Quantum algorithm for Gaussian process regression

Meng-Han Chen,[1] Chao-Hua Yu,[2] Jian-Liang Gao,[3] Kai Yu,[1] Song Lin,[1,*] Gong-De Guo,[1] and Jing Li[4]

[1]*College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350117, China*

[2]*School of Information Management, Jiangxi University of Finance and Economics, Nanchang 330032, China*

[3]*Research Computing & Data Science, Graduate School & Business School, Imperial College London, United Kingdom*

[4]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*

(Received 21 January 2022; revised 11 April 2022; accepted 9 June 2022; published 5 July 2022)

The Gaussian process is a widely used model for regression problems in supervised machine learning. However, predicting new inputs via a Gaussian process model becomes computationally inefficient when training a large data set. This paper proposes a fast quantum algorithm for prediction based on the Gaussian process regression. The proposed quantum algorithm consists of two subalgorithms: the first one aims to efficiently prepare the squared exponential covariance matrices and covariance functions vector with annihilation and creation operators; the other is to obtain predictive mean values and covariance values for new inputs. Evidence is also shown that the proposed quantum Gaussian process regression algorithm can achieve quadratic speedup over the classical counterpart.

DOI: 10.1103/PhysRevA.106.012406

I. INTRODUCTION

Resulting from the intersection of a subfield of computer science and application of statistics, machine learning is adaptive and able to learn from experience. That has attracted more and more researchers from many fields. However, classic machine learning algorithms face incredible challenges in computational performance when addressing the skyrocketing amount of data with the rapid development of information technology, now and future. Quantum computing uses the fundamental principles of quantum mechanics (such as quantum superposition and quantum entanglement) to implement computing tasks and has been demonstrated to achieve significant computational speedup, solving certain problems [1–3]. For example, Shor's quantum algorithm for large-number factoring has an exponential acceleration over the classical algorithm [4], posing a serious threat to the security of widely used RSA-based cryptography systems. In recent years, quantum computing has been applied to the field of machine learning. A variety of efficient quantum machine learning algorithms have been proposed, such as quantum clustering analysis [5,6], quantum neural networks [7,8], quantum classification [9,10], quantum decision tree [11], quantum association rules mining [12], and so on. The research of quantum algorithms and the exploration of quantum mechanical properties further augment the development of artificial intelligence [13]. Therefore it is critical to develop efficient quantum algorithms for implementing tasks in machine learning and data mining [14].

One of the most positive directions in machine learning is the development of practical Bayesian approaches to solve really challenging problems [15]. Among them, the Gaussian process regression (GPR) is a representative instance, which was originally proposed by Williams and Rasmussen in 1995 [16]. The applications of GPR range from geophysics (also known as Kriging), time series analysis, image processing, and automatic control [17,18]. The objectives of predicting the output of new input data $\vec{x}_*$ are computing the mean and variance of the output, given a number of training input-output data points. However, prediction by GPR is quite computationally expensive, especially when the size of the training dataset is sufficiently large. Causally, the rapid development of quantum computing leads to the exploration of quantum algorithms to accelerate GPR. For example, Zhao *et al.* proposed a quantum-assisted algorithm for GPR [19] which included a framework for approximately estimating the quantities of $\langle\vec{y}|(K+\sigma_n I)^{-1}|\vec{k}_*\rangle$ and $\langle\vec{k}_*|(K+\sigma_n I)^{-1}|\vec{k}_*\rangle$, corresponding to the estimates of mean and variance of the output of $\vec{x}_*$, respectively, where $\vec{y}$ is the training output vector, K is the covariance functions between training input data points, σ_n is a constant noise variance, I is the identity matrix, and $\vec{k}_*$ denotes the covariance functions between $\vec{x}_*$ and each training input data point. The algorithm achieves at least polynomial speedup over the classical GPR. Nevertheless, the speedup is satisfied by the two assumptions that both creating $|\vec{k}_*\rangle$ and simulating $K+\sigma_n I$ can be done efficiently. The assumptions do not naturally hold, because both tasks entail computing covariance functions between input data points that are hard to do classically when the training dataset is large.

In this paper we further investigate how quantum computing can be utilized to accelerate GPR and present a fast quantum algorithm for GPR without the above-mentioned two assumptions required in Zhao's algorithm [19]. Instead of encoding a whole state vector (point) in the amplitudes of a quantum state, which is commonly adopted in most quantum machine learning algorithms, we encode each element of a

*Corresponding author: lins95@gmail.com

state vector (point). Based on this data encoding scheme, we show how to efficiently create the quantum state $|\vec{k}_*\rangle$, simulate $K + \sigma_n I$, and finally estimate the mean predictor and variance predictor of the output of $\vec{x}_*$. Our quantum algorithm is exponentially faster than the classical GPR algorithm if K is well conditioned.

The rest of this paper is organized as follows. The second section reviews the classical GPR. The third section introduces how classical data are encoded on a quantum computer. Based on this data encoding, the fourth section gives a quantum algorithm for GPR, including the creation of the quantum state $|\vec{k}_*\rangle$, estimation of the mean and variance predictors, runtime analysis, and error analysis. Discussion and conclusion are given in the last section.

