Quantum-assisted Hilbert-space Gaussian process regression

Ahmad Farooq,[*] Cristian A. Galvis-Florez,[†] and Simo Särkkä
Department of Electrical Engineering and Automation, Aalto University, 02150 Espoo, Finland

(Received 2 February 2024; accepted 18 April 2024; published 7 May 2024)

Gaussian processes are probabilistic models that are commonly used as functional priors in machine learning. Due to their probabilistic nature, they can be used to capture prior information on the statistics of noise, smoothness of the functions, and training data uncertainty. However, their computational complexity quickly becomes intractable as the size of the data set grows. We propose a Hilbert-space approximation-based quantum algorithm for Gaussian process regression to overcome this limitation. Our method consists of a combination of classical basis function expansion with quantum computing techniques of quantum principal component analysis, conditional rotations, and Hadamard and SWAP tests. The quantum principal component analysis is used to estimate the eigenvalues, while the conditional rotations and the Hadamard and SWAP tests are employed to evaluate the posterior mean and variance of the Gaussian process. Our method provides polynomial computational complexity reduction over the classical method.

DOI: 10.1103/PhysRevA.109.052410

I. INTRODUCTION

Gaussian processes (GPs) are probabilistic machine learning methods widely used in applications such as robotics and control, signal processing, geophysics, climate modeling, financial markets, and data mining, as well as Bayesian optimization and probabilistic numerics [1–4]. GPs are nonparametric probabilistic models that can be used for modeling multidimensional nonlinear functions through their mean and covariance functions [5,6]. However, the traditional GP regression (GPR) methods struggle with computational efficiency, especially when handling large datasets [7]. This limitation becomes particularly pronounced in fields where rapid processing of large-scale data is critical. In this paper, to tackle this challenge, we aim to accelerate the GPR through quantum computing.

The main computational complexity of GPR arises from the computation of the mean and variance of the posterior distribution. This process becomes increasingly computationally heavy with larger datasets, with computational and memory requirements scaling as $O(N^3)$ and $O(N^2)$, respectively, for N observations of input data. To alleviate this problem, various methods have been proposed. In the inducing-point methods [8–10] the covariance matrix is approximated using a smaller number M of inducing points than the full training set, which reduces the computations to $O(NM^2)$ or $O(M^3)$ (for likelihood evaluation and prediction, respectively).

In this paper, we concentrate on low-rank methods [11–13] which are based on approximating the precision matrix via a set of M basis functions which also brings the computational complexity down to $O(NM^2)$ or $O(M^3)$. In particular, we use the method proposed by Solin and Särkkä [7] which uses the Hilbert space of eigenfunctions defined by a Laplace operator to approximate the covariance function, which offers a tunable balance between computational complexity and approximation accuracy [7,14].

In recent years, quantum computers have emerged as potential replacements for classical computers [15]. They offer exponential reductions in computational complexity for machine learning tasks. Quantum computing uses the principles of quantum mechanics to implement computational tasks and has been demonstrated for certain types of problems [16], for example, integer-number factoring [17], fast database search [18], and matrix inversion [19].

Many quantum algorithms have been proposed for accelerating machine learning tasks. Among the plethora of quantum algorithms, the Harrow-Hassidim-Lloyd (HHL) matrix-inversion algorithm [19] is often used to accelerate machine learning tasks. It serves as the foundation for various other algorithms such as quantum linear regression and quantum support vector machines [20–22]. However, the HHL algorithm has its challenges, for example, in quantum state preparation, unitary simulation, sparsity, and matrix conditioning [23].

An HHL-based algorithm for quantum-assisted Gaussian process regression was introduced in [24]. This algorithm assumes that quantum state preparation and unitary simulation can be performed efficiently. Importantly, this algorithm addresses the inherent limitations of the HHL approach by appropriately selecting the covariance function to construct s-sparse matrices and also carefully adjusts the noise parameters to ensure that the matrix remains well conditioned, as indicated by a condition number κ. To achieve a desired

[*] ahmad.farooq@aalto.fi
[†] cristian.galvis@aalto.fi

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

level of accuracy ϵ, it exhibits a run time that scales as $O(\log(N)\kappa^2 s^2/\epsilon)$.

The quantum principal component analysis (qPCA) is another quantum machine learning algorithm that draws inspiration from the HHL algorithm to estimate the dominant eigenvalues and eigenvectors [25]. The authors of [26] proposed a method to prepare the covariance matrix on a quantum computer using annihilation and creation operators and implemented the concept of qPCA to approximate the mean and variance of the GPR. This approach aims to achieve polynomial speedup compared to classical algorithms by overcoming the quantum-state-preparation and efficient-unitary-simulation assumptions.

Our methodology begins with the approximation of the kernel function using Hilbert-space basis functions. Subsequently, we provide encoding schemes for the dataset on a quantum computer. One of them aims to reduce the classical preprocessing of the algorithm by implementing the Hilbert-space functions through quantum unitary operations. We then apply qPCA to extract dominant eigenvectors and eigenvalues for a nonsparse low-rank matrix and insert them into a quantum register. To derive the posterior mean and variance for reduced-rank Gaussian process regression, we employ conditional controlled rotations followed by the Hadamard tests for the mean and the SWAP tests for the variance. We also include numerical examples to demonstrate and validate the effectiveness of our proposed method.

The contribution of this paper is to develop a fast Gaussian-process-regression algorithm by implementing the Hilbert-space approximation of the kernel presented in [7] on a quantum computer. This classical approach already shifts the prediction complexity of Gaussian processes from $O(N^3)$ to $O(M^3)$, reducing the dependence from the number of observations N to the number of eigenfunctions M used to approximate the kernel. Our implementation reduces the quantum Gaussian-process-regression complexity $O(\log(N)\kappa^2 s^2/\epsilon)$ of the method in [24], when data are given in quantum states, to $O(\log(M)\epsilon^{-3}\kappa^2)$. Furthermore, when we consider data uploading, the method has a complexity reduction to $O(\text{poly}[\log(NM)]\log(M)\epsilon^{-3}\kappa^2)$. Finding real-world applications in which quantum solutions show superior performance over classical methods is a challenging task. This paper demonstrates a polynomial speed advantage over existing classical algorithms for low-rank approximation in GP regression.

The structure of this paper is as follows. In Sec. II, we review the classical formulation for the Hilbert-space approximation of GPR. We provide the quantum-assisted Hilbert-space GPR algorithm in Sec. III. The complexity analysis of the proposed algorithm and a comparison with state-of-the-art methods are given in Sec. IV. Section V discusses the numerical implementation of our algorithm on a classical simulator. We then conclude our findings in Sec. VI.

II. HILBERT-SPACE APPROXIMATION OF GAUSSIAN PROCESS REGRESSION

In this section, we summarize the classical Hilbert-space method for reduced-rank GPR [7] and show how GPR can be rewritten in terms of eigenvalues and eigenvectors. We first briefly review the classical GPR. Then, we show how to approximate the kernel using a Hilbert space of functions defined by the eigenspace of the Laplace operator. Finally, we show how GPR can be expressed in terms of singular-value decomposition (SVD). This allows us to write these quantities in a suitable form so that they can be calculated using quantum states.

