Physics-informed neural networks for solving FRG

Takeru Yokota (Osaka Institute of Technology, RIKEN iTHEMS)

Refs. [1] TY, Phys. Rev. B 109, 214205 (2024)
[2] Miyagawa, TY, NeurIPS2024 (2024)
[3] TY, in prep

13th International Conference on the Exact Renormalization Group 2026 (ERG2026),
University of Sussex, Brighton, UK, 4th September 2026

Machine Learning ⟷ Physics

A virtuous cycle driving discovery and innovation

$$\mathcal{L} = \frac{1}{2}\left(\partial_\mu \phi \partial^\mu \phi - m^2 \phi^2 - \frac{\lambda}{4!}\phi^4\right)$$

New Algorithms & Models

Inspired by physical principles

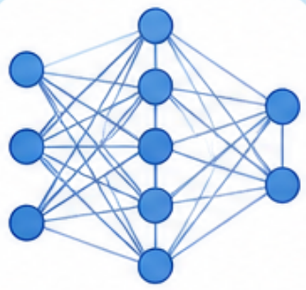
ML for Physics

Physical Principles & Constraints

Guiding models for better generalization

Efficient Computation

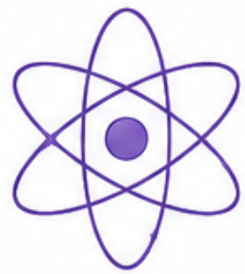
Accelerating simulations and optimization



Machine Learning

Stronger Together

Physics



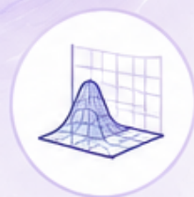
New Phenomena & Data

Providing rich problems and benchmarks

Data-Driven Discovery

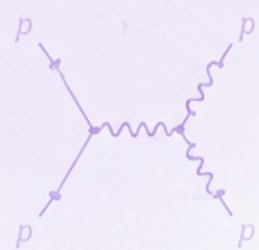
Extracting knowledge from complex data

Physics for ML



Theoretical Insights

Deepening our understanding of learning systems



Uniting data, models, and physical understanding to solve real-world challenges.

Generated with ChatGPT

Machine Learning ⟷ Physics

A virtuous cycle driving discovery and innovation

$$\mathcal{L} = \frac{1}{2}\left(\partial_\mu \phi \partial^\mu \phi - m^2 \phi^2 - \frac{\lambda}{4!}\phi^4\right)$$

ML for solving FRG eqs.

ML for Physics

New Algorithms & Models

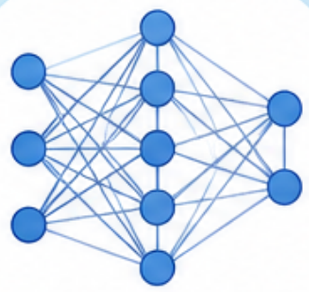
Inspired by physical principles

Physical Principles & Constraints

Guiding models for better generalization

Efficient Computation

Accelerating simulations and optimization



Machine Learning

Stronger Together

Functional renormalization group (FRG)

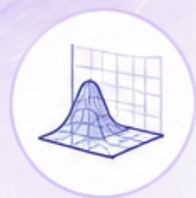
New Phenomena & Data

Providing rich problems and benchmarks

Data-Driven Discovery

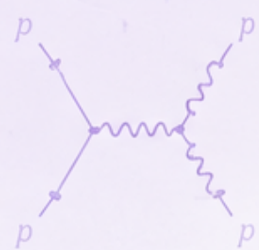
Extracting knowledge from complex data

Physics for ML



Theoretical Insights

Deepening our understanding of learning systems



Uniting data, models, and physical understanding to solve real-world challenges.

Generated with ChatGPT

Contents

1. Physics-informed neural networks (PINNs) for FRG
   - Motivation & method
   - Network architecture
2. Demonstration in a toy model
   - Calculation w/ multi-layer perceptron
   - Calculation w/ input-convex neural network

Contents

1. Physics-informed neural networks (PINNs) for FRG
    - Motivation & method
    - Network architecture
2. Demonstration in a toy model
    - Calculation w/ multi-layer perceptron
    - Calculation w/ input-convex neural network

Why is solving FRG eqs. difficult?

Wetterich eq. $$\partial_k \Gamma_k[\varphi] = \frac{1}{2}\mathrm{Tr}\left[\partial_k R_k \cdot \left(\frac{\delta^2 \Gamma_k[\varphi]}{\delta\varphi\delta\varphi} + R_k\right)^{-1}\right]$$

- Functional diff. eq. (FDE) = Infinite-dim. PDE
  - A function ($\varphi(x)$) = An infinite-dim. vector

c.f.) Basis function expansion $\varphi(x) = \varphi_1 e_1(x) + \varphi_2 e_2(x) + \cdots$

Why is solving FRG eqs. difficult?

Wetterich eq. $\partial_k \Gamma_k[\varphi] = \frac{1}{2}\mathrm{Tr}\left[\partial_k R_k \cdot \left(\frac{\delta^2 \Gamma_k[\varphi]}{\delta\varphi\delta\varphi} + R_k\right)^{-1}\right]$

- Functional diff. eq. (FDE) = Infinite-dim. PDE
  - A function ($\varphi(x)$) = An infinite-dim. vector

    c.f.) Basis function expansion $\varphi(x) = \varphi_1 e_1(x) + \varphi_2 e_2(x) + \cdots$
- Reduction to finite DOF ($\varphi(x) \to \boldsymbol{\varphi} = (\varphi_1, \ldots, \varphi_M)$)
  - Basis function expansion: $\varphi(x) \approx \varphi_1 e_1(x) + \cdots + \varphi_M e_M(x)$
  - Finite-volume lattice: $\varphi(x) \to \{\varphi_n\}_{n=1}^{M}$

$\Rightarrow$ FDE $\approx$ High-dim. PDE (HDPDE)

$$\partial_k \Gamma_k(\boldsymbol{\varphi}) = \frac{1}{2}\mathrm{tr}\left[\partial_k R_k \cdot \left(\mathrm{Hess}_{\boldsymbol{\varphi}}\left(\Gamma_k(\boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})\right)\right)^{-1}\right]$$

- Hessian: $\mathrm{Hess}_{\boldsymbol{\varphi}}(\cdots) = \frac{\partial^2}{\partial\boldsymbol{\varphi}\partial\boldsymbol{\varphi}}(\cdots)$
- Regulator term: $\Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi}) = \frac{1}{2}\boldsymbol{\varphi} \cdot R_k \cdot \boldsymbol{\varphi}$

Why is solving FRG eqs. difficult?

