



Efficient performance evaluation of Forward Error Correcting codes

Shahkar Kakakhail, Sylvain Reynal, David Declercq, Vincent Heinrich

► To cite this version:

Shahkar Kakakhail, Sylvain Reynal, David Declercq, Vincent Heinrich. Efficient performance evaluation of Forward Error Correcting codes. 11th IEEE Singapore International Conference on Communications Systems, Nov 2008, Guangzhou, China. pp.263-267, 10.1109/ICCS.2008.4737185 . hal-00512279

HAL Id: hal-00512279

<https://hal.science/hal-00512279>

Submitted on 29 Aug 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Efficient Performance Evaluation of Forward Error Correcting Codes

S. Kakakhail^{†*}, S. Reynal[†], D. Declercq[†], V. Y. Heinrich^{*}

[†]ETIS/ENSEA/UCP/CNRS UMR-8051

95014 Cergy-Pontoise, France

kakakhail,reynal,declercq@ensea.fr

^{*}STMicroelectronics

Crolles, France

vincent.heinrich@st.com

Abstract

Standard Monte Carlo (SMC) simulation is employed to evaluate the performance of Forward Error Correcting (FEC) codes. This performance is in terms of the probability of error during the transmission of information through digital communication systems. The time taken by SMC simulation to estimate the FER increases exponentially with the increase in Signal-to-Noise Ratio (SNR). We hereby present an improved version of Fast Flat Histogram (FFH) method, an Adaptive Importance Sampling (AIS) technique inspired by algorithms existing in statistical physics. We show that the improved FFH method employing Wang Landau algorithm based on a Markov Chain Monte Carlo (MCMC) sampler reduces the simulation time of the performance evaluation of complex FEC codes having different code rates.

1 Introduction

Standard Monte Carlo (SMC) simulation is carried out to gauge the performance of Forward Error Correcting (FEC) codes. An SMC simulation comprises of simulating a system by generating random inputs according to a probability distribution and then evaluating the system response. For digital communication systems, SMC simulation is used to evaluate the quality of the transmission system usually by measuring Frame Error Rate (FER) i.e., the ratio between the decoded frames that contain errors to the total number of frames sent through the system. With the advancement in the code design and better decoders, it has become very important to gauge the performance of the system at very low error rates. On the one hand, there are novel applications operating at very low error probability and Standard Monte Carlo (SMC) simulation

takes extremely long to gauge their performance. On the other hand, new codes and decoders compete for a better performance in the error floor region where the performance evaluation is curbed and it is quite difficult to study the system properties.

Importance Sampling (IS) [1] is a technique where the probability density of events is biased in a controlled way (based on a biasing function) in order to get the important events more frequently. For linear block codes, IS was first used in [2] where a regular LDPC code with a block length of 96 was used making use of the code structure to produce noise events more frequently. A recent approach [3] also makes use of both graph structure of LDPC codes and IS.

Turbo codes [4] make a class of FEC codes which are graphically represented through a trellis diagram. In the high SNR region, the probability of error is dominated mainly by the distance properties of turbo codes which depend on the interleaver design between the two constituent encoders. For turbo codes, an IS method based on the distance properties and error patterns was presented in [5].

The main problem with the methods that aim at reducing simulation time based on the code structure is that the formulation of biasing function of IS becomes extremely cumbersome if the code structure is irregular or the code is long.

Another approach to simulate the error probability, which is not dependent on the code structure for the performance evaluation, is the advanced version of IS known as Adaptive Importance Sampling (AIS). In AIS, the probability density function is biased in a controlled way during multiple iterations thus making it possible to visit the tails of the noise distribution. The system response (the output of the decoder in our case) is evaluated and stored in the process. An example of AIS technique, which does not take into account the code structure, has been successfully employed in [6] where the authors use a Dual Adaptive

Importance Sampling (DAIS) technique. DAIS has been tested on a regular LDPC code of 96 bits. Recently, another AIS method known as Fast Flat Histogram (FFH) method (inspired from [7] in statistical physics) was presented for LDPC codes of relatively large codeword lengths going up to 2640 coded bits [8].

