Tutorial Lecture on Markov Chain Monte Carlo Simulations and Their Statistical Analysis

Bernd A. Berg

Florida State University

GBA Theoretical Chemistry Lecture Series, Boston, 11/29/2006

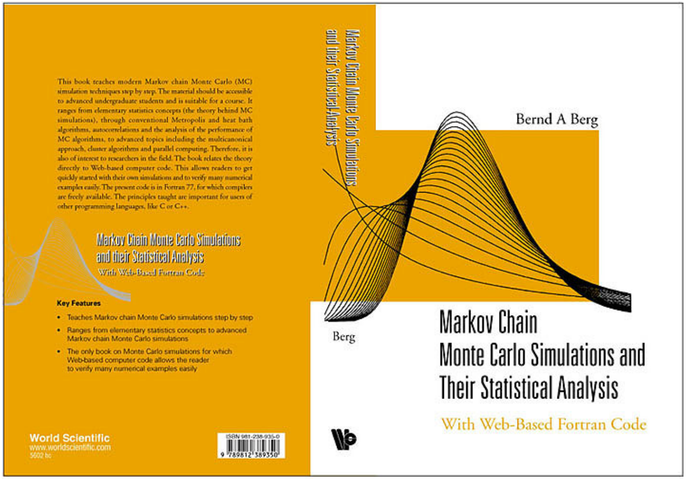
Overview and References:

1. Statistics as needed
2. Markov Chain Monte Carlo (MCMC)
3. Statistical Analysis of MCMC Data
4. Multicanonical Simulations

Markov Chain Monte Carlo Simulations and Their Statistical Analysis (With Web-Based Fortran Code), World Scientific 2004.

Link at www.hep.fsu.edu/˜ berg .

First chapter and Fortran code are freely available. All simulations of the book are number by number reproducable with the code provided (assignments in the book).



This book teaches modern Markov chain Monte Carlo (MC) simulation techniques step by step. The material should be accessible to advanced undergraduate students and is suitable for a course. It ranges from elementary statistics concepts (the theory behind MC simulations), through conventional Metropolis and heat bath algorithms, autocorrelations and the analysis of the performance of MC algorithms, to advanced topics including the multicanonical approach, cluster algorithms and parallel computing. Therefore, it is also of interest to researchers in the field. The book relates the theory directly to Web-based computer code. This allows readers to get quickly started with their own simulations and to verify many numerical examples easily. The present code is in Fortran 77, for which compilers are freely available. The principles taught are important for users of other programming languages, like C or C++.

Markov Chain Monte Carlo Simulations and their Statistical Analysis

With Web-Based Fortran Code

Key Features

- Teaches Markov chain Monte Carlo simulations step by step
- Ranges from elementary statistics concepts to advanced Markov chain Monte Carlo simulations
- The only book on Monte Carlo simulations for which Web-based computer code allows the reader to verify many numerical examples easily

World Scientific
www.worldscientific.com
5602 hc

ISBN 981-238-935-0
9 789812 389350

Markov Chain Monte Carlo Simulations and their Statistical Analysis

Berg

Bernd A Berg

Markov Chain
Monte Carlo Simulations and
Their Statistical Analysis

With Web-Based Fortran Code

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 $f(x) \ge 0$. 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 (CDF) of the random variable x^r is

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

Important is the uniform probability distribution on $[0,1)$,

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

Uniformly distributed random variables allow for the construction of general random variables: Let

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

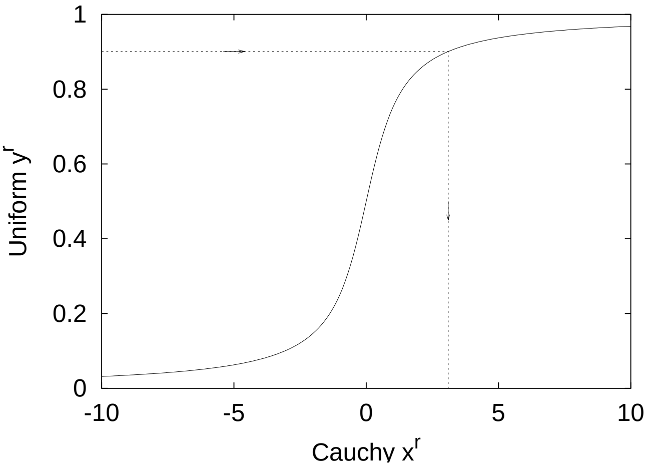
and y^r be uniform in $[0,1)$. The random variable

$$x^r = F^{-1}(y^r)$$

is then generated with the probability density $f(x)$.