Wetterich eq. $\partial_k \Gamma_k[\varphi] = \frac{1}{2}\mathrm{Tr}\left[\partial_k R_k \cdot \left(\frac{\delta^2 \Gamma_k[\varphi]}{\delta\varphi\delta\varphi} + R_k\right)^{-1}\right]$

- Functional diff. eq. (FDE) = Infinite-dim. PDE
  - A function ($\varphi(x)$) = An infinite-dim. vector

    c.f.) Basis function expansion $\varphi(x) = \varphi_1 e_1(x) + \varphi_2 e_2(x) + \cdots$
- Reduction to finite DOF ($\varphi(x) \to \boldsymbol{\varphi} = (\varphi_1, \ldots, \varphi_M)$)
  - Basis function expansion: $\varphi(x) \approx \varphi_1 e_1(x) + \cdots + \varphi_M e_M(x)$
  - Finite-volume lattice: $\varphi(x) \to \{\varphi_n\}_{n=1}^{M}$

$\Rightarrow$ FDE $\approx$ High-dim. PDE (HDPDE)

$$\partial_k \Gamma_k(\boldsymbol{\varphi}) = \frac{1}{2}\mathrm{tr}\left[\partial_k R_k \cdot \left(\mathrm{Hess}_{\boldsymbol{\varphi}}\left(\Gamma_k(\boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})\right)\right)^{-1}\right]$$

- Hessian: $\mathrm{Hess}_{\boldsymbol{\varphi}}(\cdots) = \frac{\partial^2}{\partial\boldsymbol{\varphi}\partial\boldsymbol{\varphi}}(\cdots)$
- Regulator term: $\Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi}) = \frac{1}{2}\boldsymbol{\varphi} \cdot R_k \cdot \boldsymbol{\varphi}$

Solving FRG $\approx$ Solving HDPDE

Why is solving FRG eqs. difficult?

Numerically tough!

- Direct discretization of the input φ suffers from the curse of dimensionality (#grid points $\sim \exp(cM)$)
- Conventional methods: Reduction of PDE by vertex expansion, derivative expansion…
  - Truncation of form of Γ_k based on Taylor expansion

Other possible approaches that retain richer form of $\Gamma_k(\varphi)$?

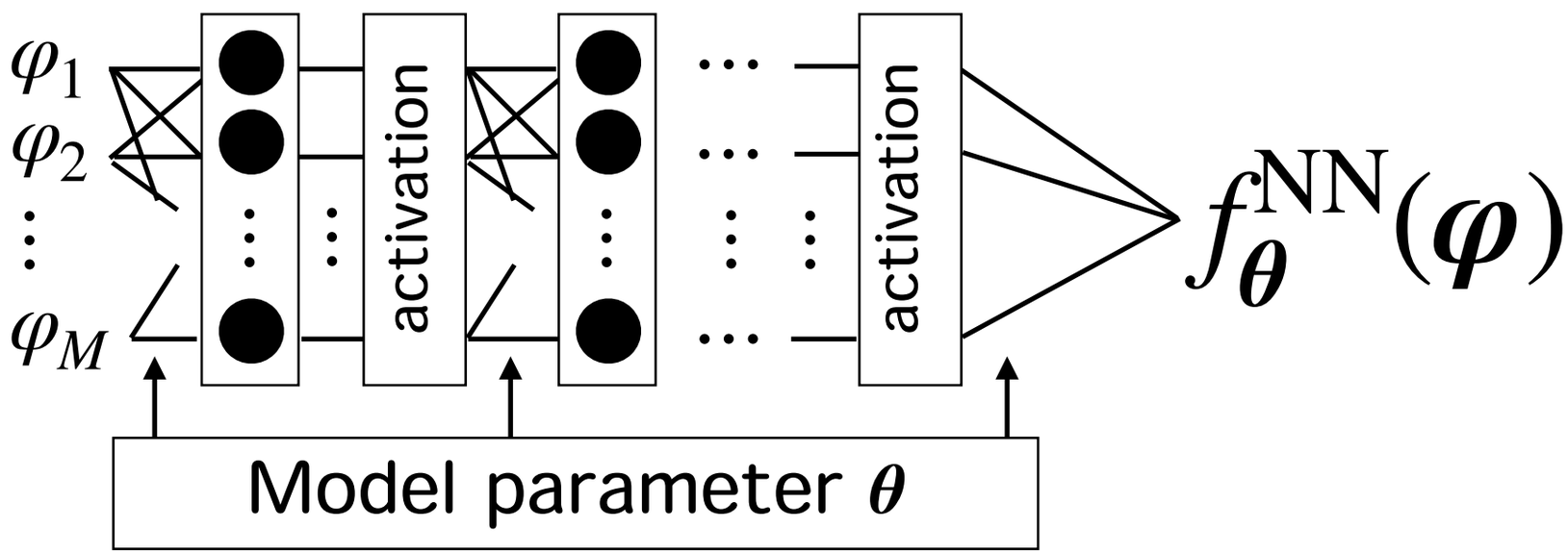
$$\partial_k \Gamma_k(\boldsymbol{\varphi}) = \frac{1}{2}\mathrm{tr}\left[\partial_k R_k \cdot \left(\mathrm{Hess}_{\boldsymbol{\varphi}}\left(\Gamma_k(\boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})\right)\right)^{-1}\right]$$

- Hessian: $\mathrm{Hess}_{\boldsymbol{\varphi}}(\cdots) = \frac{\partial^2}{\partial\boldsymbol{\varphi}\partial\boldsymbol{\varphi}}(\cdots)$
- Regulator term: $\Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi}) = \frac{1}{2}\boldsymbol{\varphi}\cdot R_k \cdot \boldsymbol{\varphi}$

Solving FRG $\approx$ Solving HDPDE

Neural networks (NNs)

Ansatz for a multivariable function



- The derivatives $(\frac{\partial}{\partial \boldsymbol{\varphi}} f_{\boldsymbol{\theta}}(\boldsymbol{\varphi}),\ \frac{\partial^2}{\partial \boldsymbol{\varphi} \partial \boldsymbol{\varphi}} f_{\boldsymbol{\theta}}(\boldsymbol{\varphi}) \cdots)$ are efficiently evaluated even for high-dim. input.
  - Efficient evaluation via the chain rule (automatic differentiation).
  - No explicit grid in the $\boldsymbol{\varphi}$-space is required!

Machine-learning-based method for HDPDE & FRG

HDPDE

- Physics-informed NNs (PINNs)

  Psichogios, Ungar, AIChE (1992), Lagaris, Likas, Fotiadis (1998)
  Raissi, Perdikaris, Karniadakis (2017); JCP (2019)

  - e.g.) 1M-dim. PDEs

    Z. Shi, Z. Hu, M. Lin, K. Kawaguchi, NeurIPS (2024) Best paper

    - Poisson eq., Sine-Gordon eq., Allen-Cahn eq.
    - Few minutes on a single NVIDIA A100 GPU
    - Typical error $\lesssim 1\,\%$ compared to exact solutions
  - FRG *TY, PRB (2024); TY, in prep*
    - Demonstration: Up to 101-dim. PDE
- Rayleigh-Ritz variational method

  E, Yu, Comm. Math. Stat. (2018), Khoo, Lu, Ying, Res. Math. Sci. (2018),…
- Backward stochastic differential equation…

  E, Han, Jentzen, Comm. Math. Stat. (2017), Han, Jetzen, E (2018), Rassi (2018),

  Beck, E, Jentzen, J. Nonlinear Science (2019),…

Other ML-based studies for solving FRG eqs.

- Gaussian process *Yang, Darcy, Hudes, Alexander, Eyink, Owhadi, J. Comp. Phys. (2026)*
  - 2-dim. PDE demonstration
- PINN-based method for LPA flow eq. *Tan, Fu, He, Wang, PRD (2026)*

Yang-yang's talk

Machine-learning-based method for HDPDE & FRG

HDPDE

- Physics-informed NNs (PINNs)

  Psichogios, Ungar, AIChE (1992), Lagaris, Likas, Fotiadis (1998)
  Raissi, Perdikaris, Karniadakis (2017); JCP (2019)

  - e.g.) 1M-dim. PDEs

    Z. Shi, Z. Hu, M. Lin, K. Kawaguchi, NeurIPS (2024) Best paper

    - Poisson eq., Sine-Gordon eq., Allen-Cahn eq.
    - Few minutes on a single NVIDIA A100 GPU
    - Typical error $\lesssim 1\,\%$ compared to exact solutions
  - FRG *TY, PRB (2024); TY, in prep*
    - Demonstration: Up to 101-dim. PDE

Today's talk!

- Rayleigh-Ritz variational method

  E, Yu, Comm. Math. Stat. (2018), Khoo, Lu, Ying, Res. Math. Sci. (2018),…

- Backward stochastic differential equation…

  E, Han, Jentzen, Comm. Math. Stat. (2017), Han, Jetzen, E (2018), Rassi (2018),

  Beck, E, Jentzen, J. Nonlinear Science (2019),…

Other ML-based studies for solving FRG eqs.

- Gaussian process

  Yang, Darcy, Hudes, Alexander, Eyink, Owhadi, J. Comp. Phys. (2026)

  - 2-dim. PDE demonstration
- PINN-based method for LPA flow eq.

  Tan, Fu, He, Wang, PRD (2026)

Yang-yang's talk

PINNs for FRG (PINN-FRG)

TY, PRB (2024)

- NN ansatz for the solution: $\Gamma_k(\boldsymbol{\varphi}) \approx f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k, \boldsymbol{\varphi})$
- Loss for optimization of $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k, \boldsymbol{\varphi})$

$$L(\boldsymbol{\theta}) = \frac{1}{N_{\mathrm{col}}} \sum_{i=1}^{N_{\mathrm{col}}} r_{\boldsymbol{\theta}} \left(k^{(i)}, \boldsymbol{\varphi}^{(i)}\right)^2$$

  - Residual (LHS – RHS of the Wetterich eq.)

$$r_{\boldsymbol{\theta}}(k, \boldsymbol{\varphi}) = \partial_k f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k, \boldsymbol{\varphi}) - \frac{1}{2} \mathrm{tr} \left[\partial_k R_k \cdot \left(\mathrm{Hess}_{\boldsymbol{\varphi}} \left(f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k, \boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi}) \right) \right)^{-1} \right]$$

    - $(k^{(i)}, \boldsymbol{\varphi}^{(i)})$ $(i = 1, \ldots, N_{\mathrm{col}})$: Collocation points
      - Sampled from $k_{\mathrm{IR}} \leq k \leq k_{\mathrm{UV}}$ & the domain of $\boldsymbol{\varphi}$

- The initial condition $\Gamma_{k_{\mathrm{UV}}}(\boldsymbol{\varphi}) = S(\boldsymbol{\varphi})$ (bare action) is implemented in the ansatz $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k, \boldsymbol{\varphi})$.

NN architecture for stable training

- Multi-layer perceptron (MLP) based model for $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi})$ *TY, PRB (2024)*
  - But, the convexity of $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})$ is not guaranteed

$$r_{\boldsymbol{\theta}}(k,\boldsymbol{\varphi}) = \partial_k f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) - \frac{1}{2}\mathrm{tr}\left[\partial_k R_k \cdot \left(\mathrm{Hess}_{\boldsymbol{\varphi}}\left(f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})\right)\right)^{-1}\right]$$

    - Loss of convexity → Loss of Hessian regularity
      → Breakdown of matrix inv. calc. → Breakdown of training

NN architecture for stable training

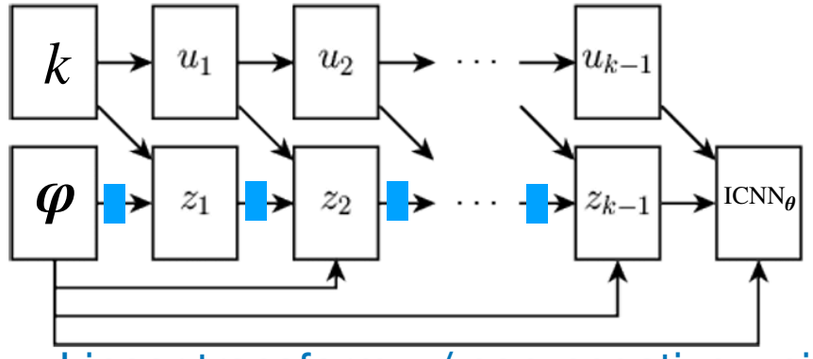
- Multi-layer perceptron (MLP) based model for $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi})$ *TY, PRB (2024)*
  - But, the convexity of $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})$ is not guaranteed

$$r_{\boldsymbol{\theta}}(k,\boldsymbol{\varphi}) = \partial_k f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) - \frac{1}{2}\mathrm{tr}\left[\partial_k R_k \cdot \left(\mathrm{Hess}_{\boldsymbol{\varphi}}\left(f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) + \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})\right)\right)^{-1}\right]$$

    - Loss of convexity → Loss of Hessian regularity
      → Breakdown of matrix inv. calc. → Breakdown of training

- Input convex NN (ICNN) for $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi})$
  - ICNN guarantees the convexity

    Amos, Xu, Kolter, ICML (2017)



Linear transform w/ non-negative weight
& convex activation

  - Ansatz for $f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi})$ *TY, in prep*

$$f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) = \lambda_{\boldsymbol{\theta}_1}(k)\left(S(\boldsymbol{\varphi}) + \Delta S_{k_{\mathrm{UV}}}^{\mathrm{reg}}(\boldsymbol{\varphi})\right) + \left(1 - \lambda_{\boldsymbol{\theta}_1}(k)\right)\mathrm{ICNN}_{\boldsymbol{\theta}_2}(k,\boldsymbol{\varphi}) - \Delta S_k^{\mathrm{reg}}(\boldsymbol{\varphi})$$

    - $\lambda_{\boldsymbol{\theta}_1}(k)$: Gate function ($\lambda_{\boldsymbol{\theta}_1}(k_{\mathrm{UV}}) = 1,\ 0 \le \lambda_{\boldsymbol{\theta}_1}(k) \le 1$)
    - The convexity & initial condition are satisfied.

Contents

1. Physics-informed neural networks (PINNs) for FRG
    - Motivation & method
    - Network architecture
2. Demonstration in a toy model
    - Calculation w/ multi-layer perceptron
    - Calculation w/ input-convex neural network

Toy-model demonstration: 0-dim classical $\mathrm{O}(N)$ model

Model $S(\boldsymbol{\varphi}) = \frac{1}{2}m^2\boldsymbol{\varphi}^2 + \frac{g}{4!}(\boldsymbol{\varphi}^2)^2$

- Exactly solvable E.g., *Keitel, Bartosch, JPA (2012)*
- Non-perturbative region: $\tilde{g} = Ng/m^4 \gtrsim 1$

- Wetterich eq.: $(N+1)$-dim. PDE (We do **not** use the reduction of variable $\boldsymbol{\varphi} \to \rho = \boldsymbol{\varphi}^2$)
- We calculate $\Gamma_k(\boldsymbol{\varphi})$ for $1 \leq N \leq 100$ and both $m^2 > 0$ and $m^2 < 0$ cases
  - The convexity problem becomes severe for $m^2 < 0$.

PINN-FRG setup

- Regulator: $R_k = k^2\,\mathbf{1}_{N\times N}$
- NN: (1) MLP based model *TY, PRB (2024)*

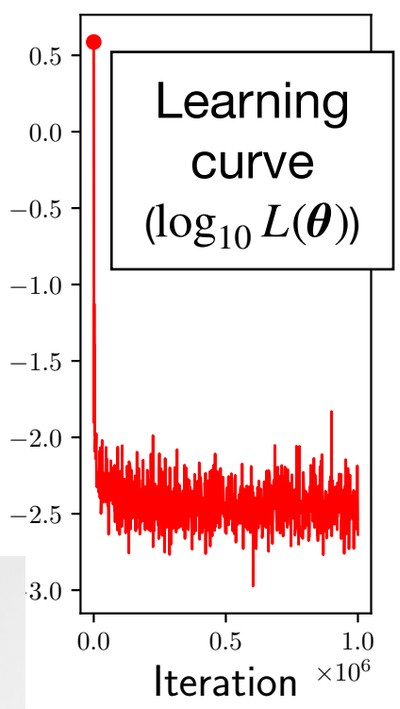
$$f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) = S(\boldsymbol{\varphi}) + \mathrm{MLP}_{\boldsymbol{\theta}}(k,\boldsymbol{\varphi}) - \mathrm{MLP}_{\boldsymbol{\theta}}(k_{\mathrm{UV}},\boldsymbol{\varphi})$$

MLP structure: 3 hidden layers, 256 units/layer, Softplus activation functions

(2) ICNN based model *TY, in prep*

$$f_{\boldsymbol{\theta}}^{\mathrm{NN}}(k,\boldsymbol{\varphi}) = \lambda_{\boldsymbol{\theta}_1}(k)\left(S(\boldsymbol{\varphi}) + \Delta S^{\mathrm{reg}}_{k_{\mathrm{UV}}}(\boldsymbol{\varphi})\right) + \left(1-\lambda_{\boldsymbol{\theta}_1}(k)\right)\mathrm{ICNN}_{\boldsymbol{\theta}_2}(k,\boldsymbol{\varphi}) - \Delta S^{\mathrm{reg}}_{k}(\boldsymbol{\varphi})$$

ICNN structure: 4 hidden layers, 64 units/layer, Softplus & tanh activation functions



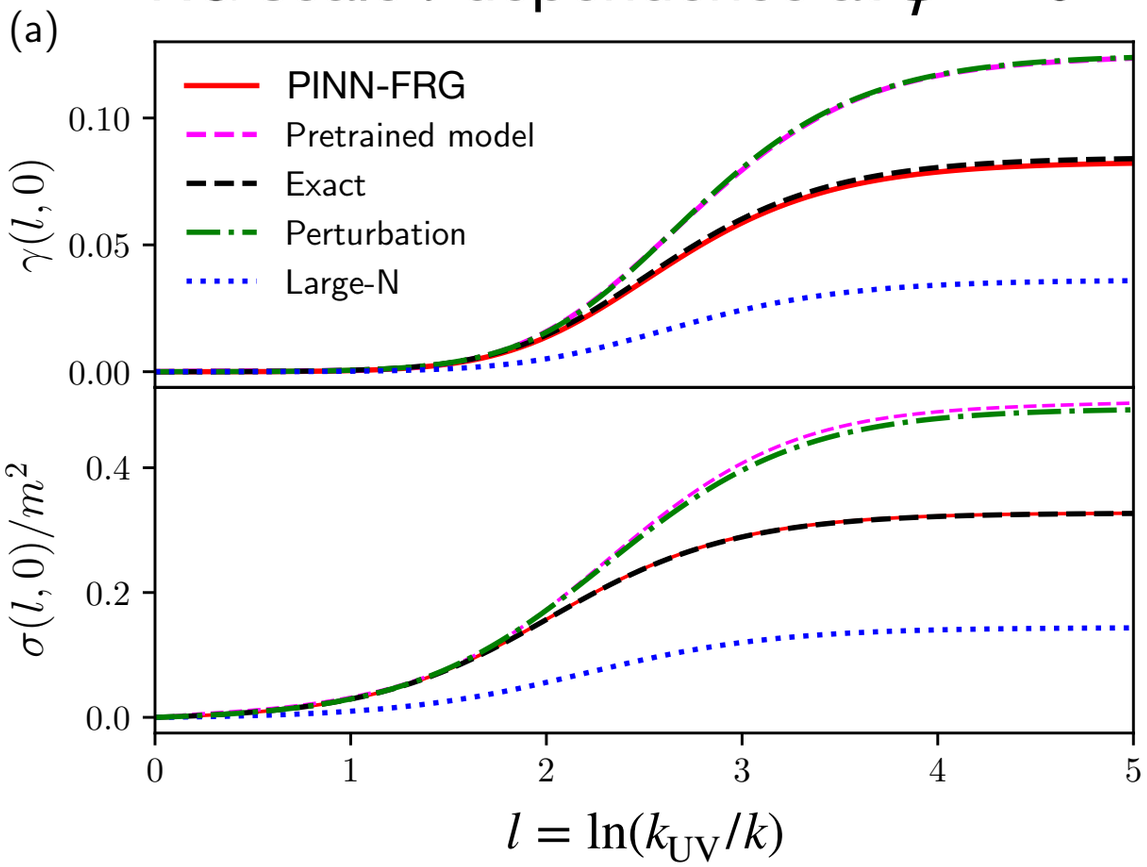
- Time for optimization: few minutes ~ few hours
  - Resource: NVIDIA A100 & RTX5080 GPU
  - Code implementation: PyTorch
    - https://github.com/TakeruYokota/PINNLFRG.git

$N = 1,\ \tilde{g} = 1,\ m^2/k_{\mathrm{UV}}^2 = 0.01$ case (MLP based model)

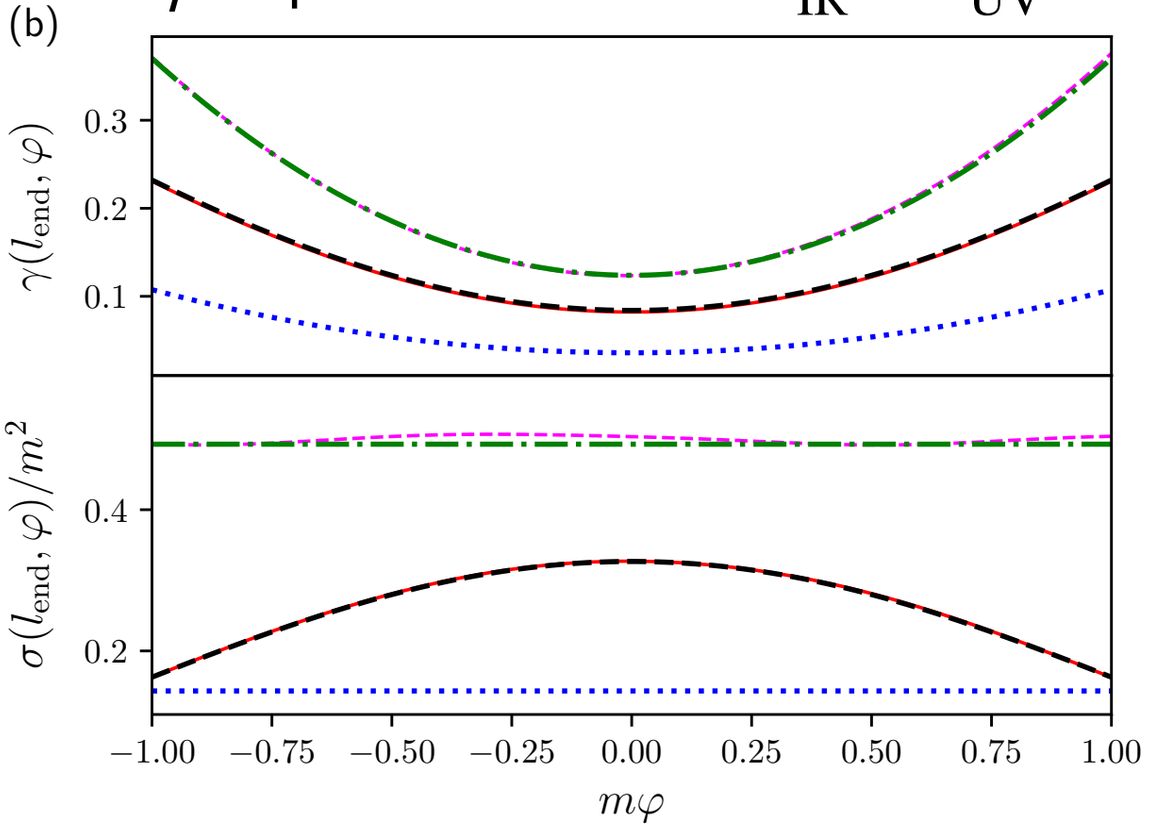
TY, PRB (2024)

- Reduced effective action: $\gamma(l, \varphi) = \Gamma_k(\varphi) - S(\varphi) - \Delta\Gamma_k^{\mathrm{free}} \quad (l = \ln(k_{\mathrm{UV}}/k))$
- Reduced self-energy $\sigma_\alpha(l, \boldsymbol{\varphi}) = \dfrac{\partial^2}{\partial\varphi_\alpha^2}\gamma(l, \boldsymbol{\varphi})$

RG scale l dependence at $\varphi = 0$



φ dependence at $k = k_{\mathrm{IR}} = k_{\mathrm{UV}}e^{-10}$

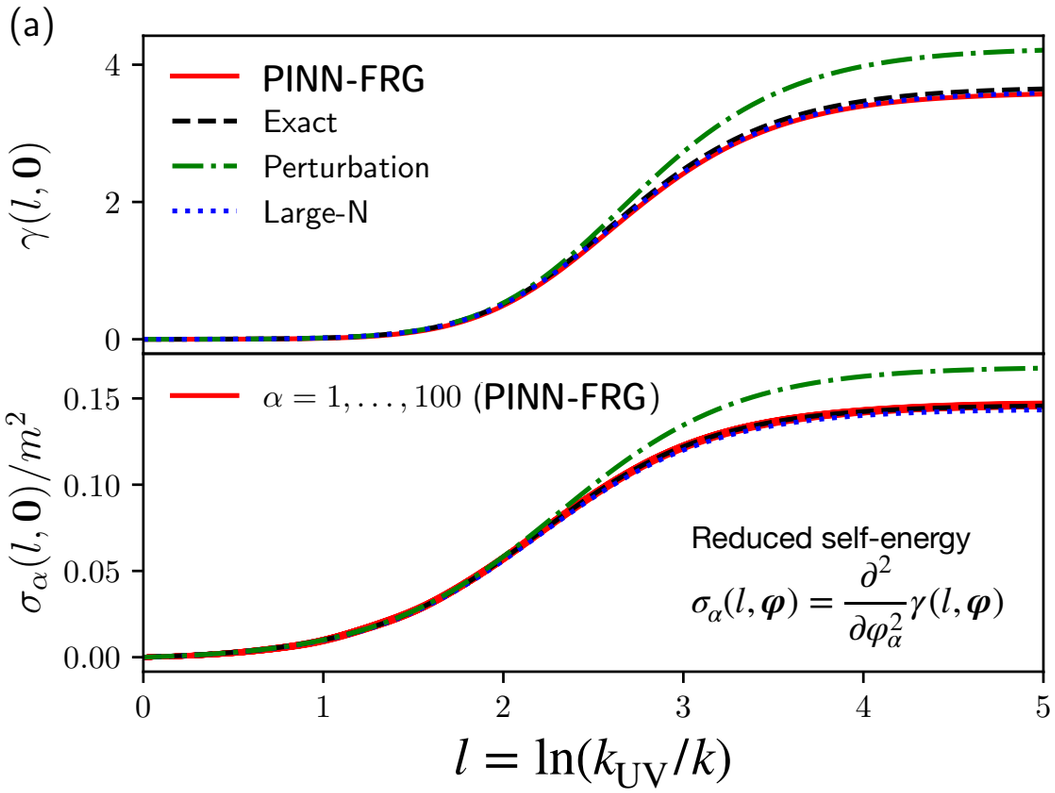


- $\gamma(l, \varphi)$ is simultaneously obtained for a domain of φ in our method (PINN-FRG).
- PINN-FRG shows accurate results compared to 1st-order perturbation & leading-order large-N expansion

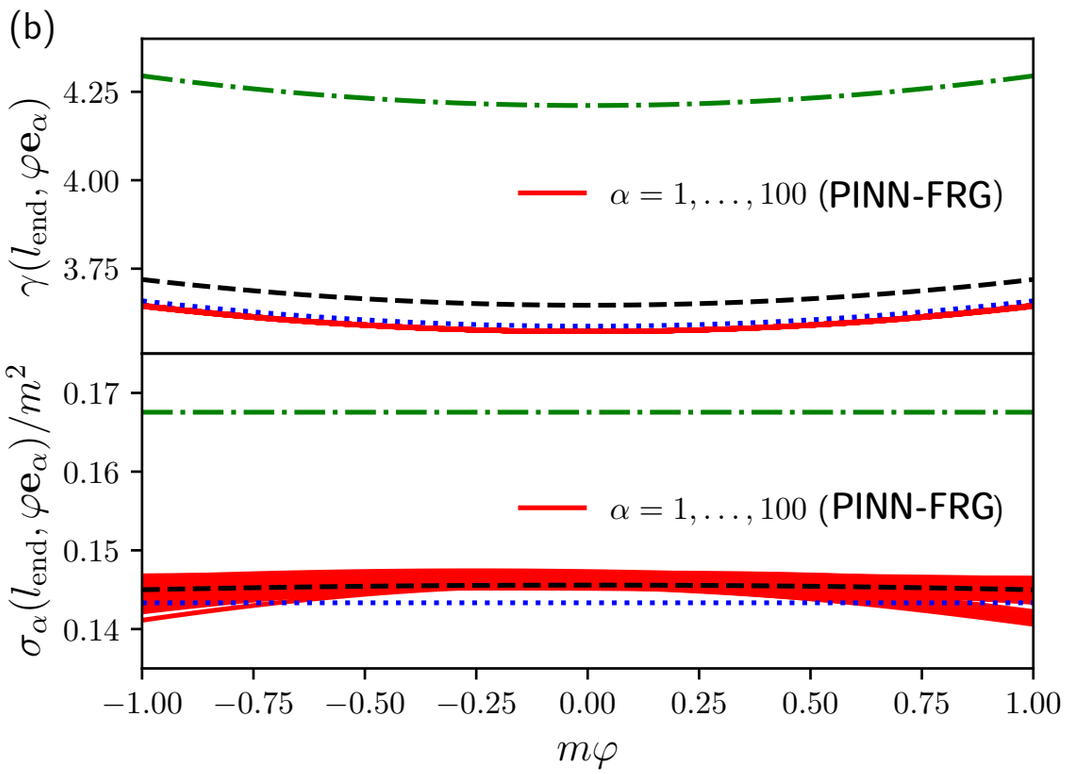
$N = 100,\ \tilde{g} = 1,\ m^2/k_{\mathrm{UV}}^2 = 0.01$ case (MLP based model)

TY, PRB (2024)

RG scale l dependence at $\varphi = 0$



φ dependence at $k = k_{\mathrm{IR}} = k_{\mathrm{UV}}e^{-10}$



- Except for $\gamma(l, \mathbf{0})$, PINN-FRG results are given by $N = 100$ lines corresponding to the N-direction in $\boldsymbol{\varphi}$ space.
- PINN-FRG shows comparable results with large-N expansion, which should be accurate for $N = 100$.
- $\mathrm{O}(N)$ symmetry is reproduced in PINN-FRG.

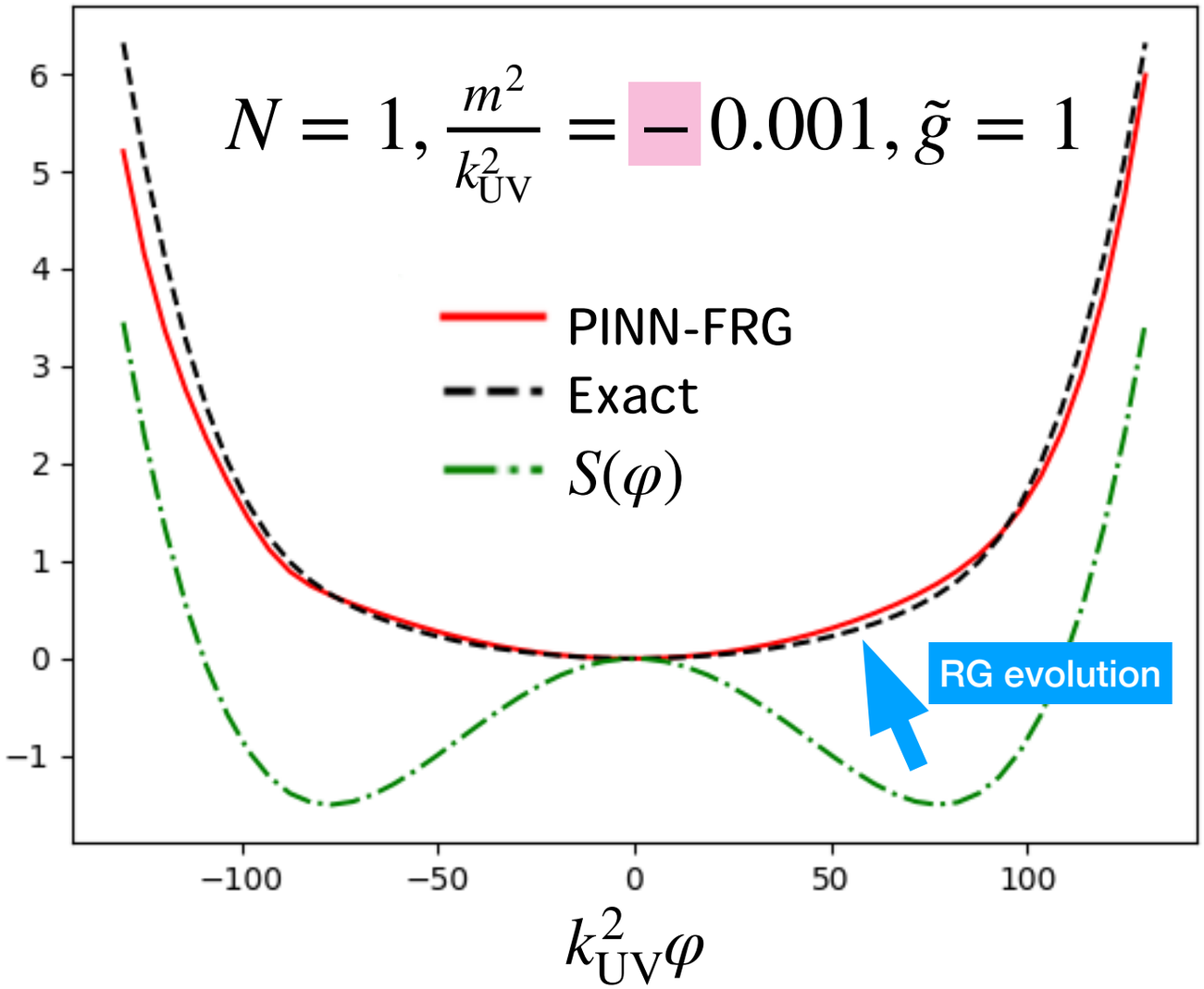
$m^2 < 0$ case (ICNN based model)

TY, in prep

- In the case of MLP-based models, the loss of convexity becomes severe, causing the training to fail.

- In models based on ICNN, there is no loss of convexity, and the calculations can be performed.

$\Gamma_{k_{\mathrm{IR}}}(\varphi) - \Gamma_{k_{\mathrm{IR}}}(0)$ (preliminary)



Next step: Localization phase transition in the Caldeira-Leggett model

- Double-well potential + external noise
- 0-dim. quantum model

Summary

TY, PRB (2024)
TY, in prep

- Physics-informed neural networks (PINNs) may offer a new approach to solving FRG eqs.
- The use of input convex neural networks (ICNNs) is key to the success of PINN-FRG.
- We have demonstrated the performance of PINN-FRG using 0-dim. classical $\mathrm{O}(N)$ model.
  - Numerical demonstration up to 101-dim. PDE

Outlook

- Application to the localization phase transition in the Caldeira-Leggett model
- Application to other functional diff. eqs. *Miyagawa, TY, NeurIPS2024 (2024)*
  - Fokker-Planck, Hopf…

Supported by JSPS KAKENHI
Grant No. 25H00177 and No. 25K23364

Backup

Hessian's invertibility & pretraining (MLP)

$$L_{\boldsymbol{\theta}} = \mathop{\mathbb{E}}_{\substack{\boldsymbol{\varphi} \sim \mathscr{P}_{\boldsymbol{\varphi}} \\ l \sim \mathscr{P}_l}} \left[\left(\partial_l \Gamma_l^{\boldsymbol{\theta}}(\boldsymbol{\varphi}) - \frac{1}{2} \mathrm{tr} \left[\partial_l R_l \left(\frac{\partial^2 \Gamma_l^{\boldsymbol{\theta}}(\boldsymbol{\varphi})}{\partial \boldsymbol{\varphi} \partial \boldsymbol{\varphi}} + R_l \right)^{-1} \right] \right)^2 \right]$$

The matrix must be invertibility during the training.

In our experience, this is frequently broken with randomly chosen $\boldsymbol{\theta}$.

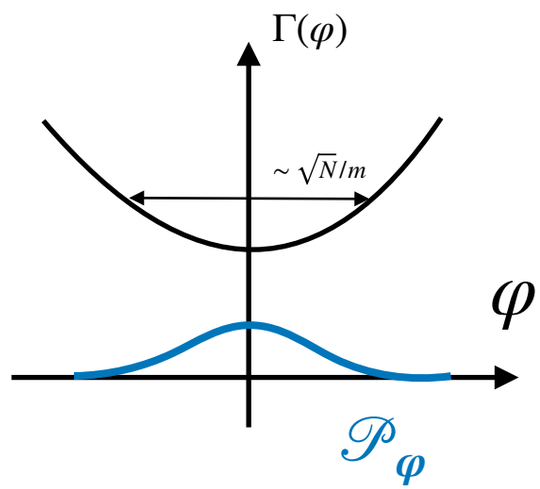
Pretraining with some approximate analytic results remedies this problem

We use 1st-order perturbative result:

$$L_{\boldsymbol{\theta}}^{\mathrm{pre}} = \mathop{\mathbb{E}}_{\substack{\boldsymbol{\varphi} \sim \mathscr{P}_{\boldsymbol{\varphi}} \\ l \sim \mathscr{P}_l}} \left[\left(\gamma(l, \boldsymbol{\varphi}; \boldsymbol{\theta}) - \gamma^{\mathrm{1pt}}(l, \boldsymbol{\varphi}) \right)^2 \right]$$

Other details of numerical procedure

$$L_{\boldsymbol{\theta}} = \mathop{\mathbb{E}}_{\substack{\boldsymbol{\varphi} \sim \mathscr{P}_{\boldsymbol{\varphi}} \\ l \sim \mathscr{P}_l}} \left[\left(\partial_l \Gamma_l^{\boldsymbol{\theta}}(\boldsymbol{\varphi}) - \frac{1}{2} \mathrm{tr} \left[\partial_l R_l \left(\frac{\partial^2 \Gamma_l^{\boldsymbol{\theta}}(\boldsymbol{\varphi})}{\partial \boldsymbol{\varphi} \partial \boldsymbol{\varphi}} + R_l \right)^{-1} \right] \right)^2 \right]$$



- $\mathscr{P}_l$: uniform distribution in $[0, l_{\mathrm{end}}]$ with $l_{\mathrm{end}} = 5$
- $\mathscr{P}_{\boldsymbol{\varphi}}$: $\|\boldsymbol{\varphi}\|$ is sampled following $N(0, N/m^2)$ (w/o sign)
  $\hat{\boldsymbol{n}} = \boldsymbol{\varphi}/\|\boldsymbol{\varphi}\|$ is uniformly sampled

  * The variance N/m^2 is needed to learn the shape of Γ.

  * Other choices such as $N(\mathbf{0}, m^{-2}\mathbf{1})$ fail to sample the neighborhoods of $\boldsymbol{\varphi} = \mathbf{0}$ due to curse of dimensionality
- 500 collocation points are used to evaluate the expectation.
- Adam optimizer
- Pytorch
- The matrix inverse is evaluated by direct method
  - This may not be efficient but is easy to implement (torch.linalg.inv)

Computational time & convergence (MLP)

We conduct computations for all the combinations of

$$N = 1,10,100 \text{ and } \tilde{g} = 0.1,1,10 \qquad \tilde{g} = Ng/m^4$$

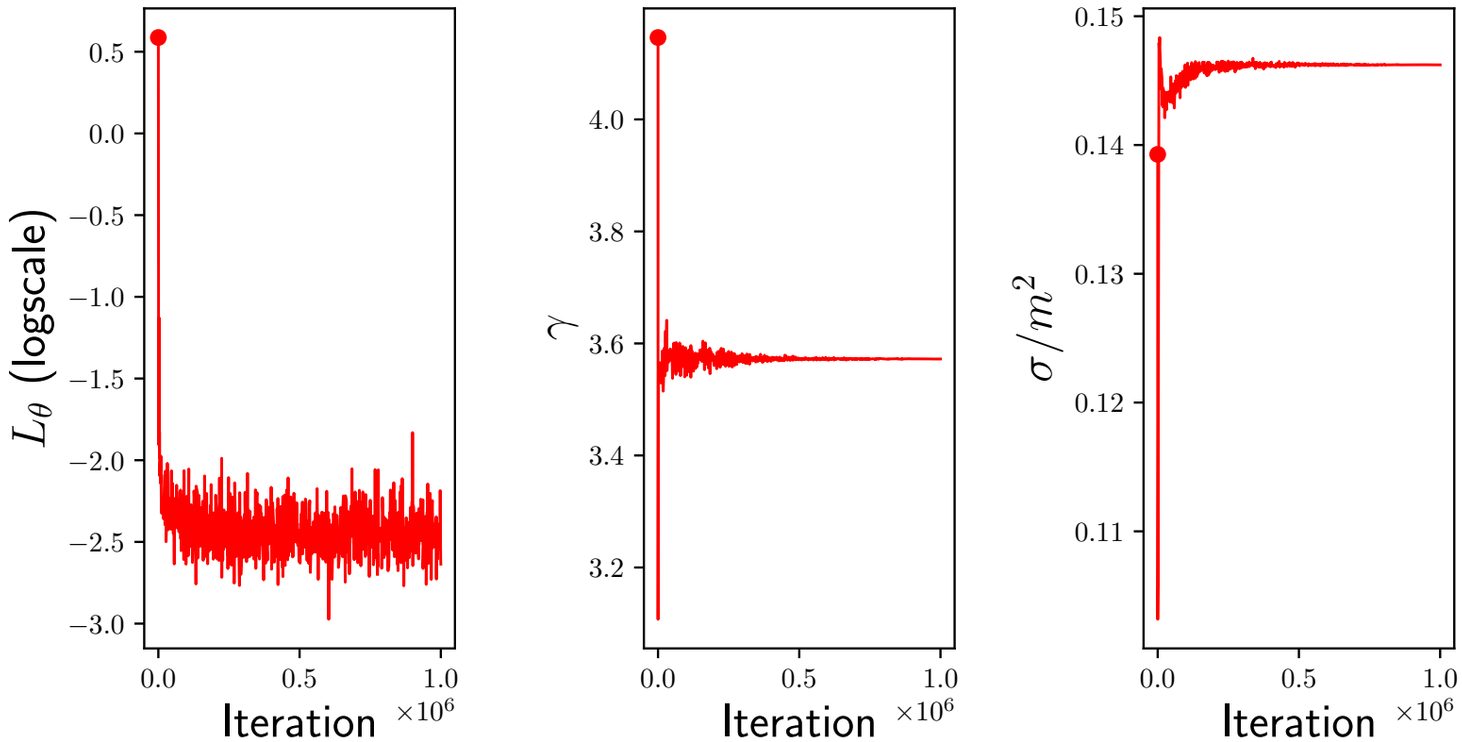
- Learning rate (Wetterich): 10^{-4} with exponential decay factor 0.99999
- Learning rate (pretraining): 10^{-3}