II. A REVIEW OF CLASSICAL GAUSSIAN PROCESS REGRESSION

In GPR we are given a training dataset with M data points $(\vec{x}_i, y_i)_{i=1}^M$, where $\vec{x}_i = (x_{i1}, x_{i2}, ..., x_{iN})^T \in R^N$ is a column vector of independent input variables and y_i is the corresponding scalar of a single output variable. The objective of GPR is to train a linear function $y = f(\vec{x}) + \varepsilon$ in the limited training dataset D that can fit the relationship between $\vec{x}_i$ and y_i, where $f(\vec{x})$ is real value, y is desired value. and $\varepsilon \sim \mathrm{N}(0, \sigma_M^2)$ is identically independently distributed Gaussian noise. Once obtained, f can be used to predict the output y_* of a new input $\vec{x}_*$.

A GPR model is fully specified by a mean function $m(\vec{x})$ and a covariance function (also known as a kernel function) $k(\vec{x}, \vec{x}')$. In 2005, Rasmussen and Williams [20] pointed out that these two functions can be obtained in terms of weight-space view and function-space view, whose expressions are as follows:

$$m(\vec{x}) = E[f(\vec{x})], \tag{1}$$

$$k(\vec{x}, \vec{x}') = E[[f(\vec{x}) - m(\vec{x})][f(\vec{x}') - m(\vec{x}')]], \tag{2}$$

where E denotes expectation value. Thus GPR can be written as $f(\vec{x}) \sim \mathrm{GP}(m(\vec{x}), k(\vec{x}, \vec{x}'))$, with GP as a Gaussian process. Generally, the classical GPR specifies the covariance function as a squared exponential covariance function [20], which is denoted as

$$\mathrm{cov}[f(\vec{x}_p, \vec{x}_q)] = k(\vec{x}_p, \vec{x}_q) = \exp\left(-\tfrac{1}{2}|\vec{x}_p - \vec{x}_q|^2\right). \tag{3}$$

Therefore the central goal of a GPR model is to predict the mean value and the covariance value of this distribution, also known as mean predictor $\bar{f}_*$ and covariance predictor $V[f_*]$. The mean predictor and covariance predictor can be re-expressed as

$$\bar{f}_* = \vec{k}_*^T \left(K + \sigma_M^2 I\right)^{-1} \vec{y}, \tag{4}$$

$$V[f_*] = k(x_*, x_*) - \vec{k}_*^T \left(K + \sigma_M^2 I\right)^{-1} \vec{k}_*, \tag{5}$$

where $\vec{k}_*$ is a vector that represents the kernel function of a test point $\vec{x}_*$ and all training points $\vec{x}$'s; K denotes an $M \times M$ dimensional covariance matrix, holding results between M training data; and $k(\vec{x}_*, \vec{x}_*)$ is the covariance of test points $\vec{x}_*$ with itself, which is a constant. Considering Eq. (4) as a linear combination of M kernel functions, with each of them focusing on training points, Eq. (4) can be written as

$$\bar{f}_* = \sum_{i=1}^{M} \alpha_i k(\vec{x}_i, \vec{x}_*), \tag{6}$$

where $\vec{\alpha} = (\alpha_1, \ldots, \alpha_M)^T = (K + \sigma_M^2 I)^{-1} \vec{y}$.

The following two steps are the implementation of Gaussian process distribution in the classical calculation. First, $L := Cholesky(K + \sigma_M^2 I)$ are calculated, and $\overrightarrow{\alpha} = L^T \backslash (L \backslash \vec{y})$. Second, the mean predictor is computed by $\bar{f}_* = \vec{k}_*^T \vec{\alpha}$. Since the calculation of the Cholesky factor is numerically stable, the runtime is proportional to $O(M^3)$. The computation of $k(\vec{x}_*, \vec{x}_*)$ requires only constant time. For $V[f_*]$, let $\vec{v} := L \backslash \vec{k}_*$, and thus

$$V[f_*] := k(\vec{x}_*, \vec{x}_*) - \vec{v}^T \vec{v}, \tag{7}$$

which can be computed with a number of basic arithmetics. Therefore the total runtime is $O(M^3)$. Nevertheless, in the current era of big data, the tremendous number of input points results in huge time complexity. That is why a quantum GPR is needed.

III. DATA ENCODING

In quantum machine learning, encoding a large number of data in quantum states is not trivial. To encode data efficiently, amplitude encoding is often used, that is, the data points (vectors) are encoded in the amplitudes of quantum states. Two primary classes, coherent and incoherent encoding, are widely used for amplitude encoding. Inspired by a quantum radial basis network [10], our quantum algorithm for GPR uses a coherent one [21,22] to encode the training input data points and any test input data points. The coherent version plays a crucial role in quantum optics and mathematical physics. They are defined in the Fock states $\{|0\rangle, |1\rangle, ...\}$, which is a basis of the infinite-dimensional Hilbert space H. Let $a, a^\dagger$ be the annihilation operator and creation operator, respectively, of the harmonic oscillator. Then we have

$$a|n\rangle = \sqrt{n}|n-1\rangle, \quad a^\dagger|n\rangle = \sqrt{n+1}|n+1\rangle. \tag{8}$$

For any $n \geqslant 1$, it is easy to see that

$$|n\rangle = \frac{(a^\dagger)^n}{\sqrt{n!}}|0\rangle. \tag{9}$$

Let r be a real number and its coherent state defined by

$$|\varphi_r\rangle = e^{-r^2/2} \sum_{k=0}^{\infty} \frac{r^k}{\sqrt{k!}}|k\rangle, \tag{10}$$

a unit eigenvector of corresponding eigenvalue r, that is, $a|\varphi_r\rangle = r|\varphi_r\rangle$. From Eqs. (8) and (9), we also have $|\varphi_r\rangle = e^{-r^2/2} e^{ra^\dagger}|0\rangle = e^{r(a^\dagger - \frac{a}{2})}|0\rangle$, and thus $|\varphi_r\rangle$ is obtained by a unitary operator of dimension infinity.

