**README file for quantum Navier-Stokes simulation software**

All simulation software for the quantum algorithm for solving the Navier-Stokes equations was written by: *Frank Gaitan, Laboratory for Physical Sciences, College Park, MD, USA*.

The various functions and scripts used to implement the simulation are written using the MATLAB language.

To run the simulation, carry out the following 2 steps:

1. To begin, run the script QNS\_InputData.m. This assigns values to physical variables that are used in Step 2 to initialize the simulation parameters.
2. Next run the function QNavStokes\_Solvr.m. This function implements the simulation and writes its results to files. The following functions are called in the following order:
3. InitCalcParms.m is called to initialize simulation parameters for quantum Navier-Stokes algorithm.
4. IntegrateODE.m is called to numerically integrate Navier-Stokes ODEs.
5. WriteResults.m is called to write the results to files.
6. Note that a number of statements are written to the MATLAB command window announcing that the simulation has finished and the results written to files.

The functions called in Step 2 do the following:

2a. InitCalcParms.m calls the following functions:

1. Set\_XGrid.m sets up the 1-dimensional spatial grid for simulation.
2. Calc\_Noz\_Area.m calculates the nozzle area at each grid-point
3. Calc\_ExactResultsmSW.m (Calc\_ExactResultspSW.m) calculates the exact solution to the nozzle flow problem when a shock-wave is not (is) present. This solution is used to determine the quality of the simulation results.
4. SetInCond.m sets the initial condition for the simulation.
5. InitParms.m initializes the parameters used in the quantum ODE algorithm.
6. IPrtn.m initializes the partition of the time axis into subintervals and sub-subintervals.

2b. IntegrateODE.m calls the following functions:

1. BldTPoly.m builds the Taylor polynomials l^{s}\_{i}(t) for subinterval i at all interior grid-points.
2. IntegrateGij.m uses the quantum ODE algorithm to evaluate the sum appearing in Eq. (5) of the paper: F. Gaitan,Finding flows of a Navier-Stokes fluid through quantum computing. npj Quantum Inf 6, 61 (2020). <https://doi.org/10.1038/s41534-020-00291-0>.
3. CalcBCmSW.m (CalcBCpSW.m) uses the boundary conditions to determine the computational flow variables at the boundary points when a shock-wave is not (is) present.
4. Calc\_FlowVarResults.m evaluates the physical flow variables from the computational flow variables.

2c. WriteResults.m uses standard MATLAB functions to open, write to, and close files that store simulation results. The filenames appearing in this function are those used by author in his simulations. The user should rename them to match his/her PATH.

The functions called in IntegrateODE call the following functions:

1. BldTPoly.m calls:
2. Derivs.m which evaluates the ODE driver function and all its time derivatives up to order r at each interior grid-point at start of current subinterval.
3. BldTMat.m calculates and stores in a matrix, for each time sub-subinterval in the current time subinterval and each interior grid-point, the Taylor polynomial coefficients for each component of the ODE driver function.
4. NextInCond.m evaluates the initial condition for the next sub-subinterval in the current subinterval.
5. IntegrateGij.m calls:
6. FuncOrc.m calculates the summand appearing in Eq. (5) of the paper, ‘’F. Gaitan, Finding flows of a Navier-Stokes fluid through quantum computing’’ cited above at each interior grid-point for all knot times in the current sub-subinterval.
7. MeanOrc.m evaluates the mean of rescaled and shifted summand over all knot times in current sub-subinterval.
8. QAmpEst.m uses quantum amplitude estimations algorithm to evaluate right-hand side of Eq. (21) in Supplementary Information for F. Gaitan, Finding flows of a Navier-Stokes fluid through quantum computing cited above.

The functions in BldTPoly.m call the following functions:

1. Derivs.m calls:
2. CalcFlux.m calculates the flow fluxes F at all interior grid-points.
3. CalcSource.m calculates the flow sources J at all interior grid-points.
4. CalcFunc.m calculates the ODE driver function at all interior grid-points.
5. CalcfBvalsmSW.m (CalcfBvalspSW.m) calculates value of ODE driver function at the boundary points when shock-wave is not (is) present.
6. Calc\_dFdt.m calculates the first time derivative of flow fluxes at all grid-points.
7. Calc\_dJdt.m calculates the first time derivative of the flow sources at all interior grid-points.
8. Calc\_dffdt.m calculates the first time derivative of the ODE driver function at all interior grid-points.
9. CalcdfdtBvalsmSW.m (CalcdfdtBvalspSW.m) evaluates the first time derivative of ODE driver function at the spatial boundary points when a shock-wave is not (is) present.
10. Calc\_d2Fdt2.m calculates second time derivative of the flow fluxes at all grid-points.
11. Calc\_d2Jdt2.m calculates the second time derivative of the flow sources at all interior grid-points.
12. Calc\_d2ffdt2.m calculates second time derivative of ODE driver function at all interior grid-points.
13. NextInCond.m calls:
14. CalcBCmSW.m (CalcBCpSW.m) uses flow boundary conditions to determine new initial condition for boundary grid-points when a shock-wave is not (is) present.

The functions in IntegrateGij.m call the following functions:

1. FuncOrc.m calls:
2. fOrc.m which evaluates ODE driver function at all knot times in a sub-subinterval at each interior grid-points.
3. QAmpEst.m calls:
4. randQAEA.m generates a random deviate with probability distribution appropriate to Quantum Amplitude Estimation Algorithm presented in Brassard et al. quant-ph/0005055.

The function fOrc.m calls:

1. CalcBCmSW.m which was previously described.
2. CalcBCpSW.m which was previously described.
3. Calcf0.m which evaluates the ODE driver function at all interior grid-points.

I’ve also written a number of MATLAB scripts that plot the simulation results as graphs:

1. PlottingScript\_MassDen.m plots the steady-state mass density vs. position along nozzle.
2. PlottingScript\_Temperature.m plots the steady-state temperature vs. position along nozzle.
3. PlottingScript\_MachNum.m plots the steady-state Mach number vs. position along nozzle.

The user should rename the files in these scripts to conform with her/his computer.