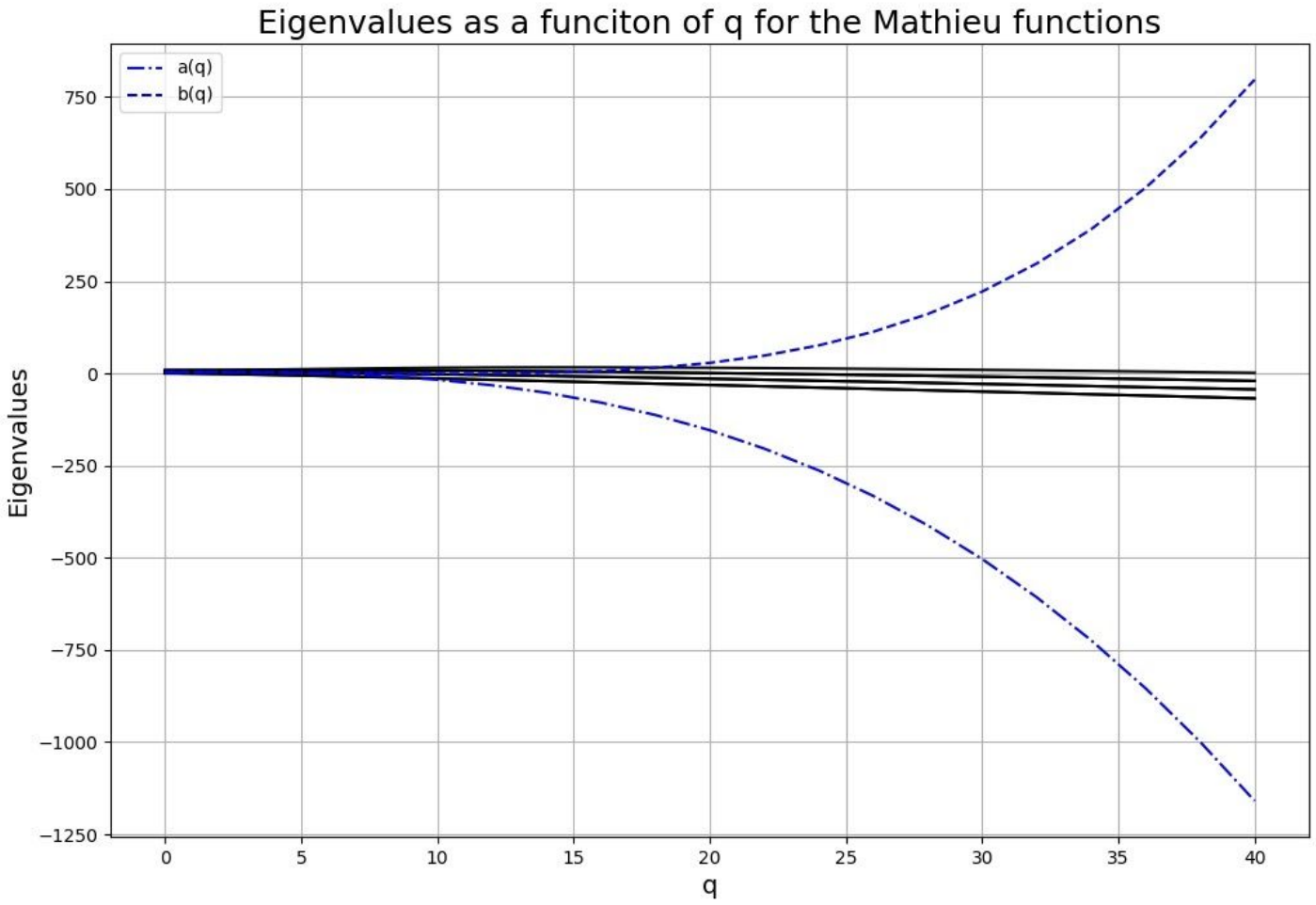
AM 129 Homework 5 Report Deliverables

Owen Morehead

Nov 22 2020



The overall purpose of this code was to write a script in Python which casts and solves the **Mathieu functions** as an eigenvalue problem and generates the eigenvalue plot.

The mathieu functions are defined as the solutions of the differential equation:

$$\frac{d^2y}{dy^2} + (a - 2qcos(2x))y(x) = 0 \tag{1}$$

We look for only the solutions that are 2π periodic. We choose *q* freely and *a* will be fixed to a discrete set of values. **Equation 1** can be re-written as an eigenvalue problem as follows:

$$(2qcos(2x) - \frac{d^2}{dx^2})y = ay \tag{2}$$
$$y(0) = y(2\pi)$$

We can see that *a* and *y* form an eigenvalue/eigenfunction pair for the operator on the left. To deal with the differential operator, we write out a discrete version of the above equation as:

$$[V + T^{-1}D_kT]y = ay \tag{3}$$

x is an array of equispaced grid points on $(0, 2\pi)$, T is the discrete fourier transform matrix for this grid, V is a diagonal matrix with entries $(cos(2x_1), cos(2x_2), cos(2x_3)$,..., and D_k is a diagonal matrix with the squares of the wavenumbers as entries $(0, 1, 1, 4, 4, 9, 9, ...)$. Everything inside the square brackets in **Equation 3** will be packed into one matrix, H, and we will use LAPACK to obtain the eigenvalue decomposition.

The code that solves this problem is separated into several files, courtesy of Ian. *mathieu.f90* calls all the other modules and defines the main program and writes the solutions. *Diffop.f90* defines the *diffop* module which builds the needed matrices. *Problemsetup.f90* holds specific parameters like x, and q, has the routines to read the input files, and contains the code which labels the output files. As per usual there is also a *utility.f90* file which defines useful constants, and other helper files that make things easier such as the *makefile* which defines the compiler flags allowing for the compilation through the program *make*. There are a couple other helper files which I don't feel the need to discuss in detail here, as they have already been discussed in previous lecture notes. The program that we wrote, mathieuRun.f90, is used to interact with the operating system, write a backup *mathieu.init* file, call the *mathieu* program, and print the desired plot of the eigenvalues as a function of the parameter q. We call the *mathieu* program on a range of *q* values to get the eigenvalues and then generate the plot. We also plot two equations, *a(q),* and *b(q)* for a range of *q* values on top of the eigenvalue plot.

Parsweep_mathieu and *plot_parsweep* both rely on hard-coded ranges for the parameter *q*. To change this behavior to something more safe and useful, we could have defined in *run_mathieu* that we want to run the program for however many values of q are inputted, instead of just one value like we have coded. Then the *parsweep_mathieu* and *plot_parsweep* functions can utilize the q range which was **defined once** in *run_mathieu*, whether it be a single value or a range of values, and so we won't have to define the range of q in either of these functions (*parsweep_mathieu*, *plot_parsweep*) as it will already be defined. This is a useful thing to do as it will lower our chances of generating errors from the definitions of q

we currently have setup. If we have to define the variable twice then there is a change we might not define both to be the exact same range or value and this can lead to errors.

To import this file into an interactive python session to explore the results generated by the Fortran program more easily, we can call *from mathieuRun import* * to import and then use the functions made in our program *mathieuRun.py* to explore and plot the results of the Fortran program as long as the *mathieu* directory is accessible.