As for the preparation of $|\varphi_r\rangle$ in a finite quantum circuit, we can consider its Taylor approximation:

$$|\tilde{\varphi}_r\rangle \propto e^{-r^2/2} \sum_{k=0}^{T-1} \frac{r^k}{\sqrt{k!}}|k\rangle. \tag{11}$$

Thus the upper bound on the error square is

$$||\varphi_r\rangle - |\tilde{\varphi}_r\rangle|^2 \leqslant \frac{r^{2T}}{T!}. \tag{12}$$

By the Stirling formula, we can get $T! \approx \sqrt{2\pi T}(\frac{T}{e})^T$. Now, keeping the error within δ, let us take Eq. (12) $\leqslant \delta^2$, that is, T only needs to satisfy $2\log\frac{1}{\delta} - \frac{1}{2}\log 2\pi \leqslant (T+\frac{1}{2})\log T - 2T\log r - T$. Thus it is easy to get the Taylor approximation of $|\varphi_r\rangle$.

From the previous analysis, $|\varphi_r\rangle$ can be obtained by a unitary operator applying to $|0\rangle$. An arbitrary one-qubit computation can be implemented as a sequence of at most three R_z and R_y gates. This is due to the ZYZ decomposition [23]: given any 2×2 unitary matrix U, there exist angles ϕ,α,β,γ satisfying the following equation:

$$U = e^{i\phi}R_z(\alpha)R_y(\beta)R_z(\gamma). \tag{13}$$

The nomenclature R_y and R_z is motivated by a picture of one-qubit states as points on the surface of a sphere of unit radius.

For any vector $\vec{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iN})^T$, we have a unitary operation $U_{\vec{x}_i}$ acting as

$$U_{\vec{x}_i} : |0\cdots 0\rangle \mapsto |\varphi_{\vec{x}_i}\rangle \equiv |\varphi_{x_{i1}}\rangle \otimes |\varphi_{x_{i2}}\rangle \otimes \cdots \otimes |\varphi_{x_{iN}}\rangle. \tag{14}$$

So we can have an even larger unitary operation U_X that implements every $U_{\vec{x}_i}$ for $i = 0, 1, \ldots, M-1$ in a controlled fashion,

$$U_X : |i\rangle|0\cdots 0\rangle \mapsto |i\rangle|\varphi_{\vec{x}_i}\rangle, \tag{15}$$

and create a superposition of the coherent states of all M training input data points via

$$U_X\left(\sum_{i=0}^{M-1}|i\rangle|0\cdots 0\rangle\right) = |\Psi\rangle \equiv \sum_{i=0}^{M-1}|i\rangle|\varphi_{\vec{x}_i}\rangle. \tag{16}$$

According to the theorem 9 of Ref. [24], disentangling a qubit of pointed that an arbitrary (n+1)-qubit state can be converted into a separable (i.e., unentangled) state by a circuit. So U_X is the block-diagonal sum $\oplus_c R_y(-\theta_c)R_z(-\varphi_c)$, and U_X can be implemented by a multiplexed R_z gate followed by a multiplexed R_y. Therefore there is an efficient quantum circuit to prepare the coherent state $|\varphi_{\vec{x}_i}\rangle$ up to precision δ in time $O(\log 1/N\delta)$ [24].

This encoding method has two advantages. First, we can directly estimate the covariance function, i.e., the kernel value $e^{-\frac{\|\vec{x}_i-\vec{x}_j\|^2}{2}}$ between any two input data points $\vec{x}_i$ and $\vec{x}_j$ by estimating the inner product between $|\varphi_{\vec{x}_i}\rangle$ and $|\varphi_{\vec{x}_j}\rangle$, which can be achieved by a swap test or its variant [25,26]. Second, a superposition $|\Psi\rangle$ of the coherent states of all training input data points can be created easily. Then it is not difficult to take the partial traces on the second register of $|\Psi\rangle\langle\Psi|$ to generate the density operator of the following covariance matrix:

$$\begin{aligned}\rho &= Tr_2|\Psi\rangle\langle\Psi| \\ &= \frac{1}{M}\sum_{i,j=0}^{M-1}\exp\left(-\frac{1}{2}|\vec{x}_i-\vec{x}_j|^2\right)|i\rangle\langle j|.\end{aligned} \tag{17}$$

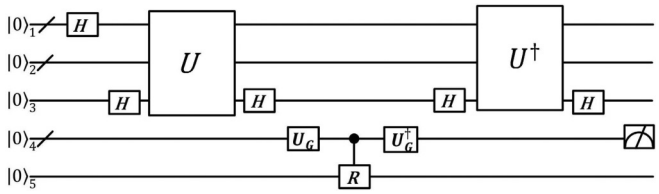

FIG. 1. The quantum circuit of the kernel vector. Here the H represents the Hadamard transform, U and $U^\dagger$ denote preparation for the coherent states of step 1.2 and the inverse process of step 1.7, respectively, and R is the controlled rotation of step 1.6.

