The generative model for heavy-ion collisions

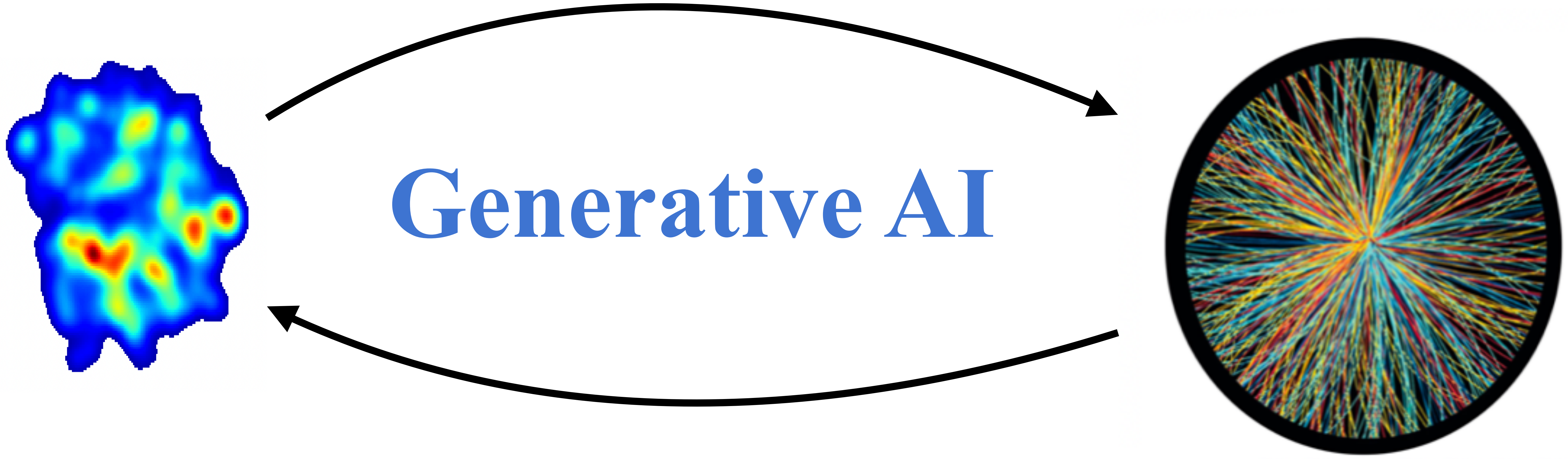
Jing-An Sun

Collaborators: Li Yan,
Sangyong Jeon, Charles Gale

2025.11.02@南华大学

第四届全国核物理及核数据中的机器学习应用研讨会

- Accelerate: High-fidelity and Fast emulator.



- Inverse-engineering: Help solve inverse problems.

The generative AI revolution

Learn the distribution $p(y|x)$ and generate new samples following such distribution.

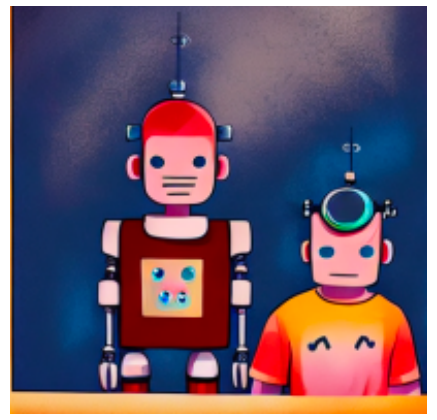
Kai Zhou Talk at 11.01

- Text generation: OpenAI deepseek
- **Text-to-Image generation: diffusion-based**

Prompts:

"An astronaut riding a pig"
"The siblings are all robots"
"An anime girl with pink hair and sunglasses"
"A spotted dog, a cat and a bird on a round table"
"Four cats surrounding a dog"
……..

Generative AI →





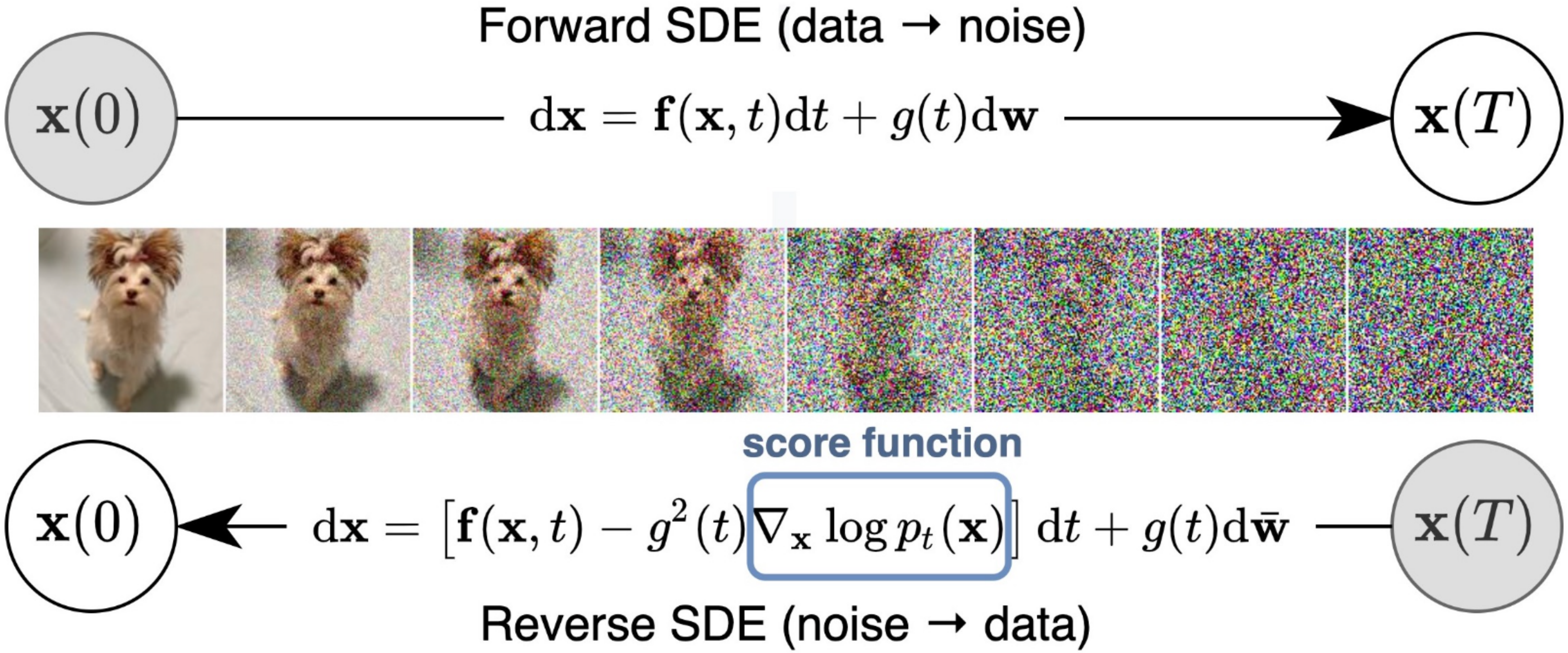
....

- Demonstrated powerful representative ability and flexibility.

Diffusion generative model

- **Learn the data distributions via noising and denoising.**

Y. Song et.al ICLR 2021



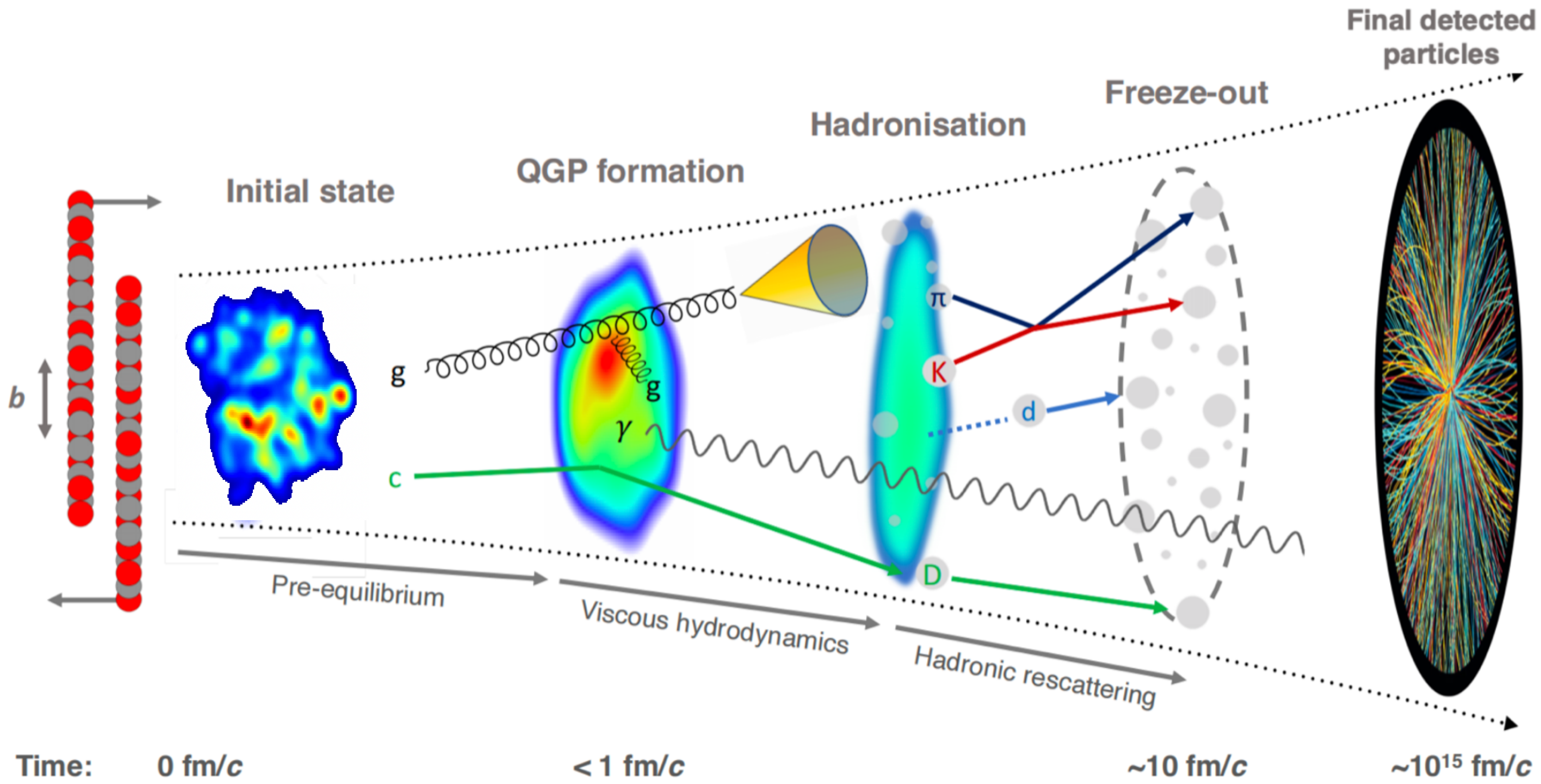
The only unknown term is the score function.

Train a network serving as **a proxy of score function**, one can generate the clean data from a standard norm distribution via **solution of the reverse stochastic differential equation(SDE)**.

The hybrid modeling for heavy-ion collisions

- **The hybrid modeling achieves tremendous success:**

  Particle spectra from experiment (collective flow, flow correlations, fluctuations) are accurately characterized.



ALICE Collaboration Eur.Phys.J.C 84 (2024) 8, 813

Trento+MUSIC+iSS+UrQMD

Next generation demands

- **Computational Limitation:**

L.G Pang et.al Phys.Rev.C 97 (2018) 6, 064918
D Bazar et.al Comput.Phys.Commun. 225 (2018) 92-113

Method	Hardware	Time/Event	Scalability (1e6Events)
Hybrid	CPU (single-core)	**~120 min**	**>20 years**
GPU-accelerated	GeForce GTX Titan Z	~1 min	~2 months
Generative AI	NVIDIA GTX 4090	**~0.1 sec**	**~1 day**

- Imaging the nuclei shape: $10^7 - 10^8$ events are typically selected.

