Markov Chain Monte Carlo Simulations and Their Statistical Analysis – An Overview

Bernd Berg

FSU, August 30, 2005

Content

1. Statistics as needed

2. Markov Chain Monte Carlo (MC)

3. Statistical Analysis of MC Data and Advanced MC.

Link to computer code at www.hep.fsu.edu/~berg .

Probability Distributions and Sampling

In N experiments we may find an event A to occur n times. The **frequency definition** of the **probability** of the event is

$$P(A) \;=\; \lim_{N\to\infty} \frac{n}{N} \; .$$

Let $P(a,b)$ be the probability that $x^r \in [a,b]$ where x^r is a **random variable** drawn in the interval $(-\infty,+\infty)$ with a **probability density** $f(x) > 0$. Then,

$$P(a,b) = \int_a^b dx\, f(x) \quad \text{and} \quad f(x) = \lim_{y\to x} \frac{P(y,x)}{x-y} \; .$$

The **(cumulative) distribution function** of the random variable x^r is defined as

$$F(x) = P(x^r \le x) = \int_{-\infty}^{x} f(x')\, dx' \; .$$

For **uniform probability distribution** between $[0,1)$,

$$u(x) = \begin{cases} 1 & \text{for} \quad 0 \leq x < 1; \\ 0 & \text{elsewhere.} \end{cases}$$

The corresponding distribution function is

$$U(x) = \int_{-\infty}^{x} u(x')\,dx' = \begin{cases} 0 & \text{for} \quad x < 0; \\ x & \text{for} \quad 0 \leq x \leq 1; \\ 1 & \text{for} \quad x > 1. \end{cases}$$

It allows for the construction of general probability distributions. Let

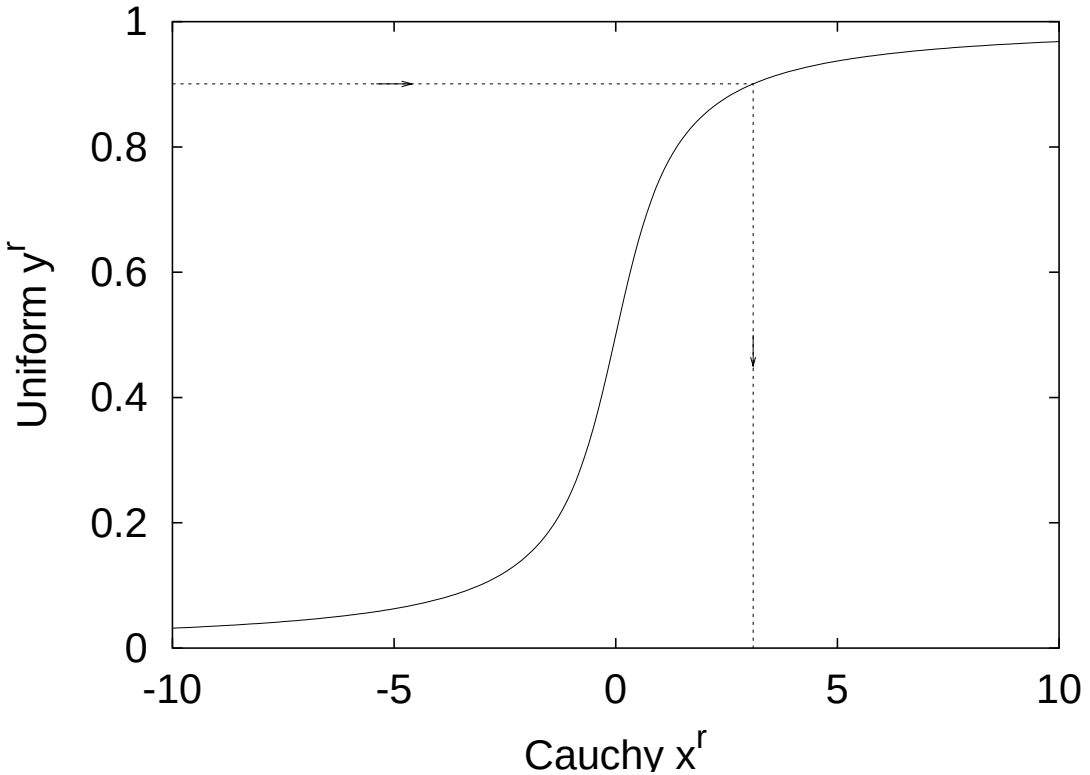
$$y = F(x) = \int_{-\infty}^{x} f(x')\,dx' \ .$$

For y^r being a uniformly distributed random variable in $[0,1)$

$x^r = F^{-1}(y^r)$ is then distributed according to the probability density $f(x)$.

Example

Mapping of the uniform to the Cauchy distribution.