IV. QUANTUM ALGORITHM FOR GPR

Keep in mind that the objective is to estimate the mean and variance of the output of a new input data point $\vec{x}_*$ according to Eqs. (4) and (5). Our quantum algorithm first creates a quantum state $|\vec{k}_*\rangle$ that encodes the normalized vector $\vec{k}_*$ on the amplitudes and estimates the norm of $\vec{k}_*$, and then it uses the state to approximately estimate the mean and the variance.

A. Creating the quantum state $|\vec{k}_*\rangle$

The covariance functions vector $\vec{k}_*$ can be mathematically written as

$$\begin{aligned}\vec{k}_* &= \sum_{i=0}^{M-1}\exp\left(-\frac{1}{2}|\vec{x}_*-\vec{x}_i|^2\right)|i\rangle \\ &= \sum_{i=0}^{M-1}\langle\varphi_{\vec{x}_i}|\varphi_{\vec{x}_*}\rangle|i\rangle \equiv \sum_{i=0}^{M-1}s_i|i\rangle.\end{aligned} \tag{18}$$

Shao [10] pointed out that the kernel function vector can be obtained by matrix-vector multiplications. But the time complexity at a scale of $\Omega(M)$ is very high, and its encoding unitary operator is hard to construct. Here we propose an alternative method to create $|\vec{k}_*\rangle$ and estimate the norm of $\vec{k}_*$, as shown in the schematic quantum circuit in Fig. 1. The following eight steps provide the process of our proposed method:

Step 1.0 Initializing the quantum state

$$|0\cdots 0\rangle_1|0\cdots 0\rangle_2|0\rangle_3|0\cdots 0\rangle_4$$

with a sufficiently large number of qubits, where the subscript numbers denote different registers. Then performing Hadamard operations on each qubit of register 1 to obtain the state

$$\sum_{i=0}^{M-1}|i\rangle_1|0\cdots 0\rangle_2|0\rangle_3|0\cdots 0\rangle_4, \tag{19}$$

and marking the indices of all the M training data points.

Step 1.1 Applying a Hadamard operation on register 3 leads to the state

$$\frac{1}{\sqrt{2M}}\sum_{i=0}^{M-1}|i\rangle_1|0\cdots 0\rangle_2(|0\rangle_3+|1\rangle_3)|0\cdots 0\rangle_4. \tag{20}$$

Step 1.2 Preparing the superposition of coherent states of training samples' on the second register [same as Eq. (13)] when the third register is $|0\rangle$ and preparing the coherent states

of test samples on the second register when the third register is $|1\rangle$, via the controlled U_X and $U_{\vec{x}_*}$ acting on the registers 1–3, i.e.,

$$U \equiv U_X \otimes |0\rangle\langle 0| + I \otimes U_{\vec{x}_*} \otimes |1\rangle\langle 1|, \tag{21}$$

which results in

$$\frac{1}{\sqrt{2M}} \sum_{i=0}^{M-1} |i\rangle_1 (|\varphi_{\vec{x}_i}\rangle_2 |0\rangle_3 + |\varphi_{\vec{x}_*}\rangle_2 |1\rangle_3) |0\cdots 0\rangle_4. \tag{22}$$

Step 1.3 Applying the Hadamard transform on the third register, we have

$$\begin{aligned} &\frac{1}{2\sqrt{M}} \sum_{i=0}^{M-1} |i\rangle_1 [(|\varphi_{\vec{x}_i}\rangle_2 + |\varphi_{\vec{x}_*}\rangle_2)|0\rangle_3 \\ &\quad + (|\varphi_{\vec{x}_i}\rangle_2 - |\varphi_{\vec{x}_*}\rangle_2)|1\rangle_3] |0\cdots 0\rangle_4, \\ &\equiv \frac{1}{\sqrt{M}} \sum_{i=0}^{M-1} |i\rangle_1 |\phi_i\rangle_{23} |0\cdots 0\rangle_4. \end{aligned} \tag{23}$$

Step 1.4 Using the quantum amplitude estimation technique [27,28] to estimate the squared amplitudes of $|0\rangle_3$ part of $|\phi_i\rangle$, we gain the following state:

$$\frac{1}{\sqrt{M}} \sum_{i=0}^{M-1} |i\rangle_1 |\phi_i\rangle_{23} |(1+s_i)/2\rangle_4. \tag{24}$$

It is worth noting that the unitary operation from steps 1.1–1.3, i.e., $H_3 U H_3$, acts as follows:

$$|i\rangle_1 |0\cdots 0\rangle_2 |0\rangle_3 \mapsto |i\rangle_1 |\phi_i\rangle_{23}. \tag{25}$$

Plus the operation $X_3 Z_3 X_3$ acting on the third register which flips the phase of $|0\rangle_3$ while keeping the phase of $|1\rangle_3$ unchanged in $|\phi_i\rangle$, we derive the unitary operation

$$U_G \equiv [H_3 U H_3 (I - |0\cdots 0\rangle_{23}\langle 0\cdots 0|_{23}) H_3 U^\dagger H_3] X_3 Z_3 X_3. \tag{26}$$

Thus, according to quantum amplitude estimation, performing a quantum phase estimation of U_G on state (23) gives rise to state (24).