Comp. time on NVIDIA A100 GPU

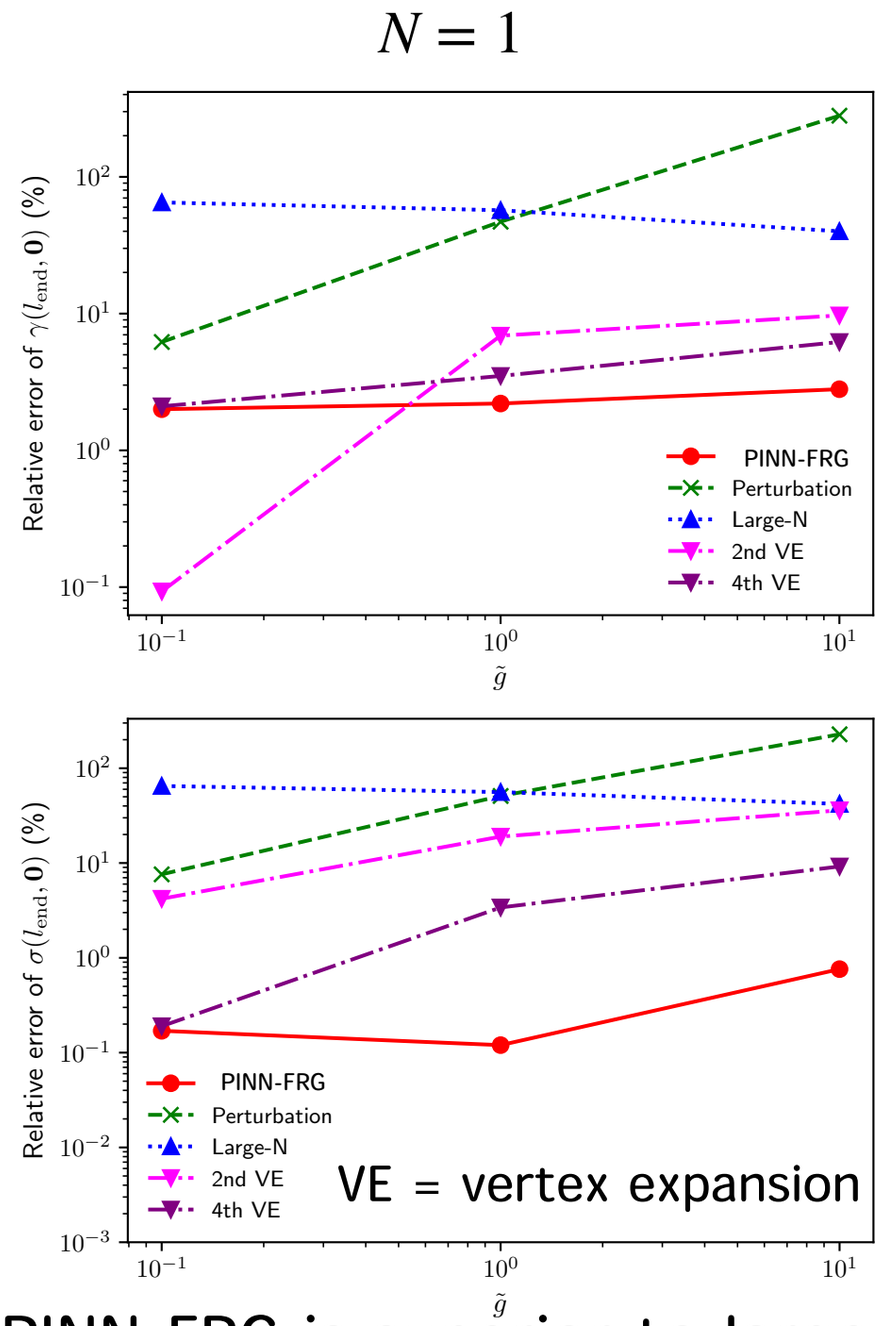
N	1	10	100
Pretraining	4m	4m	6m
Wetterich	6h	7h	11h

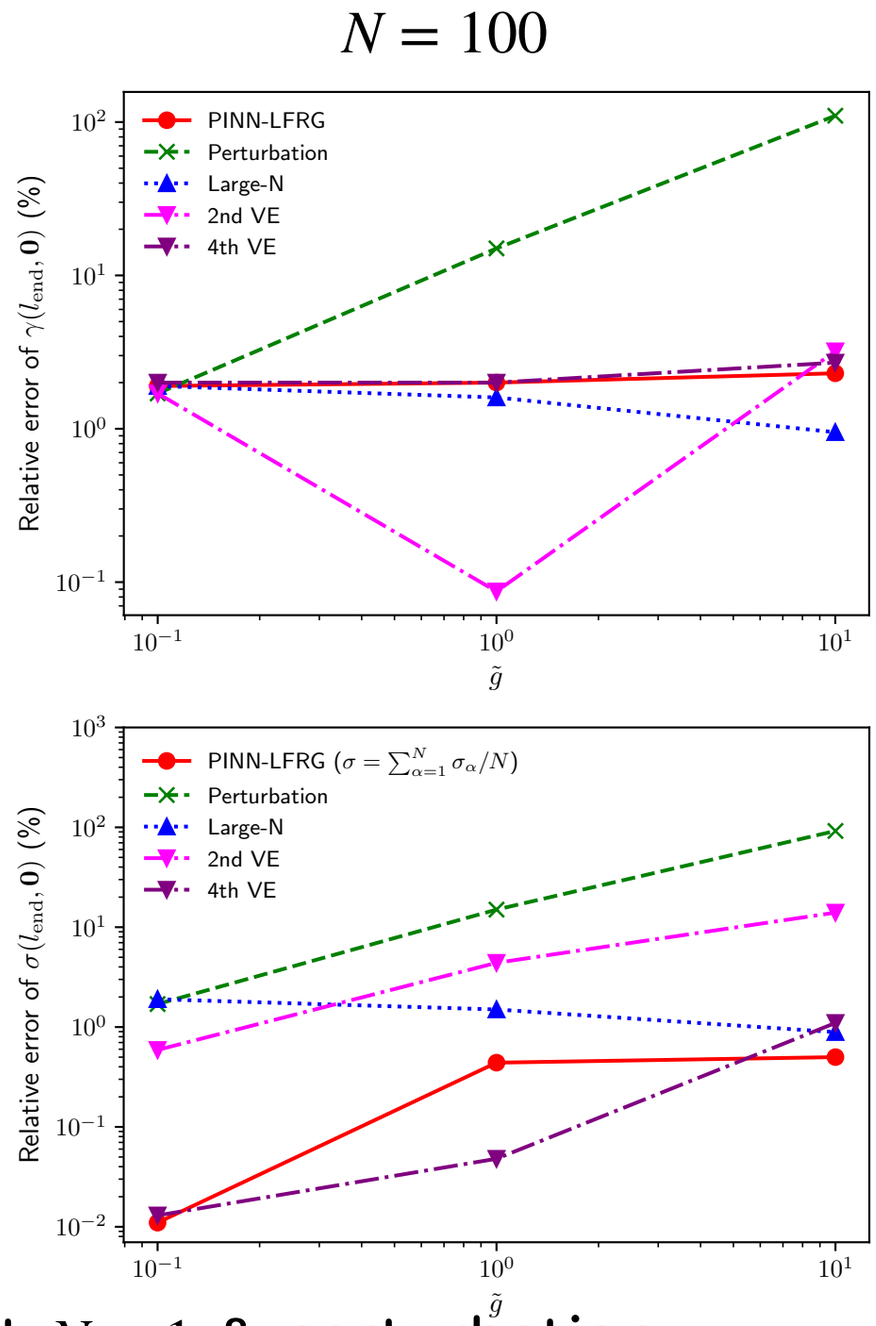
* At larger $N(\gtrsim 1000)$, we ran out of memory.

Learning curve & histories of physical quantities ($N = 100$ & $\tilde{g} = 1$ case)



Relative errors at $\varphi = 0$ for different N and $\tilde{g}$ (MLP)





- PINN-FRG is superior to large-N at $N = 1$ & perturbation
- PINN-FRG shows better accuracy at non-perturbavie region ($\tilde{g} = 10$) compared to 2nd & 4th VE.