Project 2, deadline September 30

Computational Physics I FYS3150/FYS4150

Department of Physics, University of Oslo, Norway

Sep 9, 2020

Eigenvalue problems, from the equations of a buckling beam to Schroedinger's equation for two electrons in a three-dimensional harmonic oscillator well

Introduction. The aim of this project is to develop your own code for solving eigenvalue problems. The matrix to diagonalize is the same as the one we encountered in project one, a so-called tridiagonal Toeplitz matrix. This matrix has analytical eigenpairs (eigenvalues and eigenvectors) and gives us an excellent testing ground for our algorithms. In this project we will develop an eigenvalue solver based on Jacobi's method. The project will also introduce you to units tests and we will compare our results against other eigenvalue solvers (from LAPACK and/or numpy).

This project aims also at introducing to you the concept of scaling of equations. This means often either making various variables dimensionless or introducing units which are more convenient.

We will start with the two-point boundary value problem of a buckling beam or a spring fastened at both ends. This is one of the problems which has analytical solutions. Thereafter, by simply adding a new variable along the diagonal elements, we can study quantum mechanical problems. In particular, we will study the harmonic oscillator problem in three dimensions, with one or two electrons. For the latter case we can study the role of the repulsive Coulomb interaction and extract interesting physics results. For selected frequencies, this interacting two-electron problem exhibits analytical solutions, one of the few cases of an interacting system where wecan find analytical solutions. See M. Taut, Phys. Rev. A 48, 3561 (1993) for the derivation of analytical expressions for the eigenpairs..

Electrons confined in small areas in semiconductors, so-called quantum dots, form a hot research area in modern solid-state physics, with applications spanning from such diverse fields as quantum nano-medicine to the contruction of quantum gates. The article on quantum computing with quantum dots by Loss and DiVincenzo is an excellent read for those interested in this exciting topic.

© 1999-2020, "Computational Physics I FYS3150/FYS4150":"http://www.uio.no/studier/emner/matnat/fys/FYS3150/index-eng.html". Released under CC Attribution-NonCommercial 4.0 license

The buckling beam problem, a classical wave function problem in one dimension. We start with the following differential equation, namely

$$\gamma\frac{d^2u(x)}{dx^2} = -Fu(x),$$

where $u(x)$ is the vertical displacement of the beam in the y direction. The beam has length L, $x \in [0, L]$ and F is a force applied at $(L, 0)$ in the direction towards the origin. The parameter γ is a constant defined by properties like the rigidity of the beam. We apply again so-called Dirichlet boundary conditions and set $u(0) = u(L) = 0$.

In this specific case two of the parameters γ, F and L are known. As an example, assume we know F and L. Then the eigenvalue problem we set up below will allow us to find γ.

We define a dimensional variable

$$\rho = \frac{x}{L},$$

meaning that we have $\rho \in [0, 1]$. By reordering the equation as

$$\frac{d^2u(\rho)}{d\rho^2} = -\frac{FL^2}{\gamma}u(\rho) = -\lambda u(\rho),$$

with $\lambda = FL^2/\gamma$ we have an equation that when discretized, becomes an eigenvalue problem. We use the same expression for the second derivative of a function u as we did in project 1, namely

$$u'' = \frac{u(\rho + h) - 2u(\rho) + u(\rho - h)}{h^2} + O(h^2), \tag{1}$$

where h is our step. Next we define minimum and maximum values for the variable ρ, $\rho_{\min} = 0$ and $\rho_{\max} = 1$, respectively. With a given number of mesh points, N, we define the step length h as, with $\rho_{\min} = \rho_0$ and $\rho_{\max} = \rho_N$,

$$h = \frac{\rho_N - \rho_0}{N}.$$

The value of ρ at a point i is then

$$\rho_i = \rho_0 + ih \qquad i = 1, 2, \dots, N.$$

We can rewrite the differential equation for a value ρ_i as

$$-\frac{u(\rho_i + h) - 2u(\rho_i) + u(\rho_i - h)}{h^2} = \lambda u(\rho_i),$$

or in a more compact way as

$$-\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = \lambda u_i.$$

Following our approach from project 1, we can rewrite this equation in a more a general form, but now as an eigenvalue problem, that is

$$\begin{bmatrix} d & a & 0 & 0 & \dots & 0 & 0 \\ a & d & a & 0 & \dots & 0 & 0 \\ 0 & a & d & a & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & a & d & a \\ 0 & \dots & \dots & \dots & \dots & a & d \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \dots \\ u_{N-2} \\ u_{N-1} \end{bmatrix} = \lambda \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \dots \\ u_{N-2} \\ u_{N-1} \end{bmatrix}. \tag{2}$$

