Application of Neural Network Approach to Numerical Integration

Gregory A. Shipunov, student of gr. 4181[1]
supervisors: Oksana I. Streltsova,[1, 2] Yury L. Kalinovsky[1, 2]

[1] Dubna State University, 19 Universitetskaya St, Dubna, 141980, Russian Federation
[2] Joint Institute for Nuclear Research, 6 Joliot-Curie St, Dubna, 141980, Russian Federation

2025

Table of Contents

Table of Contents

Problem Statement 1

This work is dedicated to developing the software for numerical calculations in domain of modelling of physical processes for NICA collider. Numerical modeling of physical processes involves studying of the particle properties based on Bethe-Salpeter equation:

$$\Gamma(q,P) = -\frac{4}{3}\int \frac{d^4p}{(2\pi)^4} D(q-p)\gamma_\alpha S_1 \Gamma(q,P) S_2 \gamma_\alpha . \tag{1}$$

The vertex function $\Gamma(p,P)$ depends on the relative momentum p, and the total momentum of the bound state P. $S_i(p)$ are the dressed quark propagator in the Euclidean space:

$$S_i(p_i) = \frac{1}{i(p_i\cdot\gamma)+m_i} \tag{2}$$

The momenta $p_i = p + q_i$, $q_i = b_i P$, $i = 1,2$ with $b_1 = -m_1/(m_1+m_2)$, $b_2 = m_2/(m_1+m_2)$, m_i are the constituent quark masses.

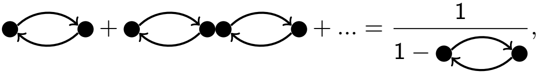
Problem Statement 2

Equation (1) contains the interaction kernel $D(q-p)$ which describes the effective gluon interaction within a meson. We consider the rank -1 separable model

$$D(q-p) = D_0 F(q^2) F(p^2)\,, \tag{3}$$

where D_0 is the coupling constant and the function $F(p^2)$ is related to scalar part of Bethe - Salpeter vertex function. We employ $F(q^2)$ in the Gaussian form $F(p^2) = e^{-p^2/\lambda^2}$ with the parameter λ which characterizes the finite size of the meson.

This separable Anzatz of the interaction kernel (3) allows to write the meson observables in the term of the polarization operator (the buble diagram):



(4)

where

$$\to \int_0^\infty \frac{dp}{2\pi^4} F(p^2) \frac{1}{[(p+q_1)^2 + m_1^2][(p+q_2)^2 + m_2^2]}. \tag{5}$$

To calculate these integrals we use the equations:

$$\frac{1}{(p+q_i)^2 + m_i^2} = \int_0^\infty dt \{\exp(-t[(p+q_i)^2 + m_i^2])\}, \tag{6}$$

$$F(p^2) = \int_0^\infty ds \{\exp(-sp^2) F(s)\}. \tag{7}$$

The case of more than two mesons is described using triple and quadruple integrals.

This way the following integral equation is produced:

$$\int_0^1 d\alpha\{\alpha^a(1-\alpha)^b\}\int_0^\infty dt\{\frac{t^m}{(1+t)^n}F[z_0]\} \equiv I(a,b,m,n;F[z_0]), \tag{8}$$

$$F[z_0] = \exp(-2z_0),$$

$$z_0 = tD + \frac{t}{1+t}R^2,$$

$$D = \alpha_1(b_1^2P^2 + m_1^2) + \alpha_2(b_2^2P^2 + m_2^2),$$

$$R^2 = (\alpha_1^2 b_1^2 + \alpha_2^2 b_2^2 + 2\alpha_1\alpha_2 b_1 b_2)P^2,$$

$$b_1 = -\frac{m_1}{m_1+m_2}, b_2 = \frac{m_2}{m_1+m_2},$$

$$\alpha_1 = \alpha, \alpha_2 = 1-\alpha.$$

This integral is used in multiple calculations in the problem and its value describes the form of the mesons. The calculations require multiple (50 times and more) integrations of (8).

Problem Statement 5

Let a continuous real function $f(\mathbf{x})$ be defined as $f : \mathbb{R}^n \to \mathbb{R}$. Let Ω be a compact subset of $\mathbb{R}^n$ and let G be a bounded convex subset of Ω. Let, also, $\mathbf{x}$ be a vector of n dimensions in Ω. Than

$$I(f) = \int_G d\mathbf{x} f(\mathbf{x}), \tag{9}$$

is a definite integral of f across set G.

Therefore, the problem is to get the $I(f)$ value calculated with use of neural network approach. The neural network model will be used to approximate the function $f(\mathbf{x})$[1]. Than this model will be used to calculate the approximate integral value:

$$\hat{I}(f) = \int_G d\mathbf{x} \hat{f}(\mathbf{x}), \tag{10}$$

[1] S. Lloyd, R. A. Irani, M. Ahmadi, Using neural networks for fast numerical integration and optimization, IEEE Access 8 (2020) 84519–84531.