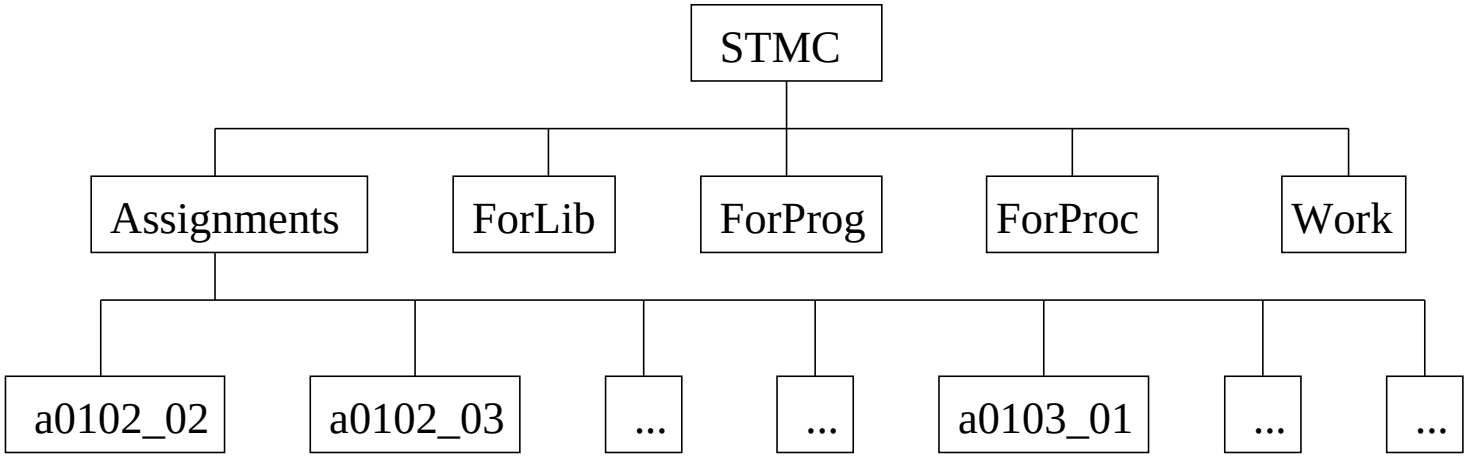


Pseudo Random Numbers and Computer Code

It is sufficient to generate uniform (pseudo) random numbers. Control your random number generator! Therefore, a portable, well-tested generator should be choosen. My code supplies a generator by Marsaglia and collaborators whith an approximate period of 2^{110}. How to get it? Download `STMC.tgz` which unfolds under (Linux)

```
tar -zxvf STMC.tgz
```

into the directory structure shown below.



Routines:

`rmaset.f` sets the initial state of the random number generator.

`ranmar.f` provides one random number per call (function version `rmafun.f`).

`rmasave.f` saves the final state of the generator.

Initial seeds

$$-1801 \leq \texttt{iseed1} \leq 29527 \quad \text{and} \quad -9373 \leq \texttt{iseed2} \leq 20708 \ .$$

give independent series (useful for parallel processing).

Illustration: Assignment a0102_02.

```
RANMAR INITIALIZED.
idat, xr = 1   0.116391063
idat, xr = 2   0.96484679
idat, xr = 3   0.882970393
idat, xr = 4   0.420486867
extra xr =     0.495856345
```

```
MARSAGLIA CONTINUATION.
idat, xr = 1   0.495856345
idat, xr = 2   0.577386141
idat, xr = 3   0.942340136
idat, xr = 4   0.243162394
extra xr =     0.550126791
```

Confidence Intervals and Sorting

One defines **q-tiles** (also **quantiles** or **fractiles**) x_q of a distribution function by

$$F(x_q) \;=\; q \;.$$

An example is the **median** $x_{\frac{1}{2}}$. The probability content of the **confidence interval**

$$[x_q, x_{1-q}] \;\text{ is }\; p = 1 - 2q \;.$$

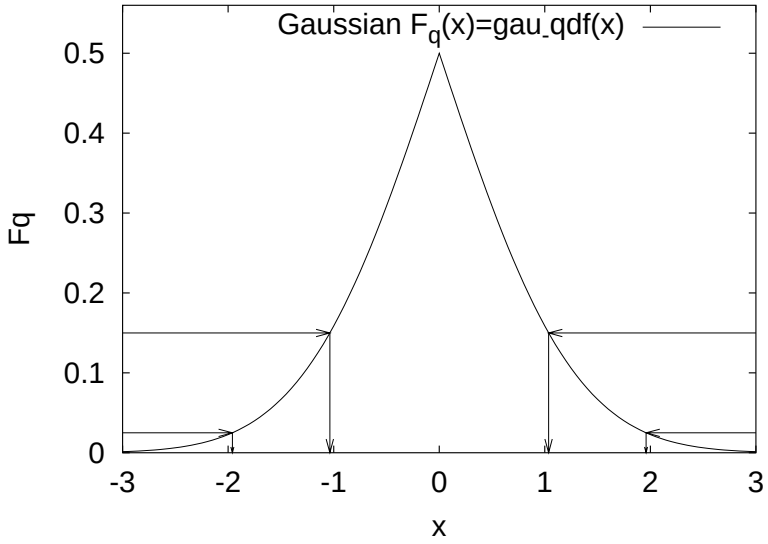
Example: Gaussian or normal distribution of variance σ^2 :

$$[-n\sigma, +n\sigma] \;\Rightarrow\; p = 0.6827 \text{ for } n = 1, \quad p = 0.9545 \text{ for } n = 2 \;.$$

The **peaked distribution function**

$$F_q(x) = \begin{cases} F(x) \text{ for } F(x) \leq \frac{1}{2}, \\ 1 - F(x) \text{ for } F(x) > \frac{1}{2}. \end{cases}$$

provides a graphical visualization of probability intervals of such a distribution:



Sorting allows for an empirical estimate. Assume we generate n random number $x_1, ..., x_n$. We may re-arrange the x_i in increasing order:

$$x_{\pi_1} \leq x_{\pi_2} \leq \ldots \leq x_{\pi_n}$$

where $\pi_1, \ldots, \pi_n$ is a permutation of $1, \ldots, n$.

An estimator for the distribution function $F(x)$ is then the **empirical distribution function**

$$\overline{F}(x) = \frac{i}{n} \quad \text{for} \quad x_{\pi_i} \le x < x_{\pi_{i+1}},\ i = 0, 1, \ldots, n-1, n$$

with the definitions $x_{\pi_0} = -\infty$ and $x_{\pi_{n+1}} = +\infty$. To calculate $\overline{F}(x)$ one needs an efficient way to **sort** n data values in ascending (or descending) order. In the STMC package this is provided by a **heapsort** routine, which arrives at the results in $O(n \log_2 n)$ steps.

Example: Gaussian distribution in assignment a0106_02 (200 and 20,000 data).

Central Limit Theorem: Convergence of the Sample Mean

$$\text{Gaussian} \quad \sigma^2(\overline{x}^r) = \frac{\sigma^2(x^r)}{N} \quad \text{for} \quad \overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \ .$$

Binning

We group NDAT data into NBINS bins, where each binned data point is the arithmetic average of

$$\texttt{NBIN} = [\texttt{NDAT}/\texttt{NBINS}] \quad (\text{Fortran integer division.})$$

original data points. Should not be confused with histogramming! Preferably NDAT is a multiple of NBINS. The purpose of the binning procedure is twofold:

1. When the the central limit theorem applies, the binned data will become practically Gaussian when NBIN is large enough. This allows to apply Gaussian error analysis methods even when the original are not Gaussian.

