Modern Methods in Protein Simulations

Christoph Junghans

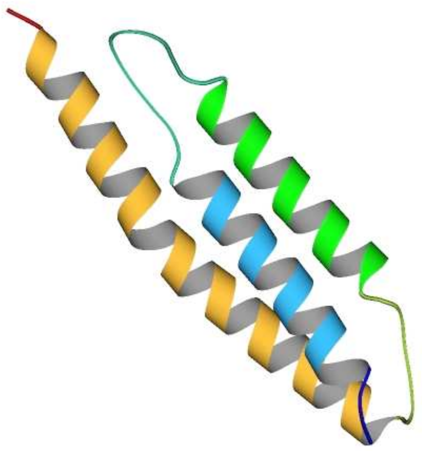
Institute of Theoretical Physics
Department of Physics and Geoscience
University Leipzig

E-mail: christoph.junghans@itp.uni-leipzig.de

Abstract: I study a realistic protein model using four advanced Monte Carlo techniques: multicanonical Monte Carlo, parallel tempering, Wang-Landau sampling and simulated tempering. The comparison showed that there are problems with ergodicity in the small energy region of big peptides due to the size of the configuration space and the high energy barriers. I discuss modifications of these algorithms that partly overcome these problems.

Introduction

Protein are the nanomachines responsible for nearly every process in our lives, e.g. transporting molecules, catalyzing biochemical reactions and fighting infections.



All proteins are build out of 20 different aminoacids. The unique 3D structure of a naturally occurring protein is determined by its amino acid sequence. These structures are responsible for the function of the protein.
My interest is in the process of folding, especially misfolding events, which cause various diseases. After reconstructing the folding process this knownledge enables the design of new drugs with customized properties.
But proteins consist of many atoms with not wellknown interactions making an investigation difficult. So I have to think about a model for proteins and methods to research it.

Basics of Computional Physics

There are two numerical ways of finding solutions: Molecular Dynamics and Monte Carlo. Molecular Dynamics solves the equations of motion for every single atom step by step numerically, but this is very time consuming because of the many interactions caused by the great number of other atoms. Monte Carlo is used to calculate the statistical properties of the whole system consisting of all atoms, Molecular Dynamics provides statistical properties, too. Such properties are, e.g., energy, radius of gyration, heat capacity and end-to-end distance. I am not interested in the time development of individual properties but in the statistic averages, so I will discuss only the statistical properties in this report.

Basics of Statistical Physics

For ease of notation I will speak about states instead of configurations and expect that the energy has discrete values, which is a simplification, but as I want to implement a model in a computer I need to discretize. If the system interacts with a reservoir at a constant temperature T, in equilibrium every state μ occurs with a certain probability p_μ. Gibbs showed in 1902 that these occupation probabilities are given by:

$$p_\mu(T) = \frac{\mathrm{e}^{-E_\mu/\mathrm{k_B}T}}{\sum_\nu \mathrm{e}^{-E_\nu/\mathrm{k_B}T}},$$

where E_μ is the energy of the state μ and $\mathrm{k_B}$ is the Boltzmann's constant. I denote $1/\mathrm{k_B}T$ with β, which is called the inverse temperature. It holds that

$$p_\mu(T) \propto \mathrm{e}^{-\beta E_\mu},$$

This distribution is called Boltzmann distribution. I can rewrite the distribution of the states into a distribution of the energies

$$P(E,T) = \frac{\Omega(E)\mathrm{e}^{-\beta E}}{\sum_E \Omega(E)\mathrm{e}^{-\beta E}},$$

where $\Omega(E)$ is the density of states counting the number of states with energy E. The distribution remains unchanged if I replace $\Omega(E)$ by $C \cdot \Omega(E)$. If the density of states is known, I obtain the distribution of the energies and therefore the meanvalues of the energy.

