Electric and Magnetic Effects on the Energy Spectrum of Hydrogen

PHY-312, Numerical Methods and Programming
Course Instructor: Dr. Sunil Pratap Singh

Anirudh Kalla, 18037
Ayyappan Shyam, 18067
Suthanth Srinivasan, 18283
Vaibhav Singh, 18303
Vatsalya Sharan, 18307
Department of Physics
IISER Bhopal

Preface

The choice of this research project originally stemmed from our passion for utilizing numerical analysis techniques to understand quantitative behaviour of physical phenomenon using programming algorithms. Modeling the electric and magnetic effects on the hydrogen atom came as a natural choice, since it provides all the basic quantitative nuances one would expect from theoretical descriptions and yet does not exceed complexity in a way that renders numerical analysis unviable. As computational and programming resources increase manifold, utilizing numerical algorithms to tackle real life physics problems becomes increasingly lucrative due to higher precision and accuracy. We strive to not only use, but to create powerful tools that push the boundaries of computational physics using intelligent inputs and utilizing intuition wherever possible, giving us a definite edge over raw brute-force driven approaches.

The completion of this assignment gives us much pleasure. We would like to show our gratitude to Dr. Sunil Pratap Singh, Course Instructor, IISER Bhopal for giving us an opportunity to work on such an enriching assignment with good guidelines throughout the course of this semester.

In addition, a thank you to Dr. Ambuj Pandey, who guided us at crucial junctures of this project, and whose prompt help had a lasting effect.

We also thank the Indian Institute for Science Education and Research, Bhopal, for providing a conducive environment to us, greatly facilitating our project.

We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in writing this assignment.

The Authors

Contents

1 Introduction **3**

2 Introduction To D.P.T. **3**

3 Stark Effect **3**
- 3.1 Selection Rules . . . 3
- 3.2 Matrix Element Calculation . . . 4
- 3.3 Results . . . 7

4 Zeeman Effect **9**
- 4.1 Kets In Total Angular Momentum Basis . . . 9
  - 4.1.1 States With l=0 . . . 9
  - 4.1.2 States With l=1 . . . 10
- 4.2 Fine-Structure Formula . . . 10
  - 4.2.1 For $j = \frac{1}{2}$ . . . 11
  - 4.2.2 For $j = \frac{3}{2}$ . . . 11
- 4.3 Magnetic Field Formula . . . 11
- 4.4 Final Hamiltonian $H^{'}$ . . . 12
  - 4.4.1 First Four Eigenvalues . . . 12
  - 4.4.2 Next Two Eigenvalues . . . 12
  - 4.4.3 Last Two Eigenvalues . . . 13
- 4.5 Final Energies And States . . . 13
  - 4.5.1 Energies . . . 13
  - 4.5.2 States . . . 14
- 4.6 Results . . . 14
- 4.7 Code Implementation . . . 15
  - 4.7.1 Characteristic Equation . . . 15
  - 4.7.2 QR Decomposition . . . 16

5 Appendix **17**
- 5.1 Proof Of The Selection Rules In Stark Effect . . . 17
- 5.2 Romberg Integration . . . 18
- 5.3 Leibniz Formula . . . 18
- 5.4 Durand-Kerner Algorithm . . . 19
- 5.5 QR Decomposition . . . 19

6 Bibliography **21**
- 6.1 Stark Effect . . . 21
- 6.2 Zeeman Effect . . . 21

1 Introduction

Hydrogen Atom is one of the most basic systems of Quantum Mechanics and is studied widely for understanding the principles of Quantum Mechanics. We here attempt to analyse electric and magnetic effects, namely the Stark and Intermediate-field Zeeman effects on the energy spectrum of the Hydrogen atom using the various Numerical tools at our disposal. We shall try and compute some of the shifts in the energy spectrum of the atom under the influence of these effects.

2 Introduction To D.P.T.

Let ψ_i for $i = 1, 2, 3....., n$ be the eigenstates of H_0 corresponding to energy E. Let $H = H_0 + H'$. If we treat H' as a perturbation and want to see what happens to the energy E as a result of this perturbation, then we use degenerate perturbation theory (D.P.T.). We shall use first order theory in this analysis.
Define the $n \times n$ matrix P as $P_{ij} = \langle \psi_i | H' | \psi_j \rangle$. We expect the perturbation to alter the degeneracy of the energy E. The eigenvalues of the matrix P give us shifts in energy due to the perturbation and diagonalizing the matrix gives us a basis consisting of linear combinations of the degenerate eigenstates of the unperturbed Hamiltonian which is a 'good' basis in the sense that we can apply non-degenerate perturbation theory to these states directly and get the same results.

3 Stark Effect

When a Hydrogen atom is placed in an external electric field of constant magnitude E, pointing along the z-axis, we have an additional term to deal with in the Hamiltonian.

$$H = \frac{-\hbar^2}{2m}\nabla^2 - \frac{e^2}{r} - Eez \tag{1}$$

The third term which is due to the applied electric field, polarizes the hydrogen atom and causes a change in its energy spectrum. This partly breaks the n^2-fold degeneracy of the hydrogen states $\psi_{nlm} = \langle \mathbf{r} | nlm \rangle$ $(n > 1)$ with principal quantum number n because radial symmetry is lost. The figure provided below shows how the perturbation changes the energy for different polar angles for $l = 2$. We try and find this change for the $n = 3$ level using degenerate perturbation theory (D.P.T.).

