Enabling Non-Linear Quantum Operations through Variational Quantum Splines

Matteo Antonio Inajetovic[1], Filippo Orazi[1⋆], Antonio Macaluso[2], Stefano Lodi[1], and Claudio Sartori[1]

[1] University of Bologna, Bologna, Italy
matteo.inajetovic@studio.unibo.it
{filippo.orazi2, stefano.lodi, claudio.sartori}@unibo.it
[2] German Research Center for Artificial Intelligence (DFKI), Saarbruecken, Germany
antonio.macaluso@dfki.de

Abstract. One of the major issues for building a complete quantum neural network is the implementation of non-linear activation functions in a quantum computer. In fact, the postulates of quantum mechanics impose only unitary transformations on quantum states, which is a severe limitation for quantum machine learning algorithms. Recently, the idea of QSplines has been proposed to approximate non-linear quantum activation functions by means of the HHL. However, QSplines rely on a problem formulation to be represented as a block diagonal matrix and need a fault-tolerant quantum computer to be correctly implemented.
This work proposes two novel methods for approximating non-linear quantum activation functions using variational quantum algorithms. Firstly, we develop the variational QSplines (VQSplines) that allow overcoming the highly demanding requirements of the original QSplines and approximating non-linear functions using near-term quantum computers. Secondly, we propose a novel formulation for QSplines, the Generalized QSplines (GQSplines), which provide a more flexible representation of the problem and are suitable to be embedded in existing quantum neural network architectures. As a third meaningful contribution, we implement VQSplines and GQSplines using Pennylane to show the effectiveness of the proposed approaches in approximating typical non-linear activation functions in a quantum computer.

Keywords: Quantum Machine Learning · Quantum Neural Networks · Quantum Computing

1 Introduction

Quantum computers are machines that leverage the properties of quantum mechanics to store and process information. Although a potential quantum advantage has already been shown in different domains, such as quantum chemistry [1],

⋆ Corresponding author

multi-agent systems [2,3], it is still unclear whether quantum computation can be used efficiently in machine learning (ML).

The majority of the approaches proposed in the field of Quantum Machine Learning (QML) relies on using hybrid quantum-classical optimization to train parameterized quantum circuits to perform typical ML tasks. Although these techniques represent the most promising attempt to leverage near-term quantum technology, it is still unclear whether they can outperform classical algorithms.

One class of QML algorithms gaining momentum in recent years is Quantum Neural Networks (QNN) which try to emulate the behavior of classical neural networks using parametrized quantum circuits. The key feature of classical neural networks is the ability to capture complex patterns by applying multiple non-linear activation functions. On the other side, QNN models utilize quantum kernels to explicitly map data into a high-dimensional space and learn the complex patterns in data. Although this approach has shown promising results, it is not a credible alternative capable of providing a robust quantum advantage over classical methods. An alternative strategy to unlock the full potential of quantum computing in ML is to implement classical neural networks using the properties of quantum computing as computational resources to obtain a robust speed-up. However, the postulates of quantum mechanics forbid non-unitary operations on quantum states, which is a strong limitation for QML algorithms. Thus, the ability to approximate non-linear activation functions in a quantum computer is essential to obtain a quantum advantage of QML algorithms over their classical counterpart. In this paper, we propose two novel approaches for quantum activation functions based on QSplines [4], which leverage hybrid quantum-classical optimization and are suitable for near-term quantum computation.

The remainder of the paper is structured as follows: a brief discussion of related work is provided in section 2. Section 3 gives a highlight of the contribution of the paper while section 4 contains the detailed methodological approach. In section 5 the results of the experiments are presented and then discussed in section 6. The paper concludes with a summary of achievements and future work.

2 Related Works

Recently, several attempts have been made to provide a routine for quantum activation functions and overcome the constraint of the unitarity of quantum operations. The quantum Splines (QSplines) [4] rely on classical B-Spline regression models that aim to find the optimal set of parameters to minimize the Residual Sum of Squares in a ridge regression problem where observed variables are augmented with polynomials. However, the QSplines suffer from several drawbacks and limitations: the use of the HHL [5] as a subroutine imposes running the algorithm on a perfectly error-corrected quantum computer, and it is not suitable to be executed on near-term quantum devices. Moreover, the output of the QSplines requires a post-processing step to obtain the value of the non-linear function which is stored in the quantum state. Furthermore, QSplines require

ad-hoc formulation for the basis expansion matrix in terms of a diagonal block matrix that is hardly generalizable.

Other works on quantum activation functions rely on repeat-until-success technique [6]. In this case, the most significant limitation is that the input must be in the range $\left[0, \frac{\pi}{2}\right]$, which is a severe constraint for real-world problems. Recently, the problem of non-linear approximation has been considered by means of a Quantum Perceptron [7]. The proposed quantum algorithm produces the output of a non-linear activation function leveraging an iterative computation of all the powers of the inner product up to an order d. The main drawback of this approach is the number of qubits required, which depends linearly on the number of input features and the order of the polynomial. in particular, given an n-dimensional feature vector and a degree of the polynomial d, the idea of the Quantum Perceptron requires $n + d$ qubits. Experimentally, this approach shows good results with a degree of polynomial $3 \leq d \leq 10$ for 1 dimensional feature vector.
In [8] the authors try to solve the non-linear functions problem by means of a quantum nonlinear processing unit in a variational circuit.