Example Cauchy distribution:

$$f(x) = \frac{1}{\pi(1+x^2)}$$

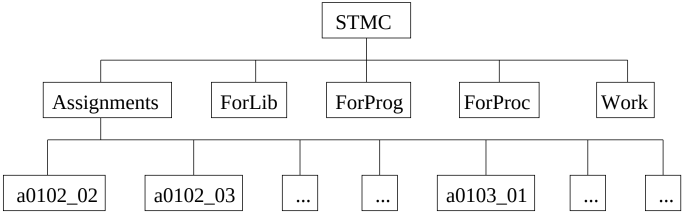


Pseudo Random Numbers and Computer Code

It is sufficient to generate uniform (pseudo) random numbers! Control your random number generator! The code of my book uses a portable generator design by Marsaglia and collaborators. How to get it? Download `STMC.tgz`, which unfolds under Linux by

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

into the directory structure shown below (Windows: Install cygwin).



Random Number Generator Routines:

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

`ranmar.f` returns one random number (function version `rmafun.f`).

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

A large number of initial seeds provide independent series.

Illustration (Assignment a0102_02):

```
CALL RMASET:
RANMAR INITIALIZED.
4 CALLS TO RANMAR:
idat = 1, xr = 0.116391063
idat = 2, xr = 0.964846790
idat = 3, xr = 0.882970393
idat = 4, xr = 0.420486867
CALL RMASAVE.
CALL RANMAR:
extra xr = 0.495856345
```

```
CALL RMASET:
MARSAGLIA CONTINUATION.
4 CALLS TO RANMAR:
idat = 1, xr = 0.495856345
idat = 2, xr = 0.577386141
idat = 3, xr = 0.942340136
idat = 4, xr = 0.243162394
CALL RAMSAVE.
CALL RANMAR:
extra xr = 0.550126791
```

Confidence Intervals and Sorting

One defines q-tiles (also quantiles or fractiles) x_q of a CDF by

$$F(x_q) \;=\; q\;. \qquad \text{An example is the median } x_{\frac{1}{2}}\;.$$

The probability content of the confidence interval

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

Example: Gaussian (normal) distribution of variance σ^2 :

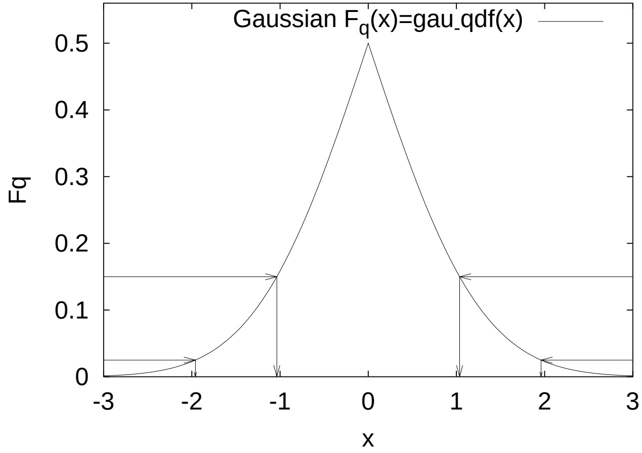
$$[-n\sigma, +n\sigma] \;\Rightarrow\; p = 0.6827 \text{ for } n = 1, \quad p = 0.9545 \text{ for } n = 2\;.$$

The peaked cumulative distribution function (PCDF)

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

provides a convenient visualization of probability intervals.