STAR Nature 635 (2024) 8037, 67-72

- Probing QGP thermodynamic properties (speed of sound) requires $10^9 - 10^{10}$ statistics.

– Experimentally accessible

– **Computationally prohibitive for hybrid models**

CMS Rept.Prog.Phys. 87 (2024) 7, 077801
ALICE PoS ICHEP2024 (2025) 600

- **Next-Generation Demands**

As heavy-ion collision physics enters **the high-precision era**:

- Current models require significant advancement to **meet growing computational demands**

How can machine learning help us?

- **Goal:**
  - End-to-end (initial state to final state)
  - Fast and flexible
  - Keeping physical consistency and physical controllable

- **The challenges in deterministic machine learning**

deterministic machine learning

- sUnet: Solves hydrodynamic equations but **accumulates errors** in long-time evolution.

- Gaussian emulator in Bayesian analysis: Maps parameters to observables (digital to digital fit) but **lacks flexibility**.

*Learn the hard map from **parameter to parameter***

H.F Huang et.al Phys.Rev.Res. 3 (2021) 2, 023256

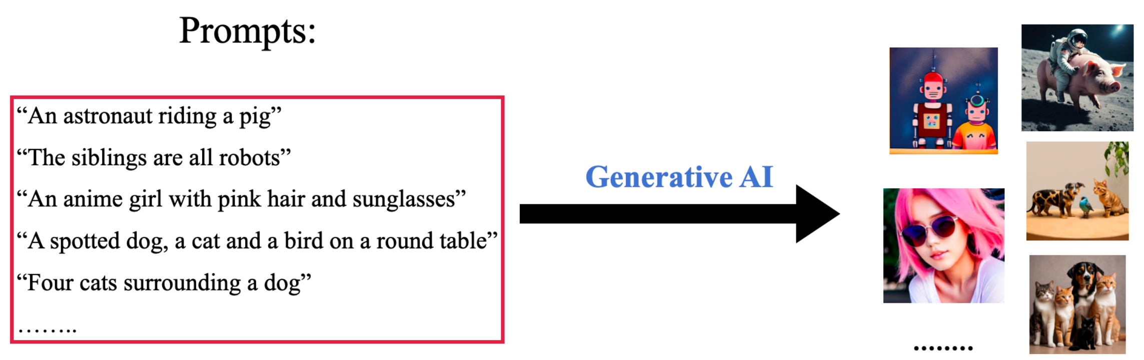
J.E Bernhard et.al Nature Phys. 15 (2019) 11, 1113-1117

The generative AI solution

- **Problem setting**

$$p(\text{particle spectra} \,|\, \text{initial state, transport coefficients,etc...})$$

Generate the final particle spectra conditioned by the initial entropy density profile and the transport coefficients, a process is governed by hydrodynamic evolution and Boltzmann transport.

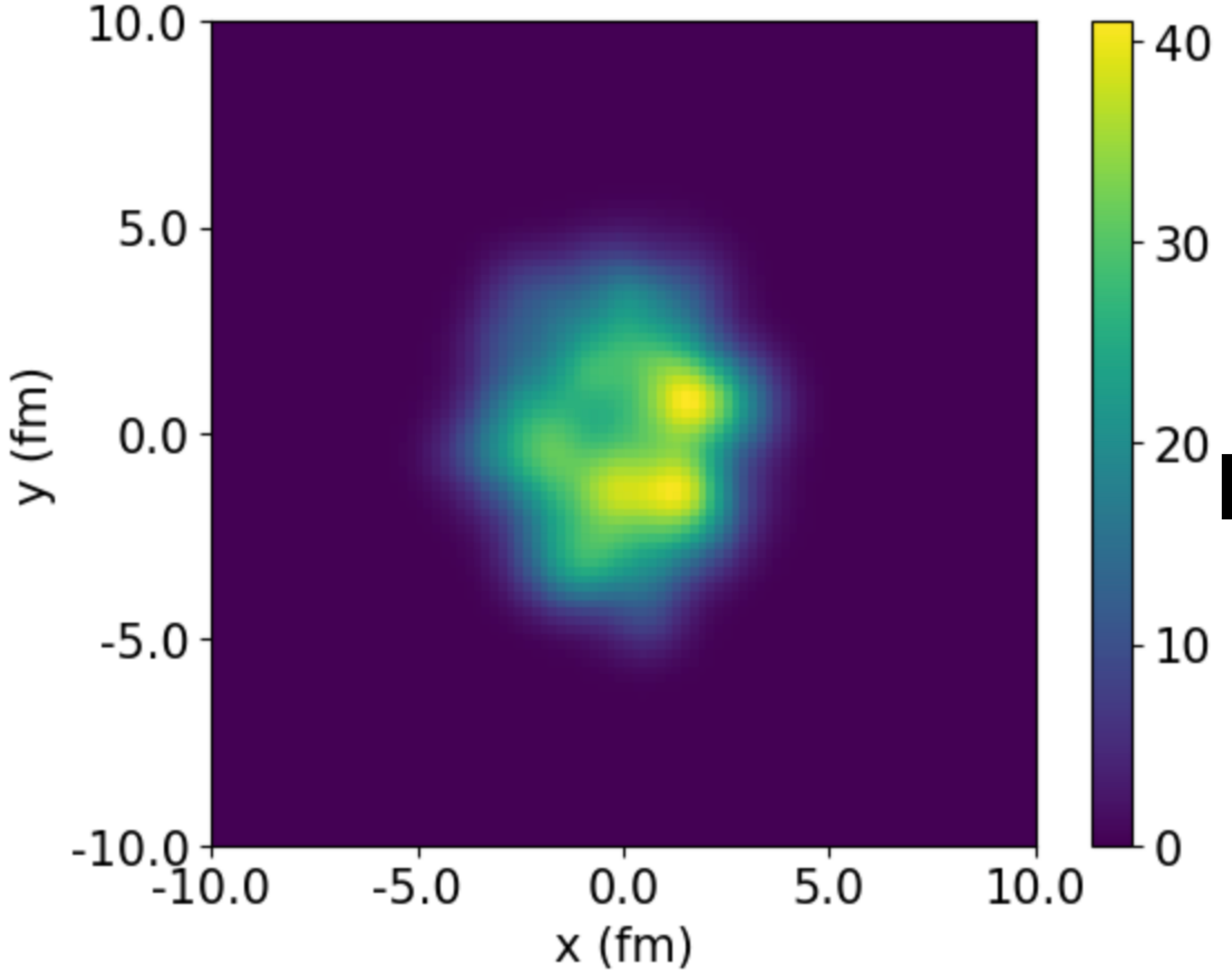


Generative AI →

"Prompts": initial profile $(p, k \ldots.)$, η/s, ζ/s, EOS, after-burner parameters,…..

