MESHCHERYAKOV LABORATORY OF INFORMATION TECHNOLOGIES

Sampling of Integrand for Integral Calculation Using Shallow Neural Network

A. Ayriyan, H. Grigorian, V. Papoyan

Meshcheryakov Laboratory of Information Technologies, Joint Institute for Nuclear Research, Dubna, Russia

Neural network method for integration

Let's spouse to have a continuous real function $f : \mathbb{R}^n \to \mathbb{R}$ over the domain S, then the integral is

$$I[f] = \int_S f(\boldsymbol{x}) d\boldsymbol{x} \,, \tag{1}$$

where S is a convex, bounded set in $\mathbb{R}^n$.

According to the universal approximation theorem [1] and theorem 2 from [4], we approximate the function $f(x)$ using a single-layer neural network

$$\widehat{f}(x) = b_2 + W_2^T \sigma(b_1 + W_1 x) = b^{(2)} + \sum_{j=1}^{k} w_j^{(2)} \sigma\left(b_j^{(1)} + \sum_{i=1}^{n} w_{ij}^{(1)} x_i\right), \tag{2}$$

with a logistic sigmoid activation function so that this network can be integrated analytically over an arbitrary domain S by the following formula from [4]:

$$\widehat{I}(f, \boldsymbol{\alpha}, \boldsymbol{\beta}) = I[\widehat{f}] = b_2 \prod_{i=1}^{n} (\beta_i - \alpha_i) + \sum_{j=1}^{k} w_j^{(2)} \left[\prod_{i=1}^{n} (\beta_i - \alpha_i) + \frac{\Phi_j}{\prod_{i=1}^{n} w_{ij}^{(1)}} \right], \tag{3}$$

where

$$\Phi_j = \sum_{r=1}^{2^n} \xi_r \mathrm{Li}_n \left(- \exp\left[- b_j^{(1)} - \sum_{i=1}^{n} w_{ij}^{(1)} \ell_{i,r} \right] \right). \tag{4}$$

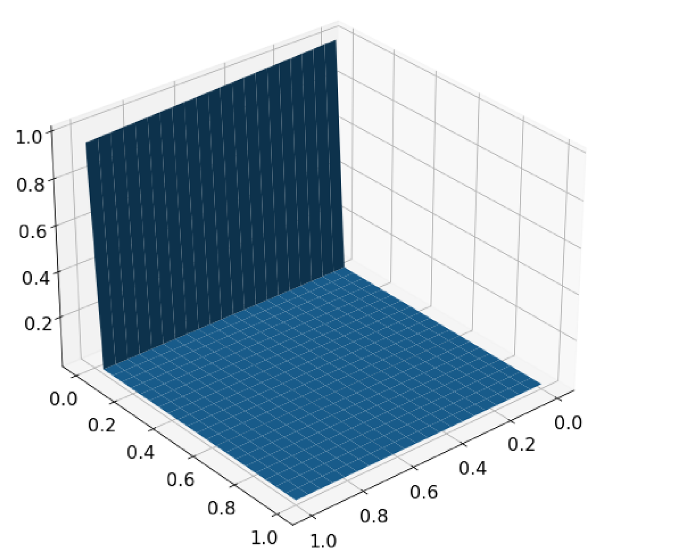
Here ξ_r is the sign in front of the r-th term of the sigmoid integration, and $\ell_{i,r}$ is the corresponding integration limit for the i-th dimension, defined by the corresponding formulas:

$$\xi_r = \prod_{d=1}^{n} (-1)^{\lfloor r/2^{n-d} \rfloor} \,, \tag{5}$$

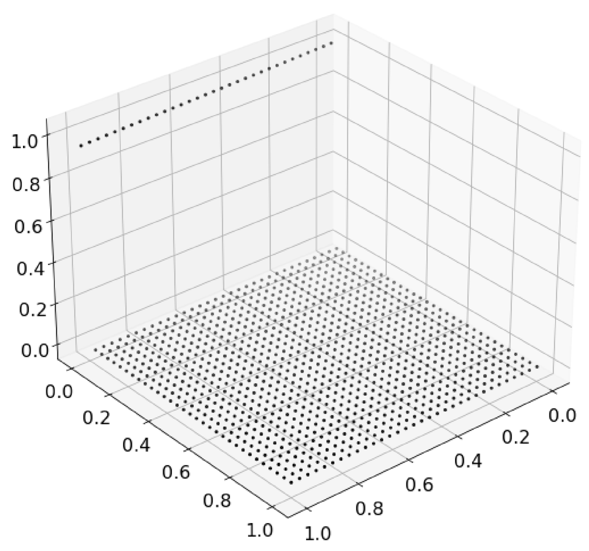
$$\ell_{i,r} = \begin{cases} \alpha_i, & \text{if } \lfloor r/2^{n-i} \rfloor \text{ is even,} \\ \beta_i, & \text{otherwise.} \end{cases} \tag{6}$$