Gaussian PCDF:



Sorting allows for an estimate of a CDF from data $x_1, ..., x_n$: We may re-arrange the x_i in increasing order:

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

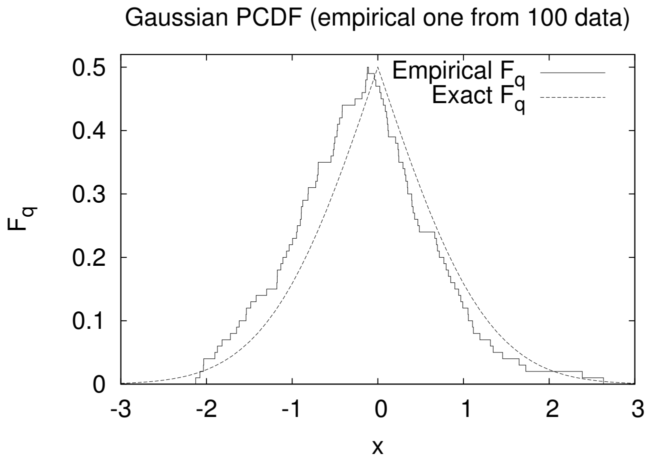
where $\pi_1, \dots, \pi_n$ is a permutation of $1, \dots, n$.
An estimator for the CDF $F(x)$ is then the empirical CDF

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

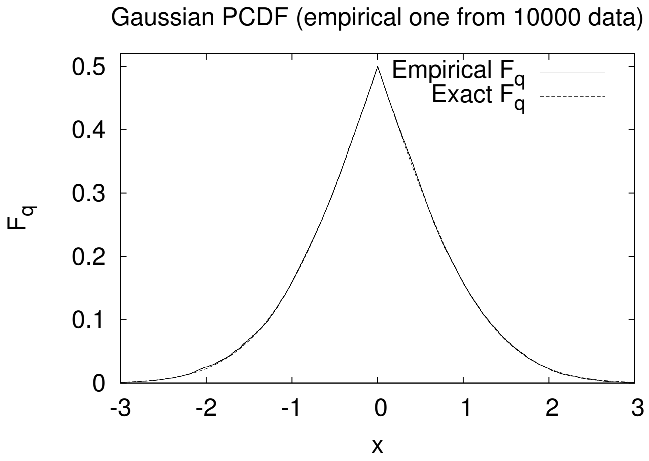
with the definitions $x_{\pi_0} = -\infty$ and $x_{\pi_{n+1}} = +\infty$. To calculate $\overline{F}(x)$ one needs an efficient way to sort n data values in ascending (or descending) order. In the STMC package this is provided by a heapsort routine, which needs $O(n \log_2 n)$ steps.

Example: Gaussian distribution in assignment a0106_02.

Empirical versus exact PCDF:



Empirical versus exact PCDF:



Quantitative Science!

Binning (Blocking)

Central Limit Theorem: Convergence of the Sample Mean

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

We block NDAT data into NBINS bins, where each binned data point is the arithmetic average of NBIN $=$ [NDAT/NBINS] sequential original data points. The purpose of the binning procedure is twofold:

1. The binned data will become practically Gaussian when NBIN is large enough. Gaussian error analysis applies even when the original are not Gaussian.
2. Data generated by a MCMC process are autocorrelated. For binned data the autocorrelations are reduced and can be neglected, once NBIN is large enough.

How Many Bins (Blocks)?

Gaussian Error Analysis for Small Samples.

Gosset's Student Distribution provides confidence levels for standard deviations $s_{\overline{x}}$ estimated from Gaussian samples of size N :

	$s_{\overline{x}} = 1$	$s_{\overline{x}} = 2$	$s_{\overline{x}} = 3$	$s_{\overline{x}} = 4$	$s_{\overline{x}} = 5$
$N = 2$	0.50000	0.70483	0.79517	0.84404	0.87433
$N = 4$	0.60900	0.86067	0.94233	0.97199	0.98461
$N = 8$	0.64938	0.91438	0.98006	0.99481	0.99843
$N = 16$	0.66683	0.93605	0.99103	0.99884	0.99984
$N = 32$	0.67495	0.94567	0.99471	0.99963	0.99998
$N = 64$	0.67886	0.95018	0.99614	0.99983	1.00000
$N = \infty$	0.68269	0.95450	0.99730	0.99994	1.00000

Conclusion: 32 bins are sufficient at two sigma (but not if one is interested in rare outliers at four sigma).

