PHYS4070 — Due date: Friday 26/05/2023, 5pm

Project 3: Dynamics of Interacting Quantum Systems

Part 1: Bright Solitons in the Non-Linear Schrödinger Equation

BACKGROUND

The non-linear Schrödinger equation can be used to model the dynamics of interacting waves in a wide range of systems, including water waves, light propagation and Bose–Einstein condensates. In one spatial dimension, the equation (in dimensionless form) reads as follows:

$$i\frac{\partial\psi}{\partial t} = -\frac{\partial^2\psi}{\partial x^2} + g|\psi|^2\psi\,, \tag{1}$$

where $\psi(x,\,t)$ is the complex wave function and g is the interaction strength. Both attractive ($g < 0$) and repulsive ($g > 0$) interactions can be of interest, but here we will restrict our analysis to the attractive case.

When the interactions are attractive, Eq. (1) is known to feature "bright soliton" solutions. These are stable wave packets, which are formed when the dispersion induced by the spatial derivative is balanced by the non-linear interaction.[1] Below, we will explore such structures in detail.

QUESTIONS

(a) Using C++, write code to numerically solve Eq. (1) on a domain $-L/2 \leq x \leq L/2$, with $L = 20$, and with the boundary conditions $\psi(-L/2,\,t) = \psi(L/2,\,t) = 0$. Use a fourth-order Runge–Kutta (RK4) algorithm to propagate the system in time. Evaluate the spatial derivative by using $f''(x) \approx [f(x+\Delta x) - 2f(x) + f(x-\Delta x)]/(\Delta x)^2$ (or another approximation with an explanation, if you prefer). A grid size of $N = 128$ and an RK4 time step of $\Delta t = 0.01$ should suffice (although feel free to use larger N and smaller Δt).

(b) Using your code, solve for the evolution of a plane wave:

$$\psi(x,\,0) = \cos(\pi x/L)\,. \tag{2}$$

What happens to the wave amplitude $|\psi|^2$ as you increase the attractive interaction strength from $g = 0$ to $|g| \gg 1$? Provide an explanation for this.

For the remaining questions set $\mathbf{g = -1}$.

(c) Solve for the evolution of the wave packet,

$$\psi(x,\,0) = \sqrt{2}\exp(iux)\,\mathrm{sech}(x)\,. \tag{3}$$

What do you observe for $u = \{-1,\,0,\,1\}$? Plot the peak position of $|\psi|^2$ as a function of time for a few different values of u. What does u determine? This wavepacket is known as a "bright soliton".

(d) Solve for the evolution of two solitons moving in opposite directions with relative phase θ:

$$\psi(x,\,0) = \sqrt{2}\exp(iux)\,\mathrm{sech}(x + L/4) + \sqrt{2}\exp(-iux + i\theta)\,\mathrm{sech}(x - L/4)\,. \tag{4}$$

Explore the dynamics of $|\psi|^2$ as a function of θ **(setting $\mathbf{u = 0.1}$)**. Determine the distance of closest approach for the two solitons by monitoring the positions of each peak. How would you conclude that the inter-soliton interaction varies as a function of θ?

[1] For more information, see: L. F. Mollenauer *et al.*, Phys. Rev. Lett. **45**, 1095–1098 (1980); M. Segev and G. I. Stegeman, Physics Today **51**, 42–48 (1998); G. I. Stegeman and M. Segev, Science **286**, 1518–1523 (1999); J. H. V. Nguyen *et al.*, Nature Phys. **10**, 918–922 (2014).

