1 Introduction

We consider a one-dimensional coupled Allen–Cahn / Cahn–Hilliard (AC–CH) system. This system models phase separation with a conserved concentration field $c(x,t)$ and a nonconserved order parameter $\phi(x,t)$. Typical applications include solidification, binary alloy phase separation, and phase-field models for microstructure evolution.

2 Physical (dimensional) PDE system

$$\text{(Cahn–Hilliard)}: \quad \frac{\partial c}{\partial t} - 2AM\frac{\partial^2 c}{\partial x^2} + 2AM\big(C_{se} - C_{le}\big)\frac{\partial^2 h(\phi)}{\partial x^2} = 0, \tag{1}$$

$$\text{(Allen–Cahn)}: \quad \frac{\partial \phi}{\partial t} - 2AL\big[c - h(\phi)\big(C_{se} - C_{le}\big) - C_{le}\big]\,(C_{se} - C_{le})\,h'(\phi) - L\,\omega_\phi\,g'(\phi) - L\,\alpha_\phi\,\frac{\partial^2 \phi}{\partial x^2} = 0. \tag{2}$$

We define

$$h(\phi) = -2\phi^3 + 3\phi^2, \qquad g(\phi) = \phi^2(1-\phi)^2.$$

2.1 Physical parameters

Symbol	Typical units	Description
A	–	Free energy density scaling parameter.
M	length^2/time	Diffusivity (mobility) in CH equation.
L	1/time	Mobility coefficient in AC equation.
α_ϕ	energy×length^2 (or length^2)	Gradient energy coefficient for ϕ.
ω_ϕ	1/(energy) (or energy scale)	Double-well potential height for ϕ.
C_{se}	concentration (dimless)	Equilibrium concentration of solid phase (scaled).
C_{le}	concentration (dimless)	Equilibrium concentration of liquid phase (scaled).

2.2 Initial and boundary conditions (dimensional)

Initial conditions at $t = 0$:

$$\phi(x,0) = \phi_{\mathrm{ic}}(x) = \frac{1}{2}\Big(1 - \tanh\Big(\sqrt{\frac{\omega_\phi}{2\alpha_\phi}}\,(x - x_c)\Big)\Big), \qquad x_c \in [x_\ell, x_r], \tag{3}$$

$$c(x,0) = c_{\mathrm{ic}}(x) = h\big(\phi_{\mathrm{ic}}(x)\big)\,C_{se}. \tag{4}$$

Boundary conditions for all $t > 0$:

$$\phi(x_\ell,t) = 1.0, \qquad \phi(x_r,t) = 0.0, \tag{5}$$

$$c(x_\ell,t) = 1.0, \qquad c(x_r,t) = 0.0. \tag{6}$$

3 Nondimensionalization

Introduce characteristic scales l_0 (length) and t_0 (time) and nondimensional variables

$$\tilde{x} = \frac{x}{l_0}, \qquad \tilde{t} = \frac{t}{t_0}, \qquad \tilde{c}(\tilde{x},\tilde{t}) = \frac{c(x,t) - C_{le}}{C_{se} - C_{le}}, \qquad \tilde{\phi}(\tilde{x},\tilde{t}) = \phi(x,t).$$

Derivative scalings:

$$\partial_t = \frac{1}{t_0}\partial_{\tilde{t}}, \qquad \partial_x = \frac{1}{l_0}\partial_{\tilde{x}}, \qquad \partial_{xx} = \frac{1}{l_0^2}\partial_{\tilde{x}\tilde{x}}.$$

Substitute into (1) and divide by $C_{se} - C_{le}$. Define

$$P_{\mathrm{CH}} \;=\; \frac{2AMt_0}{l_0^2}.$$

The nondimensional Cahn–Hilliard equation becomes

$$\boxed{\frac{\partial \tilde{c}}{\partial \tilde{t}} - P_{\mathrm{CH}}\,\frac{\partial^2 \tilde{c}}{\partial \tilde{x}^2} + P_{\mathrm{CH}}\,\frac{\partial^2 h(\tilde{\phi})}{\partial \tilde{x}^2} = 0} \tag{7}$$

or equivalently

$$\frac{\partial \tilde{c}}{\partial \tilde{t}} = P_{\mathrm{CH}}\,\frac{\partial^2}{\partial \tilde{x}^2}\big(\tilde{c} - h(\tilde{\phi})\big).$$

Substitute into (2). Define the nondimensional groups

$$\begin{aligned} P_{\mathrm{AC1}} &= 2AL(C_{se} - C_{le})^2 t_0, \\ P_{\mathrm{AC2}} &= L\,\omega_\phi\,t_0, \\ P_{\mathrm{AC3}} &= \frac{L\,\alpha_\phi\,t_0}{l_0^2}. \end{aligned}$$

The nondimensional Allen–Cahn equation is

$$\boxed{\frac{\partial \tilde{\phi}}{\partial \tilde{t}} = P_{\mathrm{AC1}}\,\big(\tilde{c} - h(\tilde{\phi})\big)h'(\tilde{\phi}) + P_{\mathrm{AC2}}\,g'(\tilde{\phi}) + P_{\mathrm{AC3}}\,\frac{\partial^2 \tilde{\phi}}{\partial \tilde{x}^2}} \tag{8}$$

3.1 Nondimensional parameter table

Nondimensional name	Definition
P_{CH}	$\frac{2AMt_0}{l_0^2}$
P_{AC1}	$2AL(C_{se} - C_{le})^2 t_0$
P_{AC2}	$L\,\omega_\phi\,t_0$
P_{AC3}	$\frac{L\,\alpha_\phi\,t_0}{l_0^2}$

3.2 Nondimensional initial and boundary conditions

Map the physical IC/BC to nondimensional coordinates $\tilde{x}$ and fields.

Let $\tilde{x}_c = x_c/l_0$. Then

$$\tilde{\phi}(\tilde{x}, 0) = \tilde{\phi}_{\mathrm{ic}}(\tilde{x}) = \frac{1}{2}\Big(1 - \tanh\Big(\sqrt{\frac{\omega_\phi}{2\alpha_\phi}}\,l_0\,(\tilde{x} - \tilde{x}_c)\Big)\Big), \tag{9}$$

$$\tilde{c}(\tilde{x}, 0) = \tilde{c}_{\mathrm{ic}}(\tilde{x}) = \frac{h\big(\tilde{\phi}_{\mathrm{ic}}(\tilde{x})\big)\,C_{se} - C_{le}}{C_{se} - C_{le}}. \tag{10}$$

Nondimensional Dirichlet boundary conditions for all $\tilde{t} > 0$:

$$\tilde{\phi}(\tilde{x}_\ell, \tilde{t}) = 1.0, \qquad \tilde{\phi}(\tilde{x}_r, \tilde{t}) = 0.0, \tag{11}$$

$$\tilde{c}(\tilde{x}_\ell, \tilde{t}) = \frac{1.0 - C_{le}}{C_{se} - C_{le}}, \qquad \tilde{c}(\tilde{x}_r, \tilde{t}) = \frac{0.0 - C_{le}}{C_{se} - C_{le}}. \tag{12}$$

If $C_{se} = 1.0$ and $C_{le} = 0.0$ these reduce to $\tilde{c}(\tilde{x}_\ell, \tilde{t}) = 1$ and $\tilde{c}(\tilde{x}_r, \tilde{t}) = 0$.

