Configurationally-Biased Monte Carlo

and

Virtual-move parallel tempering

Ivan Coluzza & Daan Frenkel

Recent Advances in Monte Carlo Based Inference

Cambridge, 3-11-2006
1. The language:

Monte Carlo sampling in Statistical Physics

The aim is to estimate averages of the type:

\[
\langle A \rangle = \frac{\int d\mathbf{r}^N A(\mathbf{r}^N) \exp[-\beta \mathcal{U}(\mathbf{r}^N)]}{\int d\mathbf{r}^N \exp[-\beta \mathcal{U}(\mathbf{r}^N)]}
\]

Observable

Probability distribution
Markov-Chain Monte Carlo: points in configuration space \((r^N)\) are visited with a frequency \(N_i\) proportional to their (Boltzmann) weight \(\exp(-\beta U(r^N))\)

Then

\[
\langle A \rangle \approx \frac{1}{L} \sum_{i=1}^{L} N_i A(r_i^N).
\]
MARKOV CHAIN:

The transition matrix $\pi$ leaves the equilibrium distribution $\mathcal{N}$ unchanged.

$$\sum_{m} \mathcal{N}(m) \pi_{mn} = \mathcal{N}(n)$$
\[ N(o) \sum_n \pi(o \rightarrow n) = \sum_n N(n) \pi(n \rightarrow o) \]

Stronger condition (detailed balance):

\[ N(o) \pi(o \rightarrow n) = N(n) \pi(n \rightarrow o). \]
With the transition probabilities

\[
\pi(o \rightarrow n) = \alpha(o \rightarrow n) \times \text{acc}(o \rightarrow n)
\]

detailed balance implies that:

\[
\mathcal{N}(o)\alpha(o \rightarrow n) \times \text{acc}(o \rightarrow n) = \mathcal{N}(n)\alpha(n \rightarrow o) \times \text{acc}(n \rightarrow o)
\]
Often, we choose

$$\alpha(o \rightarrow n) = \alpha(n \rightarrow o)$$

Then it follows that

$$\mathcal{N}(o) \times \text{acc}(o \rightarrow n)$$

$$= \mathcal{N}(n) \times \text{acc}(n \rightarrow o)$$
\[
\frac{\text{acc}(o \rightarrow n)}{\text{acc}(n \rightarrow o)} = \frac{\mathcal{N}(n)}{\mathcal{N}(o)} = \exp\{-\beta[\mathcal{U}(n) - \mathcal{U}(o)]\}
\]

Metropolis, Rosenbluth, Rosenbluth, Teller and Teller choice:

\[
\text{acc}(o \rightarrow n) = \min\left(1, \exp\{-\beta[\mathcal{U}(r^N) - \mathcal{U}(r^N)]\}\right)
\]
Solution (?): play with the a-priori probabilities of trial moves:

\[ \alpha(o \rightarrow n) \neq \alpha(n \rightarrow o) \]

\[
\frac{\text{acc}(o \rightarrow n)}{\text{acc}(n \rightarrow o)} = \frac{\alpha(n \rightarrow o)}{\alpha(o \rightarrow n)} \exp\left\{-\beta[U(n) - U(o)]\right\}.
\]

In particular, if:

\[
\frac{\alpha(n \rightarrow o)}{\alpha(o \rightarrow n)} = \exp\left\{-\beta[U(o) - U(n)]\right\}.
\]

Then

\[
\frac{\text{acc}(o \rightarrow n)}{\text{acc}(n \rightarrow o)} = 1 \quad (100\% \text{ acceptance})
\]

…if we know how to do it.
Often one can indeed improve acceptance by biasing.

Special case:

Trial moves that consist of a sequence of intermediate steps.

Examples: changing polymer conformations, moving groups of atoms, ...
What is the problem with polymer simulations?

**Illustration:**

Inserting particles in a dense liquid

Trial moves that lead to “hard-core” overlaps tend to be rejected.
Next: consider the random insertion of a chain molecule (polymer).

Random insertions of polymers in dense liquids usually fail completely…
Solution:

Insert a polymer segment-by-segment, using several trial segments at every step.

There are many algorithms that do this. To compare them, we look at all possible “insertion trees”.
A “tree” of possible polymer conformations includes all possible branches. In general: $k$ branches start at every vertex.
But many branches terminate (due to hard-core overlaps)

Insertion algorithms aim to identify acceptable branches on this tree, without enumerating all possibilities…
Now we can compare what different algorithms do:

1. Conventional (unbiased) Monte Carlo
2. Configurational Bias Monte Carlo (CBMC)
3. Dynamic Pruning-Enrichment Rosenbluth Monte Carlo
4. Recoil-growth
5. …and others

None of these methods explores the complete tree of trial directions…

…but some, more than others.

