SHPC4001 Assignment 5

Vincent Bochat
21295666

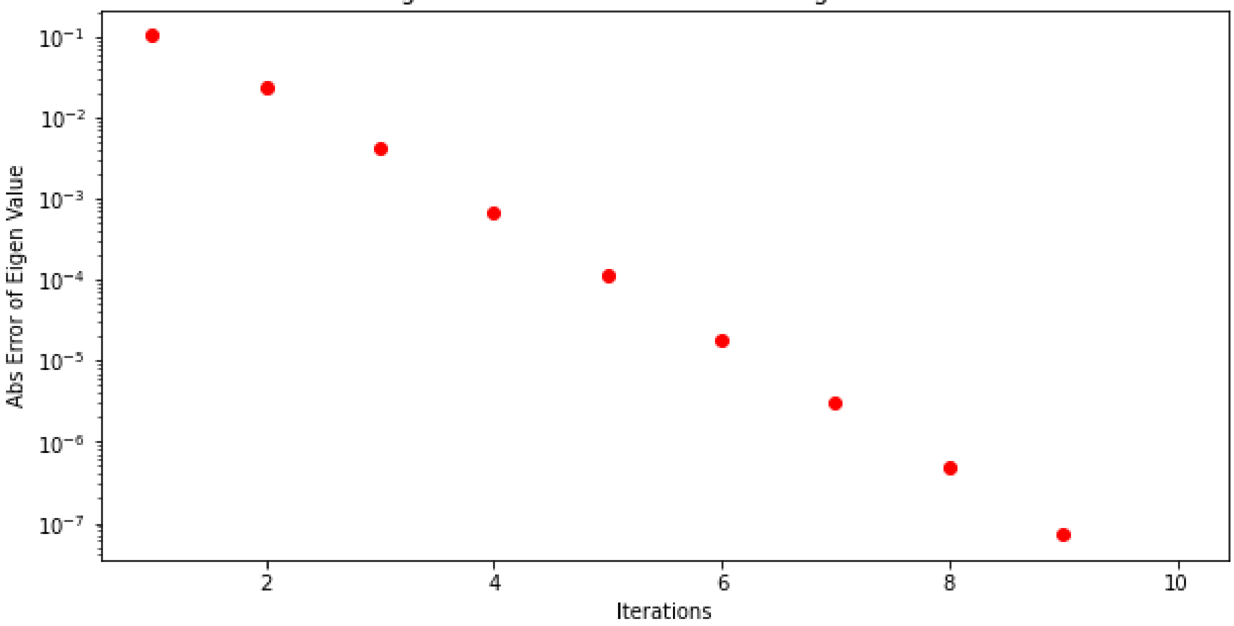
Part 1: Implementing the Inverse Power Iteration Method

---

a) Code *"Ass5_Q1_01.f90"* which uses module code *"MatInvert.f90"* calculates the dominant Eigen value and Eigenvector of the specified 3x3matrix by first inverting the matrix and then calculating the smallest Eigenvalue and Eigenvector.

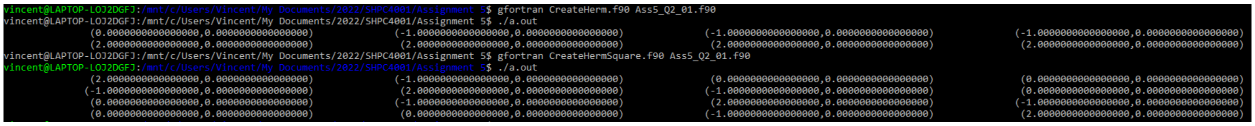
b) The relative absolute error of the calculated eigen values is plotted below. The absolute error decays exponentially, as shown in Figure 1.01.