3.1 Selection Rules

Two selection rules greatly reduce our computation. Firstly

$$\langle nlm | - Eez | n'l'm' \rangle = 0 \tag{2}$$

if $m \neq m'$.
And

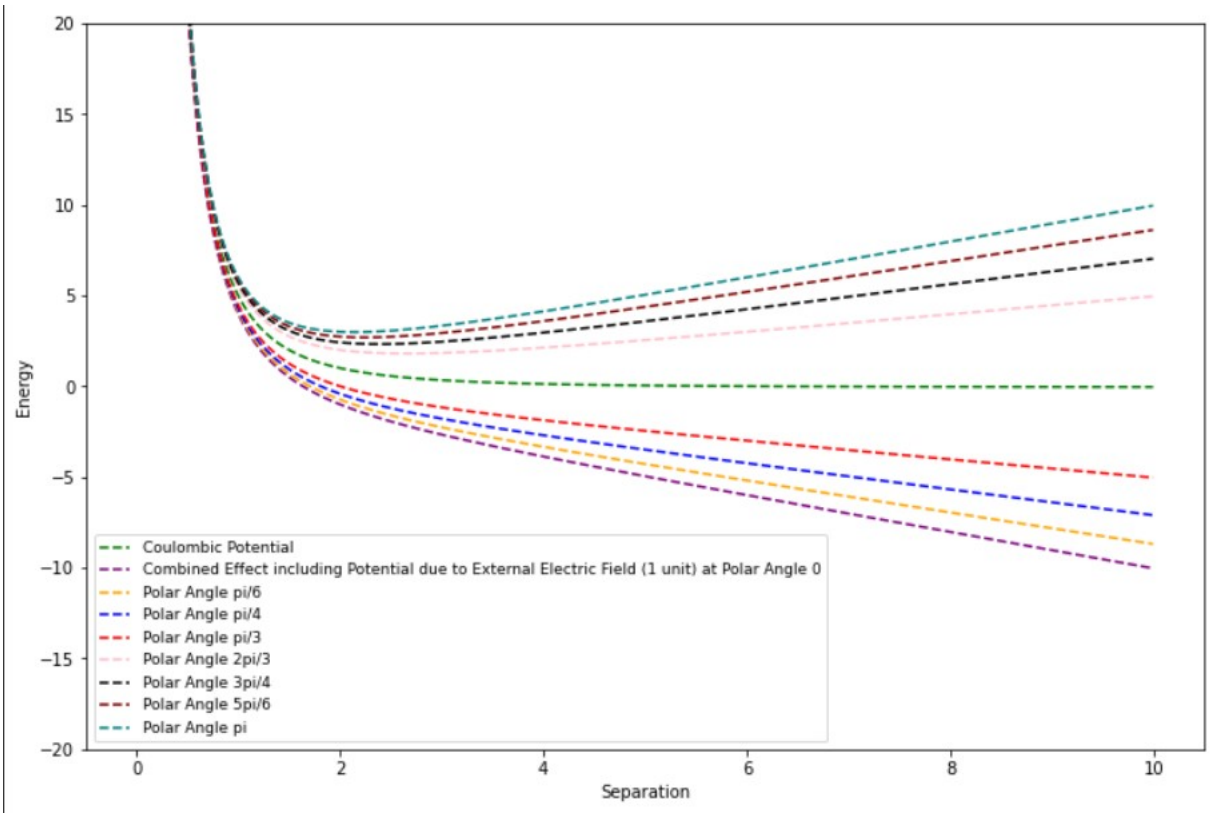

Figure 1: Representation of Stark Effect using Python code

$$\langle nlm| - Eez|n'l'm'\rangle = 0 \tag{3}$$

if $|l - l'| \neq 1$.
The proofs of the above two identities are given in the appendix.
For $n = 3$ we have nine-fold degeneracy. So we have to deal with a 9×9 matrix which seems like an arduous task. However the above rules greatly reduce our work as most of the elements vanish (only 8 matrix elements are non-zero). The non-zero matrix elements are listed as follows.

- $\langle 3, 0, 0|H'|3, 1, 0\rangle = \langle 3, 1, 0|H'|3, 0, 0\rangle$
- $\langle 3, 1, 0|H'|3, 2, 0\rangle = \langle 3, 2, 0|H'|3, 1, 0\rangle$
- $\langle 3, 1, -1|H'|3, 2, -1\rangle = \langle 3, 2, -1|H'|3, 1, -1\rangle = \langle 3, 1, 1|H'|3, 2, 1\rangle = \langle 3, 2, 1|H'|3, 1, 1\rangle$

3.2 Matrix Element Calculation

We calculated the matrix elements using three methods. First, we separated the integral into radial, azimuthal and polar parts and we used the three Newton-Cotes formulas learnt in the course to evaluate the radial integral. We implemented trapezoidal as well as Simpson's 1/3 and 3/8 algorithms. To handle the infinite upper limit of the integral, we integrated to 1000, 10000 and 100000 with a step size of 0.1 and found that the numerical integral for these

three upper limits was the same upto the maximum precision, indicating that the process converged. We rounded off the result to 3 decimal places and found that all three methods gave the same estimate.
To calculate the radial integral we also used Romberg's method (see appendix) with a provision for error tolerance. We again integrated to 1000, 10000 and 100000 and found that after rounding off to the first three decimal places all three limits gave the same result as the Newton-Cotes methods.
Then we used Python's *sympy.physics.hydrogen* library which has the wavefunctions built-in to check the closeness of our results to the actual value.
The code implementation for the Newton-Cotes integration algorithms and Romberg's method are provided below.

```
    def trapezoidal(f,a,b,n):

      h = (b-a)/n
      x = []
      y = []
      for i in range(n+1): y.append(f(a+h*i)),x.append(a+h*i)
      integral = (y[0]+y[n])/2
      for i in range(1,n):
        integral += y[i]
      return h*integral
```

```
    def simpson13(f,a,b,n):

      h =( b-a)/n
      y = []
      x = []
      for i in range(n+1): y.append(f(a+h*i)),x.append(a+h*i)
      integral = y[0]+y[n]
      for i in range(1,n):
        integral += (2+2*(i%2))*y[i]
      return (h/3)*integral
```

```
    def simpson38(f,a,b,n):

      h = (b-a)/n
      y = []
      x = []
      for i in range(n+1): y.append(f(a+h*i)),x.append(a+h*i)
      integral = 0
      for i in range(0,n,3):
        integral += y[i]+3*y[i+1]+3*y[i+2]+y[i+3];
      return (3*h/8)*integral
```

```
    def romberg(f, a, b, p, t):

        I = np.zeros((p, p))
        I_t = np.zeros(p)
        for i in range(0, p):
            I[i, 0] = trapezoidal(f, a, b, 2**(i+5))
        for k in range(1,p):
            for j in range(0,p-k):
```

```
                if(j+1!=p):
                    I[j,k] = (4**(k)*I[j+1,k-1]-I[j,k-1])/(4**(k)-1)
                else:
                    break
        ea = np.zeros(p)
        for z in range(1,p):
            ea = abs((I[0,z]-I[1,z-1])/I[0,z])*100
            I_t[z] = I[0, z]
            if(ea<t):
                break
        print(z, I_t[z])
```

For the numerical computation, we used the Hartree atomic units to non-dimensionalize the integral and also to make the results easier to interpret. This also helps in making the energy values not very small, which tend to induce substantial numerical errors. We took the electric field to be 1 unit in this system.
For the energy values in other units, we just have to multiply the values obtained by the factor eEa_0.
The Hartree atomic units are as follows.