Sampling of an Integrand

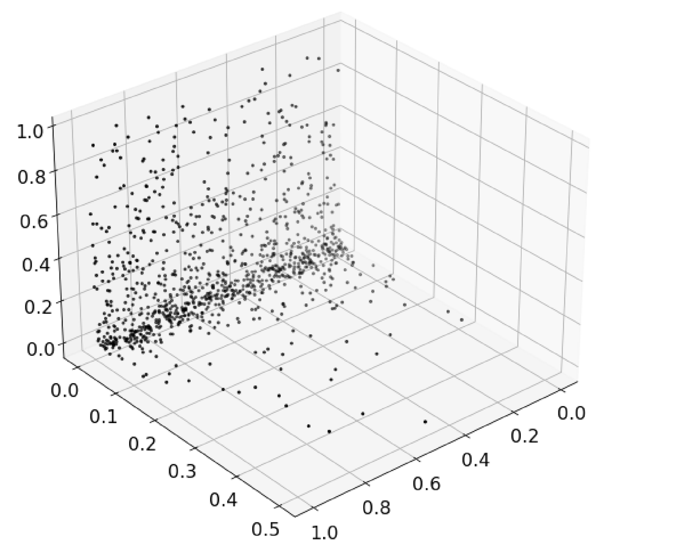
For an integrand sampling the hybrid method is proposed, in which a part of the training set D_{MH} is generated by applying the Metropolis-Hastings algorithm [3]. The other part of the sample D_{UG} consists of the nodes of the uniform grid. Thus, $D = D_{MH} \cup D_{UG}$. We introduce ρ as a ratio between an amount of MH-points to N: $\rho = |D_{MH}| / N$. An example of generating points using a hybrid method and the Metropolis-Hastings algorithm is shown in the figures below:



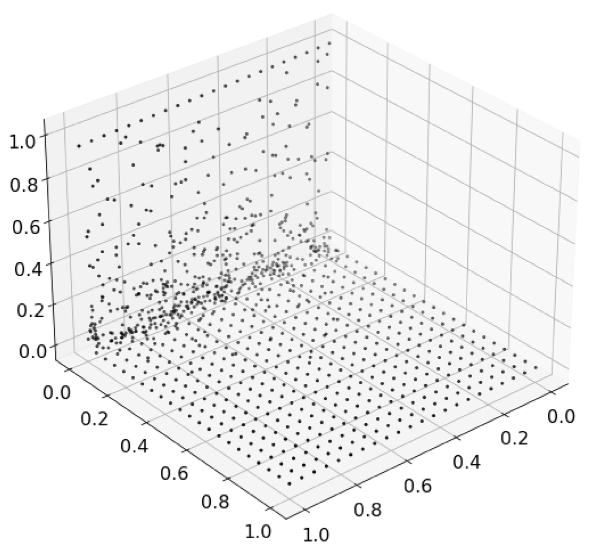
(a) The example of corner peak function.



(b) Uniform grid sampling.



(c) Metropolis-Hastings sampling



(d) Hybrid sampling with $\rho = 0.5$.

Figure: An example of various ways to sample a function.

Numerical Evaluation

We have performed the hybrid sampling for three classes of parameterized functions from [2]:

Oscillatory function:

$$f^{(1)}(\boldsymbol{x}) = cos(2\pi u_1 + \sum_{i=1}^{n} c_i x_i) \tag{7}$$

Corner Peak Function:

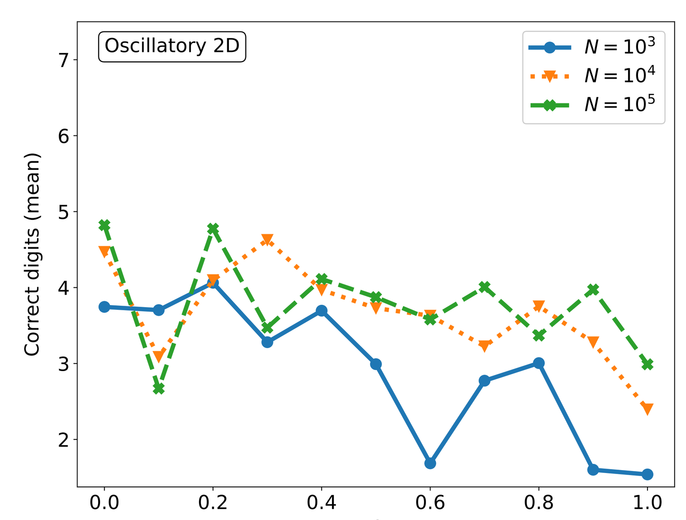
$$f^{(2)}(\boldsymbol{x}) = (1 + \sum_{i=1}^{n} c_i x_i)^{-(n+1)} \tag{8}$$

Continuous Function:

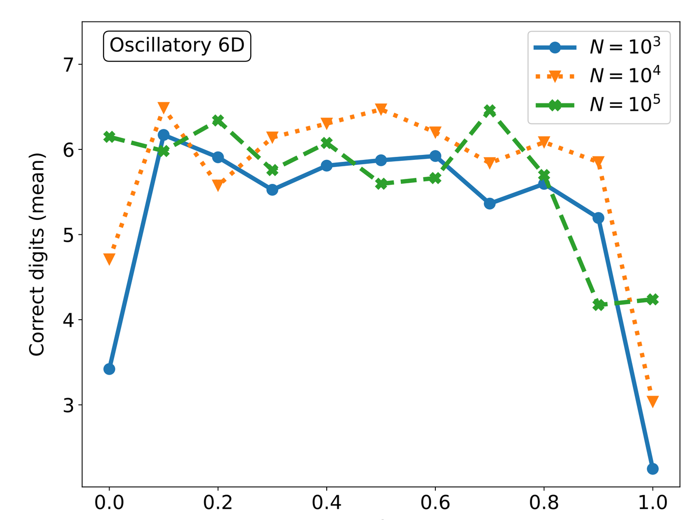
$$f^{(3)}(\boldsymbol{x}) = \exp(- \sum_{i=1}^{n} c_i |x_i - u_i|) \tag{9}$$

The integration accuracy is evaluated by determining the number of correct digits between the analytical solution and the numerical values:

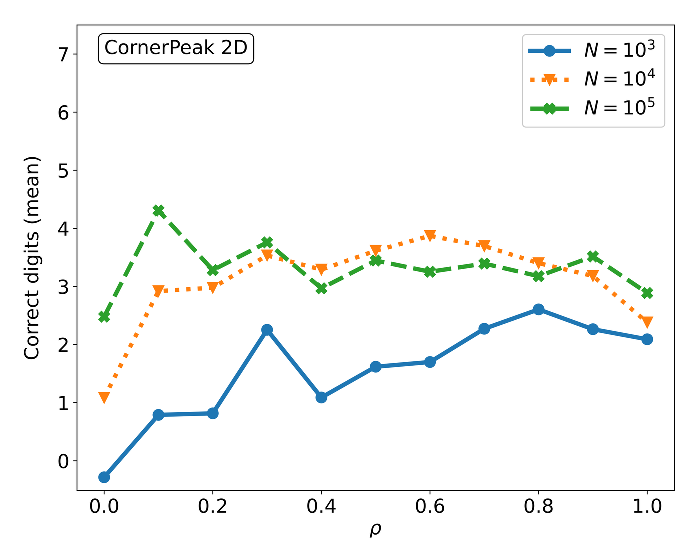
$$CD(I, \widehat{I}) = - \log_{10} \left| \frac{I - \widehat{I}}{I} \right|. \tag{10}$$