3.3 Final compact nondimensional system

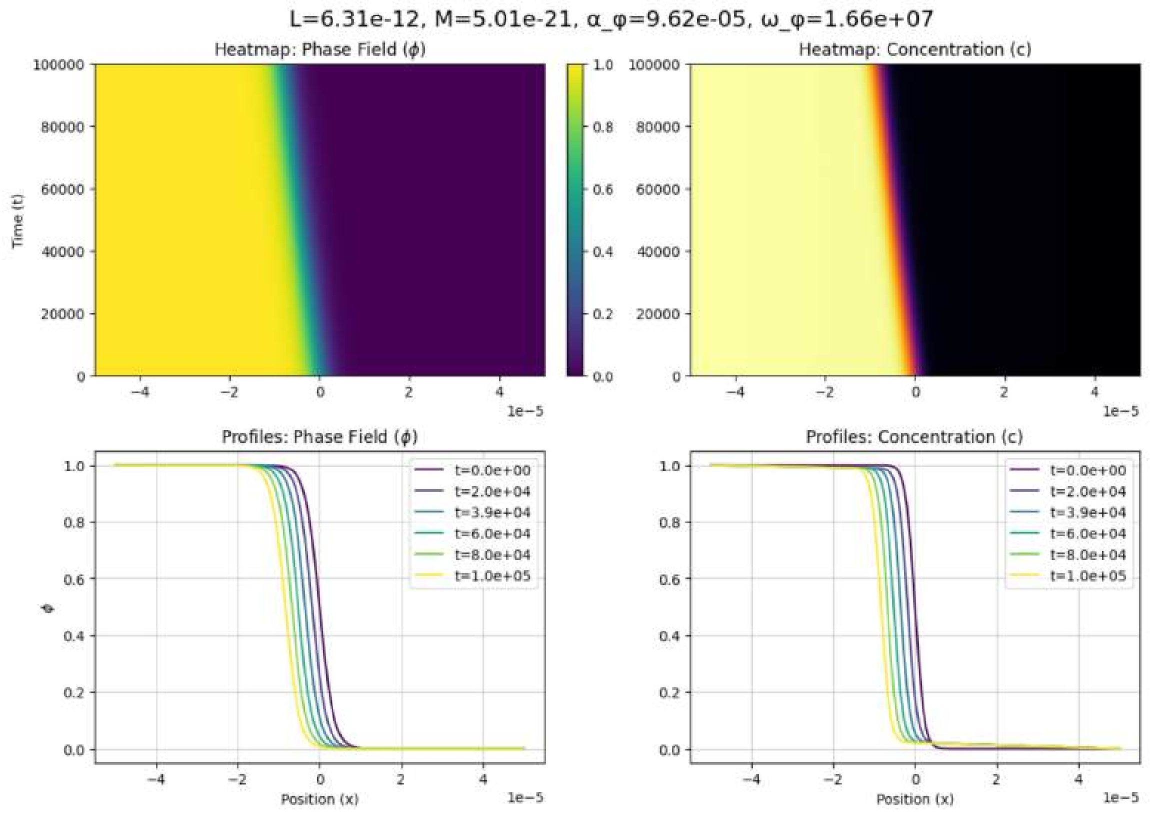
$$\frac{\partial \tilde{c}}{\partial \tilde{t}} = P_{\mathrm{CH}} \frac{\partial^2}{\partial \tilde{x}^2}\big(\tilde{c} - h(\tilde{\phi})\big), \tag{13}$$

$$\frac{\partial \tilde{\phi}}{\partial \tilde{t}} = P_{\mathrm{AC1}} \left(\tilde{c} - h(\tilde{\phi})\right) h'(\tilde{\phi}) + P_{\mathrm{AC2}}\, g'(\tilde{\phi}) + P_{\mathrm{AC3}} \frac{\partial^2 \tilde{\phi}}{\partial \tilde{x}^2}. \tag{14}$$

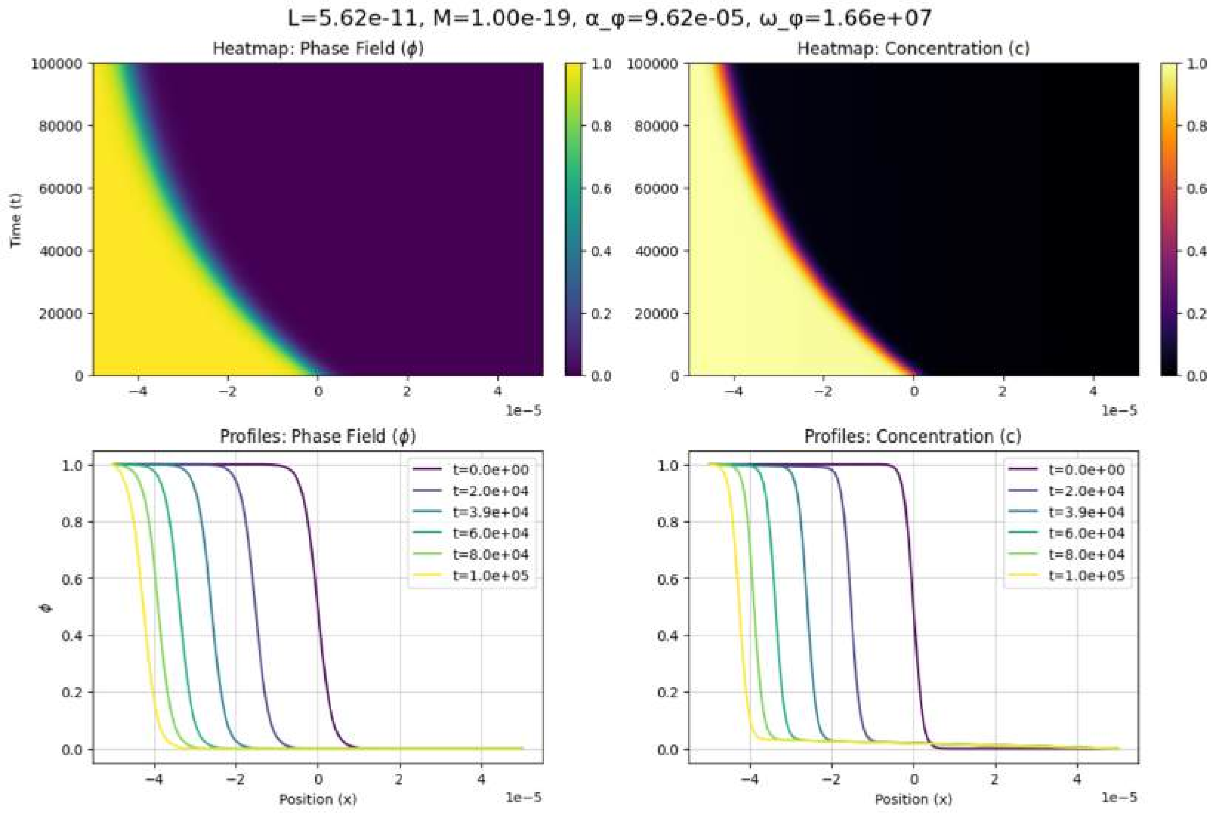
* All occurrences of $h(\cdot)$ and $g(\cdot)$ keep the same algebraic form after nondimensionalization because $\tilde{\phi} = \phi$.

4 PDE solutions/property

From now on, unless otherwise specified, we will examine the pde behaviour in the non-dimensional system unless otherwise specified. Here, we examine the PDE solutions over time



The heatmaps on the top shows how the solution evolves over the spatial-temporal domain. The spatial domain (x-axis) is in meters and the temporal domain (y-axis) is in seconds. The plots on the bottom shows the cross-section of the heatmap at several time frames, i.e. the plot of $\phi(x, t_i)$ and $c(x, t_i)$ at several t_i. It can be noticed that both $\phi(x, t)$ and $c(x, t)$ adapt to a hyperbolic tangent looking curve and is shifting towards the left (due to corrosion) at a relatively constant speed.



This is another visualization of the plot where we changed the mobility parameter L from $6.31e-12$ to $5.62e-11$ and changed the diffusivity parameter M from $5.01e-21$ to $1.00e-19$. By changing the parameters, we observe some modest change in the solution behaviour; the curves are now shifting towards the left at a non-uniform, decreasing rate.

5 Solving the PDE using physics informed neural network

5.1 fixed parameter PINN

Traditionally, PDEs are solved using method of lines (MOL), finite element methods (FEM), etc. Here, we consider am emerging approach in recent years to solve the problem, namely, the physics informed neural network (PINN).

TODO: This section should talk about what a PINN is, and how it is being used in solving PDEs (and why it's effective). We will further demonstrate how it can be used here to solve the 1D coupled ACCH equation and be able to make relative decent predictions. We should further note that by adding the "data loss" to minimize, we can significantly improve the accuracy. (note the "data loss" refers to the mean square error between prediction vs true solutions)

5.2 parameterized PINN

parameterized PINN is an extension of PINN where the model now accepts pde parameters and make prediction of the parameterized PDE. The training of parameterized PINN is extremely hard without guidance of true solutions. Therefore, we pre-compute the solutions and examine how model performs subject to using different number of solutions during training.

5.2.1 Training goal

Here, we train a neural network that takes 2 pde parameters, they are respectively the mobility parameter L and the diffusivity parameter M. The range of L goes from $1e-12$ to $1e-10$ and the range of M goes from $1e-21$ to $1e-19$. Note that despite the model takes physical parameters, it is trained using the solution of the dimensionless version of the problem and will output the result of the non-dimensionalized PDE.

5.2.2 Neural network design

We use a fully connected neural network that takes in 4 inputs (x, t, L, M) and produces 2 outputs (ϕ, c). In between, the network has 4 layers and each layer contains 64 neurons.

5.2.3 Training

The training of neural network involves minimizing a loss function using gradient descent and backpropagaion. The loss function used here is a weighted sum of 5 loss terms. They are respectively

- IC loss: the loss associated with initial conditions
- BC loss: the loss associated with the boundary conditions
- Colloc/PDE loss: the residual of the PDE, computed using automatic differentiation. Given we have two differential equations to satisfy, we split this loss into
  - AC loss: the residual of Allan-Cahn equation, the residual for ϕ
  - CH loss: Cahn-Hilliard equation, the residual for c
- Data loss: the discrepancy between the prediction of neural network $u_{nn}(x,t) = (\phi_{nn}(x,t), c_{nn}(x,t))$ and the ground truth $u(x,t) = (\phi(x,t), c(x,t))$. The details of data generation are explained in Section Pre-computed data.

Since the losses vary across several orders of magnitude, we need to balance the 5 loss terms. To do so, we use neural tangent kernel (NTK) weights. **TODO:** Explain in detail what a NTK is (both intuitively and mathematically), then explain how it can be used to weight each loss term.

5.2.4 Metric

It is important to keep track of the best model. We cannot directly use the weighted losses as the NTK weight could make the total loss increase while the actual loss decreases. Summing the losses is also problematic due to the varying magnitudes. We consider two metrics:

- Data loss: the unweighted data loss.
- Geometric mean loss: the geometric mean of the unweighted losses.

Throughout training, we track the parameters that yield the lowest data loss and geometric mean loss.

5.2.5 Pre-computed data

To generate data used for training, we sample the parameters L and M using a grid-based approach. Specifically, we define a set of evenly spaced values for L and M using `linspace`, and then form a meshgrid of all possible combinations. We then solve the PDEs for each parameter combination on a spatial grid of 128 uniformly spaced points and a temporal grid of 128 uniformly spaced points, covering the full spatiotemporal domain.

By doing so, we ensured a uniform coverage of the parameter space and have successfully trained a neural network that makes reasonable predicionts. However, it is impossible to generalize this grid-based approach in selecting parameters to a higher dimensional parameter space due to the exponential increase in the number of samples. Effectively, if we have 4 parameters of interest $(L, M, \alpha_\phi, \omega_\phi)$ and we select 10 values from each to form a meshgrid, then we will have to solve $10^4 = 10000$ PDEs and train using 10000 datasets. However, this is computational heavy and barely covers the parameter space.

Instead, we observed that the convergence of neural network may have less to do with the number of datasets. Rather, it seem to depend on dataset to cover enough PDE solutions that behave differently. Refer to PDE solutions/property. If most of the pre-computed dataset covers the solution that shifts at a constant speed and are more squished together, then it'll perform poorly when predicting the PDEs with solutions that shifts at a non-constant speed. This is not surprising as the model's convergence is guided by the data, and sacrificing some performance in the rarely occurring scenario could lead to an overall gain. However, if the training set are composed of PDE solutions that are "diverse" (diverse as in PDE solution behaviours), then they can potentially guide the model to explore a more complex landscape where a better overall solutions lies.

This suggest the use of Quasi-Monte Carlo (QMC) sampling to sample parameters. In particular, if the hypothesis that the convergence of model relies more on the diversity of the solution then the number of datasets. Then using QMC sampling might still yield a training set that has the diversity needed but with a far smaller training set. **TODO:** Note the solution behaviours does not depend on L and M in a uniform sense, We'll cover that later. However, for now, we'll examine how much could QMC sampling improves the training result if it ever does.

5.2.6 Experiment set up

To examine this, we train the parameterized PINN using training set generated by grid-based approach and data set generated using QMC-halton. For the grid-based approach, we will use the following spacing schemes: $\{3 \times 3, 5 \times 5, 7 \times 7, 10 \times 10, 15 \times 15, 20 \times 20\}$. For the QMC approach, we simply generate parameters of the same size using QMC-halton algorithm.

For each training loop, we'll obtain two "best" models using metric introduced in Metric. As a remainder here, they are the data loss and geometric mean loss.

In the end, we will have constructed 24 models and the naming pattern is {metric}_{method}_{size} where

- metric is one of "data","loss"
- method is one of "grid","halton"
- size is one of {3,5,7,10,15,20}

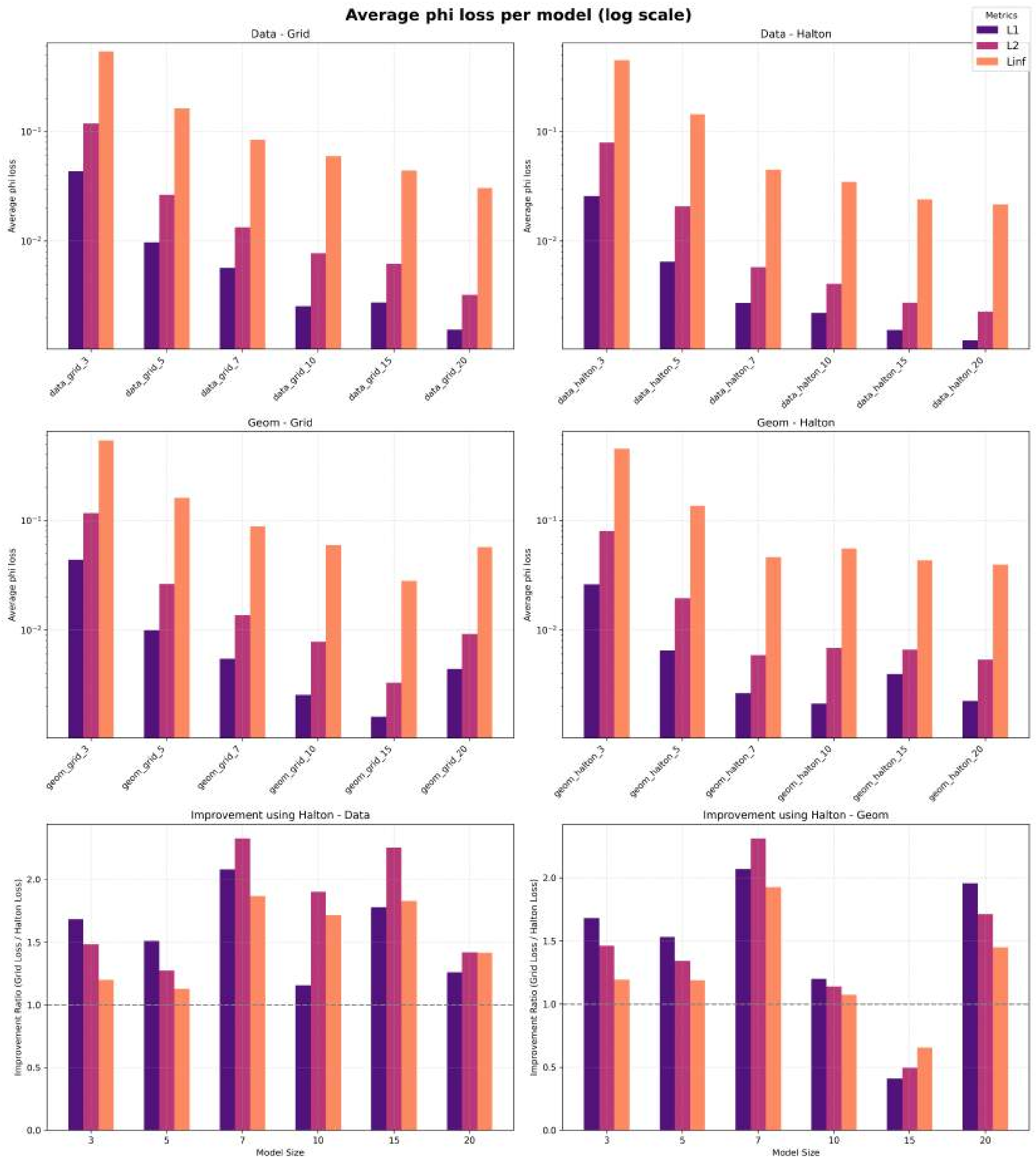
* Note if the two model names only differed by "metric" then they are the model obtained in the same training loop, just using different metrics.

5.2.7 Results

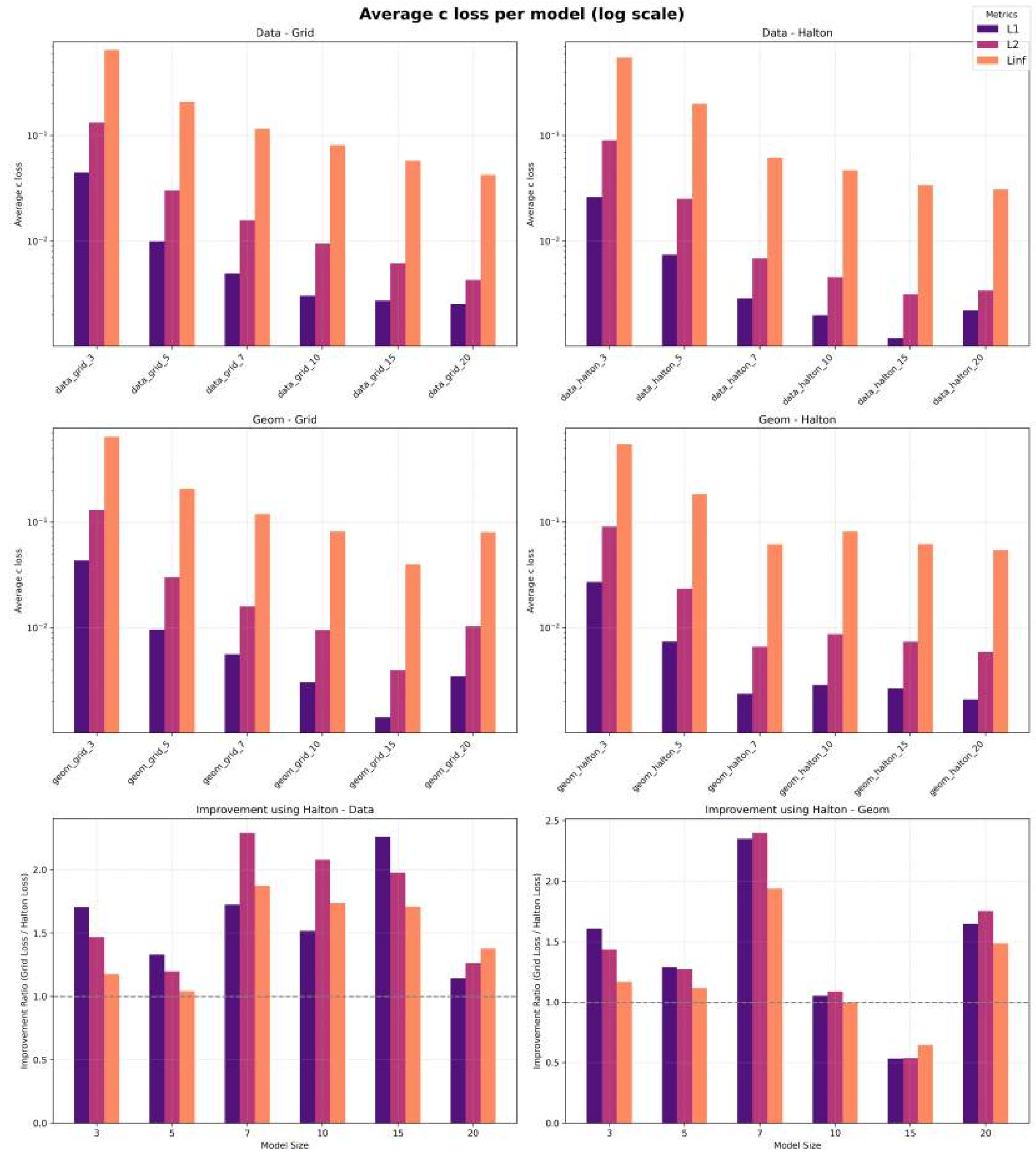
Before getting into the results, we'll explain how to quantify the model's performance. To evaluate the model's performance, we generated 80000 set of parameters 40000 set of parameters are generated using the grid-based approach, that is, a 200×200 uniform grid. Another 40000 set of parameters are generated using the QMC-halton approach. For each set of the parameter, we solved the corresponding PDE to obtain the values of ϕ and c at the 128×128 spatial-temporal coordinates.

Then after each model has been trained, we ask each model to make predictions over the 80000 set of parameters at the 128×128 spatial-temporal coordinates where we have obtained the solutions. This allows us to compute the L_1, L_2, L_∞ losses between the prediction and the ground truth of ϕ and c ofor the 80000 set of parameters.

Once we finished the above computation, for each model we have 80000 L_1 losses for ϕ, 80000 L_1 losses for c, 80000 L_2 losses for ϕ, 80000 L_2 losses for c, 80000 L_∞ losses for ϕ and 80000 L_∞ losses for c. To summarize the overall performance, we simply compute the mean L_1, L_2, L_∞ losses for ϕ and c for each model. Below are the results:

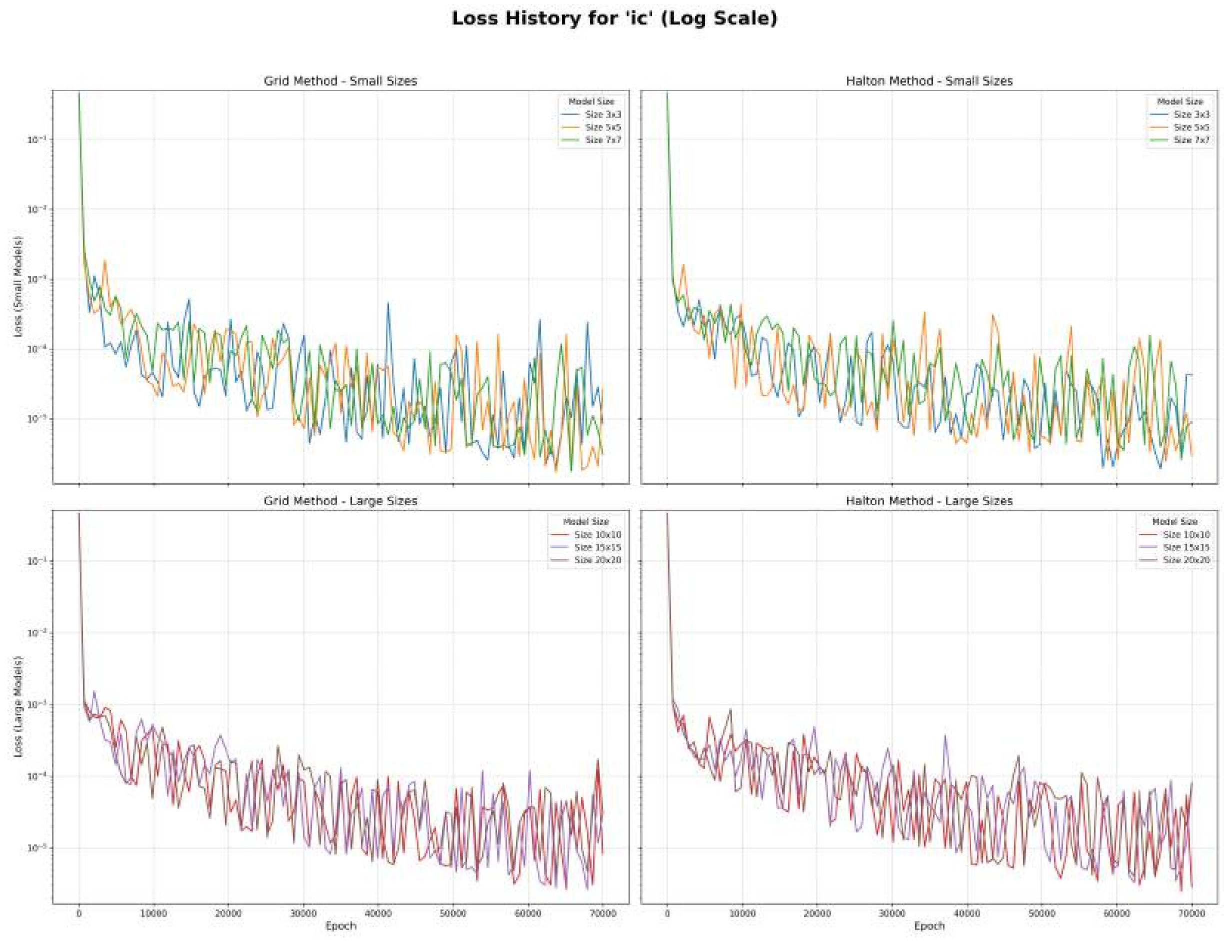


The first two rows shows the mean error plots (on the log scale) of the 24 models. The last row shows the ratio between the loss using QMC method vs the loss using grid-based approach. It is a simple division of the mean L_1, L_2, L_∞ loss. If it's over 1, it means improvements over grid-based method. If it's below 1, it means Degradation over grid-based method.

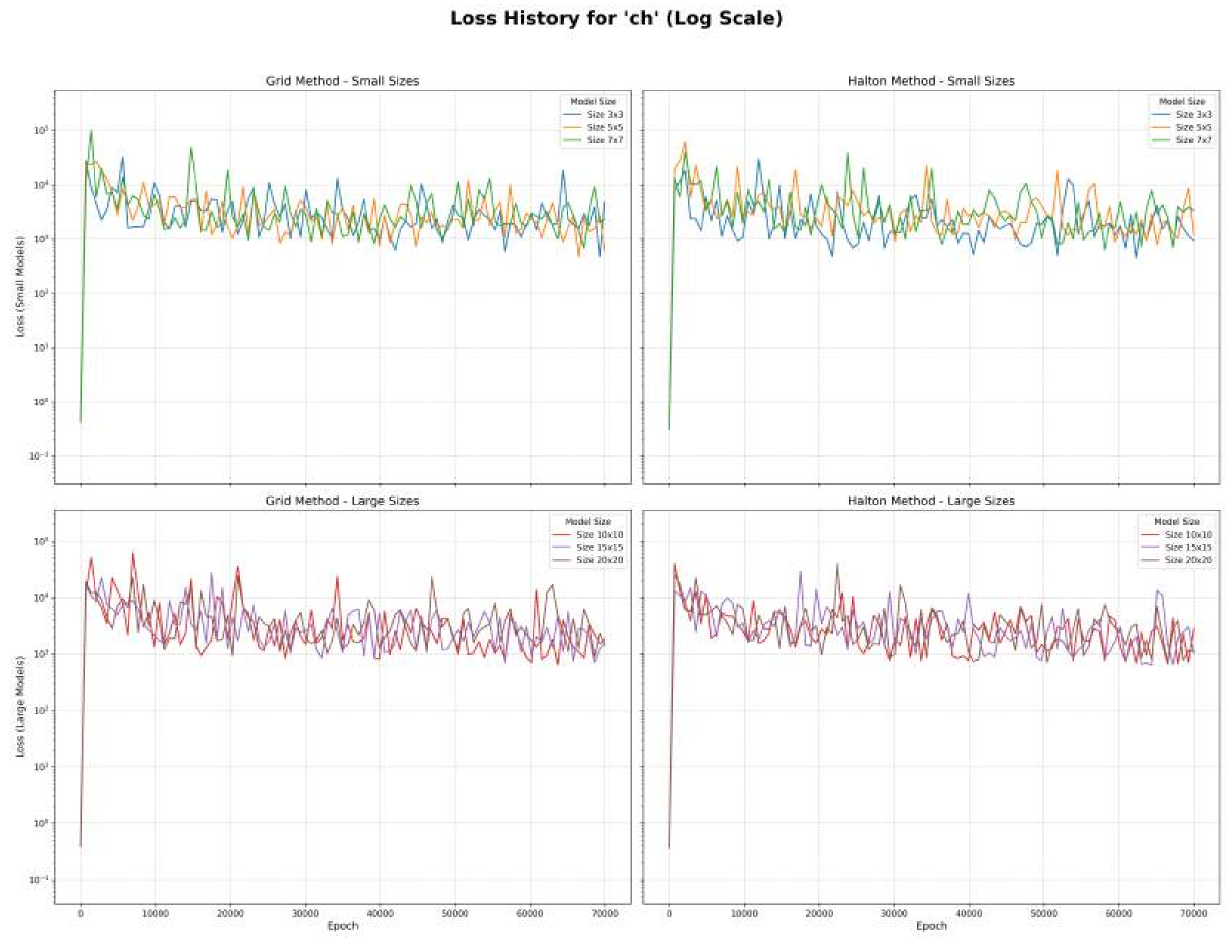


This is the model's performance when predicting c.

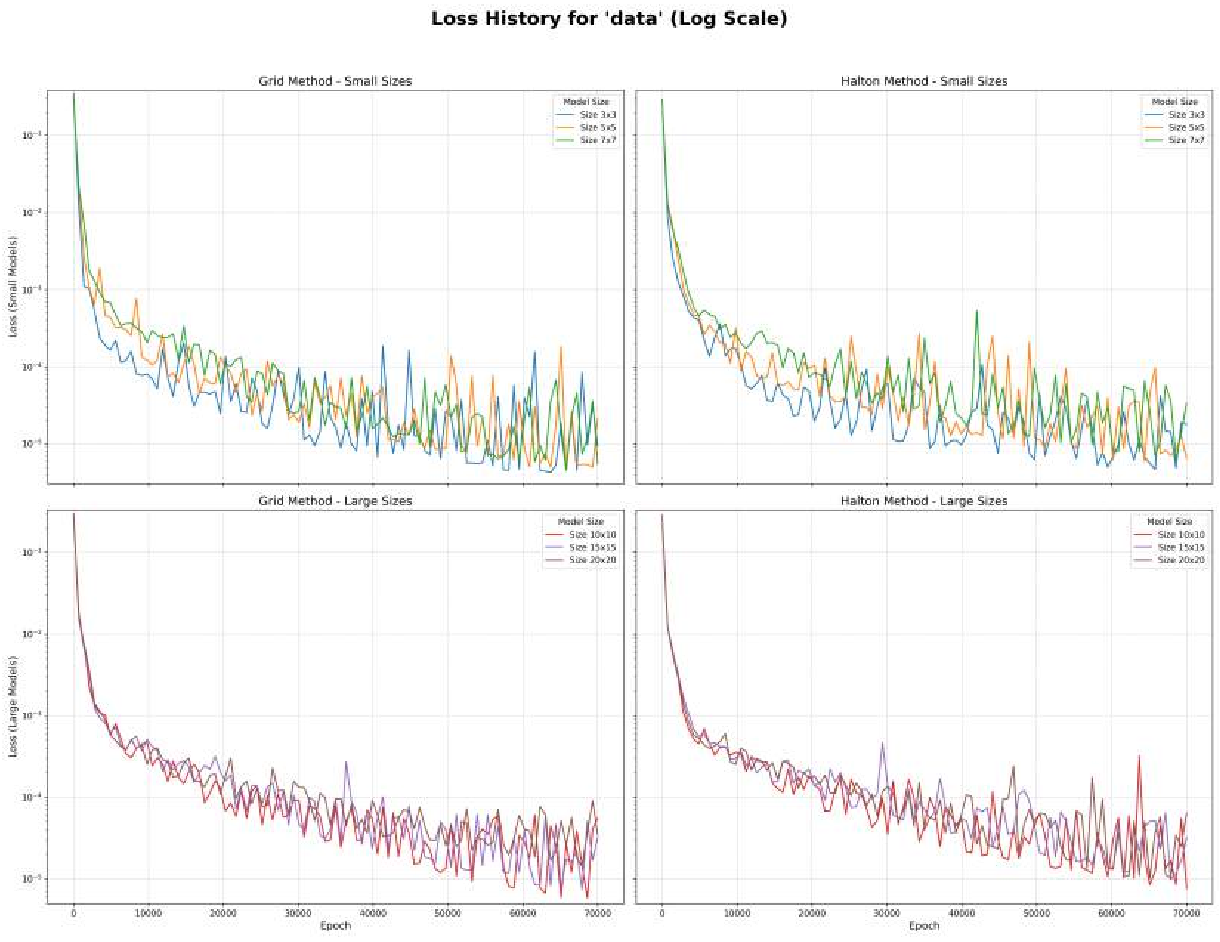
Here, we present the loss history for the ic/ch/data term, the rest plots can be found on the github repo



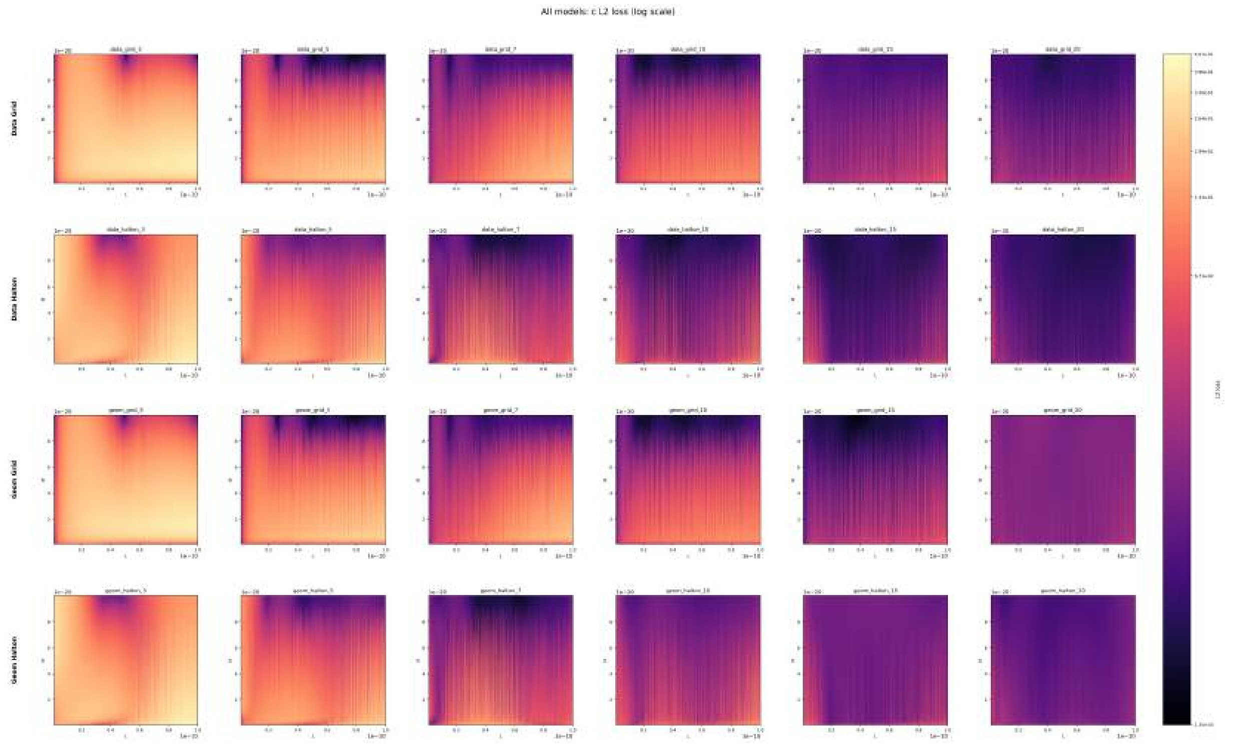
Here, we have plotted the loss history of the initial condition term during training. The top plot contains the loss history of using $3 \times 3, 5 \times 5, 7 \times 7$ data sets (generated using either grid or qmc), the bottom plots are similar but contains the loss history of using $10 \times 10, 15 \times 15, 20 \times 20$ data sets. The losses are plotted on a log scale.



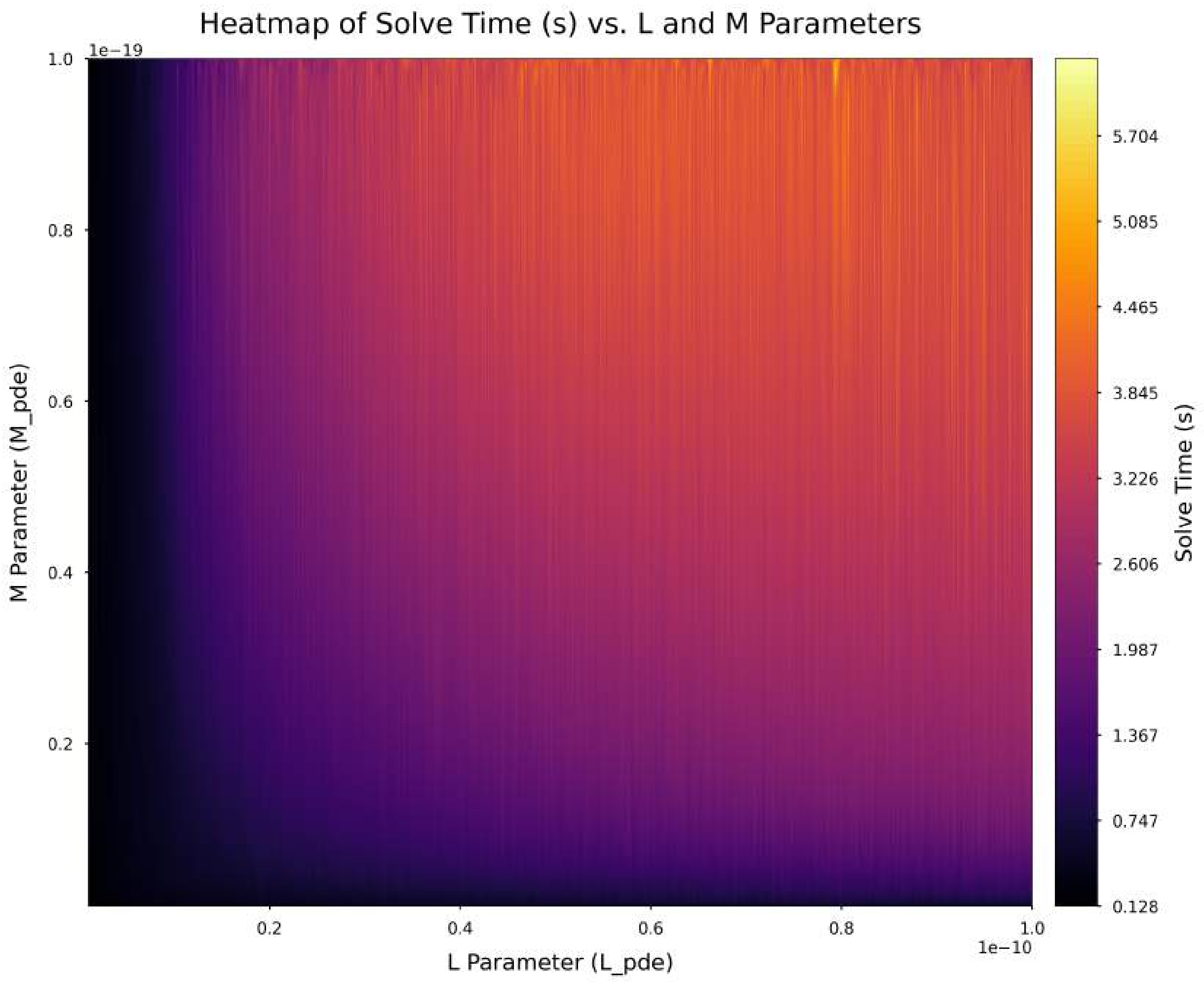
The structure of the plot is the same, however, one may notice that the losses are very high. This is because non-dimensionlization of the PDE will amplify the residual for the Cahn-Hilliard equation by a very huge value. Therefore, the losses looks huge. However, it does not mean the model failed to converge.



