Cluster update algorithms

Kari Rummukainen, Nordita
(Late) Summer School on Monte Carlo Methods

- **Expectation values ($\beta = 1/T$):**

$$\langle O \rangle_T = \frac{1}{Z} \int \prod_x d\phi_x \, O(\phi) \, e^{-\beta E(\phi)}$$

- **With Monte Carlo simulation: generate a series of configurations Φ_1, Φ_2 ... Φ_N with some MC method. Measure $O_i = O(\Phi_i)$. Now**

$$\frac{1}{N} \sum_i O_i \;\to \langle O \rangle, \quad \text{when } N \to \infty$$

- **At finite N, the estimated error:**

$$\delta O = \sqrt{\frac{\Sigma_i (O_i - \langle O \rangle)^2}{N(N-1)}}$$

if and only if the measurements O_i are statistically independent.

- **In reality (Markov chain), successive configurations are correlated: $C(\tau) \propto \exp -\tau/\tau_a$. $\tau_a =$ autocorrelation time.**
- **# of independent configs is $\sim N/(1+2\tau_a) \to \delta O \approx \sqrt{1+2\tau_a}\delta O_{\text{naive}}$.**
- **τ_a depends on the MC update algorithm: we want an update with as small τ_a as possible $\to$ Cluster algorithms.**

Fundamentals: Fortuin-Kasteleyn cluster decomposition

Ising model (arbitrary dim.):

$$E = -\sum_{<i,j>} s_i s_j \qquad Z = \sum_{\{s\}} e^{-\beta E}\,.$$

($s_i = \pm 1$, $< i, j >$ nearest neighbour sites). Consider interaction between sites $< l, m >$, and remove it from E:

$$E_{l,m} = -\sum_{<i,j>\neq<l,m>} s_i s_j\,.$$

Define

$$Z^{\text{same}}_{l,m} \equiv \sum_{\{s\}} \delta_{s_i,s_j} e^{-\beta E_{l,m}}\,, \qquad Z^{\text{diff.}}_{l,m} \equiv \sum_{\{s\}} (1-\delta_{s_i,s_j}) e^{-\beta E_{l,m}}\,.$$

Now we can write $Z = e^{\beta} Z^{\text{same}}_{l,m} + e^{-\beta} Z^{\text{diff.}}_{l,m}$. Further, using

$$Z^{\text{ind.}}_{l,m} \equiv \sum_{\{s\}} e^{-\beta E_{l,m}} = Z^{\text{same}}_{l,m} + Z^{\text{diff.}}_{l,m}\,.$$

the partition function becomes

$$Z = (e^{\beta} - e^{-\beta}) Z^{\text{same}}_{l,m} + e^{-\beta} Z^{\text{ind.}}_{l,m}$$

Since the first term restricts $s_l = s_m$, and the second term contains no restriction, the weighting factors can be considered as relative probabilities of a bond between sites l, m and no bond (= independent states).
Normalizing: $p^{\text{same}} = 1 - e^{-2\beta}$, $p^{\text{ind.}} = e^{-2\beta}$.

Repeating for all $< i, j >$,

a) sites linked to each other by bonds form clusters

b) different clusters are independent: $s_c = \pm 1$

and the partition function can be written as

$$Z = \sum_{\text{bonds}} \sum_{s_c} p^b (1-p)^n = \sum_{\text{bonds}} p^b (1-p)^n 2^{N_c}$$

Here

$$\begin{aligned} p &= 1 - e^{-2\beta} \\ b &= \#\ \textbf{of interactions that form a bond}, \\ n &= \#\ \textbf{of interactions that do not form a bond}, \\ N_c &= \#\ \textbf{of clusters}. \end{aligned}$$

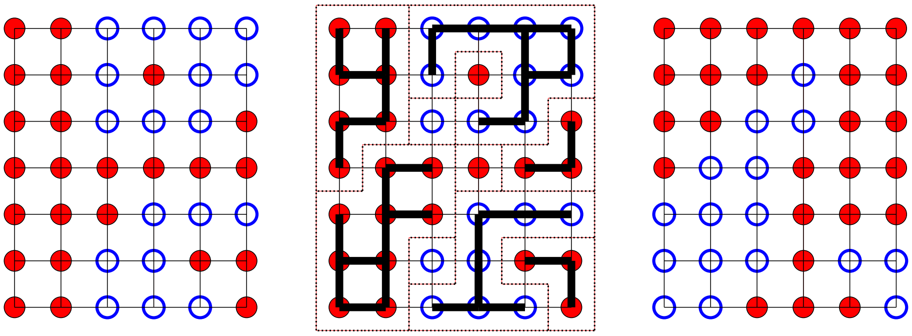
This decomposition is at the core of the cluster algorithms.