A. Gaussian process regression

Gaussian process regression [5] is a method for modeling and predicting multidimensional data. Consider a dataset $\mathcal{D} = (\mathbf{x}_i, y_i)_{i=1}^N$, where each $\{\mathbf{x}_i\}_{i=1}^N$ is a d-dimensional input vector and y_i is its corresponding measurement. In GPR, we aim to estimate an underlying function $f(\mathbf{x})$ by modeling it as a realization of a Gaussian process. The measurements are then Gaussian distributed with added Gaussian noise $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$:

$$f \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}')), \tag{1}$$

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \tag{2}$$

where $k(\mathbf{x}, \mathbf{x}')$ denotes the covariance function (kernel), which is a positive-semidefinite function $k : \Omega \times \Omega \to \mathbb{R}$. The choice of kernel function drives the quality of the estimation. A common kernel choice for GPR is the square exponential covariance function [5]:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2l^2}\|\mathbf{x} - \mathbf{x}'\|^2\right), \tag{3}$$

where σ_f and l are the signal-scale and length-scale hyperparameters, respectively.

The objective in GPR is to predict the mean and variance of the output for new inputs $\mathbf{x}_*$. These predictions are derived from the posterior distribution, which is also Gaussian:

$$p(f_* \mid \mathbf{x}_*) = \mathcal{N}(f_* \mid E[f_*], V[f_*]). \tag{4}$$

The mean and variance of the posterior distribution are given by [5]

$$E[f_*] = \mathbf{k}_*^\top \left(\mathbf{K} + \sigma^2 I\right)^{-1}\mathbf{y}, \tag{5}$$

$$V[f_*] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top \left(\mathbf{K} + \sigma^2 I\right)^{-1}\mathbf{k}_*. \tag{6}$$

Here, we denote by $\mathbf{y}$ the vector with components y_i from the dataset; $\mathbf{K}$ is the $N \times N$ matrix, with entries $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ consisting of covariance functions between all input points in the training set, and $\mathbf{k}_*$ is the covariance vector, with the ith entry being $k(\mathbf{x}_*, \mathbf{x}_i)$. The kernel function can be approximated by a set of basis functions in a suitable Hilbert space, as will be discussed next.

B. Kernel-function approximation

We can approximate a kernel function by considering the eigenvalue problem of the Laplace operator [7]:

$$\begin{aligned} -\nabla^2\phi_j(\mathbf{x}) &= \lambda_j\phi_j(\mathbf{x}), \quad \mathbf{x} \in \Omega, \\ \phi_j(\mathbf{x}) &= 0, \quad \mathbf{x} \in \partial\Omega, \end{aligned} \tag{7}$$

where the domain Ω behaves well enough that the eigenfunctions and eigenvalues exist. The functions $\phi_j(\cdot)$ are orthonormal with respect to the inner product,

$$\int_{\Omega} \phi_i(\mathbf{x})\phi_j(\mathbf{x})d\mathbf{x} = \delta_{ij}, \tag{8}$$

which also defines a Hilbert space.

All the eigenvalues λ_j of the Laplace operator are real and positive. If the kernel function is isotropic $k(\mathbf{x}, \mathbf{x}') = k(||\mathbf{x} - \mathbf{x}'||)$, then its eigenvalues are given by the scalar function $S(\omega)$, called the spectral density, which is the Fourier transform of $\mathbf{h} \mapsto k(||\mathbf{h}||)$. It turns out that we can approximate the kernel function in the domain Ω by [7]

$$k(\mathbf{x}, \mathbf{x}') \approx \sum_{j=1}^{M} S(\sqrt{\lambda_j})\phi_j(\mathbf{x})\phi_j(\mathbf{x}'). \tag{9}$$

Using this Hilbert-space approximation of the kernel function, we can reformulate Eqs. (5) and (6). This modification allows for computationally efficient approximations for the mean and covariance of the GP:

$$E[f_*] \approx \boldsymbol{\phi}_*^{\top}(\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi} + \sigma^2\boldsymbol{\Lambda}^{-1})^{-1}\boldsymbol{\Phi}^{\top}\mathbf{y}, \tag{10}$$

$$V[f_*] \approx \sigma^2\boldsymbol{\phi}_*^{\top}(\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi} + \sigma^2\boldsymbol{\Lambda}^{-1})^{-1}\boldsymbol{\phi}_*, \tag{11}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with components $\boldsymbol{\Lambda}_{jj} = S(\sqrt{\lambda_j})$, the matrix $\boldsymbol{\Phi}$ has components $\boldsymbol{\Phi}_{ij} = \phi_j(\mathbf{x}_i)$, and $\boldsymbol{\phi}_*$ has components $\phi_j(\mathbf{x}_*)$. We refer to this approximation as the Hilbert-space approximation for Gaussian process regression (HSGPR) [14]. The approximation of the kernel now depends on the domain Ω and the set of eigenfunctions chosen in this domain. For the implementation in this paper, we chose Ω in the domain $[-L, L]$. The Laplace operator in this domain gives rise to the set of sinusoidal eigenfunctions $\phi_j(x) = L^{-1/2}\sin[\pi j(x + L)/2L]$ with their corresponding eigenvalues $\lambda_j = (\pi j/2L)^2$. Additionally, we implement the approximation over the square exponential covariance function, whose spectral density is $S(\omega) = \sigma_f^2\sqrt{2\pi}\, l \exp(-\frac{l^2\omega^2}{2})$. This kernel approximation allows us to reduce the complexity of the matrix inversion needed to find the mean and variance of the GPR.

C. Mean and variance of reduced-rank GPR using singular-value decomposition

In this section, we will convert the mean and variance expressions of GPR into a form that enables them to be expressed as expected values of quantum states and calculated using a quantum computer. Before applying our quantum algorithm, we modify Eqs. (10) and (11). For the GPR, we need the eigenvalues and eigenvectors of $(\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi} + \sigma^2\boldsymbol{\Lambda}^{-1})$ which we wish to express in terms of $\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi}$. We need to reformulate in such a way that both quantities have the same set of eigenvectors, allowing us to write the mean and variance of the GPR in terms of this common set of eigenvectors. This will enable us to write these quantities in terms of the expected values of quantum states.

To address this, we define $\mathbf{X} = \boldsymbol{\Phi}\sqrt{\boldsymbol{\Lambda}} \in \mathbb{R}^{N\times M}$, where $\sqrt{\boldsymbol{\Lambda}}$ is a diagonal matrix with elements $\sqrt{\boldsymbol{\Lambda}_{ii}} = \sqrt{S(\sqrt{\lambda_i})}$, which gives

$$E[f_*] = \mathbf{X}_*^{\top}(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})^{-1}\mathbf{X}^{\top}\mathbf{y}, \tag{12}$$

$$V[f_*] = \sigma^2\mathbf{X}_*^{\top}(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})^{-1}\mathbf{X}_*, \tag{13}$$

where $\mathbf{X}_*^{\top} = \phi_*^{\top}\sqrt{\boldsymbol{\Lambda}}$. Now the eigenvectors of $(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})$ are the same as those of $\mathbf{X}^{\top}\mathbf{X}$.

