

1. 功能介绍

pyQLAS v1.1.0 是一款用于线性方程组求解的量子计算 python 库。与之前版本的 pyQLAS 库相比，pyQLAS v1.1.0 将底层从 QPanda2 替换成了 QPanda3。

除了线性方程组求解这一核心功能外，pyQLAS v1.1.0 的其他细节功能包括：

- 1) 提供量子虚拟机和量子真机求解功能；
- 2) 提供自定义算法参数功能；
- 3) 提供完整的计算数据记录功能。

2. 接口介绍

pyQLAS v1.1.0 提供 3 个接口供用户调用。

2.1 接口一

接口一的代码如图 1 所示。

```
def simple_QLS(A, b,
               api = '',
               machine = 'origin_wukong',
               amend = False,
               loss_run_kind = 'ideal',
               grad_run_kind = 'ideal',
               iter_step = 0,
               iter_pres = 1E-3,
               train_step = 0,
               train_pres = 1E-4,
               layer = 1,
               shots = 10000,
               delta = 1E-2,
               learn_rate = [0.005, 0.05, 0.2, 0.5, 1.0, 1.5, 2.0, 3.0],
               screen = False,
               write_file = False,
               check_input = False):
```

图 1 接口一的代码

其入参和返回值的详细信息如表 1 所示。

表 1 接口一的输入和输出数据信息明细

输入数据信息		
名称	类型	说明
A	Numpy 矩阵	必须为非 0 可逆方阵

		维度必须为 2 的整指数次方
b	Numpy 向量	必须为非 0 向量 维度必须与矩阵维度相同
api	string	用户自己的本源云 api_key
machine	string	本源云提供的机器名称，具体可查阅 QPanda3: 昆腾云服务 。
amend	bool	误差缓解开关
loss_run_kind	string	损失函数计算方式 = ideal，表示使用纯理论计算 = dummy，表示使用量子虚拟机计算 = real，表示使用量子真机计算
grad_run_kind	string	梯度计算方式 = ideal，表示使用纯理论计算 = dummy，表示使用量子虚拟机计算 = real，表示使用量子真机计算
iter_step	int	iter_VQLS 的最大迭代次数 = 0，表示无最大迭代步限制
iter_pres	float	iter_VQLS 的迭代收敛精度
train_step	int	VQLS 的最大训练次数 = 0，表示无最大训练步限制
train_pres	float	VQLS 的训练收敛精度
layer	int	使用的拟设层数
shots	int	线路的 shots 数
delta	float	计算梯度使用的差分步长
learn_rate	list	参数更新中使用的学习率
screen	bool	屏幕输出开关
write_file	bool	文件输出开关
check_input	bool	是否启动入参检查
返回数据信息		

名称	类型	说明
x	Numpy 向量	程序返回的 $Ax = b$ 的解

2.2 接口二

接口二的代码如图 2 所示。

```
def QLS(A, b, qlas):
    check_solver(A, b, qlas)
    x = iter_VQLS(A, b, qlas)
    return x
```

图 2 接口二的代码

其中入参 qlas 类的变量明细如表 2 所示。

表 2 pyQLAS 中 qlas 类的变量明细

输入数据信息		
名称	类型	说明
api	string	见表 1 中的 api 入参说明
machine	string	见表 1 中的 machine 入参说明
amend	bool	见表 1 中的 amend 入参说明
loss_run_kind	string	见表 1 中的 loss_run_kind 入参说明
grad_run_kind	string	见表 1 中的 grad_run_kind 入参说明
iter_step	int	见表 1 中的 iter_step 入参说明
iter_pres	float	见表 1 中的 iter_pres 入参说明
train_step	int	见表 1 中的 train_step 入参说明
train_pres	float	见表 1 中的 train_pres 入参说明
layer	int	见表 1 中的 layer 入参说明
shots	int	见表 1 中的 shots 入参说明
delta	float	见表 1 中的 delta 入参说明
learn_rate	list	见表 1 中的 learn_rate 入参说明
thetas	嵌套 list	存储每个训练步中的线路参数
loss	list	存储每个训练步中的损失函数

iter_cx	Numpy 向量	存储单次迭代步中的理论解
iter_qx	Numpy 向量	存储单次迭代步中的量子解
iter_err	float	存储迭代精度
root_path	string	存储根目录路径
data_path	string	存储输出数据文件路径
inf_path	string	存储输出信息文件路径
solve_time	string	存储计算开始时间
cx	Numpy 向量	存储方程组理论解
qx	Numpy 向量	存储方程组量子计算解
err	float	存储 pyQLAS 库最终求解的误差
screen	bool	见表 1 中的 screen 入参说明
write_file	bool	见表 1 中的 write_file 入参说明
check_input	bool	见表 1 中的 check_input 入参说明
train_count	int	存储总训练步数
iter_count	int	存储总迭代步数
task_count	int	存储总提交的任务数量，只计算量子虚拟机和量子真机的任务数
cir_count	int	存储总提交的线路数量，只计算量子虚拟机和量子真机的任务数