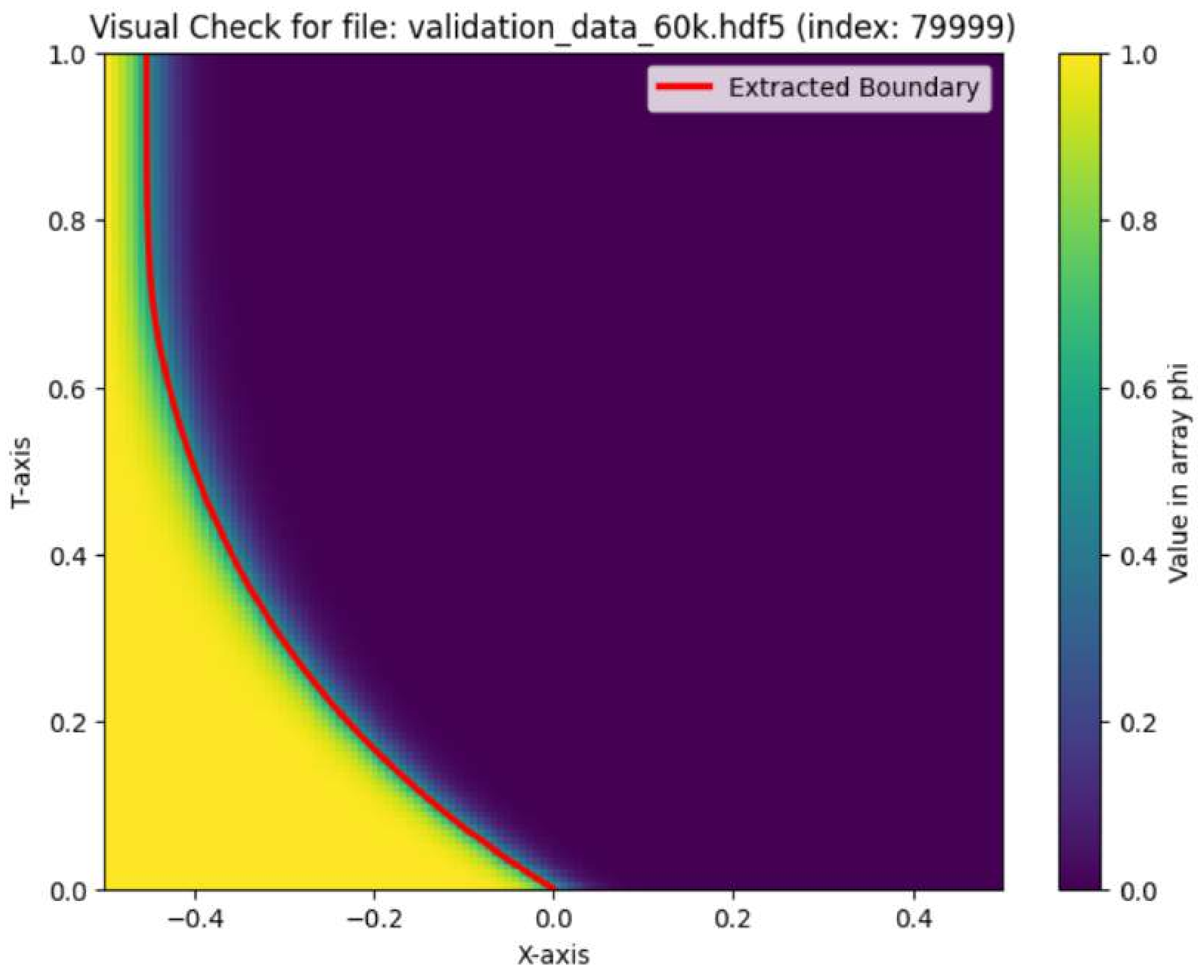
Here we provide the data loss (data loss over the training set)



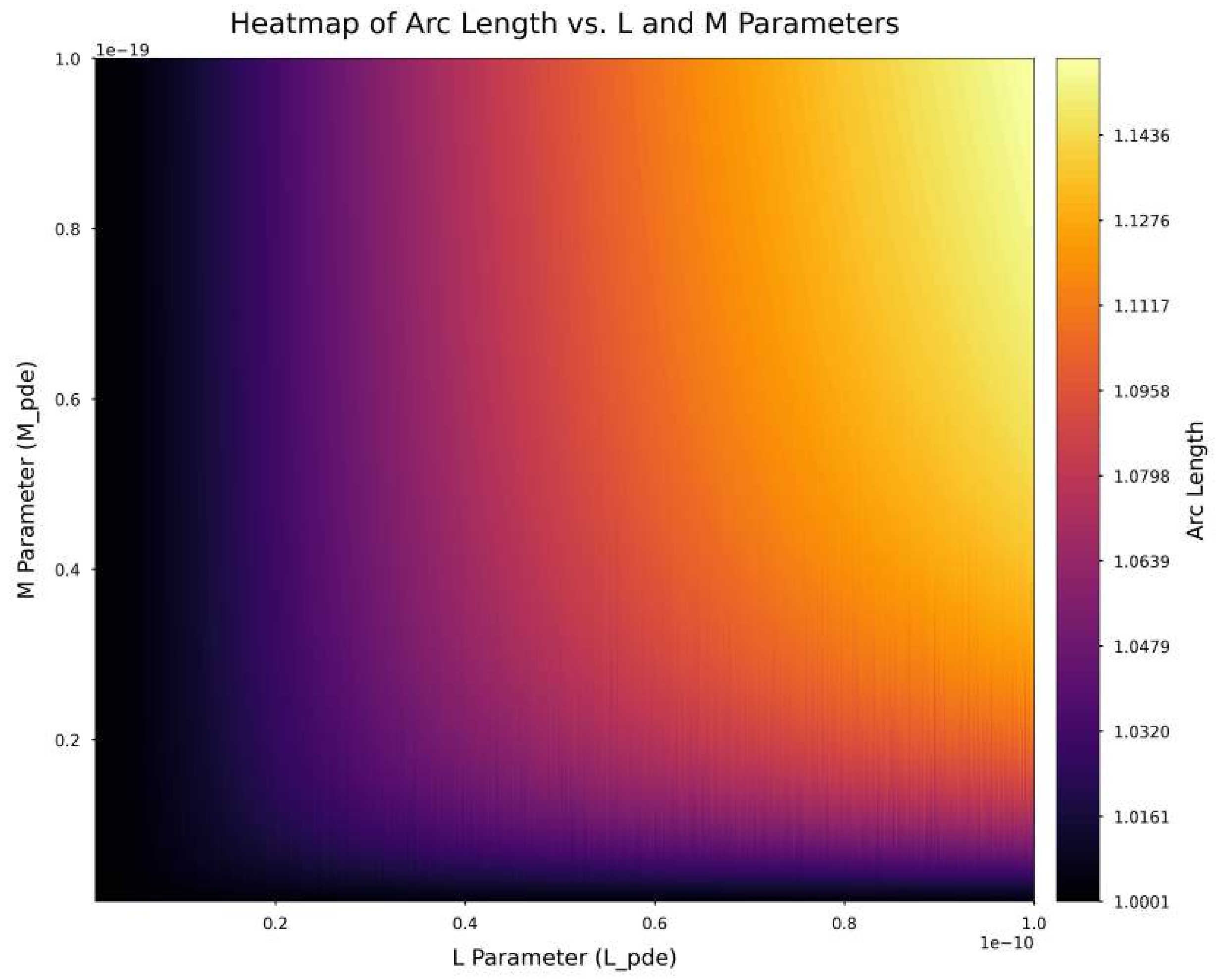
We further constructed a loss heatmap. For each heatmap, the x-axis is L, the y-axis is M, the value at each coordinate is the L_2 loss (on a log scale). The first row contains 6 models obtained using metric "data" with training set generated using grid-based approach with grid size being $3, 5, 7, 10, 15, 20$. The heatmaps on the second row follows the similar structure but for data-halton. The third row is geom-grid, the last row is geom-halton. **TODO:** the label are unexpectedly small and I'll fix later. Please see github repo for the original plot. I've also created the heatmap for L_1, L_∞ losses as well but the general pattern is the same. For illustration purpose, we'll present only one plot here



Here, we try to quantify the property of the solutions by the solve time. The x-axis is L, the y-axis is M. the value at each coordinate is the time needed to solve the problem, which reflects the stiffness of the problem to a certain extent.



Here, we propose another way to quantify the solutions by the interface (interface of corrosion of the material, it is indicated by the red line). For example, we can quantify the solution by the arc length of the red line or by it's curvature.



This is the heatmap of the arc length for different L and M. As we can see, the great the L and M, the greater the arc length. This is expected as increasing the mobility and diffusivity should lead to some more intense corrison so the interface quickly decays towards the left. **TODO:** I've also done the computation of the curvature and it's pattern is similar to the arc length (as expected)