Figure 1.01: Error of Calculateed Eigen Values



Part 2: Using LAPACK

a) Module Code *"CreateHermSqaure.f90"* creates matrix L explicitly, with L having dimensions N x N as specified. Module Code *"CreateHerm.f90"* creates a k x N matrix in the band structure data storage form for LAPACK. 'k' = 2 in this case since there is only 1 super diagonal in the matrix L. Examples of L for N = 4 are shown below.



```
vincent@LAPTOP-LOJ2DGFJ:/mnt/c/Users/Vincent/My Documents/2022/SHPC4001/Assignment 5$ gfortran CreateHerm.f90 Ass5_Q2_01.f90
vincent@LAPTOP-LOJ2DGFJ:/mnt/c/Users/Vincent/My Documents/2022/SHPC4001/Assignment 5$ ./a.out
               (0.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)
               (2.0000000000000000,0.0000000000000000)              (2.0000000000000000,0.0000000000000000)              (2.0000000000000000,0.0000000000000000)              (2.0000000000000000,0.0000000000000000)
vincent@LAPTOP-LOJ2DGFJ:/mnt/c/Users/Vincent/My Documents/2022/SHPC4001/Assignment 5$ gfortran CreateHermSquare.f90 Ass5_Q2_01.f90
vincent@LAPTOP-LOJ2DGFJ:/mnt/c/Users/Vincent/My Documents/2022/SHPC4001/Assignment 5$ ./a.out
               (2.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)              (0.0000000000000000,0.0000000000000000)              (0.0000000000000000,0.0000000000000000)
              (-1.0000000000000000,0.0000000000000000)              (2.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)              (0.0000000000000000,0.0000000000000000)
               (0.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)              (2.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)
               (0.0000000000000000,0.0000000000000000)              (0.0000000000000000,0.0000000000000000)             (-1.0000000000000000,0.0000000000000000)              (2.0000000000000000,0.0000000000000000)
```

b) BLAS stand for Basic Linear Algebra Subprograms. They were introduced in the 1970s as a form of portable high performance numerical software (Zee & Geijn, 2015). The three levels of BLAS are:

   i. Level 1 BLAS: Targeted to vector-vector operations. The original subprograms of the 1970s fit into this category, supported by the vector supercomputers of the era;
   ii. Level 2 BLAS: Subprograms capable of Matrix – Vector operations. These began in the 1980s due to the arrival of cache-based architectures; and
   iii. Level 3 BLAS: Matrix – Matrix subprograms. High performance is achieved by amortizing *$O(n^2)$* memory operations over *$O(n^3)$* floating point operations (Zee & Geijn, 2015).

c) Because of the efficiencies of BLAS, it is used in the development of LAPACK (NetLib, 2021). In fact, LAPACK routines are written in such a way that BLAS calls perform most of the computation. The advantage of this is LAPACK achieving high performance with portable code (Blackford, 1999); the framework offered by BLAS allows LAPACK to perform finer tuned tasks, aiding its overall capability.

Part 3: The 2-D QHO

a) The finite difference matrix approxiamtion to the Hamiltonian is constructed in the code *“Ass5_Q3_01”* using a Do Loop using the main diagonal as a reference point for the allocation of the v_{ij}, η and μ values. This was possible because the matrix is Hermitian with Banded structure. With N = 51 due to the chosen discretisation, N – 2 = 49, giving our H a dimension of (2401, 2401). For the discretised potential matrix of dimension N x N, the boundary conditions V(1, :), V(:, 1), V(51, :) & V(51, :) are discounted, which results in i, j = 1, … , 49 for the v_{ij} values.

b) Fortran code *“Ass5_Q3_02”* construct the Hamiltonian in a) as an upper triangular matrix and uses LAPACK to calculate the eigen vectors and values of the QHO.

   i. Figure 3.01 & 3.02 show the calculated Eigenvalues for the first 10 energy levels of the QHO and compares them to the theoretical energies of the levels. The relative absolute error of the computed value increases approximately linearly over the first 10 energy levels.