与接口一相比，接口二将 qlas 类直接作为入参，并在计算过程中直接修改 qlas 类的变量。在程序计算结束后，用户可以通过 qlas 类查看更多的计算信息，而使用接口一只能获取最终的方程组的量子解。

qlas 类型变量的初始化方式为，qlas = qlas_class()。

3. 前置依赖

pyQLAS v1.1.0 的前置依赖信息如图 3 所示。除了 os, sys, copy, math, datetime 等 python 标准库外，pyQLAS v1.1.0 还需要依赖 numpy 库和 pyqpanda3 库。

```
import pyqpanda3 as pq
import numpy as np
import os, sys, copy, math
from datetime import datetime
```

图 3 pyQLAS v1.1.0 的依赖情况

- 1) numpy 库的版本建议不低于 1.26.1;
- 2) pyqpanda3 库的版本建议不低于 0.1.1;
- 3) Python 的版本建议不低于 3.9.2rc1。

4. 算法流程图

算法流程如图 4 所示。

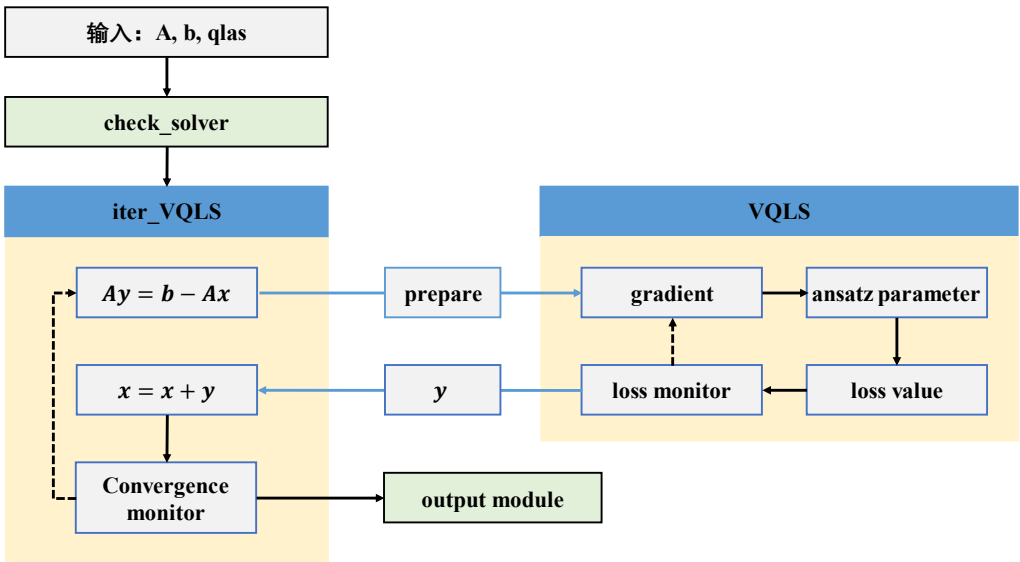


图 4 算法流程图

步骤	说明
1:	确定求解器的输入数据
2:	check_solver 模块对输入参数做检查，检查无误后调用迭代 VQLS 求解器 iter_VQLS
3:	设置方程组的初始解为 $x = 0$ 。
4:	生成迭代方程 $Ay = b - Ax$ 。
5:	调用 VQLS 求解器求解迭代方程，并得到解 y 。

- 6: 更新解 $\mathbf{x} = \mathbf{x} + \mathbf{y}$ ，并判断更新后的解是否收敛。如果不收敛则返回步骤 4，否则进入步骤 6。
- 7: 进入输出模块。

5. 注意事项

5.1 云服务后端选取

当用户提供的入参 “loss_run_kind” 或 “grad_run_kind” 为 “real” 时，代表用户希望某些线路使用真实量子计算机求解。程序会试图将任务提交给本源量子云上名为 “machine” 的真实量子计算机。

可用后端名称可能随着硬件的更新不定期做出调整。若用户给出的 “machine” 名称非法，则程序会自动爆出如下错误：

```
service online = ['72', 'WK_C102_400']
Error: illegal service name: origin_wukong
```