AIS is especially interesting for its genericity in the sense that it can be adapted to any stationary memoryless channel (AWGN, BSC, etc.), to any type of decoder (turbo decoder, Gallager B, BP, etc.) and to any class of codes (binary and duo-binary turbo codes, regular and irregular LDPC codes, etc.). In this paper, we will focus on duo-binary turbo codes [9] used on AWGN channel decoded with standard duo-binary turbo decoder. We propose an improved FFH method with a new self-adaptive procedure to determine the optimum sampling domain corresponding to the code, a more stringent stopping criterion and some additional steps to tailor the traditional Wang Landau algorithm according to our objectives. We show the application of the improved FFH method for duo-binary turbo codes used in DVB-RCS standard [10] with varying code rates thus validating the genericity of the method for FEC codes.

The rest of the paper is organized as follows: Section 2 describes the FFH method detailing how it is applied to FEC codes (duo-binary turbo codes in our case). Section 3 gives the results for a typical 8-state, 188 bytes (MPEG-size) DVB-RCS standard code using AWGN channel and an iterative soft decision decoder. Section 4 concludes our work.

2 Fast Flat Histogram Method

2.1 Description

The basic skeleton of our technique is the same as that in [6, 8], that is, we aim at increasing the number of events in the tails of the probability density function (pdf) by sampling from a biased pdf. However, compared to [8] a number of improvements are made in the algorithm to render it more powerful, robust and reliable.

We recall the notations from [6, 8]. Let Γ be the n -dimensional probability space of the noise in the n bits of a codeword. The noise vector $\mathbf{z} = (z_1, z_2, \dots, z_n)$ is a multivariate Gaussian with joint pdf $\rho(\mathbf{z}) = \prod_{l=1}^n \rho_l(z_l)$. The transmitted bit vector is represented by $\mathbf{b} = (b_1, b_2, \dots, b_n)$ and $\mathbf{y} = (y_1, y_2, \dots, y_n)$ represents the received codeword. The algorithm is controlled by a scalar control quantity V given as $V(\mathbf{z}) = \left[\frac{1}{n} \sum_{l=1}^n [H(q_l z_l)]^2 \right]^{1/2}$ where

$q_l = (-1)^{b_l}$ while b_l is the transmitted bit in the l th position and $H(x) = 1$ if $x < 0$ and $H(x) = 0$ otherwise. $V(\mathbf{z})$ is constructed such that a noise component z_l contributes to V only if it may produce a bit error at the input to the decoder.

Given a range $[V_{min}, V_{max}]$ for V , Γ is partitioned into L subsets $\Gamma_k = \{\mathbf{z} \in \Gamma \mid V_{k-1} \leq V(\mathbf{z}) < V_k\}$, where $V_k = V_{min} + k\Delta V$, $1 \leq k \leq L$ and $\Delta V = (V_{max} - V_{min})/L$ is the width of each bin in the partition of $[V_{min}, V_{max}]$.

Let P_k be the probability of selecting a realization $\mathbf{z}$ from ρ such that $\mathbf{z} \in \Gamma_k$ [11, 12]. Then,

$$P_k = \int_{\Gamma} \chi_k(\mathbf{z}) \frac{\rho(\mathbf{z})}{\rho^*(\mathbf{z})} \rho^*(\mathbf{z}) d\mathbf{z} \approx \frac{1}{N} \sum_{i=1}^N \chi_k(\mathbf{z}^{*,i}) \frac{\rho(\mathbf{z}^{*,i})}{\rho^*(\mathbf{z}^{*,i})} \quad (1)$$

where $\rho^*(\mathbf{z})$ is a positive biasing pdf, $\chi_k = 1$ if $\mathbf{z} \in \Gamma_k$ and $\chi_k(\mathbf{z}) = 0$ otherwise. $\mathbf{z}^{*,i}$ are N random sample points in Γ selected according to the pdf $\rho^*(\mathbf{z})$. The variance of the estimate of (1) is zero if the optimal biasing pdf $\rho_{opt}^*(\mathbf{z}) = \chi_k(\mathbf{z})\rho(\mathbf{z})/P_k$ is used. However, $\rho_{opt}^*(\mathbf{z})$ depends on P_k which is initially unknown. In standard IS, one uses physical intuition to guess a biasing pdf that is close to ρ_{opt}^* . Like DAIS, the FFH method instead iterates over a sequence of biasing pdfs $\rho^{*,j}$ that approach ρ_{opt}^* . We define $\rho^{*,j}$ for j th iteration by $\rho^{*,j}(\mathbf{z}) = \rho(\mathbf{z})/(c^j P_k^j)$ where k is such that $\mathbf{z} \in \Gamma_k$ is satisfied. The quantities P_k^j satisfy $P_k^j > 0$ and $\sum_{k=1}^M P_k^j = 1$ and c^j is an unknown constant that ensures $\int_{\Gamma} \rho^{*,j}(\mathbf{z}) d\mathbf{z} = 1$. The vector P_k completely determines the bias and is initialized with $1/L, \forall k = 1, \dots, L$.