- $\hbar = 1$, also known as the unit of action.
- $e = 1$, also known as the unit of charge.
- $a_0 = 1$, also known as the atomic unit of length.
- $m_e = 1$, also known as the atomic unit of mass.

Now the matrix elements we are calculating are elements of a 9×9 matrix and we are following the given convention for the ordering of the the elements. Suppose the vectors ψ_1,ψ_2,....,ψ_9 constitute an ordered basis for the degenerate subspace, then the convention is as follows.

$$\psi_{300} = |3,0,0\rangle = \psi_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Similarly, we have $|3,1,-1\rangle = \psi_2$, $|3,1,0\rangle = \psi_3$, $|3,1,1\rangle = \psi_4$, $|3,2,-2\rangle = \psi_5$, $|3,2,-1\rangle = \psi_6$, $|3,2,0\rangle = \psi_7$, $|3,2,1\rangle = \psi_8$ and $|3,2,2\rangle = \psi_9$.

$$P = \begin{pmatrix} 0 & 0 & \langle 3,0,0|H'|3,1,0\rangle & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \langle 3,1,-1|H'|3,2,-1\rangle & 0 & 0 & 0 \\ \langle 3,0,0|H'|3,1,0\rangle & 0 & 0 & 0 & 0 & 0 & \langle 3,1,0|H'|3,2,0\rangle & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \langle 3,1,1|H'|3,2,1\rangle & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \langle 3,1,-1|H'|3,2,-1\rangle & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \langle 3,1,0|H'|3,2,0\rangle & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \langle 3,1,1|H'|3,2,1\rangle & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

3.3 Results

Using both, numerical integration programs and the predefined functions from *sympy.physics.hydrogen*, we found the perturbation matrix (taking $E = 1$). We eliminated rows and columns numbered 5 and 9 from the matrix because they only had zeroes and we know that the spectrum of the matrix would simply be the spectrum of the new smaller matrix and a couple of zero eigenvalues. We used the Leibniz formula for determinant (see appendix), which was implemented using the library *permutation*, and *sympy* to get the characteristic polynomial of the new 7×7 matrix. One root was clearly zero hence we concluded that the zero eigenvalue has a degeneracy of 3.
We used the Durand-Kerner method (appendix) for approximation of roots to find the non-zero eigenvalues of the perturbation matrix. We also used *sympy diagonalize* to obtain the eigenvalues to confirm that the values we obtained are correct. These turned out to be 0 (3-fold), 9, -9, 4.5 (2-fold) and -4.5 (2-fold).
We also found the eigenvectors by using symbolic manipulation to get the eigenvalue equation and comparing the entries of the column vectors on the left and right hand sides. We also used *sympy eigenvects* to confirm our solution. The shifts in energy and corresponding normalised eigenstates were found to be as follows.

- $\frac{1}{\sqrt{3}}\psi_1 - \frac{1}{\sqrt{2}}\psi_3 + \frac{1}{\sqrt{6}}\psi_7$ is shifted by -9 units of energy.
- $\frac{1}{\sqrt{3}}\psi_1 + \frac{1}{\sqrt{2}}\psi_3 + \frac{1}{\sqrt{6}}\psi_7$ is shifted by 9 units of energy.
- $\frac{1}{\sqrt{2}}(\psi_2 - \psi_6)$ and $\frac{1}{\sqrt{2}}(\psi_4 - \psi_8)$ are shifted by -4.5 units of energy.
- $\frac{1}{\sqrt{2}}(\psi_2 + \psi_6)$ and $\frac{1}{\sqrt{2}}(\psi_4 + \psi_8)$ are shifted by 4.5 units of energy.
- ψ_5, $\frac{1}{\sqrt{3}}\psi_1 - \sqrt{\frac{2}{3}}\psi_7$ and ψ_9 are not shifted.

The above states represent the 'good basis' which diagonalizes the perturbation matrix.
Eigenvalue calculation using the Durand-Kerner method is shown below.
The Matrix used for finding eigenvectors and the code snippet for getting characteristic equations is also provided below.
Matrix and characteristic equation.

```
l = Symbol('lambda')
r, s, t, u, v, w, x, y, z = symbols("r s t u v w x y z")

init_printing()
M = Matrix([[-l, 0, matrix1, 0, 0, 0, 0],
            [0, -l, 0, 0, matrix3, 0, 0],
            [matrix1, 0, -l, 0, 0, matrix2, 0],
            [0, 0, 0, -l, 0, 0, matrix3],
            [0, matrix3, 0, 0, -l, 0, 0],
            [0, 0, matrix2, 0, 0, -l, 0],
            [0, 0, 0, matrix3, 0, 0, -l]])
sum = 0
for p in Permutation.group(7):
```

```
  product = 1
  for i in range(7):
    product = product*M[i, p(i+1)-1]
  if p.is_odd:
    product = product*(-1)
  sum = sum+product

print("Characteristic polynomial is:", sum)
print('\n')
```