Statistical Physics and MCMC Simulations

We aim now at calculating estimators of physical observables $\mathcal{O}$ in equilibrium at temperature T in the Gibbs canonical ensemble. On a computer all systems are discrete. Hence ($\beta = 1/T$),

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

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

where the sums are over all configurations (microstates) of the system, $E^{(k)}$ is the (internal) energy of configuration k, and Z the partition function.

In the following we use Potts models on d-dimensional hypercubic lattices with periodic boundary conditions to ilustrate MCMC simulations. While simple, these models allow to exemplify the essential features in which we are interested.

Potts Models

Their energy is defined by

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

The sum $\langle ij \rangle$ is over the nearest lattice sites and the Potts spins $q_i^{(k)}$ take the values $1, 2, \dots, q$. Our normalization is chosen so that the energy per spin $e_s = E/N$ agrees for $q = 2$ with the conventional Ising model definition. For the $2d$ Potts models a number of exact results are known for phase transitions at

$$\beta_c = \frac{1}{2} \ln(1 + \sqrt{q}), \qquad e_{s,c} = E_c/N = \frac{4}{q} - 2 - 2/\sqrt{q}\ .$$

The transitions are second order for $q \leq 4$ and first order for $q \geq 5$. Therefore, they make a good testing laboratory for investigations of phase transitions by MCMC methods.

Markov Chain Monte Carlo

Why? Random sampling ($\beta = 0$) does not sample the configurations, which are important at lower temperatures, while a Markov chain allows to generate configurations k with probability

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

The vector $P_B = (P_B^{(k)})$ is called Boltzmann state. A Markov chain is a dynamic process, which generates configuration k_{n+1} stochastically from configuration k_n. The transition matrix

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

where $W^{(l)(k)} = W[k \to l]$ is the probability to create configuration l from k in one step, defines the Markov process. Note, that this matrix is a very big and never stored in the computer.

The transition matrix generates configurations asymptotically with the Boltzmann probabilities, when it satisfies the following properties:

(i) Ergodicity:

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

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

(ii) Normalization:

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

(iii) Balance: The Boltzmann state is an eigenvector with eigenvalue 1 of the transfer matrix

$$W\, P_B = P_B \quad \text{or} \quad \sum_k W^{(l)(k)}\, e^{-\beta E^{(k)}} \ = \ e^{-\beta E^{(l)}} \ .$$

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

The Metropolis Algorithm

Nicolas Metropolis, Arianna and Marshall Rosenbluth, Augusta and Edward Teller, 1953 – AIP Conference Proceedings, Vol. 690, 2003.

Balance does constrain but not fix the transition probabilities $W^{(l)(k)}$. The Metropolis algorithm can be used whenever one knows how to calculate the energy of a configuration. Given a configuration k, a configuration l is proposed with some probability

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

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 l is rejected, k is counted again.

The Heatbath algorithm (Gibbs sampler)

Glauber 1964, Creutz 1980; (Geman and Geman 1984).

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

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

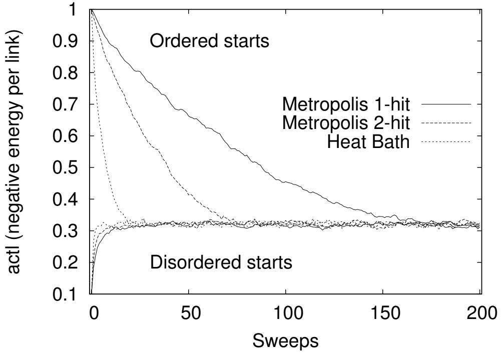
Several Metropolis hits (on the same spin) are needed to reach this distribution. Therefore, one heatbath update is more efficient than one Metropolis update. But the calculation of the local heatbath probabilities is often too CPU time consuming to make it a viable alternative.

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 $\propto 1/n$ with the number of steps n, and is finally swallowed by the error bar $\propto 1/\sqrt{n}$, factors can be large and 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.

Example: Initial time series of 200 sweeps on a 80×80 lattice for the $q = 10$ Potts model at $\beta = 0.62$ (assignment a0303_05). A sweep is defined by updating each variable on the lattice once or once in the average.