Figure 3.01: EigenValues and Energy Levels

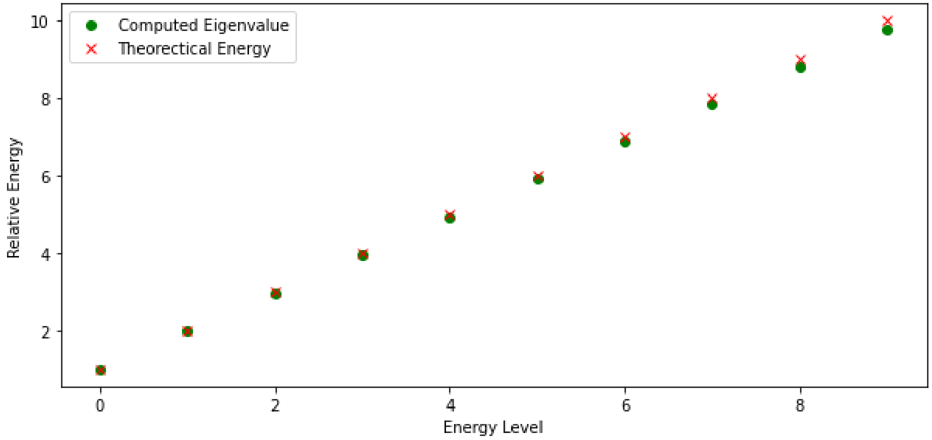
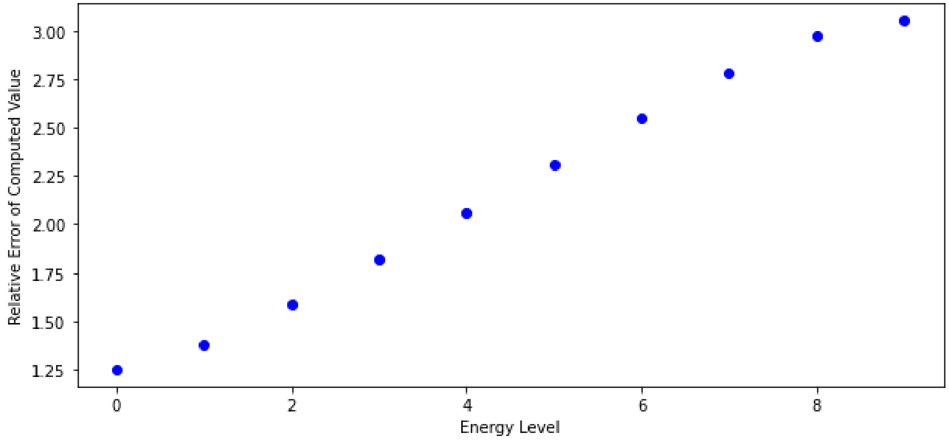
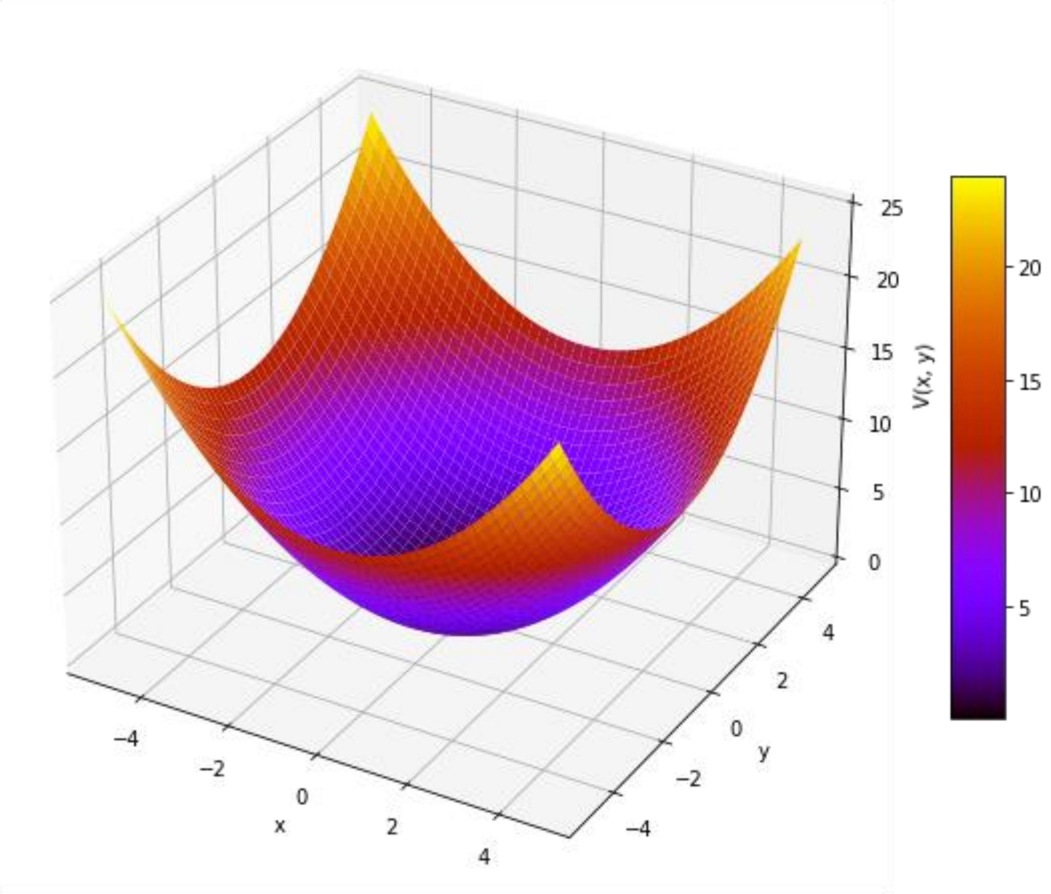