If I measure some quantity Q in an experiment repeatedly I can calculate the expectation value of the quantity $\langle Q \rangle$. This value is given as the quantity in a state times the occupation probability of that state, summed over all possible states

$$\langle Q \rangle(T) = \sum_\mu Q_\mu p_\mu(T).$$

Or, if the quantity can explicitly be described in terms of the energy,

$$\langle Q \rangle(T) = \sum_E Q(E)P(E,T).$$

Monte Carlo Simulations

Usually a measurement does not take infinite time and the system will not pass through every state in the sum of expection value. Therefore I can reduce the sum from all possible states to the important states. This procedure is called *importance sampling*.

To produce such a subset of important states I use a Markov Chain process which produces a new state ν out of a given state μ with some transition probability $P(\mu \to \nu)$.

$$\mu \overset{P(\mu\to\nu)}{\longrightarrow} \nu \overset{P(\nu\to\lambda)}{\longrightarrow} \lambda$$

This process has to fulfill some conditions:

- Normalization: $\sum_\nu P(\mu \to \nu) = 1$
  This is necessary to ensure that the transition properties are normalized and that at least one transition is possible.
- Ergodicity:
  It must be possible that every state can be reached from every state in a finite number of steps.

- Detailed Balance: $s_\mu P(\mu \to \nu) = s_\nu P(\nu \to \mu)$
  If this condition is satisfied, the occupation probabilities of the states in the chain are given by s_μ.

From a chain of M measurement of Q I can calculate an estimator of the expection value .

$$\tilde{Q}(T) = \frac{\sum_{i=1}^{M} Q_i s_i^{-1} \mathrm{e}^{-\beta E_i}}{\sum_j s_j^{-1} \mathrm{e}^{-\beta E_j}}$$

For the case of a Boltzmann distributed chain I get

$$\tilde{Q}(T) = \frac{\sum_{i=1}^{M} Q_i}{M}$$

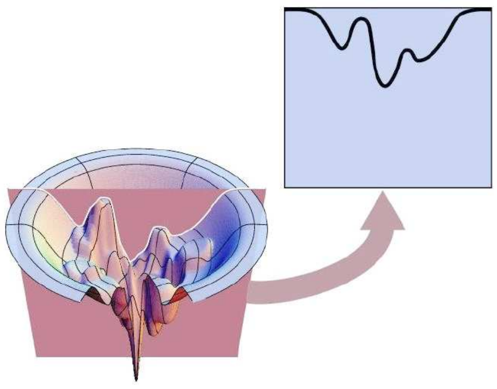
The only open question is the choice of the transition probabilities, I take the choice of Metropolis [1].

$$P(\mu \to \nu) = \min\left(1, \frac{s_\nu}{s_\mu}\right)$$

For $s_\mu = p_\mu$ this is the normal Metropolis Update.

Advanced Monte Carlo

The normal Metropolis Update is inefficient in simulations of systems such as proteins that are characterized by a rough energy landscape. Small changes in configuration can cause great changes in energy. If the system finds a state with an energy smaller than all the states the update could reach in the next step, a so called local minima (see Fig. to the right [2]), the probability to leave a local minima falls off exponentially with the height of the energy barrier. Hence there is a need for better algorithms than the normal Metropolis Update.



Multicanonical Simulations

Here, the idea [3] is to sample all possible energies with nearly equal probability. This helps solving problems caused by big differences in the density of states which can go over 100 orders of magnitude for proteins. The distribution to sample is the multicanonical distribution

$$P_{\text{MuCa}}(E) = W_{\text{MuCa}} \Omega(E) \approx \text{constant}$$

The density $\Omega(E)$ contains all energy information about the system but the density is a priori unknown, so the multicanonical weights $W_{\text{MuCa}}(E)$ are also unknown and need to be determined. In this algorithm the multicanonical weights are the input to the Metropolis update $s_\mu = W_{\text{MuCa}}(E_\mu)$.