J.I. Siepmann, J.J. de Pablo, B. Smit, M.W.Deem, G. Mooij, J.T.H.Vlugt, S.Consta, N. Wilding, N. Combe, P.R. ten Wolde, ...
Conventional Monte Carlo generates only 1 trial conformation

CHEAP...but the probability that a trial conformation of length $n$ is free of hard-core overlaps, scales as

$$P_{gen} \sim (1 - p_{overlap})^n$$

Hopeless, for all but the shortest chains.
**Configurational-Bias MC:** an off-lattice, dynamic version of the static, on-lattice Rosenbluth-Rosenbluth scheme.

Approach: Look one step ahead.

Probability to select one of the $k$ branches:

$$P_i = \frac{\exp(-\beta u_i)}{\sum_{j=1}^{k} \exp(-\beta u_j)}$$

This biases the selection of trial conformation: you never select a segment that “dies” immediately.
We correct for the bias in the acceptance step

Define the “Rosenbluth weight” \((w)\) of the old (new) conformations of a chain of length \(n\):

\[
\begin{align*}
  w^{(o)} &= \prod_{\ell=1}^{n} \left( \sum_{j=1}^{k} \exp(-\beta u^{(o)}(j; \ell)) \right) \\
  \end{align*}
\]

The trial conformation is accepted with a probability:

\[
P_{acc} = \min\{1, w^{(n)} / w^{(o)}\}
\]

(I skip the demonstration that this is correct)
Problems:

1. Chain generation can easily terminate in a dead alley
2. Much computational effort may be wasted in the generation of conformations with a low Rosenbluth weight

More recent developments
**DPERM** (population-based algorithm)

(Dynamic **Pruning-Enrichment** Rosenbluth Monte Carlo)

- based on a static MC scheme due to Grassberger et al. -

1. **Pruning**

Grow trial conformations as in Configurational Bias MC. For every length \( m \), compute the **partial** Rosenbluth weight \( w(m) \):

\[
    w(m) = w(m - 1) \times \frac{\sum_{j=1}^{k} \exp(-\beta u^{(n)}(j; \ell))}{k}
\]

If \( w(m) < w_{\text{min}} \) then, with 50% probability, **kill** conformation, otherwise, **double** \( w(m) \).
2. Enrichment:

If \( w(m) > w_{\text{max}} \) then: make \( r \) copies of the partial conformation and give each a (partial) Rosenbluth weight \( w(m)/r \).

At the end, several chain conformations may survive.

The move is accepted with a probability proportional to the total Rosenbluth weight of the surviving chains.

This scheme tends to eliminate low Rosenbluth weights, and is less sensitive to dead-alleys.
Recoil-growth:
A bit like CBMC, but the scheme looks n steps ahead (here n=2).

First trial chains end in dead alley
Recoil and regrow. Success!

Finally: Check n steps along the unexplored conformations
Intermediate summary:

Biased sampling of “composite” MC moves increases acceptance (often by dozens of orders of magnitude), but it does not lead to 100% acceptance.
But rejection-free algorithms do exist
(although, to my knowledge, not for polymers).

Old example: Swendsen-Wang cluster algorithm
Illustration: 2D Ising model ("spins" on a 2D (square) lattice)
Snapshot: some neighbors are parallel, others anti-parallel
1. Draw lines to connect parallel nearest-neighbor spins
2. Define “bonds” between parallel neighbors. The probability to have a bond (red lines) between parallel neighbors is $p$ (as yet undetermined). With a probability $1-p$, parallel neighbors are not connected (blue dashed line).
Form clusters of all spins that are connected by bonds. Some clusters are all “spin up” others are all “spin down”.

Denote the number of clusters by $M$. 
Now randomly flip clusters. This yields a new cluster configuration with probability $P_{\text{flip}} = (1/2)^M$.

Then reconnect parallel spins.
Next: forget about the “bonds”…
New spin configuration!

For a special (but known) choice of the bond probability $p$, we obtain 100% acceptance!
For Swendsen-Wang, we generate very many trial states: with $n$ clusters, $2^n$ possible states…

…and yet we accept only one!
Include “rejected” trial moves in the sampling
Recall: key property of MCMC

\[ \sum_{m} \rho(m) \pi_{mn} = \rho(n) \]

The transition matrix \( \pi \) leaves the equilibrium distribution \( \rho \) unchanged.
\[ \langle A \rangle_\rho = \sum_n A_n \rho_n \]

This, we can rewrite as:

\[ \sum_n A_n \rho_n = \sum_n \sum_m A_n \rho_m \pi_{mn} = \sum_m \rho_m \sum_n A_n \pi_{mn} \]

\[ = \sum_m \rho_m \sum_n A_n \pi_{mn} \leftrightarrow \langle A \rangle_\rho = \left\langle \sum_n \pi_{mn} A_n \right\rangle_{\rho_m} \]
Note that $\langle A \rangle$ is no longer an average over “visited” states – we also include “rejected” moves in the sampling.

\[
\langle A \rangle_{\rho} = \left\langle \sum_{n} \pi_{mn} A_{n} \right\rangle_{\rho_{m}}
\]
\[ \langle A \rangle_\rho = \left\langle \sum_n \pi_{mn} A_n \right\rangle_{\rho_m} \]

This relation also holds for any set of “connected” trial states: i.e. the possible final states of a single (decent) MC trial move.
For instance: in conventional MC, there would be only two states (the “old” state and the “new” state)

\[
\langle A \rangle_\rho = \left\langle \sum_n \pi_{mn} A_n \right\rangle_{\rho_m}
\]
\[ \langle A \rangle_{\rho} = \left\langle \sum_n \pi_{mn} A_n \right\rangle_{\rho_m} \]

But in other algorithms, there are many. e.g. in the Swendsen-Wang algorithm: \( n \) clusters that can be flipped \( \Rightarrow 2^n \) connected states.