Data preparation

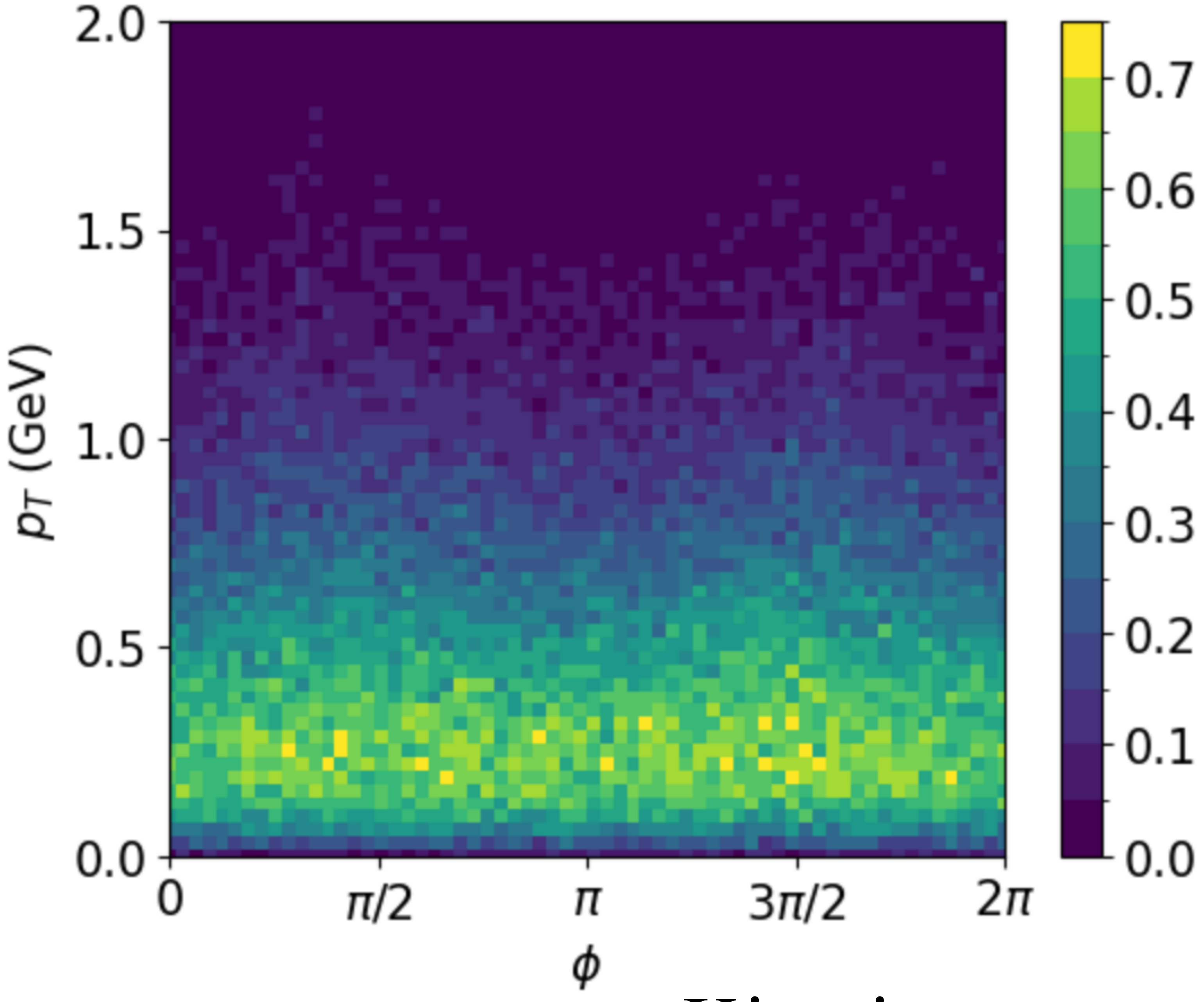
Initial entropy density profile



2+1 D hydrodynamics

Trento+MUSIC+iSS+UrQMD

charged particle spectra

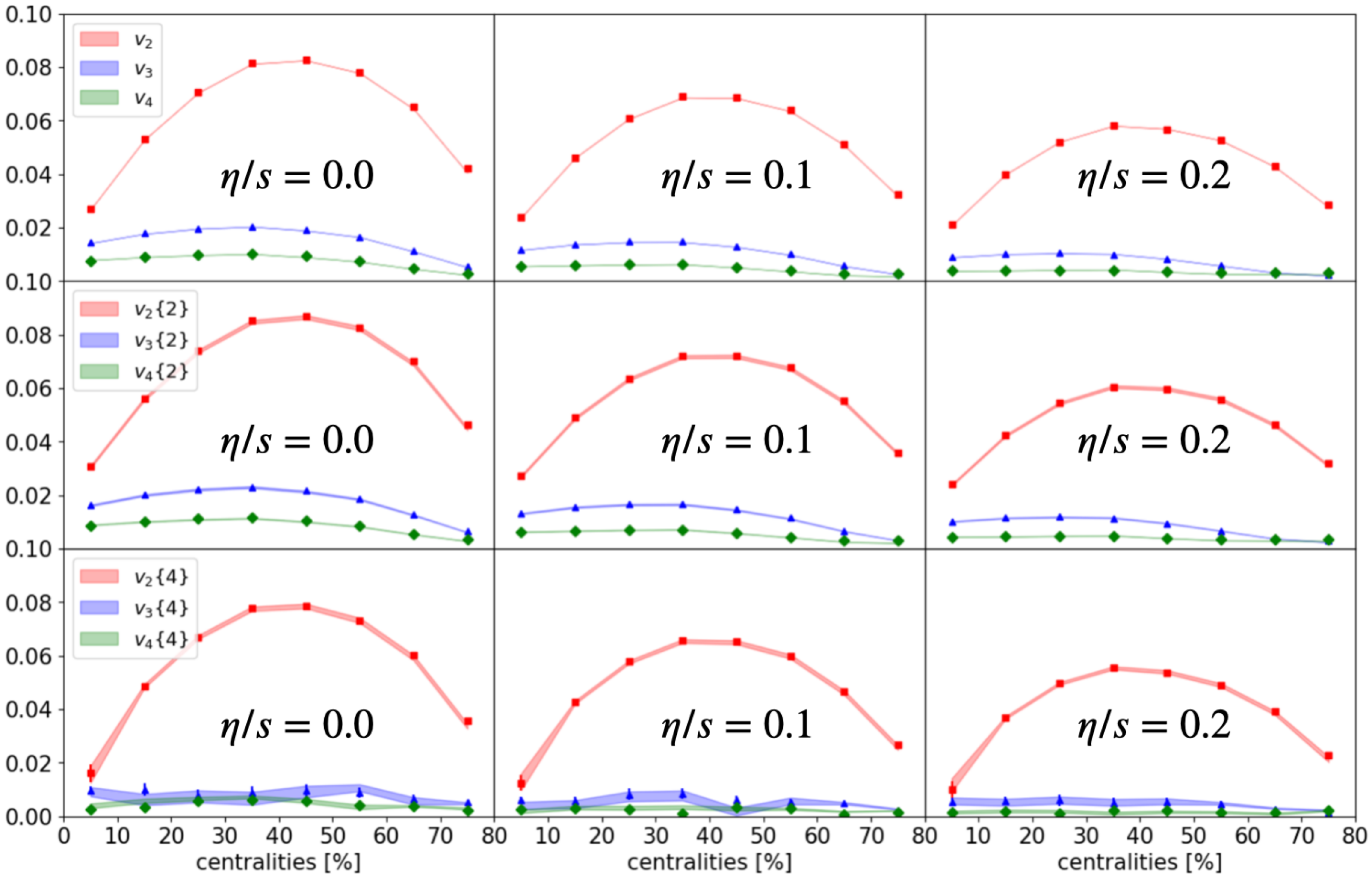


Kinetic cut:
$0 < p_T < 2$ GeV,
$|y| < 0.5$

- **Parameter set**: initial profile $(p, k \ldots.)$, η/s, ζ/s, EOS, after-burner parameters,…..
- We take three different shear viscosity $\eta/s = 0{,}0.1{,}0.2$
- **For each η/s, we prepare 12,000 mini-bias events** for training.

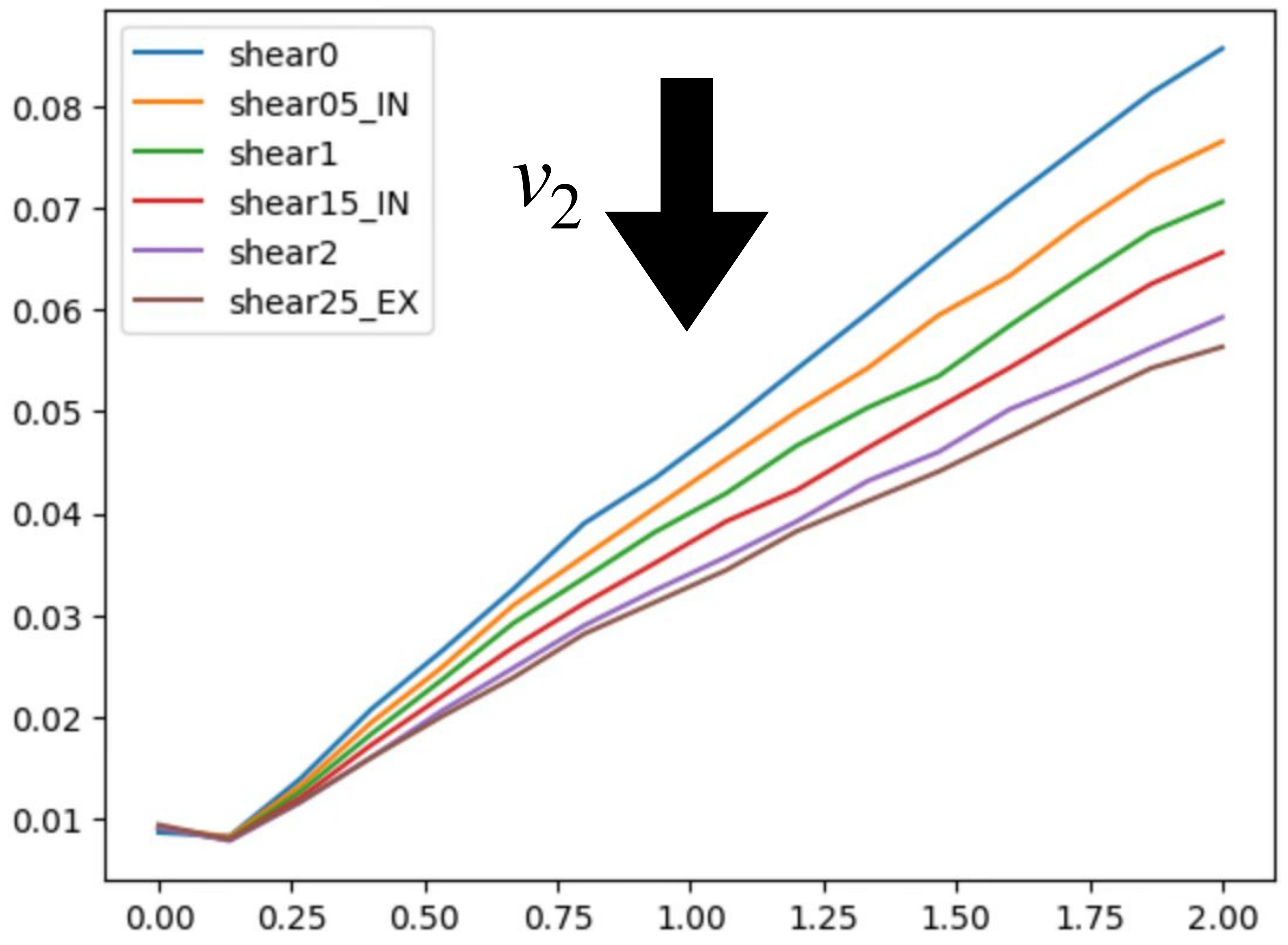
For test, we run **extra 10,000 events for each η/s at each centrality.**

The integrated flow and cumulants



Symbols: Hydro

Bands: Generative AI



Excellent performance for collective flow!

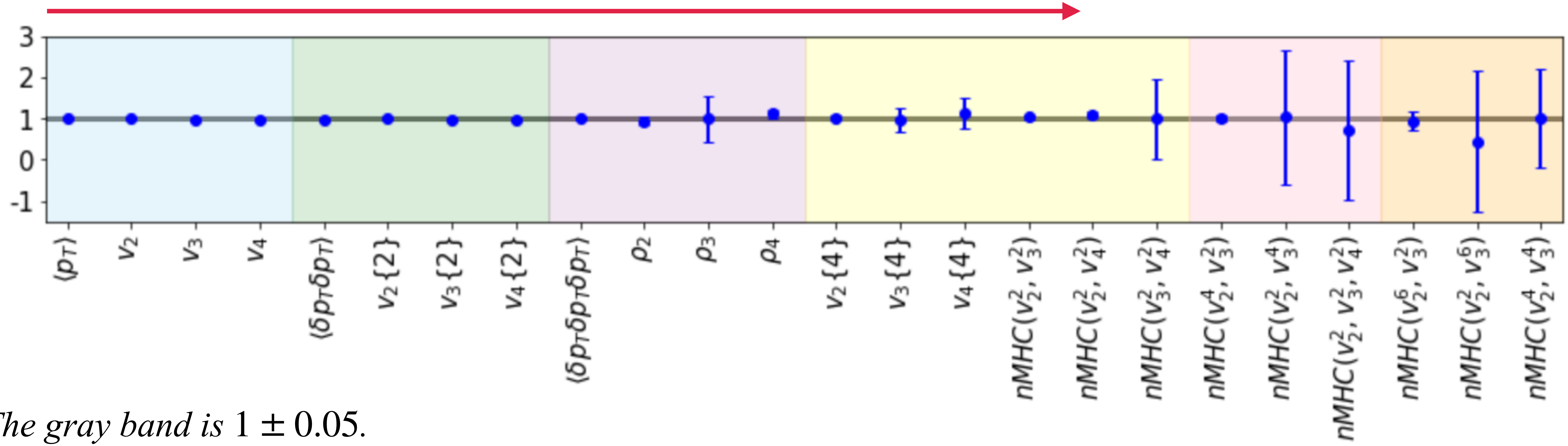
Physical expectation

The comprehensive comparison

Generative

Hydro

The increased number of particles involved (from 1 to 8)



The gray band is 1 ± 0.05.

- These ratios are close to unity, indicating the validity of diffusion model.
- From left to right, model precision decreases systematically as the number of correlated particles increases.

What if parameters are ‘out-of-distribution’?

