Supporting Information:

Multiscale Reweighted Stochastic Embedding (MRSE): Deep Learning of Collective Variables for Enhanced Sampling

Jakub Rydzewski* † and Omar Valsson* ‡

†Institute of Physics, Faculty of Physics, Astronomy and Informatics, Nicolaus Copernicus University, Grudziadzka 5, 87-100 Torun, Poland
‡Max Planck Institute for Polymer Research, Ackermannweg 10, D-55128 Mainz, Germany

E-mail: jr@fizyka.umk.pl; valsson@mpip-mainz.mpg.de
S1 Entropy of the Reweighted Feature Pairwise Probability Distribution

Here, we derive eq 11 and eq 12 from Section 2.2.2. The Shannon entropy of a discrete Gaussian kernel is:

\[ H(x_i) = -\sum_j p_{ij}^i \log p_{ij}^i, \]  

(S1)

where \( p_{ij}^i \) are matrix elements of the \( i \)th row of the reweighted feature probability distribution:

\[ p_{ij}^i = \frac{\tilde{K}_{\epsilon_i}(x_i, x_j)}{\sum_k \tilde{K}_{\epsilon_k}(x_i, x_k)}, \]  

(S2)

for the reweighted Gaussian kernel \( \tilde{K}_{\epsilon_i}(x_i, x_j) = r(x_i, x_j)K_{\epsilon_i}(x_i, x_j) \), where \( r(x_i, x_j) \) is a reweighting factor. We define a row-wise normalization constant of eq S2 as \( \bar{p}_i = \sum_k \tilde{K}_{\epsilon_i}(x_i, x_k) \). By plugging eq S2 to eq S1, we arrive at:

\[ H(x_i) = -\frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \log \left( \frac{1}{\bar{p}_i} \tilde{K}_{\epsilon_i}(x_i, x_j) \right) \]

\[ = -\frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \log \left( \frac{1}{\bar{p}_i} r(x_i, x_j)K_{\epsilon_i}(x_i, x_j) \right) \]

\[ = -\frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \left( \log r(x_i, x_j) + \log K_{\epsilon_i}(x_i, x_j) - \log \bar{p}_i \right) \]

\[ = -\frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \left( \log r(x_i, x_j) - \epsilon_i \|x_i - x_j\|_2^2 - \log \bar{p}_i \right), \]  

(S3)
which further translates to:

\[
H(x_i) = \frac{1}{\bar{p}_i} \left( \epsilon_i \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \|x_i - x_j\|^2_2 - \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \log r(x_i, x_j) + \log \bar{p}_i \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \right),
\]

(S4)

and finally, the entropy of the reweighted feature probability distribution is given by:

\[
H(x_i) = \frac{1}{\bar{p}_i} \left( \epsilon_i \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \|x_i - x_j\|^2_2 - \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \log r(x_i, x_j) \right) + \log \bar{p}_i \\
= \frac{1}{\bar{p}_i} \epsilon_i \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \|x_i - x_j\|^2_2 - \frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \log r(x_i, x_j) + \log \bar{p}_i. 
\]

(S5)

The full Shannon entropy vector is \( H = [H(x_1), H(x_2), \ldots, H(x_N)]^T \).

If weights are of exponential form (like in WT-MetaD), \( w(x_i) = e^{\beta V(x_i)} \), then \( H_V(x_i) \) may be further reduced:

\[
H_V(x_i) = -\frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \log r(x_i, x_j) \\
= -\frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) \log \sqrt{e^{\beta V(x_i)} e^{\beta V(x_j)}} \\
= -\frac{\beta}{2\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) (V(x_i) + V(x_j)) \\
= -\frac{\beta}{2} \left( \frac{1}{\bar{p}_i} \sum_j \tilde{K}_{\epsilon_i}(x_i, x_j) V(x_j) + V(x_i) \right). 
\]

(S6)
S2 Kullback-Leibler Divergence Loss for a Full Set of Training Data

Here, we present a derivation of the Kullback-Leibler (KL) divergence between two pairwise probability distributions:

\[ M = (p_{ij})_{1 \leq i, j \leq N} \quad \text{and} \quad p_{ij} = \frac{1}{N_p} \sum_{p} p_{ij}^p \quad (S7) \]

and

\[ Q = (q_{ij})_{1 \leq i, j \leq N} \quad \text{and} \quad q_{ij} = \frac{(1 + \|s_i - s_j\|_2^2)^{-1}}{\sum_k (1 + \|s_i - s_k\|_2^2)^{-1}}, \quad (S8) \]

as is described in Sections 2.2.3 and 2.2.4 in the main text.