2. When data are generated by a Markov process subsequent events are correlated. For binned data these autocorrelations are reduced and can be neglected, once NBIN is sufficiently large.

Example:

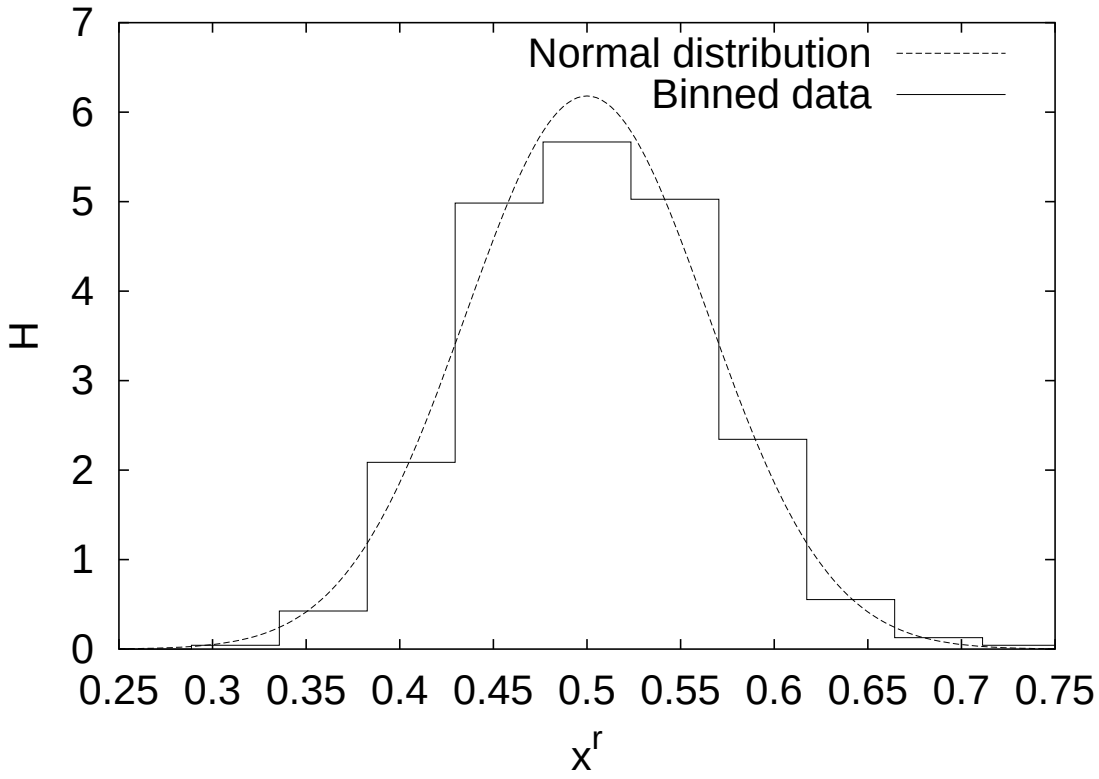


Figure 1: Comparison of a histogram of 500 binned data with the normal distribution $\sqrt{(120/\pi)}\,\exp[-120\,(x-1/2)^2]$. Each binned data point is the average of 20 uniformly distributed random numbers. Assignment a0108_02.

Gaussian Difference Test

One is often faced with comparing two different empirical estimates of some mean. How large must $D = \overline{x} - \overline{y}$ be in order to indicate a real difference? The quotient

$$d^r \;=\; D^r/\sigma_D\,, \qquad \sigma_D^2 = \sigma_{\overline{x}}^2 + \sigma_{\overline{y}}^2$$

is normally distributed with expectation zero and variance one. The **likelihood that the observed difference** $|\overline{x} - \overline{y}|$ **is due to chance** is defined to be

$$Q \;=\; 1 - P(|d^r| \le d) \;=\; 2\,G_0(-d) \;=\; 1 - \mathrm{erf}\left(d/\sqrt{2}\right)\;.$$

If the assumption is correct, then Q is a uniformly distributed random variable in the range $[0, 1)$. **Examples:** (interactive in `ForProc/Gau_dif/`)

$\overline{x}_1 \pm \sigma_{\overline{x}_1}$	1.0 ± 0.1	1.0 ± 0.1	1.0 ± 0.1	1.0 ± 0.05	1.000 ± 0.025
$\overline{x}_2 \pm \sigma_{\overline{x}_2}$	1.2 ± 0.2	1.2 ± 0.1	1.2 ± 0.0	1.2 ± 0.00	1.200 ± 0.025
Q	0.37	0.16	0.046	0.000063	0.15×10^{-7}

Gaussian Error Analysis for Small Samples:

Gosset's Student Distribution and the Student Difference Test.

The Error of the Error Bar:

χ^2 Distribution and the Variance Ratio Test (F-Test).

The Jackknife Approach:

Jackknife estimators correct for the bias and the error of the bias. If there is not bias, their results are identical with the conventional error analysis. As the extra effort of using Jacknife routines is minimal, they should be the standard of error analysis.

Bias problems occur when one estimates a non-linear function of a mean $\widehat{x}$:

$$\widehat{f} = f(\widehat{x}) \ .$$

Jacknife estimators of the function are then defined by

$$\overline{f}^J \;=\; \frac{1}{N}\sum_{i=1}^{N} f_i^J \quad \text{with} \quad f_i^J \;=\; f(x_i^J) \quad \text{and} \quad x_i^J \;=\; \frac{1}{N-1}\sum_{k\neq i} x_k \;.$$

The estimator for the variance is

$$s_J^2(\overline{f}^J) \;=\; \frac{N-1}{N}\sum_{i=1}^{N}(f_i^J - \overline{f}^J)^2 \;.$$

Straightforward algebra shows that in the unbiased case the jackknife variance reduces to the normal variance. Of order N (not N^2) operations are needed to construct the jackknife averages $x_i^J,\ i = 1, \ldots, N$ from the original data.

Determination of Parameters (Fitting):

Levenberg-Marquardt approach besides simple linear regression.