Our aim is to explore the whole of probability space Γ using random walk [13]. By employing Metropolis algorithm [14], we produce a random walk of *samples* $\mathbf{z}^{*,i}$ whose pdf equals $\rho^{*,j}(\mathbf{z})$. We consider a Markov chain of transitions consisting of small steps in the noise space. Each transition goes from $\mathbf{z}^{*,i} = \mathbf{z}_a^* \in \Gamma_{k_a}$ to $\mathbf{z}_b^* = (\mathbf{z}_a + \epsilon \Delta \mathbf{z}) \in \Gamma_{k_b}$ where $\Delta \mathbf{z}$ is random and symmetric, i.e., it does not favor any direction in Γ and the transition is accepted with probability π_{ab} . Here, ϵ is the perturbation constant. If a transition from $\mathbf{z}^{*,i}$ to $\mathbf{z}_b^*$ is accepted, we set $\mathbf{z}^{*,i+1} = \mathbf{z}_b^*$, else we set $\mathbf{z}^{*,i+1} = \mathbf{z}^{*,i} = \mathbf{z}_a^*$. The ratio π_{ab}/π_{ba} equals $\rho^{*,j}(\mathbf{z}_b^*)/\rho^{*,j}(\mathbf{z}_a^*)$ which is the *detailed balance equation* that ensures that the limiting (stationary) pdf for infinitely many steps of this random walk is $\rho^{*,j}$ [14].

We consider the perturbation of the noise component in each bit $z_{a,l}^*$ of $\mathbf{z}_a^*$ separately and accept it or reject it independently with the probability $\min[\rho(z_{b,l}^*)/\rho(z_{a,l}^*), 1]$. We pick each perturbation Δz_l from a zero mean symmetric

pdf. We obtain a trial state $\mathbf{z}_b^*$ in which only some of the components are different from their previous values in $\mathbf{z}_a^*$. Then we compute k_b , the bin corresponding to $\mathbf{z}_b^*$ and finally accept the step from $\mathbf{z}_a^*$ to $\mathbf{z}_b^*$ with the probability $\min[P_{k_a}^j/P_{k_b}^j, 1]$. The compound transition probability thus becomes

$$\pi_{ab} = \left\{ \prod_{l=1}^n \min \left[\frac{\rho(z_{b,l}^*)}{\rho(z_{a,l}^*)}, 1 \right] \right\} \min \left[\frac{P_{k_a}^j}{P_{k_b}^j}, 1 \right] \quad (2)$$

The Asymptotically Optimal Acceptance Rate AOAR $\alpha \triangleq (\text{number of accepted steps})/(\text{total number of steps})$ for a Metropolis algorithm for target distributions with IID components is 0.234 [15]. The perturbation constant ϵ is adjusted so as to keep α close to this value. The noise realizations are recorded in the histogram $H^{*,j}$ where $H_k^{*,j} = \sum_{i=1}^N \chi_k(\mathbf{z}^{*,i})$ is the number of $\mathbf{z}^{*,i}$ in iteration j that fall into Γ_k . Each noise vector is used in the channel to deteriorate the transmitted codeword which is then fed into the decoders to verify if the errors are corrected within the specified number of decoder iterations. To keep a record of errors in bin k , we produce an error histogram $G_k^{*,j}$.

P_k is updated *on the fly* such that when k bin is visited, P_k is modified by the refinement parameter $f > 1$, i.e. $P_k \rightarrow P_k \cdot f$ [16, 17]. In practice, we have to use the log domain $\ln P_k \rightarrow \ln P_k + \ln f$ in order to fit all possible P_k into double precision numbers. If the random walk rejects a possible move and stays in the same bin k , we modify the same P_k with the modification factor to keep the detailed balance equation in equilibrium.