Durand-Kerner is shown below.

```
r1 = 1
r2 = 2
r3 = 3
r4 = -1
r5 = -2
r6 = -3

d1 = (l-r2)*(l-r3)*(l-r4)*(l-r5)*(l-r6)
d2 = (l-r1)*(l-r3)*(l-r4)*(l-r5)*(l-r6)
d3 = (l-r2)*(l-r1)*(l-r4)*(l-r5)*(l-r6)
d4 = (l-r2)*(l-r3)*(l-r1)*(l-r5)*(l-r6)
d5 = (l-r2)*(l-r3)*(l-r4)*(l-r1)*(l-r6)
d6 = (l-r2)*(l-r3)*(l-r4)*(l-r5)*(l-r1)

terminate = np.zeros(6, dtype=int)

for i in range(30):
  d1 = (l-r2)*(l-r3)*(l-r4)*(l-r5)*(l-r6)
  if abs((sum.subs(l, r1)/r1*d1.subs(l, r1)))<0.00001:
    terminate[0] = 1
  r1 = r1-(sum.subs(l, r1)/d1.subs(l, r1))
  d2 = (l-r1)*(l-r3)*(l-r4)*(l-r5)*(l-r6)
  if abs((sum.subs(l, r2)/r2*d2.subs(l, r2)))<0.00001:
    terminate[1] = 1
  r2 = r2-(sum.subs(l, r2)/d2.subs(l, r2))
  d3 = (l-r2)*(l-r1)*(l-r4)*(l-r5)*(l-r6)
  if abs((sum.subs(l, r3)/r3*d3.subs(l, r3)))<0.00001:
    terminate[2] = 1
  r3 = r3-(sum.subs(l, r3)/d3.subs(l, r3))
  d4 = (l-r2)*(l-r3)*(l-r1)*(l-r5)*(l-r6)
  if abs((sum.subs(l, r4)/r4*d4.subs(l, r4)))<0.00001:
    terminate[3] = 1
  r4 = r4-(sum.subs(l, r4)/d4.subs(l, r4))
  d5 = (l-r2)*(l-r3)*(l-r4)*(l-r1)*(l-r6)
  if abs((sum.subs(l, r5)/r5*d5.subs(l, r5)))<0.00001:
    terminate[4] = 1
  r5 = r5-(sum.subs(l, r5)/d5.subs(l, r5))
  d6 = (l-r2)*(l-r3)*(l-r4)*(l-r5)*(l-r1)
  if abs((sum.subs(l, r6)/r6*d6.subs(l, r6)))<0.00001:
    terminate[5] = 1
  r6 = r6-(sum.subs(l, r6)/d6.subs(l, r6))
  if i%5 == 0:
    print(r1)
```

```
    print(r2)
    print(r3)
    print(r4)
    print(r5)
    print(r6)
    print('\n')
  is_all_one = np.all((terminate == 1))
  if is_all_one:
    print(r1)
    print(r2)
    print(r3)
    print(r4)
    print(r5)
    print(r6)
    print('\n')
    break
```

4 Zeeman Effect

When an atom is placed in a uniform external magnetic field B_{ext}, the energy levels are shifted. This phenomenon is known as the Zeeman effect. The nature of this splitting depends critically on the strength of external magnetic field in comparison to internal field that gives rise to spin-orbit coupling. If $B_{ext} << B_{int}$, then fine structure dominates and the Hamiltonian due to the external field can be treated as perturbation. However if $B_{ext} >> B_{int}$, then the Zeeman effect dominates and the fine structure becomes the perturbation.
In this section, we will look at the peculiar case of Intermediate-field Zeeman (IfZ) effect which is characterized by the non-dominance of both Hamiltonians, $H_z^{'}$ and $H_{fs}^{'}$ which denote the perturbations due to the external magnetic field and the fine structure of Hydrogen respectively.

$$H' = H_z^{'} + H_{fs}^{'}$$

We will consider the case where $n = 2$ as a part of this study.

4.1 Kets In Total Angular Momentum Basis

We shall express $|jm_j\rangle$ as a linear combination of $|lm_l\rangle\,|sm_s\rangle$ states. While l, m_l, m_s make the matrix elements of $H_z^{'}$ slightly easier to compute, they make those of $H_{fs}^{'}$ far more difficult, thereby increasing the overall complexity of computation. We will, however, get the same eigenvalues in both cases as they are clearly basis-independent. Now for this change of basis, we will use the Clebsch-Gordan coefficients as follows.

4.1.1 States With l=0

$$\psi_1 = \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle = |0\ \ 0\rangle \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle \tag{4}$$

$$\psi_2 = \left|\frac{1}{2}\ \ -\frac{1}{2}\right\rangle = |0\ \ 0\rangle \left|\frac{1}{2}\ \ -\frac{1}{2}\right\rangle \tag{5}$$

4.1.2 States With l=1

$$\psi_3 = \left|\frac{3}{2}\ \ \frac{3}{2}\right\rangle = |1\ \ 1\rangle \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle \tag{6}$$

$$\psi_4 = \left|\frac{3}{2}\ \ -\frac{3}{2}\right\rangle = |1\ \ -1\rangle \left|\frac{1}{2}\ \ -\frac{1}{2}\right\rangle \tag{7}$$

$$\psi_5 = \left|\frac{3}{2}\ \ \frac{1}{2}\right\rangle = \sqrt{\frac{2}{3}}\,|1\ \ 0\rangle \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle + \sqrt{\frac{1}{3}}\,|1\ \ 1\rangle \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle \tag{8}$$

$$\psi_6 = \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle = -\sqrt{\frac{1}{3}}\,|1\ \ 0\rangle \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle + \sqrt{\frac{2}{3}}\,|1\ \ 1\rangle \left|\frac{1}{2}\ \ -\frac{1}{2}\right\rangle \tag{9}$$

$$\psi_7 = \left|\frac{3}{2}\ \ -\frac{1}{2}\right\rangle = \sqrt{\frac{1}{3}}\,|1\ \ -1\rangle \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle + \sqrt{\frac{2}{3}}\,|1\ \ 0\rangle \left|\frac{1}{2}\ \ -\frac{1}{2}\right\rangle \tag{10}$$

$$\psi_8 = \left|\frac{1}{2}\ \ -\frac{1}{2}\right\rangle = -\sqrt{\frac{2}{3}}\,|1\ \ -1\rangle \left|\frac{1}{2}\ \ \frac{1}{2}\right\rangle + \sqrt{\frac{1}{3}}\,|1\ \ 0\rangle \left|\frac{1}{2}\ \ -\frac{1}{2}\right\rangle \tag{11}$$

