Simulation of Michelson Interferometer Setup

By Anish Rajan

Abstract

This project is based on simulating a graphic user interface for the well known Michelson Interferometer setup. The programming of this project has been done using javascript, html and css and is rendered on a web browser. The interface has a window with options to adjust the tilt angles for mirrors M1 and M2 and the distance of the movable mirror M2 , a screen for displaying the fringes that are being formed, beam correction options to check if the incident beam is parallel. The primary objective of this project is to simulate the misalignment effects that would occur in a real setup so that students are able to identify them when they perform the experiment. The secondary objective of this project is to help students understand how different kinds of fringes (fringes of equal inclination and fringes of equal width) are formed in the Michelson interferometer and what parameters influence them. A real time cause effect connection can help students build an intuition and also understand how equations affect the observed pattern.

Keywords - Interferometer, fringes, misalignment

Simulating heat capacity of $La_{07}Gd_{01}Sr_{02}MnO_3$

By Amandeep Kaur

Abstract

This simulation models and analyzes the temperature-dependent behavior of the heat capacity of the compound $La_{07}Gd_{01}Sr_{02}MnO_3$ using Python. The purpose is to computationally reproduce and interpret the experimental results obtained from the Physical Property Measurement System (PPMS). This system is helpful in determining various physical properties such as, heat capacity, AC transport, resistivity, thermal conductivity, and more.The measurement of heat capacity provides information about lattice, electronic, and even magnetic properties of materials. Heat capacity helps to determine how a substance responds to heat changes. The measurements taken at different temperatures allow us to distinguish phonon, electronic and magnon contributions at theoretical and experimental levels.The Simulated data is generated over a temperature range of 2–300 K, and linear regression is applied to extract the constants β (lattice term) and δ (magnon term). Additional analyses, including residual evaluation, contribution separation, and Debye temperature estimation, will incorporated to better understand the physical origins of the observed behavior.The sample used for the calculation of heat capacity is$La_{07}Gd_{01}Sr_{02}MnO_3$ which is an insulator. The electronic contribution was negligible and hence, results obtained contain only magnetic and lattice contributions. These contributions are calculated using a relation; $C/T^3 = \beta + \delta T^{-3/2}$. Overall,the simulation demonstrates how heat capacity increases with temperature, exhibits dominant magnetic effects at low temperatures, and transitions to lattice-controlled behavior at higher temperatures. This computational approach validates theoretical models and provides deeper insight into phonon–magnon interactions in insulating materials.

Enhancing computational efficiency to test Kibble-Zurek's scaling law in out-of-equilibrium Ising model simulations

By Jose Armando Perez-Loera

Abstract

The Kibble mechanism plays a prominent role in early Universe theories, providing an explanation for the possible formation of cosmic strings. Zurek [1] suggested an analogous analysis in liquid helium under rapid cooling, leading to the establishment of a scaling law that relates density of topological defects with the quenching rate when a system passes through its critical temperature, which has been verified in condensed matter systems. Although equilibrium studies of the Ising model are well known, testing the Kibble-Zurek scaling law in the out-of-equilibrium system proved successful in tracking domain walls in dimensions d = 1, 2, 3. It demonstrated not only that this mechanism holds more generally than originally predicted, but also that the resulting exponents do not depend on the update algorithm used, which hints at a physical meaning. This work presents an improved computation-time study of simulations for the classical 2d Ising model out of equilibrium, examining domain evolution dynamics during cooling processes for two Monte Carlo algorithms.

[1] P. Laguna and W. H. Zurek, Physical Review Letters 78, 2519 (1997).

Solidstate-Lab: a Haskell Quantum Mechanics Solver Pack

By Liu Wei

Abstract:

We present solidstate-lab, a compact, pedagogical Haskell package for learning and exploring quantum mechanics and condensed-matter models through functional programming. The library implements a unified suite of solvers: a finite-difference stationary Schrödinger eigen-solver (1D/2D/3D), a high-accuracy Numerov–Cooley method for 1D bound states, split-operator and Chebyshev propagators for real-time dynamics, a lightweight tight-binding/Bloch engine for band structures (including SSH and graphene), phonon dispersions for mono/diatomic chains, and a radial solver for central potentials (hydrogen).

The design is pure and compositional: grids, stencils, Hamiltonian application, and boundary conditions are decoupled, enabling reuse and clear derivations that mirror textbook equations. A built-in self-verification framework quantifies accuracy via benchmarks (infinite well, harmonic oscillator, hydrogenic levels). Typical precision should be able to to reach 10^{-3} to 10^{-6} in energies with modest grids; time-evolution conserves norm to machine precision. The solvers in the package are verified using this framework and those resulting codes are given as tutorials of this project.

A simple Command Line Interface runs canonical problems and exports CSVs/plots, supporting a two-week instructional workflow that spans: wells and oscillators, band gap formation, tight-binding dispersions, phonon branches (acoustic/optical), and 3D central-potential spectra. The package demonstrates how Haskell's abstraction and strong types produce concise, testable scientific code, making it a practical study companion for physics students transitioning to functional programming.