Step 1.5 Subtracting 1/2 [29] in register 4 produces the following state:

$$\frac{1}{\sqrt{M}} \sum_{i=0}^{M-1} |i\rangle_1 |\phi_i\rangle_{23} |s_i/2\rangle_4. \tag{27}$$

Step 1.6 Appending one qubit and rotating it from $|0\rangle_5$ to $(\sqrt{1-\frac{s_i^2}{4}}|0\rangle_5 + \frac{s_i}{2}|1\rangle_5)$ controlled on $|\frac{s_i}{2}\rangle_4$, we obtain the state

$$\frac{1}{\sqrt{M}} \sum_{i=0}^{M-1} |i\rangle_1 |\phi_i\rangle_{23} |s_i/2\rangle_4 \left(\sqrt{1-\frac{s_i^2}{4}}|0\rangle_5 + \frac{s_i}{2}|1\rangle_5\right). \tag{28}$$

Step 1.7 Undoing step 1.4 and steps 1.1–1.3 makes the registers 2–4 return to their initial states, i.e., $|0\cdots 0\rangle_2 |0\rangle_3 |0\cdots 0\rangle_4$. Then, discarding registers 2–4, we attain the state

$$\frac{1}{\sqrt{M}} \sum_{i=0}^{M-1} |i\rangle_1 \left(\sqrt{1-\frac{s_i^2}{4}}|0\rangle_5 + \frac{s_i}{2}|1\rangle_5\right). \tag{29}$$

Step 1.8 Measuring the fifth register to see $|1\rangle_5$, once succeeded the final state of the first register approximates the desirable state as follows:

$$\frac{\sum_{p=0}^{M-1} s_i |i\rangle_1}{\sqrt{\sum_{p=0}^{M-1} s_i^2}} \equiv |\vec{k}_*\rangle. \tag{30}$$

The probability of success is $P_k \equiv \frac{\sum_{i=0}^{M-1} s_i^2}{4M}$, which means we need $O(1/P_k)$ repetitions to get $|\vec{k}_*\rangle$ with a large probability. It is notable that this can be improved by amplitude amplification with only $O(1/\sqrt{P_k})$ repetitions.

The aim of this section is to prepare kernel function vector $|\vec{k}_*\rangle$. The state $|\Psi\rangle$ is acquired by first preparing all the coherent states $|\varphi_{\vec{x}_i}\rangle$ and then generating their superposition by the Fourier transform. Therefore the time complexity is $O(\frac{1}{\sqrt{P_k}} MN \log 1/N\delta)$.

B. Estimating the mean predictor and variance predictor

There is an inverse process to observe the mean predictor and covariance predictor. Thus it is easy to think of using the HHL algorithm [30] as a subroutine to get the desired result. First, the covariance matrix K is a real symmetric matrix, namely, K is the Hermitian matrix, which can be written in spectral decomposition form [31,32],

$$K = \sum_{j=0}^{M-1} \lambda_j |\vec{u}_j\rangle\langle\vec{u}_j|, \tag{31}$$

where $\{\lambda_j\}_{j=0}^{M-1}$ are the eigenvalues of K, and $\{|\vec{u}_j\rangle\}_{j=0}^{M-1}$ are the corresponding eigenvectors. Without loss of generality, we assume $\lambda_j \in [\frac{1}{\kappa}, 1]$ (κ is the conditional number of matrix K). $\vec{k}_*/\|\vec{k}_*\|$ can be represented by the linear combination of $\{|\vec{u}_j\rangle\}_{j=0}^{M-1}$, that is, $\vec{k}_*/\|\vec{k}_*\| = \sum_{j=0}^{M-1} \alpha_j |\vec{u}_j\rangle_1$. Similarly, $\vec{y}/\|\vec{y}\|$ can also be denoted as $\{|\vec{u}_j\rangle\}_{j=0}^{M-1}$, namely, $\vec{y}/\|\vec{y}\| = \sum_{j=0}^{M-1} \beta_j |\vec{u}_j\rangle_1$. Therefore Eqs. (4) and (5) can be expressed in the following forms:

$$\bar{f}_* = \sum_{j=0}^{M-1} \frac{\alpha_j \beta_j}{\lambda_j + \sigma_M^2} \|\vec{k}_*\| \|\vec{y}\|, \tag{32}$$

$$V[f_*] = k(x_*, x_*) - \sum_{j=0}^{M-1} \frac{\alpha_j^2}{\lambda_j + \sigma_M^2} \|\vec{k}_*\|^2. \tag{33}$$

Next, the following five quantum steps, B1–B5, demonstrate the process of acquiring the values of Eqs. (26) and (27) as described by the following quantum steps. A corresponding schematic of computing is shown in Fig. 2.

Step B1. Preparing quantum state $|\vec{y}\rangle$, so that $\vec{y}$ is $|\vec{y}\rangle = \sum_{j=0}^{M-1} \beta_j |\vec{u}_j\rangle$, following the detailed procedures proposed by Yu *et al.* [33].

Step B2. Preparing the quantum kernel function initial state $|\vec{k}_*\rangle_1 = \sum_{j=0}^{M-1} \alpha_j |u_j\rangle_1$ when given new test point $\vec{x}_*$.

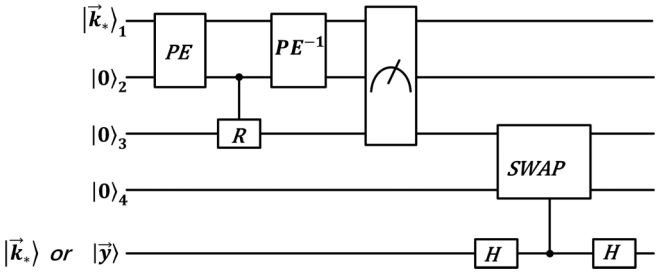

FIG. 2. The preparation circuit of the predictive mean value and variance value. Here PE and $PE^{\dagger}$ denote the phase estimation algorithm and inverse phase estimation algorithm of steps B2 and B3, respectively. The R' is controlled rotation of step B3. SWAP is the swap operation of step B4.