One way to determine the weights is the multicanonical recursion [4] which is based on some conditions:

- Discrete energies with smallest energy difference ϵ
- Histogram entries $H(E)$ are uncorrelated

I rewrite the weight factor:

$$W_{\text{MuCa}} = \mathrm{e}^{-b(E)E+a(E)}$$

where $b(E)$ is the mircocanonical temperature and $a(E)$ the mircocanonical free energy. These two quantities are linked by the definition of the mircocanonical temperature:

$$a(E-\epsilon) = a(E) + (b(E-\epsilon) - b(E))E.$$

I choose $a(E_{\text{max}}) = 0$. For the beginning I use $b^0(E) = 0 = a^0(E)$ and $g^0(E) = 0$ as normalization parameter. Then the recursion works as follows:

- Determine the nth histogram of the energies $H^n(E)$ with the help of W^n_{MuCa}
- $\kappa^n(E) = \frac{H^n(E+\epsilon)H^n(E)}{H^n(E+\epsilon)+H^n(E)}$
- $g^{n+1}(E) = g^n(E) + \kappa^n(E)$
- $b^{n+1}(E) = b^n(E) + \frac{\kappa^n(E)}{g^{n+1}(E)} \cdot \frac{\ln H^n(E+\epsilon) - \ln H^n(E)}{\epsilon}$

After doing the recursion n_{MuCa} times with l_{MuCa} sweeps, I get a good estimate for the multicanonical weights. The number of runs n_{MuCa} is reached if the weights change only very little in comparison to the last run. Then I have to do a final run of m_{MuCa} sweeps with fixed weights in which I measure all interesting quantities.

Wang-Landau Algorithm

All the energy information are given by the density of states $\Omega(E)$. The Wang-Landau algorithm [5] was designed to estimate the density of states directly. This is a different starting point than the idea of multicanonical simulations, but it turns out to be a good input to the multicanonical final run.

First I initialize all weights $W_{\text{WL}}(E_\mu) = 1 = s_\mu$ and the shape parameter $f = \mathrm{e}^1$. Now I perform a Metropolis update and change the weight of the visited energy by $W_{\text{WL}}(E) \to W_{\text{WL}}(E)/f$. After n_{WL} sweeps I change the shape parameter by $f \to \sqrt{f}$ and go on. This change also gives the beginning of the next run. If the shape parameter is approximately 1 the simulation is finished and an estimate for the density of states is given by:

$$\Omega(E) = 1/W_{\text{WL}}(E)$$

This algorithm does not fulfill detailed balance and there is no way of calculating the errors of the density. To solve these problems I using the Wang-Landau weights as input for the multicanonical final run because:

$$W_{\text{MuCa}}(E) \propto 1/\Omega(E) = W_{\text{WL}}(E)$$

Another problem is that the energies have to be discrete, so binning is necessary.

Random Tempering

Simulated Tempering [6] expands the configuration space with an extra temperature coordinate. I perform two kinds of updates:

- Normal Metropolis update:

$$P(E \to E') = \min\left(1, \frac{\mathrm{e}^{-\beta E'}}{\mathrm{e}^{-\beta E}}\right)$$

- Temperature change update:

$$P(\beta \to \beta') = \min\left(1, \frac{\mathrm{e}^{-\beta' E - g(\beta')}}{\mathrm{e}^{-\beta E - g(\beta)}}\right)$$

The system has different equilibration energies at different temperatures, so that the sampled configuration space is much bigger. Without the weights $g(\beta)$ the acceptance ratio of the temperature changes are very small so I introduce these weights, but now I have to determine them first. I choose to do it with the help of the Wang-Landau method.

First I initial all weights $g(E) = 0$ and the shape parameter $h = 1$. Then I perform the updates in alternating order changing the weights of the visited temperature after every cycle by $g(T) \to g(T) + h$. If the histogram of the visited temperatures $H(T)$ becomes sufficiently flat I change the shape parameter by $h = h/2$. Sufficiently flat means the deviation from the mean is at most 20% of the mean. Continue until $h \approx 0$. Because this new method performs a random walk in temperature space I call this enriched version of simulated tempering the random tempering method.