Statistical Physics and Markov Chain Monte Carlo Simulations

MC simulations of systems described by the Gibbs canonical ensemble aim at calculating estimators of physical observables at a temperature T. In the following we consider the calculation of the expectation value of an observable $\mathcal{O}$. All systems on a computer are discrete, because a finite word length has to be used. Hence,

$$\widehat{\mathcal{O}} = \widehat{\mathcal{O}}(\beta) = \langle \mathcal{O} \rangle \quad = \quad Z^{-1} \sum_{k=1}^{K} \mathcal{O}^{(k)} \, e^{-\beta \, E^{(k)}}$$

$$\text{where} \quad Z \; = \; Z(\beta) \quad = \quad \sum_{k=1}^{K} e^{-\beta \, E^{(k)}}$$

is the partition function. The index $k = 1, \ldots, K$ labels all configurations (or microstates) of the system, and $E^{(k)}$ is the (internal) energy of configuration k.

In the following I use Potts models on d-dimensional cubic lattices with periodic boundary conditions. Without being overly complicated, these models allow to illustrate essential features we are interested in. We define the energy of the system by

$$E_0^{(k)} = -2 \sum_{<ij>} \delta(q_i^{(k)}, q_j^{(k)}) + \frac{2\,d\,N}{q} \quad \text{with} \quad \delta(q_i, q_j) = \begin{cases} 1 \text{ for } q_i = q_j \\ 0 \text{ for } q_i \neq q_j \end{cases} .$$

The sum $< ij >$ is over the nearest neighbor lattice sites and the Potts spins or states of $q_i^{(k)}$ take the values $1, \ldots, q$. For the energy per spin the notation is $e_s = E/N$ and our normalization is chosen so that e_s agrees for $q = 2$ with the conventional Ising model definition. For the $2d$ Potts models a number of exact results are known, e.g.,

$$\beta_c \; = \; \frac{1}{T_c} = \frac{1}{2} \ln(1 + \sqrt{q}) = \beta_c^{\text{Potts}} , \qquad e_{0s}^c = E_0^c/N = \frac{4}{q} - 2 - 2/\sqrt{q} .$$

The phase transition is second order for $q \leq 4$ and first order for $q \geq 5$.

Markov Chain Monte Carlo

A Markov chain allows to generate configurations k with probability

$$P_B^{(k)} = c_B \, w_B^{(k)} = c_B \, e^{-\beta E^{(k)}} , \qquad c_B \;\; \text{constant} .$$

The state vector $(P_B^{(k)})$, for which the configurations are the vector indices, is called Boltzmann state. A Markov chain is a simple dynamic process, which generates configuration k_{n+1} stochastically from configuration k_n. Let the transition probability to create the configuration l in one step from k be given by $W^{(l)(k)} = W[k \to l]$. Then, the transition matrix

$$W = \left(W^{(l)(k)} \right)$$

defines the Markov process. Note, that this matrix is a very big and never stored in the computer. The matrix achieves our goal and generates configurations with the desired probabilities, when it satisfies the following properties:

(i) **Ergodicity**:

$$e^{-\beta E^{(k)}} > 0 \quad \text{and} \quad e^{-\beta E^{(l)}} > 0 \quad \text{imply :}$$

an integer number $n > 0$ exists so that $(W^n)^{(l)(k)} > 0$ holds.

(ii) **Normalization**:

$$\sum_l W^{(l)(k)} = 1 \; .$$

(iii) **Balance**:

$$\sum_k W^{(l)(k)} \, e^{-\beta E^{(k)}} \;\; = \;\; e^{-\beta E^{(l)}} \; .$$

Balance means: The Boltzmann state is an eigenvector with eigenvalue 1 of the matrix $W = (W^{(l)(k)})$.

With that we have replaced the canonical ensemble average by a time average over an artificial dynamics and one distinguishes *dynamical universality classes*.

The Metropolis Algorithm

Detailed balance still does not uniquely fix the transition probabilities $W^{(l)(k)}$. The Metropolis algorithm is a popular choice can be used whenever one knows how to calculate the energy of a configuration. Given a configuration k, the Metropolis algorithm proposes a configuration l with probability

$$f(l,k) = f(k,l) \quad \text{normalized to} \quad \sum_l f(l,k) = 1 \ .$$

The new configuration l is accepted with probability

$$w^{(l)(k)} = \min\left[1, \frac{P_B^{(l)}}{P_B^{(k)}}\right] = \begin{cases} 1 & \text{for} \quad E^{(l)} < E^{(k)} \\ e^{-\beta(E^{(l)} - E^{(k)})} & \text{for} \quad E^{(l)} > E^{(k)}. \end{cases}$$

If the new configuration is rejected, the old configuration has to be counted again.

The Heatbath algorithm

The heat bath algorithm chooses a spin q_i directly with the local Boltzmann distribution defined by its nearest neighbors

$$P_B(q_i) \;=\; \text{const}\; e^{-\beta\, E(q_i)}\;.$$

As many Metropolis hits (on the same spin) are needed to reach this distribution, the heatbath is more efficient than the Metropolis algorithm. However, it models more complicated than Potts models the calculation of the local heatbath probabilities is often too involved to make it a viable alternative.

Start and equilibration

Initially we have to start with a microstate which may be far off the Boltzmann distribution. Although the weight of states decreases with $1/n$ where n is the steps of the Markov process, one should exclude the initial states from the equilibrium statistics. Many ways to generate start configurations exist, e.g.,

1. A random configuration corresponding to $\beta = 0$.

2. An ordered configuration for which all Potts spins take on the same q-value.

Examples: (assignments a0303_01 and a0303_05)

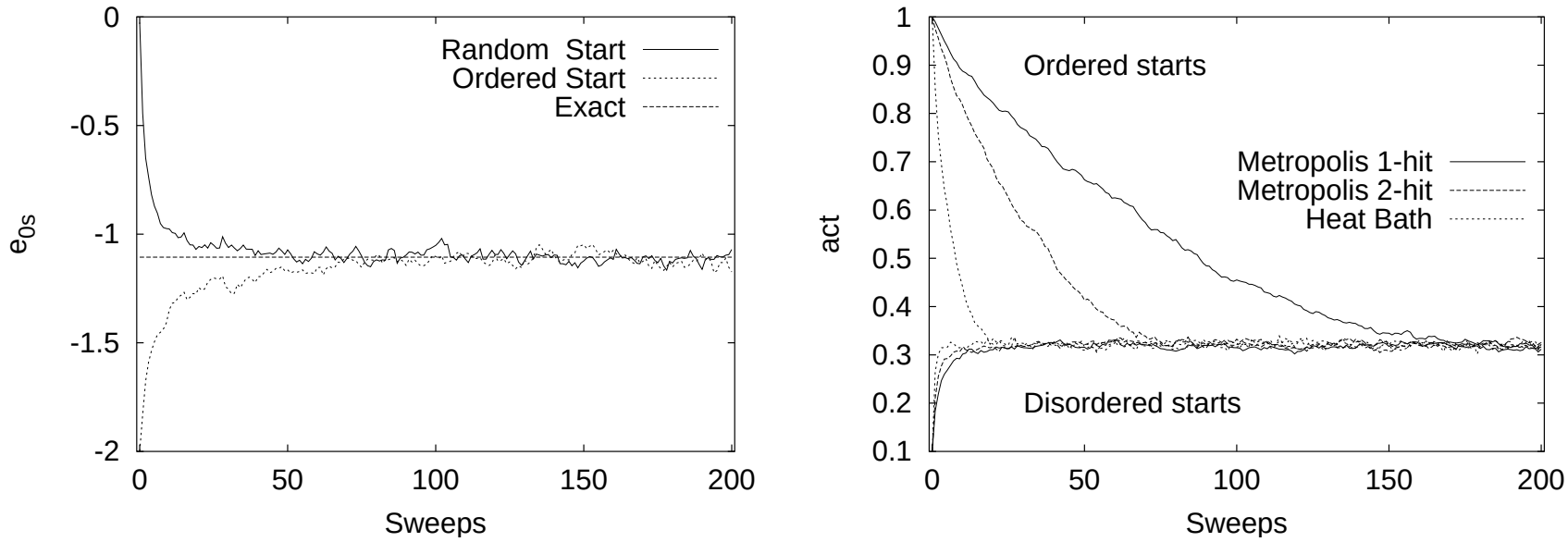


Figure 2: Initial time series of 200 sweeps each on a 80×80 lattice. Left: Metropolis for a $2d$ Ising model at $\beta = 0.4$. Right: $q = 10$ Potts model at $\beta = 0.62$.

Consistency Checks

For the $2d$ Ising model we can test against the exact finite lattice results of Ferdinand and Fisher. We simulate a 20^2 lattice at $\beta = 0.4$ using 10 000 sweeps for reaching equilibrium 64 bins of 5 000 sweeps for measurement (a careful justification is given later). We find (assignment a0303_06)

$$\overline{e}_{0s} = -1.1172\ (14)\ \ (\text{Metropolis}) \quad \text{versus} \quad \widehat{e}_s = -1.117834\ \ (\text{exact})\ .$$

The Gaussian difference test gives a perfectly admissible value $Q = 0.65$.

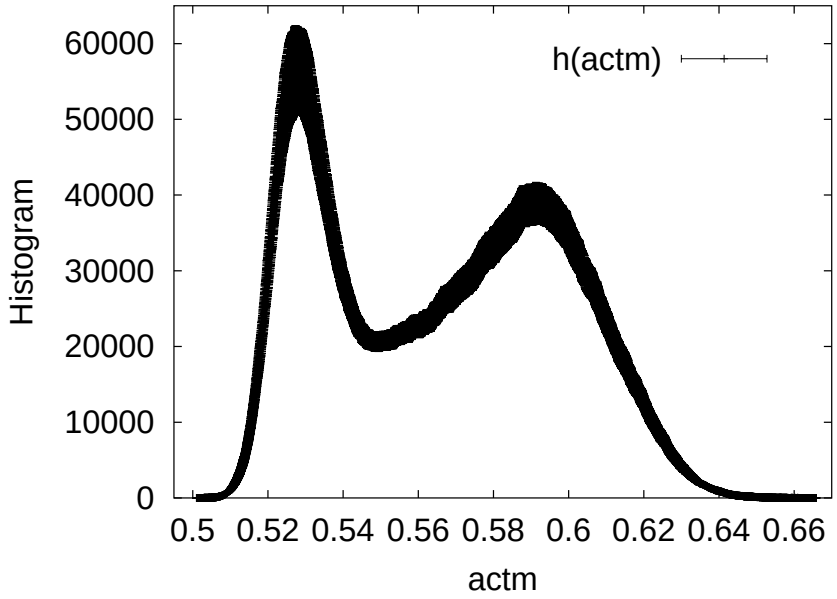
For the $2d$ 10-state Potts model at $\beta = 0.62$ we test our Metropolis versus our heat bath code on a 20×20 lattice to find (assignment a0303_08)

$$\texttt{actm} = 0.321772\ (75)\ \ (\text{Metropolis}) \ \ \text{versus} \quad \texttt{actm} = 0.321661\ (70)\ \ (\text{heat bath})$$

and $Q = 0.28$ for the Gaussian difference test. Another perfectly admissible value.

$3d$ 3-state Potts model

To illustrate features of a first order phase transition we simulate the $3d$ 3-state Potts model on a 24^3 lattice at a pseudo-transition temperature and plot its energy histogram. A characteristic double peak structure is found (assignment a0303_10):



Self-Averaging Illustration for the $2d$ Heisenberg model

We compare the peaked distribution function of a mean energy density per link for different lattice sizes. The property of **self-averaging** is observed: The larger the lattice, the smaller the confidence range. The other way round, the peaked distribution function is very well suited to exhibit observables for which self-averaging does not work, as for instance encountered in spin glass simulations.

