通用赛道：量子离散绝热线性算法模拟验证

June 10, 2024

Abstract

量子线性算法的目标是求解线性方程组 $Ax = b$ 并得到解的量子态 $|x\rangle$。自从 HHL（Harrow, Hassidim, and Lloyd）算法问世以来，研究人员相继提出了多种创新性的量子线性算法，这些算法在复杂度方面不断取得重要突破。其中，量子离散绝热线性算法基于离散绝热理论，被公认为当前最为高效的量子线性算法。在本项目中，我们首先实现了基于 LCU (Linear combination of unitaries) 的块编码 (Block-Encoding)，它能够将形状为 $(2^n, 2^n)$ 的任意矩阵嵌入到一个更大的酉矩阵中，是许多量子算法实现的基础。基于此模块，我们进一步利用量子离散绝热线性算法使用 QPanda 完成了一个具体线性方程组的求解。实验显示，对于 2×2 维的线性方程组，我们的方法得到的解的量子态与归一化的真实解之间的保真度达到了 99% 以上，证明了我们方案的有效性。

1 基于 LCU 的块编码方案

1.1 算法原理

假设我们有一个 $2^n \times 2^n$ 的矩阵 M 和一个 n 量子比特态 $|\psi\rangle$，我们想制备态 $M|\psi\rangle / \|M|\psi\rangle\|$。这里 M 不必是酉矩阵，但假设我们可以把 M 写成酉矩阵的线性组合（事实上每个 M 都可以这样进行分解，因为 4^n 个 n 比特的泡利矩阵构成了所有 $2^n \times 2^n$ 矩阵线性空间的基。我们将在下一节详细讨论这一点）：

$$M = \sum_{j=1}^{m} \alpha_j V_j, \tag{1}$$

其中 α_j 为非负实数 (我们总能把复相位吸收到 V_j 中)。令 $\|\alpha\|_1 = \sum_j \alpha_j$，定义 W 是作用在 $\lceil logm \rceil$ 个辅助量子比特上的酉矩阵，它的作用类似于振幅编码，即：

$$W : |0\rangle \mapsto \frac{1}{\sqrt{\|\alpha\|_1}} \sum_j \sqrt{\alpha_j} |j\rangle, \tag{2}$$

实现酉矩阵 W 的对应线路的代码为如下所示，

```
#include "QAlg/Encode/Encode.h"
// 获得数据编码电路的酉矩阵，以得到 W
QCircuit get_prep_circuit(QVec qlist, std::vector<double> data)
{
    int n_qubits = ceil(log2(data.size()));
    Encode encode;
    encode.schmidt_encode(qlist, data);
    QCircuit cir = encode.get_circuit();
    return cir;
}
```

定义 V 是一组受控酉矩阵，其控制位为辅助量子比特，目标位为原始量子比特，它的作用为：

$$V : |j\rangle|\phi\rangle \mapsto |j\rangle V_j|\phi\rangle \tag{3}$$

即当控制位为 $|j\rangle$ 时，对目标位作用酉操作 V_j。因此 V 可以写为：

$$V = \sum_{j=1}^{m} |j\rangle\langle j| \otimes V_j, \tag{4}$$

实现酉矩阵 V 的电路的代码对应为：

```
QCircuit get_control_circuit(QVec control_qlist, QVec target_qlist, Eigen::MatrixXcd V_j, int j)
{
    int n_ancillary = control_qlist.size();
    QCircuit cir, cir1, cir2, cir3, cir4;
    // 当control_qlist为 j 时，对 target_qlist 作用酉矩阵 V_vec[j]
    for (int i = 0; i < n_ancillary; i++)
    {
        if (((j >> i) & 1) == 0)
        {
            cir << X(control_qlist[i]);
        }
    }
    cir << matrix_decompose_qr(target_qlist, Eigen_to_QStat(V_j), false).control(control_qlist);
    for (int i = 0; i < n_ancillary; i++)
    {
        if (((j >> i) & 1) == 0)
        {
            cir << X(control_qlist[i]);
        }
    }
    return cir;
}
```

利用酉矩阵 V 和 W，我们可以实现非酉矩阵 M 作用在量子态 $|\psi\rangle$ 上的效果。考虑以下步骤：

1. 准备初始量子态 $|0\rangle^{\otimes m}|\psi\rangle$。其中 $|0\rangle$ 为辅助量子位，包含 $\lceil logm \rceil$ 个量子比特。
2. 对辅助量子位作用酉操作 W。
3. 以辅助量子位为控制量子位，对整个量子态作用酉操作 V。
4. 对辅助量子位作用酉操作 $W^\dagger$。

最终我们得到的量子态为：

$$\frac{1}{\|\alpha\|_1}|0\rangle M|\psi\rangle + \sqrt{1 - \frac{\|M|\psi\rangle\|^2}{\|\alpha\|_1^2}}|\phi\rangle \tag{5}$$

其中，$|\phi\rangle$ 是一些我们不关心的归一化态。当我们测量辅助量子比特使之塌缩到 $|0\rangle$ 态时，原始量子比特将塌缩为 $M|\psi\rangle / \|M|\psi\rangle\|$，这是正是我们想要实现的块编码的效果。

由于 W、V 均是酉矩阵，因此可以使用量子线路实现。图1显示了一个具体的例子。其中，W 操作用于制备特定的量子态，其振幅由分解得到的酉矩阵的系数决定；V 操作则根据辅助量子位的状态选择对目标态作用哪个酉矩阵。

下列是我们构建电路的对应代码，其中 $alpha\_vec$ 代表公式 1.1中的 α, 以及 V_vec 代表 V，我们将在下一节详细讲述如何得到 $alpha\_vec$ 和 V_vec。

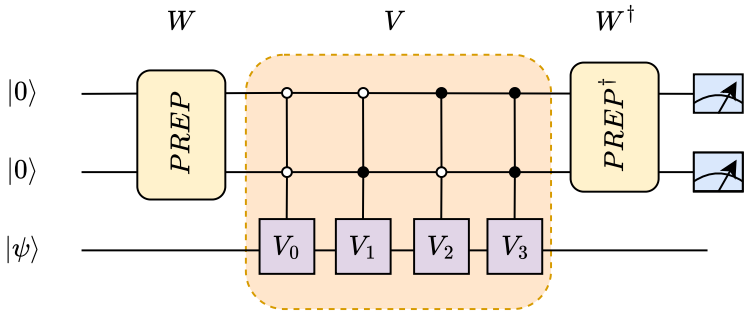

Figure 1: 基于 LCU 的块编码量子线路示意图

```
    auto qvm = new CPUQVM();
    qvm->init();
    QProg prog;
    auto q1 = qvm->qAllocMany(n);
    auto q2 = qvm->qAllocMany(n_ancillary);
    // 对辅助量子位作用 W
    QCircuit circuit1 = get_prep_circuit(q2, alpha_vec);
    prog << circuit1;
    // 以辅助量子位为控制量子比特，当辅助量子位为 i 时，对 q1 作用酉矩阵 V_vec[i]
    for (int i = 0; i < m; i++)
    {
        auto circuit2 = get_control_circuit(q2, q1, V_vec[i], i);
        prog << circuit2;
    }
    // 对辅助量子位作用 W.dagger
    prog << circuit1.dagger();
    QStat cir_matrix = getCircuitMatrix(prog, true);
    destroyQuantumMachine(qvm);
    return QStat_to_Eigen(cir_matrix);
```

1.2 矩阵分解

利用 LCU 对任意形状为 $2^n \times 2^n$ 的矩阵 M 进行块编码的必要条件是 M 可以写成一系列酉矩阵的线性组合，下面我们将在 n 比特的泡利基上证明这一点。

