\times\ d^4q_1\,\delta^4\left(q_1-\sum_{j=2}^{n}\right)\ du_1\ \delta(q_1^2-u_1) \\
=\ dV_2(P;p_1,q_1)\ du_1\ dV_{n-1}(q_1;p_2,\ldots,p_n)\ \ . \qquad (227)
\end{aligned}
$$

We can continue this so that eventually

$$
\begin{aligned}
dV_n(P;p_1,\ldots,p_n) &= dV_2(P;p_1,q_1)\,du_1 \\
&\quad \times dV_2(q_1;p_2,q_2)\,du_2 \\
&\quad \times dV_2(q_2;p_3,q_3)\,du_3\cdots \\
\cdots &\quad \times dV_2(q_{n-3};p_{n-2},q_{n-2})\,du_{n-3} \\
&\quad \times dV_2(q_{n-2};p_{n-1},p_n)\,du_{n-2}\ \ . \qquad (228)
\end{aligned}
$$

The $n$-body problem has now been split up into a cascade of $n-1$ two-body problems and a selection of $n-2$ invariant masses squared. Using the result for the two-body case, the *volume* of the $n$-body phase space is then given by an $(n-2)$-dimensional integral:

$$V_n(P) = \left(\frac{\pi}{2}\right)^{n-1} \int du_1 \cdots du_{n-2} \left( \prod_{j=1}^{n-2} \mathcal{F}\left(\frac{u_j}{u_{j-1}}, \frac{m_j^2}{u_{j-1}}\right)\right) \mathcal{F}\left(\frac{m_{n-1}^2}{u_{n-2}}, \frac{m_j^2}{u_{n-2}}\right) \quad , \tag{229}$$

where $u_0 = s$. The integration boundaries are

$$\sqrt{u_j} < \sqrt{u_{j-1}} - m_j \;\; (j = 1, \ldots, n-2) \quad , \quad \sqrt{u_{n-2}} > m_{n-1} + m_n \quad . \tag{230}$$

Unsurprisingly the general result for $V_n(P)$ is not known, nor do we have an algorithm for generating the $u$'s uniformly.

## 10.3 The relativistic problem

The phase space problem becomes simpler if we can neglect the masses, or at least assume that they are small(ish).

### 10.3.1 Two-body reduction algorithm

If we let all the particle masses vanish, Eq.(229) becomes simpler:

$$V_n^{(0)}(P) = \left(\frac{\pi}{2}\right)^{n-1} \int du_1 \cdots du_{n-2} \prod_{j=1}^{n-2} \left(1 - \frac{u_j}{u_{j-1}}\right) \quad , \tag{231}$$

with $s > u_1 > u_2 > \cdots > u_{n-2} > 0$. Following [32] we introduce new variables $v_j = u_j/u_{j-1}$ $(j = 1, \ldots, n-2)$ and then we find

$$\begin{aligned} V_n^{(0)}(P) &= \left(\frac{\pi}{2}\right)^{n-1} s^{n-2} \int_0^1 dv_1 \cdots dv_{n-2} \prod_{j=1}^{n-2} v_j^{n-2-j}(1-v_j) \\ &= \left(\frac{\pi}{2}\right)^{n-1} \frac{s^{n-2}}{(n-1)!(n-2)!} \quad . \end{aligned} \tag{232}$$

The remaining problem is to generate variables $v$ according to $v^k(1-v)$. It can be done by inversion, where we solve

$$\int_0^v dw\, w^k(1-w) = \rho \int_0^1 dw\, w^k(1-w) \quad \to \quad (k+2)v^{k+1} - (k+1)v^{k+2} = \rho \quad . \tag{233}$$

All this then leads to the following algorithm for massless $n$-body phase space.

**Algorithm 17** The Platzer algorithm for $n \geq 3$

```
generate $v_j$ according to $(k+1)(k+2)v_j^{n-2-k}(1-v_j)$ $(j = 1, \ldots, n-2)$
compute $u_j = v_j u_{j-1}$ $(j = 1, \ldots, n-2)$
use algorithm 15 for the cascade $P \to p_1 + q_1$ , $q_1 \to p_2 + q_2$ , $q_2 \to p_3 + q_3$ , $\ldots$, $q_{n-2} \to p_{n-1} + p_n$
```

ℵ This way of handling many-body phase space has a long history; I have simply referred to its most recent incarnation.

### 10.3.2 Massless `RAMBO`

The algorithm of the previous section imposes a hierarchy on the momenta that has no physical basis, and involves $n-1$ Lorentz boosts than may lead to numerical inaccuracies [32]. On the other hand, we can generalise the Euler algorithm to impose the overall constraint of momentum conservation, and arrive at a more 'democratic' approach : the `RAMBO` algorithm that is another case of the Rule of Nifty. We start by generating unconstrained massless momenta, and then modify them to enforce the correct overall momentum. From

$$\int d^4q\,\delta(q^2)\,\exp(-q^0) = \int\limits_0^\infty dq^0\,\frac{q^0}{2}\,\exp(-q^0)\,\int d\Omega = 2\pi \tag{234}$$

we see that a UA reads

$$1 = \frac{1}{(2\pi)^n}\int\prod_{j=1}^n d^4q_j\,\delta(q_j^2)\,\exp(-q_j^0)\quad . \tag{235}$$

The momenta $q_j$ add up to a total momentum $Q$. We then perform a scaling by a factor $\sqrt{Q^2/s}$ so that $Q$ gets the right invariant mass, and a Lorentz boost $\Lambda$ that takes the vector $Q$ into its rest frame. We can write this out as

$$\begin{aligned}1 \;=\; & \frac{1}{(2\pi)^n}\int\prod_{j=1}^{n} d^4q_j\,\delta(q_j^2)\,\exp(-q_j^0)\\ & \qquad d^4Q\,\delta^4\left(Q-\sum_{j=1}^{n}q_j\right)\,d(x^2)\,\delta\left(x^2-\frac{Q^2}{s}\right)\\ & \qquad \prod_{j=1}^{n} d^4p_j\,\delta^4\left(p_j-\frac{1}{x}\Lambda q_j\right)\quad . \qquad (236)\end{aligned}$$