Xianping Zhong Talk at 10.31

Huge data

Efficient model

Large data

Specific data

ChatGPT moment for heavy-ion collisions

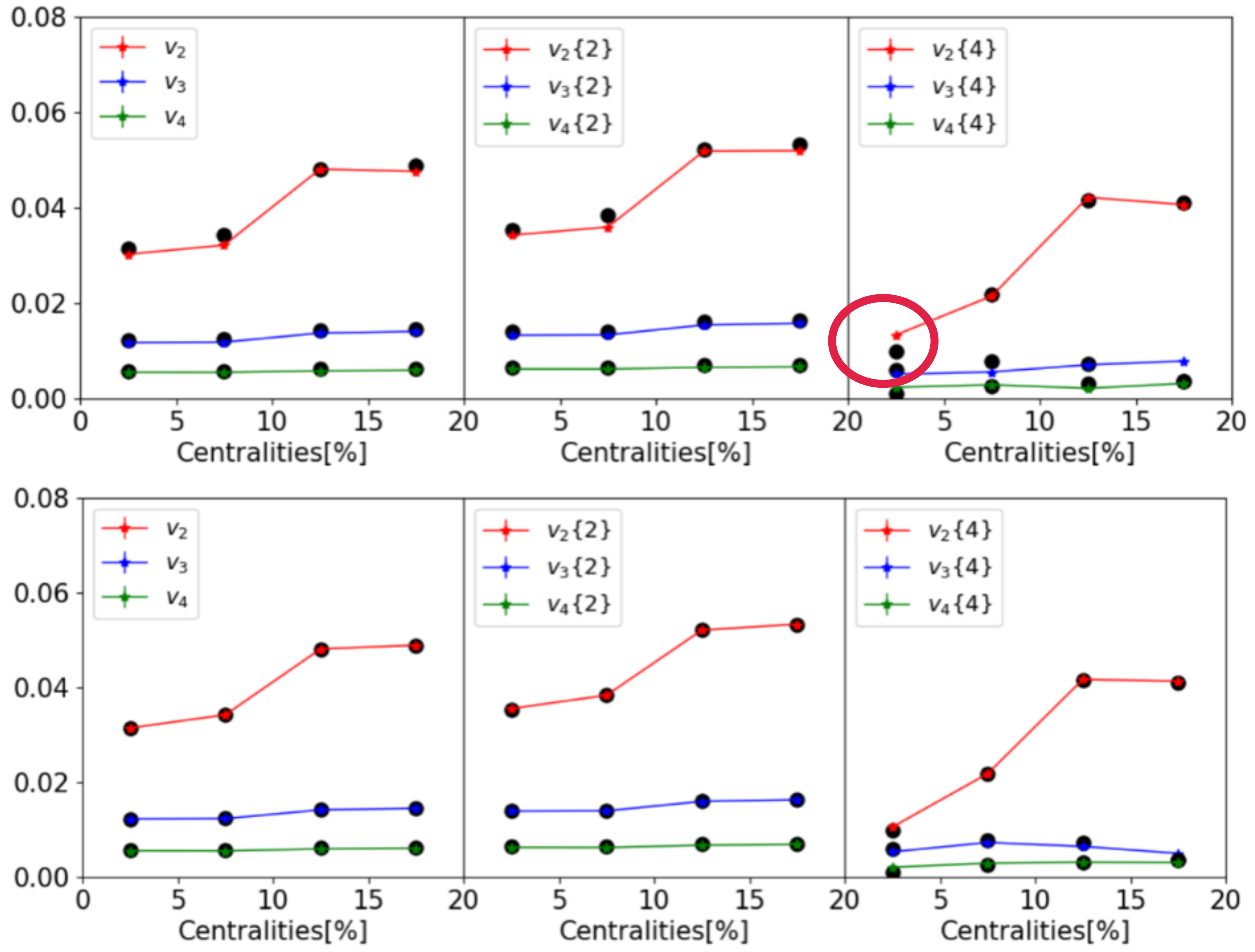
Pretrain-Finetune

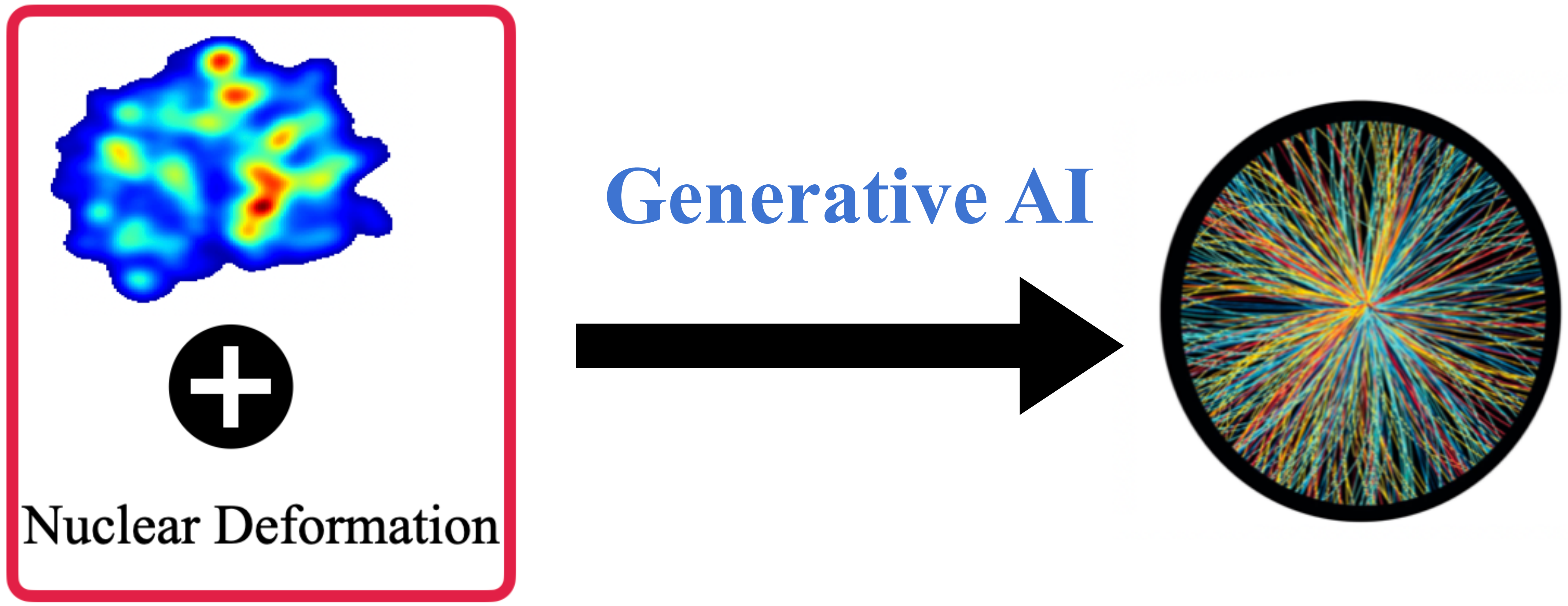
Not easy.

More pratical.

Extension parameter set with fine-tuning

The nuclear deformation can be captured in Generative-AI.





STAR Nature 635 (2024) 8037, 67-72

Symbols: Hydro
Lines: Generative AI

- The first row shows the results form pre-trained model
- The second row shows the results from fine-tuned model with new **500** events.

Scale the model to more parameters and systems

Parameters:

AuAu@200GeV

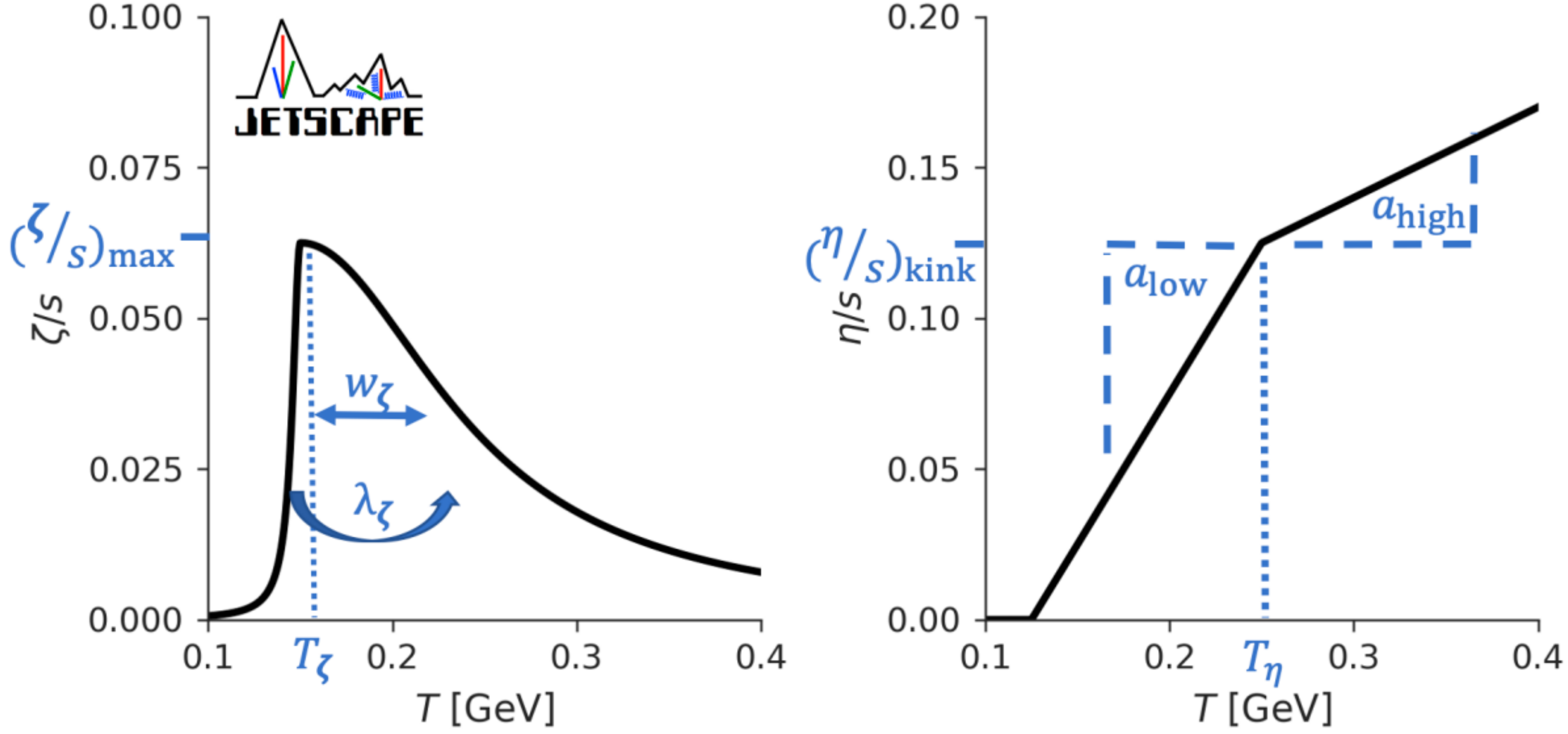
α: 0.45,0.50,0.55

β_2:0.12,0.14,0.16

PbPb@2760GeV

α: 0.45,0.50,0.55

β_2:0.00,0.05,0.10



Totally 100,000 events

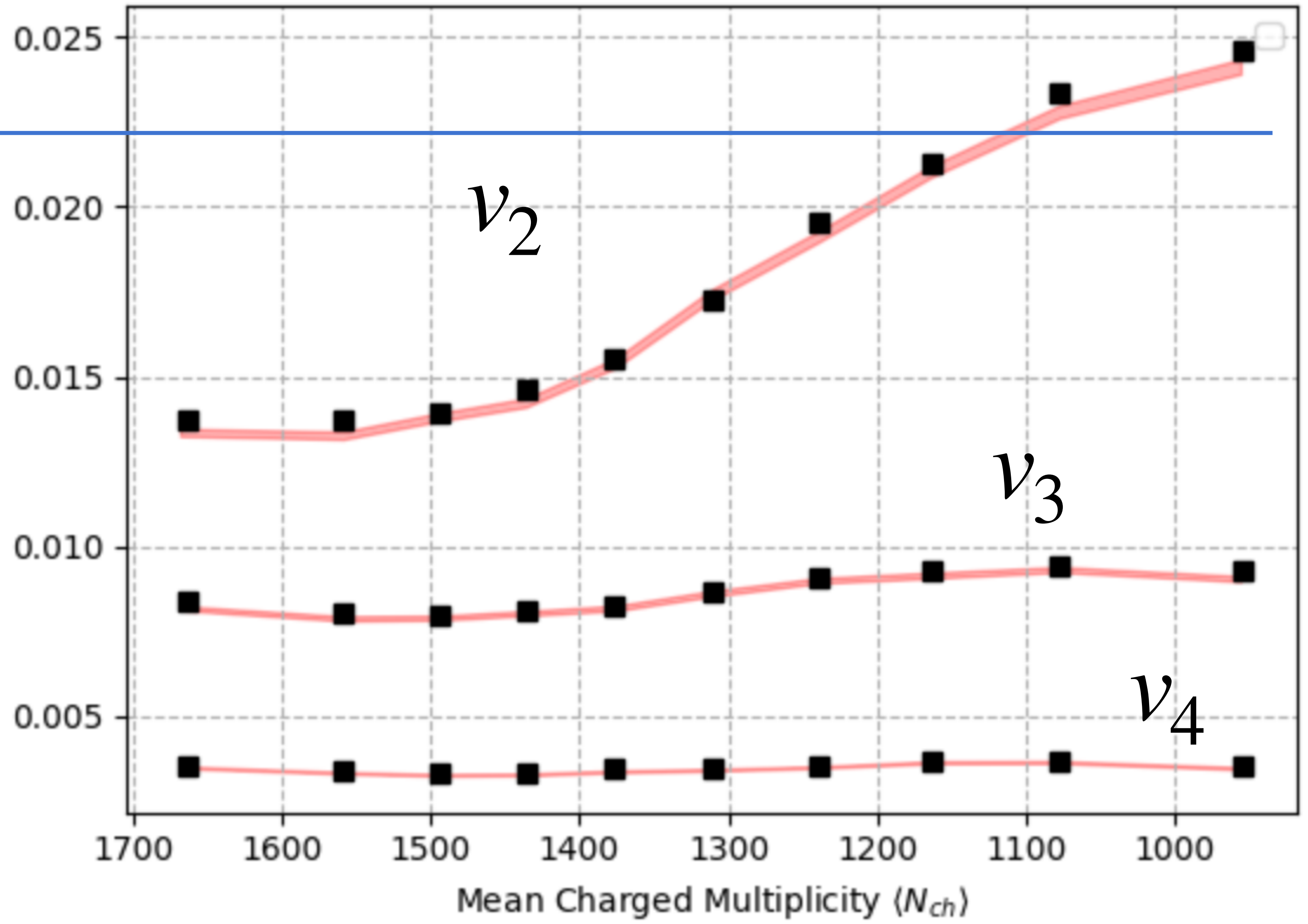
$$(\zeta/s)_{max}, (\eta/s)_{kink} = [0.01,0.05,0.1,0.15]$$

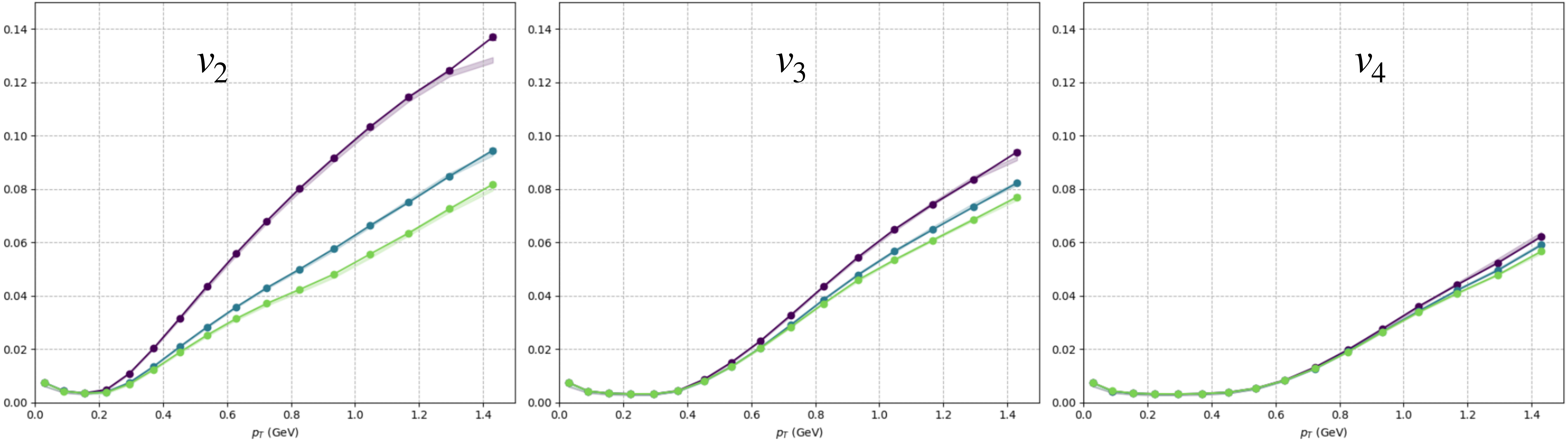
Comparison results

PbPb@2760GeV

$\alpha = 0.546, \beta_2 = 0.075$

$(\eta/s)_{kink} = 0.096, (\zeta/s)_{max} = 0.133$



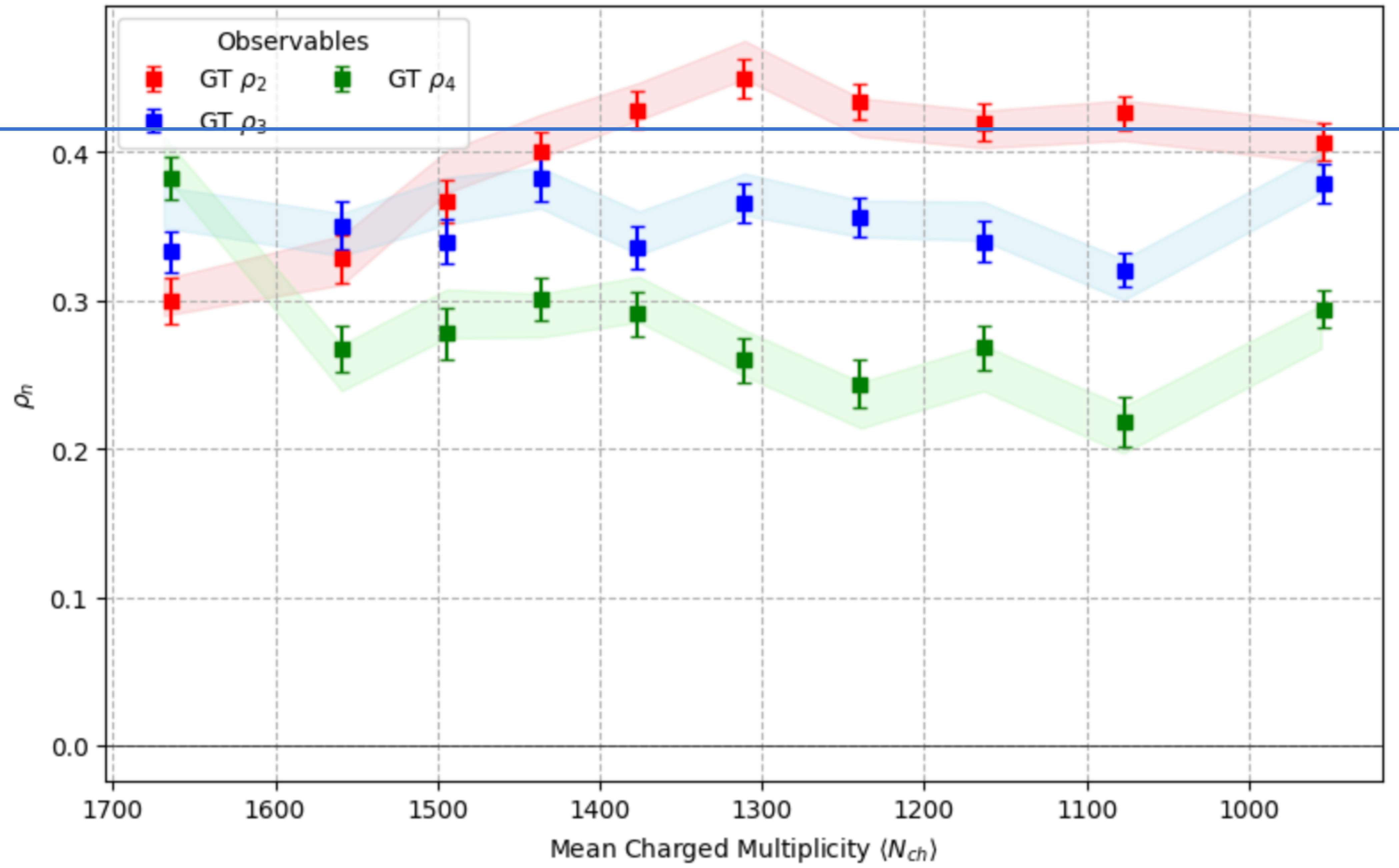


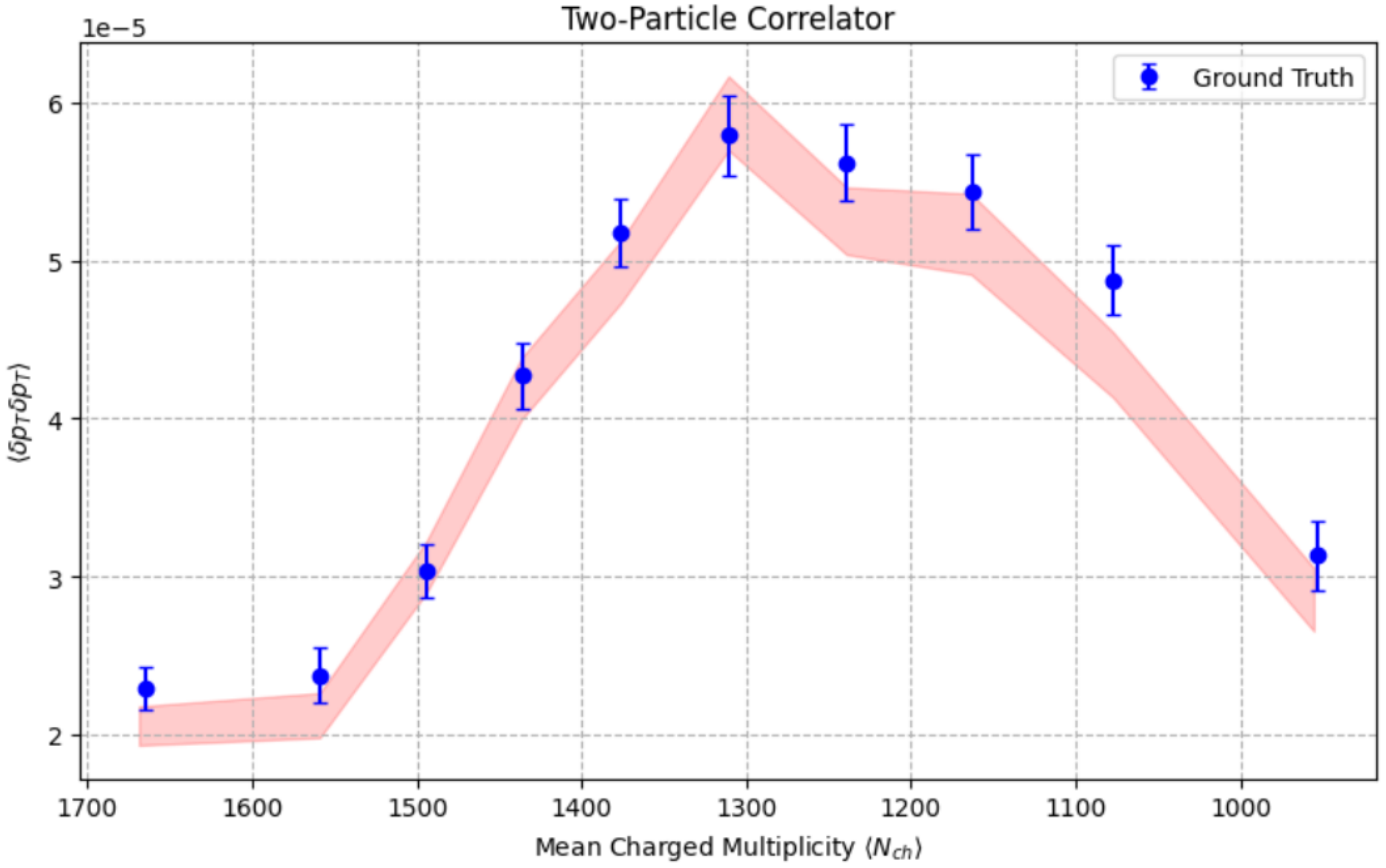
Comparison results

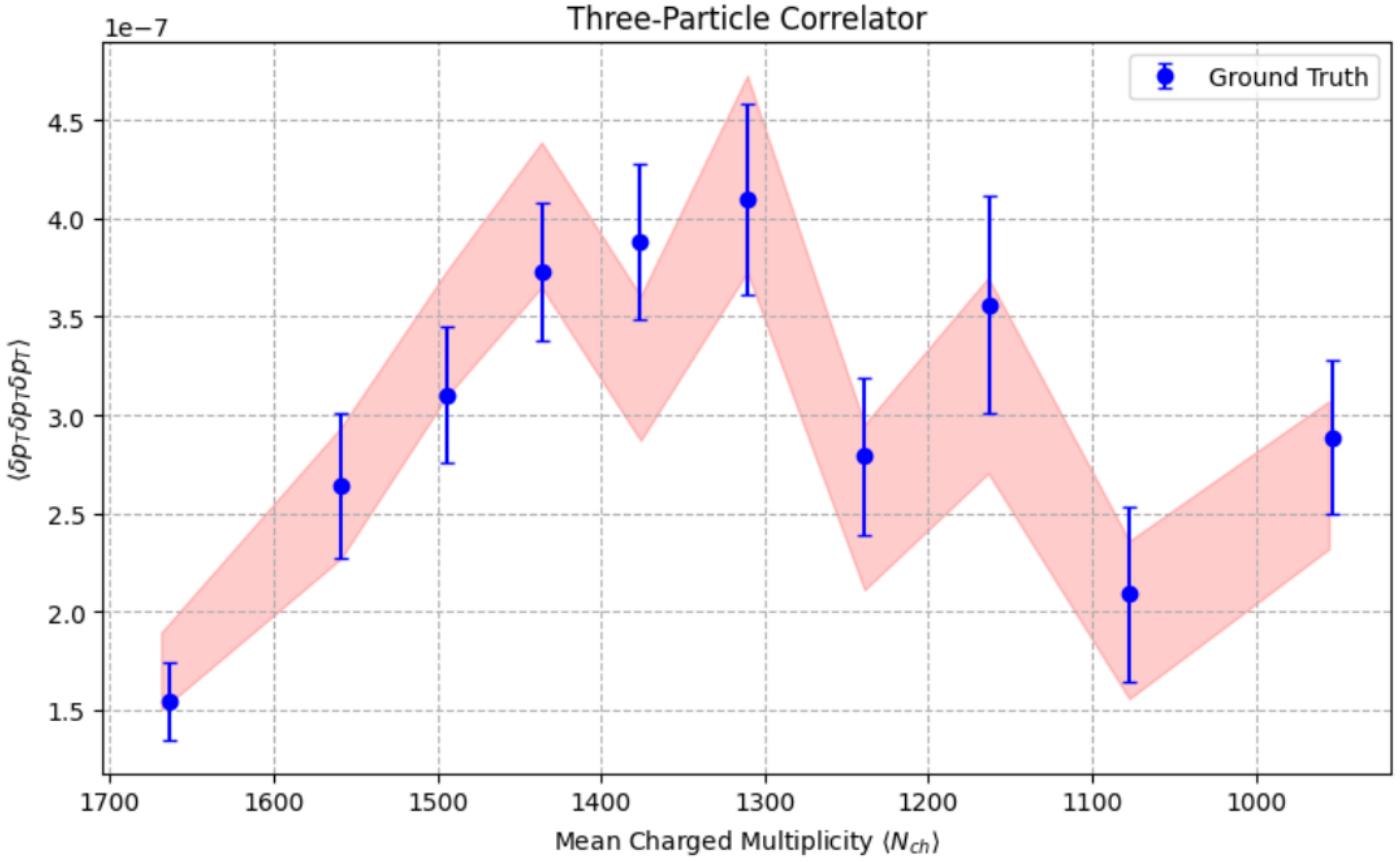
$$\rho_n = \frac{\langle v_n^2\{2\}\langle p_T\rangle\rangle - \langle v_n^2\{2\}\rangle\langle\langle p_T\rangle\rangle}{\sigma_{v_n^2\{2\}}\sigma_{\langle p_T\rangle}}$$

$$\langle \delta p_T \delta p_T \rangle = \left\langle \frac{\sum_{i\neq j}(p_i - \langle\langle p_T\rangle\rangle)(p_j - \langle\langle p_T\rangle\rangle)}{N_{ch}(N_{ch}-1)} \right\rangle_{\text{ev}}$$