The more parallel the algorithm, the better…
Example.

Compute probability to observe a macrostate with total spin $S$

Compare:

1. Normal (Swendsen-Wang) MC (sample one out of $2^n$ states)

2. Idem + “waste recycling” (sample all $2^n$ states)
Consider the total "magnetization" $S$. This is equal to the sum of the cluster spins:

$$S = \sum_{i=1}^{n} s_i$$

We can compute the characteristic function $f(k)$ defined as

$$f(k) = \langle \exp(ikS) \rangle = \int dS \ P(S)e^{ikS}$$

The angular brackets denote thermal averaging.

$P(S)$ is the probability distribution of the total magnetization.
Using the fact that
\[ \exp(i k S) = \prod_{i=1}^{n} \exp(i k s_i) , \]
we can now sum over all $2^n$ cluster flips to obtain a sub-move average for the total cluster move $m$. We denote this sub-move average by $f_m$:
\[ f_m = \prod_{i=1}^{n} \cos(k s_i) \]
Our estimate for $f(k)$ is then

$$f_{est}(k) = \frac{1}{M} \sum_{m=1}^{M} f_m$$

This quantity we can average in a normal (SW) Monte Carlo run and then Fourier transform back to get the probability distribution $P(S)$. 
Swendsen-Wang

Waste-recycling MC
Note that the transition matrix that is used in the averaging need not be the same as the one used in sampling the $\rho_m$.

(e.g. one could be “Barker” and the other “Metropolis”)
\[ \langle A \rangle_\rho = \left\langle \sum_n \pi_{mn} A_n \right\rangle_{\rho_m} \]

How to sample this?

\[ \pi_{nm} = \alpha_{nm} P_{acc}(nm) \]

\[ \langle A \rangle_\rho = \sum_m \rho_m \left\langle \sum_n P_{acc}(mn) A_n \right\rangle_\alpha \]
For example: Barker (symmetric) rule

\[ P_{acc}(nm) = \frac{e^{-\beta U_m}}{e^{-\beta U_n} + e^{-\beta U_m}} \]

\[ \langle A \rangle_\rho = \sum_m \rho_m \left\langle \frac{e^{-\beta U_n} A_n + e^{-\beta U_m} A_m}{e^{-\beta U_n} + e^{-\beta U_m}} \right\rangle_\alpha \]

(gives exact result in one step, for two-level system)
One can do many “test moves” starting from a single starting.

Is there any advantage in doing so?

YES, SOMETIMES:
Example: “virtual move parallel tempering”
Parallel Tempering


(Inference-based MC)
System 1 at temperature $T_1$

Boltzmann factor

$$\exp(-\beta_1 U_1(r^N))$$

System 2 at temperature $T_2$

Boltzmann factor

$$\exp(-\beta_2 U_2(r^N))$$

Total Boltzmann factor

$$\exp(-\beta_1 U_1(r^N)) \exp(-\beta_2 U_2(r^N))$$
SWAP move

System 1 at temperature $T_2$

Boltzmann factor

$$\exp(-\beta_2 U_1(r^N))$$

System 2 at temperature $T_1$

Boltzmann factor

$$\exp(-\beta_1 U_2(r^N))$$

Total Boltzmann factor

$$\exp(-\beta_2 U_1(r^N)) \exp(-\beta_1 U_2(r^N))$$
Ratio

\[
\frac{\text{new Boltzmann weight}}{\text{old Boltzmann weight}} = \exp\{-(\beta_1 - \beta_2)[\mathcal{U}_2(r^N) - \mathcal{U}_1(r^N)]\}
\]
Systems may swap temperature if their combined Boltzmann factor allows it.

\[
\text{acc}(\text{swap}) = \min \left( 1, \exp\left\{ - (\beta_1 - \beta_2) [\mathcal{U}_2(r^N) - \mathcal{U}_1(r^N)] \right\} \right)
\]
Consider all possible “swaps” starting from any initial state $i$ (e.g. $i=3$)
Then from some other initial state $j$ (e.g. $j=5$), etc.

**Example:**

IN THIS WAY, WE CAN SAMPLE VERY MANY POTENTIAL TEMPERING MOVES AT VIRTUALLY ZERO ADDED COST.
Translocation free-energy landscape
(computed, using VMPT)

Without VMPT, this free energy landscape would be full of holes.
Virtual-move parallel tempering can speed up the estimate of free-energy landscapes by 1-2 orders of magnitude.