We now eliminate the $q_j$, using

$$\begin{aligned}&\delta^4\left(\sum_{j=1}^{n}q_j-Q\right)=\frac{1}{x^4}\,\delta\left(\sum_{j=1}^{n}p_j-\frac{1}{x}\Lambda Q\right)=\frac{1}{x^4}\,\delta^4\left(\sum_{j=1}^{n}p_j-P\right)\quad ,\\ &\delta^4\left(p_j-\frac{1}{x}\Lambda q_j\right)=x^4\,\delta^4(q_j-x\Lambda^{-1}p_j)\ \ ,\ \ \delta(q_j^2)=\frac{1}{x^2}\,\delta(p_j^2)\quad , \qquad (237)\end{aligned}$$

so as to arrive at

$$\begin{aligned}1 \;&=\; \mathcal{A}\int dV_n(P;p_1,\ldots,p_n)\quad ,\\ \mathcal{A} \;&=\; \frac{1}{(2\pi)^n}\int d^4Q\,d(x^2)\,\exp(-Q^0)\,x^{2n-4}\,\delta(x^2-Q^2/s)\\ &=\; \frac{2(2\pi)^{n-1}}{s^{n-2}}\int\limits_0^\infty dQ^0\int\limits_0^{Q^0} d|Q|\;|Q|^2\left((Q^0)^2-|Q|^2\right)^{n-2}\exp(-Q^0)\end{aligned}$$

$$= \quad 1/V_n^{(0)}(P) \quad . \tag{238}$$

This proves the correctness of the `RAMBO` algorithm. Note that, as before, the damping factor $\exp(-q_j^0)$ precisely guarantees a uniform density. In addition we see that particle masses cannot be easily included because of the scaling by the *a priori* unknown scaling factor $x$. Finally, we see that $4n$ pseudo-random numbers are used to sample points in $(3n-4)$-dimensional space. Therefore, information is lost, and the $4n$-dimensional space of random numbers contains subspaces of dimension $n+4$ in which every point ends up in the same phase space point.

**Algorithm 18** The `RAMBO` algorithm for $n$ momenta with total invariant mass squared $s$

```
generate energies $q_j^0$ according to $q_j^0 \exp(-q_j^0)$, $1 \le j \le n$
generate momenta $\mathbf{q}_j$ isotropically, with $|\mathbf{q}_j| = q_j^0$, $1 \le j \le n$
compute $Q^\mu = \sum_j q_j^\mu$, and $x = \sqrt{Q^2/s}$
$p_j \leftarrow \Lambda q_j/x$, $1 \le j \le n$ {$\Lambda$ is the Lorentz boost that brings $Q$ to its rest
frame, see algorithm19}
each event carries a weight given by Eq.(232)
```

**Algorithm 19** Lorentz boost from $P^\mu$, with $P^2 = s$, to rest from given form, applied on vector $p^\mu$. The resultant vector is $q^\mu$.

```
$q^0 \leftarrow (p^0 P^0 - p^1 P^1 - p^2 P^2 - p^3 P^3)/\sqrt{s}$ {Note signs!}
$\vec{q} \leftarrow \vec{p} - \vec{P}\,(p^0 + q^0)/(\sqrt{s} + P^0)$
```

### 10.3.3 Inclusion of masses

For the case where the particle masses are nonzero no algorithm is known that populates phase space uniformly[85]. However, if the masses (and $n$) are not too large the following procedure can be used. We start with generating massless momenta $p_j$. We then proceed to scale down the 3-momenta of the particles with a common factor $\xi$, which opens up room to increase the energies such that masses can be accommodated. We therefore set

$$\Phi(\xi) = \sum_{j=1}^{n} \sqrt{m_j^2 + \xi^2 |\mathbf{p}_j|^2} = \sqrt{s} \ . \tag{239}$$

Since $\Phi(\xi)$ is monotonic and a solution with $0 < \xi < 1$ exists, it is not difficult to find the right value of $\xi$, and the UA description becomes

$$1 \ = \ \frac{1}{V_n(P)} \int \left( \prod_{j=1}^{n} d^4 p_j \, \delta(p_j^2 - m_j^2) \right) \delta^4 \left( \sum_{j=1}^{n} p_j - P \right)$$

[85]Except for $n = 2$, see above, and for $n = 3$ with $m_1 > 0, m_{2,3} = 0$.

$$
\begin{aligned}
&= \frac{1}{V_n(P)} \int \left( \prod_{j=1}^{n} \frac{1}{2|\mathbf{p}_j|} d^3\mathbf{p}_j \right) \delta^3\left( \sum_{j=1}^{n} \mathbf{p}_j \right) \delta\left( \sum_{j=1}^{n} |\mathbf{p}_j| - \sqrt{s} \right) \\
&\quad d\xi\, \Phi'(\xi)\, \delta(\Phi(\xi) - \sqrt{s}) \\
&\quad \left( \prod_{j=1}^{n} d^3\mathbf{k}_j\, \delta^3(\mathbf{k}_j - \xi\mathbf{p}_j)\, dk_j^0\, \delta\left( k_j^0 - \sqrt{|\mathbf{k}_j|^2 + m_j^2} \right) \right) \quad , \qquad (240)
\end{aligned}
$$

which after standard[86] manipulations results in

$$
\begin{aligned}
1 &= \int \left( \prod_{j=1}^{n} d^4k_j\, \delta(k_j^2 - m_j^2) \right) \delta^4\left( \sum_{j=1}^{n} k_j - P \right) \frac{\mathcal{G}}{V_n(P)} \quad , \\
\mathcal{G} &= \left( \prod_{j=1}^{n} \frac{k_j^0}{|\mathbf{k}_j|} \right) \left( \sum_{j=1}^{n} \frac{|\mathbf{k}_j|^2}{k_j^0 \sqrt{s}} \right) \left( \sum_{j=1}^{n} \frac{|\mathbf{k}_j|}{\sqrt{s}} \right)^{3-2n} \quad . \qquad (241)
\end{aligned}
$$

The scaling algorithm therefore results in a nonuniform sampling, with event weights given by $V_n(P)/\mathcal{G}$. The scaling operation is reversible, and no information is lost[87]. On the other hand, it is not known what the maximum weight is for general masses.