Consistency Checks

2*d* Ising model: Exact finite lattice results of Ferdinand and Fisher. Simulation on a 20^2 lattice at $\beta = 0.4$ using 10 000 sweeps for reaching equilibrium and 64 bins of 5 000 sweeps for measurement gives (assignment a0303_06):

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

Gaussian difference test: $Q = 0.64$ (consistent).

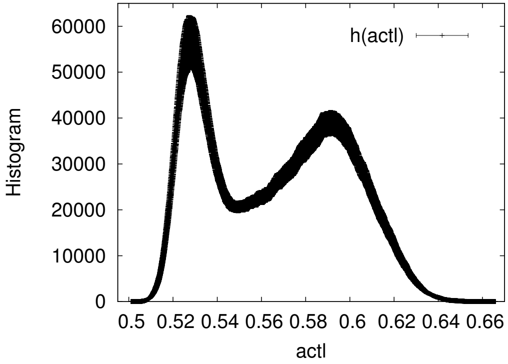
2*d* 10-state Potts model on a 20×20 lattice at $\beta = 0.62$: Metropolis (Met) versus heatbath (HB) updating gives (assignment a0303_08)

$$\texttt{actm} = 0.321772\,(75)\ \ (\text{Met})\quad \text{versus}\quad \texttt{actm} = 0.321661\,(70)\ \ (\text{HB})\,.$$

$Q = 0.28$ for the Gaussian difference test (consistent).

Observation of a first order phase transition

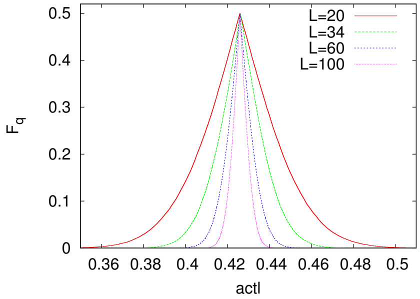
To illustrate features of a first order phase transition we simulate the $3d$ 3-state Potts model on a 24^3 lattice at a pseudo-transition temperature and plot its energy histogram. A characteristic double peak structure is found (assignment a0303_10, takes about 17.5 minutes on a 2.8GHz PC):



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

$$E = -\sum_{\langle ij \rangle} \vec{s}_i \cdot \vec{s}_j \quad \text{with } 3d \text{ unit vectors } \vec{s}_i^{\,2} = 1 \ .$$

Self-averaging: The larger the lattice, the smaller the confidence range found from the PCDF of the energy distribution (a0304_08):



The other way round, the PCDF is well suited to exhibit observables for which self-averaging does not work (e.g., in spin glasses).

Statistical Errors of Markov Chain MC Data

A typical MC simulation falls into two parts:

1. Equilibration without measurements.
2. Production with measurements.

If your measurements are CPU time extensive:

Spend 50% of your CPU time on measurements!

Autocorrelations

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

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

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

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 factor in the bracket defines the integrated autocorrelation time

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

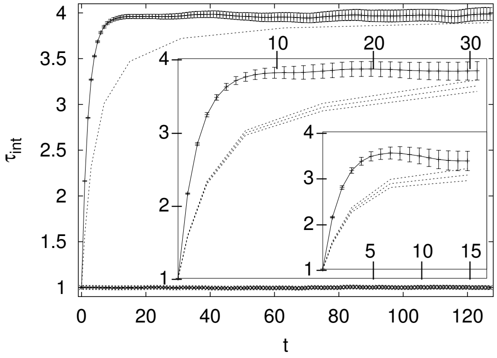
But: The variance of $\tau_{\rm int}$ divergent! Solution: Window method.

$$\tau_{\rm int}(t) \;=\; 1 + 2\sum_{t'=1}^{t}\widehat{c}(t')\;.$$

Take the estimate as soon as this function becomes flat.
Alternative: Blocking method $\tau_{\rm int} = \sigma^2_{\rm NBIN\to\infty}/\sigma^2_{\rm NBIN=1}$.

Example:

Metropolis Generation of Gaussian random numbers. Estimates from the blocking method are also shown. Statistics from up to down: $2^{21} = 2,097,152$, $2^{17} = 131,072$ and $2^{14} = 16,384$ updates (assignment a0401_02).