Simulation of Thin-film X-ray Reflectivity Using the Parratt Formalism

Genevieve Uzoma

October 28, 2025

Abstract: This project proposes the development of a computational simulation framework for modeling X-ray reflectivity (XRR) profiles of thin films using the Parratt formalism. The primary goal is to reproduce and analyze the reflectivity curve R(Qz) for a single film on a substrate, where the key structural parameters film thickness, interface roughness, and refractive index contrast can be systematically varied. Implementing this model will enable a predictive understanding of Kiessig fringe formation and help optimize experimental scan ranges before laboratory measurements. The simulation will also estimate film thickness directly from fringe spacing $\Delta Q \approx 2\pi/t$, allowing comparison with expected deposition outcomes. By integrating numerical modeling with experimental principles, this work bridges the gap between computation and characterization, demonstrating how simulation can enhance data interpretation and experimental efficiency in materials physics.

Keywords: X-ray reflectivity, Parratt formalism, thing film simulation, Kiessig fringes, computational physics.

Simulating magnetic hysteresis loops using Preisach-like approach

By Jianhua Yue

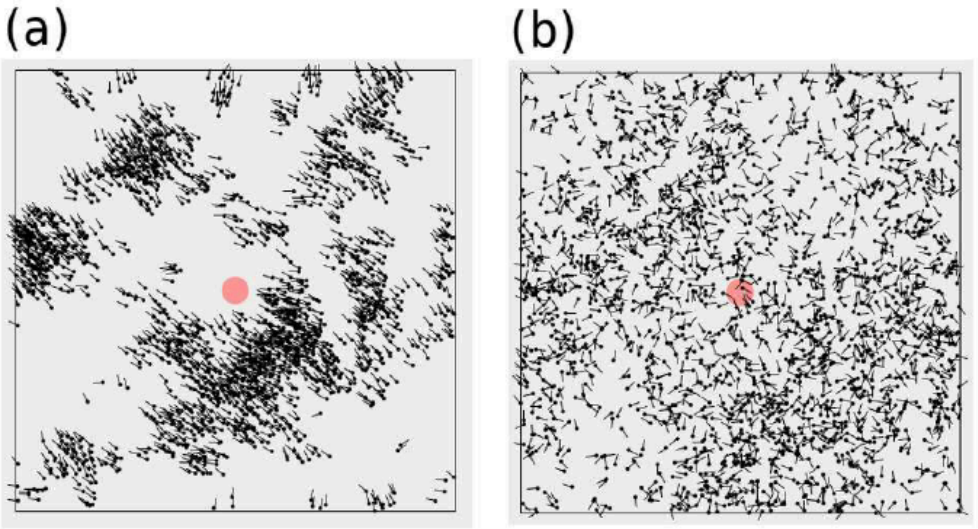
Abstract

In this presentation, I demonstrate how a simple multi-element bistable model, often referred to as a Preisach-like approach, can be used to simulate magnetic hysteresis loops. Instead of relying on experimental measurements alone, this model allows us to reproduce key magnetic features—such as saturation, remanence, coercivity, and minor-loop behavior—by adjusting a small set of intuitive parameters. The entire simulation is implemented in Octave, making it easy to visualize how distributions of switching thresholds among many independent hysterons collectively generate the macroscopic hysteresis loop. This computational approach provides a flexible and transparent way to understand magnetic hysteresis and to explore how material properties influence the shape of M–H curves.

Viscek Models

By Justus McRae

From the marching of penguins to bacteria colony formation, collective motion is everywhere around us in nature. In 1995, Tamás Vicsek introduced the Vicesk model, which has been widely used to model and study the collective motion of active particles. Over time, the Vicsek model has been changed in order to obtain a clearer description of the movement of more and more complex systems, like fish schooling and bird flocking. In order to explore the Vicsek model and its applications to real world phenomena further, various variations of the model will be implemented using a combination of Python, just in time compilation, and C++. These variations from the original Vicsek model include: variable field of view, multiple particle types, and particle cohesion. By observing several parameters of the system to determine order, the properties of the Vicsek model will be explored in both high noise and low noise systems.



Examples of an ordered system (a) under a high amount of noise, and a disordered system (b) under a low amount of noise. Taken from *Aldana et al.*, (2009).

Computational Efficiency of Metropolis method Simulations of XY Magnetism

By Sam Studdy

Abstract

The statistical mechanics of a magnetic crystal system can be approximated using the Metropolis method. It performs changes to the configuration of the spins based on the energy and temperature of the system. After many iterations, it uses the path in configuration space to calculate physical properties such as specific heat *C*. This method will be implemented for a square crystal lattice of XY spins, where the energy is defined as

$$E = \sum_{i,j} J_{i,j} \vec{S}_i \cdot \vec{S}_j$$

with the spin vectors confined to a plane. From the set of energies obtained along the path, the Fluctuation-Dissipation theorem will be used to get the specific heat, with

$$C \propto \frac{\langle E^2 \rangle - \langle E \rangle^2}{T^2}$$

for various system temperatures T . The program is in Octave, with the run times compared between pre-compiled and line-by-line compilation.