4.2 Fine-Structure Formula

We have

$$E'_{fs} = \frac{E_2^2}{2mc^2}(3 - \frac{8}{(j+\frac{1}{2})}) \tag{12}$$

$$\implies E'_{fs} = \frac{E_1^2}{32mc^2}(3 - \frac{8}{(j+\frac{1}{2})}) \tag{13}$$

where

$$\frac{E_1}{mc^2} = -\frac{\alpha^2}{2} \tag{14}$$

$$\alpha = \frac{e^2}{4\pi\epsilon_0\hbar c} \approx \frac{1}{137} \tag{15}$$

α is termed the fine-structure constant or Sommerfeld's constant. Now, we can write E'_{fs} as

$$\left(\frac{-E_1\alpha^2}{64}\right)\left(3 - \frac{8}{j+\frac{1}{2}}\right) = -\left(\frac{13.6\alpha^2}{64}\right)\left(3 - \frac{8}{j+\frac{1}{2}}\right) \tag{16}$$

$$\implies \gamma\left(3 - \frac{8}{j+\frac{1}{2}}\right) \tag{17}$$

4.2.1 For $j = \frac{1}{2}$

$$H_{fs}^{'} = \gamma(3 - 8) = -5\gamma \tag{18}$$

4.2.2 For $j = \frac{3}{2}$

$$H_{fs}^{'} = \gamma(3 - \frac{8}{2}) = -\gamma \tag{19}$$

4.3 Magnetic Field Formula

$$H_{z}^{'} = \frac{e}{2m} B_{ext} \left(L_z + 2S_z\right) \tag{20}$$

$\psi_1, \psi_2, \psi_3, \psi_4$ are eigenstates of L_z and S_z. This implies that there are only diagonal elements for these states.

$$\langle H_{z}^{'} \rangle = \frac{eh}{2m} B_{ext}(m_l + 2m_s) = \beta(m_l + 2m_s) \tag{21}$$

We can now calculate the elements of this particular matrix as

$$\langle H_{z}^{'} \rangle_{11} = \beta \tag{22}$$

$$\langle H_{z}^{'} \rangle_{22} = -\beta \tag{23}$$

$$\langle H_{z}^{'} \rangle_{33} = 2\beta \tag{24}$$

$$\langle H_{z}^{'} \rangle_{44} = -2\beta \tag{25}$$

As for the remaining elements (which are not the eigenstates of L_z and S_z), we have

$$\left(L_z + 2S_z\right) |\psi_5\rangle = \sqrt{\frac{2}{3}} \hbar \, |1 \;\; 0\rangle \left| \frac{1}{2} \;\; \frac{1}{2} \right\rangle \tag{26}$$

$$\left(L_z + 2S_z\right) |\psi_6\rangle = -\sqrt{\frac{1}{3}} \hbar \, |1 \;\; 0\rangle \left| \frac{1}{2} \;\; \frac{1}{2} \right\rangle \tag{27}$$

$$\left(L_z + 2S_z\right) |\psi_7\rangle = -\sqrt{\frac{2}{3}} \hbar \, |1 \;\; 0\rangle \left| \frac{1}{2} \;\; -\frac{1}{2} \right\rangle \tag{28}$$

$$\left(L_z + 2S_z\right) |\psi_8\rangle = -\sqrt{\frac{1}{3}} \hbar \, |1 \;\; 0\rangle \left| \frac{1}{2} \;\; -\frac{1}{2} \right\rangle \tag{29}$$

Now, using $26-29$, the rest of the elements of the matrix may be written down as

$$\langle H_z^{'}\rangle_{55} = \frac{2}{3}\beta \tag{30}$$

$$\langle H_z^{'}\rangle_{56} = -\frac{\sqrt{2}}{3}\beta \tag{31}$$

$$\langle H_z^{'}\rangle_{65} = -\frac{\sqrt{2}}{3}\beta \tag{32}$$

$$\langle H_z^{'}\rangle_{66} = \frac{1}{3}\beta \;\; \langle H_z^{'}\rangle_{77} = -\frac{2}{3}\beta \tag{33}$$

$$\langle H_z^{'}\rangle_{78} = -\frac{\sqrt{2}}{3}\beta \tag{34}$$

$$\langle H_z^{'}\rangle_{87} = -\frac{\sqrt{2}}{3}\beta \tag{35}$$

$$\langle H_z^{'}\rangle_{88} = -\frac{1}{3}\beta \tag{36}$$

4.4 Final Hamiltonian $H^{'}$

Finally, the complete matrix of $-H^{'}$ can be written as

$$-\begin{pmatrix} 5\gamma-\beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5\gamma+\beta & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \gamma-2\beta & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \gamma+2\beta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma-\frac{2}{3}\beta & \frac{\sqrt{2}}{3}\beta & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\sqrt{2}}{3}\beta & 5\gamma-\frac{1}{3}\beta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \gamma+\frac{2}{3}\beta & \frac{\sqrt{2}}{3}\beta \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\sqrt{2}}{3}\beta & 5\gamma+\frac{1}{3}\beta \end{pmatrix}$$

We would now be required to find the eigenvalues of this matrix. For this, we will use the following approach.

4.4.1 First Four Eigenvalues

The first four eigenvalues are listed directly as they are simply the first four diagonal elements. Also, we may recall that the first four states, namely $\psi_1, \psi_2, \psi_3, \psi_4$ are eigenstates of L_Z and S_Z, giving us the eigenvalues without any hassle.