We then begin by applying the SVD to the real data matrix $\mathbf{X}$, which is then expressed as $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\top}$. Here, $\boldsymbol{\Sigma} \in \mathbb{R}^{R\times R}$ is a diagonal matrix containing the real singular values $\lambda_1, \lambda_2, \dots, \lambda_R$, and the orthogonal matrices $\mathbf{U} \in \mathbb{R}^{N\times R}$ and $\mathbf{V} \in \mathbb{R}^{R\times M}$ correspond to the left and right singular vectors, respectively. Taking into account the sum of $\mathbf{X}^{\top}\mathbf{X}$ and $\sigma^2\mathrm{I}$, we derive $\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I} = \mathbf{V}\boldsymbol{\Sigma}'\mathbf{V}^{\top}$, where $\boldsymbol{\Sigma}'$ is a diagonal matrix with elements $\boldsymbol{\Sigma}'_{ii} = \lambda_i^2 + \sigma^2$. Then, the eigendecomposition of $(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})^{-1}\mathbf{X}^{\top}$ is given by

$$(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})^{-1}\mathbf{X}^{\top} = \mathbf{V}\boldsymbol{\Sigma}''\mathbf{U}^{\top}, \tag{14}$$

where $\boldsymbol{\Sigma}''$ has diagonal components $\boldsymbol{\Sigma}''_{ii} = \frac{\lambda_i}{\lambda_i^2 + \sigma^2}$. We can write Eq. (14) as

$$(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})^{-1}\mathbf{X}^{\top} = \sum_{r=1}^{R} \frac{\lambda_r}{\lambda_r^2 + \sigma^2}\mathbf{v}_r\mathbf{u}_r^{\top}. \tag{15}$$

Then, the mean of the GPR can be expressed using the SVD as

$$E[f_*] = \mathbf{X}_*^{\top}(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})^{-1}\mathbf{X}^{\top}\mathbf{y} = \sum_{r=1}^{R} \frac{\lambda_r}{\lambda_r^2 + \sigma^2}\mathbf{X}_*^{\top}\mathbf{v}_r\mathbf{u}_r^{\top}\mathbf{y}. \tag{16}$$

Similarly, we can write the variance of GPR using the SVD as

$$\begin{aligned} V[f_*] &= \sigma^2\mathbf{X}_*^{\top}(\mathbf{X}^{\top}\mathbf{X} + \sigma^2\mathrm{I})^{-1}\mathbf{X}_* \\ &= \sigma^2\sum_{r=1}^{R} \frac{1}{\lambda_r^2 + \sigma^2}\mathbf{X}_*^{\top}\mathbf{v}_r\mathbf{v}_r^{\top}\mathbf{X}_*. \end{aligned} \tag{17}$$

We have now expressed the mean and variance of GPR in a form that allows us to compute each of them as expected values of quantum states, which we will do in the next section.

III. QUANTUM-ASSISTED HILBERT-SPACE GPR ALGORITHM

In this section, we propose a low-rank method for quantum-assisted Gaussian process regression which we call quantum-assisted Hilbert-space Gaussian process regression (QA-HSGPR). For its implementation, we have to encode the data matrix $\mathbf{X}^{\top}\mathbf{X}$ into a quantum state. After that, we can implement a quantum algorithm that allows us to extract its eigenvalues. Then, we build the quantum circuits whose expected values correspond to the mean and variance that characterize the GPR.

A. Quantum state preparation from the dataset

Quantum computers encode classical information into quantum states using qubits [27]. A quantum state with n qubits can be expressed as a 2^n-dimensional vector $|\psi\rangle =$

$\sum_{i=0}^{2^n-1} a_i \left|i\right\rangle$, where $\{\left|i\right\rangle\}$ represents the computational basis $\{\left|0\cdots 0\right\rangle = \left|0\right\rangle, \dots, \left|1\cdots 1\right\rangle = \left|2^n-1\right\rangle\}$. The coefficients a_i are complex numbers that satisfy the normalization condition $\sum_{i=0}^{2^n-1} |a_i|^2 = 1$. We use the notation $\left\langle\psi\right|$ to represent the conjugate transpose of the quantum state $\left|\psi\right\rangle$.

We use an amplitude-state-encoding scheme to prepare the quantum state [28]. The amplitude quantum state encoding encodes the classical vector $(\alpha_1\,\alpha_2\,\cdots\,\alpha_n)^\top$ into the coefficients of the quantum state. We begin by obtaining a matrix $\mathbf{X} \in \mathbb{R}^{N\times M}$ using the eigenfunction of the Laplace operator in the given domain. We then vectorize the matrix $\mathbf{X}$ and encode it using the amplitude-state-encoding scheme [28]. We can encode the vectorized matrix $\mathbf{X}$ into a quantum state in two ways. First, the computation of the sinusoidal functions and spectral density is performed on a classical computer. We can compute these functions classically with $O(NM)$ computational complexity. A second method is provided in the Appendix. That method aims to reduce the classical preprocessing of the algorithm by implementing the basis functions through quantum unitary operations. The resulting state is

$$\left|\psi_{\mathbf{X}}\right\rangle = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x_n^m \left|m\right\rangle\left|n\right\rangle. \tag{18}$$

Here, x_n^m represents the value of the classical data at position n, m of the data matrix $\mathbf{X}$. It is important to note that the entries x_n^m must satisfy the condition $\sum_{n,m}|x_n^m|^2 = 1$. This ensures that the quantum state is properly normalized.

Encoding data into a specific quantum state $\left|\psi_{\mathbf{X}}\right\rangle$ as given in Eq. (18) generally involves a computational complexity of $O(NM)$ in the conventional quantum-state-preparation methodologies [28,29]. This complexity measure refers to the total number of quantum gates necessary to achieve the intended outcome. An approximate quantum-amplitude-encoding procedure was recently proposed for more efficient state preparation [30]. In that scheme, quantum state preparation is achieved in $O(\text{poly}[\log(NM)])$ when dealing with a real data matrix. The low-rank matrix $\mathbf{X}$ for HSGPR consists of real-valued entries, which would allow us to prepare it efficiently.

B. Estimation of eigenvalues

In this section, we show how to extract the eigenvalues of the symmetric matrix $\mathbf{X}^\top\mathbf{X}$ and store them in an ancillary quantum register. This allows us to easily perform the conditional rotation operation, which is necessary to obtain the desired amplitude quantities in Eqs. (16) and (17). Using the Gram-Schmidt decomposition in Eq. (18), we can reexpress $\left|\psi_{\mathbf{X}}\right\rangle$ as [31]

$$\left|\psi_{\mathbf{X}}\right\rangle = \sum_{r=1}^{R}\lambda_r\left|v_r\right\rangle\left|u_r\right\rangle. \tag{19}$$

Let us consider the density matrix $\rho_{\mathbf{X}^\top\mathbf{X}} = \text{Tr}_n\left|\psi_{\mathbf{X}}\right\rangle\left\langle\psi_{\mathbf{X}}\right|$ by disregarding the $\left|n\right\rangle$ register, where Tr_n is the partial trace on n qubits, which can be written as

$$\rho_{\mathbf{X}^\top\mathbf{X}} = \text{Tr}_j\{\left|\psi_{\mathbf{X}}\right\rangle\left\langle\psi_{\mathbf{X}}\right|\} = \sum_{r=1}^{R}\lambda_r^2\left|v_r\right\rangle\left\langle v_r\right|. \tag{20}$$

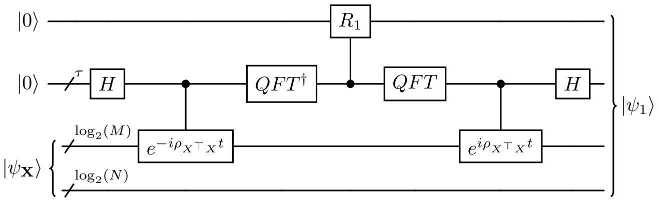

FIG. 1. Here, qPCA is first employed on the matrix $\rho_{\mathbf{X}^\top\mathbf{X}}$. Following this, a conditionally controlled unitary operation is executed based on the eigenvalue register. Finally, we revert the additional τ-qubit register to its original state by executing the corresponding inverse quantum operations to prepare the quantum state $\left|\psi_1\right\rangle$.

This method of construction eliminates the requirement for explicit computation of the term $\mathbf{X}^\top\mathbf{X}$ on classical systems. Next, we apply the unitary-evolution technique of qPCA [25] $\rho_{\mathbf{X}^\top\mathbf{X}}$ to register $\left|m\right\rangle$ of $\left|\psi_{\mathbf{X}}\right\rangle$, resulting in

$$\left|\xi_1\right\rangle = \sum_{z=0}^{Z}\left|z\Delta t\right\rangle\left\langle z\Delta t\right| \otimes e^{-iz\rho_{\mathbf{X}^\top\mathbf{X}}\Delta t}\left|\psi_{\mathbf{X}}\right\rangle\left\langle\psi_{\mathbf{X}}\right| e^{iz\rho_{\mathbf{X}^\top\mathbf{X}}\Delta t} \tag{21}$$

for some large Z, where the states $\left|\xi_i\right\rangle$ are intermediate states along the algorithm. By utilizing the quantum-phase-estimation algorithm, we can take the R dominant eigenvalues of the operator $\rho_{\mathbf{X}^\top\mathbf{X}}$ and write (cf. [31])

$$\left|\xi_2\right\rangle = \sum_{r=1}^{R}\lambda_r\left|v_r\right\rangle\left|u_r\right\rangle\left|\lambda_r^2\right\rangle, \tag{22}$$

in which the singular values λ_r are encoded in the τ qubits of an extra register.

C. Mean of Gaussian process regression

In this section, we provide the quantum method for computing the mean of GPR. We employ the conditional unitary on the ancilla qubit to invert the singular values. We add an extra ancilla qubit. The added ancilla qubit is conditionally rotated based on the eigenvalue register such that

$$\left|\xi_3\right\rangle = \sum_{r=1}^{R}\lambda_r\left|v_r\right\rangle\left|u_r\right\rangle\left|\lambda_r^2\right\rangle \times\left[\sqrt{1-\left(\frac{c_1}{\lambda_r^2+\sigma^2}\right)^2}\left|0\right\rangle + \frac{c_1}{\lambda_r^2+\sigma^2}\left|1\right\rangle\right],$$

where the parameter c_1 is chosen such that the quantity $\frac{c_1}{\lambda_r^2+\sigma^2}$ remains upper bounded by 1. After the conditional unitary, we reverse the computation in the τ-qubit register by performing inverse operations of qPCA to bring them back into $\left|0\right\rangle$ states,

$$\left|\psi_1\right\rangle = \sum_{r=1}^{R}\lambda_r\left|v_r\right\rangle\left|u_r\right\rangle\left|0\right\rangle \times\left[\sqrt{1-\left(\frac{c_1}{\lambda_r^2+\sigma^2}\right)^2}\left|0\right\rangle + \frac{c_1}{\lambda_r^2+\sigma^2}\left|1\right\rangle\right].$$

The quantum circuit for preparing the quantum state $\left|\psi_1\right\rangle$ is shown in Fig. 1.

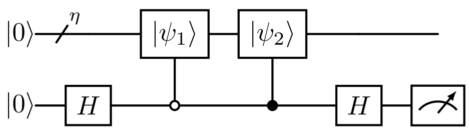

FIG. 2. Hadamard test circuit to estimate the mean of GPR. Here, $\eta = \log_2(NM) + \tau + 1$ qubits.

We then prepare another quantum state $|\psi_2\rangle = |\mathbf{X}_*\rangle\,|\mathbf{y}\rangle\,|0\rangle\,|1\rangle$, where $|\mathbf{X}_*\rangle = \sum_l x_*^l\,|l\rangle$ and $|\mathbf{y}\rangle = \sum_l y_l\,|l\rangle$ are normalized quantum states that encode the $\mathbf{X}_*$ and $\mathbf{y}$ vectors, respectively. We use the Hadamard test to estimate the inner product between these two states. The circuit diagram of the Hadamard test is shown in Fig. 2.

The implementation of the Hadamard test begins with the application of a Hadamard gate on the ancilla qubit. Depending on the state of the ancillary qubit, different quantum states are generated: $|\psi_1\rangle$ for state $|0\rangle$ and $|\psi_2\rangle$ for state $|1\rangle$. This results in the composite quantum state

$$|\psi_3\rangle = \frac{|0\rangle\,|\psi_1\rangle + |1\rangle\,|\psi_2\rangle}{\sqrt{2}}. \tag{23}$$

Applying the Hadamard gate on the ancilla qubit leads to

$$|\psi_3\rangle = \tfrac{1}{2}[|0\rangle \otimes (|\psi_1\rangle + |\psi_2\rangle) + |1\rangle \otimes (|\psi_1\rangle - |\psi_2\rangle)]. \tag{24}$$

Both $|\psi_1\rangle$ and $|\psi_2\rangle$ are real vectors, and their inner products $\langle\psi_1|\psi_2\rangle$ and $\langle\psi_2|\psi_1\rangle$ are equal. When measuring the ancilla qubit, the probability $p(0)$ of measuring the ancilla in state 0 is given by

$$p(0) = \tfrac{1}{2} + \tfrac{1}{2}\,\langle\psi_1|\psi_2\rangle, \tag{25}$$

where

$$\langle\psi_1|\psi_2\rangle = c_1 \sum_{r=1}^{R} \frac{\lambda_r}{\lambda_r^2 + \sigma^2}\,\langle\mathbf{X}_*|v_r\rangle\,\langle y|u_r\rangle. \tag{26}$$

Thus, we obtain an expression equal to the GPR mean as given in Eq. (16) up to a multiplicative constant. This mean value approximates the output function based on the data points and can be estimated using a quantum circuit.

D. Variance of Gaussian process regression

In this section, we build the quantum circuit which computes the variance of the Gaussian process regressor. For that purpose, conditional rotation is applied on the eigenvalue register such that

$$|\xi_4\rangle = \sum_{r=1}^{R} \lambda_r\,|v_r\rangle\,|u_r\rangle\,\big|\lambda_r^2\big\rangle \times \left[\sqrt{1 - \left(\frac{c_2}{\lambda_r\sqrt{\lambda_r^2+\sigma^2}}\right)^2}\,|0\rangle + \frac{c_2}{\lambda_r\sqrt{\lambda_r^2+\sigma^2}}\,|1\rangle\right],$$

where the parameter c_2 is chosen such that the quantity $\frac{c_2}{\lambda_r\sqrt{\lambda_r^2+\sigma^2}}$ remains upper bounded by 1. We proceed with the algorithm for measuring the ancilla qubit and consider only the measurements in state $|1\rangle$. Then, discarding the eigenvalue register, ancilla register, and right eigenvector register results in the final state

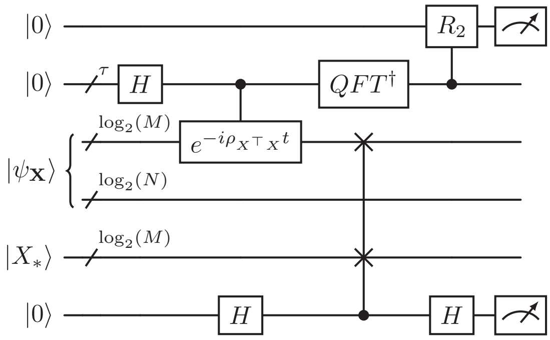

FIG. 3. Here, we illustrate the application of qPCA on the matrix $\rho_{\mathbf{X}^\top\mathbf{X}}$. Initially, qPCA identifies the eigenvalues and eigenvectors of the matrix. We then apply a conditionally controlled unitary on the ancilla register based on the eigenvalue register. Finally, the SWAP test is employed to estimate the variance of the GPR.

$$|\psi_1'\rangle = \frac{1}{\sqrt{p(1)}} \sum_{r=1}^{R} \frac{c_2}{\sqrt{\lambda_r^2+\sigma^2}}\,|v_r\rangle, \tag{27}$$

where the probability of acceptance is given by

$$p(1) = \sum_r \left|\frac{c_2}{\sqrt{\lambda_r^2+\sigma^2}}\right|^2. \tag{28}$$

We use the SWAP test to obtain the variance of GPR. We prepare another quantum state $|\psi_2'\rangle = |X_*\rangle$. Using the SWAP operations multiple times between $\langle\psi_1'\rangle$ and $\langle\psi_1'\rangle$, we can calculate $|\langle\psi_1'|\psi_2'\rangle|^2$, which corresponds to the posterior variance

$$|\langle\psi_1'|\psi_2'\rangle|^2 = \frac{c_2^2}{p(1)} \sum_{r=1}^{R} \frac{1}{\lambda_r^2+\sigma^2}|\langle\mathbf{X}_*|v_r\rangle|^2. \tag{29}$$

This is the same expression as we derived in Eq. (17), up to a multiplicative constant. We then multiply by the noise variance σ^2 to obtain the variance of the Gaussian process regressor. Figure 3 shows the circuit implementation for computing the variance.

IV. COMPLEXITY ANALYSIS

In this section, we analyze the computational complexity associated with our proposed method. The algorithm starts with the quantum-state-preparation step. We employ an approximate quantum encoding scheme to prepare the quantum state $|\psi_{\mathbf{X}}\rangle \in \mathbb{R}^{N\times M}$. This process requires a computational complexity of $O(\text{poly}[\log(NM)])$. Similarly, the preparation of the quantum state $|\psi_2\rangle$ mirrors this complexity. The total complexity for the preparation of the quantum state is $O(\text{poly}[\log(NM)])$.

Following the state-preparation step, we implement qPCA. The computational complexity for qPCA is $O(\log(M)\epsilon^{-3})$, where ϵ denotes the desired error tolerance. The next phase involves a conditional unitary operation, achievable in $O(\log(\frac{1}{\epsilon}))$. However, its complexity is relatively negligible compared with the complexity of qPCA. For both the mean

TABLE I. The time complexity of each step in the proposed method.

Step	Time complexity
Quantum state encoding	$O(\text{poly}[\log(NM)])$
qPCA	$O(\log(M)\epsilon^{-3})$
Ancilla rotation	$O(\log(\frac{1}{\epsilon}))$
Ancilla measurement	$O(\kappa^2)$

TABLE II. The time complexity of our proposed algorithm against existing quantum and classical counterparts.

	Computational complexity	
Algorithm	With data loading	Without data loading
QA-HSGPR	$O(\text{poly}(\log NM)\log M\epsilon^{-3}\kappa^2)$	$O(\log M\epsilon^{-3}\kappa^2)$
Zhao *et al.* [24]		$O(\log N\kappa^2 s^2/\epsilon)$
Chen *et al.* [26]	$O(\frac{1}{\sqrt{P_k}}dN\log\frac{d}{\delta}\log N\epsilon^{-3}\kappa)$	
HSGPR [7]	$O(NM^2)$	

and variance calculations in GPR, the initial algorithmic steps remain the same.

To calculate the mean of the GPR, we employ the Hadamard test. The computational complexity of this test is linear in the number of qubits, with measurement accounting only for a constant factor which can be ignored. In the variance computation of the QA-HSGPR algorithm, the method involves measurement after the unitary conditional rotation. This requires $O(\kappa^4)$ operations on average to measure the ancilla in the excited state. However, applying the techniques of [19,20], we can reduce this to $O(\kappa^2)$. Following this, the SWAP test, which is linear in the number of qubits, is applied. The measurement accounts for only a constant factor which can be ignored. Therefore, the overall computational complexity of the GPR is $O(\text{poly}[\log(NM)]\log(M)\epsilon^{-3}\kappa^2)$. A detailed comparison of the computational complexity of each step is summarized in Table I.

Classical Hilbert-space methods for GPR generally endure a computational load of $O(M^3)$ [7]. In contrast, the mean and variance computations of our algorithm demonstrate a polynomially faster speed. We also compare our model with that of Zhao *et al.* [24], whose algorithm complexity is $O(\log(N)\kappa^2 s^2/\epsilon)$, depending on the number of observations N, assuming that the data matrix is already prepared in the quantum state. If we consider such an assumption, our method would have a complexity $O(\log(M)\epsilon^{-3}\kappa^2)$, which is primarily dependent on the number of eigenfunctions M. This shifts the focus in complexity to M rather than N in our model, which significantly reduces the computational complexity, especially in scenarios with large datasets where, usually, $M \ll N$.

Furthermore, our method demonstrates significant improvements over the recently proposed quantum algorithm for Gaussian process regression. This contemporary model reports a time complexity of $O(\kappa[\frac{1}{\sqrt{P_k}}dN\log(\frac{d}{\delta})\log(N)\epsilon^{-3} + \text{poly}[\log(N)]])$, where P_k denotes the probability of success for creating the quantum state and δ indicates the precision of the preparation of the state [26]. This complexity depends on the dimension of the data points, which is not the case for our method. We present a detailed comparison of our method with existing approaches in Table II. This comparison reveals that the overall complexity of our proposed scheme is substantially lower than that of other existing methods.