[86]By now, hopefully!

[87]We can reconstruct the ‘original’ $p_j$ from the $k_j$.

**Algorithm 20** Giving masses to massless momenta

```
generate $n$ massless momenta $p_j$ using algorithm 18
find $\xi$ such that $\sum_j \sqrt{|\mathbf{q}_j|^2\xi^2 + m_j^2} = \sqrt{s}$
$\mathbf{k}_j \leftarrow \xi\mathbf{q}_j$, $k_j^0 \leftarrow \sqrt{|\mathbf{k}_j|^2 + m_j^2}$, $1 \le j \le n$
each event carries a weight given by $V_n(P)/\mathcal{G}$, see Eq.(241)
```

ℵ Consider $n = 3$. In the massless case, the phase space (the *Dalitz plot*) is a triangle in terms of two of the energies: $0 < p^0_{1,2} < \sqrt{s}/2$ and $p^0_1 + p^0_2 < \sqrt{s}$. When masses are introduced, the triangle is 'contracted' into a figure with rounded edges and no sharp points. A *uniform* mapping is therefore not possible, and the events will have nonconstant weights. Another message is that the 'contraction' may be expected to be minimal (and hence the weight maximal) for those events that are 'as far as possible' from the edges of phase space. This translates into the rule of thumb that the maximum weight is attained when all massless particles have zero energy, and all massive particles have the highest possible velocity (in whatever sense this makes sense).

## 10.4 Nonrelativistic phase space: BOLTZ

At the other extreme of the energy range we have nonrelativistic phase space. This also allows for a 'democratic' approach, as we shall now show. From

$$\int d^3\mathbf{q}\,\exp\left(-\frac{|\mathbf{q}|^2}{2m}\right) = (2\pi m)^{3/2} \tag{242}$$

we see that a UA can be given:

$$1 = A\int\prod_{j=1}^{n} d^3\mathbf{q}_j\,\exp\left(-\frac{|\mathbf{q}_j|^2}{2m}\right) \quad , \quad A = \prod_{j=1}^{n}(2\pi m_j)^{-3/2} \quad . \tag{243}$$

Obviously, no mass is allowed to vanish here. Having generated the $\mathbf{q}_j$ we then perform a *Galilei* transformation that takes the momenta over into new momenta $\mathbf{k}_j$ that add up to zero. Subsequently we scale the momenta $\mathbf{k}_j$ into $\mathbf{p}_j$ that have the correct total kinetic energy $U$. With the notation $\mathbf{Q} = \sum_j \mathbf{q}_j$, $M = \sum_j m_j$, $E_k = \sum_j |\mathbf{k}_j|^2/(2m_j)$, and $\hat{A} = (2\pi M)^{3/2}A$, we can analyse this procedure as follows, by successive elimination of the $\mathbf{q}_j$ and the $\mathbf{k}_j$:

$$\begin{aligned}
1 &= A\int\left(\prod_{j=1}^{n} d^3\mathbf{q}_j\,\exp\left(-\frac{|\mathbf{q}_j|^2}{2m}\right)\right) \\
&\quad \times d^3\mathbf{v}\,\delta^3\left(\frac{\mathbf{Q}}{M}-\mathbf{v}\right)\,\left(\prod_{j=1}^{n} d^3\mathbf{k}_j\,\delta^3(\mathbf{k}_j-\mathbf{q}_j+m_j\mathbf{v})\right)
\end{aligned}$$

$$\begin{aligned}
&= A\int\left(\prod_{j=1}^n d^3\mathbf{k}_j\right)\,\delta^3\left(\sum_{j=1}^n \mathbf{k}_j\right)\,d^3\mathbf{v}M^3\,\exp\left(-E_k-\frac{M|\mathbf{v}|^2}{2}\right)\\
&= \hat{A}\int\left(\prod_{j=1}^n d^3\mathbf{k}_j\right)\,\delta^3\left(\sum_{j=1}^n \mathbf{k}_j\right)\,\exp(-E_k)\\
&\quad\times d(x^2)\,\delta\left(x^2-\frac{E_k}{U}\right)\,\left(\prod_{j=1}^n d^3\mathbf{p}_j\,\delta^3\left(\mathbf{p}_j-\frac{1}{x}\mathbf{k}_j\right)\right)\\
&= \hat{A}\int\left(\prod_{j=1}^n \mathbf{p}_j\right)\,\delta^3\left(\sum_{j=1}^n \mathbf{p}_j\right)\,\delta\left(\sum_{j=1}^n\frac{|\mathbf{p}_j|^2}{2m_j}-U\right)\,d(x^2)x^{3n-5}\,\exp(-Ux^2)\,U\\
&= \hat{A}\,\Gamma\left(\frac{3n-3}{2}\right)\,U^{(5-3n)/2}\int\left(\prod_{j=1}^n \mathbf{p}_j\right)\,\delta^3\left(\sum_{j=1}^n \mathbf{p}_j\right)\,\delta\left(\sum_{j=1}^n\frac{|\mathbf{p}_j|^2}{2m_j}-U\right)\quad .
\end{aligned} \tag{244}$$

We have thus proven that we can sample nonrelativistic phase space uniformly, and have also computed the volume of this phase space: it reads

$$V_{nr}(U;m_1,\ldots,m_n)=\frac{\prod_{j=1}^n(2\pi m_j)^{3/2}}{(2\pi M)^{3/2}}\frac{U^{(3n-5)/2}}{\Gamma((3n-3)/2)}\quad . \tag{245}$$

This Rule of Nifty result is the basis of the `BOLTZ` algorithm[88]. In statistical-physics-speak, it generates the *microcanonical* ensemble.