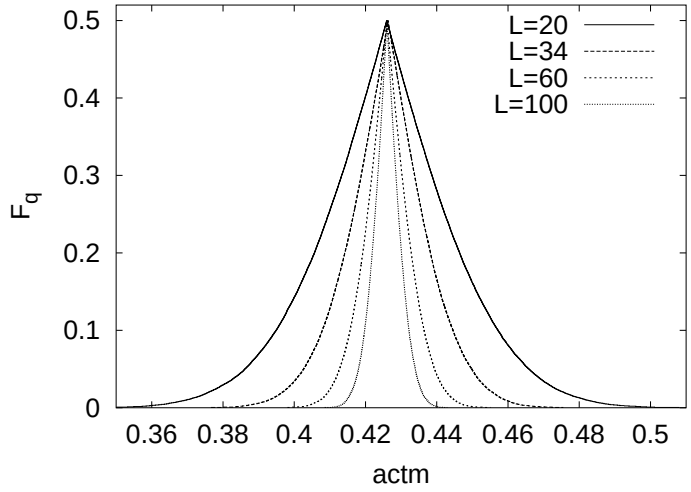


Figure 3: $O(3)$ σ-model at $\beta = 1.1$ (assignments a0304_06 and a0304_08).

Statistical Errors of Markov Chain MC Data

A typical MC simulation falls into two parts:

1. Equilibration without measurements.

2. Production with measurements.

Rule of thumb (for long calculations): **Do not spend more than 50% of your CPU time on measurements!**

Autocorrelations

We like to estimate the expectation value $\widehat{f}$ of some physical observable. We assume that the system has reached equilibrium. How many MC sweeps are needed to estimate $\widehat{f}$ with some desired accuracy? To answer this question, one has to understand the autocorrelations within the Markov chain.

Given is a time series of measurements f_i, $i = 1, \ldots, N$. With the notation $t = |i - j|$ the autocorrelation function of the mean $\widehat{f}$ is defined by

$$\widehat{C}(t) = \widehat{C}_{ij} = \langle \left(f_i - \langle f_i \rangle\right)\left(f_j - \langle f_j \rangle\right) \rangle = \langle f_i f_j \rangle - \langle f_i \rangle \langle f_j \rangle = \langle f_0 f_t \rangle - \widehat{f}^2$$

Some algebra shows that the variance of the estimator $\overline{f}$ for the mean and the autocorrelation functions are related by

$$\sigma^2(\overline{f}) \;=\; \frac{\sigma^2(f)}{N}\left[1 + 2\sum_{t=1}^{N-1}\left(1 - \frac{t}{N}\right)\widehat{c}(t)\right] \quad \text{with} \quad \widehat{c}(t) = \frac{\widehat{C}(t)}{\widehat{C}(0)}\,.$$

This equation ought to be compared with the corresponding equation for uncorrelated random variables $\sigma^2(\overline{f}) = \sigma^2(f)/N$. The difference is the factor in the bracket which defines the integrated autocorrelation time as

$$\tau_{\text{int}} \;=\; \lim_{N\to\infty}\left[1 + 2\sum_{t=1}^{N-1}\left(1 - \frac{t}{N}\right)\widehat{c}(t)\right]\,.$$

Self-consistent versus reasonable error analysis

The calculation of the integrated autocorrelation provides a self-consistent error analysis. But in practice this is often out of reach.

According to the Student distribution about twenty independent data are sufficient to estimate mean values reliably, while the error of the error bar the χ^2 distribution implies that about one thousand are needed for an estimate of the integrated autocorrelation time with 10% accuracy on the two σ confidence level.

In practice, one may rely on the binning method with a fixed number of ≥ 16 bins. How do we know then that the statistics has become large enough? There can be indirect arguments like finite size scale extrapolations, which suggest that the integrated autocorrelation time is (much) smaller than the achieved bin length. This is no longer self-consistent, but a reasonable error analysis.

Comparison of Markov chain MC algorithms

The $d = 2$ Ising model at the critical temperature

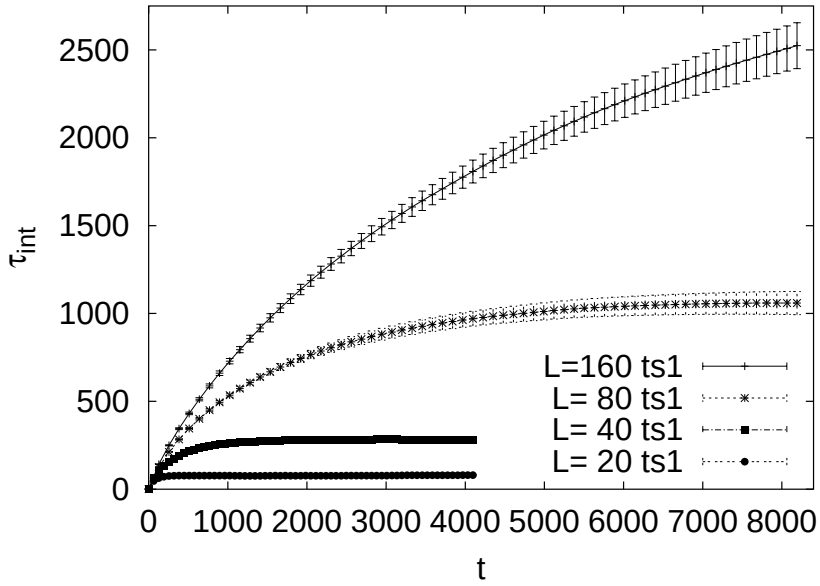


Figure 4: One-hit Metropolis algorithm with sequential updating: critical slowing down, $\tau_{\mathrm{int}} \sim L^z$ where $z \approx 2.17$ is the dynamical critical exponent (assignment a0402_02 D).