4.4.2 Next Two Eigenvalues

The next two eigenvalues can be obtained by solving the characteristic equation of the third diagonal block. This can be done as follows.

$$\lambda^2 - \lambda(6\gamma-\beta) + \left(5\gamma^2 - \frac{11}{3}\gamma\beta\right) = 0 \tag{37}$$

We can get the eigenvalues by solving 37 for λ.

$$\lambda_{\pm}^{5/6} = -3\gamma + \frac{\beta}{2} \pm \sqrt{4\gamma^2 + \frac{2}{3}\gamma\beta + \frac{\beta^2}{4}} \tag{38}$$

4.4.3 Last Two Eigenvalues

Similar to the last section, we can solve the (very similar) characteristic equation for the fourth diagonal block to obtain the final two eigenvalues. The eigenvalues are shown directly for brevity.

$$\lambda_{\pm}^{7/8} = -3\gamma - \frac{\beta}{2} \pm \sqrt{4\gamma^2 - \frac{2}{3}\gamma\beta + \frac{\beta^2}{4}} \tag{39}$$

4.5 Final Energies And States

4.5.1 Energies

After this extensive exercise, we have obtained the final 8 shifts in energy for Intermediate-field Zeeman effect (for $n = 2$). They are listed in the following equation block.

$$\begin{aligned}
E1 &= -5\gamma + \beta \\
E2 &= 5\gamma - \beta \\
E3 &= -\gamma + 2\beta \\
E4 &= -\gamma - 2\beta \\
E5 &= -3\gamma + \beta/2 + \sqrt{4\gamma^2 + (2/3)\gamma\beta + \beta^2/4} \\
E6 &= -3\gamma + \beta/2 - \sqrt{4\gamma^2 + (2/3)\gamma\beta + \beta^2/4} \\
E7 &= -3\gamma - \beta/2 + \sqrt{4\gamma^2 - (2/3)\gamma\beta + \beta^2/4} \\
E8 &= -3\gamma - \beta/2 - \sqrt{4\gamma^2 - (2/3)\gamma\beta + \beta^2/4}
\end{aligned} \tag{40}$$

4.5.2 States

As for the normalized states, we have

$$\begin{aligned}
&\psi_1 \\
&\psi_2 \\
&\psi_3 \\
&\psi_4 \\
&\alpha_1 \left(\psi_5 + \frac{3}{\sqrt{2}\beta} \left\{ 2\gamma - \frac{\beta}{6} - \delta_1 \right\} \psi_8 \right) \\
&\alpha_2 \left(\psi_5 + \frac{3}{\sqrt{2}\beta} \left\{ 2\gamma - \frac{\beta}{6} + \delta_1 \right\} \psi_8 \right) \\
&\alpha_3 \left(\psi_4 + \frac{3}{\sqrt{2}\beta} \left\{ 2\gamma + \frac{\beta}{6} - \delta_2 \right\} \psi_7 \right) \\
&\alpha_4 \left(\psi_4 + \frac{3}{\sqrt{2}\beta} \left\{ 2\gamma + \frac{\beta}{6} + \delta_2 \right\} \psi_7 \right)
\end{aligned} \tag{41}$$

Here, the pre-factors $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are

$$\begin{aligned}
\alpha_1 &= \left\{ 1 + \tfrac{9}{2\beta^2} \left(2\gamma - \tfrac{\beta}{6} - \delta_1 \right)^2 \right\}^{-\frac{1}{2}}, \\
\alpha_2 &= \left\{ 1 + \tfrac{9}{2\beta^2} \left(2\gamma - \tfrac{\beta}{6} + \delta_1 \right)^2 \right\}^{-\frac{1}{2}}, \\
\alpha_3 &= \left\{ 1 + \tfrac{9}{2\beta^2} \left(2\gamma + \tfrac{\beta}{6} - \delta_2 \right)^2 \right\}^{-\frac{1}{2}}, \\
\alpha_4 &= \left\{ 1 + \tfrac{9}{2\beta^2} \left(2\gamma + \tfrac{\beta}{6} + \delta_2 \right)^2 \right\}^{-\frac{1}{2}}
\end{aligned} \tag{42}$$

4.6 Results

The final shifts in energy obtained by solving the characteristic equation can be listed as the following for a given external field of, say, 10 T.

$$\begin{aligned}
E1 &= 0.00063544748269339230000 \\
E2 &= -0.000522228877546607700 \\
E3 &= 0.00116899822075467850000 \\
E4 &= 0.00116899822075467850000 \\
E5 &= 4.0706177487566744 * 10^{-05} \\
E6 &= 0.00060606316572050420000 \\
E7 &= -0.000553186886810157600 \\
E8 &= 4.22798697782283300 * 10^{-05}
\end{aligned} \tag{43}$$

The final shifts in energy obtained from QR algorithm can be listed as the following for

a given external field of, say, 10 T.

$$\begin{aligned} E1 &= 0.00063544748269339230000 \\ E2 &= -0.000522228877546607700 \\ E3 &= 0.00116899822075467850000 \\ E4 &= 0.00116899822075467850000 \\ E5 &= 4.0706177487566744 * 10^{-05} \\ E6 &= 0.00060606316572050400000 \\ E7 &= -0.000553186886810157600 \\ E8 &= 4.22798697782283400 * 10^{-05} \end{aligned} \tag{44}$$

Thus the two algorithms give us the same result upto a remarkable precision.

4.7 Code Implementation

We start off by defining our perturbation matrix as a novel function called ifz_matrix. This function captures the matrix representation of the perturbation and returns it when called. It takes two values, gamma and beta which are passed to generate the final form of the matrix once the user inputs the strength of the magnetic field.
We then ask the user to input the strength of the external magnetic field (which affects the value of β) and determine the shifted energies using two methods.