Swendsen-Wang cluster update

[Swendsen and Wang, PRL 58 (1987) 86]

Beginning with an arbitary configuration s_i, one SW cluster update cycle is:

1. Inspect all nn-states s_i, s_j. If $s_i = s_j$, create a bond between sites i, j with probability $p = 1 - \exp(-2\beta)$ (otherwise no bond).
2. Construct clusters = sets of points connected by bonds.
3. Set each cluster to a random value ± 1.



The configuration changes a lot in one update!

Is this a valid update? It must satisfy

a) ergodicity (obvious)

b) detailed balance: $\dfrac{P(A \mapsto B)}{P(B \mapsto A)} = \exp -\beta(E_B - E_A)$?

Proof: consider $A \mapsto \mathcal{C} \mapsto B$, where $\mathcal{C}$ is some bond configuration compatible with both A and B. Now $P(\mathcal{C} \mapsto A) = P(\mathcal{C} \mapsto B) = 1/2^{N_c}$, and

$$\frac{P(A \mapsto \mathcal{C})}{P(B \mapsto \mathcal{C})} = \frac{(1-p)^{d_A}}{(1-p)^{d_B}} = \exp -\beta(E_B - E_A)$$

where $d_{A,B}$ are the numbers of similar nn-states which are not connected by a bond. Thus $A \mapsto \mathcal{C} \mapsto B$ and $B \mapsto \mathcal{C} \mapsto A$ satisfy detailed balance for arbitrary $\mathcal{C}$, and the total transition probabilities $A \mapsto B$, $B \mapsto A$ must do it also.

Wolff single cluster update

[U. Wolff, PRL 62 (1989) 361]

Principle: do the cluster decomposition as in S-W, but invert ('flip') only one randomly chosen cluster! In practice:

1. **Choose random site i.**
2. **Study neighbouring sites j. If $s_j = s_i$, join site j to cluster with probability $p = 1 - \exp(-2\beta)$.**
3. **Repeat step 2 for site j, if it was joined to the cluster. Keep on doing this as long as the cluster grows.**
4. **When the cluster is finished, invert the spins which belong to it.**

• Usually slightly more effective than S-W (the average size of the clusters is larger. Why?).
• The minimum cluster size = 1, maximum = volume.
• Nicely recursive.
• Satisfies detailed balance.

As an example, let us consider 2-dimensional Ising model. It has 2nd order phase transition at $\beta = \log(1+\sqrt{2}) \approx 0.44$ (Curie point). Correlation length (and cluster size!) diverges when one approaches the critical point.

Compare the performance of
a) Metropolis and
b) Wolff cluster algorithm.

Cluster update in c-pseudocode:

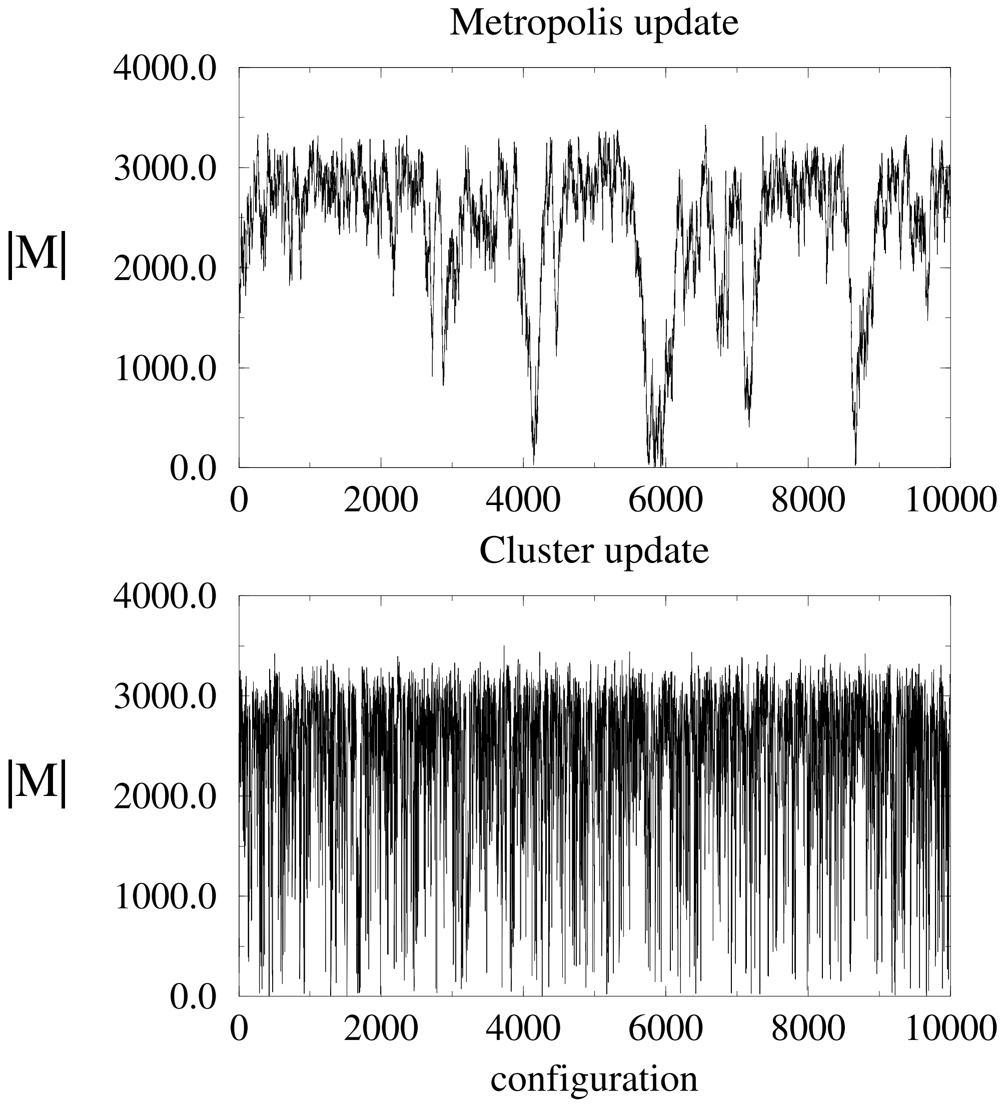
Here is a c-code snippet which should perform Wolff cluster update. `s[i]`: spin array, and `dran()` is a random number generator which returns a number fro 0 to 1.

```
void update_cluster()
{
  int start,state;

  start = dran() * volume;    /* starting location */
  state = s[start];           /* starting spin value */

  grow_cluster(start, state); /* start growing and inverting the cluster */
}

void grow_cluster(int loc,int state)
{
  int i,new_loc;

  s[loc] = -s[loc];        /* invert the spin at this location */

  /* begin loop over neighbour locations */
  for (i=0; i<n_neighbours; i++) {
    new_loc = neighbour[i][loc];   /* array of neighbour indices */

    if (s[new_loc] == state && exp(-2.0*beta) < dran())
      grow_cluster(new_loc,state);
  }
}
```

- **`s[loc]` is flipped during the cluster growth: this prevents revisiting sites already included in the cluster.**
- **Recursion (as above) is neat but not efficient! Better to unroll the recursion with loops and temporary arrays.**

System size 64^2, $\beta = 0.44$. Compare measurements of absolute value of magnetization