[88]Boltz, man!

**Algorithm 21** The `BOLTZ` algorithm for total energy $U$ and masses $m_{1,2,\ldots,n}$

```
generate $q_j^r$ according to $\exp(-(q_j^r)^2/(2m_j))$ for $r = x, y, z$ and $1 \leq j \leq n$
$\mathbf{v} \leftarrow \sum_j \mathbf{q}_j / \sum_j m_j$
$\mathbf{k}_j \leftarrow \mathbf{q}_j - m_j\mathbf{v}$ for $1 \leq j \leq n$
compute $E_k$ and $x = \sqrt{E_k/U}$
$\mathbf{p}_j \leftarrow \mathbf{k}_j/x$ for $1 \leq j \leq n$
each event carries a weight given by Eq.(245)
```

ℵ I do not know of a spacetime transformation that 'interpolates' between a Galilei and a Lorentz transform. This makes it understandable why the general phase space problem is so hard: there is no Rule of Nifty way of computing the phase spave volume for general mass configurations.

# 11 Appendices

## 11.1 About estimators

### 11.1.1 Falling powers

The falling powers are defined as

$$N^{\underline{k}} = N(N-1)(N-2)\cdots(N-k+1) = \frac{N!}{(N-k)!} \tag{246}$$

By its definition, we have $N^{\underline{k}} = 0$ when $k > N$, and $N^{\underline{N}} = N!$. For large $N$ and finite $k$ we can approximate

$$N^{\underline{k}} \approx N^k \left(1 - \frac{k(k-1)}{2N}\right) \quad . \tag{247}$$

From $N^{\underline{k}} = N!/(N-k)!$ we find immediately the binomial sum

$$\sum_{k\geq 0} \frac{x^k}{k!} N^{\underline{k}} = \sum_{k\geq 0} \binom{N}{k} x^k = (1+x)^N \quad . \tag{248}$$

We can extend this by summing over $N$ as well:

$$\begin{aligned} \sum_{N\geq 0}\sum_{k\geq 0} \frac{x^k}{k!} N^{\underline{k}}\, y^N &= \sum_{N\geq 0} y^N (1+x)^N \\ &= \frac{1}{1-y-xy} = \sum_{m\geq 0} \frac{x^m y^m}{(1-y)^{m+1}} \quad . \end{aligned} \tag{249}$$

Isolating from this the power $x^k$ gives

$$\sum_{N\geq 0} y^N N^{\underline{k}} = \frac{k!\,y^k}{(1-y)^{k+1}} \quad . \tag{250}$$

Eq.(248) also leads to the 'binomial theorem for falling powers':

$$(x+y)^{\underline{s}} = \sum_{r=0}^{s} \binom{s}{r} x^{\underline{r}}\, y^{\underline{s-r}} \quad . \tag{251}$$

### 11.1.2 Relations between direct sums and unequal-sums

$$\begin{aligned}
{S_1}^2 &= S_2 + S_{1,1} \quad , \\
S_2 S_1 &= S_3 + S_{2,1} \quad , \\
{S_1}^3 &= S_3 + 3\,S_{2,1} + S_{1,1,1} \quad , \\
S_3 S_1 &= S_4 + S_{3,1} \quad , \\
{S_2}^2 &= S_4 + S_{2,2} \quad , \\
S_2 {S_1}^2 &= S_4 + 2\,S_{3,1} + S_{2,2} + S_{2,1,1} \quad , \\
{S_1}^4 &= S_4 + 4\,S_{3,1} + 3\,S_{2,2} + 6\,S_{2,1,1} + S_{1,1,1,1} \quad ;
\end{aligned} \tag{252}$$

and conversely,

$$S_{1,1} = {S_1}^2 - S_2 \quad ,$$

$$
\begin{aligned}
S_{2,1} &= S_2S_1 - S_3 \;, \\
S_{1,1,1} &= {S_1}^3 - 3\,S_2S_1 + 2\,S_3 \;, \\
S_{3,1} &= S_3S_1 - S_4 \;, \\
S_{2,2} &= {S_2}^2 - S_4 \;, \\
S_{2,1,1} &= S_2{S_1}^2 - 2\,S_3S_1 - {S_2}^2 + 2\,S_4 \;, \\
S_{1,1,1,1} &= {S_1}^4 - 6\,S_2{S_1}^2 + 8\,S_3S_1 + 3\,{S_2}^2 - 6\,S_4 \;. \qquad (253)
\end{aligned}
$$

## 11.2 About integrals

### 11.2.1 An exponential sum and the Poisson formula

Consider the function

$$\delta(s;x) = \sum_n s^{|n|}\, e^{2i\pi nx} = \frac{1-s^2}{1-2s\cos(2\pi x)+s^2} \;, \qquad (254)$$

where $0 < s < 1$. As $s \to 1$, $\delta(s;x)$ goes to zero except for integer $x$, where it approaches infinity. Moreover,

$$\int\limits_{k-1/2}^{k+1/2} dx\ \delta(s;x) = 1 \qquad (255)$$

for any integer $k$. Consequently we may write

$$\sum_n e^{2i\pi nx} = \sum_k \ \delta(x-k) \qquad (256)$$

in the sense of distributions, that is integrated with a suitable test function. This implies the Poisson formula:

$$\sum_k f(k) = \sum_n g(n) \quad , \quad g(y) = \int dx\, f(x)\, e^{2i\pi xy} \quad . \tag{257}$$