4.7.1 Characteristic Equation

First, we solve the quadratic characteristic equations of the two diagonal blocks to get the eigenvalues. This is the most traditional way to solve for eigenvalues and works well for small matrices. This may, however, get unusable for larger matrices.

```
import math as mth
def eigenvalues(array):
  det = array[0][0]*array[1][1]-array[0][1]*array[1][0]
  tr = array[0][0] + array[1][1]
  e1 = (tr + mth.sqrt(tr*tr-4*det))/2
  e2 = (tr - mth.sqrt(tr*tr-4*det))/2
  return e1 , e2
A = [[gamma1-2*beta1/3, sqrt(2)*beta1/3],
   [sqrt(2)*beta1/3, 5*gamma1-beta1/3]]
b = eigenvalues(A)
A1 = [[gamma1+2*beta1/3, sqrt(2)*beta1/3],
   [sqrt(2)*beta1/3, 5*gamma1+beta1/3]]
b1 = eigenvalues(A1)
eval1 = 5*gamma1-beta1
eval2 = 5*gamma1+beta1
eval3 = gamma1-2*beta1
```

```
eval4 = gamma1-2*beta1
print('Shifts in energy are: \n' , -1*eval1, '\n' , -1*eval2, '\n' , -1*
    eval3, '\n' , -1*eval4, '\n' , -1*b[0], '\n' , -1*b[1],  '\n', -1*b1
    [0], '\n', -1*b1[1] )
```

4.7.2 QR Decomposition

The second method we use is the QR decomposition algorithm which is explained in the appendix. We use this to get upper triangular matrices that are similar to the two diagonal blocks and hence have the same spectrum. Since the eigenvalues of a triangular matrix are simply the diagonal elements, we are done. This method is more useful for larger matrices than the method involving the characteristic equation.

```
def qr_decomp_alg(matrix_org, debug=False):
    m, n = matrix_org.shape
    matrix_ortho = matrix_org.copy()
    matrix_ortho = np.asarray(matrix_ortho, dtype=np.float)
    coefficient = np.zeros(shape=(m, n))
    coefficient[0, 0] = 1
    for i in range(1, n):
        coefficient[i, i] = 1
        for j in range(i):
            b_j = matrix_ortho[:, j]
            k_j = np.dot(b_j, matrix_org[:, i]) / np.dot(b_j, b_j)
            coefficient[j, i] = k_j
            matrix_ortho[:, i] = matrix_ortho[:, i]-(k_j * b_j)
    for i in range(n):
        divider = np.dot(matrix_ortho[:, i], matrix_ortho[:, i])
        if abs(divider) < 1e-16:
            matrix_ortho[:, i] *= 0
        else:
            divider = np.sqrt(divider)
            matrix_ortho[:, i] /= divider
            coefficient[i, :] *= divider

    return matrix_ortho, coefficient

A2 = np.array(A)

for i in range(100):
  q, r = qr_decomp_alg(A2)
  A2 = np.dot(r, q)

print(A2)
print('\n')
print('The bottom left element is practically zero (precision) compared to
    the other entries.')
A2[1][0] = 0
print('\n')
print(A2)
print('\n')
```

```
print('Since the matrix is in upper triangular form, the eigenvalues are
    the diagonal elements, that is:')
print(A2[0][0])
print(A2[1][1])
print('\n')
A3 = np.array(A1)

for i in range(100):
  q, r = qr_decomp_alg(A3)
  A3 = np.dot(r, q)

print(A3)
print('\n')
print('The bottom left element is practically zero (precision) compared to
     the other entries.')
A3[1][0] = 0
print('\n')
print(A3)
print('Since the matrix is in upper triangular form, the eigenvalues are
    the diagonal elements, that is:')
print(A3[0][0])
print(A3[1][1])
print('\n')

print('Shifts in energy are: \n' , -1*eval1, '\n' ,-1*eval2, '\n' , -1*
    eval3, '\n' , -1*eval4, '\n' , -1*A2[0][0], '\n' , -1*A2[1][1],  '\n',
    -1*A3[0][0], '\n', -1*A3[1][1] )
```

5 Appendix

5.1 Proof Of The Selection Rules In Stark Effect

The proofs for the above stated rules require basic knowledge about the angular momentum operators. The commutator of L_z and z that is $[L_z, z]$ is 0. The basis used is the $|n,l,m\rangle$ basis and these vectors are simultaneous eigenvectors of the L_z and L^2 operators. Now, sandwiching the $[L_z, z]$ between any two such vectors

$$\langle nlm|[L_z, z]|n'l'm'\rangle = 0$$

which gives

$$\hbar(m - m') \langle nlm|z|n'l'm'\rangle = 0$$

which clearly implies that $m = m'$ for these matrix elements to be non-zero. The selection rule for l is not subtle. It can be shown with the knowledge of commutation relations and some algebraic manipulation that

$$[L^2, z] = 2i\hbar(xL_y - L_x y)$$

Similarly

$$[L^2, [L^2, z]] = 2\hbar^2(L^2 z + zL^2)$$

which implies,

$$\langle nlm|L^4z - 2L^2zL^2 + zL^4 - 2\hbar^2(L^2z + zL^2)|n'l'm'\rangle = 0$$

Now using the fact that these states are the eigenvectors of the L^2 operators with eigenvalues $l(l+1)\hbar^2$ we can simplify this expression to get

$$(l + l' + 2)(l + l')(l - l' + 1)\,\langle nlm|z|n'l'm'\rangle = 0$$

which implies that the matrix elements vanish for all pairs of states except when

$$l = l' = 0$$

or when

$$l = l' \pm 1$$

However, a state with $l = 0$ is spherically symmetric so the expectation value of z turns out to be zero in these cases. In conclusion, the matrix elements vanish except when $l - l' = \pm 1$.

5.2 Romberg Integration

Romberg integration uses the principle of Richardson extrapolation based on the integral estimates of multiple applications of trapezoidal rule. Trapezoidal method gives an error of the order $O(h^2)$ and for the first iteration, it combines two such order 2 estimates to give error of order $O(h^4)$. Successive iterations produce estimates of order $O(h^{2+2n})$. This method gives better estimate than Newton-Cotes formulas for most of the cases and uses lesser computational resources.
The formula implemented is

$$I_{j,k} = (4^{k-1}I_{j+1,k-1} - I_{j,k-1})/(4^{k-1} - 1)$$

5.3 Leibniz Formula

It can be shown that there is only one multi-linear (linear separately in each variable) alternating (is 0 whenever the set of arguments is linearly dependent) function F from the set of all $n \times n$ matrices over a field to the field such that $F(I) = 1$ (determinant). That is, there exists only one determinant function for the set of $n \times n$ matrices. The interesting proof for the existence and uniqueness of the determinant leads us to the aforementioned formula.

$$det(A) = \sum_{\sigma \in S_n} sgn(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)}$$

where S_n is the permutation group and $sgn(\sigma)$ is the sign or parity of the permutation, which depends on whether the number of transpositions that need to be multiplied to get the permutation is odd (-1) or even (1).