In case of rejection of a possible move, a very significant additional step is to permute the components of the noise vector and to add this permuted sequence to the transmitted codeword on which decoding is carried out for error correction. It is important to note that the random walk is performed with the new permuted sequence so as not to disturb the detailed balance equilibrium. We keep on permuting the noise components until a possible move is accepted. The preceding step stems from the fact that different sequences of the same components of a noise vector lead to different decoding outputs since we are employing a message passing decoding algorithm. The orientation of the permuted noise components remain the same leading to the same V value and consequently staying in the same bin. Without effecting the basic modification of P_k values, we are thus able to check the system response of all entries in histogram $H^{*,j}$ thus adding to the robustness of the method.

It is to be noted that if the proposed noise vectors which move the system outside the permitted $[V_{min}, V_{max}]$ interval are systematically rejected, the P_k value would

increase at edges by an unwanted excessive amount. This problem is counteracted by adopting the n -fold way [18], i.e., leaving P_k value unchanged whenever a move update attempts to take the system outside the allowed interval.

The histogram $H_k^{*,j}$ is checked after about each $10L$ Monte Carlo (MC) sweeps. When the histogram is flat (flatness criterion is the same as in [16, 17]), the modification factor is reduced to a finer one using the function $f_{j+1} = \sqrt{f_j}$ ($f_{init} = e = 2.7182818$), the histogram is reset and the next iteration of random walk is started where P_k are now modified with the finer modification factor. We continue doing so until the histogram is flat again and then we begin the next Wang-Landau (WL) iteration with a finer f and so on. The above detailed random walk can also be carried out in a parallel fashion by dividing the range $[V_{min}, V_{max}]$ into W partitions and then exploring each partition separately, combining the results in the end.

It is extremely important to determine the optimum $[V_{min}, V_{max}]$ interval with the optimum number of bins since the accuracy and speed of the simulation depend heavily on it. Following is a self adaptive procedure to determine this interval which intrinsically takes into account the code length and the code error correcting capacity. Lines of similarity can be drawn between our procedure of determining the optimum $[V_{min}, V_{max}]$ interval and Domain Sampling Run of [19].

$[V_{min}, V_{max}]$ is initialized to $[0, 1]$ and this interval is divided into 1000 bins. Let the Global Acceptance Ratio (GAR) correspond to the ratio of the number of accepted noise vectors to the number of noise vectors produced in total. We initialize GAR with a value (0.99 in our case). The bins are initialized with $P_k = 1/1000, \forall k = 1, \dots, 1000$. Now the random walk is performed to produce noise vectors for which the corresponding bins are visited with the consequent update of the P_k value. With the bin filling, we start getting rejections for the proposed move. At each step, we calculate the GAR and as soon as we obtain its pre-defined value, the walk is ceased and the farthest bins on either side are detected which were approached by the random walk. These two bins on either side determine the $[V_{min}, V_{max}]$ interval. While determining the interval, the noise vectors produced are not added to the codewords and no decoder runs are performed. Their sole purpose is to locate the bins naturally accessible for the code.

The $[V_{min}, V_{max}]$ interval is to be divided into a suitable number of bins. Our choice of number of bins depends on the bin width, an important parameter for the accuracy control [20]. We define B , a control parameter to determine the optimum number of bins using the relation $L = (V_{max} - V_{min})B$ (rounded off to the nearest integer). The bins are now initialized with $P_k = 1/L, \forall k = 1, \dots, L$.

The random walk is performed to produce noise vectors for which the corresponding bins are visited with the consequent update of the P_k value. The noise vectors are added to the codeword (in a permuted sequence in case of rejection) in the channel and decoding is performed for the noisy received vector. If we do not get errors and we reach a flat histogram, we reiterate over the above two steps by again choosing a natural $[V_{min}, V_{max}]$ interval for $GAR = GAR - \Delta GAR$ where $\Delta GAR = 0.01$. With each step, we increase the number of bins by $B_{i+1} = 1.5B_i$. If we get errors before reaching a flat histogram, we take the current $[V_{min}, V_{max}]$ interval with the optimum number of bins. We then continue on to perform the WL iterations within this optimum interval.