As in project 1, we have not included the endpoints u_0 and u_N. We have defined $d = 2/h^2$ and the non-diagonal ones as $a = -1/h^2$. This eigenvalue problem has analytical eigenpairs, with eigenvalues given as

$$\lambda_j = d + 2a\cos\left(\frac{j\pi}{N+1}\right), \quad j = 1, 2, \dots N.$$

The eigenvectors are

$$u_j = \sin\left(\frac{j\pi}{N+1}\right), \quad j = 1, 2, \dots N.$$

The lecture notes of Tom Lyche provide an excellent description of this matrix. Chapters 1.2 and 1.3 of this text present a thourough discussion of tridiagonal matrices and two-point boundary value problems.

Project 2 a): Mathematical intermezzo. A unitary transformation preserves the orthogonality of the obtained eigenvectors. To see this consider first a basis of vectors $\mathbf{v}_i$,

$$\mathbf{v}_i = \begin{bmatrix} v_{i1} \\ \dots \\ \dots \\ v_{in} \end{bmatrix}$$

We assume that the basis is orthogonal, that is

$$\mathbf{v}_j^T \mathbf{v}_i = \delta_{ij}.$$

Show that an orthogonal or unitary transformation

$$\mathbf{w}_i = \mathbf{U}\mathbf{v}_i,$$

preserves the dot product and orthogonality.

Project 2 b): Setting up a code for tridiagonal Toeplitz matrix. Your task now is to write a function which implements Jacobi's rotation algorithm (see Lecture notes chapter 7) in order to solve Eq. (2). However, the first simple check is to set up the matrix to diagonalize for a given N and use either armadillo's or numpy's functions for diagonalizing a matrix. You can then check that you obtain the analytical eigenvalues.

For Jacobi's method, we define the quantities $\tan\theta = t = s/c$, with $s = \sin\theta$ and $c = \cos\theta$ and

$$\cot 2\theta = \tau = \frac{a_{ll} - a_{kk}}{2a_{kl}}.$$

We can then define the angle θ so that the non-diagonal matrix elements of the transformed matrix a_{kl} become non-zero and we obtain the quadratic equation (using $\cot 2\theta = 1/2(\cot\theta - \tan\theta)$

$$t^2 + 2\tau t - 1 = 0,$$

resulting in

$$t = -\tau \pm \sqrt{1+\tau^2},$$

and c and s are easily obtained via

$$c = \frac{1}{\sqrt{1+t^2}},$$

and $s = tc$.

How many similarity transformations are needed before you reach a result where all non-diagonal matrix elements are essentially zero? Try to estimate the number of transformations and extract a behavior as function of the dimensionality of the matrix. Compare your results with the analytical ones.

Plot also the eigenvector for the lowest eigenvalue and compare this eigenvector with the analytical solution.

You can check your results against the Armadillo function for solving eigenvalue problems. The armadillo function eigsym can be used to find eigenvalues and eigenvectors. A Python program which solves this part of the project is available under the project writing slides.

Comment your results (here you could for example compute the time needed for both algorithms for a given dimensionality of the matrix).

Project 2 c): Implementing tests in your code. In this project (and later ones as well) we will implement so-called **unit** tests. Our unit tests are mainly meant to test mathematical properties of our algorithm. During the development phase of a program it is useful to devise tests that your program should pass. One of these is to make sure that for a given simple test matrix (say a 5×5 matrix) our algorithm for searching for the largest non-diagonal element always returns the correct answer. Furthermore, for a known simple matrix, irrespective of changes made, we should always get the same and correct eigenvalues. Another test is to make sure that the orthogonality shown in exercise (a) is preserved. Try to figure out other tests your code should pass, based either on the mathematical properties of the algorithms or more program specific tests. Implement at least two unit tests in this project.

Extending our machinery to quantum mechanics. Here we will assume that these electrons move in a three-dimensional harmonic oscillator potential (they are confined by for example quadrupole fields) and repel each other via the static Coulomb interaction. We assume spherical symmetry. You don't need to think of quantum mechanics when solving this project, look at it as another diagonalization problem.