Figure 3.02: Absolute Error of Computed Value



ii. The potential is plotted in Figure 3.03 below.

Figure 3.03: Potential Funtion of QHO



iii. The Computed groundstate of the QHO is plotted in Figure 3.04.

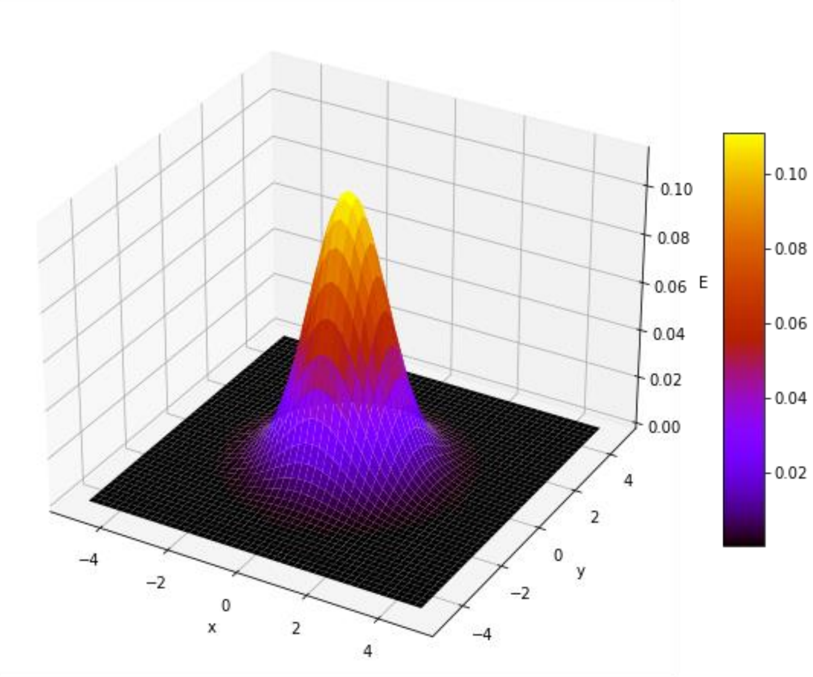


Figure 3.04: Computed Groundstate of QHO

iv. The analytical solution to the ground state of the QHO is plotted in Figure 3.05 below. Figure 3.06 shows the absolute error of the calculated groundstate in (iii) compared to this analytical solution. The maximum magnitude of the error is 0.05.