For discrete probability distributions \( p \) and \( q \) defined on the same probability space \( \chi \), the KL divergence of \( p \) and \( q \) is defined as:

\[ D_{KL}(p \| q) = \sum_{x \in \chi} p(x) \log \left( \frac{p(x)}{q(x)} \right), \quad (S9) \]

where \( q(x) = 0 \) implies \( p(x) = 0 \), and the base of the logarithm is arbitrary. For pairwise probability matrices, eq S9 needs to be rewritten as the row-wise average of the KL divergences:

\[ D_{KL}(M \| Q) = \frac{1}{N_b} \sum_{i=1}^{N_b} D_{KL}(M_i \| Q_i), \quad (S10) \]

where \( M_i \) and \( Q_i \) are the \( i \)th rows of \( M \) and \( Q \), respectively, and \( N_b \) is the number of rows in the matrices or the batch size. Then, if we take a sum of the partial KL divergence losses for all \( B \) batches of the data using eq S10, we arrive at the KL divergence for a full set of the...
training data:

\[
D_{\text{KL}}(M||Q) = \sum_{b=1}^{B} \frac{1}{N_b} \sum_{i=1}^{N_b} \sum_{j=1}^{N_b} \sum_{i \neq j} p_{ij} \log \left( \frac{p_{ij}}{q_{ij}} \right)
\]

\[
= \frac{1}{N_b} \sum_{b=1}^{B} \sum_{i=1}^{N_b} \sum_{j=1}^{N_b} p_{ij} \log \left( \frac{p_{ij}}{q_{ij}} \right).
\]  
(S11)
S3 Description of Well-Tempered Farthest-Point Sampling (WT-FPS)

Well-tempered farthest-point sampling\(^1\) (WT-FPS) bridges two selection landmark selection schemes, weighted random sampling\(^2\) and farthest point sampling (FPS).\(^3\) Random sampling with weights should, in principle, give an equilibrium representation.\(^1\) However, there might be issues with capturing metastable states lying higher in free energy. FPS ignores the underlying data density and results in a subset that is most broadly spread out in the feature space.

In WT-FPS, we select \(N\) landmarks from a dataset that consists of \(M\) feature samples. We start by selecting \(K = \sqrt{NM}\) samples using FPS, and treat them as the centers of a Voronoi diagram. The remaining samples are assigned to their nearest center. Next, we assign a weight to each center, which is obtained by summing over the weights of all the samples that belong to a center, \(\omega_i = \sum_k w(x_k)\), where \(i\) is the label of a center. We define a weight of the \(i\)th center as:

\[
d_i = \left( \frac{\omega_i}{\sum_{j=1}^{K} \omega_j} \right)^{1/\alpha},
\]

where we introduce a broadening parameter \(\alpha\) as in the well-tempered distribution. To select the landmarks, we proceed by randomly selecting a center according to its weight \(d_i\). We then choose at random a feature sample from this center (taking in account the weights of feature samples \(w(x_k)\)). We repeat this procedure until we have \(N\) landmarks.

If \(\alpha = 1\), we recover weighted random sampling. However, by choosing \(\alpha > 1\), we increasingly favor the algorithm to ignore the underlying probabilities and pick a more uniformly distributed set of landmarks. In the limit of \(\alpha \to \infty\), we obtain FPS. Thus, by using \(\alpha > 1\) that is not too large, we can obtain a landmark set that also represents higher-lying metastable states while not being far from the equilibrium distribution.
**Effective Landmark CV Distribution for Weight-Tempered Random Sampling**

Starting from eq 16 in the main manuscript:

\[ P_{\alpha}(x) = \frac{\int dR \left[ w(R) \right]^{1/\alpha} \delta [x - x(R)] P_V(R)}{\int dR \left[ w(R) \right]^{1/\alpha} P_V(R)} \]  

(S13)

for the landmark feature distribution in weight-tempered random sampling, we want to derive the distribution in CV space, \( P_{\alpha}(s) \).

Setting \( x = s \), and assuming that the weights are given by \( w(R) = e^{\beta V(s(R))} \) (i.e., come from a CV-based enhanced sampling method), we obtain:

\[
P_{\alpha}(s) \propto \int dR \left[ e^{\frac{1}{\alpha} \beta V(s(R))} \right]^{1/\alpha} \delta [s - s(R)] P_V(R)
\]

\[
= \int dR \ e^{\frac{1}{\alpha} \beta V(s)} \delta [s - s(R)] P_V(R)
\]

\[
= e^{\frac{1}{\alpha} \beta V(s)} \int dR \delta [s - s(R)] P_V(R)
\]

\[
= e^{\frac{1}{\alpha} \beta V(s)} P_V(s),
\]