$$\langle \delta p_T \delta p_T \delta p_T \rangle = \left\langle \frac{\sum_{i\neq j\neq k}(p_i - \langle\langle p_T\rangle\rangle)(p_j - \langle\langle p_T\rangle\rangle)(p_k - \langle\langle p_T\rangle\rangle)}{N_{ch}(N_{ch}-1)(N_{ch}-2)} \right\rangle_{\text{ev}}.$$







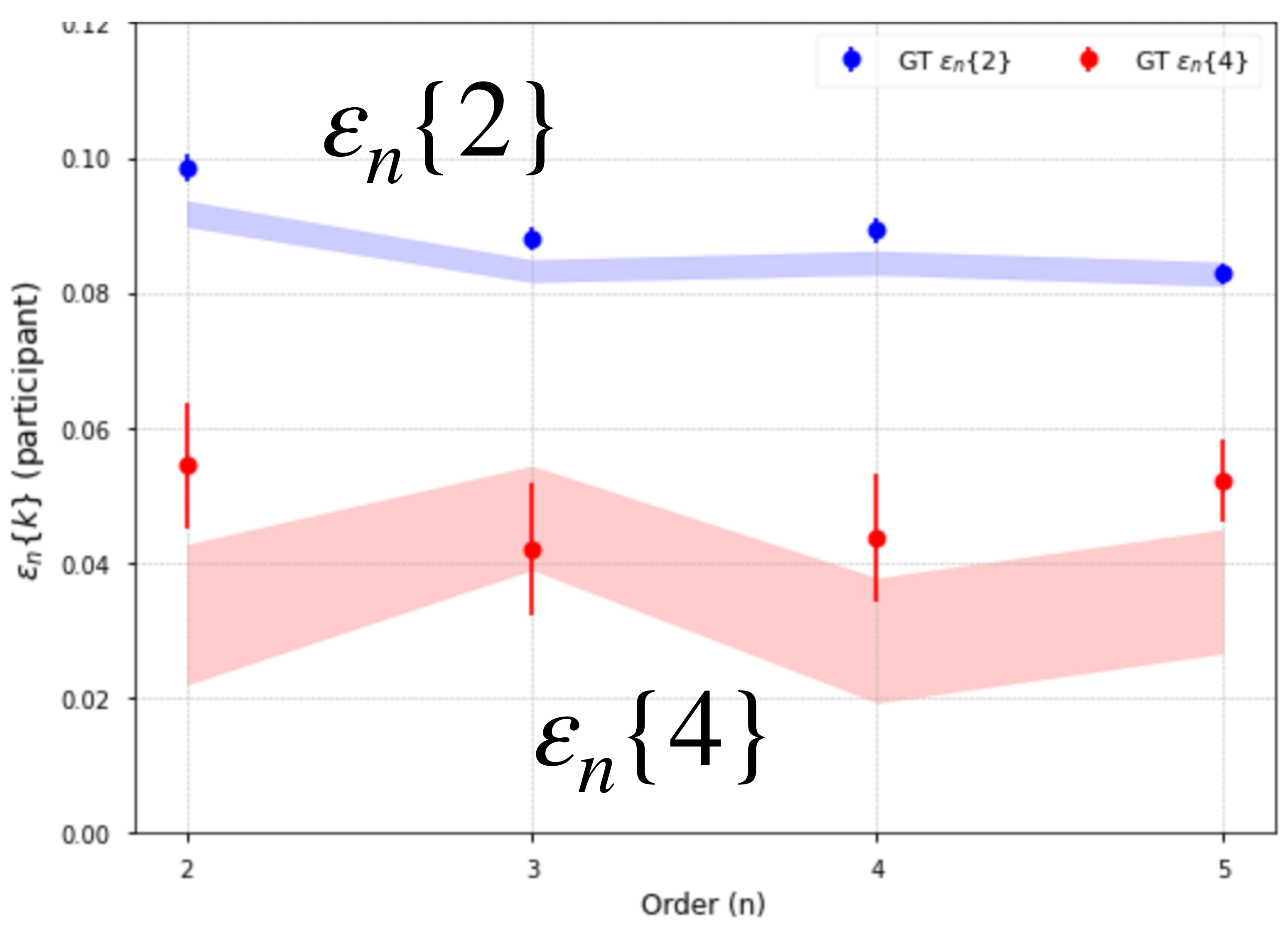
Inverse problem

Inverse: physical entropy decrease and information entropy increase.

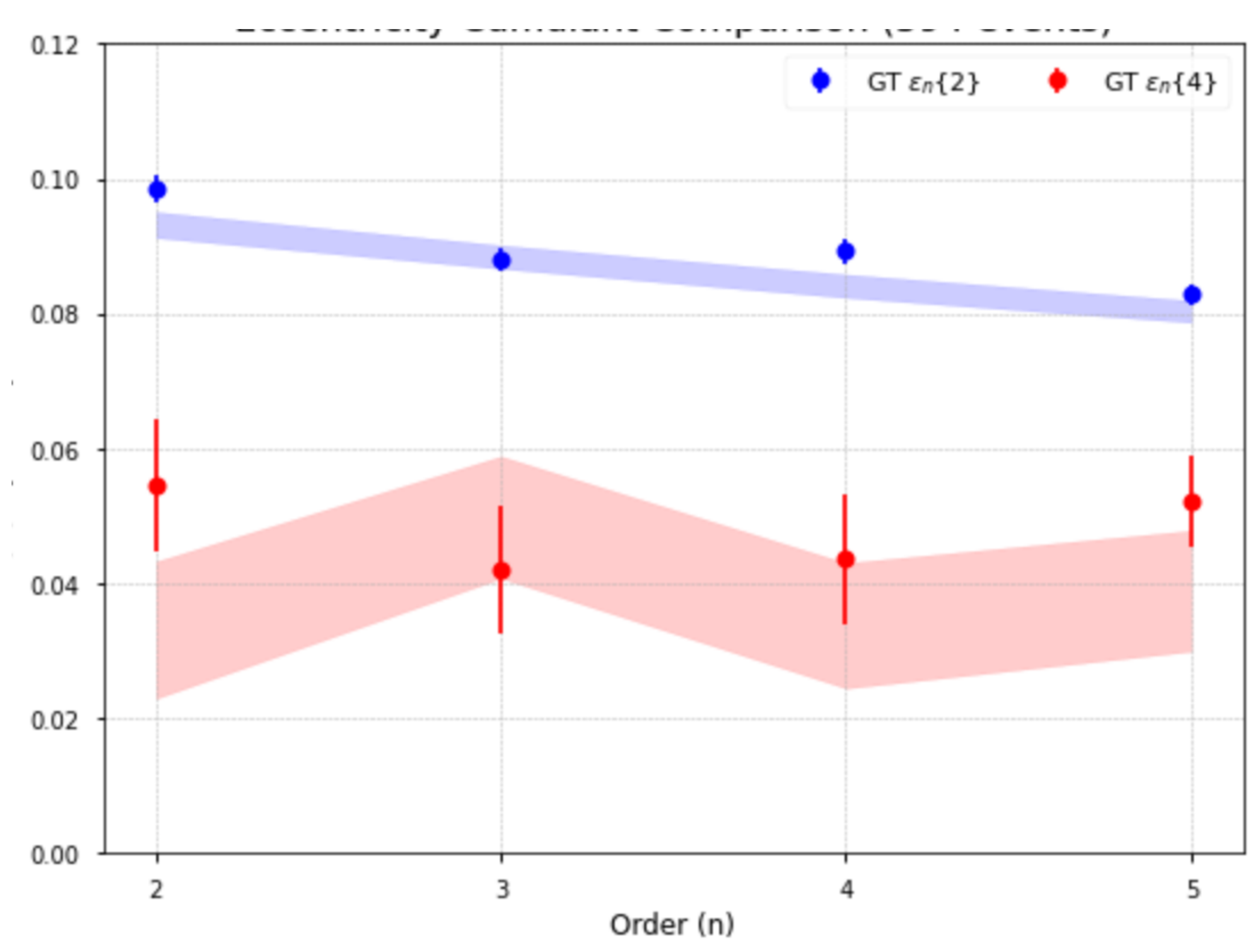
Eccentricity comparison

- ~5% error
- Gen1: Reverse the **hydro part**. Best performance
- Gen2+Gen1: Reverse the UrQMD and hydro parts in a **cascade** way.
- Gen3: Reverse the **(UrQMD + hydro)** directly. Worst performance.