We are first interested in the solution of the radial part of Schroedinger's equation for one electron. This equation reads

$$-\frac{\hbar^2}{2m}\left(\frac{1}{r^2}\frac{d}{dr}r^2\frac{d}{dr}-\frac{l(l+1)}{r^2}\right)R(r)+V(r)R(r)=ER(r).$$

In our case $V(r)$ is the harmonic oscillator potential $(1/2)kr^2$ with $k=m\omega^2$ and E is the energy of the harmonic oscillator in three dimensions. The oscillator frequency is ω and the energies are

$$E_{nl}=\hbar\omega\left(2n+l+\frac{3}{2}\right),$$

with $n=0,1,2,\dots$ and $l=0,1,2,\dots$.

Since we have made a transformation to spherical coordinates it means that $r\in[0,\infty)$. The quantum number l is the orbital momentum of the electron. Then we substitute $R(r)=(1/r)u(r)$ and obtain

$$-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}u(r)+\left(V(r)+\frac{l(l+1)}{r^2}\frac{\hbar^2}{2m}\right)u(r)=Eu(r).$$

The boundary conditions are $u(0)=0$ and $u(\infty)=0$.

We introduce a dimensionless variable $\rho=(1/\alpha)r$ where α is a constant with dimension length and get

$$-\frac{\hbar^2}{2m\alpha^2}\frac{d^2}{d\rho^2}u(\rho)+\left(V(\rho)+\frac{l(l+1)}{\rho^2}\frac{\hbar^2}{2m\alpha^2}\right)u(\rho)=Eu(\rho).$$

We will set in this project $l=0$. Inserting $V(\rho)=(1/2)k\alpha^2\rho^2$ we end up with

$$-\frac{\hbar^2}{2m\alpha^2}\frac{d^2}{d\rho^2}u(\rho)+\frac{k}{2}\alpha^2\rho^2u(\rho)=Eu(\rho).$$

We multiply thereafter with $2m\alpha^2/\hbar^2$ on both sides and obtain

$$-\frac{d^2}{d\rho^2}u(\rho)+\frac{mk}{\hbar^2}\alpha^4\rho^2u(\rho)=\frac{2m\alpha^2}{\hbar^2}Eu(\rho).$$

The constant α can now be fixed so that

$$\frac{mk}{\hbar^2}\alpha^4=1,$$

or

$$\alpha = \left(\frac{\hbar^2}{mk}\right)^{1/4}.$$

Defining

$$\lambda = \frac{2m\alpha^2}{\hbar^2}E,$$

we can rewrite Schroedinger's equation as

$$-\frac{d^2}{d\rho^2}u(\rho) + \rho^2 u(\rho) = \lambda u(\rho).$$

This is the first equation to solve numerically. In three dimensions the eigenvalues for $l = 0$ are $\lambda_0 = 3, \lambda_1 = 7, \lambda_2 = 11, \dots$.

We define minimum and maximum values for the variable ρ, $\rho_{\min} = 0$ and $\rho_{\max}$, respectively. You need to check your results for the energies against different values $\rho_{\max}$, since we cannot set $\rho_{\max} = \infty$.

With a given number of mesh points, N, we define the step length h as, with $\rho_{\min} = \rho_0$ and $\rho_{\max} = \rho_N$,

$$h = \frac{\rho_N - \rho_0}{N}.$$

The value of ρ at a point i is then

$$\rho_i = \rho_0 + ih \qquad i = 1, 2, \dots, N.$$

We can rewrite the Schroedinger equation for a value ρ_i as

$$-\frac{u(\rho_i + h) - 2u(\rho_i) + u(\rho_i - h)}{h^2} + \rho_i^2 u(\rho_i) = \lambda u(\rho_i),$$

or in a more compact way

$$-\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + \rho_i^2 u_i = -\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + V_i u_i = \lambda u_i,$$

where $V_i = \rho_i^2$ is the harmonic oscillator potential.

We define first the diagonal matrix element

$$d_i = \frac{2}{h^2} + V_i,$$

and the non-diagonal matrix element

$$e_i = -\frac{1}{h^2}.$$

In this case the non-diagonal matrix elements are given by a mere constant. *All non-diagonal matrix elements are equal.* With these definitions the Schroedinger equation takes the following form

$$d_i u_i + e_i u_{i-1} + e_i u_{i+1} = \lambda u_i,$$

where u_i is unknown. We can write the latter equation as a matrix eigenvalue problem

$$\begin{bmatrix} d_1 & e_1 & 0 & 0 & \dots & 0 & 0 \\ e_1 & d_2 & e_2 & 0 & \dots & 0 & 0 \\ 0 & e_2 & d_3 & e_3 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots e_{N-3} & d_{N-2} & e_{N-2} \\ 0 & \dots & \dots & \dots & \dots & e_{N-2} & d_{N-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{N-1} \end{bmatrix} = \lambda \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{N-1} \end{bmatrix}. \quad (3)$$

Note here that we have not included the endpoints since we assume that they are knowm (as we did in project 1). The matrix is also symmetric in this case.