USEFUL TIPS

You can define π at the beginning of your code as "const double pi = 4 * atan(1)". The other math functions that you will need are {sech = 1/ cosh, cos, exp, pow, sqrt}.
[requires "#include <cmath>"]

Complex variables are initialized as type "complex<double>". You can define the imaginary unit as "const complex<double> i_unit(0.0, 1.0)". Use "norm(z)" to calculate $|z|^2$.
[requires "#include <complex>"]

To obtain the index of the maximum element of a vector V, use "int maxV = max_element(V.begin(), V.end()) − V.begin()".
[requires "#include <algorithm>"]

Part 2: The Transverse Ising Spin Model

BACKGROUND

Most thermal phase transitions arise when the symmetry properties of the equilibrium state of a system change. As the system is heated past the phase transition point it becomes more disordered, and vice versa. This change in symmetry manifests as a point of non-analyticity in the free energy as a function of temperature. In a many-body quantum system, an analogous effect can occur at zero temperature: a point of non-analyticity appears in the ground-state energy, associated with a change in the symmetry properties of the ground-state wave function.[2] A sufficient condition for a function to be non-analytic is that it possesses an ill-defined derivative at some order.

The one-dimensional *transverse Ising model* is a well known system that exhibits a quantum phase transition. The Hamiltonian for this model (setting $\hbar = 1$) reads as follows:

$$\hat{H} = -\omega \sum_{m=0}^{N-1} \hat{\sigma}_z^{(m)} - g \sum_{m=0}^{N-1} \hat{\sigma}_x^{(m)} \hat{\sigma}_x^{(m+1)}, \tag{5}$$

where

$$\hat{\sigma}_z^{(m)} = I^{\otimes m} \otimes \begin{pmatrix} 1/2 & 0 \\ 0 & -1/2 \end{pmatrix} \otimes I^{\otimes(N-1-m)} \quad \text{and} \quad \hat{\sigma}_x^{(m)} = I^{\otimes m} \otimes \begin{pmatrix} 0 & 1/2 \\ 1/2 & 0 \end{pmatrix} \otimes I^{\otimes(N-1-m)} \tag{6}$$

are the z and x Pauli spin matrices, respectively, for a single spin-one-half particle located at the m^{th} lattice site. Here, I is the 2×2 identity matrix, and $A^{\otimes n}$ denotes the tensor product of A with itself n times, e.g., $A^{\otimes 3} = A \otimes A \otimes A$ [also $A^{\otimes 1} = A$ and $A^{\otimes 0} = 1$ (note this is a 1, not an I)]. We impose periodic boundary conditions (i.e., for a ring of particles) by defining

$$\hat{\sigma}_x^{(N)} = \hat{\sigma}_x^{(0)} \tag{7}$$

in the second sum of Eq. (5) [note, Eq. (6) cannot be used to evaluate $\hat{\sigma}_x^{(N)}$ because it would give $I^{\otimes -1}$]. The first term in Eq. (5) describes the coupling of one atom to an external field that drives a transition between two single-particle states. The second term describes a nearest-neighbour "spin-spin" interaction with $g \geq 0$. **Without loss of generality, below we set $\omega = 1$.**

QUESTIONS

(a) Using C++, construct the Hamiltonian $\hat{H}$ [Eq. (5)] as a $2^N \times 2^N$ matrix, and determine its eigenvalues and eigenvectors. Feel free to re-use code from previous worksheets and

[2] For more information, see: Lev Landau, "On the theory of phase transitions", Zh. Eksp. Teor. Fiz. **7**, 19–32 (1937); Subir Sachdev, "Quantum phase transitions", Cambridge University Press, 2nd Ed. (2011).

assignments. How does your code runtime scale with atom number N, for $N = 2, 3, ..., 8$? Show this with a suitable plot.

(b) Denoting the ground-state energy of Eq. (5) by $\mathcal{E}_0$, plot $N^{-1}\mathcal{E}_0$ and $N^{-1}\partial^2\mathcal{E}_0/\partial g^2$ as a function of g, for $0 \leq g \leq 8$, and with $N = 8$. Also include the analytical result in the thermodynamic limit,

$$\lim_{N\to\infty}\left\{\frac{\mathcal{E}_0}{N}\right\} = -\frac{1}{2\pi}\,(2+g)\,\mathrm{E}\left[\frac{8g}{(2+g)^2}\right], \tag{8}$$

where

$$\mathrm{E}[x] = \int_0^{\pi/2} d\theta \sqrt{1 - x\sin^2(\theta)} \tag{9}$$

represents the complete elliptic integral of the second kind. What do you observe? Does your result for $\mathcal{E}_0$ seem reasonable at small and large g? (Think about what you would expect for the ground-state energy in each of these limits.) To evaluate the second derivative, use

$$f''(x) \approx \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{(\Delta x)^2}\,. \tag{10}$$