其中，service online 列出了目前可用的真实量子计算机名。用户可在其中选取自己想用的后端。

5.2 shots 数量约束

当使用 “ideal” 或 “dummy” 求解器时，shots 数量只需要大于 0 即可。当使用 “real” 求解器时，shots 数量最高为 10000。

5.3 check_input 参数

当 check_input = True 时，程序会在屏幕上输出所有入参，并等待用户的屏幕输入以决定是否继续执行程序，方便用户检查入参。

5.4 计算精度定义

pyQLAS 的迭代收敛精度 (iter_precision) 是相对精度，其定义为

$$\text{iter_precision} = \frac{\|\mathbf{x}_c - \mathbf{x}_q\|_2}{\|\mathbf{x}_c\|_2} \quad (1)$$

其中 $\mathbf{x}_c$ 是理论解， $\mathbf{x}_q$ 是 pyQLAS 给出的量子解。

6. 测试用例

这里给出 `ideal`, `dummy` 和 `real` 三种模式下的 `simple_VQLS` 接口返回结果。测试使用的线性方程组如下图所示：

```
dim = 4
A = np.eye(dim)
b = np.random.rand(dim)
```

`loss_run_kind = ideal`, `grad_run_kind = ideal` 下的求解结果如下图所示。可见 `ideal` 求解器经过 3 次迭代，8 次训练后得到满足精度要求（ $1E-3$ ）的解。

```
0 10 20
iter count = 3
train count = 8
error = 3.647E-05
task count = 0
circuit count = 0

4.27942E-01  4.27930E-01
3.33762E-01  3.33796E-01
5.13442E-01  5.13437E-01
4.70848E-01  4.70849E-01
```

`loss_run_kind = dummy`, `grad_run_kind = dummy` 模式下的求解结果如下图所示。可见 `dummy` 求解器经过 5 次迭代，5 次训练，运行 150 次线路后得到满足精度要求的解。

```
iter count = 5
train count = 5
error = 8.701E-04
task count = 0
circuit count = 150

5.12724E-01  5.13502E-01
5.95588E-01  5.95815E-01
7.11694E-01  7.11769E-01
2.14397E-01  2.14704E-01
```

`loss_run_kind = real`, `grad_run_kind = ideal` 模式下的求解结果如下图所示。可见此时经过 5 次迭代，5 次训练，提交 15 次任务，运行 130 个线路后得到满足精度要求的解。

```

iter count = 5
train count = 5
error = 2.691E-04
task count = 15
circuit count = 130

2.65024E-01  2.65037E-01
8.74431E-01  8.74494E-01
6.32687E-01  6.32944E-01
4.99844E-01  4.99893E-01

```

7. 性能评估

pyQLAS 使用高效硬件拟设，在高比特数下表达能力有限制。这里使用运行的线路数量表征求解器性能，求解器使用 dummy，结果如图 5 所示。图中横坐标 N 表示使用 N 个比特和 N 层拟设线路。从结果可以看出，pyQLAS 求解所执行的线路数随着 N 的增长呈指数级增加。这表明：pyQLAS 使用的迭代 VQLS 求解器并不能支撑大比特数求解。建议使用本版本时，输入的矩阵维度不高于 32（比特数不高于 5）。

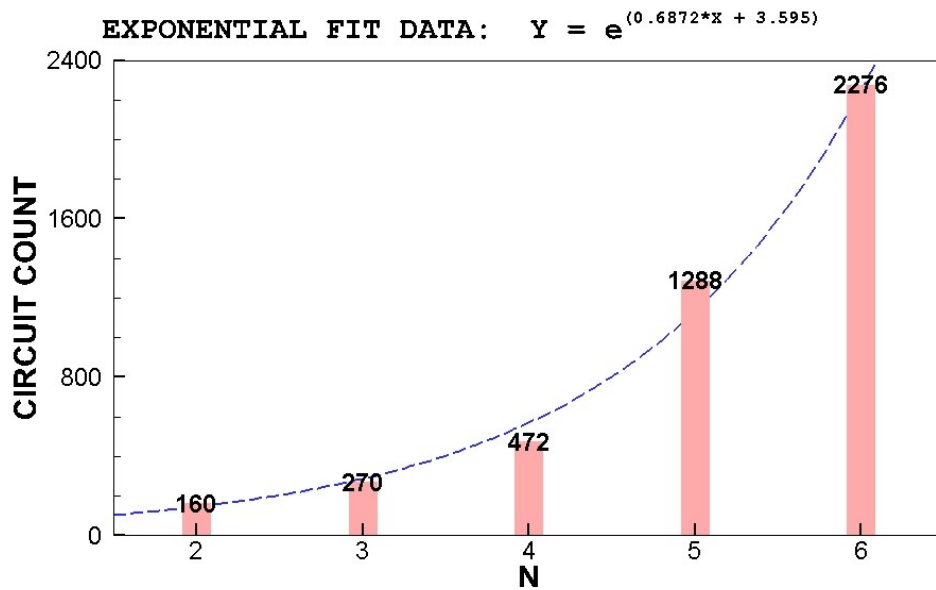


图 5 pyQLAS 求解器性能