### 11.2.2 About the integral (40)

For $M$ large, we can approximate

$$\begin{aligned} \log\left((1+x/M)^M\right) &= M\log(1+x/M) \\ &= M\left(\frac{x}{M} - \frac{x^2}{2M^2} + \frac{x^3}{3M^3} - \frac{x^4}{4M^3} + \cdots\right) \\ &= x - \frac{x^2}{2M} + \frac{x^3}{3M^2} - \frac{x^4}{4M^3} + \cdots \end{aligned} \tag{258}$$

We can therefore estimate the integral

$$\int_0^\infty dx\, e^{-x}\left(1+\frac{x}{M}\right)^M = \int_0^\infty dx\ \exp\left(-\frac{x^2}{2M} + \frac{x^3}{3M^2} - \frac{x^4}{4M^4} + \cdots\right)$$
$$\approx \int_0^\infty dx\ e^{-x^2/2M}\ \left(1 + \frac{x^3}{3M^2} - \frac{x^4}{4M^3} + \frac{x^6}{18M^4} + \cdots\right)$$

$$
\begin{aligned}
&= \frac{1}{2}\sqrt{2\pi\, M}\left(1 - \frac{(3M^2)}{4M^3} + \frac{(15M^3)}{18M^4}\right) + \frac{(2M)^2}{6M^2} + \cdots \\
&= \sqrt{\frac{\pi\, M}{2}}\left(1 + \frac{1}{12M}\right) + \frac{2}{3} + \cdots
\end{aligned} \tag{259}
$$

The neglected terms are of order $1/M$ and smaller.

## 11.3 About PRNGs

### 11.3.1 Selfies

Consider an 'arbitrarily chosen algorithm' of the type of Eq.(35). The number $a$ is a *selfie* if $f(a) = a$. The probability that a given number is a selfie is $1/M$, and therefore the probability of having exactly $k$ selfies in an arbitrarily chosen algorithm is

$$
S_M(k) = \binom{M}{k}\left(\frac{1}{M}\right)^k\left(1 - \frac{1}{M}\right)^{M-k} \quad . \tag{260}
$$

Since

$$
S_M(0) = \left(1 - \frac{1}{M}\right)^M \approx \frac{1}{e}\left(1 - \frac{1}{2M} + \mathcal{O}\left(\frac{1}{M^2}\right)\right) \quad , \tag{261}
$$

the probability to have *at least* one selfie is about $1 - 1/e \approx 63\%$. The largest probability is $S_M(1)$, and incidentally

$$
\sum_{k=0}^{M} k S_M(k) = 1 \quad , \tag{262}
$$

so that the *expected number* of selfies is exactly 1, independent of $M$. This therefore also holds for shift-register PRNGs.

### 11.3.2 Selfies in linear congruential PRNGs

We can investigate whether a linear congruential generator contains a selfie. We define the *selfie property* $\mathcal{S}(m,b,c)$ which states that the algorithm $x_n \leftarrow ((b+1)x_{n-1}+c) \bmod m$ contains a selfie. That means that there are some integers $x$ and $y$ in $(0,\ldots,m-1)$ such that

$$bx + c = (y+1)m \quad . \tag{263}$$

But *that* implies that

$$ym + (m-c) = bx \quad ,\text{or} \quad \left\{ y(m \bmod b) + (m-c) \bmod b \right\} \bmod b = 0 \quad , \tag{264}$$

another selfie property! We conclude that

$$\begin{aligned} &\mathcal{S}(m,b,c) \quad \Leftrightarrow \quad \mathcal{S}(m',b',c') \quad , \\ &m' = b \quad , \quad b' = m \bmod b \quad , \quad c' = (m-c) \bmod b \quad . \end{aligned} \tag{265}$$

We can repeat this argument, and the arguments of $\mathcal{S}$ will keep getting smaller and smaller. Once we reach $\mathcal{S}(m,b,0)$ we are done since then there *is* indeed a selfie, namely $x=y=0$, and the original algorithm also has a selfie. On the other hand, we might attain $\mathcal{S}(m,0,c)$ and we see that there is *no* selfie. In particular, if the Hull-Dobell conditions of sect. 3.3.3 hold there is no selfie, since they imply that $m \bmod b = 0$. If $m$ and $b$ are relatively prime (and *a fortiori* if $m$ is a prime number), then so are $m'$ and $b'$, and $c' < b'$. In that case we shall always reach $\mathcal{S}(m', b,', 0)$ and there is a selfie. Finally, for $b = 0$ $(a = 1)$ there cannot be a selfie for prime $m$.

### 11.3.3 Serial correlation in a real-number model

Consider a multiplicative congruential PRNG with modulus $m$, multiplier $a$ and increment $c$. The serial correlation is

$$r = \frac{\sum\limits_{n=1}^{k} x_n x_{n+1} - \left(\sum\limits_{n=1}^{k} x_n\right)^2}{\sum\limits_{n=1}^{k} x_n^2 - \left(\sum\limits_{n=1}^{k} x_n\right)^2} \quad , \tag{266}$$

a version of the Pearson correlation coefficient. We can compute an approximate value for this correlation using the 'real number' model for the PRNG: after scaling by $1/m$, the $z_n = x_n/m$ values will be approximately the real numbers in $(0, 1)$. Then $z_{n+1} = y(z_n)$ with

$$y(z) = (az + \delta) \bmod 1 = (az_n + \delta) - \sum_{k=1}^{a} \theta\left(z > \frac{k - \delta}{a}\right) \quad . \tag{267}$$

Then, assuming uniform distribution of the $z_n$ values we will have approximately

$$\left\langle z_n^2 \right\rangle \approx 1/3 \quad , \quad \left\langle z_n \right\rangle^2 \approx 1/4 \quad , \tag{268}$$

and

$$\begin{aligned}
\langle z_n\, y(z_n)\rangle &\approx \int\limits_0^1 dz\; z\, y(z) = \int\limits_0^1 dz\,(az^2+\delta z) - \sum_{k=1}^{a}\int\limits_{(k-\delta)/a}^{1} dz\, z \\
&= -\frac{a}{6}+\frac{\delta}{2}-\frac{1}{2a^2}\sum_{k=1}^{a}\left(k^2-2\delta k+\delta^2\right) \\
&= \frac{1}{4}+\frac{1}{12a}-\frac{1}{2a}\delta(1-\delta) \quad . \qquad (269)
\end{aligned}$$

The estimate for the correlation is therefore

$$r \approx \frac{1}{a}\Bigg(1-6\,\delta(1-\delta)\Bigg) \quad . \qquad (270)$$

## 11.4 About diaphonies

### 11.4.1 The two-point function for the Euler diaphony

For the one-dimensional Euler diaphony we have

$$\beta(x) = \sum_{n\neq 0}\frac{3}{\pi^2\, n^2}\exp(2i\pi n\, x) \quad . \qquad (271)$$

Now assume that $0 < x < 1$. By differentiating twice we have

$$\beta''(x) = -12\sum_{n\neq 0}\exp(2i\pi nx) = 12 \qquad (272)$$

by virtue of Eq.(256). Thus,

$$\beta(x) = 6x^2 + px + q \tag{273}$$

There are two conditions to be met:

$$\beta(x) = \beta(1-x) \quad , \quad \int\limits_0^1 dx\ \beta(x) = 0 \quad . \tag{274}$$

This leads to

$$\beta(x) = 1 - 6x(1-x) \ \ , \ \ 0 < x < 1 \ \ . \tag{275}$$

At every integer value of $z$ the two-point function has a kink to make it periodic, so the final result must be

$$\beta(z) = 1 - 6\{x\}\left(1 - \{x\}\right) \ \ , \ \ \{x\} = x - \lfloor x \rfloor \ \ . \tag{276}$$

If we are interested only in $-1 \le x \le 1$ we may replace $\{x\}$ by $|z|$.

## 11.5 About QRNGs

### 11.5.1 Rational denominators for continued fractions

The continued fraction with the smallest possible coefficients is

$$\phi_1 = [1,1,1,1,1,\ldots] = \frac{1}{1+\phi_1} = \frac{1}{2}\left(\sqrt{5}-1\right) \ \ , \tag{277}$$

the golden ratio; its approximant denominator obeys

$$q_n = \theta(n=0,1) + \theta(n \geq 2)(q_{n-1} + q_{n-2}) \quad , \tag{278}$$

so that $q_n = F_n$ are the Fibonacci numbers $1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$. We can find the asymptotic behaviour by introducing the generating function

$$\begin{aligned} f_1(x) &= \sum_{n\geq 0} x^n\, q_n = 1 + x + x(f_1(x) - 1) + x^2 f_1(x) \\ &= \frac{1}{1 - x - x^2} \quad . \end{aligned} \tag{279}$$

Its nearest singularity[89] is at $x = \phi_1$. We can therefore approximate[90]

$$f_1(x) \sim \frac{1}{(\phi_1 - x)(2\phi_1 + 1)} = \frac{1}{\phi_1(2\phi_1 + 1)}\,\frac{1}{1 - x/\phi_1} \quad , \tag{280}$$

which gives[91]

$$q_n \sim \frac{1}{\phi_1(2\phi_1 + 1)} \left(\frac{1}{\phi_1}\right)^{n+1} \sim (0.724)\;(1.618)^n \quad . \tag{281}$$

[89]That singularity that lie closest to the origin in the complex plane.

[90]A polynomial $P(x)$ with nondegenerate roots $x_j$ is approximated by $(x - x_j)P'(x_j)$ close to the root.

[91]The *exact* result also contains powers of $1/x_1$ that are therefore exponentially suppressed: the approximation is therefore very good already for moderate $n$.

A continued fraction with nonminimal coefficients is

$$\phi_2 = [2,2,2,2,2,\ldots] = \frac{1}{2+\phi_2} = -1+\sqrt{2} \quad . \tag{282}$$

Its approximant denominators, successively $1, 2, 5, 12, 29, 70, 169, 408, \ldots$ obey the recursion

$$q_n = \theta(n=0) + \theta(n \geq 1)(2q_{n-1} + q_{n-2}) \quad , \tag{283}$$

with generating function

$$f_2(x) = \sum_{n\geq 0} x^n \, q_n = \frac{1}{1-2x-x^2} \quad . \tag{284}$$

Its nearest singularity is at $x = \phi_2$ and proceeding as before we find

$$q_n = \frac{1}{\phi_2(2\phi_2+2)} \left(\frac{1}{\phi_2}\right)^n \sim (0.854)\,(2.414)^n \quad . \tag{285}$$

A continued fraction with 'quite small' coefficients is for instance

$$\phi_{12} = [1,2,1,2,1,2,\ldots] = \frac{1}{1+\frac{1}{2+\phi_{12}}} = \frac{2+\phi_{12}}{3+\phi_{12}} = -1+\sqrt{3} \quad , \tag{286}$$

so that for its approximant denominator we have

$$\begin{aligned} q_n &= \theta(n=0,1) + \theta(n \geq 2, \text{even})(2q_{n-1} + q_{n-2}) \\ &\quad + \theta(n \geq 2, \text{odd})(q_{n-1} + q_{n-2}) \\ &= \theta(n=0,1) + \theta(n \geq 2)(q_{n-1} + q_{n-2}) + \theta(n \geq 2, \text{even}) q_{n-1}, \end{aligned} \tag{287}$$

and for the generating function we find

$$f_{12}(x) = \sum_{\geq 0} x^n\, q_n = 1 + x f_{12}(x) + x^2 f_{12}(x) + x \sum_{n\geq 1} x^{2n-1} q_{2n-1} \quad . \tag{288}$$