V. NUMERICAL SIMULATIONS

In this section, we present the numerical results of our proposed scheme. Our focus is on demonstrating the effectiveness of the method by performing simulations on a classical computer. We use the built-in QISKIT function for quantum state encoding in our simulation [32].

A. Quantum circuit simulation

Several factors influence the performance of our method. A key aspect is the time parameter in the qPCA algorithm, which we use to estimate the eigenvalues λ_r^2 in the quantum register, as shown in Eq. (22). This estimation is done using the unitary operator $U = e^{-i\rho_{\mathbf{X}^\top\mathbf{X}}t}$, where we define the time parameter as $t = 2\pi/\delta_R$. Following the phase-estimation bounds detailed in [33], we can assert that $\delta_R > \lambda_{\max}^2$, where $\lambda_{\max}^2$ represents the largest eigenvalue of operator $\rho_{\mathbf{X}^\top\mathbf{X}}$. But for a good approximation of eigenvalues, δ_R should be slightly greater than $\lambda_{\max}^2$. The qPCA estimation algorithm could also give two different approximations to the same eigenvalue. To avoid this problem, we checked the similarity between the different excited states after the qPCA algorithm and discarded the states that likely represent the same eigenvalue in our simulation. The distinguishing of the eigenvalues improves when we increase the τ-qubit register.

We then select the dominant R eigenvalues. The selection of the dominant eigenvalues R is a critical factor here. In our demonstration, the selection is made such that the lowest of the Rth eigenvalues exceeds 0.01. Specifically, the probability p of finding the desired state, as outlined in [31], is bounded by

$$p \leqslant R\left|\frac{\lambda_{\min}}{\lambda_{\max}}\right|^2.$$

It is important to note that a significant decrease in the smallest eigenvalue will proportionally decrease the probability of measuring the desired state, requiring a higher number of shots for an accurate estimation. We define the constant c_1 as $\lambda_r^2 + \sigma^2$ and constant c_2 as $\lambda_r\sqrt{\lambda_r^2 + \sigma^2}$ in our simulations.

To precisely mirror the classical results using a quantum computer, a substantial number of qubits and a high number of shots are required. Furthermore, optimizing the hyperparameters is crucial for effective implementation of the algorithm. Our algorithm is ideally suited for fault-tolerant quantum computers.

B. Simulation results

To demonstrate the functionality of our method, we successfully implemented it on a much smaller scale. Our simulation involved $N = 16$ data samples derived from an oscillating function in a symmetric length interval $L = 2\pi$,

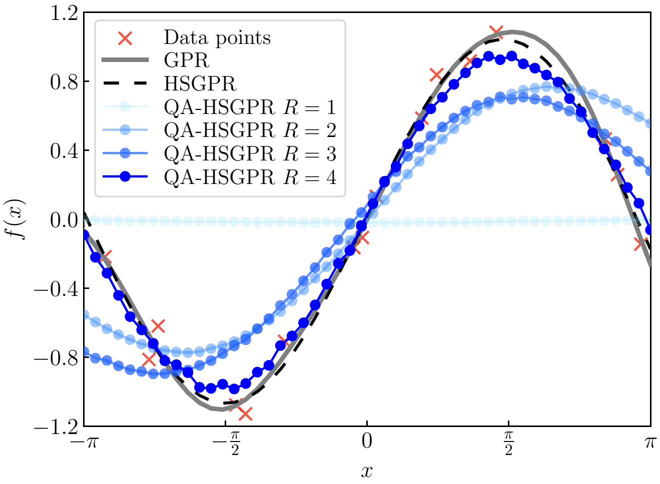

FIG. 4. Mean of the GPR using the squared exponential kernel (gray), the Hilbert-space approximation of the kernel with $M = 4$ eigenfunctions (black dashed line), and our reduced-rank approximation using a quantum circuit (blue lines) with $N = 16$ data points (red cross). The blue lines range over $R = 1, 2, 3, 4$, showing how taking a larger rank increases the accuracy of the estimation.

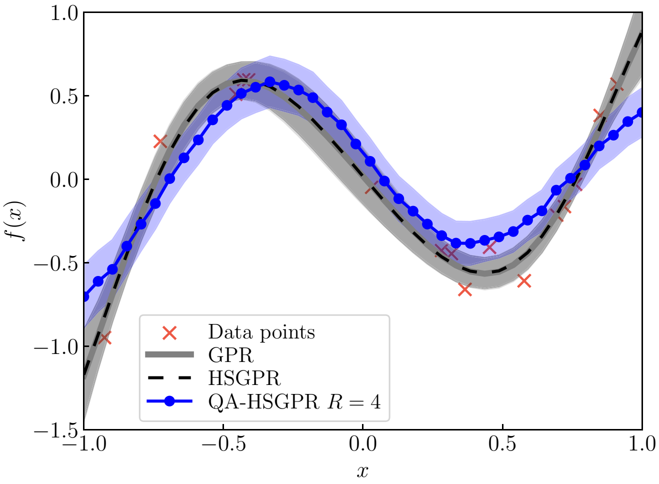

FIG. 5. Mean and variance of GPR using the squared exponential kernel (gray solid line), the Hilbert-space approximation of the kernel with $M = 8$ eigenfunctions (black dashed line), and our QA-HSGPR (blue line) with $N = 16$ data points (red cross). Each point on the blue line represents a simulation. The shaded areas around each approximation line indicate the 95% confidence intervals, providing a visual representation of the uncertainty associated with each method. We can see that our proposed scheme approximates the HSGPR method well with $R = 4$.

with additive white Gaussian noise $\sigma = 0.1$, length scale $l = 1$, and signal variance $\sigma_f = 1.5$. We use 10^6 shots for this demonstration and $\tau = 13$ qubits for the eigenvalue register. We implement the approximation using a set of sinusoidal eigenfunctions in the domain $\Omega = [-L, L]$ to approximate the kernel. First, we chose $M = 4$ and performed estimations for $R = 1, 2, 3, 4$. We show the behavior of the mean for different R.

Figure 4 compares the GPR using the exponential kernel, its Hilbert-space approximation, and the reduced-rank approximation implementing a quantum circuit proposed in this paper. We can see how with $R = 4$ the estimation already follows the tendency of the data. However, the estimation result is not exact. There are several reasons for this; first, we are using a limited number of qubits in the precision of the eigenvalues, which reduces the precision of the mean estimation. Moreover, along the circuit, we have to implement controlled gates of the unitary operator $e^{i\rho_{\mathbf{x}^\top \mathbf{x}} t}$ multiple times, as well as controlled rotations of small angles, which lead to numerical errors in the simulations.

We also performed another simulation with $M = 8$ and the same number of data points with a different function. We performed an estimation with $R = 4$, as illustrated in Fig. 5. The additive white Gaussian noise $\sigma = 0.1$, length interval $L = 2$, length scale $l = 1$, and signal variance $\sigma_f = 0.5$. We used $\tau = 16$ qubits for the eigenvalue register and 10^6 shots. As can be observed in Fig. 5, the estimation of the mean and variance of GPR through a quantum computer gives a close approximation of HSGPR. These simulations demonstrate the effectiveness of our algorithm and how it could be implemented when fault-tolerant quantum computers are available.