四个泡利矩阵为：

$$\sigma_0 = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \sigma_1 = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \sigma_2 = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \sigma_3 = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{6}$$

注意到每个泡利矩阵 σ_i 同时都是酉矩阵和厄密矩阵，因此自逆：$\sigma_i^{-1} = \sigma_i$。这意味着它们的特征值都在 $\{-1, 1\}$ 内。非单位阵的泡利矩阵之间反对称，即若 $\{P, Q\} \in \{X, Y, Z\}$ 且 $P \neq Q$，那么有 $PQ = -QP$。此外，两个不同的泡利矩阵乘积的迹为 0。

定义 $2^n \times 2^n$ 矩阵空间上的 Hilbert-Schmidt 内积为 $\langle A, B\rangle = \frac{1}{d}Tr(A * B)$。关于这个内积（对于 n = 1），四个泡利矩阵构成了一组标准正交基。这意味着每一个 2×2 矩阵 A 都可以写成泡利矩阵的线性组合：

$$A = \alpha_0\sigma_0 + \alpha_1\sigma_1 + \alpha_2\sigma_2 + \alpha_3\sigma_3 \tag{7}$$

其中 $\alpha_i = \langle \sigma_i, A\rangle$ 为复系数。但当 A 为厄密矩阵时，这些系数则为实数。

同理，考虑 n 量子比特的泡利矩阵，它是上述 2×2 泡利矩阵的 n 重张量积。对于任意一个 $2^n \times 2^n$ 的矩阵 A，我们可以将其分解到这 4^n 个 n 量子比特的泡利矩阵构成的正交基上。当 A 为厄密矩阵时，这些系数为实数。我们首先实现了一个给定量子比特输出由 pauli 矩阵的张量组成的一组正交基的函数，对应的代码实现为：

```cpp
// 获得由pauli矩阵的张量组成的一组正交基
std::vector<Eigen::MatrixXcd> get_pauli_bases(int n_qubits)
{
    // 定义pauli矩阵
    std::complex<double> image_(0.0, 1.0);
    Eigen::MatrixXcd matrix_I(2, 2);
    matrix_I << 1, 0, 0, 1;
    Eigen::MatrixXcd matrix_X(2, 2);
    matrix_X << 0, 1, 1, 0;
    Eigen::MatrixXcd matrix_Y(2, 2);
    matrix_Y << 0, -image_, image_, 0;
    Eigen::MatrixXcd matrix_Z(2, 2);
    matrix_Z << 1, 0, 0, -1;
    std::vector<Eigen::MatrixXcd> pauli_basis = {matrix_I, matrix_X, matrix_Y, matrix_Z};
    std::vector<Eigen::MatrixXcd> pauli_tensors;

    for (int i = 0; i < (1 << (2 * n_qubits)); ++i)
    {
        Eigen::MatrixXcd tensor_product = Eigen::MatrixXcd::Identity(1, 1);
        for (int j = 0; j < n_qubits; ++j)
        {
            // 提取第 j 位数字，每 4 轮一个循环
            int idx = (i >> (2 * j)) & 0b11;
            tensor_product = Eigen::kroneckerProduct(tensor_product, pauli_basis[idx]).eval();
        }
        pauli_tensors.push_back(tensor_product);
    }
    return pauli_tensors;
}
```