Figure 3.05: Analytical Groundstate of QHO

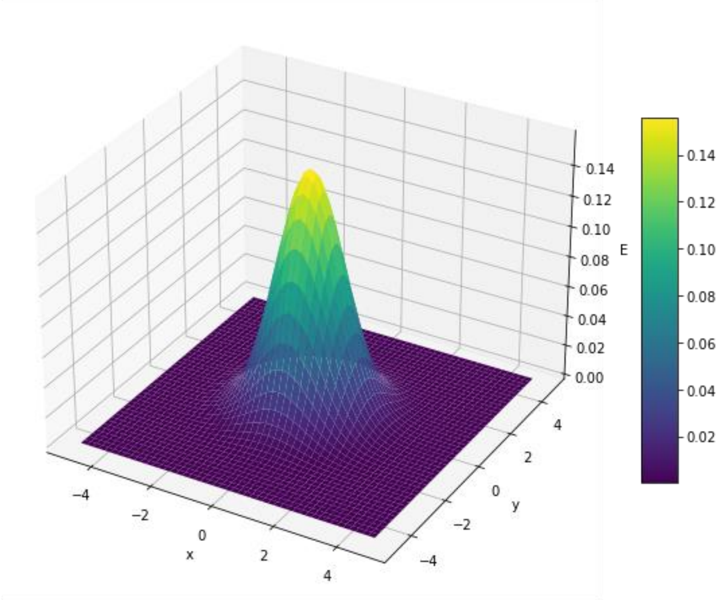
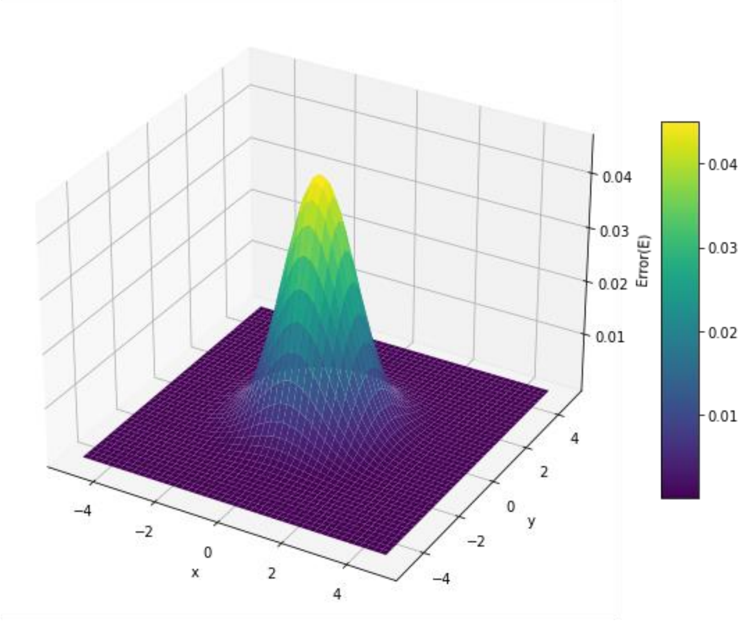


Figure 3.06: Absolute Error of Computee QHO Groundstate



References

Blackford, S. (1999, October 1). *NetLib*. Retrieved from LAPACK and the BLAS: https://www.netlib.org/lapack/lug/node11.html

Ling, S. J. (2020). *38.6 Motion in a Magnetic Field*. Retrieved March 22, 2022, from Physics Bootcamp: http://www.physicsbootcamp.org/Motion-in-Magnetic-Field.html?fbclid=IwAR2FGtZ-FRDgQAYlyex0_jCVxMwTgpMRiyKA6jhiML1LCdONb56YsZ6l690

NetLib. (2021, June 29). Retrieved from BLAS (Basic Linear Algebra Subprograms): http://www.netlib.org/blas/

Zee, F. G., & Geijn, R. A. (2015). BLIS: A Framework for Rapidly Instanting BLAS Functionality. *ACM Trans. Math. Softw., 41*(3). doi:http://dx.doi.org/10.1145/2764454