Step B3. Adding another register in the state $|0...0\rangle_2$ to the above state $|\vec{k}_*\rangle_1$. Then applying the matrix exponentiation technique of the quantum principal component analysis [34] to $|\vec{k}_*\rangle_1$ results in

$$|\varphi_1\rangle_1 = \sum_{r=1}^{R} |r\Delta t\rangle\langle r\Delta t|e^{-iKr\Delta t}|\vec{k}_*\rangle\langle\vec{k}_*|e^{iKr\Delta t} \tag{34}$$

for some large R. According to Lloyd *et al.* [34], the quantum phase estimation algorithm leads to

$$|\varphi_2\rangle_{12} = \sum_{j=0}^{M-1} \alpha_j|u_j\rangle_1|\lambda_j\rangle_2. \tag{35}$$

Step B4. Adding one qubit and rotating it from $|0\rangle_3$ to $\sqrt{1-(\frac{c}{\lambda_j+\sigma_M^2})^2}|0\rangle_3 + \frac{c}{\lambda_j+\sigma_M^2}|1\rangle_3$ controlled on $|\lambda_j\rangle_2$, where $c = O[\max_{\lambda_j}(\frac{1}{\lambda_j+\sigma_M^2})]^{-1} = O(\frac{1}{\kappa})$ is a chosen constant so that the value $\frac{c}{\lambda_j+\sigma_M^2}$ is as close to 1 as possible while still being less than 1. Yu *et al.* [33] pointed out that the maximum of $\frac{1}{\lambda_j+\sigma_M^2}$ as well as c depends on the actual choice of σ_M, but $\frac{c}{\lambda_j+\sigma_M^2} = \Omega(\frac{1}{\kappa})$ for all possible σ_M. Then we undo the phase estimation algorithm and discard the second register to obtain the state

$$|\varphi_3\rangle_{13} = \sum_{j=0}^{M-1} \alpha_j|\vec{u}_j\rangle_1 \left(\sqrt{1-\left(\frac{c}{\lambda_j+\sigma_M^2}\right)^2}|0\rangle_3 + \frac{c}{\lambda_j+\sigma_M^2}|1\rangle_3\right). \tag{36}$$

Step B5. Measuring the last register to get $|1\rangle$, the final state of the first register then approximates

$$|\varphi_4\rangle_1 = \frac{\sum_{j=0}^{M-1}\frac{c\alpha_j}{\lambda_j+\sigma_M^2}|\vec{u}_j\rangle_1}{\sqrt{\sum_{j=0}^{M-1}\left(\frac{c\alpha_j}{\lambda_j+\sigma_M^2}\right)^2}}. \tag{37}$$

The success probability of getting $|1\rangle$ is $\sum_{j=0}^{M-1}\frac{c^2\alpha_j^2}{(\alpha_j+\sigma_M^2)^2} = \Omega(\frac{1}{\kappa^2})$, which implies that $O(\kappa^2)$ repetitions are enough to yield the desirable state with a high probability, and that can be improved by amplitude amplification with $O(\kappa)$ repetitions.

Given the kernel function vector $|\vec{k}_*\rangle$ of a new test point $\vec{x}_*$, the value $|\varphi_4\rangle_1$ can be used to predict output $\bar{f}_*$ by evaluating the inner product of $|\vec{k}_*\rangle$ and $|\varphi_4\rangle_1$ via swap test [35], which employs the Hadamard transform on ancilla qubit $|0\rangle$. Then, by executing swap operations on $|\vec{k}_*\rangle$ and $|\varphi_4\rangle_1$ when the ancilla qubit is $|1\rangle$, followed by measuring the ancilla qubit in the basis $\{|0\rangle, |1\rangle\}$, the probability of success to get $|1\rangle$ is $\frac{1}{2} + \frac{1}{2}|\langle\varphi_4 \mid \vec{k}_*\rangle|^2$. Therefore we can get a specific value $\sum_{j=0}^{M-1}\frac{c\alpha_j^2}{\lambda_j+\sigma_M^2}$. Similarly, we perform the swap test on $|\vec{y}\rangle$ and $|\varphi_4\rangle_1$, in which the probability of success to get $|1\rangle$ is $\frac{1}{2} + \frac{1}{2}|\langle\varphi_4 \mid \vec{y}\rangle|^2$. Obviously, the probability is greater than $\frac{1}{2}$. It seems that we need to measure a constant number of times to get the desired result $\sum_{j=0}^{M-1}\frac{c\alpha_j\beta_j}{\lambda_j+\sigma_M^2}$ with probability close to 1. Because $\|\vec{k}_*\|$ and $\|\vec{y}\|$ can be acquired efficiently, we can get the mean value $\bar{f}_*$ and variance value $V[f_*]$ efficiently.