Another MC dynamics, Swendsen-Wang (SW) and Wolff (W) cluster algorithm:

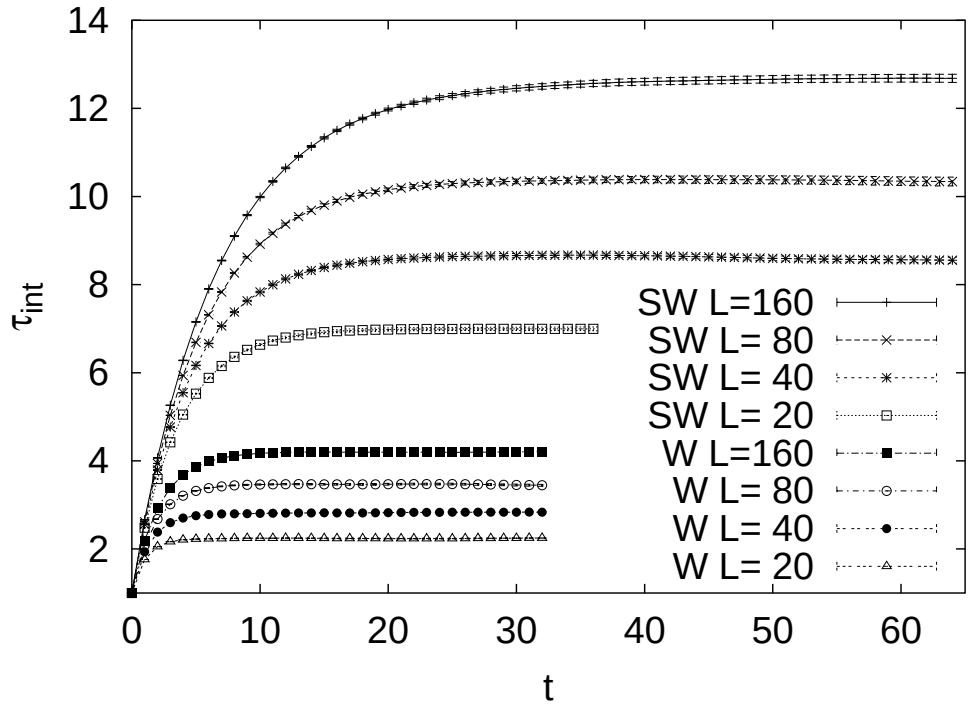


Figure 5: Estimates of integrated autocorrelation times from simulations of the $d = 2$ Ising model at the critical temperature (assignment a0503_05).

Simulations of the Multicanonical Ensemble

One of the questions which ought to be addressed before performing a large scale computer simulation is "What are suitable weight factors for the problem at hand?" So far we used the Boltzmann weights as this appears natural for simulating the Gibbs ensemble. However, a broader view of the issue is appropriate.

Conventional simulations can by re-weighting techniques only be extrapolated to a vicinity of the simulation temperature. For multicanonical simulations this is different. A single simulation allows to obtain equilibrium properties of the Gibbs ensemble over a range of temperatures. Of particular interest are two situations for which canonical simulations do not provide the appropriate implementation of importance sampling:

1. The physically important configurations are rare in the canonical ensemble.

2. A rugged free energy landscape makes the physically important configurations difficult to reach.

Multicanonical simulations sample, in an appropriate energy range, with an approximation to the weights

$$\widehat{w}_{mu}(k) = w_{mu}(E^{(k)}) = e^{-b(E^{(k)})\,E^{(k)}+a(E^{(k)})} = \frac{1}{n(E^{(k)})}$$

where $n(E)$ is the spectral density. The function $b(E)$ defines the inverse microcanonical temperature and $a(E)$ the dimensionless, microcanonical free energy. The function $b(E)$ has a relatively smooth dependence on its arguments, which makes it a useful quantity when dealing with the weight factors. The multicanonical method requires two steps:

1. Obtain a working estimate the weights. Working estimate means that the approximation has to be good enough so that the simulation covers the desired eneryg or temperature range.

2. Perform a Markov chain MC simulation with the fixed weights. The thus generated configurations constitute the multicanonical ensemble.

Re-Weighting to the Canonical Ensemble

Given the multicanonical time series, where $i = 1, \ldots, n$ labels the generated configurations. The formula

$$\overline{\mathcal{O}} = \frac{\sum_{i=1}^{n} \mathcal{O}^{(i)} \, \exp\left[-\beta \, E^{(i)} + b(E^{(i)}) \, E^{(i)} - a(E^{(i)})\right]}{\sum_{i=1}^{n} \exp\left[-\beta \, E^{(i)} + b(E^{(i)}) \, E^{(i)} - a(E^{(i)})\right]} \; .$$

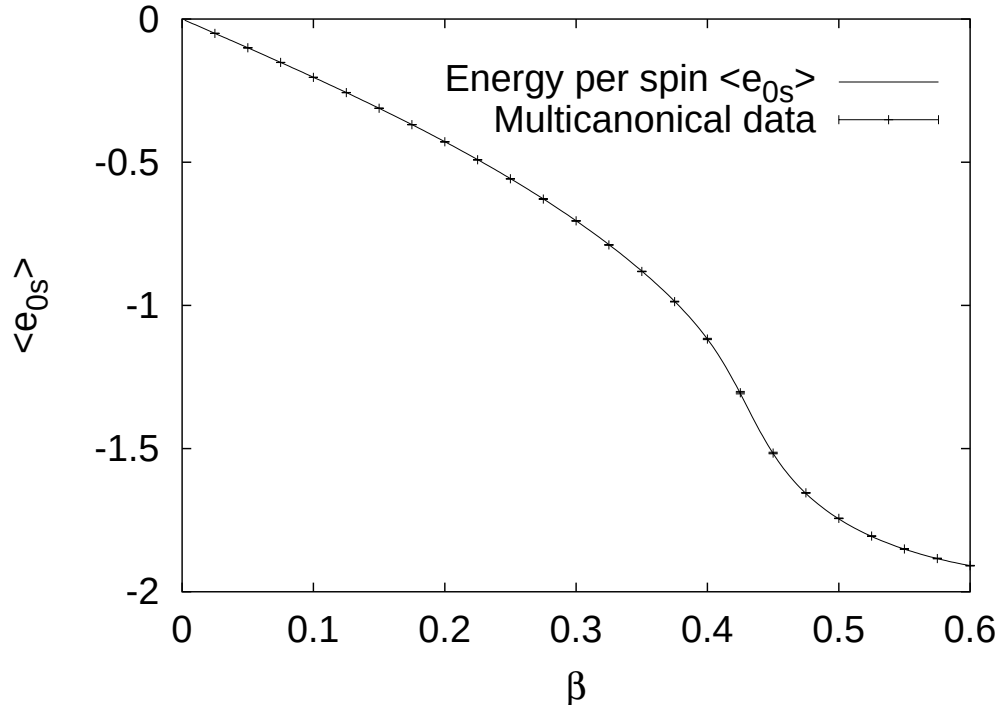
replaces the multicanonical weighting of the simulation by the Boltzmann factor. The denominator differs from the partition function Z by a constant factor which drops out (for discrete systems this simplifies for functions of the energy using histograms). The computer implementation of these equations requires care and a Jackknife analysis with logarithmic coding relying on the formula

$$\ln C = \max\left(\ln A, \ln B\right) + \ln\{1 + \exp\left[-|\ln A - \ln B|\right]\}$$

ought to be used.