Splitting $f(_{12}x)$ into even and odd parts we find that there must be an even function $g(x)$ such that $f_{12}(x) = (1 + x - x^2)g(x)$, and we finally arrive at

$$f_{12}(x) = \frac{1 + x - x^2}{(1 - x^2)^2 - 2x^2} \quad . \tag{289}$$

The resulting denominators are, successively, $1, 1, 3, 4, 11, 15, 41, 56, 153, \ldots$, growing faster than the Fibonacci numbers. The generating function has its nearest singularities at $\pm y_0$, with $y_0 = (\sqrt{3} - 1)/\sqrt{2}$, and we can therefore approximate

$$\begin{aligned} q_n &\sim \frac{1}{4\sqrt{3}} \left[(\sqrt{2}+1) + (-1)^n(\sqrt{2}-1)\right] \left(\frac{1}{y_0}\right)^{n+1} \\ &= \frac{\sqrt{3}+1}{\sqrt{24}} \left(\frac{1}{y_0}\right)^n \left[\theta(n \text{ odd}) + \sqrt{2}\,\theta(n \text{ even})\right] \\ &\sim (0.789)\,(1.932)^n\,, n \text{ even} \quad , \quad (0.558)\,(1.932)^n\,, n \text{ odd} \quad . \end{aligned} \tag{290}$$

The growth rate of the denominators is indeed inbetween that for $\phi_1$ and $\phi_2$.

## 11.6 About phase space

### 11.6.1 The case of a single nonzero mass

In this appendix we derive the phase space volume $n \geq 3$ particles, where $m_1 = m_2 = \cdots = m_{n-1} = 0$ and $m_n > 0$. We shall be interested in the *phase space mass supression*, that is the ratio between this phase space volume and that of the completely massless one, given by Eq.(232). Using this result, and the two-body reduction discussed in section 10.2.2, we can write this as

$$\begin{aligned} V(n;s;m) &= \left(\frac{\pi}{2}\right)^{n-1} \frac{s^{n-2}}{(n-2)!(n-3)!} A_{n-3}(m^2/s) \;, \\ A_k(\alpha^2) &= \int\limits_0^{(1-\alpha)^2} dx\; x^k \;\left((1+\alpha^2-x)^2 - 4\alpha^2\right)^{1/2} \;. \end{aligned} \tag{291}$$

Here $s$ is the total invariant energy, and $m = \alpha\sqrt{s}$ the single nonezero particle mass. The best way to compute this for general $k$ appears to be to use a generating function:

$$\begin{aligned} A(\alpha^2, z) &= \sum_{k\geq 0} A_k(\alpha) z^k = \int\limits_0^{(1-\alpha)^2} dx\; \frac{\left((1+\alpha^2-x)^2 - 4\alpha^2\right)^{1/2}}{1-zx} \\ &= \frac{\alpha}{z} \int\limits_1^{1/\alpha} dw\; \frac{(w^2-1)}{w^2(w^2+2\beta w+1)} \;, \end{aligned} \tag{292}$$

where we have used

$$
\begin{aligned}
w &= \frac{1}{2\alpha}\left(1+\alpha^2-x+\left((1+\alpha^2-x)^2-4\alpha^2\right)^{1/2}\right) \quad , \\
\beta &= (1-z(1+\alpha^2))/(2\alpha z) \quad . \qquad (293)
\end{aligned}
$$

The integral (292) can now be performd by elementary means, and we find

$$
\begin{aligned}
A(\alpha^2,z) &= \frac{1}{z}\Bigg(1-\alpha^2+2\alpha\beta\log(\alpha) \\
&\qquad -2\alpha\sqrt{\beta^2-1}\log\left(\frac{1+\alpha(\beta+\sqrt{\beta^2-1})}{\alpha+\beta+\sqrt{\beta^2-1}}\right)\Bigg) \quad . \qquad (294)
\end{aligned}
$$

The various $A_k(\alpha^2)$ can now be obtained by expansion in powers of $z$[92]. Since

$$
A(0,z) = \frac{1}{z} - \frac{1-z}{z^2}\log(1-z) \ , \qquad (295)
$$

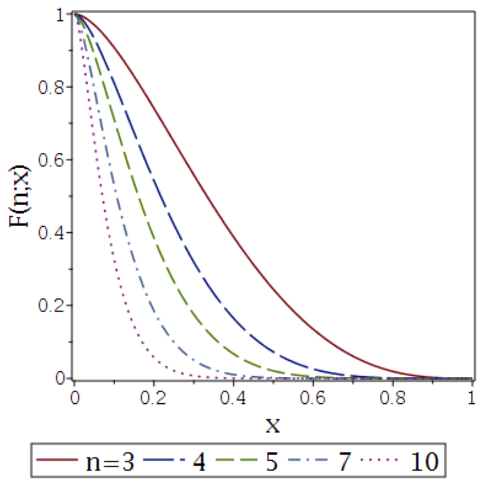
We have $A_k(0) = 1/(k+1)(k+2)$ as we ought to. The phase-space mass suppression function is given by

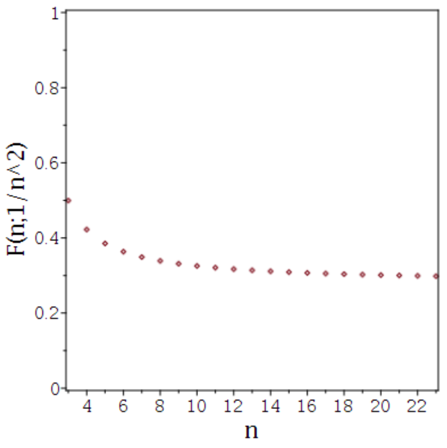
$$
F(n;\alpha^2) = A_{n-3}(\alpha^2)/A_{n-3}(0) \quad . \qquad (296)
$$

The first few instances read

$$
\begin{aligned}
F(3;x) &= 1-x^2+2x\log(x) \quad , \\
F(4;x) &= 1+9x-9x^2-x^3+(6x+6x^2)\log(x) \quad , \\
F(5;x) &= 1+28x-28x^3-x^4+(12x+36x^2+12x^3)\log(x) \quad . \qquad (297)
\end{aligned}
$$

[92]Note that this is not really a closed-form result!