In the above operations, we may encounter a problem that the sign of the swap test result is ambiguous. $\sum_{j=0}^{M-1}\frac{c\alpha_j^2}{\lambda_j+\sigma_M^2}$ is always greater than zero, but $\sum_{j=0}^{M-1}\frac{c\alpha_j\beta_j}{\lambda_j+\sigma_M^2}$ may be a negative number. To trace the signs, the following method is applied: When the initial state $\frac{|0\rangle_1|\vec{k}_*\rangle_2+|1\rangle_1|\vec{y}\rangle_2}{\sqrt{2}}$ is created, follow steps B1–B4 to get state $|\varphi_4\rangle_1$ when the first register is $|0\rangle$ and do nothing when the first register is $|1\rangle$, that is, the state of the system equals $\frac{|0\rangle_1|\varphi_4\rangle_2+|1\rangle_1|\vec{y}\rangle_2}{\sqrt{2}}$, which then equals the swap test on the first register with $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$. The probability of success is $\Pr = \frac{1}{4} + \frac{1}{4}\langle\varphi_4 \mid \vec{y}\rangle$, which means we are able to track the signs.

TABLE I. The time complexity of each step of the whole algorithm.

Step of section	Time complexity
Step B1	$O(\text{poly}\log M)$
Step B2	$O(\frac{1}{\sqrt{P_k}}MN\log 1/N\delta)$
Step B3	$O(\log M\tau^{-3})$
Step B4 of Sec. IV	$O(\frac{\log(\frac{1}{\delta'})\log^2(\frac{\kappa}{\delta'})}{\log\log(\frac{1}{\delta'})} + \log M\tau^{-3})$

C. Runtime analysis

Let us start by discussing the time complexity of the whole algorithm. An overview of the time complexity of each step is listed in Table I. A detailed analysis of each step of this algorithm is depicted as follows.

The state $|\vec{y}\rangle$ can be generated in time $O(\text{poly}\log M)$ with the help of QRAM [36,37], so the time complexity of step B1 is $O(\text{poly}\log M)$. In step B2, the time complexity of preparing $|\vec{k}_*\rangle$ is analyzed in Sec. A, that is, $O(\frac{1}{\sqrt{P_k}}MN\log 1/N\delta)$. Then the density matrix exponentiation is performed in step B3, which is a powerful tool to investigate the properties of unknown density matrices. Lloyd *et al.* [34] pointed out that the ability to use r copies of K to apply the unitary operator e^{-iKt} allows us to exponentiate nonsparse matrices to accuracy $\tau = O(t^2/r)$ and to construct the eigenvalues and eigenvectors of matrix K in time complexity of $O(\log M\tau^{-3})$. Here M is the size of the training set. Next, in step B3 we also apply controlled rotation. The runtime is $O(\frac{\log(\frac{1}{\delta'})\log^2(\frac{\kappa}{\delta'})}{\log\log(\frac{1}{\delta'})})$ (κ is the conditional number of covariance matrix K), which is

relatively negligible compared to the time taken in step B2. In step B4, the inverse phase estimation has the same time analysis as the phase estimation in step B3. The final measurement only accounts for a constant factor, so the runtime of these two sections is negligible.

Therefore the total runtime of getting the predictive mean value and covariance value is $O[\kappa(\frac{1}{\sqrt{P_k}}MN\log\frac{1}{N\delta}\log M\tau^{-3}+poly\log M)]$. Certainly, it is still difficult to implement QRAMwith current technology. If QRAM is not used, the time complexity of that step will increase to $O(M)$. But this does not affect the time complexity of the whole algorithm. It is determined by the preparation of the kernel vector $|\vec{k}_*\rangle$, the time complexity of which is much larger than that of preparing $|\vec{y}\rangle$.

In general, dimension N is much smaller than sample number M. Our algorithm can achieve polynomial speedup against the classical counterparts. When $N, P_k, \kappa = O(\text{poly}\log M)$, quadratic acceleration is achievable.

D. Error analysis

In addition to time complexity analysis, another important criterion to measure the quality of a quantum algorithm is error analysis of the algorithm. Therefore we briefly analyze the error of the algorithm in this section.

In the proposed algorithm there are two steps that may introduce errors. One is from the preparation of $|\varphi_r\rangle$ in a finite quantum circuit, and the other is from the implementation of the phase estimation algorithm.

For the former, $|\varphi_r\rangle$ is obtained by a unitary operator of dimension infinity, which is impossible to implement. So we can consider its Taylor approximation $|\tilde{\varphi}_r\rangle$, it is prepared in a finite quantum circuit. Thus there is an error between the two states. The error is related to the dimension T, so we choose a good T to ensure that the error within δ. And any vector $\vec{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iN})^T$ has N $|\varphi_r\rangle$, so that of $|\varphi_{\vec{x}_i}\rangle$ is less than $N\delta$.

The latter error occurs in estimating λ_j by $O(1/t_0)$, which translates into a relative error of $O(1/\lambda_j t_0)$ in λ_j^{-1} [30]. If $\lambda_j \geqslant 1/\kappa$, taking $t_0 = O(\kappa/\varepsilon)$ induces a final error of ε.

Above, we explained that this error generated by the preparation of $|\varphi_r\rangle$ can be minimized within a controllable range by the dimension T. And the impact of the error generated by the phase estimation is not very large [30]. Therefore the error of this algorithm is acceptable, that is, it can ensure that the whole training model can predict the new input data well.