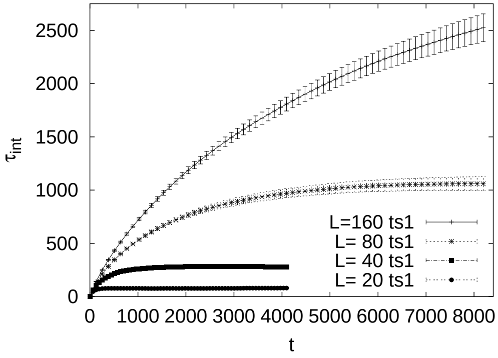
Self-consistent versus reasonable error analysis

Calculation of the integrated autocorrelation time provides a self-consistent error analysis. In practice this is often out of reach: About twenty independent data are sufficient to estimate mean values reliably (Student distribution), while one thousand are needed for an estimate of the integrated autocorrelation time (10% accuracy on the 2σ confidence level as implied by the χ^2 distribution for error bars of Gaussian data: Variance ratio F-test).

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 scaling or other extrapolations of the integrated autocorrelation time. This is no longer a self-consistent, but a reasonable error analysis.

Comparison of Markov chain MC algorithms

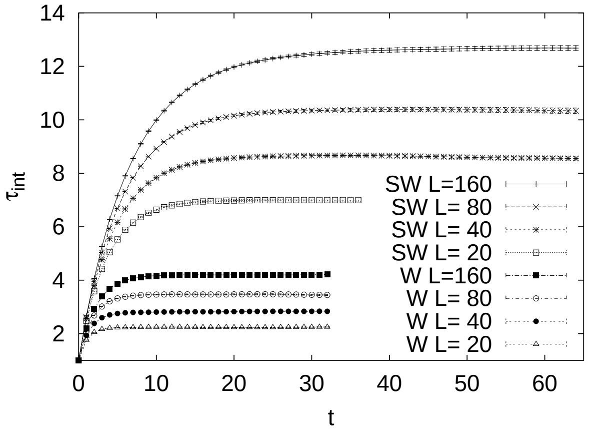
The $d = 2$ Ising model at the critical temperature



One-hit Metropolis algorithm with sequential updating: critical slowing down, $\tau_{\rm int} \sim L^z$ where $z \approx 2$ is the dynamical critical exponent (assignment a0402_02 D).

Another MC dynamics:

Swendsen-Wang (SW) and Wolff (W) cluster algorithms



(Assignment a0503_05)

Multicanonical Simulations

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. In contrast, a single multicanonical 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 number of states with energy E. 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.

How to get the Weights?

To get the weights is at the heart of the method. Some approaches used are listed in the following:

1. Overlapping constrained (microcanonical) MC simulations. Potential problem: ergodicity.
2. Finite Size Scaling (FSS) Estimates. Best when it works! Problem: There may be no FSS theory or the prediction may be so inaccurate that the initial simulation will not cover the target region.
3. General Purpose Recursions. Alternatives: Multicanonical recursion (animation), Wang-Landau recursion, ...

Example Run

Ising model on a 20×20 lattice: The multicanonical recursion is run in the range

$$\texttt{namin} = 400 \leq \texttt{iact} \leq 800 = \texttt{namax}\ .$$

The recursion is terminated after a number of random walk cycles. events. A random walk cycle (also tunneling event) is defined as an updating process which finds its way from

$$\texttt{iact} = \texttt{namin} \quad \text{to} \quad \texttt{iact} = \texttt{namax} \quad \text{and} \quad \text{back}\ .$$

For most applications ten cycling events lead to acceptable weights. For an example run of the Ising model we find the requested 10 random walk cycles after 787 multicanonical recursions and 64,138 sweeps. The subsequent production run of $32 \times 10,000$ sweeps gives 84 random walk cycles (assignment a0501_01).

Reweighting to the Canonical Ensemble

Given the multicanonical time series, where $i = 1, \dots, n$ 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 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.

Error Bars of Nonlinear Functions

$$\widehat{f} = \widehat{f}(x_1, \dots, x_N)\,, \quad \text{biased estimator :} \quad \langle \overline{f} \rangle \neq \widehat{f}\ .$$

Use the Jackknife Method!

