

Assignment 1

Quantum Information and Computing AA 2022–23

James Chryssanthacopoulos
1 November 2022



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

- First program to test basic features
- Contains module and function to compute square root

```
module first_module
  real*8 var1, var2

  contains
    function mysqrt(x) result(sx)
      real*8 x
      real*8 sx
      sx = sqrt(x)
    end function

end module first_module
```

- Program output

```
The square root of 5.0 is 2.2361
```

Exercise 1: Cloud Veneto



- Key-pair and virtual machine created on CloudVeneto
- Private key copied onto gateway machine
- SSHed into gateway machine, then VM
- Installed gfortran
- `git` cloned my repository (see Slide 7)
- All code compiled and executed

Exercise 2: Number precision



- Program to test limits of integers and real numbers

- **Part (a)**

- Add 2.000.000 and 1 using `INTEGER*2` and `INTEGER*4`
- Since `INTEGER*2` only has range of $\approx 10^4$, storing 2.000.000 causes overflow

The sum of -31616 and 1 using `INTEGER*2` is -31615

The sum of 2000000 and 1 using `INTEGER*4` is 2000001

- **Part (b)**

- Sum $\pi \cdot 10^{32}$ and $\sqrt{2} \cdot 10^{21}$ with single and double precision
- Since single has 8 digits of precision, summing has no effect

The sum of 3.14159278E+32 and 1.41421360E+21 using `REAL*4` is 3.14159278E+32

The sum of 3.1415926535897933E+32 and 1.4142135623730950E+21 using `REAL*8` is 3.1415926536039354E+32

Exercise 3: Performance testing



- Program to implement matrix multiplication and time it
- Matrices multiplied using three for loops, resulting in $\mathcal{O}(n^3)$ time complexity

```
do i = 1, n_1
  do j = 1, n_4
    do k = 1, n_2
      matrix3(i, j) = matrix3(i, j) + matrix1(i, k) * matrix2(k, j)
    end do
  end do
end do
```

- Two different loop orders used, row-col-inner and inner-col-row
- Methods timed against builtin matmul method, for example

Elapsed time for row-col-inner = 1.800000000E-05

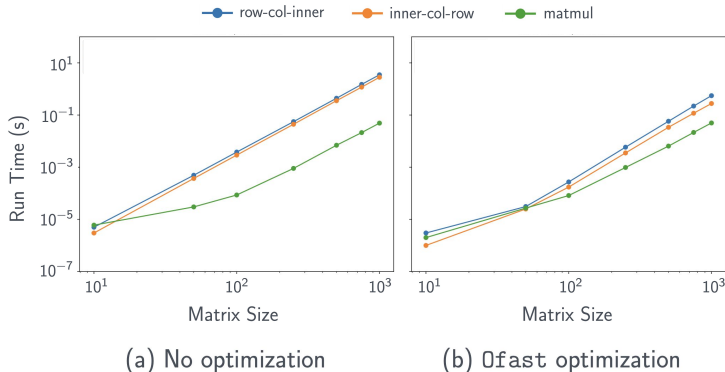
Elapsed time for inner-col-row = 1.000000000E-05

Elapsed time for matmul = 3.520000000E-04

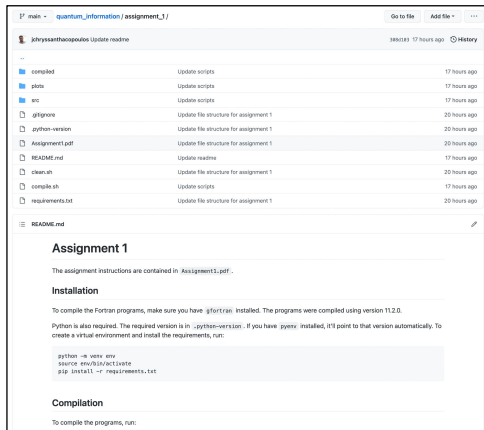
Exercise 3: Performance testing (cont'd)



- Different optimizations were used: 01–03, 0s, and 0fast
- 0fast reduced performance gap between `matmul` and custom methods the most, $\sim \mathcal{O}(n^{2.8})_{\text{custom}}$ to $\sim \mathcal{O}(n^{2.2})_{\text{matmul}}$



Code on GitHub with instructions to install, compile, and run
https://github.com/jchryssanthacopoulos/quantum_information



The screenshot shows a GitHub repository page for 'quantum_information' by user 'jchryssanthacopoulos'. The repository has 388 files and 17 hours ago was updated. The file list includes:

| File | Update | Time |
|------------------|--|--------------|
| compiled | Update scripts | 17 hours ago |
| plots | Update scripts | 17 hours ago |
| src | Update scripts | 17 hours ago |
| .gitignore | Update file structure for assignment 1 | 20 hours ago |
| .python-version | Update file structure for assignment 1 | 20 hours ago |
| Assignment1.pdf | Update file structure for assignment 1 | 20 hours ago |
| README.md | Update readme | 17 hours ago |
| clean.sh | Update file structure for assignment 1 | 20 hours ago |
| compile.sh | Update scripts | 17 hours ago |
| requirements.txt | Update file structure for assignment 1 | 20 hours ago |

The README.md file content is as follows:

Assignment 1

The assignment instructions are contained in 'Assignment1.pdf'.

Installation

To compile the Fortran programs, make sure you have 'gfortran' installed. The programs were compiled using version 11.2.0. Python is also required. The required version is in '.python-version'. If you have 'pyenv' installed, it'll point to that version automatically. To create a virtual environment and install the requirements, run:

```
python -m venv env
source env/bin/activate
pip install -r requirements.txt
```

Compilation

To compile the programs, run: