LIP Internship: *High Energy Physics numerical simulations with GPUs and ML*

# Compendium of problems

**Author**: Marco Leitão[a]

## Contents



[a]Email: mleitao@lip.pt

# I. Numerical Methods

**Problem 1. Numerical errors in derivatives**

In Lecture 1, we have introduced the total numerical error in derivatives as $E = E_{\text{acc}} + E_\epsilon$, where $E_{\text{acc}}$ is the error associated to the analytical Taylor expansion on the resolution step and $E_\epsilon$ the error associated to the numerical precision. Check the lecture slides for the full expressions in the case of central difference derivatives.

In this exercise, your task is to do the same analysis for the *forward difference derivative*, which has an accuracy error of $\mathcal{O}(h)$, and the *central difference second derivative*, which is of $\mathcal{O}(h^2)$. You should be able to derive all the formulas, including the Taylor expansions, by yourself (you can check literature for help). Plot the error total as a function of $h$ (in full log-axis), for a function of your choice, and compare the numerical error of both methods. What can you conclude?

**Problem 2. Lotka-Volterra prey-predator problem**

A classical model for the population of prey and predators in a given ecosystem are the Lotka-Volterra equations. This is a coupled set of two differential equations, given by

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= \alpha x - \beta xy \\ \frac{\mathrm{d}y}{\mathrm{d}y} &= \theta xy - \gamma y, \end{aligned} \tag{I.1}$$

where $x$ and $y$ correspond to the number of preys and the number of predators, respectively. The constant $\alpha$ corresponds to the growth rate of the preys, $\beta$ is the predation rate, $\theta$ the growth rate of predators per prey eaten and $\gamma$ the predator death rate.

You are free to explore this problem however you want, by solving the ODEs for different combinations of variables and initial conditions, using your preferred method. Here are some suggested studies:

i) Find the equilibrium points of the system and study their stability;

ii) Plot the phase space of the system, i.e., plot $x$ vs. $y$, for different configurations;

iii) Do some numerical analysis;

iv) Explore variations of the problem, e.g., add a new predator higher in the *food chain*, add a quadratic term to the equations, etc.;

v) Do animations and/or get nice and informative plots (this is quite important). Tip: use *matplotlib* for your plots (you can write a script that imports your results from, for instance, your C++ routine, and builds plots with that data).

**Problem 3. Lennard-Jones potential**

The interaction of two atoms can be modeled using the Lennard-Jones Potential, expressed by

$$V(r) = 4\epsilon \left[ \left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], \tag{I.2}$$

where $r$ is the distance between the two atoms, $\epsilon$ is the depth of the potential and $\sigma$ is the distance at which the potential is zero.

In the file `lennard_jones.csv`, you can find experimental data with uncertainties for the measurement of the potential of between a pair of atoms. Solve the following questions:

i) Read the data, *fit* the data to the Lennard-Jones potential in Eq.(I.2), thus extracting the parameters $\epsilon$ and $\sigma$. You should explore techniques and/or libraries to fit the data with uncertainties;

ii) Plot the data and the fitted curve, both with uncertainties;

iii) Perform an analytical harmonic oscillator (HO) approximation around the minimum of the potential ($V(r) \sim r^2$) (*small oscillations*).

iv) Consider that the motion of the atoms are approximately classical and one dimensional, ie, described by the Lagrangian

$$L = \frac{1}{2}\mu \dot{r}^2 - V(r), \tag{I.3}$$

where $\mu$ is the reduced mass of the atoms (use $\mu = 1$). Use the Euler-Lagrange equations to derive the equation of motion for the distance between the atoms. Then, solve analytically the equations of motion for small oscillations (HO approximation).

v) Implement a numerical routine to solve the equation of motion with a general potential $V(r)$. You can use RK methods, or you can try to implement the Chebyshev polynomial method. Then, use your numerical method to solve the equation of motion for the HO approximated potential, for adequate initial conditions. Solving numerically a limiting case of a problem that has analytical solution is a good way to benchmark your method. Finally, after making sure that the method matches the analytical predictions, you should solve the full case for $V(r)$. Do the simulation for various initial conditions and compare them with the HO case. Study the regimes of compatibility and discrepancies between the full solution and the HO.

vi) Verify that your numerical method conserves the total energy. You may find minor discrepancies due to numerical errors. Thus, try to do some numerical analysis to estimate the error of your method and verify if it is compatible with the discrepancies you find in the energy conservation.

vii) At the beginning you were given experimental data with uncertainties, and then fit into the Lennard-Jones potential. Now, you will plan and do the extraction of the potential yourself based on numerical realizations. Starting from the equations of motion for parameters of

your choice, simulate several motions of the atoms, for different initial conditions, and store all the positions registered. This data is what you are going to work with. Then, from that data, you should already be able to extract the potential, based on the connection between the potential and the equations of motion. *Hint:* find the accelerations from the recorded positions.

viii) Explore variations of the problem. For instance, do animations, let the atoms rotate in polar coordinates (this changes the Lagrangian), add an external magnetic field, etc.

ix) Finally, let's do some Quantum Mechanics! Consider the Lennard-Jones potential. The Hamiltonian operator for the system is given by

$$H = -\frac{\hbar^2}{2\mu}\frac{\partial^2}{\partial r^2} + V(r), \tag{I.4}$$

where $\hbar$ is the reduced Planck constant. The time-dependent Schrödinger equation for the system is hence given by

$$i\hbar\frac{\partial\psi(r,t)}{\partial t} = \left(-\frac{\hbar^2}{2\mu}\frac{\partial^2}{\partial r^2} + V(r)\right)\psi(r,t), \tag{I.5}$$

where $\psi(r,t)$ is the wave function of the system (if you are not familiar with the Schrödinger equation, you can check the literature/Wikipedia).

In quantum mechanics, everything is probabilistic, so one needs to talk about distributions and expectation values of observables (e.g. positions, momenta, etc.), rather than their exact values. The expectation value of the position is given by

$$\langle r(t)\rangle = \int_{-\infty}^{\infty} r|\psi(r,t)|^2 \mathrm{d}r, \tag{I.6}$$

where $|\psi(r,t)|^2$ is the probability density of finding the particle at position $r$ at time $t$. Your task is to solve the Schrödinger equation using the Chebyshev polynomial method, a gaussian pulse around on the "average" initial position, and compute the expectation value of the position as a function of time, using the same parameters you found in i) and $\mu = 1$. Test for values of $\hbar \in \{1, 0.1, 0.01, 0.001, 0.0001\}$[1] and see how the expectation value changes with $\hbar$. Compare with the classical case and discuss the results.

**Problem 4. Recreating phase transitions in the early universe**

In the beginning, the universe was in a highly symmetric state, with a balance of matter and anti-matter, and in a unified picture of interactions. However, as the universe evolved, these symmetries started to *break*, evolving into a different phase and eventually leading to the universe we know today. These transitions were driven by what we call *fields*, which are the fundamental entities of nature, such as the Higgs field, the electromagnetic field, etc. Mathematically, they are described functions of the position and time, $\phi(\boldsymbol{r}, t)$.

[1] The Planck's constant has a well defined value. It regulates the quantum effects in a given system. Our goal here is to explore this and take the classical limit, i.e., $\hbar \to 0$

Near a phase transition, these fields are approximately ruled by what we call a $\phi^4$ theory. This gives us the possibility of studying the phase transitions in the early universe using other simpler systems that exhibit similar phase transitions near critical points, for instance thin ferromagnetic films. Thus, we can say, at some extent, that ferromagnetic films constitute an *analogue system* of the early universe.

In a ferromagnetic system, the field that describes the system is the magnetization $m(\boldsymbol{r}) \in [-1, 1]$. For such system around the critical point, the Free Energy takes the form

$$F = \int \mathrm{d}^d x \left[ \frac{1}{2}(\nabla m)^2 + Tm^2 + \lambda(T)m^4 + hm \right], \tag{I.7}$$

where $T$ is the temperature, $\lambda(T) = \frac{T - T_c}{T}$, (what is $T_c$?) $h$ is an external magnetic field and $\nabla m$ is the gradient of the magnetization, and $d$ the dimensions of the system.

In this problem, you will use the *Hybrid Monte Carlo* (HMC) method to simulate this system. You should do a small research on the HMC method procedure (ask for references if needed!) and work in team, in order to establish and plan the methodology to attack this problem. For this method, you will need the pseudo-Hamiltonian of the system in a discretized lattice, which can be written as

$$H = a^d \sum_i \left[ \frac{1}{2}\pi_i^2 + \frac{1}{2a^2} \sum_{j,\text{neighbor}} (m_i - m_j)^2 + Tm_i^2 + \lambda m_i^4 + hm_i \right], \tag{I.8}$$

where $m_i$ is the field at site $i$, $\pi_i$ is the conjugate momentum of the field, and the summation in $j$ is done over **immediate neighbors**. The quantity $a$ corresponds to the lattice spacing, and you can set $a = 1$. The goal of this problem is to study, from simulations, the behaviour of the average value of the magnetization $\langle m \rangle$ in terms of the temperature $T$ and the magnetic field $h$, in a lattice of **2 dimensions**. Some suggestions:

i) Use the HMC method to sample evolve the system until the thermal equilibrium (constant $\langle m \rangle$ over time), as described above;

ii) Start by exploring $h = 0$ and study what happens to $\langle m \rangle$. You should be able to find a critical temperature $T_c$ where the system undergoes a phase transition.