(S14)

where in the first step we use the property of the delta function to sift out the dependence of \( R \) in \( e^{\frac{1}{\alpha} \beta V(s)} \), and in the last step the definition of \( P_V(s) \) from eq 2 in the main manuscript. As \( P_V(s) \propto P(s)e^{-\beta V(s)} \), we obtain two equivalent ways of writing \( P_{\alpha}(s) \), using either the equilibrium distribution \( P(s) \) or the biased distribution \( P_V(s) \):

\[
P_{\alpha}(s) = \frac{e^{\frac{1}{\alpha} \beta V(s)} P_V(s)}{\int ds \ e^{\frac{1}{\alpha} \beta V(s)} P_V(s)}
\]

\[
= \frac{e^{(\frac{1}{\alpha} - 1) \beta V(s)} P(s)}{\int ds \ e^{(\frac{1}{\alpha} - 1) \beta V(s)} P(s)}.
\]

(S15)

These are the equations for a general CV-based bias potential.

In the case of WT-MetaD, the bias potential at convergence is given by \( V(s) = -\left(1 - \frac{1}{\gamma}\right) F(s) \)
(eq 3 in the main manuscript). Inserting this for $V(s)$ and using $P(s) \propto e^{-\beta F(s)}$ leads to:

$$P_\alpha(s) \propto e^{-\left(\frac{1}{\alpha} - 1\right)\left(1 - \frac{1}{\gamma}\right) F(s)} e^{-\beta F(s)}$$

$$= e^{-\left(\frac{1}{\alpha} - \frac{1}{\alpha\gamma} + \frac{1}{\gamma}\right) F(s)}$$

$$= \left[e^{-\beta F(s)}\right]^{\left(\frac{1}{\alpha} - \frac{1}{\alpha\gamma} + \frac{1}{\gamma}\right)}$$  \hspace{1cm} (S16)

or

$$P_\alpha(s) = \frac{\left[P(s)\right]^{\left(\frac{1}{\alpha} - \frac{1}{\alpha\gamma} + \frac{1}{\gamma}\right)}}{\int ds \left[P(s)\right]^{\left(\frac{1}{\alpha} - \frac{1}{\alpha\gamma} + \frac{1}{\gamma}\right)}}$$

$$= \frac{\left[P(s)\right]^{1/\tilde{\alpha}}}{\int ds \left[P(s)\right]^{1/\tilde{\alpha}}}.$$  \hspace{1cm} (S17)

where we introduce an effective tempering parameter $\tilde{\alpha}$ as:

$$\tilde{\alpha} = \left(\frac{1}{\alpha} - \frac{1}{\alpha\gamma} + \frac{1}{\gamma}\right)^{-1} = \frac{\gamma\alpha}{\gamma + \alpha - 1}. \hspace{1cm} (S18)$$

Figure S1: Effective alpha $\tilde{\alpha}$ as a function of $\alpha$ and $\gamma$. 
We perform the cluster analysis on the MRSE embedding shown in Figure 7 by fitting a Bayesian Gaussian mixture to the data. We run this procedure with the number of components (clusters) equal to 3, where each component has its own general covariance matrix. The clustering takes the periodicity of \((\Phi, \Psi)\) into account. We use the \texttt{scikit-learn} machine-learning library\textsuperscript{S4} to perform this analysis.
Figure S2: Free energy profiles for the PT embeddings of alanine dipeptide (top row) and alanine tetrapeptide (bottom row). The profiles are obtained using using histogramming (“Discrete”, black lines) using 200 grid points and kernel density estimation (“KDE”, blue lines) using bandwidth values estimated based on Silverman’s rule (see Section 3.4).
S7 Loss Function Learning Curves

Figure S3: Results for the Müller-Brown potential (Section 4.1). Learning curves from the minimization of the Kullback-Leibler loss function for the MRSE embeddings shown in Figure S1. The upper row shows results obtained by incorporating weights into the training (red lines, labeled reweighted in Figure S1), while the lower row shows results obtained by not incorporating weights into the training (blue lines, labeled not reweighted in Figure S1). The y-axis is shown as a logarithmic (base 10) scale.

Figure S4: Results for alanine dipeptide in vacuum at 300 K (Section 4.2). Learning curves from the minimization of the Kullback-Leibler loss function for the MRSE embeddings shown in Figure 7(c) in the main paper. The upper row shows results obtained by incorporating weights into the training (red lines, labeled reweighted in Figure 7(c)), while the lower row shows results obtained by not incorporating weights into the training (blue lines, labeled not reweighted in Figure 7(c)). The y-axis is shown as a logarithmic (base 10) scale.
Figure S5: Results for alanine tetrapeptide in vacuum at 300 K (Section 4.3). Learning curves from the minimization of the Kullback-Leibler loss function for the MRSE embeddings shown in Figure 11(b-c) in the main paper. The upper row shows results obtained by incorporating weights into the training (red lines, labeled reweighted in Figure 11(c)), while the lower row shows results obtained by not incorporating weights into the training (blue lines, labeled not reweighted in Figure 11(b)). The y-axis is shown as a logarithmic (base 10) scale.