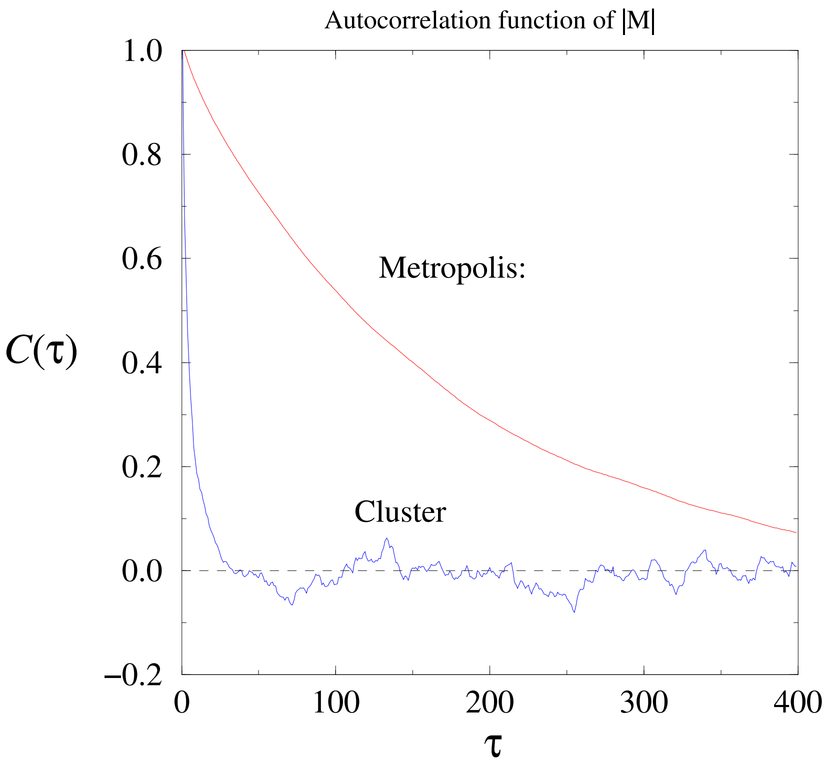
$$|M| = |\sum_i s_i|$$



Cluster update covers the phase space much faster than Metropolis!

The autocorrelation function describes how fast measurements become decorrelated.
Let now $O_i = |M|_i - < |M| >$, where $|M|_i$ is the measurement $\#i$ of $|M|$. Autocorrelation function $C(\tau)$ of $|M|$:

$$C(\tau) \equiv \sum_i O_i O_{i+\tau} \quad \propto \quad \exp -\tau/\tau_a$$

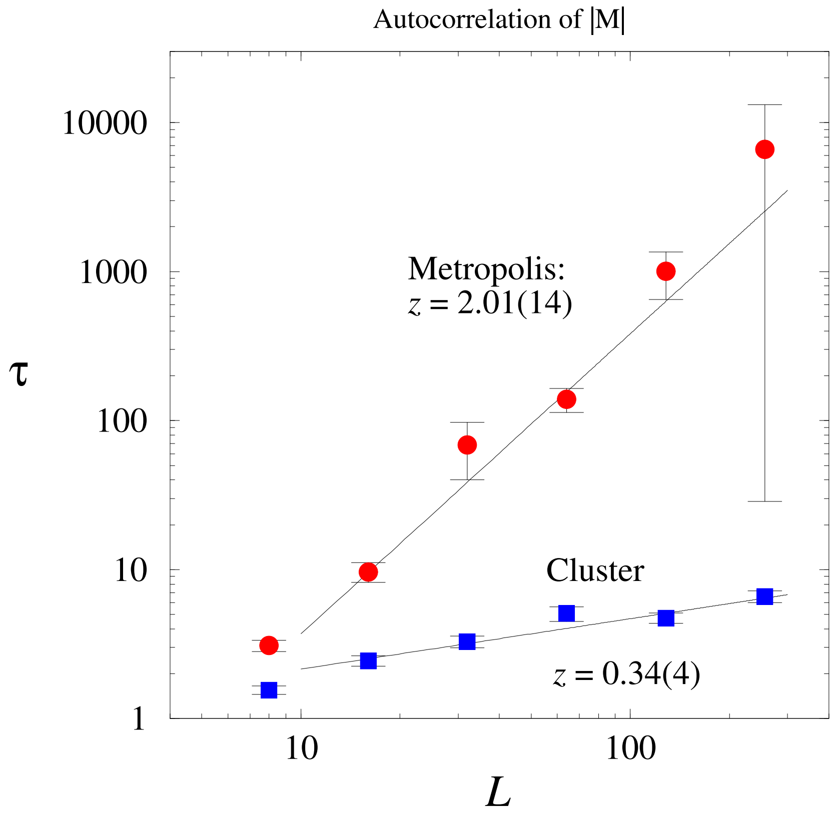


- **Exponential autocorrelation time: the exponential decay length of $C(\tau)$ at large τ.**