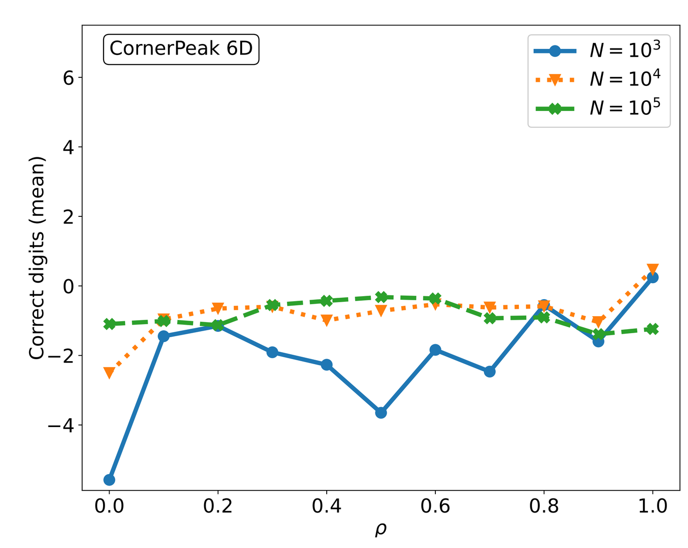
(a) Results for Oscillatory function in 2D domain



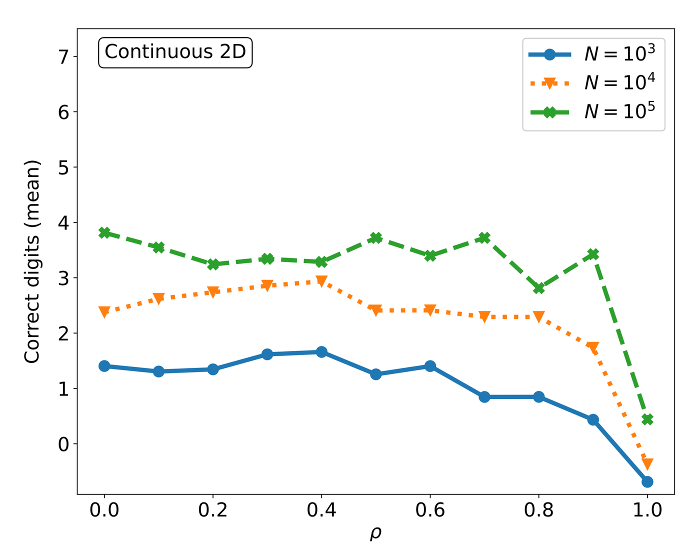
(b) Results for Oscillatory function in 6D domain



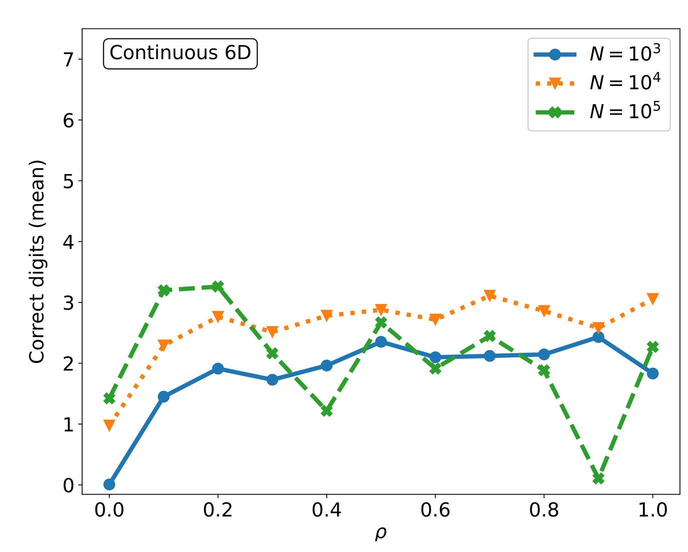
(c) Results for Corner Peak Function in 2D domain



(d) Results for Corner Peak Function in 6D domain



(e) Results for Continuous Function in 2D domain



(f) Results for Continuous Function in 6D domain

Figure: Results of integration with hybrid sampling for (7)–(9) functions. Each point on the graphs is the average value of 20 integrals for a given ρ. For large values of $N = 10^5$, the number of trials was reduced to 5 due to the long training time.

References

[1] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. doi: 10.1007/BF02551274.

[2] A. Genz. *A Package for Testing Multiple Integration Subroutines*, pages 337–340. Springer Netherlands, Dordrecht, 1987. doi: 10.1007/978-94-009-3889-2_33.

[3] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57 (1):97–109, 1970. doi: 10.1093/biomet/57.1.97.

[4] S. Lloyd, R. A. Irani, and M. Ahmadi. Using neural networks for fast numerical integration and optimization. *IEEE Access*, 8:84519–84531, 2020. doi: 10.1109/ACCESS.2020.2991966.