Problem Statement 6

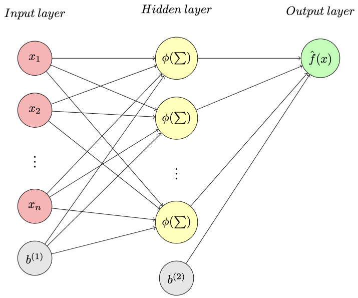


Figure 1: The MLP structure used in the neural network approach

The MLP structure will have logistic sigmoid as activation function on the hidden layer:

$$\phi(z) = \frac{1}{1 + \exp(-z)}. \tag{11}$$

With $\phi(z)$ defined, network structure's mathematical form is:

$$\hat{f}(x) = b^{(2)} + \sum_{j=1}^{k} w_j^{(2)} \phi(b_j^{(1)} + \sum_{i=1}^{n} w_{ji}^{(1)} x_i). \tag{12}$$

We can apply following substitution[2] here:

$$-Li_0(-\exp(z)) = \frac{1}{1+\exp(-z)} = \phi(z), \tag{13}$$

where $Li_0(u(z))$ is a Jonquière's function or the polylogarithm of order 0.

[2] S. Lloyd, R. A. Irani, M. Ahmadi, Using neural networks for fast numerical integration and optimization, IEEE Access 8 (2020) 84519–84531.

Let number of dimensions $n = 1$. With substitution (13) equation (12) can be integrated over given boundaries $[\alpha, \beta]$ to produce the following numerical integration formulae for 1-dimensional case[3]:

$$\hat{I}(f) = \int_{\alpha}^{\beta} dx \left(b^{(2)} + \sum_{j=1}^{k} w_j^{(2)} \phi(b_j^{(1)} + w_{1j}^{(1)} x) \right) =$$

$$= b^{(2)}(\beta - \alpha) + \sum_{j=1}^{k} w_j^{(2)} \left((\beta - \alpha) + \frac{\Phi_j}{w_{1j}^{(1)}} \right), \tag{14}$$

$$\Phi_j = Li_1(-\exp[-b_j^{(1)} - w_{1j}^{(1)} \alpha]) - Li_1(-\exp[-b_j^{(1)} - w_{1j}^{(1)} \beta]). \tag{15}$$

Formulae (14-15) can be extrapolated to higher dimensions.
Thus we have got the numerical integration method.

[3] S. Lloyd, R. A. Irani, M. Ahmadi, Using neural networks for fast numerical integration and optimization, IEEE Access 8 (2020) 84519–84531.

Table of Contents

Results 1

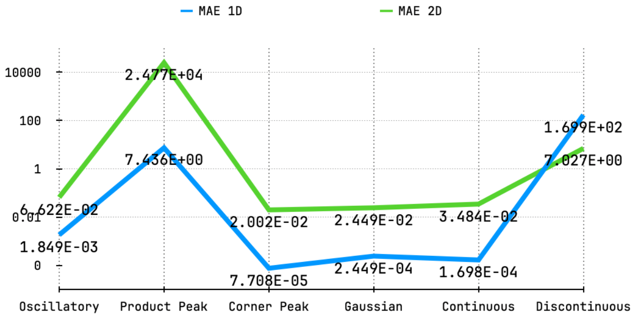


Figure 2: The MAE between neural network numerical integration values and quad numerical integration values

Figure 2 depicts the mean absolute errors of integration of Alan Genz's package[4] 1D and 2D functions compared to results calculated using *scipy.integrate.quad* fucntion of Python programming language. The neural network model had hidden layer size was k = 100.

[4] A. Genz, Testing multidimensional integration routines, in: Proc. of international confer- ence on Tools, methods and languages for scientific and engineering computation, 1984, pp. 81–94.

Results 2

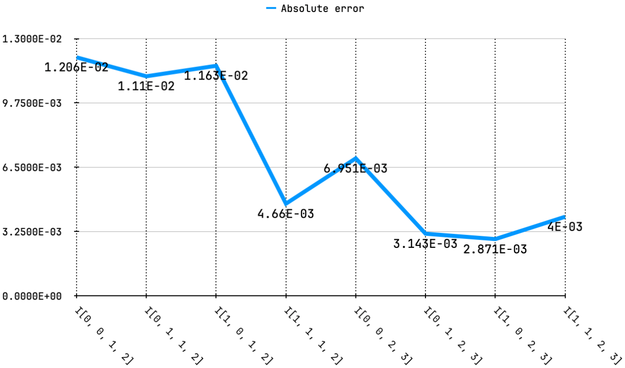


Figure 3: The absolute errors between numerical integral values calculated using neural network approach and FORTRAN function qdag

Figure 3 depicts the absolute errors of integration of equation (8) compared to results calculated previously using *qdag* function of FORTRAN programming language. The neural network model had hidden layer size was k = 100.

Benefits of using neural network approach

- The integration of similar function across different boundaries time efficiency.
- The integration of functions of high-dimensions.

Table of Contents

Future plans

- Improve accuracy of integration.
- Implement higher dimensions numerical integration functionality.
- Consider other ways to use neural networks for numerical integration.

Thank You for Your Attention!