Parallel Tempering

This method [7] is very similar to simulated tempering but works with n copies of the system instead of one. Every system is simulated at a different temperatures and I perform two kind of updates:

- Normal Metropolis update:

$$P(E \to E') = \min\left(1, \frac{\mathrm{e}^{-\beta E'}}{\mathrm{e}^{-\beta E}}\right)$$

- Configuration swap of 2 systems:

$$P(\mu \leftrightarrow \nu) = \min\left(1, \frac{\mathrm{e}^{-\beta_\mu E_\nu - \beta_\nu E_\mu}}{\mathrm{e}^{-\beta_\mu E_\mu - \beta_\nu E_\nu}}\right)$$

There are no weights to determine. This method has a natural implementation on parallel computers.

Protein Simulations

The methods described above are used in many different fields of computer simulation and also in protein simulation. What makes protein folding so complicated is the rough energy landscape caused by the interactions of the great number of atoms. One simplified description of the interactions is given by the following force field.

Force Field

The energy function [8] I used is measured in kcal/mol:

$$E_{\text{tot}} = E_{\text{LJ}} + E_{\text{el}} + E_{\text{hb}} + E_{\text{tors}}$$

where

- Lennard-Jones term $E_{\text{LJ}} = \sum_{j>i}\left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}}\right)$

- Electrostatic term $E_{\text{el}} = \sum_{i,j} \frac{332 q_i q_j}{\epsilon r_{ij}}$
- Hydrogen-bond term $E_{\text{hb}} = \sum_{j>i} \left(\frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^{10}} \right)$
- Torsion term $E_{\text{tors}} = \sum_l U_l (1 \pm \cos(n_l \chi_l))$

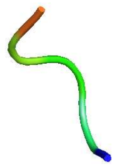
The missing constants are taken from [8]. The Lennard-Jones term takes the radii of the atoms and repulsion of the electron clouds of the atoms into account, the electrostatic term gives the interactions between charged particles, the Hydrogen-bond term describes the energy caused by the polarization of the atom and the torsion term stands for the energy stored in the torsion of the bonds.

Objects of Studies

It is not possible to understand all aspects of folding in one universal protein. I study three different small peptides.

- Met-enkephalin

  $$\text{Tyr} - \text{Gly} - \text{Gly} - \text{Phe} - \text{Met}$$

  This is the work horse of algorithms tested in the field of protein folding.

- Alanine 10

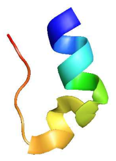
  $$\text{Ala} - \text{Ala} - \text{Ala} - \text{Ala} - \text{Ala} - \text{Ala} - \text{Ala} - \text{Ala} - \text{Ala} - \text{Ala}$$

  A littler bigger than Met-enkephalin, but still with simple properties.

- Trp-Cage

  $$\text{Asn} - \text{Leu} - \text{Tyr} - \text{Ile} - \text{Gln} - \text{Trp} - \text{Leu} - \text{Lys} - \text{Asp} - \text{Gly} -$$

  $$\text{Gly} - \text{Pro} - \text{Ser} - \text{Ser} - \text{Gly} - \text{Arg} - \text{Pro} - \text{Pro} - \text{Pro} - \text{Ser}$$

  This is a much bigger molecule with nontrivial folding behavior and a characteristic groundstate.

Results

Numerical Comparison of Multicanonical Simulation and Wang-Landau Sampling

As it is the first time that the Wang-Landau algorithm was used together with this force field I first compare the density of states calculated with Wang-Landau with results from a multicanonical simulation. Good analogy can be seen in Figure 1. The graphs are normalized on the energy bin next to the highest energy. For technical reasons the last bin of the multicanonical simulation is filled with all higher energies. Out of this density the meanvalue of the energy was calculated (see Fig. 2) and showed good results at temperatures above room temperature and small deviation at lower temperatures.