- **Integrated autocorrelation time:**

$$\tau_a^{\mathrm{int.}} \equiv \frac{1}{C(0)} \int d\tau C(\tau)\,.$$

Autocorrelation time $\tau_a^{\text{int.}}$ at different lattice sizes $L^2 = 8^2 - 256^2$:



• **Expected behaviour: $\tau_a \propto L^z$, where z is called dynamical critical exponent. For local (dissipative) algorithms, $z \geq \gamma/\nu \sim 2.125$ for the 2d Ising model.**

• **For volumes $\sim 256^2$, cluster algorithm is ~ 1000 times better than Metropolis!**

• **Measured in real (cpu/wallclock) time, cluster is even better (at the critical temperature $\beta_c = 0.44$):**

time/update $\propto V$ (local Metropolis)

time/update $\propto V^x$, where $x < 1$

What about other models?

Cluster algorithms can work, if one can embed an Ising system into the original system. Example: O(N) non-linear sigma model:

The Hamiltonian of the model is

$$E = -\sum_{<ij>} \sigma_i \cdot \sigma_j\,, \qquad \text{where } |\sigma_i|^2 = \sum_{\alpha=1}^{N} (\sigma_i^\alpha)^2 = 1$$

Thus, spins σ_i are N-dimensional vectors of unit length.
Embedding: [U.Wolff, PRL 62 (1989) 361]

Map arbitrary sigma model configuration to Ising model with Hamiltonian

$$E_I = -\sum_{<ij>} J_{ij} s_i s_j$$

as follows:

- **Choose a random O(N) vector r.**
- **Set all Ising model links $J_{ij} = (r\cdot\sigma_i)(r\cdot\sigma_j)$.**
- **Ising update $s_i \to -s_i$ corresponds to reflection of σ_i along the vector r: $\sigma_i \to \sigma_i - 2(r\cdot\sigma_i)\,r$**
- **Initially, all $s_i = +1$.**

After the mapping, we can use either S-W or Wolff update on the Ising spins. In practice, one Wolff update cycle proceeds as follows:

1. **Choose random O(N) vector r.**
2. **Choose random site i as the starting point for the cluster.**
3. **Study neighbouring sites j. Join site j to cluster with probability $p = 1 - \exp(-2\beta J_{ij})$, where $J_{ij} = (r\cdot\sigma_i)(r\cdot\sigma_j)$.**
4. **Repeat step 3 for all sites joined to the cluster. Keep on doing this as long as the cluster grows.**

5. When the cluster is finished, reflect $\sigma_i \to \sigma_i - 2(r \cdot \sigma_i)$ for all sites which belong to the cluster.

Typically the reflection is performed during the cluster growth.

Single cluster O(N) sigma model algorithm works even better than for Ising: In 3-dimensional O(4) sigma model, the dynamical critical exponent z becomes negative.

In what models clusters work, in what models they fail?
Clusters (usually) fail, if

- there are frustrated couplings (spin glasses, gauge theories ...)
- one cannot construct a symmetric reflection operation
- spins are 'frozen' in place by external fields etc.

Cluster updates are (normally) usable only in proximity of a 2nd order phase transitions: large correlation lengths $\to$ large clusters.

Nevertheless, sometimes they are useful when correlation lengths are finite but still large ($\gg 1$) in lattice units.

In order to work, the reflection operation $R\sigma = \sigma'$ must satisfy the following: the set of the fixed points of the reflection ($R\sigma = \sigma$) must separate the phase space of σ in two disconnected sets.
For example, in O(3), the fixed points are the points with $r \cdot \sigma = 0$, i.e. the points on the 'equator' perpendicular to the reflection vector r. This divides O(3) into disconnected 'northern' and 'southern' hemisphere.
Mathematically: fixed point set should have codimension 1 [Lüscher, Weisz, Wolff, NPB 359 (1991) 221].