3 Contribution

This work proposes two alternative approaches for quantum activation functions: Variational QSpline (VQsplines) and Generalized Quantum Splines (GQSplines). The VQSplines adopt the problem formulation of the QSplines [4], translating it in the context of hybrid quantum-classical computation using the VQLS as a quantum routine for matrix inversion and uses the quantum dot product to calculate the value of the non-linear activation function. The advantage of this approach is twofold: on one side, the use of the HHL is avoided allowing the VQSplines to be executed using near-term quantum technology. On the other side, the use quantum dot product provides a quantum state encoding the value of the activation function without requiring a post-processing step.

The second main methodological contribution is a novel formulation for QSplines, the GQSpline, that relies on a more flexible problem formulation in terms of basis expansion and allows obtaining an end-to-end quantum routine that can approximate any non-linear activation function. Importantly, the GQSplines can be adopted as a sub-routine in existing quantum neural networks encoding into the amplitudes of a quantum states the value of the non-linear function.

Importantly, the proposed algorithms encode data (basis expansion of the input features and the value of the non-linear activation function) into the amplitudes of quantum states. This means that the qubit complexity (i.e., the number of qubits required) scales logarithmically with respect to the input size which is a significant improvement with respect to existing approaches whose number of qubits is extremely demanding.

4 Methods

In this section, the methodological contributions are presented. We describe two different approaches for quantum splines: the *VQSplines* adopts the piece-wise formulation of QSplines but replaces the HHL with the Variational Quantum Linear Solver (VQLS) [9]. As a consequence, we obtain a quantum algorithm suitable for NISQ devices capable of estimating any non-linear function. Second, we propose the *GQSplines* model, a novel formulation of QSplines that allows us to obtain an end-to-end quantum algorithm for non-linear approximation suitable for existing quantum neural networks.

4.1 Preliminaries

Quantum Splines
Spline functions are smoothing methods for modeling the relationships between variables, typically adopted either as a visual aid in data exploration or for estimation purposes [10]. The underlying idea is to use linear models in which the input features are augmented with the *basis expansions*. Technically, splines are constructed by dividing the sample data into sub-intervals delimited by breakpoints, also referred to as knots. A fixed degree polynomial is then fitted in each of the segments, thus resulting in piecewise polynomial regression.

While the formulation in terms of truncated basis functions is conceptually simple, its numerical and computational properties are not very attractive. For this reason, in practice, the *B-splines* parametrization [11] is adopted. This generates a block design matrix where the *sparsity* is constant and depends on the degree of the polynomial fitted in each local interval. Given a sequence of knots $\xi_1, \xi_2, \cdots, \xi_T$, we fit a line in each interval $[\xi_k, \xi_{k+1}]_{k=1,\cdots,T-1}$ without derivability constraints.

$$\tilde{\boldsymbol{y}} = \boldsymbol{S}\boldsymbol{\beta} \rightarrow \begin{pmatrix} \tilde{y}_1 \\ \tilde{y}_2 \\ \cdots \\ \tilde{y}_K \end{pmatrix} = \begin{pmatrix} S_1 & 0 & \cdots & 0 \\ 0 & S_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & S_K \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \cdots \\ \beta_K \end{pmatrix}, \tag{1}$$

where $\tilde{y_k}$ contains the function evaluations in ξ_k and ξ_{k+1}, β_ks are the spline coefficients and $\boldsymbol{S}_{(2K)\times(2K)}$ is a block diagonal matrix with each block S_k that represents the basis expansions in the k-th interval. Therefore, solving the linear system in Eq. (1) allows computing the splines coefficients, which serve to approximate non-linear functions encoded in the vector $\tilde{\boldsymbol{y}}$.

The idea of the Quantum Splines (*QSplines*)[4] is to adopt the B-spline formulation in the context of quantum computation to approximate non-linear functions. In particular, the computation of the QSplines is performed in three steps. First, the HHL computes the spline coefficients for the k-th interval encoded into the quantum state $|\beta_k\rangle$. Second, $|\beta_k\rangle$ interacts with the quantum state $|x_k\rangle$ encoding the input in the k-th interval via quantum interference through the swap-test [12]. This allows generating the state $|f_k\rangle$ which encodes the estimate of the

non-linear function evaluated in x_k; Third, $|f_k\rangle$ is measured and post-processed to obtain y_k.

Although QSplines allow overcoming the limitation of unitary operation on quantum states, their applicability is very limited since the use of the HHL as a subroutine requires a massive number of error-corrected qubits to be executed. Furthermore, in the current formulation, the final quantum state is obtained using the swap test, whose results are not directly encoded in the amplitude and need a post-processing step to be calculated. All these factors forbid the adoption of QSplines in current models of quantum neural networks, which run on a limited set of noisy qubits.