Let P_{err} be the probability that a received word with noise realization $\mathbf{z}$ selected from ρ leads to an error and $P_{err,k}$ the probability that $\mathbf{z}$ leads to an error and falls into bin k . Then

$$P_{err,k} = P_{err|k} P_k \quad (3)$$

$$P_{err} = \sum_{k=1}^M P_{err,k} \quad (4)$$

where $P_{err|k}$ is the conditional probability of an error given that $\mathbf{z}$ falls into bin k . We can approximate $P_{err|k} \approx P_{err|k}^{j_{max}} = \sum_{j=1}^{j_{max}} G_k^{*,j} / \sum_{j=1}^{j_{max}} H_k^{*,j}$ after j_{max} iterations where j_{max} is the iteration when f gets very close to 1 and we stop further refinement of the modification factor. Using (3) and (4), we get P_{err} .

For each and every WL iteration, we calculate the P_{err} value. We stop the random walk when $max_k |(P_{err}^j - P_{err}^{j+1}) / P_{err}^j| < 0.1$. This criterion should be satisfied for 2 consecutive WL iterations so as to avoid all surges and to ensure the convergence of the simulation. In addition, such a stringent criterion ensures enough samples for both histograms thus ensuring a good level of accuracy. The P_{err} value has been chosen as the convergence parameter since it takes into account both noise sample and error histograms over all previous WL iterations.

3 Simulation Results

The standard Digital Video Broadcast with Return Channel via Satellite (DVB-RCS) [10] 8-state, 188 bytes (MPEG-size) duo-binary turbo code was used as a test bench. Maximum possible iterations are 8 with 4 quantization bits. BPSK modulation is employed using symmetric signal levels of +1 and -1 for logical 0s and 1s respectively. An all zeros codeword is transmitted since the code is linear and the noise is symmetric.

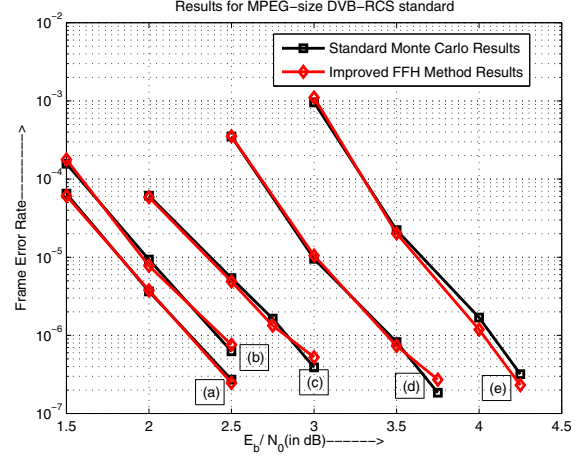


Figure 1: Results for MPEG-size DVB-RCS standard (a) $R = 0.5$ (b) $R = 0.4$ (c) $R = 0.5$ (d) $R = 0.67$ (e) $R = 0.75$

DVB-RCS was chosen as a test bench since its performance is well known. It employs a powerful and complex coding system so the error correcting capacity is very high. The error floors are steep and the Minimum Hamming Distance (MHD) is high. The simulation results for 5 different code rates for MPEG-size DVB-RCS are shown in Fig. 1. The first remark is that our simulation results stick to the MC curves for all code rates and for all FERs.

The efficiency of the FFH method as compared to standard MC can be measured by the simulation gain [1] i.e., the ratio of codewords simulated in MC simulation to those simulated in FFH method. The simulation gain for all code rates for MPEG-size DVB-RCS is given in Table 1.

The gain is good and the result is a considerable reduction in simulation time. They are determined for a high E_b/N_0 value for the particular code rate as indicated by the values within parenthesis in column 1. For FFH method, the decoder runs include codeword simulations for the permuted sequence of noise samples in case of random

Code Rate (E_b/N_0)	MC Codewords	FFH Codewords	Simul. Gain
0.33 (2.5 dB)	367476540	24315960	15.11
0.4 (2.5 dB)	159946900	15247500	10.49
0.5 (3 dB)	256226600	30425510	8.42
0.67 (3.75 dB)	539287910	38629450	13.96
0.75 (4.25 dB)	313316840	30530120	10.26

Table 1: Simulation Gain

walk step rejection. Our current experience suggests that the simulation gain increases with decreasing FER and the FFH method is intricately dependent on the code length, rate and error correcting capacity.