Project 2 d): Quantum dots in three dimensions, one electron. Add the harmonic oscillator potential to your tridiagonal Toeplitz matrix from 2a-2c and diagonalize the matrix. Study the results as functions of the number of integration points N and your approximation to $\rho_{\max}$. The analytical results with our scaling for the one-electron energies are $\lambda = 3, 7, 11, 15, \dots$. How many integration points do you need in order to reproduce the analytical results with say four leading digits after the decimal point?

You can reuse the code you wrote for part (b), but you need to add the potential ρ^2 to the diagonal elements.

Project 2 e): Quantum dots in three dimensions, two electrons. Note: if you are not interested in the explicit quantum physics case here, you can replace this part with either 2g or 2h below. The latter ones are optional exercises if you do this.

We will now study two electrons in a harmonic oscillator well which also interact via a repulsive Coulomb interaction. Let us start with the single-electron equation written as

$$-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}u(r) + \frac{1}{2}kr^2 u(r) = E^{(1)}u(r),$$

where $E^{(1)}$ stands for the energy with one electron only. For two electrons with no repulsive Coulomb interaction, we have the following Schroedinger equation

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dr_1^2} - \frac{\hbar^2}{2m}\frac{d^2}{dr_2^2} + \frac{1}{2}kr_1^2 + \frac{1}{2}kr_2^2\right)u(r_1, r_2) = E^{(2)}u(r_1, r_2).$$

Note that we deal with a two-electron wave function $u(r_1, r_2)$ and two-electron energy $E^{(2)}$.

With no interaction this can be written out as the product of two single-electron wave functions, that is we have a solution on closed form.

We introduce the relative coordinate $\mathbf{r} = \mathbf{r}_1 - \mathbf{r}_2$ and the center-of-mass coordinate $\mathbf{R} = 1/2(\mathbf{r}_1 + \mathbf{r}_2)$. With these new coordinates, the radial Schroedinger equation reads

$$\left(-\frac{\hbar^2}{m}\frac{d^2}{dr^2} - \frac{\hbar^2}{4m}\frac{d^2}{dR^2} + \frac{1}{4}kr^2 + kR^2\right)u(r,R) = E^{(2)}u(r,R).$$

The equations for r and R can be separated via the ansatz for the wave function $u(r,R) = \psi(r)\phi(R)$ and the energy is given by the sum of the relative energy E_r and the center-of-mass energy E_R, that is

$$E^{(2)} = E_r + E_R.$$

We add then the repulsive Coulomb interaction between two electrons, namely a term

$$V(r_1, r_2) = \frac{\beta e^2}{|\mathbf{r}_1 - \mathbf{r}_2|} = \frac{\beta e^2}{r},$$

with $\beta e^2 = 1.44$ eVnm.

Adding this term, the r-dependent Schroedinger equation becomes

$$\left(-\frac{\hbar^2}{m}\frac{d^2}{dr^2} + \frac{1}{4}kr^2 + \frac{\beta e^2}{r}\right)\psi(r) = E_r\psi(r).$$

This equation is similar to the one we had previously in (b) and we introduce again a dimensionless variable $\rho = r/\alpha$. Repeating the same steps as above, we arrive at

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \frac{1}{4}\frac{mk}{\hbar^2}\alpha^4\rho^2\psi(\rho) + \frac{m\alpha\beta e^2}{\rho\hbar^2}\psi(\rho) = \frac{m\alpha^2}{\hbar^2}E_r\psi(\rho).$$

We want to manipulate this equation further to make it as similar to that in (a) as possible. We define a new 'frequency'

$$\omega_r^2 = \frac{1}{4}\frac{mk}{\hbar^2}\alpha^4,$$

and fix the constant α by requiring

$$\frac{m\alpha\beta e^2}{\hbar^2} = 1$$

or

$$\alpha = \frac{\hbar^2}{m\beta e^2}.$$

Defining

$$\lambda = \frac{m\alpha^2}{\hbar^2}E,$$

we can rewrite Schroedinger's equation as

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \omega_r^2\rho^2\psi(\rho) + \frac{1}{\rho} = \lambda\psi(\rho).$$

We treat ω_r as a parameter which reflects the strength of the oscillator potential.

Here we will study the cases $\omega_r = 0.01$, $\omega_r = 0.5$, $\omega_r = 1$, and $\omega_r = 5$ for the ground state only, that is the lowest-lying state.