Variational Quantum Linear Solver
An alternative quantum algorithm to solve a linear system of equations is the Variational Quantum Linear Solver (*VQLS*) [9]. Specifically, the VQLS solves a linear system of equations using a variational hybrid quantum-classical approach which is suitable for near-term quantum devices. Given a matrix A and a state vector $|b\rangle$, the VQLS prepares the state $|x\rangle$ such that:

$$A|0\rangle = A|x\rangle \propto |b\rangle . \tag{2}$$

The matrix A is defined as a linear combination of unitary matrices A_l:

$$A = \sum_{l=0}^{L} A_l c_l \tag{3}$$

where c_l are complex numbers. With this input, the VQLS generates the state $|x\rangle$ employing an ansatz for the gate sequence $V(\theta)$ such that $|x(\theta)\rangle = V(\theta)|0\rangle$. The parameters θ are input to a quantum computer, which prepares $|\theta\rangle$ and runs an efficient quantum circuit that estimates a cost function $C(\theta)$. The value of $C(\theta)$ from the quantum computer is returned to the classical computer which then adjusts θ (via a classical optimization algorithm) in an attempt to reduce the cost. This process is iterated many times until one reaches a termination condition of form $C(\theta) < \gamma$, at which point we say that $\theta = \theta_{\text{opt}}$.

Once the cost function is minimized, the Ansatz $V(\theta_{opt})$ with the optimal parameters θ_{opt} prepares the normalized state $|x(\theta_{\text{opt}})\rangle$ which approximates $|x\rangle$:

$$V(\theta_{\text{opt}})|0\rangle = |x(\theta_{\text{opt}})\rangle . \tag{4}$$

Notice that this description of the problem is coherent with the original VQLS. However, in the case of QSplines, the linear system of equation in Eq.(1) represents the target solution (the B-spline coefficients) as $|\beta\rangle$, the A_l circuit encodes the matrix S, and $|y\rangle$ replaces $|b\rangle$.

4.2 VQSplines: Variational Algorithm for QSplines

The idea behind the *VQSplines* is to implement the QSplines in the context of hybrid quantum-classical computation and provide an efficient method to

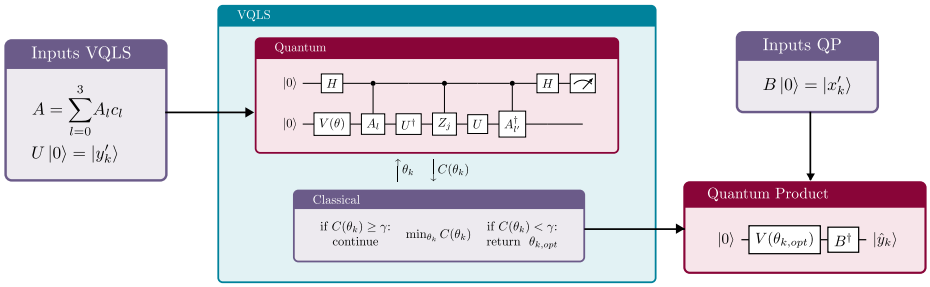


Fig. 1: The VQSplines architecture. The quantum part of the VQLS is optimized classically and then $\theta_{k,opt}$ is used in the quantum inner product to compute $|\hat{y}_k\rangle$

estimate non-linear functions using near-term quantum computers. In particular, the VQSplines replace the HHL with the VQLS and adopt the quantum inner product [13] to generate a quantum state encoding the value of the non-linear target function, avoiding the use of the swap-test.

In practice, the computation of the VQSplines is performed in two steps. First, the VQLS estimates the spline coefficients for the k-th interval:

$$S_k |\beta'_k\rangle = |y'_k\rangle \xrightarrow{VQLS} |\beta'_k\rangle = \beta'_{k,0} |0\rangle + \beta'_{k,1} |1\rangle \approx S_k^{-1} |y'_k\rangle \tag{5}$$

In order to obtain the quantum state $|\beta'_k\rangle$, each 2×2 linear system requires the training of a parametrized quantum circuit which needs three different quantum gates: a unitary $V(\theta_k)$ implemented with a Pauli rotation gate R_y that allows generating a quantum state whose amplitudes encode the B-spline coefficients, once the optimal set of parameters $\theta_{k,\text{opt}}$ is obtained. Notice that, as for the QSplines, the entire problem is decomposed into K 2×2 linear systems, and each set of coefficients can be encoded using a single qubit; the second set of quantum gates are the unitaries A_l, which encode the information related to the S_k matrix. In particular, the matrix S_k is decomposed by means of a linear combination of known unitary matrices as follows:

$$S_k = \begin{bmatrix} 1 & a \\ 1 & b \end{bmatrix} = \sum_{l=0}^{3} A_l c_l = I c_0 + X c_1 + Z c_2 + R_y(3\pi) c_3 \tag{6}$$

where the coefficients of the linear combination are initialized as follows:

$$c_0 = (b+1)/2; \qquad c_1 = (a+1)/2; \qquad c_2 = (1-b)/2; \qquad c_3 = (a-1)/2;$$

Third, the unitary U encodes the classical vector y_k into the amplitude of $|y'_k\rangle$ [14]. The QSplines model for the k-th interval is depicted in Figure 1.