The final simulation runs always had a length of $m_{\text{MuCa}} = 100.000$ sweeps. The bin size was 1 kcal/mol. The interesting energy region was $[-12 \ldots 60]$ kcal/mol for Alanine 10 and $[-12 \ldots 20]$ kcal/mol for Met-enkephalin. For weight determination in multicanonical simulation I used 20 recursion run with

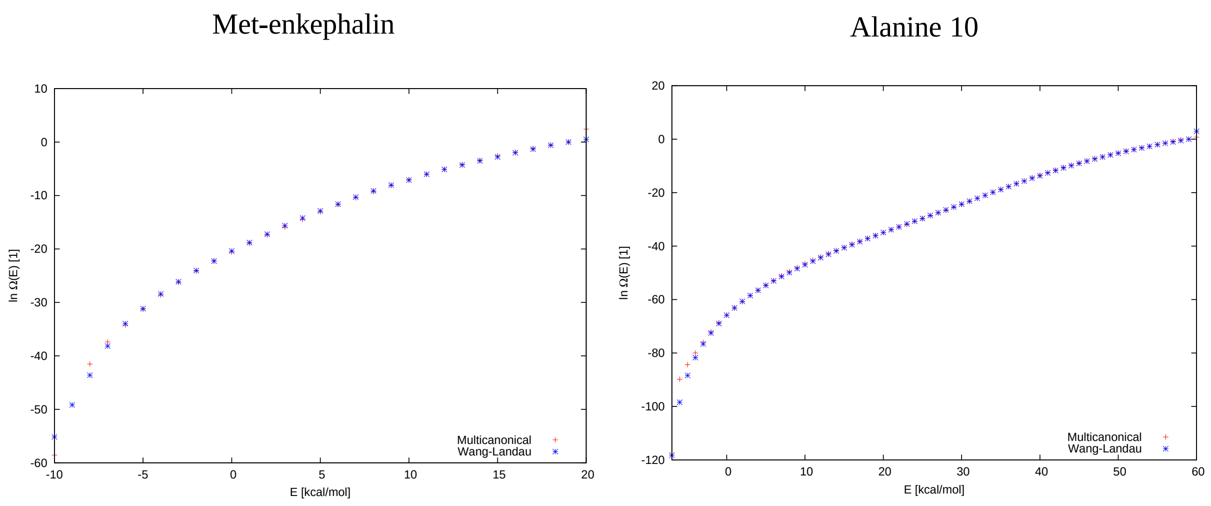


Fig. 1: Comparison of the density of states with multicanonic simulation and Wang Landau sampling for Met-enkephalin and Alanine 10. Good analogy showed up in both cases.

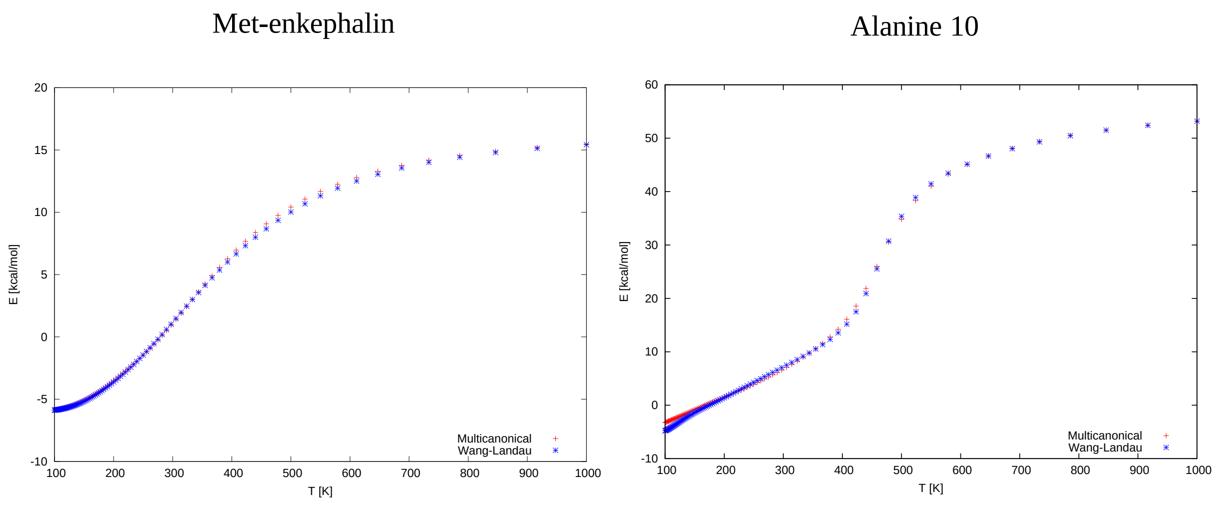


Fig. 2: Comparison of the meanvalue of the energies with multicanonical simulation and Wang Landau sampling for Met-enkephalin and Alanine 10. Good analogy can be seen for room temperature or higher and small deviation for the lower energy area.