5.4 Durand-Kerner Algorithm

Let $r_1, r_2, ..., r_n$ be the n complex roots of an n degree polynomial $f(x)$. Note that we do not know the roots beforehand. For each of the roots r_i define

$$d_i(x) = x - \frac{f(x)}{\prod_{j \neq i}(x - r_j)}$$

Note that $d_i(r_i) = r_i$. We start with initial guesses for the roots and then pass the guess for r_1 to d_1. This is the updated value of r_1. Then we use the updated value of r_1 in the denominator of the second term of d_2 and pass our guess for r_2 into d_2. We then use the updated value of r_2 and r_1 for d_3 and pass r_3 into it. We repeat this till d_n. This is one iteration. Next time we use all the updated values for d_1 and follow the same scheme. If the superscript denotes iteration, then iteration k of the algorithm can be stated as follows.

$$r_i^k = r_i^{k-1} - \frac{f(r_i^{k-1})}{(r_i^{k-1} - r_1^k)(r_i^{k-1} - r_2^k).....(r_i^{k-1} - r_{i-1}^k)(r_i^{k-1} - r_{i+1}^{k-1})(r_i^{k-1} - r_{i+2}^{k-1})....(r_i^{k-1} - r_n^{k-1})}$$

for all $i = 1, 2, ..., n$. We re-iterate till the r_i essentially stop changing relative to the desired precision. Note that as we get closer and closer to the root, r_i changes by a lesser and lesser amount. This shows the convergence of the method.

5.5 QR Decomposition

This method entails the factorisation of the matrix at hand, say A, into a product of an orthogonal matrix (columns and rows are orthonormal vectors) (Q) with an upper triangular matrix (R).
Consider the Gram-Schmidt procedure, with the vectors to be considered in the process as columns of the matrix A. That is

$$A = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n \end{bmatrix}$$

Then by applying the Gram-Schmidt procedure on these vectors

$$\mathbf{u}_1 = \mathbf{a}_1,\ \mathbf{e}_1 = \frac{\mathbf{u}_1}{||\mathbf{u}_1||}$$

$$\mathbf{u}_2 = \mathbf{a}_2 - (\mathbf{a}_2.\mathbf{e}_1)\mathbf{e}_1,\ \mathbf{e}_2 = \frac{\mathbf{u}_2}{||\mathbf{u}_2||}$$

$$\mathbf{u}_{k+1} = \mathbf{a}_{k+1} - (\mathbf{a}_{k+1}.\mathbf{e}_1)\mathbf{e}_1 - ..... - (\mathbf{a}_{k+1}.\mathbf{e}_k)\mathbf{e}_k,\ \mathbf{e}_{k+1} = \frac{\mathbf{u}_{k+1}}{||\mathbf{u}_{k+1}||}$$

This gives us a set of vectors $\mathbf{e}_k$ that are orthonormal.
Thus the QR decomposition is

$$A = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \dots & \mathbf{e}_n \end{bmatrix} \begin{bmatrix} \mathbf{a}_1.\mathbf{e}_1 & \mathbf{a}_2.\mathbf{e}_1 & \dots & \mathbf{a}_n.\mathbf{e}_1 \\ 0 & \mathbf{a}_2.\mathbf{e}_2 & \dots & \mathbf{a}_n.\mathbf{e}_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{a}_n.\mathbf{e}_n \end{bmatrix} = QR$$

To find the eigenvalues of the given matrix, we must perform a QR decomposition, writing the matrix as a product of an orthogonal matrix and an upper triangular matrix, multiply the factors in the reverse order, and iterate.
Let A be the input matrix and let A_k be the matrix at step k where $A_0 = A$. Then compute the QR decomposition $A_k = Q_k R_k$. Therefore the iteration is

$$A_{k+1} = R_k Q_k$$

This is a similarity transformation, hence the eigenvalues do not change with the transformation.

$$A_{k+1} = R_k Q_k = Q_k^{-1} Q_k R_k Q_k = Q_k^{-1} A_k Q_k$$

Under certain conditions, the matrices A_k converge to a triangular matrix. The eigenvalues of a triangular matrix are listed on the diagonal, and the eigenvalue problem is solved by taking the diagonal elements of A_k.

6 Bibliography

6.1 Stark Effect

- http://www.physics.drexel.edu/~bob/Manuscripts/stark.pdf
- https://en.wikipedia.org/wiki/Stark_effect
- https://en.wikipedia.org/wiki/Romberg%27s_method
- https://en.wikipedia.org/wiki/Durand%E2%80%93Kerner_method
- http://farside.ph.utexas.edu/teaching/qmech/Quantum/node121.html
- Professor Subhash Chaturvedi's class notes
- J.J. Sakurai and Jim Napolitano, Modern Quantum Mechanics, 2nd ed. (Pearson)
- Steven C. Chapra and Raymond P. Canale, Numerical Methods for Engineers, 6th ed. (McGraw-Hill)

6.2 Zeeman Effect

- http://phys.iit.edu/~segre/phys406/15S/lecture_05.pdf
- http://www.fizika.unios.hr/kvm2/wp-content/uploads/sites/76/2013/11/Lecture_3_Perturbation_theory_applications.pdf
- https://royalsocietypublishing.org/doi/10.1098/rspa.1928.0051
- http://hitoshi.berkeley.edu/221A/final.nb.pdf
- https://ocw.mit.edu/courses/physics/8-06-quantum-physics-iii-spring-2018/lecture-notes/MIT8_06S18ch2.pdf
- https://www.cmi.ac.in/~dev/ATOM/6a.pdf
- https://www.cmi.ac.in/~govind/teaching/qm2-o11/qm2-lecture-notes-gk.pdf
- https://iopscience.iop.org/article/10.1088/1361-6404/aab821/pdf
- https://en.wikipedia.org/wiki/QR_decomposition
- https://en.wikipedia.org/wiki/QR_algorithm
- D.J. Griffiths, Introduction to Quantum Mechanics, 2nd ed. (Pearson Prentice Hall)