As a result of the optimization process, we obtain a variational circuit that creates a quantum state encoding the solution of the linear system:

$$V(\theta_{k,opt}) |0\rangle = |\beta'_k\rangle = \beta'_{k,0} |0\rangle + \beta'_{k,1} |1\rangle \tag{7}$$

The second step of the VQSplines computes the inner product between the basis expansion of the input $|x'_k\rangle$ and the B-splines coefficients $|\beta'_k\rangle$. To this end, the quantum inner product [13] generates a quantum state whose amplitudes encode the value of the non-linear function:

$$B^\dagger V(\theta_{k,opt})\,|0\rangle_n = a_o\,|0\rangle_n + \sum_{i=1}^{N-1} a_i\,|i\rangle_n \tag{8}$$

$$a_0 = \langle 0|\,B^\dagger V(\theta_{k,opt})\,|0\rangle = \langle x'_k|\,BB^\dagger\,|\beta'_k\rangle = \langle x'_k|\beta'_k\rangle = \hat{y}_k \tag{9}$$

where B is the amplitude encoding quantum routine [14] of the basis expansion of the input x_k. The quantum circuit of the non-linear target function for the k-th interval is built as shown in Figure 1 (*Quantum Inner product*).
These steps are repeated for each sub-interval and the solutions of the quantum system in Eq. (1) are obtained. Therefore, as for QSplines, once $|\beta'\rangle$ is known, the VQSplines require $\mathcal{O}(K)$ repetitions of the quantum inner product circuit to calculate a spline function with K knots.

Importantly, the use of the quantum inner product allows one to directly obtain the value of the target function as amplitudes of a quantum state and does not require any post-processing as for QSplines.

4.3 GQSplines: Generalized Quantum Splines

The original formulation of the QSplines and VQSplines relies on a block diagonal B-Spline matrix which allows decomposing the entire problem into K sub-problems that can be solved independently. However, this approach does not allow the estimation of a non-linear function in a full quantum manner, which is a requirement for embedding a quantum activation function into a quantum neural network. For this reason, we propose the *Generalized Quantum Splines* (GQSplines), a novel approach that relies on a more general formulation of the B-splines and allows estimating the value of a non-linear function with a single quantum circuit without problem decomposition.

Given the recursive definition of B-Spline [11] and the related basis expansion with knots list $\xi = [\xi_1, .., \xi_i, \xi_{i+1}, .., \xi_T]$, a non linear function f can be estimated using the observed values $Y = \{y_1, .., y_K\}$ given the inputs $X = \{x_1, .., x_K\}$. In this case, the linear system of equations describing the relation between the estimates of the activation function Y, the matrix S and the spline coefficients β is the following:

$$Y_{K\times 1} = S_{K\times K}\beta_{K\times 1} \implies \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{bmatrix} = \begin{bmatrix} B_{1,d}(x_1) & \dots & B_{l,d}(x_1) \\ B_{1,d}(x_2) & \dots & B_{l,d}(x_2) \\ \vdots & \ddots & \vdots \\ B_{1,d}(x_K) & \dots & B_{l,d}(x_K) \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_K \end{bmatrix}, \tag{10}$$

where d is the degree of the B-spline, T is number of knots and $l = T - d - 1$.

Nonetheless, to adopt the VQLS (or the HHL) for solving the linear system of equations, the basis expansion matrix S has to be hermitian and, therefore, square and non-singular. The GQSplines formulation imposes the matrix S to be Hermitian, i.e., $K = l = T - 2$, and adopts a quantum linear solver to find the set of optimal parameters $|\beta\rangle$[3]. Assuming to fit a linear function in each interval (i.e., $d = 1$), the linear system of the GQSplines is:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \vdots \\ y_{k-1} \\ y_K \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & \dots & 0 & 0 \\ 0 & 1-x_2 & x_2 & \dots & \dots & 0 \\ \dots & 0 & 1-x_3 & x_3 & \dots & \dots \\ \dots & \dots & 0 & 1-x_4 & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & 1-x_{K-1} & x_{K-1} \\ 0 & 0 & \dots & \dots & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \vdots \\ \beta_{K-1} \\ \beta_K \end{bmatrix}. \tag{11}$$

which leads to the following quantum linear system of equations:

$$S\,|\beta\rangle = |Y\rangle \tag{12}$$

where $|\beta\rangle$ and $|Y\rangle$ are two quantum states that encode in their amplitudes the vectors Y and β. Importantly, the normalization constraint of quantum states $|\beta\rangle$ and $|Y\rangle$ imposes rescaling the values of the target variable such that $\langle Y|Y\rangle = \langle\beta|\beta\rangle = 1$.

With a particular focus on the VQLS as a quantum linear solver, the size of the linear system in Eq. (12) is K which gives us the number of qubits required to find the optimal coefficients. Precisely, since the VQLS requires the vectors to be encoded as quantum state, the number of qubits scale logarithmically in the number of knots K, which is directly related to the quality of the fitting of the curve (the higher is K, the better the fitting is). This exponential scaling with respect to the number of inputs is a significant improvement when compared to other quantum approaches for quantum activation functions [7] whose number of qubits scales linearly with the size of the inputs.