Listing 1: 获得由 pauli 矩阵的张量组成的一组正交基

回顾 LCU 的前提条件，我们需要将任意一个形状为 $2^n \times 2^n$ 的矩阵 M 分解为如下形式：

$$M = \sum_{j=1}^{m} \alpha_j V_j, \tag{8}$$

其中 α_j 为非负实数。

基于上述实现的 *get_pauli_bases*() 函数得到的正交基，我们在泡利基上分解得到复系数再转化为非负实数。实际上，任意一个复数 α 可以写为 $\gamma e^{i\theta}$ 的形式，其中 γ 是复数的模长,θ 是复数的辐角，且有 $\|e^{i\theta}\| = 1$。因此，我们可以将复相位 $e^{i\theta}$ 吸收进相应的泡利矩阵 P 中，得到 $e^{i\theta}P$，其仍然是一个酉矩阵。对应的代码实现为：

```cpp
// 定义LCU，前一项是正实系数，后一项是对应的酉矩阵
typedef std::vector<std::pair<double, Eigen::MatrixXcd>> LCU;
// 将矩阵分解为一系列酉矩阵的线性组合的函数
LCU linear_combination_pauli(Eigen::MatrixXcd hamiltonian)
{
    int n_qubits = log2(hamiltonian.rows());
    assert(1 << n_qubits == hamiltonian.rows());
    // 生成n量子比特的Pauli矩阵的张量积
```

```
    auto pauli_tensors = get_pauli_bases(n_qubits);
    LCU decompose;
    // 分解Hamiltonian
    for (const auto &pauli_matrix : pauli_tensors)
    {
        std::complex<double> coefficient = (pauli_matrix.adjoint() * hamiltonian).trace() /
    static_cast<std::complex<double>>(hamiltonian.rows());
        double abs_coefficient = abs(coefficient);
        std::complex<double> phase = coefficient / abs_coefficient; // 相位因子 e^{i\theta}
        if (abs_coefficient > 1e-10)                                // 避免添加零系数
        {
            // 应用相位因子，将系数转为正实数
            Eigen::MatrixXcd adjusted_pauli_matrix = phase * pauli_matrix;
            decompose.push_back({abs_coefficient, adjusted_pauli_matrix});
        }
    }
    return decompose;
}
```

Listing 2: 将任意一个 $2^n \times 2^n$ 形状的矩阵分解为一系列酉矩阵的线性组合

基于以上实现的两个函数，我们可以将给定哈密顿量，分解为公式 1.1中的 α 和 V，他们分别用 $alpha\_vec$ 和 V_vec 表示。

```
    // 将输入矩阵分解为若干酉矩阵的线性组合 (LCU)
    LCU M = linear_combination_pauli(hamiltonian);
    int n = int(log2(hamiltonian.rows()));
    int m = M.size(); // 分解得到的酉矩阵个数
    if (m == 1)
        return hamiltonian;
    int n_ancillary = ceil(log2(m)); // 辅助量子比特数
    std::vector<double> alpha_vec;       // 存储实系数
    std::vector<Eigen::MatrixXcd> V_vec; // 存储酉矩阵
    double sum_alpha = 0;
    for (int i = 0; i < m; ++i)
    {
        sum_alpha += M[i].first;
        V_vec.push_back(M[i].second);
    }
    for (int i = 0; i < (1 << n_ancillary); ++i)
    {
        if (i < m)
            alpha_vec.push_back(sqrt(M[i].first / sum_alpha));
        else
            alpha_vec.push_back(0);
    }
```

2 基于 QPanda 的量子离散绝热线性算法

2.1 算法原理

假设 $A \in \mathbb{C}^{N\times N}$ 是一个可逆矩阵，$|b\rangle \in \mathbb{C}^N$ 是一个归一化向量。给定目标误差 ϵ，量子线性系统求解器 (QLSP) 的目标是制备一个归一化态 $|x_a\rangle$，其为线性系统归一化解 $|x\rangle = A^{-1}|b\rangle/|A^{-1}|b\rangle|_2$ 的 ϵ-近似态，即满足 $\||x_a\rangle\langle x_a| - |x\rangle\langle x|\|_2 \leq \epsilon$。

基于绝热量子计算 (AQC) 的 QLSP 算法的第一步是将 QLSP 转换为等效的特征值问题。按照题目要求，令 $Q_b = I_N - |b\rangle\langle b|$。构造

$$H_0 = \sigma_x \otimes Q_b = \begin{pmatrix} 0 & Q_b \\ Q_b & 0 \end{pmatrix}, \tag{9}$$