Allows to correct for the bias and the error of the bias. Jackknife estimators simply regroup the bins:

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

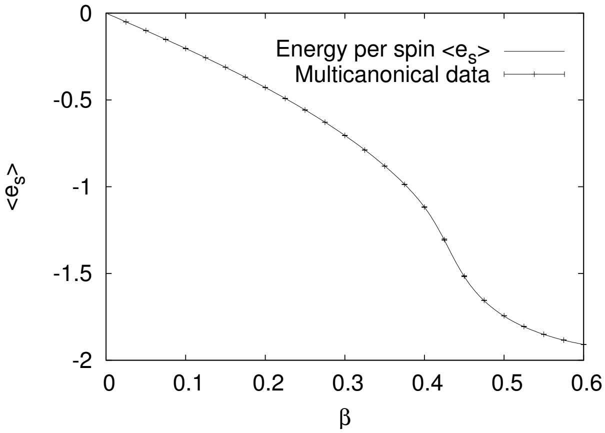
The estimator for the variance is

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

Straightforward algebra shows that in the unbiased case the jackknife variance reduces to the normal variance. Of order N operations are needed to construct the jackknife bins x_i^J. The extra effort is minimal. Jackknife should be the standard of error analysis!

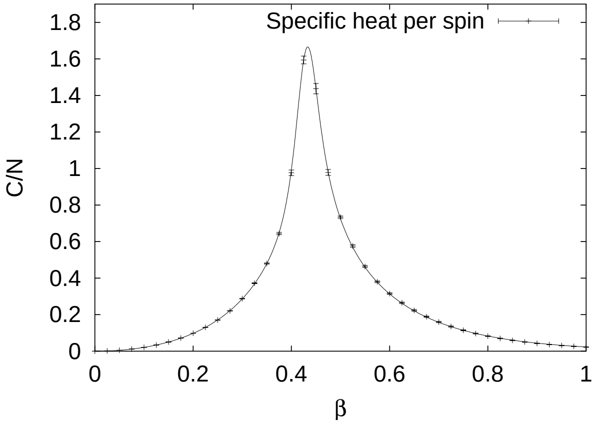
Energy and Specific Heat Calculation

We are now ready to produce multicanonical data for the energy per spin (with jackknife errors) of the $2d$ Ising model on a 20×20 lattice and compared them with the exact results of Ferdinand and Fisher (assignment a0501_03):



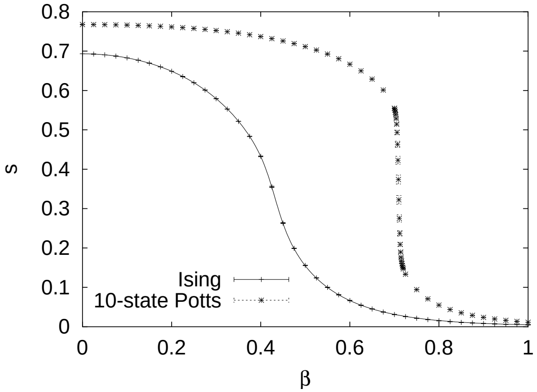
The same numerical data allow to calculate the specific heat

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



Free Energy and Entropy Calculation

At $\beta = 0$ the Potts partition function is $Z = q^N$. Multicanonical simulations allow for normalization of the partition function, if the temperature range includes $\beta = 0$. Example: Entropy of the 2d Ising and 10-state Potts models on a 20×20 lattice (assignment `a0501_03` for Ising; `a0501_02` and `a0501_05` for 10-state Potts).



Summary and Conclusions

- We considered Statistics, Markov Chain Monte Carlo simulations, the Statistical Analysis of Markov chain data and, finally, Multicanonical Sampling.
- Each method comes with its entire source code. Base provided on which improvements can be build and documented. No culture of improving code? Often encountered extremes: Bad code, too specialized (assembler) code.
- Standard for MCMC simulations of the book: Reproducability! Also for computational papers!?
- It is a strength of computer simulations that one can generate artificial ensembles (not realized by nature), which enhance the probabilities of rare events one may be interested in, or speed up the dynamics.