The plot on the left shows $F(n;x)$ for various $n \geq 3$, with $x = m^2/s$. For larger $n$ the phase-space suppression apears to become more significant. If, however, we decide on $m = \sqrt{s}/n$, say, the phase space suppression depends only very mildly on $n$, as indicated by the plot on the right.

## References

[1] W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press (1986).

[2] *Nineteenth Century Clouds over the Dynamical Theory of Heat and Light*, by the Right. Hon. Lord Kelvin, G.C.V.O., D.C.L., LL.D., F.R.S, M.R.I, *Philosophical Magazine* S. 6. Vol. **2**, no. 7. July 1901.

[3] N. Metropolis, S. Ulam, *The Monte Carlo Method*, J. Am. Stat. Ass. **44** (1949) 335.

[4] *Monte Carlo Method* (Proceedings of a symposium in Los Angeles CA, sponsored by the RAND Corporation, the National Bureau of Standards, and the Oak Ridge National Laboratory, June 29 - July 1, 1949), National Bureau of Standards Applied Mathematics Series **12** (1951).

[5] T.F. Chan, G.H. Golub, R.J. Leveque, *Algorithms for Computing the Sample Variance: Analysis and Recommendations*, The American Statistician **37** (1983) 242.

[6] R. Bakx, R. Kleiss, F. Versteegen, *First- and second-order error estimates in Monte Carlo integration*, Comp. Phys. Comm. **208** (2016) 29.

[7] N. Eldredge, S. J. Gould , *Punctuated equilibria: an alternative to phyletic gradualism*, in T.J.M. Schopf, ed., Models in Paleobiology. (San Francisco: Freeman, Cooper and Company), pp. 82-115.

[8] D.E. Knuth, *The Art of Computer Programming, vol.2: Seminumerical Algorithms*, Addison-Wesley, 1981.

[9] T.E. Hull, A.R. Dobell, *Random Number Generators* SIAM Review. **4**(3) 230.

[10] R.D. Carmichael, *Note on a new number theory function* Bull. Am.Math.Soc. **16**(1910)232.

[11] G. Marsaglia, *Random Numbers Fall Mainly in the Planes*, Proc. Nat. Acad. Sci. U.S.A. **61**(1)25.

[12] System/360 Scientific Subroutine Package, Version III, Programmer's Manual. IBM, White Plains, New York, 1968, p. 77; Compaq Fortran Language Reference Manual (Order Number: AA-Q66SD-TK) September 1999 (formerly DIGITAL Fortran and DEC Fortran 90).

[13] http://random.mat.sbg.ac.at/results/karl/server/server.html

[14] M. Lüscher, *A Portable high quality random number generator for lattice field theory simulations*, Comput.Phys.Commun. **79**(1994) 100. This is an adaptation from the original algorithm proposed in
G. Marsaglia and A. Zaman, *A New Class of Random Number Generators*, Ann. Appl. Prob. **1** (1991)462.

[15] M. Matsumoto and T. Nishimura, *Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator*, ACM Trans. Mod. Comp. Sim. **8** (1998).

[16] H. Leeb, MSc thesis, University of Salzburg, 1995.

[17] L. Kuipers and H. Niederreiter, *Uniform Distribution if Sequences*, (Dover, 2006).

[18] H. Woźniakowski, *Average case complexity of multivariate integration*, Bull. AMS **24** (1991) 185.

[19] F. James, J. Hoogland, R. Kleiss, *Multidimensional sampling for simulation and integration: Measures, discrepancies, and quasirandom numbers*, Comp. Phys. Comm. **99** (1997) 180.

[20] A. van Hameren, R. Kleiss , J. Hoogland, *Gaussian limits for discrepancies*, Nucl.Phys.Proc.Suppl. **63** (1998) 988-990.

[21] J.M. Hammersley and D.C. Handscomb, *Monte Carlo Methods* (London: Methuen 1964).

[22] R. Kleiss and R. Pittau, *Weight optimization in multichannel Monte Carlo*, Comp. Phys. Comm. **83** (1994) 141.

[23] B. Bratley, P. Fox and H. Niederreiter. AMC Transactions on Modeling and Computer Simulation 2, 3:195.

[24] A. Lazopoulos, PhD thesis, Radboud University, Nijmegen (2007).

[25] L. Devroye, *Non-uniform Random Variate Generation* (Springer, New York 1986). Out of print, it is now available for free on www.nrbook.com/devroye.

[26] J. von Neumann, *Various Techniques Used in Connection With Random Digits*, contribution to [4]. This 3-page paper is an absolute must for anyone interested in Monte Carlo and its history.

[27] M. Abramowitz and I. Stegun (eds.), *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, (Nat. Bur. Standards, 1964).

[28] G.E.P. Box, M.E. Muller, *A Note on the Generation of Random Normal Deviates.*, Ann. Math. Stat. **29 (2)** (1958) 610.

[29] A.J. Kinderman, J.F. Monahan, *Computer Generation of Random Variables Using the Ratio of Uniform Deviates*, ACM Trans. Math. Softw., **3 (3)** (1977).

[30] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, *Equations of State Calculations by Fast Computing Machines*, J. Chem. Phys. **21 (6)** (1953) 1087;
W.K. Hastings, *Monte Carlo Sampling Methods Using Markov Chains and Their Applications*, Biometrika. **57 (1)** (1970): 97.

[31] S.D. Ellis, R. Kleiss, W.J. Stirling, *A New Monte Carlo Treatment of Multiparticle Phase Space at High energies*. Comp. Phys. Comm. **40** (1986) 359

[32] S. Platzer, *RAMBO on diet*, arXiv:1308.2922

# Index