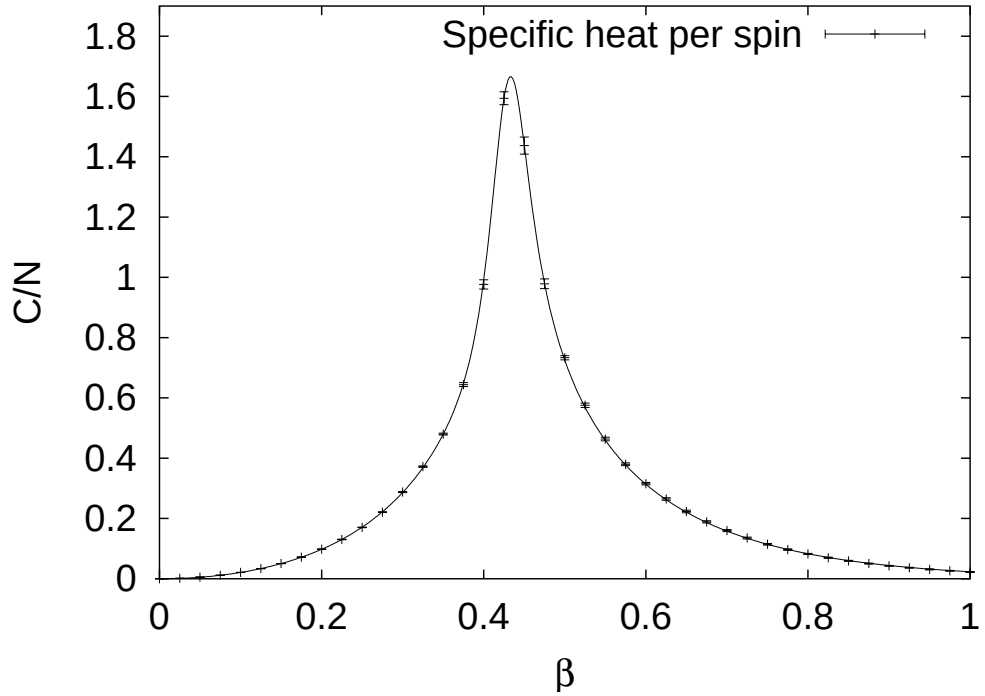
Energy and Specific Heat Calculation

Multicanonical data for the energy per spin (with jackknife errors) of the $2d$ Ising model on a 20×20 lattice are produced in assignment a0501_01 and compared with the exact results of Ferdinand and Fisher in assignment a0501_03. Results:



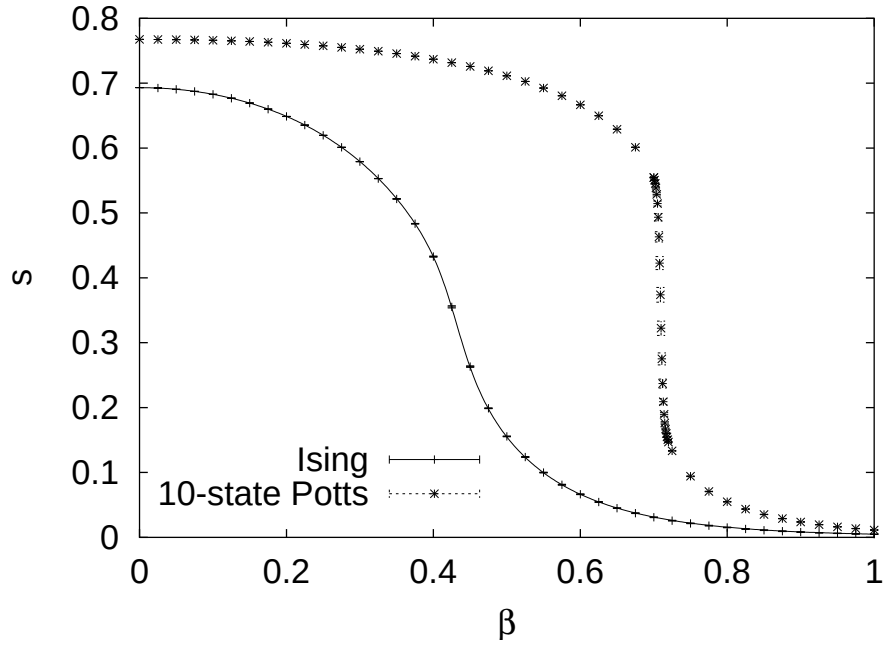
The same numerical data allow to calculate the **specific heat** defined by

$$C \;=\; \frac{d\,\widehat{E}}{d\,T} \;=\; \beta^2 \,\left(\langle E^2 \rangle - \langle E \rangle^2 \right) \;\; .$$



Free Energy and Entropy Calculation

At $\beta = 0$ the Potts partition function is $Z = q^N$. Therefore, multicanonical simulations allow for proper normalization of the partition function, if $\beta = 0$ is included in the temperature range. Example: Entropies from multicanonical simulations of the The 2d Ising and 10-state Potts models on a 20×20 lattice (assignment a0501_03 for Ising and a0501_05 for 10-state Potts).



Summary

- We considered Statistics, Markov Chain Monte Carlo simulations, the Statistical Analysis of Markov chain data and, finally, Multicanonical Sampling.

- It is a strength of computer simulations that one can generate artificial (not realized by nature) ensembles, which enhance the probabilities of rare events on may be interested in, or speed up the dynamics.

- Each method comes with its entire computer code.