V. CONCLUSION

GPR makes sense in real-world applications, especially when a problem involves extrapolating from large data sets. However, classical algorithms cost too much when data sets are large, thus quantum GPR algorithm is proposed. In our GPR algorithm, we first propose a way to prepare the covariance matrix, which makes the algorithm distinct. This is also an innovational contribution of this paper. In the existing papers on quantum GPR [19], the authors used the given default covariance matrix. Second, a kernel vector is obtained by annihilation operator and creation operator, and the kernel function vector can be acquired by amplitude estimation without block encoding [38]. Finally, in order to get the exact measurement values, we propose a sign tracing algorithm. Our algorithm has a polynomial speedup or is even, compared with the classical counterpart. We hope that our algorithm, specifically, the key technologies used in our algorithm, will inspire more efficient quantum machine learning algorithms for application in a wider range of fields.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grants No. 61772134, No. 62171131, No. 61976053, and No. 62006105), Jiangxi Provincial Natural Science Foundation (Grant No. 20202BABL212004), Fujian Province Natural Science Foundation (Grant No. 2018J01776), and the Program for New Century Excellent Talents at Fujian Province University.

[1] J. Biamonte, P. Wittek, and N. Pancotti, Nature (London) **549**, 195 (2017).
[2] A. D. Alhaidari and T. J. Taiwo, J. Math. Phys. **58**, 022101 (2017).
[3] J. Li, S. Lin, K. Yu *et al.*, Quantum. Inf. Process. **21**, 18 (2022).
[4] P. W. Shor, SIAM J. Comput. **26**, 1484 (1997).
[5] D. Cao, IEEE. J. Quantum. Electron. **32**, 58 (2015).
[6] Y. Z. Xu, G. D. Guo, B. B. Cai, and S. Lin, Comput. Sci. **43**, 80 (2016).
[7] I. Cong, S. Lukin, and D. Mikhail, Nat. Phys. **15**, 1273 (2019).
[8] E. Farhi and H. Neven, arXiv:1802.06002.
[9] M. Zidan, A. H. Abdel-Aty, M. El-Shafei, M. Feraig, Y. Al-Sbou, H. Eleuch, and M. Abdel-Aty, Appl. Sci. **9**, 1277 (2019).
[10] C. Shao, Phys. Rev. A **102**, 042418 (2020).
[11] Y. Shi, Inf. Process. Lett. **81**, 23 (2002).
[12] C. H. Yu, F. Gao, Q. L. Wang, and Q. Y. Wen, Phys. Rev. A **94**, 042311 (2016).
[13] C. S. R. Silva and J. M. Fonseca, in *Artificial Intelligence and Algorithms in Intelligent Systems - Proceedings of 7th Computer Science On-line Conference, 2018*, edited by R. Silhavy (Springer Verlag, Cham, 2018), pp. 308–317.
[14] D. Loss and D. P. DiVincenzo, Phys. Rev. A **57**, 120 (1998).
[15] S. M. Miller and R. D. Luark, Int. J. Rock Mech. Min. Sci. Geomech. Abstr. **30**, 1631 (1993).
[16] C. Williams and C. Rasmussen, Adv. Neural Info. Proc. Sys., 8 (1995).
[17] R. Dürichen, M. A. F. Pimentel, L. Clifton, A. Schweikard, and D. A. Clifton, IEEE Trans. Biomed. Eng. **62**, 314 (2015).
[18] L. Ling, C. Pan, and P. Li, Appl. Mech. Mater. **421**, 523 (2013).
[19] Z. Zhao, J. K. Fitzsimons, and J. F. Fitzsimons, Phys. Rev. A **99**, 052331 (2019).
[20] C. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning* (The MIT Press, Cambridge, MA, 2005).
[21] B. C. Sanders, J. Phys. A: Math. Theor. **45**, 244002 (2012).

[22] K. Fujii, arXiv:quant-ph/0112090.
[23] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Phys. Rev. A **52**, 3457 (1995).
[24] V. V. Shende, S. S. Bullock, and I. L. Markov, IEEE Trans. Comput. Aided Des. **25**, 1000 (2006).
[25] M. Schuld, I. Sinayskiy, and F. Petruccione, Phys. Rev. A **94**, 022342 (2016).
[26] S. Lloyd, M. Mohseni, and P. Rebentrost, arXiv:1307.0411 v2.
[27] G. Brassard, P. Hoyer, and M. Mosca, AMS Contemp. Math. **305**, 53 (2002).
[28] A. E. Rastegin, Quantum Inf. Process. **17**, 179 (2018).
[29] N. Wiebe, A. Kapoor, and K. Svore, Quantum Inf. Comput. **15**, 316 (2015).
[30] A. Harrow, Phys. Rev. Lett. **92**, 097902 (2004).
[31] M. F. Ochs, R. Stoyanova, and F. Ariasmendoza, J. Magn. Reson. **137**, 161 (1999).
[32] A. Gily'en, Y. Su, G. H. Low, and N. Wiebe, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (Association for Computing Machinery, New York, 2019), pp. 193–204.
[33] C. H. Yu, F. Gao, and Q. Y. Wen, IEEE Trans. Knowl. Data. Eng. **33**, 858 (2021).
[34] S. Lloyd, M. Mohseni, and P. Rebentrost, Nat. Phys. **10**, 631 (2014).
[35] H. Buhrman, R. Cleve, J. Watrous, and R. deWolf, Phys. Rev. Lett. **87**, 167902 (2001).
[36] V. Giovannetti, S. Lloyd, and L. Maccone, Phys. Rev. Lett. **100**, 160501 (2008).
[37] W. Manuela, J. Barzen, F. Leymann, and M. O. Salm, in *2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C)* (IEEE, New York, 2021), pp. 95–101.
[38] C. P. Shao, J. Phys. A: Math. Theor. **53**, 045301 (2020).