Given the linear system in Eq. (10), the GQSplines proceed in two steps. Firstly, the coefficients $|\beta'\rangle$ are generated using the VQLS (or the HHL):

$$|\beta'\rangle = VQLS(S, Y) = \sum_{i=1}^{K} \beta_i\,|i\rangle\,. \tag{13}$$

Thus, quantum state $|\beta'\rangle$ interacts via interference with $|x'\rangle$ representing the basis expansion of x by means of the quantum inner product, as for VQSplines.

In the case of GQSplines, the VQLS circuit requires $n + 1$ qubits, where $K = 2^n$ is the number of knots. Therefore, the Hadamard test [15] employs n qubits to implement the operators ($V(\theta)$,A and U) and one for the ancilla qubit. Furthermore, if $d = 1$, we end up with a diagonal block matrix which allows

[3] It is also possible to define the Hermitian matrix H from S as $H = \begin{pmatrix} 0 & S \\ S^\dagger & 0 \end{pmatrix}$.

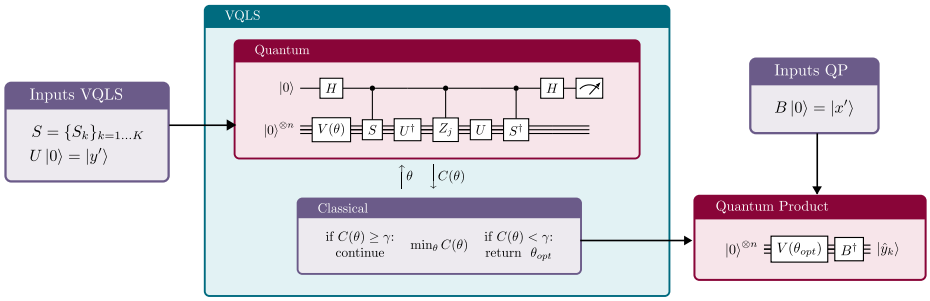


Fig. 2: GQSplines architecture. The quantum part of the VQLS is optimized classically to obtain θ_{opt} that is then used in the quantum inner product to compute $|\hat{y}_k\rangle$. All $y'_k \in y$ can be approximated with the results of a single optimization of the VQLS.

decomposing the matrix S (Eq. (3)) in terms of quantum gates in such a way to define the quantum circuit of the VQLS efficiently. Specifically, we can act independently on each interval to define the k-th matrix decomposition as follows:

$$S_k = \begin{bmatrix} 1-a & a \\ 0 & 1-b \end{bmatrix} = \sum_{l=0}^{3} A_l c_{k,l} = I c_{k,0} + X c_{k,1} + Z c_{k,2} + R_y(3\pi) c_{k,3} \quad (14)$$

where the coefficients of the linear combination are computed as:

$$c_0 = 1 - a/2 - b/2; \qquad c_1 = a/2; \qquad c_2 = (b-a)/2; \qquad c_3 = a/2$$

This method allows the S matrix to be efficiently decomposed and obtain the linear combination of quantum gates required for the VQLS ([9]). The operator U is realized by amplitude encoding state preparation [14]. Still, the normalization of Y is required. With this new formulation, we are able to solve a singular linear system and encode all the spline coefficients β in a unique quantum state by implementing the Ansatz only once. Subsequently, this state is used to compute the inner product with the k-th row of the matrix S (encoded through the routines B_i) and return the $\hat{y}_k$ estimates describing $\hat{Y}$. The workflow of GQSplines is depicted in Figure 2.

5 Evaluation

In this section, we implement and evaluate the proposed VQSplines and GQSplines for approximating three typical non-linear activation functions usually adopted in classical neural networks (*sigmoid*, *elu*, and *relu*) and the *sine* function[4]. The algorithms are implemented through the use of Pennylane [5](0.27.0 version).

[4] Though the primary objective of QSplines is to embed non-linearity into quantum neural networks, they can serve to approximate other types of non-linearity.

[5] https://pennylane.ai/

5.1 Experimental Settings

Data Generation We consider *sigmoid*, *elu*, and *relu* activation functions and the *sine* function to showcase the non-linearity of our models. Eq. (15) shows the formulation of the functions tested for VQSplines, which is coherent with the experimental setting of original QSplines [4]:

$$\begin{aligned} \text{ReLu}(x) &= max(0, x) \qquad \text{Elu}(x) = \begin{cases} x \text{ if } x > 0 \\ 0.3(e^x - 1) \text{ otherwise} \end{cases} \\ \text{Sigmoid}(x) &= \frac{1}{1 + e^{-4x}} \qquad f_{\sin}(x) = sin(\pi x) \end{aligned} \tag{15}$$

In the case of GQSplines, the input features and the value of the non-linear activation function are encoded into the amplitude of a quantum state. In this respect, the same functions are normalized in the interval $[0, 1]$. Thus, for the VQSplines the approximation of *Elu*, *ReLu*, and *Sigmoid* is carried out according to the experiments of the original QSplines paper, while *sin* allows for comparison of other approaches [7].