iii) Check David Tong's Statistical Field Theory, Chapter 1, for analytical study of the Ising Model, and the CompPhysics 24/25 IST first homework. You can use this as a reference to compare your results and to understand how to build a phase diagram;

iv) Do a very detailed analysis, both from a physical and a numerical point of view.

v) Establish and divide tasks among the team members.

vi) BONUS: can you derive the discretized pseudo-Hamiltonian?

With this problem, you are doing your very first state-of-the-art *lattice gauge theory* simulation!

# II. GPU parallelization

**Problem 1. Parallelizing operations**

Implement a code in C++ that performs the following operations in parallel, both on CPU (OpenMP) and GPU (CUDA):

i) Vector addition: $c_i = a_i + b_i$;

ii) Matrix multiplication: $C_{ij} = \sum_k A_{ik} B_{kj}$;

iii) Matrix-vector multiplication: $c_i = \sum_j A_{ij} b_j$;

iv) Matrix transpose: $B_{ij} = A_{ji}$.

v) Evaluation of a function $f(x) = \sin(\cos(\log(x)))$ on a vector $x_i$.

You should compare the performance of the CPU and GPU implementations, as a function of the dimensions of the system, considering both allocation time and execution time.

**Problem 2. Parallelizing previous exercises**

In Problem 2. of the previous section, you have implemented a code which solves the Schrödinger equation in 1D using the Chebyshev expansion method. You will do an implementation of the same code, but now implemented on GPU. Use a large domain and compare the performance with the CPU implementation. Do you think you could also get a speed-up for the classical equation of motion case?

# III. Machine Learning

**Problem 1. Getting familiar with NN structures**

Use Neural Networks to fit the following functions $f$, with adequate choices of number of layers, neurons and activation functions, in a broad interval of $x$:

i) $f(x) = \sin(x)$;

ii) $f(x) = e^{-x^2}$;

iii) $f(x) = \ln(x)$;

iv) $f(x) = \Theta(x)e^{-x} - \Theta(-x)e^{x}$, where $\Theta$ is the Heaviside step function;

You should build a set of $\sim 10^3$ points, and then take $90\%$ and $10\%$ for tests. Use the mean squared error as a loss function, and train the networks using the Adam optimizer. Plot both the training and test loss. Then, create a new separate validation set and compute the ROC curve and the associated AUC.

**Problem 2. PINNs in equations of motion**

Implement a Physics-Informed Neural Network (PINN) to solve the harmonic oscillator with damping, given by the equation:

$$\frac{d^2x}{dt^2} + 2\gamma\frac{dx}{dt} + \omega^2 x = 0, \tag{III.1}$$

where $\gamma$ is the damping coefficient and $\omega$ is the frequency. First, you should generate solve the equation with standard methods, with sufficient points for a domain $t \in [0, 2]$, for some initial condition of choice. This will be your training set. Then, you should implement a PINN that learns the data and also the equation as two different sources of loss. Suggestion: before preceding with training, make sure your losses are well-balanced, (test different weights) and are adimensionally normalized.

**Problem 3. Your first surrogate model**

Consider a pendulum, described by the equation:

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin(\theta) = 0, \tag{III.2}$$

acceleration of gravity and $l$ is the length of the pendulum, You will solve this equation using a standard method, and then convert this into a Cartesian 3D frame of your choice (preferably one where all axes are populated). Your data will be a set of points $(x, y, z)$. The goal of this problem is implement a model (autoencoder) that map the (unnecessarily high-dimensional) data to a lower-dimensional 1D space (the latent space, equivalent to the space of $\theta$), which will correspond to the pendulum's angle $\theta$. Furthermore, using SINDy methods, you shall be able to reconstruct the equation of motion of the pendulum in this latent space.

Note: using ML in all these problems is naturally massive overkill, and the goal is to get familiar with the these tools, and how they can be applied to physics problems.