Building a Shell Model Code: The Pairing Model and the sd-Shell (and comparing results with NuShellX)

Gianluca Salvioni[1], Ina K. B. Kullmann[2], Matthew Shelley[3], and Gilho Ahn[4]

[1]Department of FILL IN, LaTeX University, *youremail@edu.com*
[2]Department of Physics, University of Oslo, *i.k.b.kullmann@fys.uio.no*
[3]Department of FILL IN, LaTeX University, *youremail@edu.com*
[4]Department of FILL IN, LaTeX University, *youremail@edu.com*

July 20, 2017

Abstract

We have first implemented the pairing model which have a analytical solution (to benchmark the code). Then implemented the sd shell —> more general shell-model program that allows you to study general nuclear structure problems.

developing our own shell-model code that can perform shell-model studies of the oxygen isotopes using standard effective interactions (provided by us) using as example the 1s0d shell as model space.

We have also used the NushellX code in order to perform more advanced shell-model studies and compare the results obtained with your own shell-model code to those of NushellX

and found that.....results...

I. Introduction

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

II. Theory

i. Exact results: The Pairing Model

The pairing model is one of a few systems that has a simple analytic solution. It is therefore very useful to use this model to check that the numerical solution matches the analytic solution while developing the code.

The pairing problem consists of a system where the fermions combines together in pairs of two, one with spin up and one with spin down. This model does not allow so-called 'breaking of pairs' meaning that the pairs of particles always will be coupled together. An excitation must excite two particles at the same time.

Mathematically the pairing model can be described by a simplified Hamiltonian consisting of a unperturbed one-body operator H^0 and a pertubation so-called 'pairing interaction term' H^I. We will limit ourselves to at most two-body interactions so that we can write the operators as:

$$\hat{H}_0 = \xi \sum_{p,\sigma} (p-1)\hat{a}^\dagger_{p\sigma}\hat{a}_{p\sigma}, \tag{1}$$

$$\hat{V} = -\frac{1}{2} g \sum_{p,q} \hat{a}^\dagger_{p+}\hat{a}^\dagger_{p-}\hat{a}_{q-}\hat{a}_{q+}, \tag{2}$$

so that the full Hamiltonian is given by the sum of the unperturbed term and the interacting part $\hat{H} = \hat{H}_0 + \hat{V}$. The fermion creation and annihilation operators are given by $\hat{a}^\dagger_p$ and $\hat{a}_q$ respectively and $pqrs$ represent all possible single-particle quantum numbers. To simplify the expressions we set the spacing between successive single-particle states given by $\xi = 1$.

We will let the single-particle states$|p\rangle$ be eigenfunctions of the one-particle operator $\hat{h}_0$. The above Hamiltonian acts in turn on various many-body Slater determinants constructed from the single-basis defined by the one-body operator $\hat{h}_0$.

The two-body operator $\hat{V}$ consists of one term:

$$\langle q_+ q_- | \hat{V} | s_+ s_- \rangle = -g \tag{3}$$

representing the pairing contribution with (for simplicity) constant strength g. The labeling requires that for a given matrix element $\langle pq|\hat{V}|rs\rangle$ the states p and q (or r and s) must have opposite spin ($\sigma = \pm 1$)

It can be shown that the unperturbed Hamiltonian $\hat{H}_0$ and $\hat{V}$ commute with the spin projection $\hat{S}_z$ and the total spin $\hat{S}^2$. This allows us to block-diagonalize the full Hamiltonian. For the pairing case we have assumed that $S = 0$ giving the so-called 'no-broken pair' case.

Constructing the Hamiltonian matrix As an exact analytic solution we chose a system consisting of only four particles with a single-particle space consisting of only the four lowest levels $p = 1, 2, 3, 4$. In our system every level p contains two particles, one with spin up and one with spin down. The goal is to set up all possible Slater determinants and the Hamiltonian matrix using second quantization and find all eigenvalues by diagonalizing the Hamiltonian matrix.

We construct the basis:

$$|\Phi_0\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad |\Phi_1\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \cdots \quad |\Phi_5\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix},$$

where

$$
\begin{aligned}
|\Phi_0\rangle &= \hat{a}^\dagger_{2+}\hat{a}^\dagger_{2-}\hat{a}^\dagger_{1+}\hat{a}^\dagger_{1-}|0\rangle = \hat{P}^+_2\hat{P}^+_1|0\rangle,\\
|\Phi_1\rangle &= \hat{P}^+_3\hat{P}^+_1|0\rangle,\\
|\Phi_2\rangle &= \hat{P}^+_4\hat{P}^+_1|0\rangle,\\
|\Phi_3\rangle &= \hat{P}^+_3\hat{P}^+_2|0\rangle,\\
|\Phi_4\rangle &= \hat{P}^+_4\hat{P}^+_2|0\rangle,\\
|\Phi_5\rangle &= \hat{P}^+_4\hat{P}^+_3|0\rangle.
\end{aligned}
$$