Metrics As discussed in the previous sections, different models are tested using different renormalizations. In order to perform a fair comparison between all methods, we consider the Normalized Root Mean Squared Errors (NRMSE) are defined as:

$$\textbf{NRMSE} = \frac{\sqrt{N^{-1} \sum_{i=0}^{N} (\hat{y}_i - y_i)^2}}{y_{max} - y_{min}}. \tag{16}$$

The NMRSE allows for a robust comparison that investigates the quality of the fitting independently of the scale of the target variable and the number of points used to approximate it.

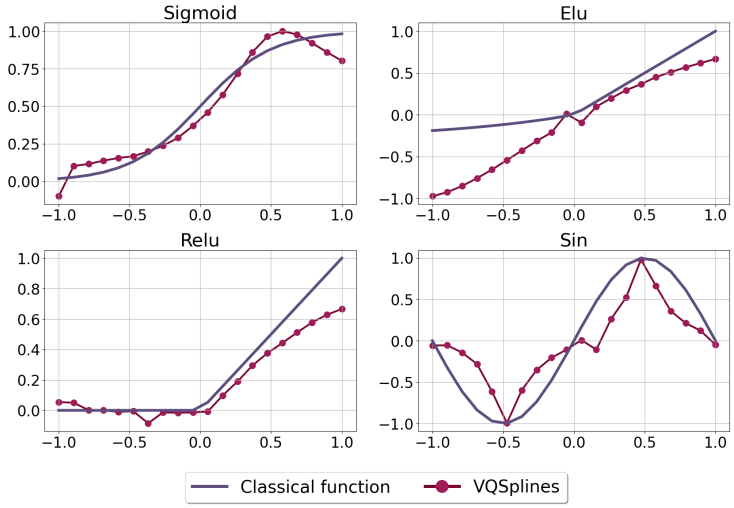


Fig. 3: VQSpline experimental results. The model is able to approximate the function and obtain non-linearity. Each point j is computed as a product between $|x_j\rangle$ and the spline coefficient $|\beta'_j\rangle$

5.2 Results

We test the VQSplines by training the VQLS and computing the quantum inner product to produce the final estimates $\tilde{Y}$ for curve described in Section 5.1. The results are shown in Figure 3.

We can see that in all the cases, the VQSplines can capture the non-linearity of the curves as expected. For *relu* and *sigmoid*, we have a good approximation, while the same cannot be observed for the *elu* and *sine*. In particular, the approximation of the curves deteriorates at the boundaries of the activation functions with input close to -1 or 1.

Considering the results of the GQSplines, the experiments are performed on the four functions normalized into the interval $[0, 1]$. For each curve, we show the fitting results in Figure 4, using 16 knots and 4 qubits. In this case, the approximation quality seems to be sensitively better with respect to the VQSplines. However, while increasing the number of knots allows a better fitting, it implies using a larger quantum state which flatters the curve since the vectors $|Y\rangle$ and $|\beta\rangle$ are normalized to 1.

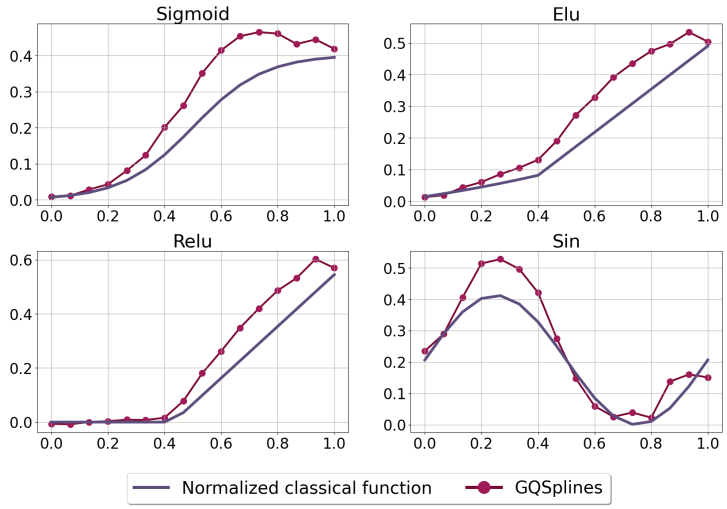


Fig. 4: GQSplines experimental results. The model is able to approximate and emulate the trend of all 4 normalized functions. Each point is computed as the product of the j-th row of S and the B-spline coefficients $|\beta'\rangle$.

We can observe that there is a slight tendency of the GQSplines to overestimate the target function. Nevertheless, the method can capture the non-linearity of the curves while approximating their, especially in the case of the activation function. These results are achieved using only 4 qubits which is a significant improvement with respect to the experiments of other proposed approaches in the literature.

In order to make a fair comparison between QSplines, VQSplines and GQSplines, we calculate the NRMSE for the four curves and report the results in Table 1. We can observe that both proposed methods outperform the original QSplines. Although

VQSplines perform better in terms of NRMSE than GQSplines for the *relu* and *sigmoid*, the differences between the two methods are minimal. The same happens when looking at the approximation of *elu* and *sigmoid*, where the GQSplines perform better.

