"A Hartree-Fock Approach to the Energy of a 2D Quantum Dot with N electrons." README

Supervisor: Prof. Grüneis, Author: Manuel Ulreich

The simulation can either be run with input from the console (the program asks for input after starting it), or directly with precompiled parameters. The readme is intended as a general overview, as the code has a lot of comments and is more or less self-explanatory, especially in context of the written project thesis.

The codebase essentially consists of four classes in the "PA" namespace:

1. Hamiltonian
2. Solver
3. BicubicInterpolator
4. SolverInput

# Hamiltonian:

Contains functions for setting up the Hamiltonian, symmetrically or asymmetrically, as well as functions for printing the Hamiltonian to the console (or some other stream). It also contains functions for applying the Hamiltonian to a vector, as well as a few debug functions. Debug functions include methods for generating and printing reference harmonics.

# Solver:

Contains the majority of the code involved in the simulation. It has all the functions for diagonalizing the Hamiltonian, both symmetrically and asymmetrically as well as functions for normalizing/orthogonalizing the eigenfunctions. The diagonalization methods rely on LAPACK functions, meaning it is possible to use any LAPACK implementation. The Intel MKL was used in the course of the project thesis. Additionally it can save the eigenfunctions to a file and load them from there as well. This is possible both in ASCII as well as binary format. It also contains the code for the Hartree-Fock component of the simulation, namely the coulomb-correction integrals.

# BicubicInterpolator:

This contains the code for bicubic interpolation of a grid, which is necessary for the optional interpolation step in the previously mentioned integrals.

# SolverInput:

This is mainly a wrapper for the input necessary for the solver, as well as reading input from the console and performing sanity checks on the input.

# Additional Information:

* The simulation also has a “Constants” file with some preprocessed constants, as well as a macro for choosing the precision (single versus double precision). The LAPACK functions are also implemented on basis of these macros, so the right variants are chosen. The project thesis data was simulated with double precision.
* The code was developed within the Microsoft Visual Studio IDE, thus there may be some headers left over which need to be removed. I also have had the experience, that Visual Studio adds some header data to the code files, which can hinder compilation with a different compiler than the Microsoft Visual C++ compiler, however copy and pasting the code into a text editor takes care of this, if necessary. Considering that there are eight relevant files, this is easily carried out.
* The memory and time requirements of the simulation naturally scale with grid size, and no artificial limit was hard coded to prohibit too high requirements. The only checks performed are those which are required for the LAPACK functions to run in general (sign, desired number of eigenvectors, etc.).
* Some code fragments have been left in the codebase in form of comments. For example, the coulomb integral is multithreaded with use of asynchronous tasks (part of the c++ STL).
* C++17 is the minimum requirement.
* Samples of hard-coding simulations are provided in the code.