(c) A quantum state ψ evolves under a Hamiltonian $\hat{H}$ according to the equation, $i\partial\psi/\partial t = \hat{H}\psi$. Hence,

$$\psi(t) = e^{-i\hat{H}t}\psi(0) = Ue^{-iDt}U^\dagger\psi(0)\,, \tag{11}$$

where D is a diagonal matrix of the eigenvalues of $\hat{H}$, and U is a matrix of the corresponding eigenvectors. A common way to explore the non-equilibrium dynamics of a system is to begin in the ground state of one phase and "quench" (i.e., rapidly change) a parameter so that the initial state is no longer the ground state. The subsequent evolution can then be studied.

Explore the dynamics that arise from starting in the ground state at $g = 0$, and then evolving according to Eq. (5) with $g = 4$, for $N = 8$. Plot the time evolution of the following physical observables:

$$\begin{aligned} S_z &= \sum_{m=0}^{N-1} \langle\hat{\sigma}_z^{(m)}\rangle\,, \\ S_x &= \sum_{m=0}^{N-1} \langle\hat{\sigma}_x^{(m)}\rangle\,, \\ C_{xx} &= \sum_{m=0}^{N-1}\sum_{\substack{n=0\\n\neq m}}^{N-1} \langle\hat{\sigma}_x^{(m)}\hat{\sigma}_x^{(n)}\rangle\,. \end{aligned} \tag{12}$$

Describe the salient features of the dynamics of these observables. You may want to explore how changing g affects the dynamics. You should find that S_x remains zero. Why is this, and can you reconcile that with C_{xx} being non-zero? (Hint: think about the symmetry of the problem.)

You may wish to evaluate the time evolution [Eq. (11)] by finding U and D exactly, noting that the matrix U will be real, while e^{-iDt} will be complex. Alternatively, you may find it simpler to solve the equation of motion by using a Runge–Kutta integrator. Many other methods also exist[3] (although I would not recommend utilizing a more complicated approach unless you are already well acquainted with it).

[3]For more information, see: Pierre Pfeuty, "The one-dimensional Ising model with a transverse field", Ann. Phys. **57**, 79–90 (1970).

USEFUL TIPS

You should already have code that is able to diagonalize a symmetric matrix, which you may re-use (e.g., see the worksheet from Week 2). Since $\hat{H}$ is symmetric, you do not need to worry about its column-major versus row-major indexing. However, the matrix of eigenvectors returned will not be symmetric and its indexing will be important. I suggest checking the output for a matrix of which you already know the eigenvectors to see how it is formatted. Also, remember that DSYEV overwrites inputs, so be careful if you are re-using quantities passed to this function.

The symbol $\otimes$ denotes the tensor (or "Kronecker") product of two matrices. It will be useful to write a function that calculates the tensor product between a 2×2 matrix and an $n \times n$ matrix. Then, for instance, you can generate $\hat{\sigma}^{(i)}$ via the following pseudocode:

$\hat{\sigma}^{(i)} = 1$
for $k = 0,\ k < N,\ k^{++}$
  if $i == k$
    $\hat{\sigma}^{(i)} =$ Tensor_Product(σ, $\hat{\sigma}^{(i)}$)
  else
    $\hat{\sigma}^{(i)} =$ Tensor_Product(I, $\hat{\sigma}^{(i)}$)
  end
end

Here, σ is the 2×2 matrix of interest and I is the 2×2 identity matrix. [Note, to be consistent with the indexing of Eq. (6), you should actually replace $\hat{\sigma}^{(i)}$ above by $\hat{\sigma}^{(N-1-i)}$, however this is inconsequential due to symmetry.] For the interaction terms, you will need two conditions to obtain the two non-trivial parts, i and j. So far, I have omitted any mention of memory allocation, which will need to be considered in the function calls, $\hat{\sigma}^{(i)} =$ Tensor_Product(A, $\hat{\sigma}^{(i)}$), since the matrix $\hat{\sigma}^{(i)}$ will change in size. You could introduce some intermediate matrix to accommodate this, or more elegantly, modify your matrix class to do this automatically, inside a tensor product function which belongs to that class.