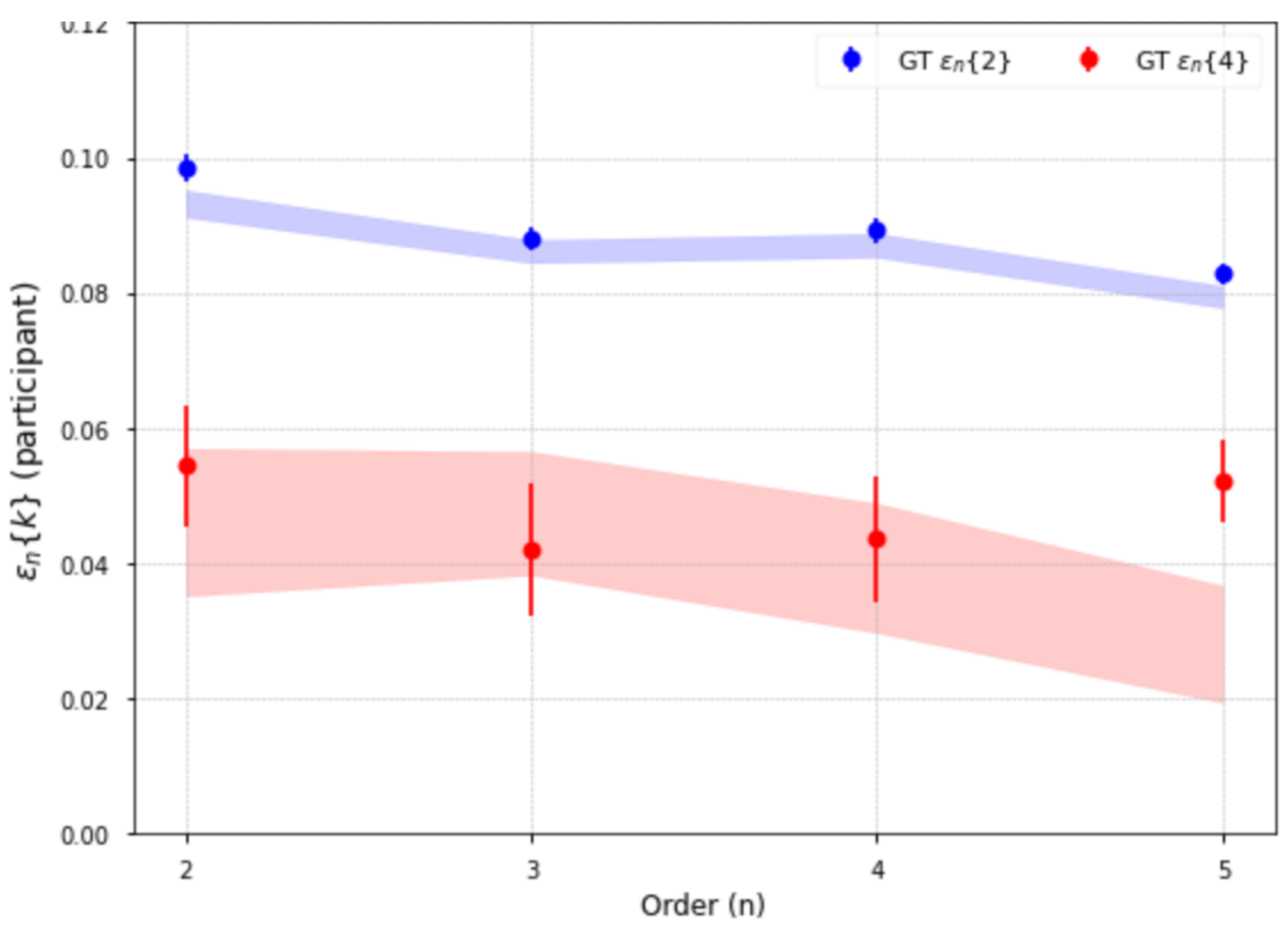
Gen3



Gen2+Gen1

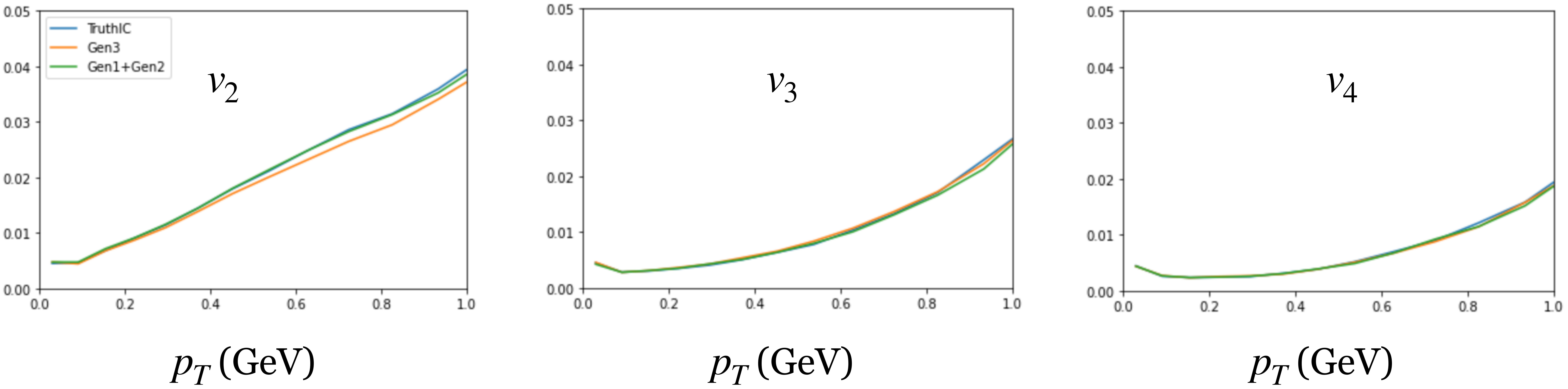


Gen1



Feed the generated initial profile to the emulator

No strong difference.

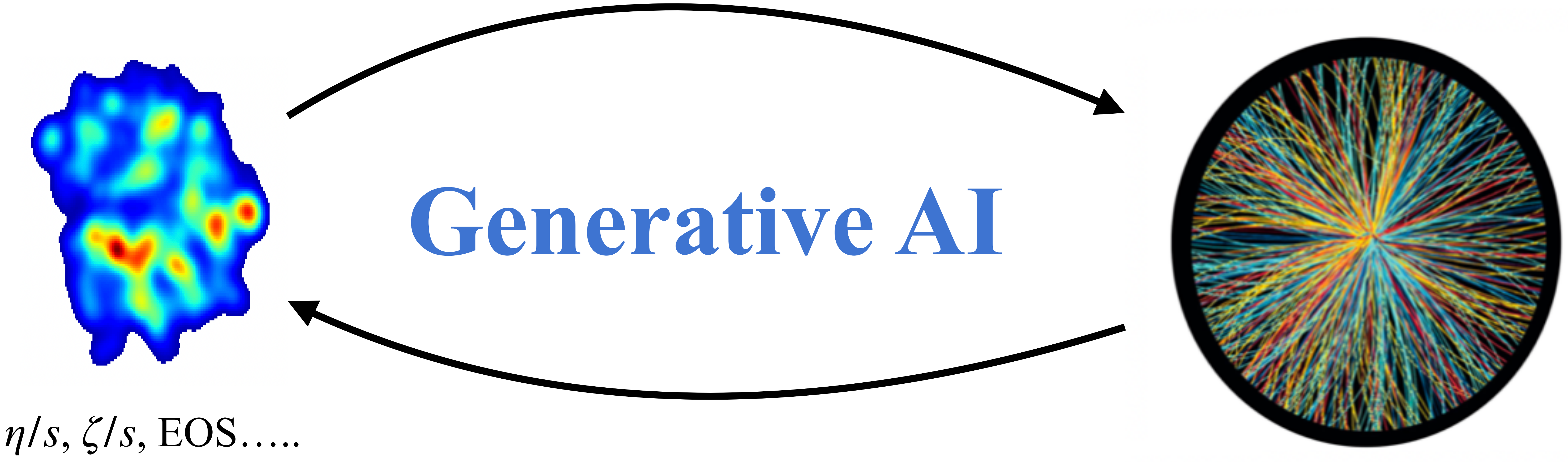


Forward process: physical entropy increase and information entropy decrease.

Summary and outlook

- **End-to-end generation**
- **~10^5× speedup**
- **2D particle spectra are well captured**

- **Scale or Finetune towards 'out-of-distributions'**
- Towards to **3D** hydrodynamic simulation and **particle cloud generation.**



- Inverse problems: if machine can learn a physically forbidden reverse dynamic process?
  - Training optimization: embed the physical dynamics…
  - Do diffusion in the latent space

…..

Thanks for your attention.

The generative models map

Kai Zhou Talk at 11.01

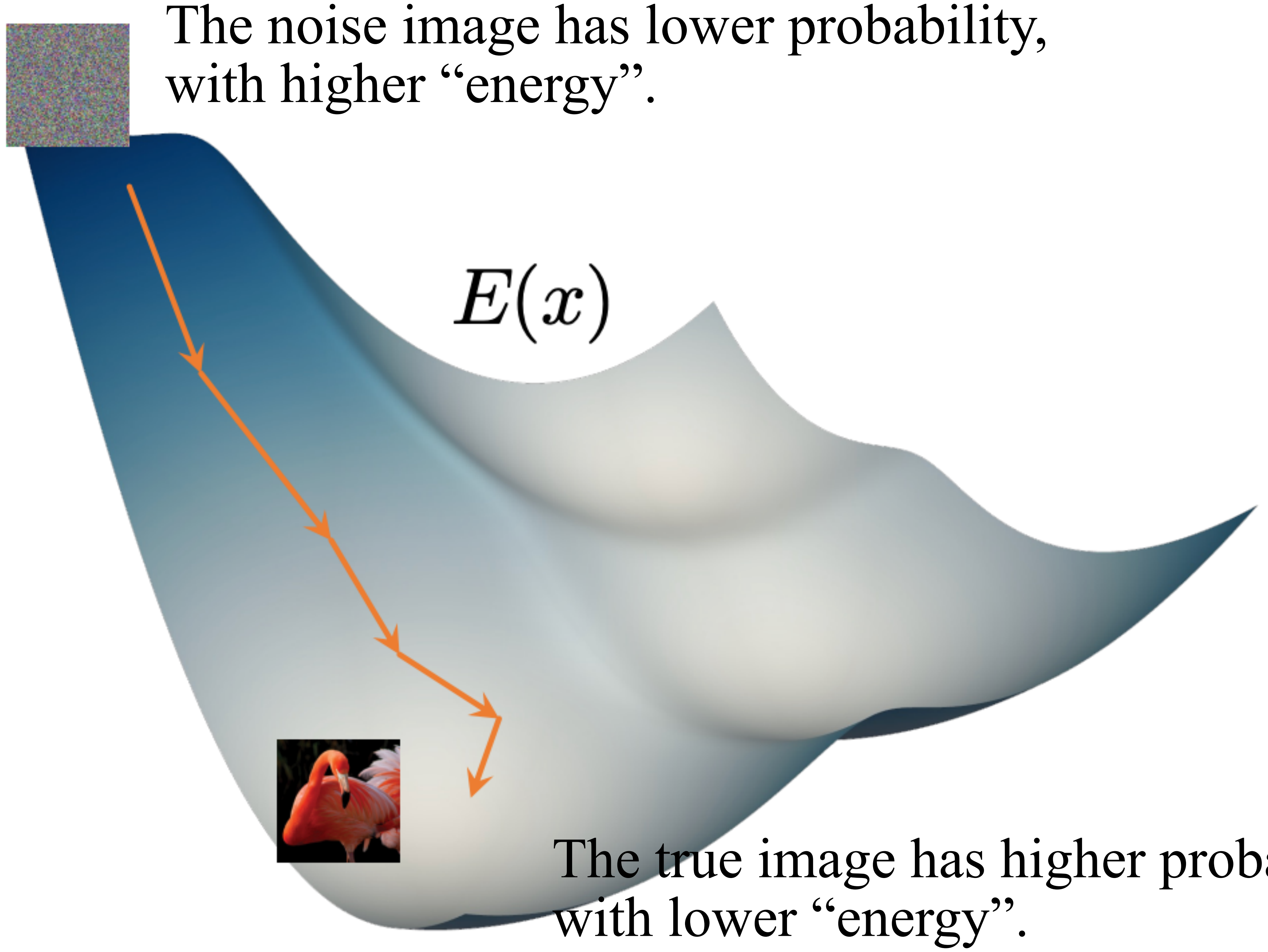
Distribution transformation formalism

Model data distribution using energy function

- Energy-based model training and inference involves **the large numbers of sampling** to estimate the averaged value.
- Not to learn the energy function but learn the gradient of the energy function, i.e., **gradient of log-likelihood or score function**

$$p_{\text{data}}(x) = \frac{\exp(-E(x))}{Z}$$

$$\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} \log \frac{\exp(-E(\boldsymbol{x}))}{Z} = -\nabla_{\boldsymbol{x}} E(\boldsymbol{x})$$



The noise image has lower probability, with higher "energy".

The true image has higher probability, with lower "energy".

- With the score function, one can perform sampling via **Langevin dynamics**

$$d\boldsymbol{x} = -\nabla_{\boldsymbol{x}} E(\boldsymbol{x})dt + \sqrt{2}dW$$

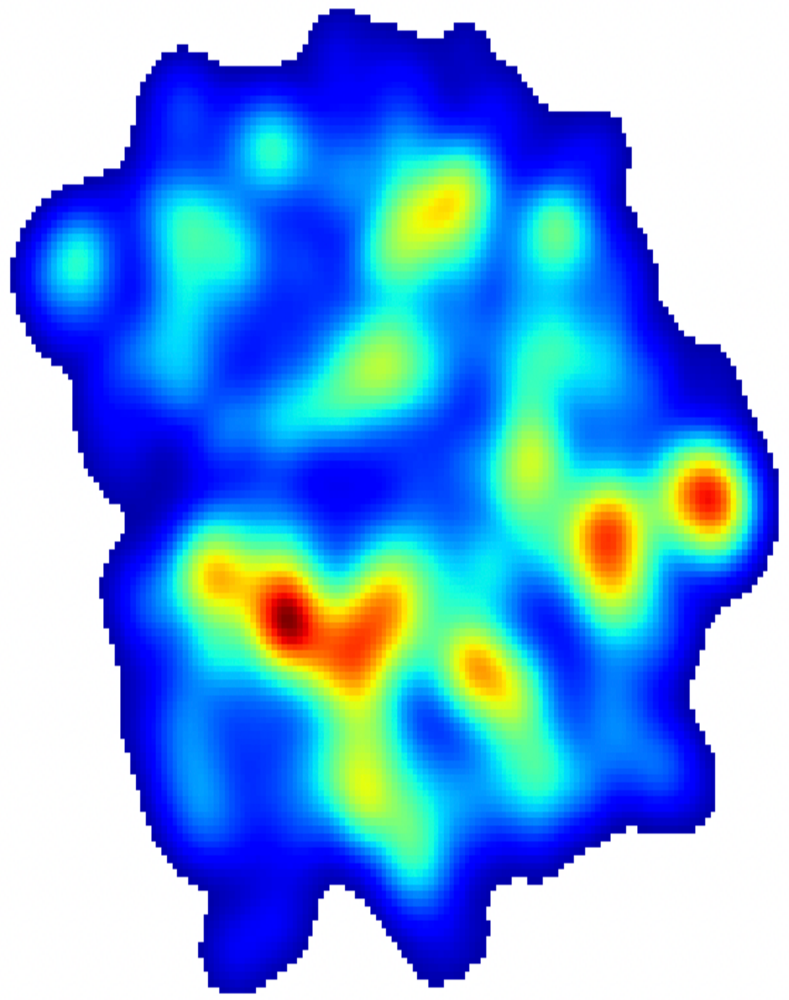
$$\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \Delta t \nabla \log p(\boldsymbol{x}) + \sqrt{2\Delta t}\boldsymbol{\epsilon}$$

How can machine learning help us?

- **Goal:**
  - End-to-end (initial state to final state)
  - Fast and flexible
  - Keeping physical consistency.

- **The challenges in deterministic machine learning**

  Learn the hard map from x to y



deterministic machine learning

- sUnet: Solves hydrodynamic equations but **accumulates errors** in long-time evolution.

- Gaussian emulator in Bayesian analysis: Maps parameters to observables (digital to digital fit) but **lacks flexibility**.

H.F Huang et.al Phys.Rev.Res. 3 (2021) 2, 023256

J.E Bernhard et.al Nature Phys. 15 (2019) 11, 1113-1117

Score matching

- **Explicit score matching**

Intractable

$$\mathcal{L}_t^{\mathrm{ESM}} \equiv \mathbb{E}_{p(\boldsymbol{x}_t)} ||\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \boxed{\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})}||^2$$

- **Denoising score matching**

$$\mathcal{L}_t^{\mathrm{DSM}} \equiv \mathbb{E}_{p(\boldsymbol{x}_t, \boldsymbol{x}_0)} ||\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}_t | \boldsymbol{x}_0)||^2$$

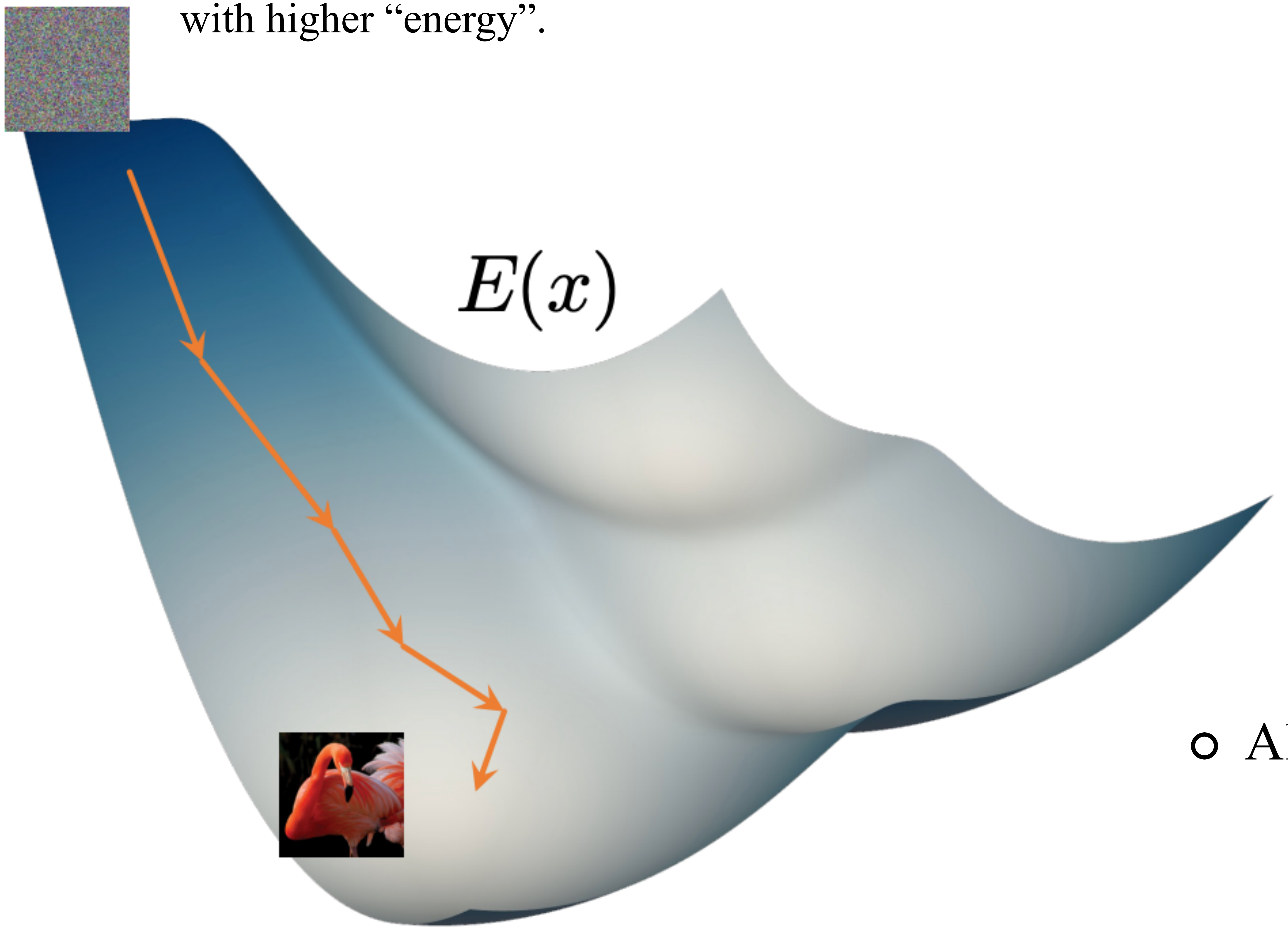
$$\boldsymbol{\theta}^* = \mathrm{argmin}_{\boldsymbol{\theta}} \mathcal{L}_t^{\mathrm{DSM}} = \mathrm{argmin}_{\boldsymbol{\theta}} \mathcal{L}_t^{\mathrm{ESM}}$$

The equivalence is promised by: $p_t(\mathbf{x}) = \int d\mathbf{x}_0 p(\mathbf{x}|\mathbf{x}_0)$

- From the forward SDE, $p_t(x_t | x_0)$ (the noised data x_t conditioned on clean data x_0) follows the gaussian distribution.

Model data distribution using energy function

The noise image has lower probability,
with higher "energy".



The true image has higher probability,
with lower "energy".

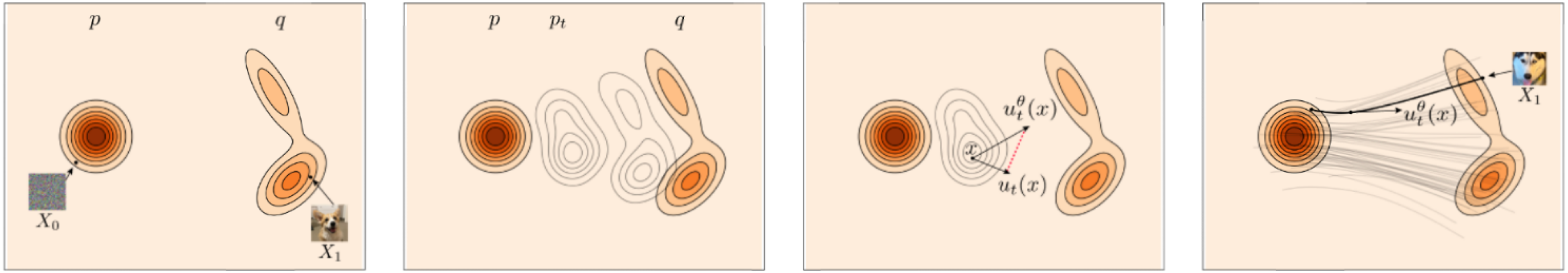
$$E(\boldsymbol{x}) = -\sum_{i} a_i x_i - \sum_{ij} \frac{J_{ij}}{2} x_i x_j$$

$$p_{\text{data}}(x) = \frac{\exp(-E(x))}{Z}$$

- Along the path minimizing the energy function

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]\mathrm{d}t$$

$$dx_t = u_t(x_t)dt$$



(a) Data. **(b)** Path design. **(c)** Training. **(d)** Sampling.

AuAu@200GeV

$\alpha = 0.535, \beta_2 = 0.098$

$(\eta/s)_{kink} = 0.096, (\zeta/s)_{max} = 0.133$