With no repulsive Coulomb interaction you should get a result which corresponds to the relative energy of a non-interacting system. Make sure your results are stable as functions of $\rho_{\max}$ and the number of steps.

We are only interested in the ground state with $l = 0$. We omit the center-of-mass energy.

You can reuse the code you wrote for part (b), but you need to add the potential $\omega_r^2\rho^2 + 1/\rho$ to the diagonal elements.

Comment the results for the lowest state (ground state) as function of varying strengths of ω_r.

For specific oscillator frequencies, the above equation has answers in an analytical form, see the article by M. Taut, Phys. Rev. A 48, 3561 (1993).

Project 2 f): First optional exercise: Quantum physics analysis of the results. This exercise is a continuation of the previous and adds more quantum physics to the analysis. In this exercise we want to plot the wave function for two electrons as functions of the relative coordinate r and different values of ω_r. With no Coulomb interaction you should have a result which corresponds to the non-interacting case. Plot either the function itself or the probability distribution (the function squared) with and without the repulsion between the two electrons. Varying ω_r, the shape of the wave function will change.

We are only interested in the wave function for the ground state with $l = 0$ and omit again the center-of-mass motion.

The eigenvectors are normalized. Plot then the normalized wave functions for different values of ω_r and comment the results.

Project 2 g): Second optional challenge: Smarter way of finding the eigenvalues. This exercise is also optional and is an algorithmic challenge. Your matrix is already tridiagonal. Can you devise a more efficient way to find the eigenvalues and eigenvectors instead of the brute force application of Jacobi's method? Hint: Use the bisection method discussed in for example Barth, Martin and Wilkinson, Numerische Mathematik 9, 386 (1967).

Project 2 h): Third optional challenge: Implementing Lanczos' algorithm. This exercise is optional and is meant more as a challenge for those of you with an interest in other algorithms for solving eigenvalue problems. Implement the iterative Lanczos' algorithm discussed in the lecture notes and compute the lowest eigenvalues as done in exercise (c) or (d) above. Compare your results and discuss faults and merits of the iterative method versus direct methods like Jacobi's method.

Introduction to numerical projects

Here follows a brief recipe and recommendation on how to write a report for each project.

- Give a short description of the nature of the problem and the eventual numerical methods you have used.
- Describe the algorithm you have used and/or developed. Here you may find it convenient to use pseudocoding. In many cases you can describe the algorithm in the program itself.
- Include the source code of your program. Comment your program properly.
- If possible, try to find analytic solutions, or known limits in order to test your program when developing the code.
- Include your results either in figure form or in a table. Remember to label your results. All tables and figures should have relevant captions and labels on the axes.
- Try to evaluate the reliabilty and numerical stability/precision of your results. If possible, include a qualitative and/or quantitative discussion of the numerical stability, eventual loss of precision etc.
- Try to give an interpretation of you results in your answers to the problems.
- Critique: if possible include your comments and reflections about the exercise, whether you felt you learnt something, ideas for improvements and other thoughts you've made when solving the exercise. We wish to keep this course at the interactive level and your comments can help us improve it.
- Try to establish a practice where you log your work at the computerlab. You may find such a logbook very handy at later stages in your work, especially when you don't properly remember what a previous test version of your program did. Here you could also record the time spent on solving the exercise, various algorithms you may have tested or other topics which you feel worthy of mentioning.

Format for electronic delivery of report and programs

The preferred format for the report is a PDF file. You can also use DOC or postscript formats or as an ipython notebook file. As programming language we prefer that you choose between C/C++, Fortran2008 or Python. The following prescription should be followed when preparing the report:

- Use Devilry to hand in your projects, log in at `http://devilry.ifi.uio.no` with your normal UiO username and password and choose either 'fys3150' or 'fys4150'. There you can load up the files within the deadline.
- Upload **only** the report file! For the source code file(s) you have developed please provide us with your link to your github domain. The report file should include all of your discussions and a list of the codes you have developed. Do not include library files which are available at the course homepage, unless you have made specific changes to them.
- In your git repository, please include a folder which contains selected results. These can be in the form of output from your code for a selected set of runs and input parameters.
- In this and all later projects, you should include tests (for example unit tests) of your code(s).
- Comments from us on your projects, approval or not, corrections to be made etc can be found under your Devilry domain and are only visible to you and the teachers of the course.

Finally, we encourage you to work two and two together. Optimal working groups consist of 2-3 students. You can then hand in a common report.