VI. CONCLUSION

In this paper, we introduced a quantum-assisted GPR algorithm leveraging a low-rank representation of the GP. Our algorithm addresses the high computational demand of GPR, showcasing how quantum computing can significantly enhance the scalability and efficiency of GPR models. A novel element of our contribution is the incorporation of the Hilbert-space basis-function approximation [7] into the quantum computing paradigm. This integration leads to significant improvements in computational efficiency, particularly in terms of reducing the computational complexity compared to classical algorithms. We also provided numerical examples within a quantum setting, which showed that the method also works in practice.

As for future work, probabilistic numerics techniques [3] provide a means to obtain probabilistic approximations for numerical integrals. A Bayesian quadrature treats the integral as a Gaussian process [34,35]. Because it is based on Gaussian process regression, Bayesian quadrature is faced with a significant computational challenge when evaluating the integral. The present methodology provides a promising method to evaluate large-scale integrals using Bayesian quadrature on a quantum computer.

The algorithm proposed here is suitable for fault-tolerant quantum computers, which makes its implementation in noisy intermediate-scale quantum devices a challenge for further work. The complexity of the circuit is mainly dominated by the qPCA and quantum-phase-estimation algorithm; then, alternative versions of these algorithms can be considered to reduce the complexity of the circuit. For the qPCA algorithm, a hybrid classical-quantum approach that implements a variational circuit can be considered to reduce the depth of the circuit [36,37]. The previous proposal would reduce the depth of the circuit but increase the classical resources needed to execute the algorithm. On the other hand, it has been shown that iterative approaches of the quantum-phase-estimation algorithm reduce the complexity of the circuit needed for this task [38,39]. The application of iterative versions of the quantum-phase-estimation algorithm would reduce the complexity of the circuit needed to implement our method, enabling the possibility of executing it on quantum hardware.

The open-source implementation of our simulation is available from GitHub [40].

ACKNOWLEDGMENTS

We want to gratefully acknowledge funding from the Research Council of Finland, Project No. 350221.

A.F. and C.A.G.-F. contributed equally to this work.

APPENDIX: QUANTUM-STATE-PREPARATION SCHEME TO ENCODE THE HILBERT-SPACE BASIS FUNCTIONS

The vectorization of the matrix $\mathbf{X} = \mathbf{\Phi}\sqrt{\mathbf{\Lambda}}$ can be represented as

$$\mathbf{X} = \begin{pmatrix} \phi_1(x_1)\sqrt{\Lambda_1} \\ \vdots \\ \phi_1(x_N)\sqrt{\Lambda_1} \\ \phi_2(x_1)\sqrt{\Lambda_2} \\ \vdots \\ \phi_M(x_N)\sqrt{\Lambda_M} \end{pmatrix}. \tag{A1}$$

We can directly encode the vectorized matrix $\mathbf{X}$ on a quantum computer using only the data inputs x_j instead of computing the sinusoidal functions and spectral density on a classical computer. This approach reduces the dependence on classical preprocessing using Hadamard gates and conditional multicontrolled R_y gates.

In our quantum computing framework, we utilize two distinct registers: the first one consists of a two-qubit ancilla, and the second register is dedicated to encoding the data matrix. The initial state of the quantum system is represented as

$$|X\rangle_1 = |00\rangle_1 |0\cdots 0\rangle_2. \tag{A2}$$

Applying the Hadamard gate on all qubits of the data-matrix register results in

$$|X\rangle_2 = \frac{1}{2^{NM/2}} \sum_{i=0}^{NM-1} |00\rangle_1 |i\rangle_2. \tag{A3}$$

For simplicity, we can decompose the single-index state $\sum_{i=0}^{NM-1} |i\rangle$ into a tensor product of two states, $\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} |m\rangle |n\rangle$. We then apply the control rotation on the first ancilla qubits by all possible states of the data qubits using sequences of the so-called multicontrolled $R_y(\theta_{mn})$ rotations, such as [29]

$$|X\rangle_3 = \frac{1}{2^{NM/2}} \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} |0\rangle_1 [\cos(\theta_{mn})\,|0\rangle_1 + \sin(\theta_{mn})\,|1\rangle_1]\,|m\rangle_2\,|n\rangle_2, \tag{A4}$$

where $\theta_{mn} = (m+1)\pi(x_{n+1} + L)/2L$. Next, we apply an operation similar to the previous one on the second qubit of the ancilla register with multicontrolled rotation $R_y(\varphi_m)$, such as

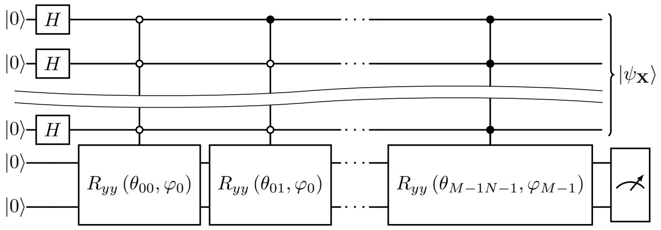
$|0\rangle$
H
$|0\rangle$
H
$|0\rangle$
H
$|0\rangle$
$|0\rangle$
$R_{yy}(\theta_{00}, \varphi_0)$
$R_{yy}(\theta_{01}, \varphi_0)$
$R_{yy}(\theta_{M-1N-1}, \varphi_{M-1})$
$|\psi_{\mathbf{X}}\rangle$

FIG. 6. Here, we show quantum circuit for efficiently preparing the normalized quantum state $|\psi_{\mathbf{X}}\rangle$ as given in Eq. (18). The procedure begins by applying Hadamard gates to each qubit in the data register, excluding those in the ancilla register, to create a superposition of states. Subsequently, a series of controlled R_{yy} rotations target the first and second qubits of the ancilla register, with the control conditioned on every possible state combination of the data qubits. The ancilla qubits are then measured with successful preparation of the desired quantum state when the measurement outcome is 11.

$$\begin{aligned} |X\rangle_4 = \frac{1}{2^{NM/2}} \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} &[\cos(\theta_{mn})\cos(\varphi_m)\,|00\rangle_1 \\ &+ \sin(\theta_{mn})\cos(\varphi_m)\,|01\rangle_1 + \sin(\theta_{mn})\cos(\varphi_m)\,|10\rangle_1 \\ &+ \sin(\theta_{mn})\sin(\varphi_m)\,|11\rangle_1]\,|m\rangle_2\,|n\rangle_2\,, \end{aligned} \tag{A5}$$

where $\varphi_m = \sin^{-1}[2^{NM/2}L^{-1/2}\sqrt{\Lambda_{m+1}}]$. The angles φ_m can be computed just once since they are independent of the data, meaning that they can be precomputed and therefore do not affect the complexity of the algorithm.