其中 H_0 是厄米矩阵，且 H_0 的解空间为 $\mathrm{Null}(H_0) = \mathrm{span}\{|b\rangle, |\bar{b}\rangle\}$。这里 $|b\rangle = |0, b\rangle := (b, 0)^\top, |\bar{b}\rangle = |1, b\rangle := (0, b)^\top$。$H_0$ 的维度是 2^N，需要一个辅助量子比特来扩大矩阵块。定义

$$H_1 = \sigma_+ \otimes (AQ_b) + \sigma_- \otimes (Q_b A) = \begin{pmatrix} 0 & AQ_b \\ Q_b A & 0 \end{pmatrix}. \tag{10}$$

这里 $\sigma_\pm = \frac{1}{2}(\sigma_x \pm i\sigma_y)$。注意如果 $|x\rangle$ 满足 $A|x\rangle \propto |b\rangle$，则有 $Q_b A|x\rangle = Q_b|b\rangle = 0$。于是 $\mathrm{Null}(H_1) = \mathrm{span}\{|x\rangle, |\bar{b}\rangle\}$ 其中 $|x\rangle = |0, x\rangle$。

如果我们能够准备 H_1 的零能态 $|x\rangle$，则可以通过 AQC 方法解决 QLSP。令

$$H(f(s)) = (1 - f(s))H_0 + f(s)H_1, \quad 0 \le s \le 1 \tag{11}$$

函数 $f : [0, 1] \to [0, 1]$ 是一个严格递增的映射，其中 $f(0) = 0, f(1) = 1$。

考虑如下绝热演化过程：

$$\partial_S |\psi_T(s)\rangle = -iTH(s) |\psi_T(s)\rangle \tag{12}$$

其中 $0 \le s = \frac{t}{T} \le 1$，$|\psi_T(0)\rangle = |\tilde{b}\rangle$，参数 T 称为 AQC 的演化时间。

量子绝热定理指出 [SSO19]，对于任意的 $0 \le s \le 1$，

$$1 - |\langle\psi_T(s)|\widetilde{x}(s)\rangle|^2 \le \eta^2(s) \tag{13}$$

此处 $\eta(s)$ 为

$$\eta(s) = C\left\{\frac{\|H^{(1)}(0)\|_2}{T\Delta^2(0)} + \frac{\|H^{(1)}(s)\|_2}{T\Delta^2(f(s))} + \frac{1}{T}\int_0^s \left(\frac{\|H^{(2)}(s')\|_2}{\Delta^2(f(s'))} + \frac{\|H^{(1)}(s')\|_2^2}{\Delta^3(f(s'))}\right) ds'\right\} \tag{14}$$

因此当 $s = 1$ 时，我们有：

$$|\langle\psi_T(1)|\widetilde{x}(1)\rangle|^2 \ge 1 - \eta^2(s) \tag{15}$$

其中，$|\tilde{x}(1)\rangle = |\tilde{\tilde{x}}\rangle = |0, x\rangle$，包含我们要求得的真实解 x。

则上式被简化成：

$$H(s) = (1 - s)H_0 + sH_1, \quad s \in [0, 1], \quad \Delta s = \frac{1}{200} \tag{16}$$

因此我们的目标转化为求解 $|\psi_T(1)\rangle$。但值得注意的是，使用该方法得到的解与真实解之间可能存在相位差，这从式15中可以看出。

下面根据题目要求，我们使用数值方法近似求解绝热演化过程，我们有:

$$|\psi_T(n \cdot \Delta s)\rangle \approx exp(-iT\Delta sH((n - \frac{1}{2}) \cdot \Delta s))|\psi_T((n - 1)\Delta s)\rangle \tag{17}$$

当 $\Delta s = \frac{1}{200}$ 时，我们通过连续 200 步的迭代，可以得到：

$$|\psi_T(1)\rangle \approx \prod_{n=1}^{200} exp(-iT\Delta sH((n - \frac{1}{2}) \cdot \Delta s)) |\psi_T(0)\rangle \tag{18}$$

进一步使用一阶近似 $e^{-iT\Delta sH((n-\frac{1}{2})\cdot\Delta s)} \approx I - iT\Delta sH((n-\frac{1}{2})\cdot\Delta s)$, 可以得到:

$$|\Psi_T(1)\rangle \approx \prod_{n=1}^{200}(I - iT\Delta sH((n-\frac{1}{2})\cdot\Delta s))\,|\Psi_T(0)\rangle\,. \tag{19}$$

我们将 $\prod_{n=1}^{200} I - iT\Delta sH((n-\frac{1}{2})\cdot\Delta s)$ 得到的矩阵记为 H', 然后使用块编码矩阵得到需要的 $U_{H'}$。

由于 $s = \frac{t}{T}$, 且绝热演化的要求时间 t 变化缓慢, 即 $\Delta(t) \approx 0$, 同时题目要求 $\Delta s = \frac{1}{200}$。由公式 $\Delta(s) = \frac{\Delta(s)}{T} = \frac{1}{200}$, 可以看出 T 的选择将影响算法的成功率。本方案将测试 T, 选取最优 T 进行计算。

2.2 实验验证

下面的实验以题目给出的方程组为例:

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \tag{20}$$

其中, 真解对应的量子态为 $|x\rangle = \frac{\sqrt{2}}{2}[1,1]^T$。

设使用绝热演化得到的解的量子态为 x_a。从理论上来说, 如果演化时间 T 越大, x_a 与 x 之间的保真度也越大。于是我们首先在不进行一阶近似的情况下, 也就是使用公式18验证 T 与保真度 $F = |\langle x_a|x\rangle|$ 的关系。得到的结果如图2所示。从图中可以发现, 随着演化时间 T 的增大, 解的保真度 F 将会趋近于 1。但当 T 较小时, 保真度会出现抖动的情况。

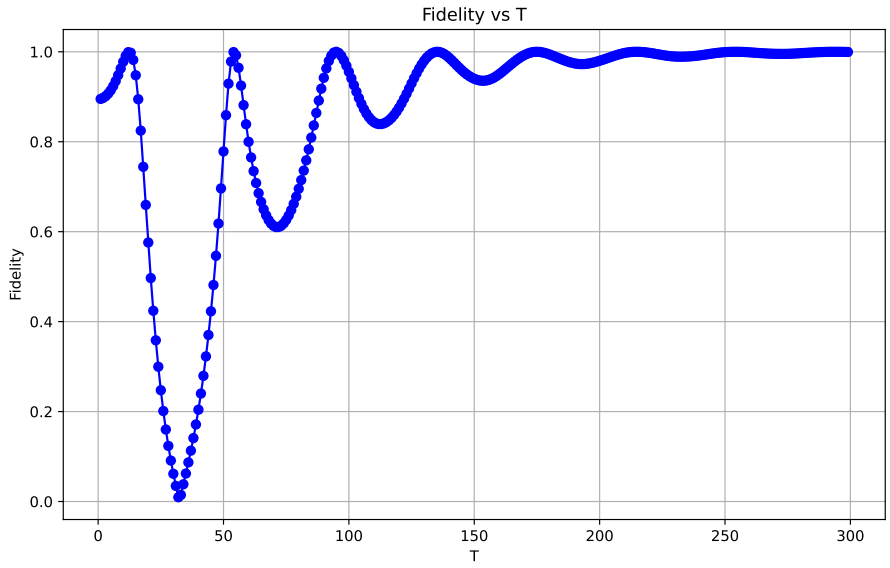

Figure 2: 数值计算得到的演化时间与保真度的关系

再次使用一阶近似对保真度与演化时间的关系进行探索, 即使用公式19验证, 得到的结果如图3所示。可以发现当 T 较小时, 一阶近似得到的结果与数值计算得到的结果基本一致, 但当 T 较大时, 一阶近似得到的结果非常不稳定, 容易出现较大的偏差。