$n_{\mathrm{MuCa}} = 5.000$ sweeps and 10 runs with $n_{\mathrm{WL}} = 10.000$ sweeps for Wang-Landau sampling. Depending on the start configuration the weight finding with Wang-Landau was around 10% faster than the multicanonical recursion.

I also tried using the Wang-Landau algorithm for Trp-cage, but it fails. The problem is the big size of the configuration space. The interesting energy range is $[-170 \ldots 50]$ kcal/mol. This caused also a big difference in the order of magnitude of the density of states. The system gets stuck in some state around the ground state and has no chance to move to higher energies again so the meanvalue for small temperatures becomes too small (see Fig. 4). Multicanonical simulations have the same problems here. For big peptides there are too many configurations with the same energy. It is not possible to visit all in finite time. Configuration with similar energy do not have large overlap, which means a big distance in configuration space, so I need to try other methods.

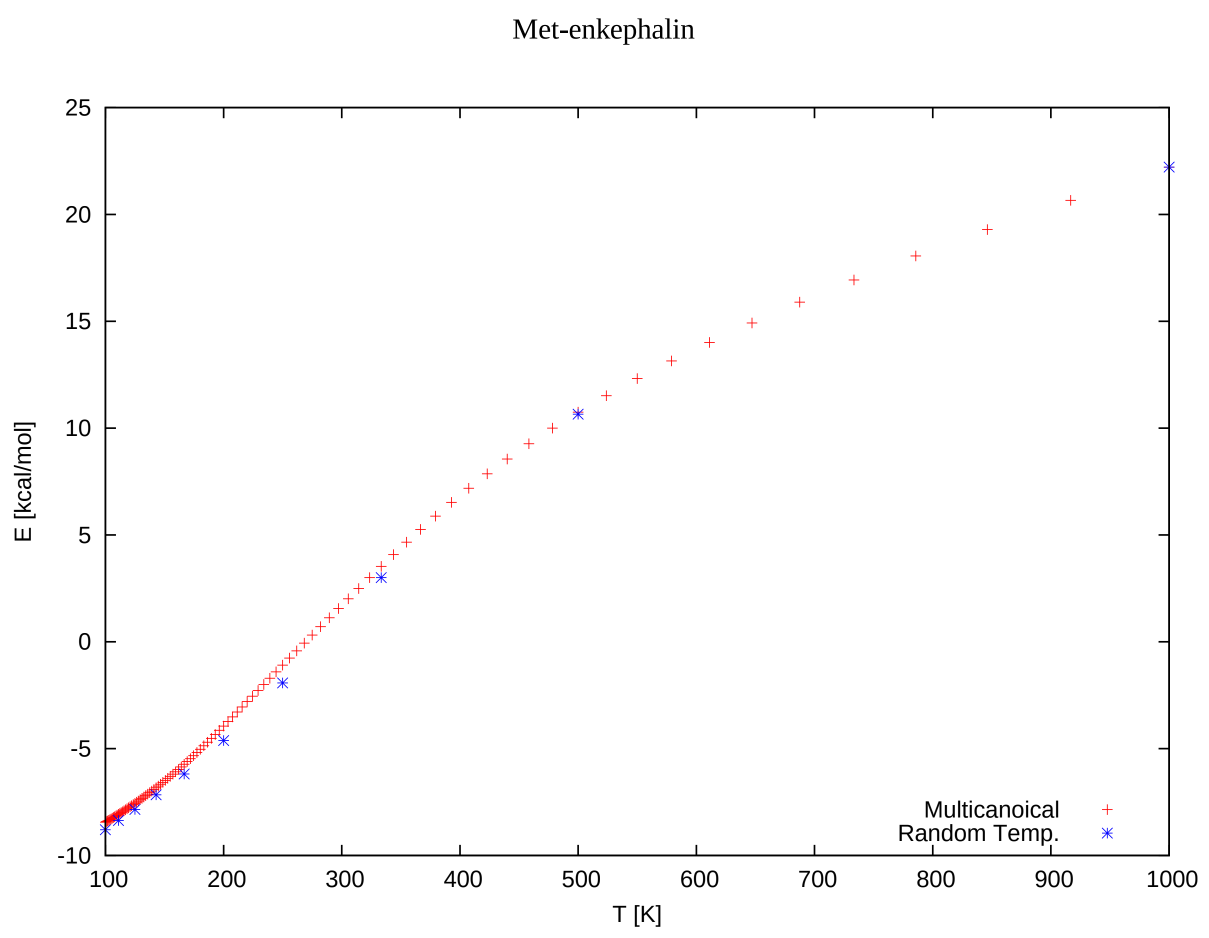


Fig. 3: Comparison of multicanonical simulation and random tempering for Met-enkephalin. The sampled temperature point of random tempering fit together with the curve of the multicanonical simulation.

Comparison of Random Tempering and Multicanonical Simulation

Multicanonical simulations and Wang-Landau sampling give us information about all temperatures in one simulation. In most cases I am just interested in values at some fixed values of temperature. As in the last section, I first want to test if random tempering works correctly for Met-enkephalin. I choose ten temperatures between 100 K and 1000 K. The temperatures are equidistant on the inverse temperature scale because the distribution depends on β not T.

The determination of the weights for random tempering takes around 1.000.000 sweeps in 15 runs to get a sufficiently flat distribution in temperature. It also takes the same number of sweeps to get a flat distribution in the final run with fixed weights. In comparison to multicanonical simulation this is a long time but the results look similar. Results for Met-enkephalin are shown in Fig. 3.

For Trp-cage the distribution is not flat even after the 10th run of determining the weights. In the final run the distribution is not flat either and causes some problem in low energy regions (see Fig. 4).

Comparison with Parallel Tempering

As even random tempering has problem with low energy regions, I test parallel tempering for Trp-cage on the supercomputer JUMP in the Research Center Jülich. One time 16 CPUs $\times$ 10 h with 1.100.000 sweeps and the other time 10 CPUs $\times$ 10 h with 1.000.000 sweeps. The results are consistent even with