Model	Knots	Elu	Relu	Sigmoid	Sin
Qsplines	20	0.4874	0.5240	0.1589	—
GQSpline	16	**0.0126**	0.0111	0.0156	**0.0099**
VQSpline	20	0.1278	**0.0069**	**0.0067**	0.0677

Table 1: NRMSE on each function for the proposed models and the baseline. The best approximation for each function is highlighted, and we can see that our model provides a considerable increase in performance with respect to QSplines.

6 Discussion

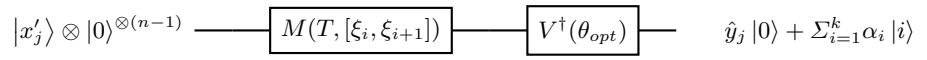
GQSplines and VQSplines allow estimating the value of non-linear functions by means of parameterized quantum circuits. Both methods outperform the original proposal of QSplines in terms of fitting and require a significantly less number of qubits with respect to other existing approaches for quantum activation functions [7,6].

In the case of VQSplines, the quality of the estimates in each interval depends to the specific condition number of linear systems, that for activation functions increases as one moves towards the extremes of the function. In fact, if the matrix S in Eq. (12) is ill-conditioned (i.e., it has a high condition number) the quality of the solution provided by a quantum solver is negatively affected. In particular, the larger the condition number is, the higher the probability of obtaining numerical errors in solving the linear system [9] since ill-conditioned systems converge slowly [16]. This implies that subsystems where the condition number is high lead to worse estimated coefficients.

The GQSplines algorithm uses a problem formulation that allows obtaining an end-to-end quantum algorithm representing the spline coefficients with a single linear system of equations. In this case, results are more stable, the shape of the curve is well-approximated, and the non-linearity is well-captured. However, the limitation is that the curves must be normalized, and the normalization depends on the number of Knots which defines the size of the linear system. To tackle this problem, one can consider adopting amplitude amplification [17] to stretch the shape of the curve and increase the amplitudes to correct the normalization effects. Nonetheless, increasing the number of Knots (which scales logarithmically with the number of qubits of the VQLS) allows to build smaller intervals and potentially improves the fitting quality. This is a significant improvement with respect to existing quantum approaches [7] where the input scale linearly with the number of qubits.

Furthermore, to use of GQSplines as a method for quantum activation functions requires the ability to calculate the target variable $\hat{y_j}$ from a generic input x_j. In this regard, we have to devise a mapping circuit M that creates a vector representing the basis expansion of x_j. Then, in order to obtain $\hat{y}_j$ we have to apply the dot-product between the basis expansion of x_j and the β coefficients, both encoded as a quantum state. As before, the output $\hat{y}_j$ will be encoded by the quantum state calculated by the

following circuit:



(17)

Note that the only difference between this approach and the one described for the GQSplines is that here the Ansatz ($V\dagger$) is transposed and applied in reverse order with respect to the encoding of the basis expansion of x_j. This procedure allows the proposed GQSplines method to produce a non-linear function f for a given input x_j which approximates the target variable y. This approach can be adopted in existing quantum neural networks such as the quantum Single Layer Perceptron [18,19] or the more general MAQA Framework [20], which require an end-to-end quantum routine storing the input-output of the non-linear activation function into the amplitudes of a quantum state.

Table 2 summarizes the properties of QSplines [4] and the two proposed methods. The VQSplines overcome the issue of performing the post-processing step and replace the HHL with the VQLS. However, they still require an ad-hoc problem decomposition with a diagonal block matrix. The GQSplines overcome this issue by defining a single linear system of equations representing the splines function.

Method	End-to-end	Linear Problem Solver	Quantum Inner Product	Post
QSplines	×	HHL	Swap Test	×
VQSplines	×	VQLS	Inner Product	✓
GQSplines	✓	VQLS	Inner Product	✓

Table 2: Comparison between QSplines [4], VQSplines, and GQSplines. The table describes whether the approach allows obtaining an end-to-end quantum routine for non-linear approximation (*end-to-end*), the quantum linear solver in use (VQLS [9] or HHL [5]), and whether the results are directly encoded into a quantum state or not depending on the use either the swap-test [21] (not accessible) or the quantum inner product [13] (accessible). This dichotomy is described by the column *Post*.

7 Conclusion

Quantum Machine Learning (QML) has recently attracted ever-increasing attention and promises to impact various applications by leveraging quantum computational power and novel algorithmic models, such as Variational Algorithms. However, the field is still in its infancy, and its practical benefits need further investigation. One of the major issues in building a complete quantum neural network is the limitation of unitary, and therefore linear, operations on quantum states. In this work, we move toward a non-linear approximation of quantum activation functions using parametrized quantum circuits. In particular, we showed that it is possible to circumvent the constraint of unitarity in quantum computation by presenting an efficient version of the QSplines, whose implementation falls within the context of fault-tolerant quantum computation. The proposed methods do not require a fault-tolerant quantum subroutine (such as

HHL) and allow the approximation of non-linear functions using near-term quantum technology.

