

Assignment 10

314652021 羅瑋翔

Q1.

Consider a forward SDE

$$dx_t = f(x_t, t)dt + g(x_t, t)dW_t$$

show that the corresponding PF ODE is written as

$$\frac{\partial p}{\partial t} = \left[f(x_t, t) - \frac{1}{2} \frac{\partial}{\partial x} g^2(x_t, t) - \frac{g^2(x_t, t)}{2} \frac{\partial}{\partial x} \log p(x_t, t) \right] p$$

Proof:

Consider a forward SDE

$$dx_t = f(x_t, t)dt + g(x_t, t)dW_t$$

with corresponding Fokker-Planck equation

$$\frac{\partial p}{\partial t} = -\frac{\partial}{\partial x}(fp) + \frac{1}{2} \frac{\partial^2}{\partial x^2}(g^2 p).$$

We can rewrite the Fokker-Planck equation as

$$\begin{aligned} \frac{\partial p}{\partial t} &= -\frac{\partial}{\partial x} fp + \frac{1}{2} \frac{\partial^2}{\partial x^2} g^2 p \\ &= \frac{\partial}{\partial x} \left(-fp + \frac{1}{2} \left(p \frac{\partial}{\partial x} g^2 + g^2 p \frac{\partial}{\partial x} \log p \right) \right) \end{aligned}$$

Then rewrite the FPE as continuity equation:

$$\frac{\partial p}{\partial t} = -\frac{\partial}{\partial x}(\tilde{f}p)$$

Following, replace the term $\partial_t p$ with $-\partial_x \tilde{f} p$ and integrating both sides with respect to x

$$-\tilde{f}p = -fp + \frac{p}{2} \frac{\partial}{\partial x} g^2 + \frac{g^2 p}{2} \frac{\partial}{\partial x} \log p$$

And deviding both sides by $-p$, we can obtain the result

$$\tilde{f} = f - \frac{1}{2} \frac{\partial}{\partial x} g^2 - \frac{g^2 p}{2} \frac{\partial}{\partial x} \log p$$

Therefore, the PF-ODE is:

$$dx_t = [f(x_t, t) - \frac{1}{2} \frac{\partial}{\partial x} g^2(x_t, t) - \frac{g^2(x_t, t)}{2} \frac{\partial}{\partial x} \log p(x_t, t)] dt$$

Q2.

1. A future AI capability

I think the fully automated chip physical behavior simulation and design optimization will come true.

The primary challenge can be partitioned into:

- Scientific/ Modeling Challenges: Multi-scale coupling complexity, difficult embedding physical constraints...
- Technical/ computational Challenges: high-fidelity simulation cost, hardware limitations...

I think the most important challenge is the difficulty of embedding physical constraints when modeling the chip's behavior.

If we can overcome this, we could simulate the chip and predict which regions might overheat. Since excessively high local temperatures can cause thermal throttling, we could then adjust the layout or power distribution to ensure the chip operates at high performance without increasing the clock time.

With this improvement, hardware limitations could also be alleviated, since the main constraints in IC design are becoming more pronounced due to the slowing of Moore's law.

As a result, we could obtain more efficient processors, reducing the time cost of simulations.

2. Involved machine learning types

To implement it,

- Supervised learning: To learn the mapping between chip design parameters and local physical behaviors such as temperature, current desity, or quantum states.
- Reinforcement learning: To automatically optimize chip design strategies, such as component layout or power distribution.
- Unsupervised learning: Extract latent patterns in chip structure and physical behaviors(finding low-dimensional representations of heart flux, current, or quantum states from high-dimensional input data).
- Physics-informed learning: Embedding PDEs or physical constraints directly into NN(Hard-PINN)

3. The first step of Modelization

Consider the energy conservation:

$$\sum_i C_i \frac{d\tilde{T}_i}{dt} = \sum_i P_i + \text{heat flux}$$

Notation:

- \tilde{T}_i : The predicted temperature of node i
- C_i : The heat capacity of node i
- $\frac{d\tilde{T}_i}{dt}$: The change of temperature of node i overtime.
- P_i : The power dissipated of i-th node.
- heat flux: The heat flux pass through the boundary

Since the boundary conditions is complicated and non-linear, stiff multi-physics system, gradient vanishing.

I want to implement Hard-PINN to improve these.

Because Hard-PINN can be written as:

$$u(x) = g(x) + u_{\theta}(x) \cdot f(x)$$

where $u(x)$ is the true solution, $g(x)$ satisfies the boundary conditions, $u_{\theta}(x) \cdot f(x)$ not contribute to BCs.

Hard-PINN embeds the boundary or initial conditions directly into the neural network architecture, rather than enforcing them through the loss function. This eliminates the need for boundary-loss balancing and can partially stabilize the training process, although the PDE residual loss may still remain highly non-convex.

Then the main challege here is to find $g(x)$, and $f(x)$.

I would use stereographic projection to project the whole neural network and the PDE(heat flux) to the sphere of a unit ball.

If the chip surface or module geometry is mapped onto a sphere (θ, ϕ) , the boundary conditions can be represented using spherical harmonics $Y_l^m(\theta, \phi)$:

$$g(\theta, \phi) = \sum_{l=0}^L \sum_{m=-l}^l c_l^m Y_l^m(\theta, \phi)$$

where $c_l^m = \int_{s^2} T_b(\theta, \phi) \overline{Y_l^m(\theta, \phi)} d\Omega$.

And $f(\theta, \phi)$ is also easy to find.

Then, improve the PDE constraints is the most important part.

We could induces **adaptive loss balancing**:

$$L_{PDE} = \sum_{i=1}^N w_i |r_i|^2$$

where w_i is the adaptive weight to balance the PDE loss of multi-scale PDE loss.

And induces the **conservation-aware term** to force the energy conservation hold.

$$L_{cons} = (\oint_{\partial\Omega} q_{\theta} \cdot dS - Q_{total})^2$$

- $q_{\theta} = -k\nabla u_{\theta}$ is the heat flow or energy flow.
- Q_{total} : The total energy input/output of the system.

Then the new loss function is define as:

$$L_{\theta} = \sum_{i=1}^N w_i |r_i|^2 + \lambda (\oint_{\partial\Omega} q_{\theta} \cdot dS - Q_{total})^2$$

Unanswered Questions

I wondered is there any efficient way to decide the weight w_i of image interpolation with PF-ODE, where $\sum_i w_i = 1$?

Since we have no idea of the distribution of the initial state x_0^i , $1 \leq i \leq n$.

So, deciding weight to preserve the characteristic of the image will cost a lot of time while doing the iterations.

Reference:

Stereographic projection:

<https://ncatlab.org/nlab/show/stereographic+projection>

(<https://ncatlab.org/nlab/show/stereographic+projection>).

PINNs: <https://arxiv.org/html/2403.00599v1> (<https://arxiv.org/html/2403.00599v1>).