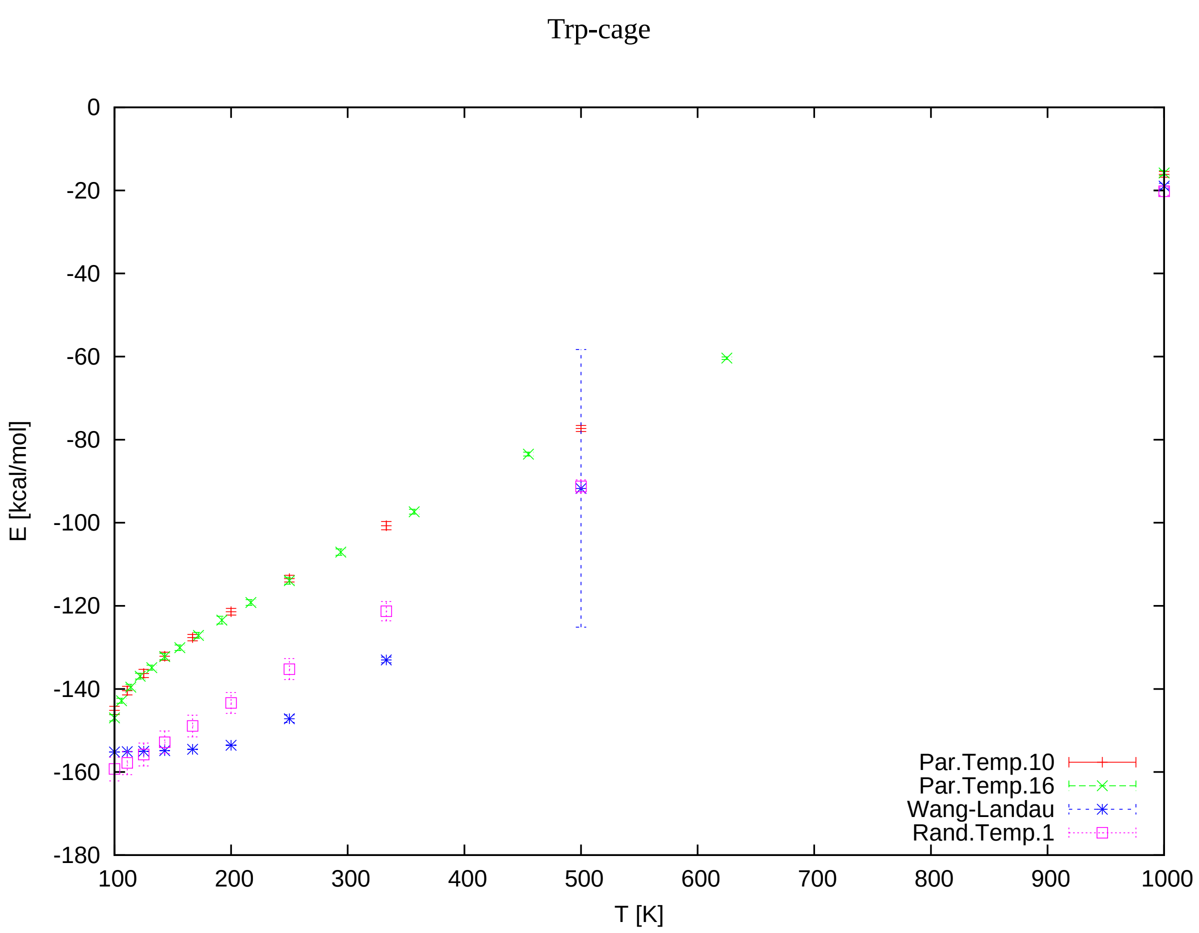


Fig. 4: Comparison of several methods for Trp-cage. Random tempering and Wang-Landau sampling have problem in the low energy regions. High temperature points fit together.

different temperature points (Fig. 4). They have small error bars and are repeatable.

Conclusion

All methods I have tested except parallel tempering have produced problems with low temperatures and therefore with the low energy region of big peptides. The statistical results are not repeatable with other starting conditions. This is caused by the size of the configuration space and the large energy barriers. For the smallest tested peptides all methods work well. Maybe it is possible to get better results in a longer simulation run. But in comparison to parallel tempering the efficency is bad.

Outlook

Because of the very long simulation times it is necessary to parallelize the broad histogram methods like multicanonical simulation and Wang-Landau sampling. This would help to get the same information in shorter time, so that I can have more information about the system in the same time. Till now I cannot even think about simulating a real protein on a supercomputer.

Also I have to think about a way to overcome those configuration barriers. Maybe the choice of the interestering regions in the configuration space have to be done first. Also getting a flat histogram in energy space is not always the best, a flat distribution in overlap to the ground state, which must be experimental known, may be better.

Acknowledgment

I thank Ulrich Hansmann for giving me this topic near to the current research interests. Also I like to thank Jan Meinke for helping me with all my scientific, technical and administrative problems. Last but not least I thank Thomas Neuhaus for interesting discussion on Advanced Monte Carlo Methods.

References

1. Metropolis et al., Jour. Chem. Phys. 21(1953), 1087-1092
2. Dill and Chan,Natu. Stru. Biol. 1997, 4
3. Berg and Neuhaus, Phys. Rev. Lett. 68(1992), 9
4. Berg, Comp. Phys. Comm. 153(2003), 397-406
5. Wang and Landau, Phys. Rev. E 64(2002), 056101
6. Marinari and Parisi, Europhys. Lett. 19(1992), 451-458
7. Hukushima and Nemoto, Jour. Phys. Soc. Jap. 65(1996), 1604-1608
8. Hansmann et al., Comp. Phys. Comm. 138(2001), 192-212