The contribution of this paper is twofold. The first part proposes the Variation Quantum Splines (VQSplines), an implementation of the fault-tolerant QSplines [4] in the context of hybrid quantum-classical computation. Additionally, Generalized QSplines (GQSplines) are formulated and discussed. The benefit of this new formulation lies in the ability to be generalizable with respect to the structure of the spline matrix. The GQSplines adopt a new basis expansion matrix formulation, avoid the problem decomposition, and allow for tackling the problem of the matrix inversion in an end-to-end manner, with one single linear system and the number of qubits that scales logarithmically with the number of knots. Furthermore, the GQSplines are more efficient with respect to the number of qubits required with respect to existing quantum approaches for quantum activation functions. Experiments showed that both methods outperform the QSplines and can efficiently capture the non-linearity of typical activation functions adopted in classical neural networks.

Future work will be dedicated to embedding the GQSplines as a subroutine in existing quantum neural networks to leverage the properties of quantum computing in typical machine learning tasks where the non-linearity is crucial.

Acknowledgments

This work has been partially funded by the German Ministry for Education and Research (BMB+F) in the project QAI2-QAICO under grant 13N15586.

Code Availability

All code to generate the data, figures, analyses, and additional details on the experiments are available at https://github.com/inajetovic/Variational-Quantum-Splines.

References

1. Xiao Yuan. A quantum-computing advantage for chemistry. *Science*, 369(6507):1054–1055, 2020.
2. Supreeth Mysore Venkatesh, Antonio Macaluso, and Matthias Klusch. Bilp-q: quantum coalition structure generation. In *Proceedings of the 19th ACM International Conference on Computing Frontiers*, pages 189–192, 2022.
3. Supreeth Mysore Venkatesh, Antonio Macaluso, and Matthias Klusch. Gcs-q: Quantum graph coalition structure generation. *arXiv preprint arXiv:2212.11372*, 2022.
4. Antonio Macaluso, Luca Clissa, Stefano Lodi, and Claudio Sartori. Quantum splines for non-linear approximations. In *Proceedings of the 17th ACM International Conference on Computing Frontiers*, CF '20, pages 249–252, New York, NY, USA, 2020. Association for Computing Machinery.
5. Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15), 10 2009.
6. Yudong Cao, Gian Giacomo Guerreschi, and Alán Aspuru-Guzik. Quantum neuron: an elementary building block for machine learning on quantum computers.

7. Marco Maronese, Claudio Destri, and Enrico Prati. Quantum activation functions for quantum neural networks, 2022.
8. Michael Lubasch, Jaewoo Joo, Pierre Moinier, Martin Kiffner, and Dieter Jaksch. Variational quantum algorithms for nonlinear problems. *Phys. Rev. A*, 101:010301, Jan 2020.
9. Carlos Bravo-Prieto, Ryan LaRose, M. Cerezo, Yigit Subasi, Lukasz Cincio, and Patrick J. Coles. Variational quantum linear solver, 2019.
10. Trevor Hastie, Jerome Friedman, and Robert Tibshirani. *The Elements of Statistical Learning*. Springer New York, 2001.
11. Carl d. Boor. *A Practical Guide to Splines*. Springer Verlag, New York, 1978.
12. Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Phys. Rev. Lett.*, 87:167902, Sep 2001.
13. Vanio Markov, Charlee Stefanski, Abhijit Rao, and Constantin Gonciulea. A generalized quantum inner product and applications to financial engineering. *arXiv preprint arXiv:2201.09845*, 2022.
14. M. Mottonen and J. J. Vartiainen. Decompositions of general quantum gates. *Ch. 7 in Trends in Quantum Computing Research (NOVA Publishers, New York), 2006*, 2005.
15. R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca. Quantum algorithms revisited. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):339–354, 1998.
16. John R. Rice. A theory of condition. *SIAM Journal on Numerical Analysis*, 3(2):287–310, 1966.
17. Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Quantum Computation and Quantum Information,*, 2000.
18. Antonio Macaluso, Luca Clissa, Stefano Lodi, and Claudio Sartori. A variational algorithm for quantum neural networks. In *Computational Science–ICCS 2020: 20th International Conference, Amsterdam, The Netherlands, June 3–5, 2020, Proceedings, Part VI 20*, pages 591–604. Springer, 2020.
19. Antonio Macaluso, Filippo Orazi, Matthias Klusch, Stefano Lodi, and Claudio Sartori. A variational algorithm for quantum single layer perceptron. In *Machine Learning, Optimization, and Data Science*, pages 341–356, Cham, 2023. Springer Nature Switzerland.
20. Antonio Macaluso, Matthias Klusch, Stefano Lodi, and Claudio Sartori. MAQA: a quantum framework for supervised learning. *Quantum Information Processing*, 22(3), mar 2023.
21. Adriano Barenco, André Berthiaume, David Deutsch, Artur Ekert, Richard Jozsa, and Chiara Macchiavello. Stabilization of quantum computations by symmetrization. *SIAM Journal on Computing*, 26(5):1541–1557, oct 1997.