This controlled operation can be written as a controlled operation whose targets are both ancilla registers with rotation $R_{yy}(\theta_{mn}, \varphi_m) = R_y(\theta_{mn}) \otimes R_y(\varphi_m)$. After that we measure the ancilla qubits and continue with the quantum-state-preparation procedure only if the measurement outcome corresponds to 11, which results in the normalized quantum state of the vectorized data matrix $\mathbf{X}$ being

$$|X\rangle_5 = \frac{1}{2^{NM/2}\sqrt{p(11)}} \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \sin(\theta_{mn})\sin(\varphi_m)\,|m\rangle_2\,|n\rangle_2, \tag{A6}$$

where

$$p(11) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} |L^{-1/2}\sqrt{\Lambda_{m+1}}\phi_{m+1}(x_{n+1})|^2. \tag{A7}$$

Following successful quantum state preparation, the ancilla qubits are no longer required and are thus discarded. Figure 6 shows the quantum state preparation of the data matrix $\mathbf{X}$. If we denote the elements of the state vector as x_n^m, then the normalized equation derived through the quantum-state-preparation technique results in Eq. (18). We can also easily extend the quantum-state-preparation circuit to handle d-dimensional GPR data at the expense of additional ancilla qubits.

The preparation of the quantum state using multicontrolled rotations involves NM gates. From [29], we know that we can decompose the multicontrolled rotations into equivalent NM basic quantum gates. The computational cost of the final measurement compared to the sequence of multiple gates is almost negligible in this case because it involves only measuring in ancilla qubits. This results in the preparation of a quantum state using multicontrolled rotations with a computational complexity of $O(NM)$, similar to the standard quantum-state-preparation techniques.

[1] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, IEEE Trans. Pattern Anal. Mach. Intell. **37**, 408 (2013).

[2] S. Särkkä, A. Solin, and J. Hartikainen, IEEE Signal Process. Mag. **30**, 51 (2013).

[3] P. Hennig, M. A. Osborne, and H. P. Kersting, *Probabilistic Numerics: Computation as Machine Learning* (Cambridge University Press, Cambridge, 2022).

[4] R. Dürichen, M. A. F. Pimentel, L. Clifton, A. Schweikard, and D. A. Clifton, IEEE Trans. Biomed. Eng. **62**, 314 (2015).

[5] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning* (The MIT Press, Cambridge, MA, 2005).

[6] C. E. Rasmussen, in *Advanced Lectures on Machine Learning: ML Summer Schools 2003*, edited by O. Bousquet, U. von Luxburg, and G. Rätsch (Springer, Berlin, Heidelberg, 2003), pp. 63–71.

[7] A. Solin and S. Särkkä, Stat. Comput. **30**, 419 (2020).

[8] E. Snelson and Z. Ghahramani, in *Advances in Neural Information Processing Systems 18: Proceedings of the 2005 Conference*, edited by Y. Weiss, B. Schölkopf, and J. Platt (MIT Press, Cambridge, MA, 2005).

[9] S. Rossi, M. Heinonen, E. V. Bonilla, Z. Shen, and M. Filippone, Proc. Mach. Learn. Res. **130**, 1837 (2021).

[10] A. Nakai-Kasai and T. Tanaka, Mach. Learn. **111**, 1671 (2022).

[11] M. Lázaro-Gredilla, J. Quinonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal, J. Mach. Learn. Res. **11**, 1865 (2010).

[12] J. Quinonero-Candela and C. E. Rasmussen, J. Mach. Learn. Res. **6**, 1939 (2005).

[13] A. Rahimi and B. Recht, in *Advances in Neural Information Processing Systems 20: Proceedings of the 21st Annual Conference on Neural Information Processing*, edited by J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis (Curran Associates Inc., New York, 2007), Vol. 2, pp. 1160–1167.

[14] G. Riutort-Mayol, P.-C. Bürkner, M. R. Andersen, A. Solin, and A. Vehtari, Stat. Comput. **33**, 17 (2023).

[15] E. Pelucchi, G. Fagas, I. Aharonovich, D. Englund, E. Figueroa, Q. Gong, H. Hannes, J. Liu, C.-Y. Lu, N. Matsuda, J.-W. Pan, F. Schreck, F. Sciarrino, C. Silberhorn, J. Wang, and K. D. Jöns, Nat. Rev. Phys. **4**, 194 (2022).

[16] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Nature (London) **549**, 195 (2017).

[17] P. W. Shor, in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science, 20–22 November 1994, Santa Fe, NM, USA* (IEEE, Piscataway, NJ, 1994), pp. 124–134.

[18] L. K. Grover, in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing, STOC '96* (Association for Computing Machinery, New York, 1996), pp. 212–219.

[19] A. W. Harrow, A. Hassidim, and S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009).

[20] P. Rebentrost, M. Mohseni, and S. Lloyd, Phys. Rev. Lett. **113**, 130503 (2014).

[21] G. Wang, Phys. Rev. A **96**, 012335 (2017).

[22] C.-H. Yu, F. Gao, and Q.-Y. Wen, IEEE Trans. Knowl. Data Eng. **33**, 858 (2021).

[23] S. Aaronson, Nat. Phys. **11**, 291 (2015).

[24] Z. Zhao, J. K. Fitzsimons, and J. F. Fitzsimons, Phys. Rev. A **99**, 052331 (2019).

[25] S. Lloyd, M. Mohseni, and P. Rebentrost, Nat. Phys. **10**, 631 (2014).

[26] M.-H. Chen, C.-H. Yu, J.-L. Gao, K. Yu, S. Lin, G.-D. Guo, and J. Li, Phys. Rev. A **106**, 012406 (2022).

[27] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 10th anniversary ed. (Cambridge University Press, New York, 2011).

[28] M. Schuld and F. Petruccione, *Machine Learning with Quantum Computers* (Springer, Cham, 2021).

[29] M. Möttönen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, Phys. Rev. Lett. **93**, 130502 (2004).

[30] K. Nakaji, S. Uno, Y. Suzuki, R. Raymond, T. Onodera, T. Tanaka, H. Tezuka, N. Mitsuda, and N. Yamamoto, Phys. Rev. Res. **4**, 023136 (2022).

[31] M. Schuld, I. Sinayskiy, and F. Petruccione, Phys. Rev. A **94**, 022342 (2016).

[32] Qiskit Contributors, QISKIT: An open-source framework for quantum computing, 2023.

[33] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, Proc. R. Soc. London, Ser. A **454**, 339 (1998).

[34] A. O'Hagan, J. Stat. Plann. Inference **29**, 245 (1991).

[35] T. P. Minka, Statistics Department, Carnegie Mellon University, Technical Report, 2000 (unpublished).

[36] T. Xin, L. Che, C. Xi, A. Singh, X. Nie, J. Li, Y. Dong, and D. Lu, Phys. Rev. Lett. **126**, 110502 (2021).

[37] J. Preskill, Quantum **2**, 79 (2018).

[38] J. G. Smith, C. H. W. Barnes, and D. R. M. Arvidsson-Shukur, Phys. Rev. A **106**, 062615 (2022).

[39] P. M. Q. Cruz, G. Catarina, R. Gautier, and J. Fernández-Rossier, Quantum Sci. Technol. **5**, 044005 (2020).

[40] https://github.com/EEA-sensors/qa-hsgpr-codes.