Recall that $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$, and in particular, $(A \otimes I)(I \otimes B) = A \otimes B$. By considering the latter result, you can calculate the interaction terms directly as follows by using a tensor product:

$$\hat{\sigma}_x^{(m)} \hat{\sigma}_x^{(m+1)} = I^{\otimes m} \otimes \begin{pmatrix} 0 & 1/2 \\ 1/2 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1/2 \\ 1/2 & 0 \end{pmatrix} \otimes I^{\otimes (N-2-m)}. \tag{13}$$

This will most likely be easier than calculating $\hat{\sigma}_x^{(m)}$ and $\hat{\sigma}_x^{(m+1)}$ independently, and then matrix multiplying to obtain $\hat{\sigma}_x^{(m)} \hat{\sigma}_x^{(m+1)}$.

The "chrono" library will be useful for timing your calculation in Question (a), for example:

```
#include <chrono>

auto start = chrono::steady_clock::now();
{code body}
auto end = chrono::steady_clock::now();
double time = chrono::duration<double>(end − start).count();
```

The "auto" type tells the compiler to determine the data type from the initialization; we change this to a human-readable format in "time".

Although E[x] is available to use via the C++ libraries if you wish, for our purposes it suffices to calculate E[x] by using numerical integration according to its definition:

$$\mathrm{E}[x] \approx \frac{\pi}{2M} \sum_{k=0}^{M} \sqrt{1 - x \sin^2\left(\frac{\pi k}{2M}\right)}. \tag{14}$$

Please ensure that you read this page carefully.

1 Important Additional Information

You should submit your assignment as a combination of (i) a written report, and (ii) any scripts [C++, gnuplot, Python, bash, etc.] utilized to run the simulations and plot the data.

1.1 Report

The format and general appearance of your report should be similar to a scientific journal article. It should be divided into **two** separate sections: Part 1 on *Bright Solitons in the Non-Linear Schrödinger Equation*; and Part 2 on *the Transverse Ising Spin Model*. In **each** section, you should write a brief introduction outlining the problem and main objectives, as well as a brief conclusion summarising what you have accomplished (however, no external references are necessary).

In the main body of each section of your report, you should provide the following points:

1. All the results and plots that you are asked to produce, including any important steps from your working (making sure that you answer **all** aspects of the questions);
2. A full explanation of the steps that you have followed to obtain your data/figures/results;
3. An analysis and interpretation of the **physics** embodied by your results.

1.2 Submission

You should submit your assignment via Blackboard, preferably by uploading a single zipped file which includes:

1. Your report as a .pdf file with the filename "surname_studentID.pdf";
2. All C++ scripts that you have used to obtain your results, clearly labelled (e.g., as "P1_Qa.cpp", "P2_Qb.cpp", etc.);
3. All plotting scripts that you have used (from gnuplot, Python, or any other plotting program).

Important: Make sure that your submission is self-contained, and that your source codes can be compiled on their own and produce results consistent with your report. I should not need to modify your scripts in any way to generate your answers to the assignment [varying input options is fine, so long as these are read in by your codes (e.g., from the command line), and they do not need to be recompiled].

1.3 Marking Guide

Your assignment will be marked based on the following criteria (Part 1, Part 2):

1. **Correctness (30%, 20%):** Is your code correct, or are there bugs? Are your answers to the questions correct? Did you produce all the plots that were requested?
2. **Explanation (30%, 20%):** Have you explained your steps comprehensively? Did you describe the data shown in your figures? Is the report well written and does it flow logically?
3. **Interpretation and Insight (30%, 40%):** Have you demonstrated a proper understanding of the system studied and your results? Did you provide the correct physical interpretation?
4. **Code Quality (10%, 20%):** Is your code well written and well implemented, optimized for the task at hand? Is it clear, straightforward to understand, and commented where necessary?