4 Conclusions

In this paper, we have presented an improved version of Fast Flat Histogram Method employing Wang Landau Algorithm in an adaptive importance sampling framework to enhance the simulation speed of FER calculation of FEC codes. The improvements have rendered the FFH method more robust and reliable. We have applied this improved version to a powerful and complex duo-binary turbo code with different code rates showing that the FFH method is generic and is applicable for different families of FEC codes having any length, rate and structure.

5 Acknowledgements

The authors wish to thank Prof. Claude Berrou of ENST-Bretagne for his valuable support related to this work.

References

- [1] R. Srinivasan. “Importance Sampling - Applications in Communications and Detection”. Springer-Verlag, 2002.
- [2] B. Xia and W. E. Ryan. “On importance sampling for linear block codes”. *Proc. IEEE International Conference on Communications (ICC 03)*, pages 2904–2908, 2003.
- [3] C. A. Cole, S. G. Wilson, E. K. Hall, and T. R. Giallorenzi. “A general method for finding low error rates of LDPC codes”. *CoRR*, vol abs/cs/0605051, 2006.
- [4] C. Berrou, A. Glavieux, and P. Thitimajshima. “Near Shannon limit error-correcting coding and decoding: turbo-codes”. In *ICC '93*, pages 1064–1070, Geneva, Switzerland, May 1993.
- [5] U. Gunawardana, K. C. Nguyen, and R. Liyanapathirana. “Fast simulation of turbo codes over AWGN channels”. In *ISCIT '07*, pages 1176–1181, Sydney, Australia, Oct. 2007.
- [6] R. Holzlohner, A. Mahadevan, C. Menyuk, J. Morris, and J. Zweck. “Evaluation of the very low BER of FEC codes using dual adaptive importance sampling”. *IEEE Comm. Letters*, vol 2, Feb. 2005.
- [7] S. Reynal and H. T. Diep. “Fast flat histogram algorithm for generalized spin models”. *Phys. Rev.*, E 72(56710), 2005.
- [8] S. Kakakhail, S. Reynal, D. Declercq, and V. Y. Heinrich. “Fast simulation for the performance evaluation of LDPC codes using Fast Flat Histogram Method”. In *IEEE Sarnoff Symposium*, Princeton NJ, USA, Apr. 2008.
- [9] C. Berrou and M. Jézéquel. “Non binary convolutional codes for turbo coding”. *Elect. Letters*, vol 35(1):39–40, Jan. 1999.
- [10] European Telecommunications Standards Institute. “Interaction channel for satellite distribution systems”. V1.3.1 301 790, ETSI EN, Mar. 2003.
- [11] R. Holzlohner and C. R. Menyuk. “Use of multicanonical Monte Carlo simulations to obtain accurate bit error rates in optical communications systems”. *Opt. Lett.*, vol 28:1894–1896, Oct. 2003.
- [12] B. A. Berg. “Algorithmic aspects of multicanonical Monte Carlo simulations”. *Nucl. Phys. Proc. Suppl.*, vol 63:982–984, 1998.
- [13] W. Krauth. “Introduction to Monte Carlo Algorithms”. *Advances in computer simulation, lectures held at Eotvos summer school in Budapest, Hungary*, 1996.
- [14] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. M. Teller, and E. Teller. “Equation of state calculations by fast computing machines”. *J. Chem. Phys.*, vol 21:1087–1092, 1953.
- [15] G. O. Roberts, A. Gelman, and W. R. Gilks. “Weak Convergence and Optimal Scaling of Random Walk Metropolis Algorithms”. *Ann. Appl. Probab.*, vol 7:110–20, 1997.
- [16] F. Wang and D. P. Landau. “Determining the density of states for classical statistical models: a random walk algorithm to produce a flat histogram”. *Phys. Rev.*, E 64(056101), 2001.
- [17] F. Wang and D. P. Landau. “Efficient, multiple range random walk algorithm to calculate the density of states”. *Phys. Rev. Lett.*, 86(2050), 2001.
- [18] A. B. Bortz, M. H. Calos, and J. L. Lebowitz. *J. Comput. Phys.*, 17(10), 1975.
- [19] A. Troster and C. Dellago. *Phys. Rev.*, E 71(066705), 2005.
- [20] Y. W. Li, T. Wust, D. P. Landau, and H. Q. Lin. *Comput. Phys. Commun.*, vol 177:524–529, 2007.