S8 Additional Embeddings for the Müller-Brown Potential

Figure S6: Results for the Müller-Brown potential (Section 4.1). FESs for the MRSE embeddings obtained from biased simulation data. We show results obtained from WT-MetaD simulations using different bias factors $\gamma$ (3, 4, 5, 7). The upper row (rewighted) shows results obtained by incorporating weights into the training via a reweighted feature pairwise probability distribution, while the lower row shows results obtained by not incorporating weights (not reweighted). The embeddings are not rotated using the Procrustes algorithm to fit to the MB potential, unlike those from Figure 4 in the main text. The units for the MRSE embeddings are arbitrary and thus not shown.
Figure S7: Results for alanine dipeptide in vacuum at 300 K (Section 4.2). Heavy-atom distances used as the input features for the MRSE embeddings. The first column shows the histograms of the unprocessed distances (black), while the second column depicts standardized distances. We discarded all distances with a variance below $2 \times 10^{-4}$ nm$^2$ from the training set (shown in red). We can see that the remaining 21 distances (shown in blue) used for the training are mostly binomial.
Figure S8: Results for alanine dipeptide in vacuum at 300 K (Section 4.2). FESs for the MRSE embedding obtained from a WT-MetaD simulation data (γ = 5), where we use the weight-tempered random sampling for landmark selection with α = \{1, 2, 3, 5, 10, 20\} (see Figure 5). In the cases for α > 2, we obtain inadequate mappings of the metastable states. The units for the MRSE embeddings are arbitrary and thus not shown.
S11  Alanine Dipeptide Embeddings for $\alpha = 2$ in WT-FPS

Figure S9: Results for alanine dipeptide in vacuum at 300 K (Section 4.2). FESs for the MRSE embedding obtained from a WT-MetaD simulation data, where we use the WT-FPS landmark selection with $\alpha = 2$. The units for the MRSE embeddings are arbitrary and thus not shown.
Figure S10: Results for alanine dipeptide in vacuum at 300 K (Section 4.2). FESs for MRSE embeddings trained using the PT simulation data obtained using different random seed values. The seed affects both the landmark selection and the shuffling of the landmarks during the training. The embeddings differ mostly in how they are rotated. The units for the MRSE embeddings are arbitrary and thus not shown.
Figure S11: Results for alanine tetrapeptide in vacuum at 300 K (Section 4.3). The conditional FESs (eq 17) as a function of $\Phi_1$ and $\Phi_2$ for the (a) A and (b) B minima of $\Phi_3$ shown in Figure 10(c) in the main paper. These are the same conditional FESs as shown in Figure 10(a).
Figure S12: Results for alanine tetrapeptide in vacuum at 300 K (Section 4.3). Free energy profiles for the $\Phi$ dihedral angles. We show free energy profiles for the PT simulation (black lines), the WT-MetaD simulation ($\gamma = 5$, red line) and two bias-exchange metadynamics simulations ($\gamma = 5$ and $\gamma = 10$, green and blue, respectively). The free energy profiles for the WT-MetaD and the bias-exchange simulations are obtained by summing up the Gaussians using `sum_hills` command line tool in PLUMED. The free energy profiles for the PT simulation are obtained using histogramming with 200 grid points. In the bias-exchange simulations, we employ 6 replicas each one biasing one of the dihedral angles ($\Phi, \Psi$) using WT-MetaD with bias factors 5 or 10. We attempt exchanges between neighboring replicas every 10 ps.

As expected, the PT simulation does not capture the higher-lying regions. We can see that all the simulations agree well for the minima. Furthermore, the WT-MetaD and bias-exchange simulations are in good agreement for the whole curve apart from the highest barrier regions. Therefore, we conclude that the WT-MetaD and PT simulations are converged.
Figure S13: Results for alanine tetrapeptide in vacuum at 300 K (Section 4.3). Time series of the dihedral angles for the WT-MetaD simulations ($\gamma = 5$) biasing $\Phi$ and for the 300 K replica in the PT simulation.
References

(S1) Ceriotti, M.; Tribello, G. A.; Parrinello, M. Demonstrating the transferability and the descriptive power of sketch-map. *J. Chem. Theory Comput.* **2013**, *9*, 1521–1532.

(S2) Tribello, G. A.; Gasparotto, P. *Biomolecular Simulations: Methods and protocols*; Springer, 2019; p 453.

(S3) Hochbaum, D. S.; Shmoys, D. B. A best possible heuristic for the $k$-center problem. *Math. Oper. Res.* **1985**, *10*, 180–184.

(S4) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. scikit-learn: Machine learning in Python. *J. Mach. Lear. Res.* **2011**, *12*, 2825–2830.