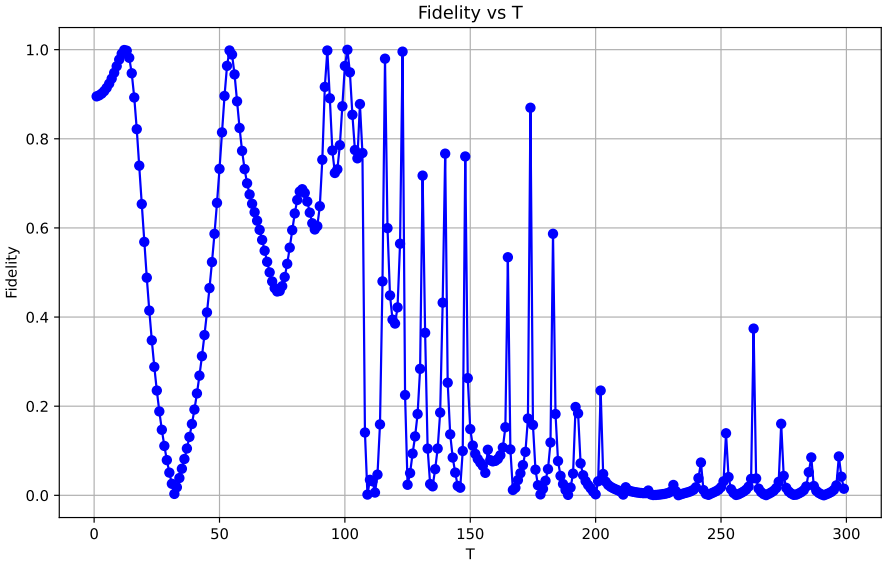

Figure 3: 一阶近似计算得到的演化时间与保真度的关系

受题目要求所限，在考虑一阶近似的情况下，我们需要找到一个合适的 T 值，使得一阶近似的结果与数值计算的结果基本一致且保真度足够大，即最佳的 T 满足：

$$\max_T \mathrm{fid}_T := |\langle\psi_T|\widetilde{x}\rangle|. \tag{21}$$

然而，保真度的最大化需要知道精确解 x，这是不实际的。但我们可以参考 QAOA 的目标函数，转化为解决下面的极小化问题：

$$\mathrm{obs}_T = \min_T \langle\psi_T|H_1^2|\psi_T\rangle. \tag{22}$$

对于每一个选择的 T，我们都会计算出期望值 $\langle\psi_T|H_1^2|\psi_T\rangle$。然后，在经典计算机上调整下一个 T，使目标函数最小。由于参数数量较少（仅有 T 一个可变量），因此我们直接在 1-200 上遍历 T 值，记录最小期望值 obs 对应的 T，代入公式19中。

2.3 实验结果

I). 对于题目指定的方程组如式 20, 我们的算法给出的解的保真度达到 99.96%。

II). 对于任意 2×2 维的厄密系数矩阵 A 和任意复向量 b 组成的线性方程组的求解问题。本方案随机生成 100 个实例，用我们的方法得到的解 x_a 与真实解之间保真度的平均值为 99.35%，证明了我们方法的有效性。

III). 对于任意 4×4 维的厄密系数矩阵 A 和任意复向量 b 组成的线性方程组的求解问题。本方案随机生成 100 个实例，用我们的方法得到的解 x_a 与真实解之间保真度的平均值为 76.67%，这主要是由于 Δs 太大导致的，当我们把 Δs 取为 1/1000 时，得到的平均保真度为 92.34%。

References

[SSO19] Yiğit Subaşı, Rolando D Somma, and Davide Orsucci. Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing. *Physical review letters*, 122(6):060504, 2019.