Given the above Slater determinants we can now compute the matrix elements $\langle\Phi_i|\hat{H}|\Phi_j\rangle$ using the Hamiltonian of Eq. (1). The one-body operator acts only on the diagonal and results in terms proportional with $(p1)$. The interaction will excite or deexcite a pair of particles from level q to level p. Using this it is easy to see that the Hamilotnian matrix becomes: (REWRITE THIS PARAGRAPH, ALL COPY)

$$
\hat{H} = \begin{pmatrix}
2-g & -g/2 & -g/2 & -g/2 & -g/2 & 0\\
-g/2 & 4-g & -g/2 & -g/2 & 0 & -g/2\\
-g/2 & -g/2 & 6-g & 0 & -g/2 & -g/2\\
-g/2 & -g/2 & 0 & 6-g & -g/2 & -g/2\\
-g/2 & 0 & -g/2 & -g/2 & 8-g & -g/2\\
0 & -g/2 & -g/2 & -g/2 & -g/2 & 10-g
\end{pmatrix}. \tag{4}
$$

For a given g this matrix can be used as the analytic result to compare with the output from the shell model code for the pairing case. ....

III. Building the shell model code

develop a code which sets up the above Hamiltonian matrices for two and four particles in 2 and 4 single-particles states (the same as what you did in exercises b) and c) and obtain the eigenvalues.

What did we choose?

- Decide whether you want to read from file the single-particle data and the matrix elements in m-scheme, or set them up internally in your code. The latter is the simplest possibility for the pairing model, whereas the first option gives you a more general code which can be extended to the more realistic cases discussed in the second part.
- Based on the single-particle basis, write a function which sets up all possible Slater determinants which have total M = 0. Test that this function reproduces the cases in b) and c). If you make this function more general, it can then be reused for say a shell-model calculation of sd-shell nuclei in the second part.
- Use the Slater determinant basis from the previous step to set up the Hamiltonian matrix.
- With the Hamiltonian matrix, you can finally diagonalize the matrix and obtain the final eigenvalues and test against the results of b) and c).

Include some of this? Or too much? :
The lecture slides contain a rather detailed recipe on how to construct a Slater determinant basis and how to set up the Hamiltonian matrix to diagonalize.

i. first step....

ii. Unit tests and benchmarks

(have not done this?)

One obvious case is that of removing the two-body interaction. Then we have only the single-particle energies. For the case of degenerate single-particle orbits, that is one value of total single-particle angular momentum only j, with degeneracy $\Omega = 2j + 1$, one can show that the ground state energy E_0 is with n particles

$$E_0 = -\frac{g}{4} n(\Omega - n + 2) \tag{5}$$

Enlarge now your system to six and eight fermions and to $p = 6$ and $p = 8$ single-particle states, respectively. Run your program for a degenerate single-particle state with degeneracy Ω and test against the exact result for the ground state. Introduce thereafter a finite single-particle spacing and study the results as you vary g, as done in b) and c). Comment your results.

(describe the unit test that we have implemented and how)

IV. NuShellX

(describe what is, then put results in results section?)

V. Results

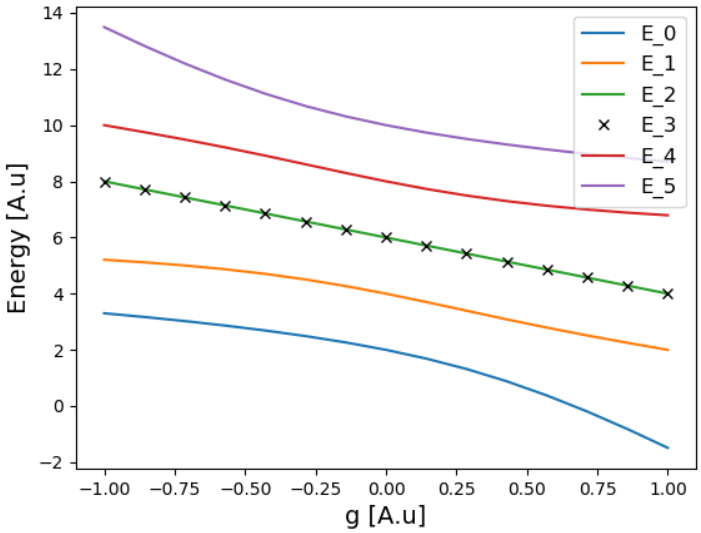

Figure 1: *The energy levels as a function of the strength g for the analytic case of the pairing model.*

In figure 1 we see the eigenvalues of the Hamiltonian matrix (4) for the pairing model as a function of the strength $g \in [-1, 1]$. (The diagonalization was done with numpy)

–>Comment the behavior of the ground state as function of g?

VI. Discussion

VII. Conclusions

Structure of report (crude first idea):

1. Title
2. Abstract
3. Introduction
   - Why useful
   - Where useful (nuc. chart)
4. The Pairing problem
   - theory
   - analytic / exact results
5. Building a shell-model code
   - Describe the different parts/blocks of the code (basis, SD, states, int, hamiltonian...)
   - Unit tests / benchmarks (analytic/exact results)
   - Psedo code?
   - Accuracy of our code
   - Where our code fails, why and proposing solutions
6. NuShellX
   - describe why useful / 'powerful tool' and what may be calculated
   - compare with our own code
   - Possible 'future calculations'
7. Conclusions / Summary

References

[Github of the TALENT School] MHJ and Alex Brown *link goes here*,