ML4Jets2026 on 09.16.2026
"Foundation Models Session"
University of Vienna, Austria

The Ψai Lab
Curiosity, Creativity, Credibility

What can representation geometry reveal about collider foundation models?

— *Towards Scientific AI as a New Microscope*

Authors:

Tianji Cai,
Anna Hallin,
Shivasankar K.A.,
Michael Krämer.

Presented by Tianji Cai

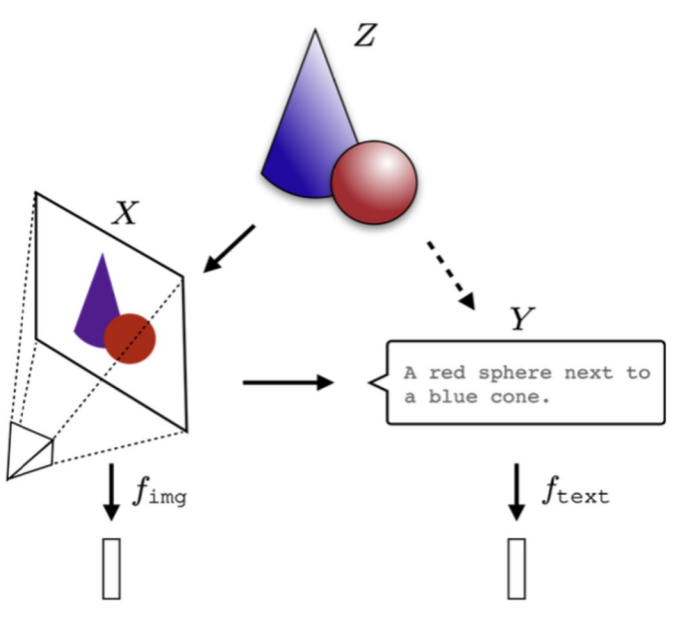
School of Physical Science & Engineering
Tongji University, Shanghai, China

The “Platonic Hypothesis” for us?

Inspiration from our CS colleagues:
2405.07987

The Platonic Representation Hypothesis

Neural networks, trained with different objectives on different data and modalities, are converging to a shared statistical model of reality in their representation spaces.

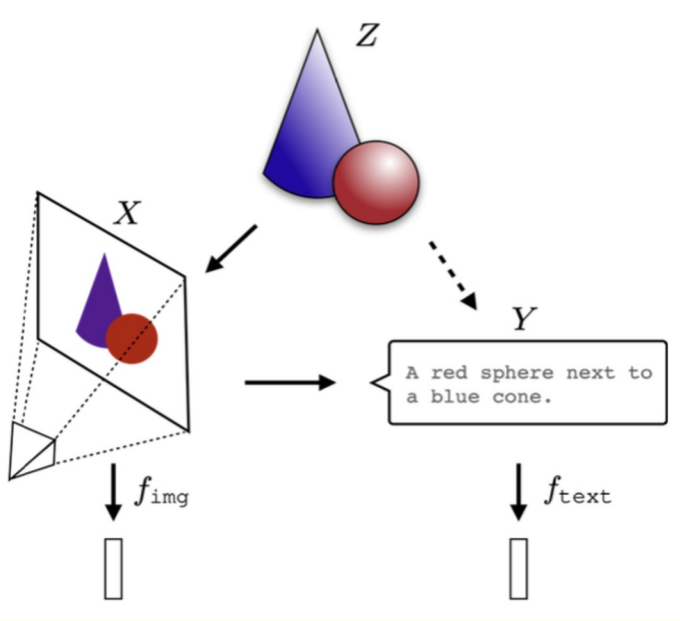


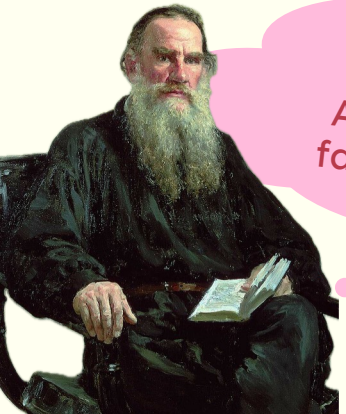
The “Platonic Hypothesis” for us?

Inspiration from our CS colleagues:
2405.07987

The Platonic Representation Hypothesis

Neural networks, trained with different objectives on different data and modalities, are converging to a shared statistical model of reality in their representation spaces.





Anna Karenina Principle:

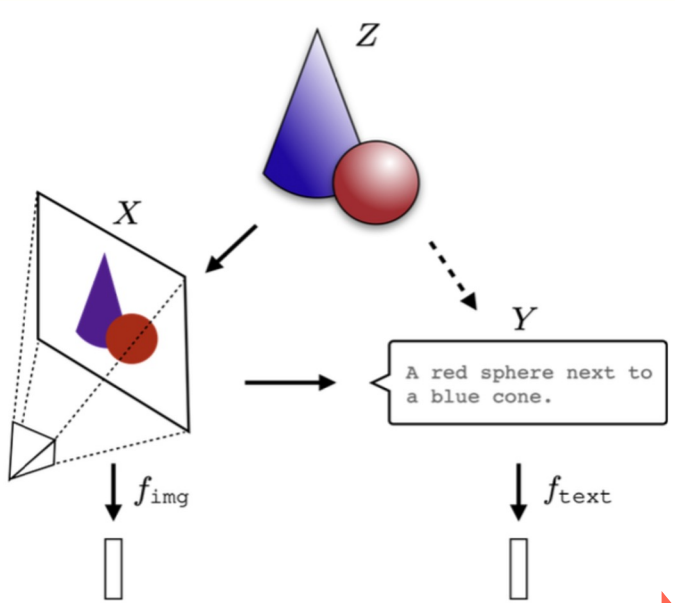
All good models are alike; each failed model fails in its own way.

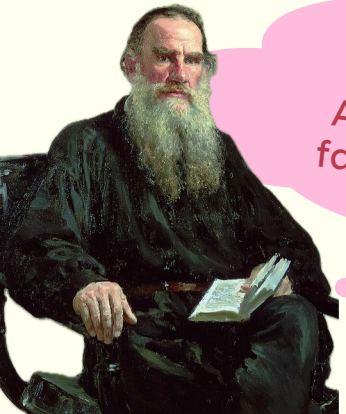
The “Platonic Hypothesis” for us?

Inspiration from our CS colleagues:
2405.07987

The Platonic Representation Hypothesis

Neural networks, trained with different objectives on different data and modalities, are converging to a shared statistical model of reality in their representation spaces.





Anna Karenina Principle:

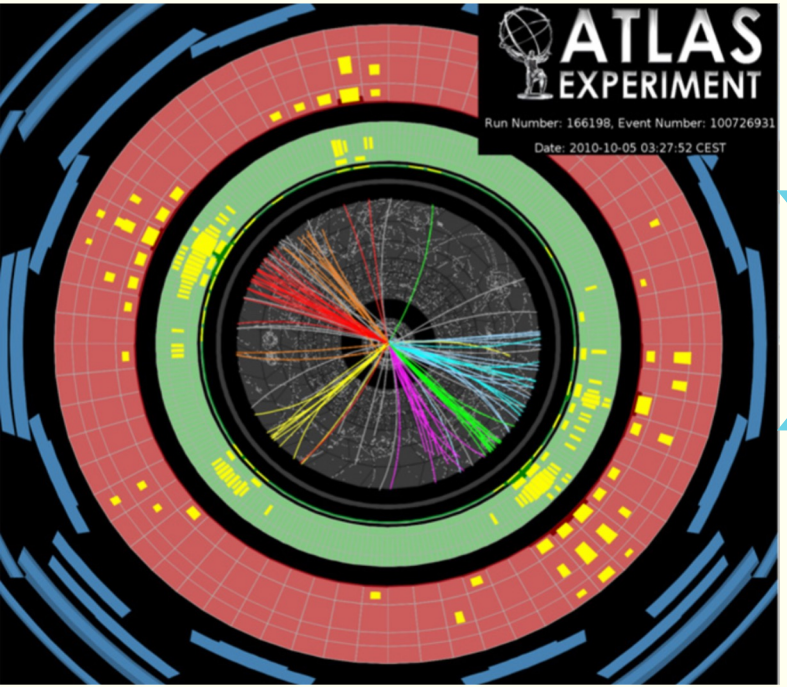
All good models are alike; each failed model fails in its own way.

The “underlying reality” of our particle data is far richer than texts/images. It’s *Nature* itself!

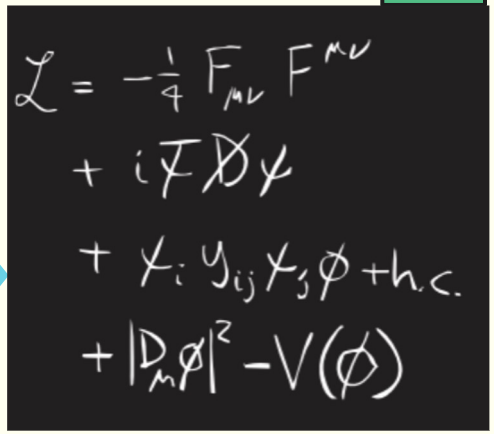
What then can our model representation spaces uncover?

Scientific(Ψ) AI as a New Microscope for Particle Physics

Scientific(Ψ) AI as a New Microscope for Particle Physics



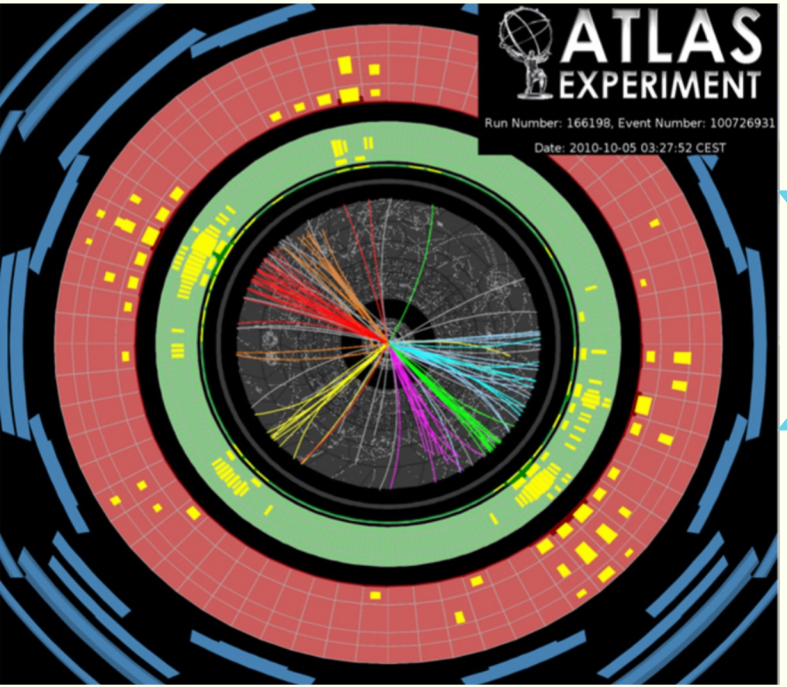
Good NN Models for Collider Data Analysis



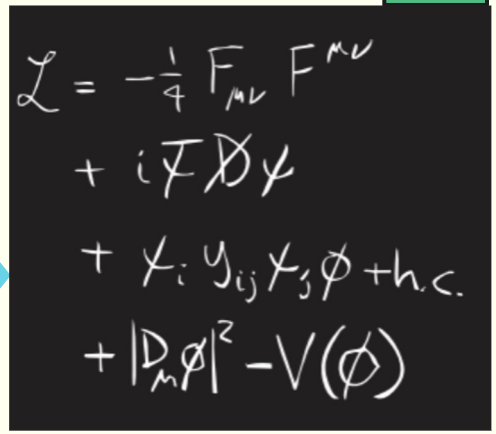
Now performance is a **prerequisite**, not the focus. Models themselves are our **discovery tool**.

Or perhaps something completely alien?

Scientific(Ψ) AI as a New Microscope for Particle Physics



Good NN Models for Collider Data Analysis



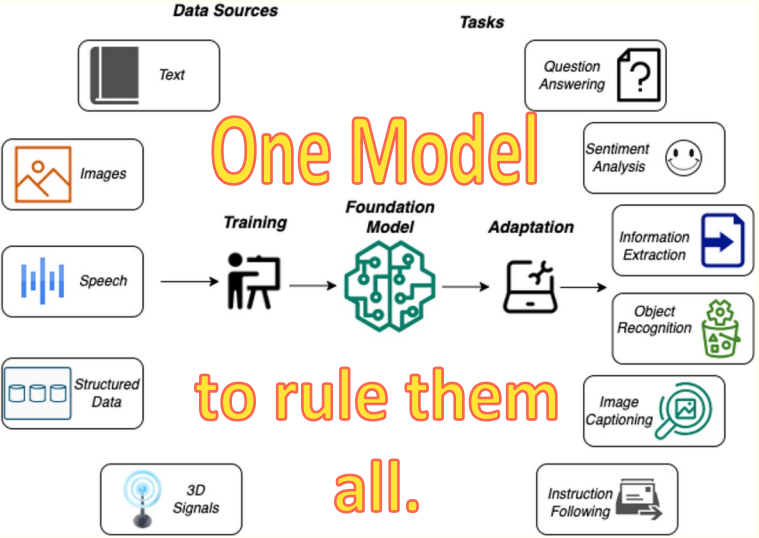
Now performance is a **prerequisite**, not the focus. Models themselves are our **discovery tool**.

Or perhaps something completely alien?

Before putting the “microscope” to use on our particle data, we first need to study the “properties” of our instrument in a more rigorous and systematic way.

→ **GOAL OF THIS WORK!**

So, what are our good models?

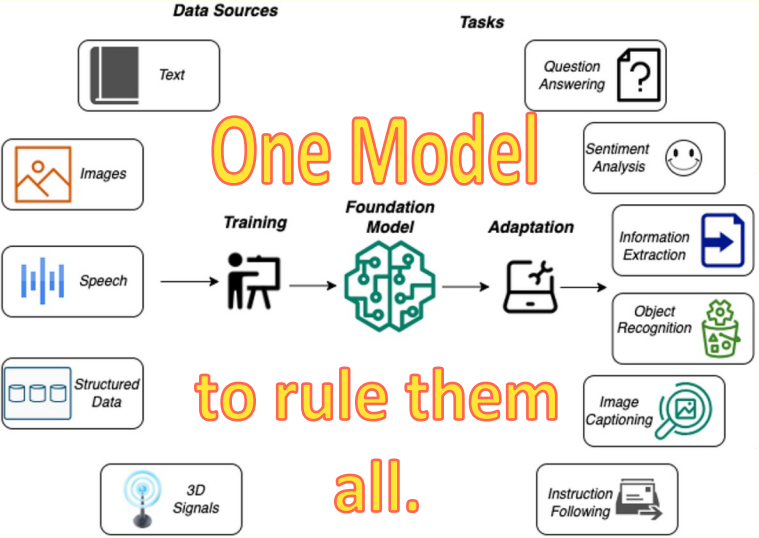


Foundation Models (FMs) for Jets:

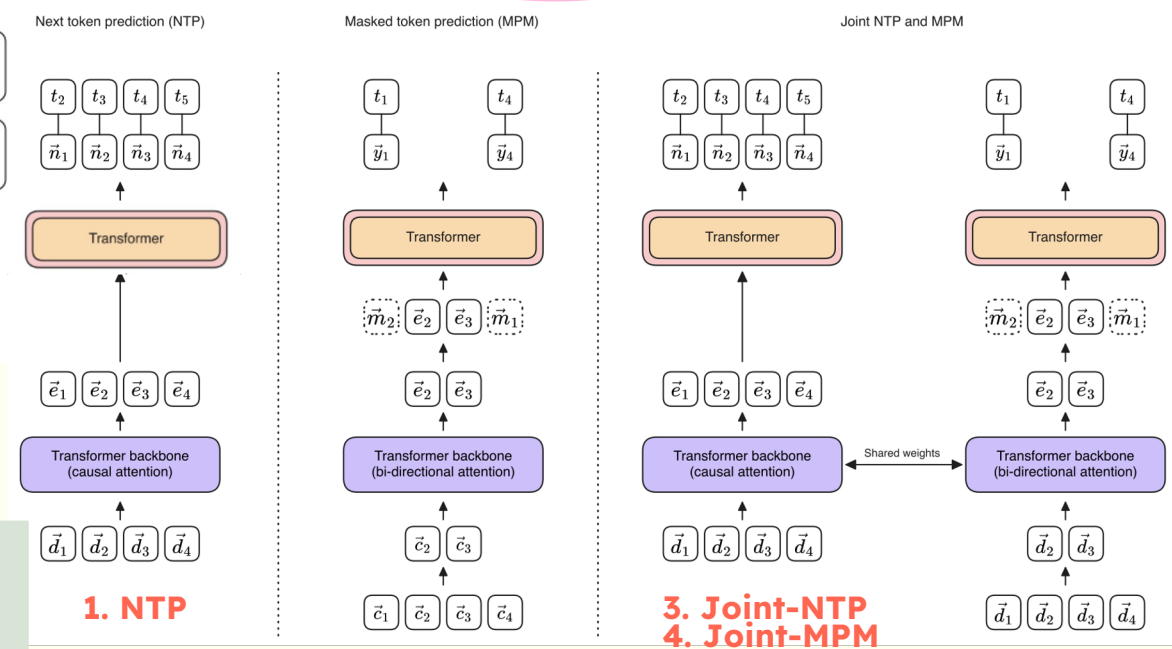
- OmniJet-α
- OmniLearn
- J-JEPA
- HEP-JEPA
- …

Presumably better at many downstream tasks, including jet tagging, jet generation, unfolding, anomaly detection, etc.

So, what are our good models?



Here we focus on the **OmniJet** family, which has **3** different **pre-training objectives**.



Foundation Models (FMs) for Jets:

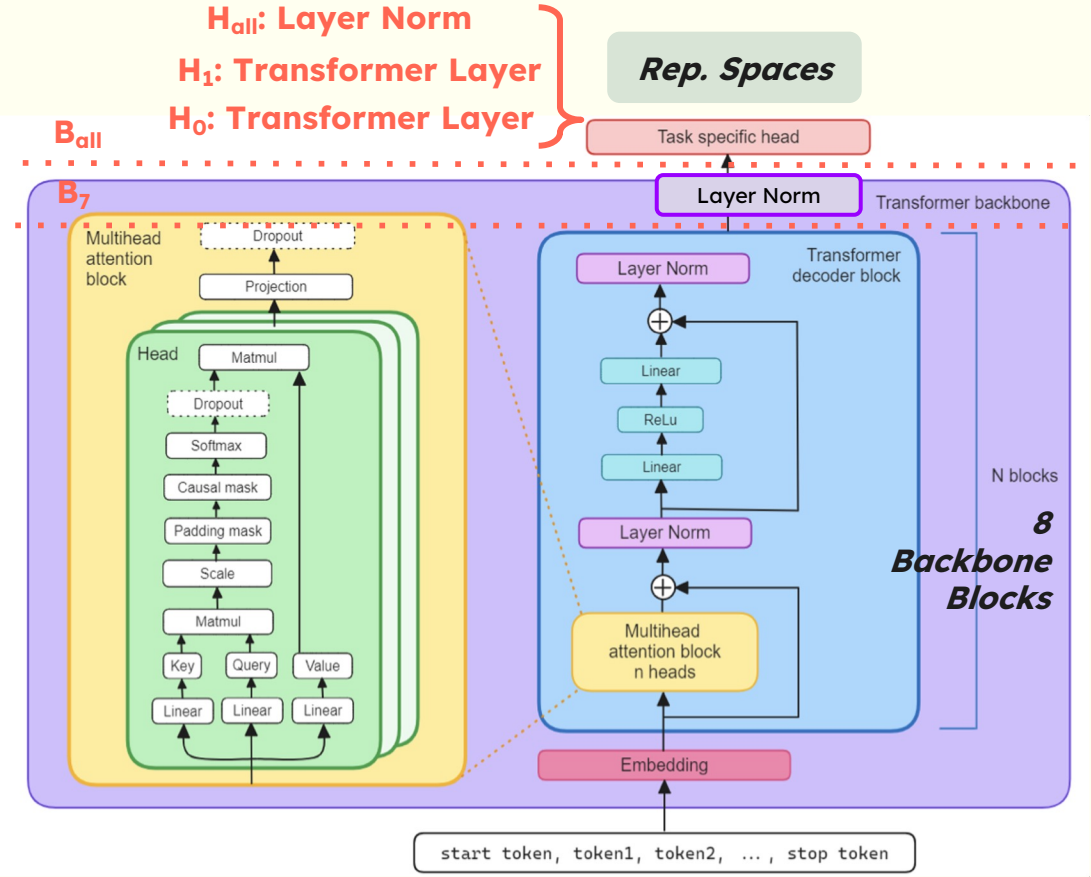
- OmniJet-α
- OmniLearn
- J-JEPA
- HEP-JEPA
- …

Presumably better at many downstream tasks, including jet tagging, jet generation, unfolding, anomaly detection, etc.

For more info about the OmniJet models, please refer to 2512.04149 & 2403.05618.

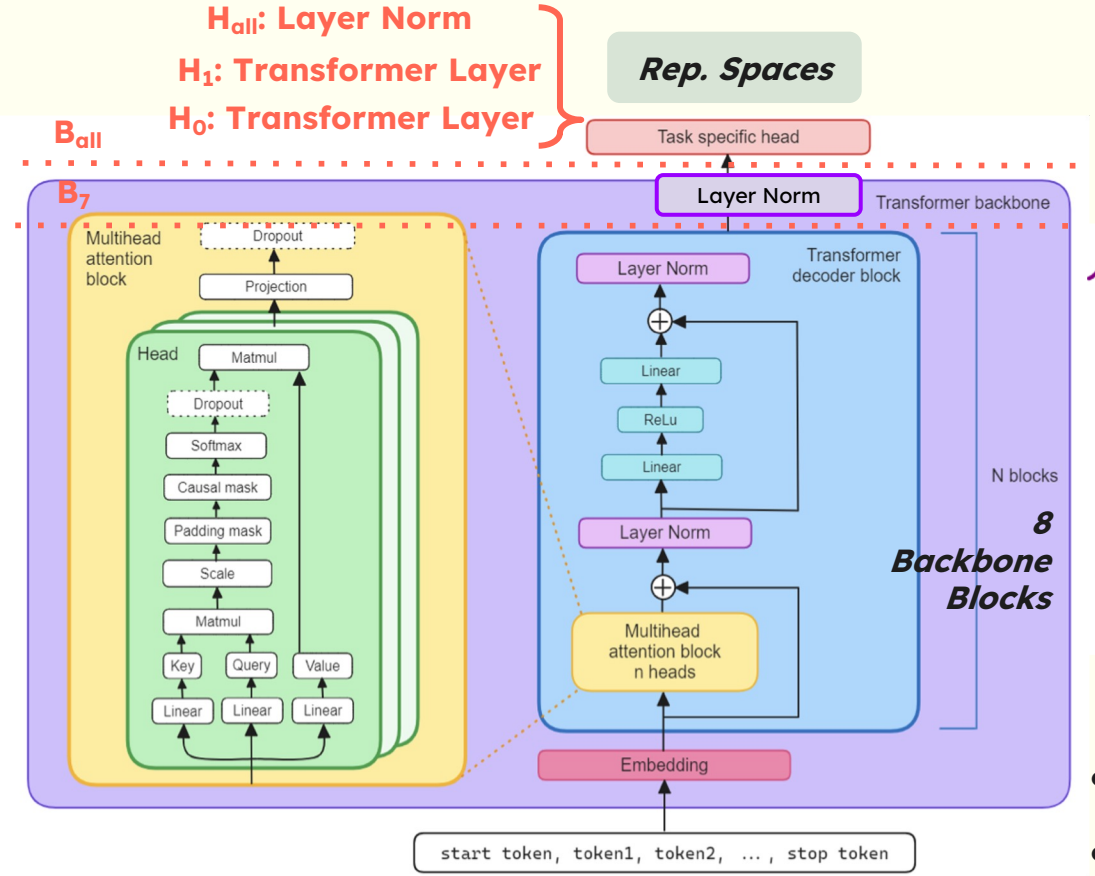
How do we extract & analyze the representation spaces?

How do we extract & analyze the representation spaces?



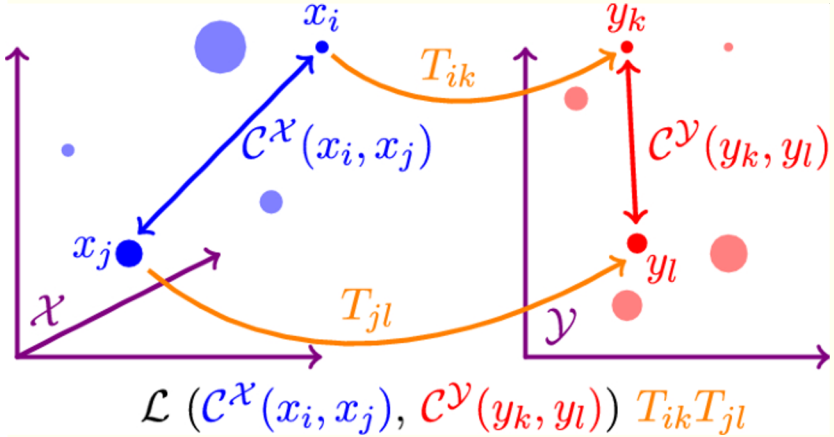
To construct rep. spaces, we extract the activation after each **backbone block (8; B_0-B_7)** & its final **norm layer (1; B_{all})**, and each **head layer (2; H_0-H_1)** & its final **norm layer (1; H_{all})**.

How do we extract & analyze the representation spaces?



To construct rep. spaces, we extract the activation after each **backbone block (8; B_0-B_7)** & its final **norm layer (1; B_{all})**, and each **head layer (2; H_0-H_1)** & its final **norm layer (1; H_{all})**.

The Gromov-Wasserstein Distance (GW) to compare the intrinsic geometric structure of different rep. spaces.



minimize over all (i, k)(j, l) combinations

- 500 vs 500 jets (uniform weight) at inference for rep. spaces X/Y.
- Average particle rep. vectors to get jet rep. vector.
- Use cosine distance between jet rep. vectors in the rep. space, i.e., $C(x_i, x_j) = 1 - x_i \cdot x_j / \|x_i\| \|x_j\|$. L is the square loss.

Note: 3 seeds per model (error bands) and 4 sets of 500 jets (too small to notice) are always used to estimate result variations.

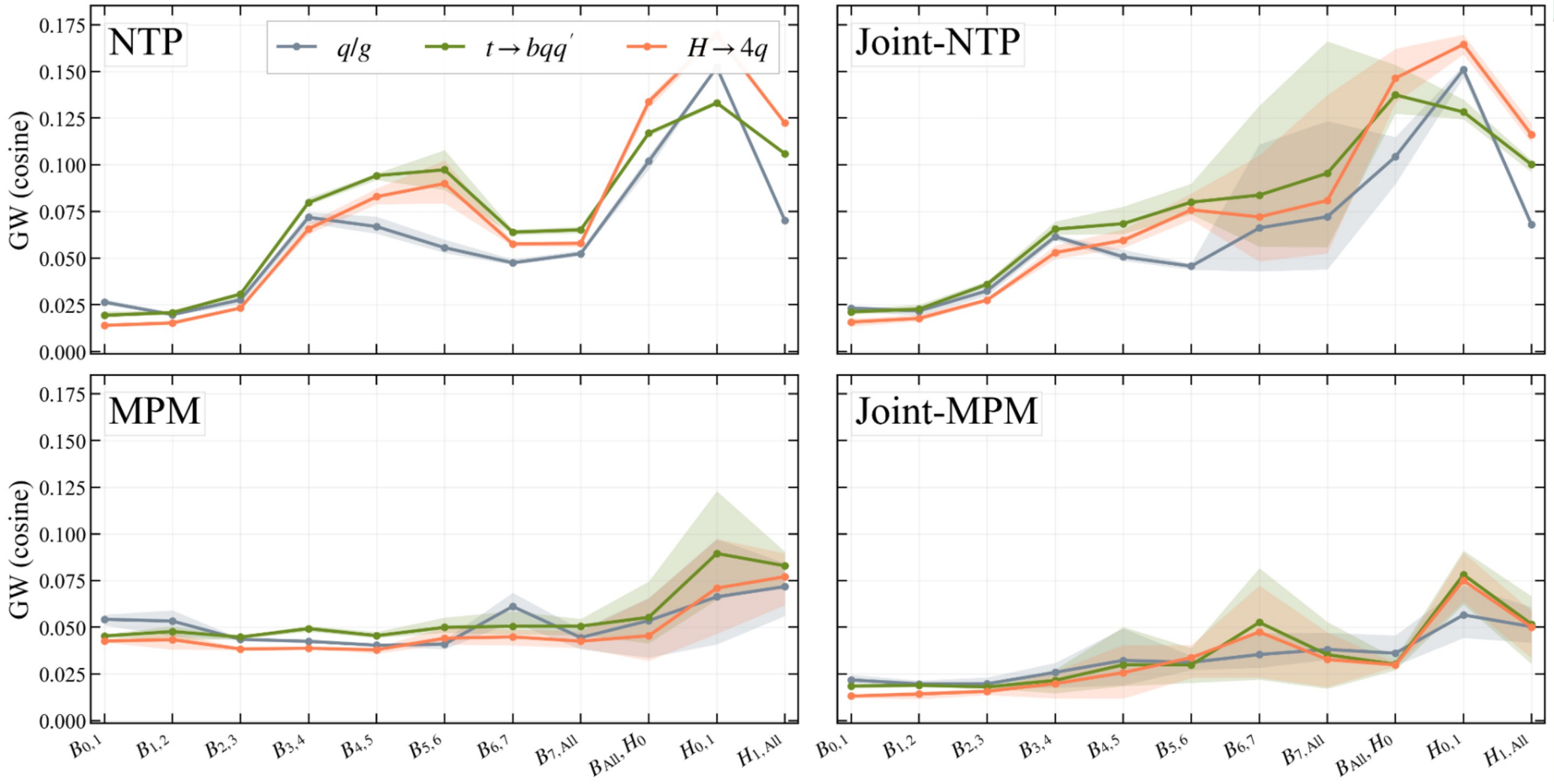
Q1 *How does the rep. space of the same model change w.r.t the internal model blocks at the end of training?*

End of Training: 2.7 times the entire JetClass.

Preliminary results only; Proceed with caution ⚠

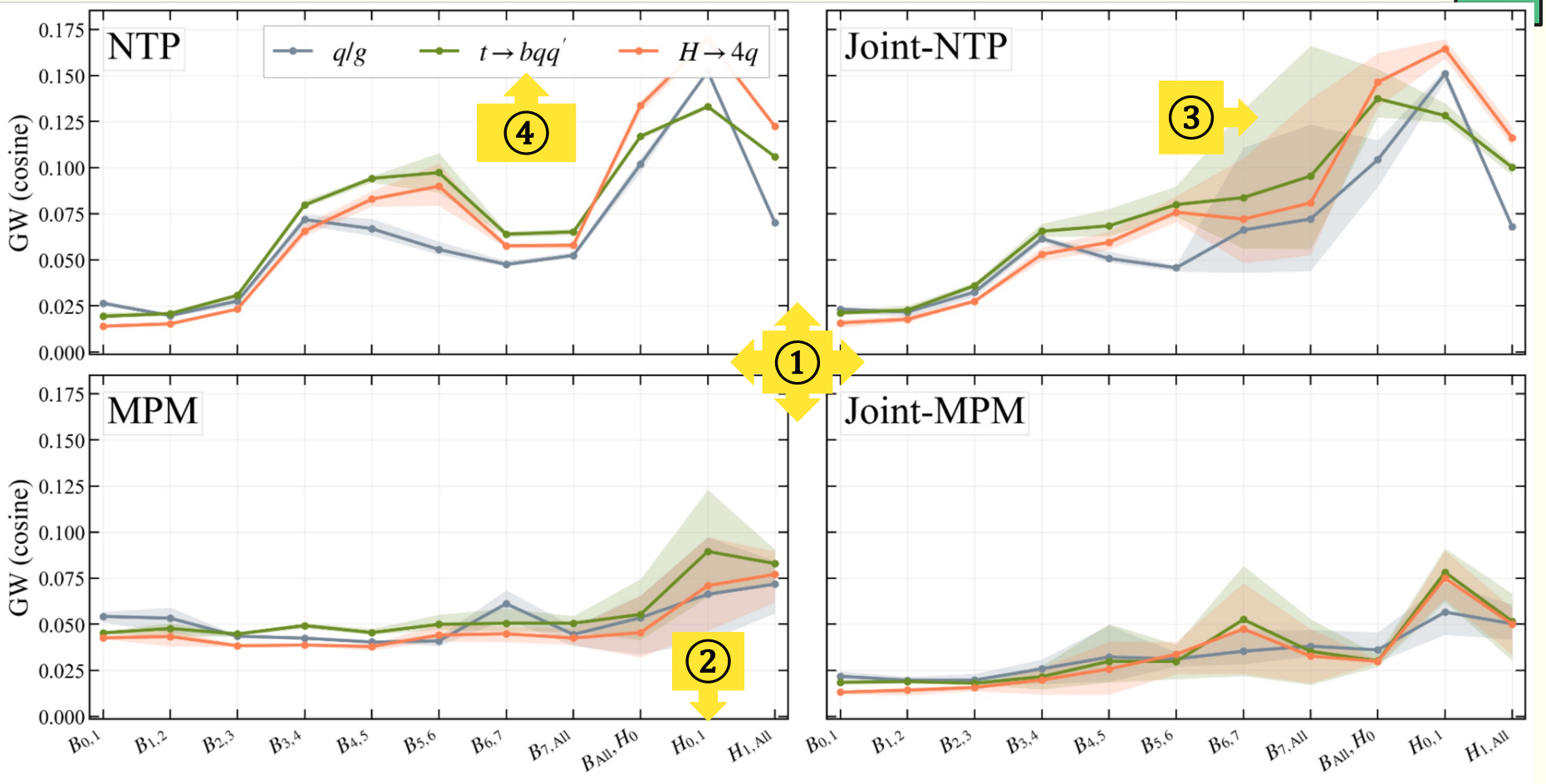
Block Evolution of Same Jet Type

$$\mathrm{GW}(\mathcal{A}^{b_i}, \mathcal{A}^{b_{i+1}})|_{s=\mathrm{EoT},c=\mathrm{fix}}$$

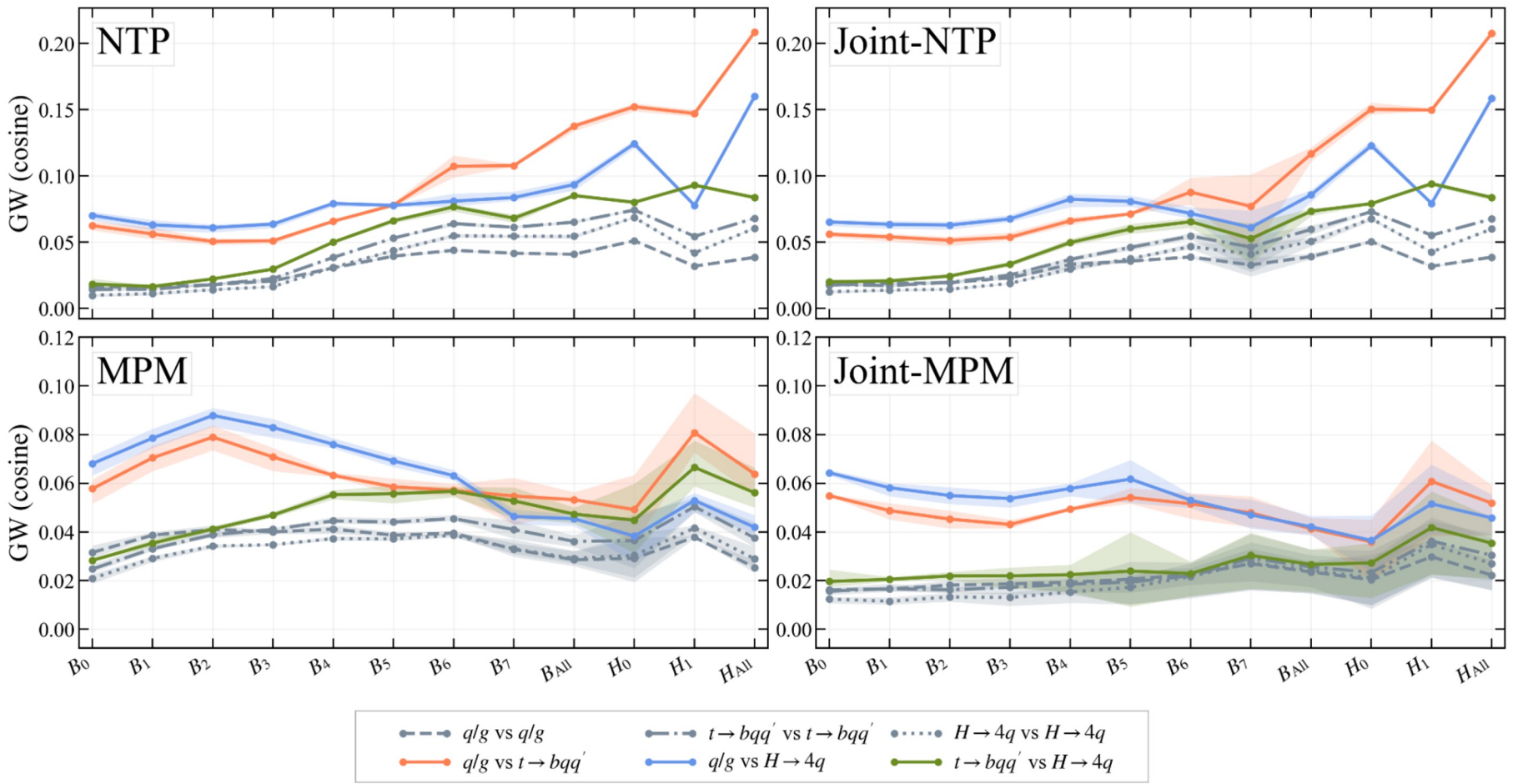


Block Evolution of Same Jet Type

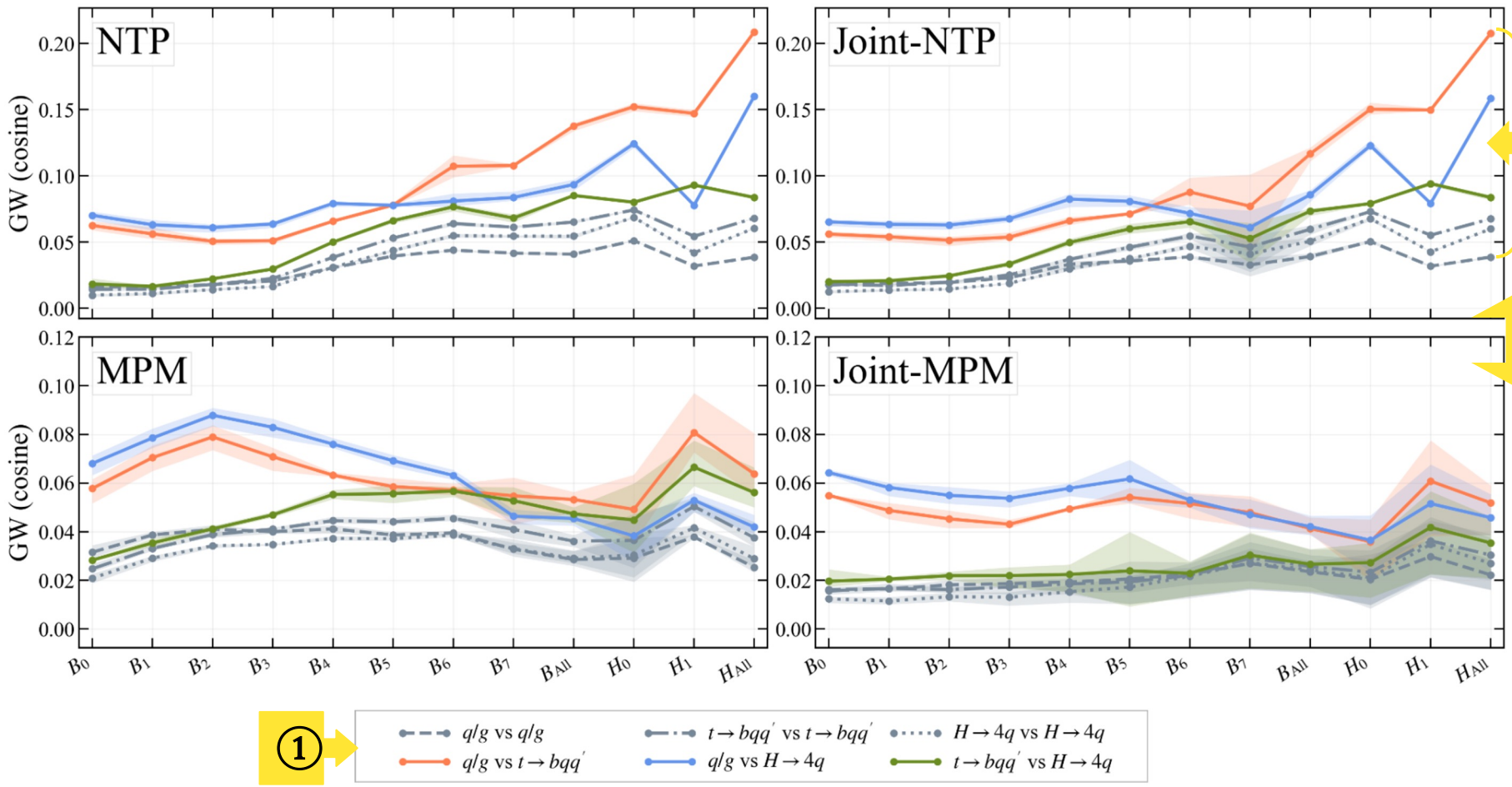
$$\mathrm{GW}(\mathcal{A}^{b_i}, \mathcal{A}^{b_{i+1}})|_{s=\mathrm{EoT},c=\mathrm{fix}}$$



… of Differences between Diff Jet Types $\mathrm{GW}(c_1, c_2)|_{\mathcal{A}^{bi}, s=\mathrm{EoT}}$

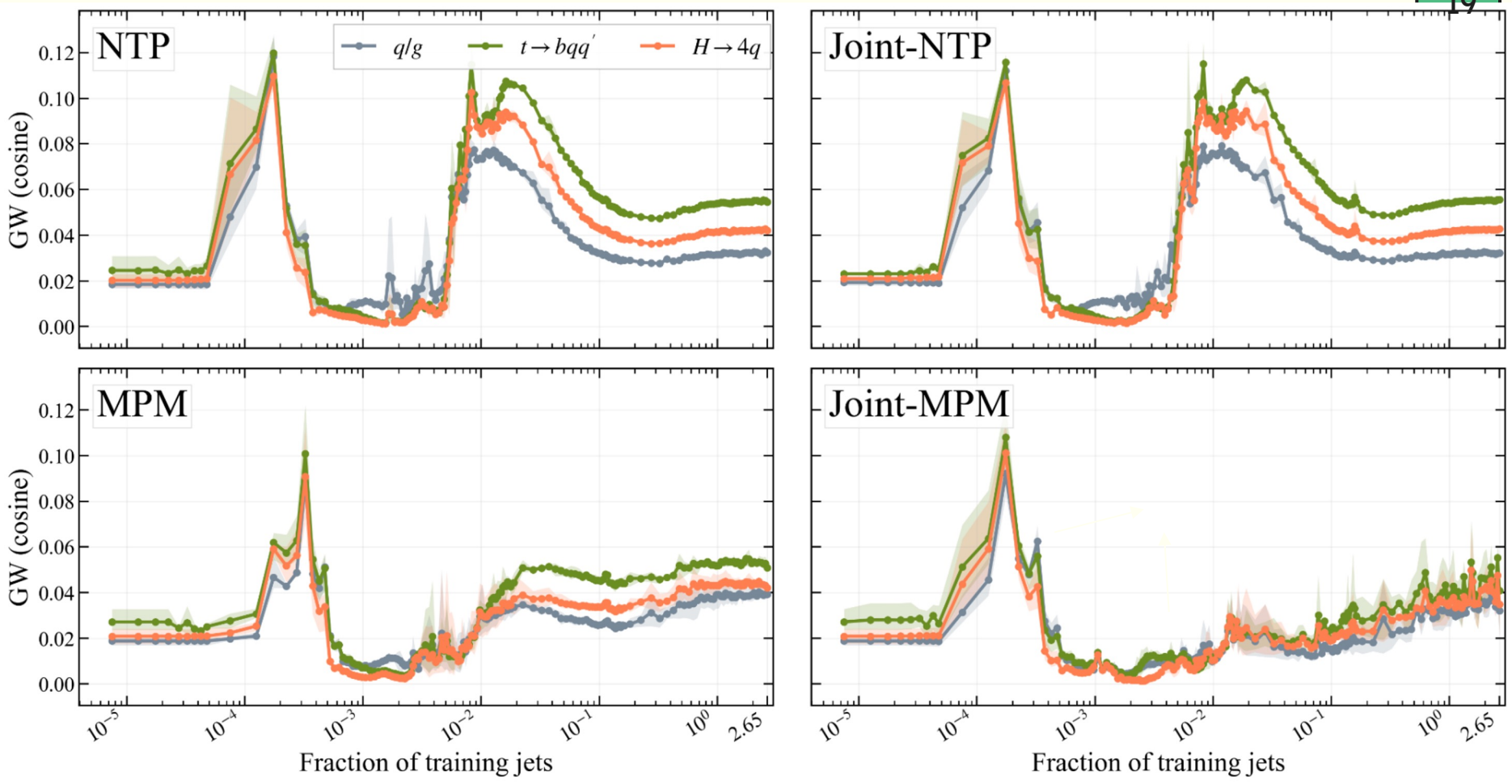


… of Differences between Diff Jet Types $\mathrm{GW}(c_1, c_2)|_{\mathcal{A}^{bi}, s=\mathrm{EoT}}$

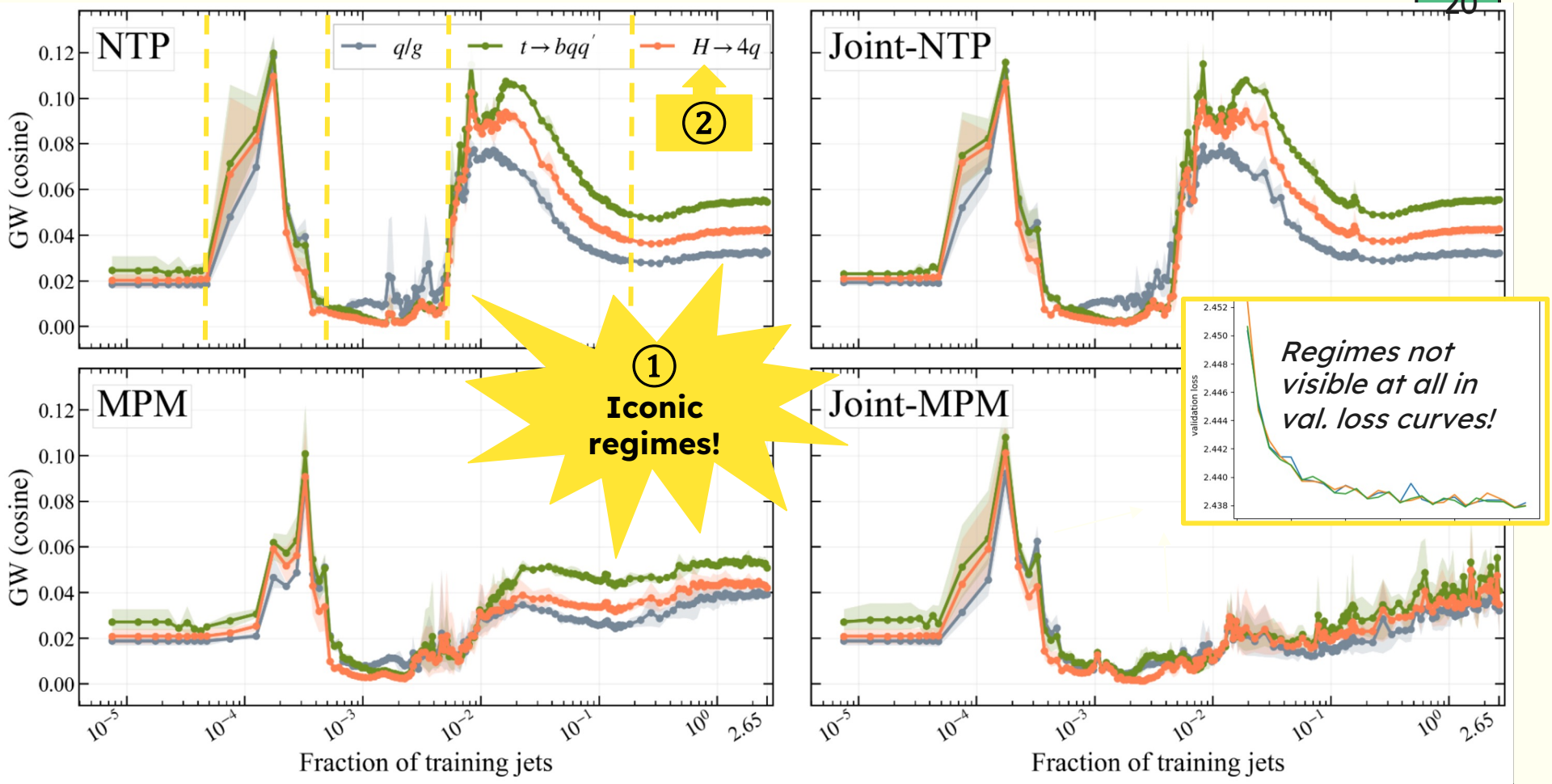


Q2 *How does the "final" rep. space (H_1) of the same model change w.r.t training?*

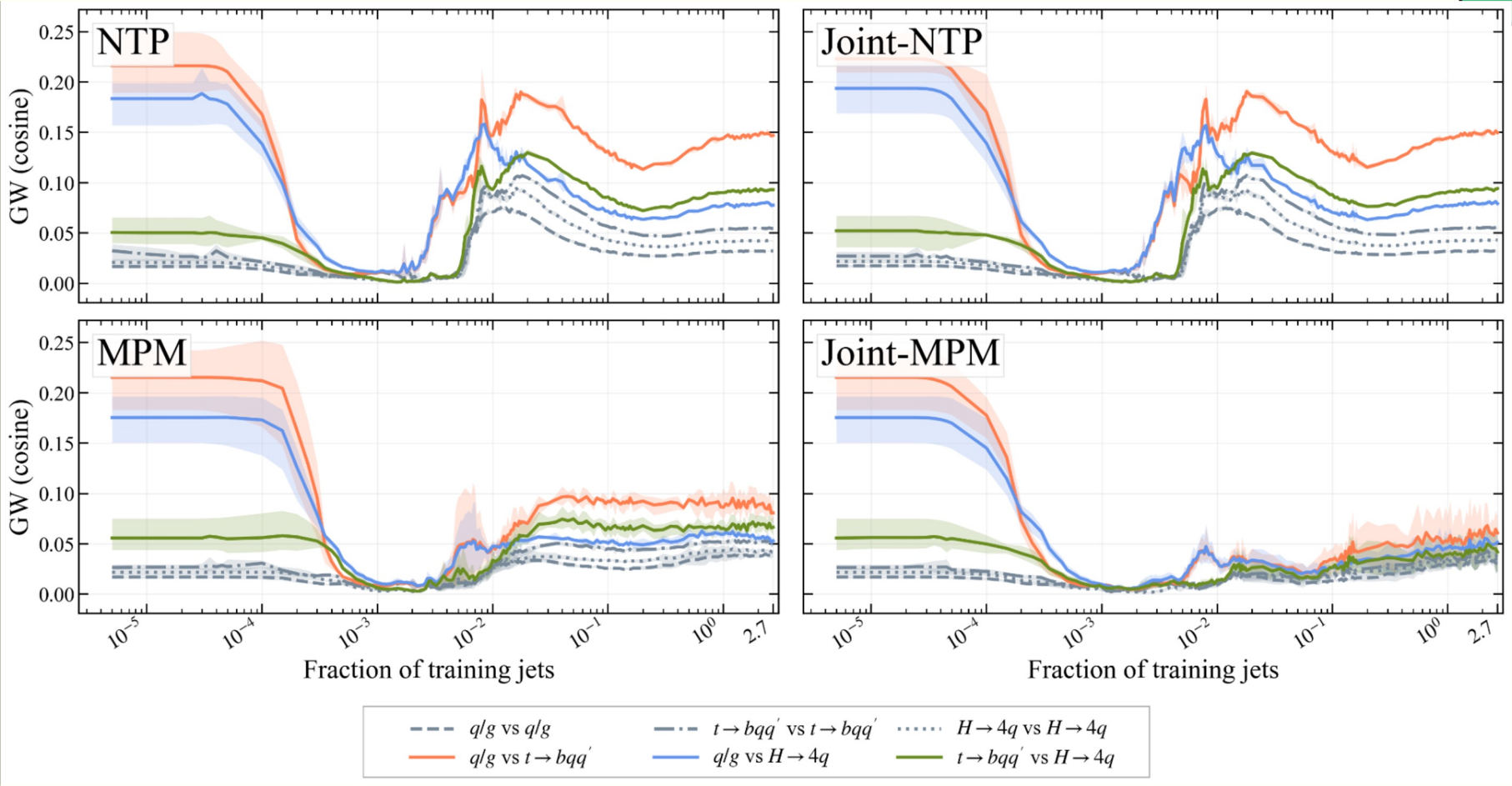
Training Evolution of Same Jet Type $\mathrm{GW}(\mathcal{A}_{s_i}, \mathcal{A}_{s_{i+1}})|_{b_i=\texttt{final},c=\texttt{fix}}$



Training Evolution of Same Jet Type $\mathrm{GW}(\mathcal{A}_{s_i}, \mathcal{A}_{s_{i+1}})|_{b_i=\texttt{final},c=\texttt{fix}}$

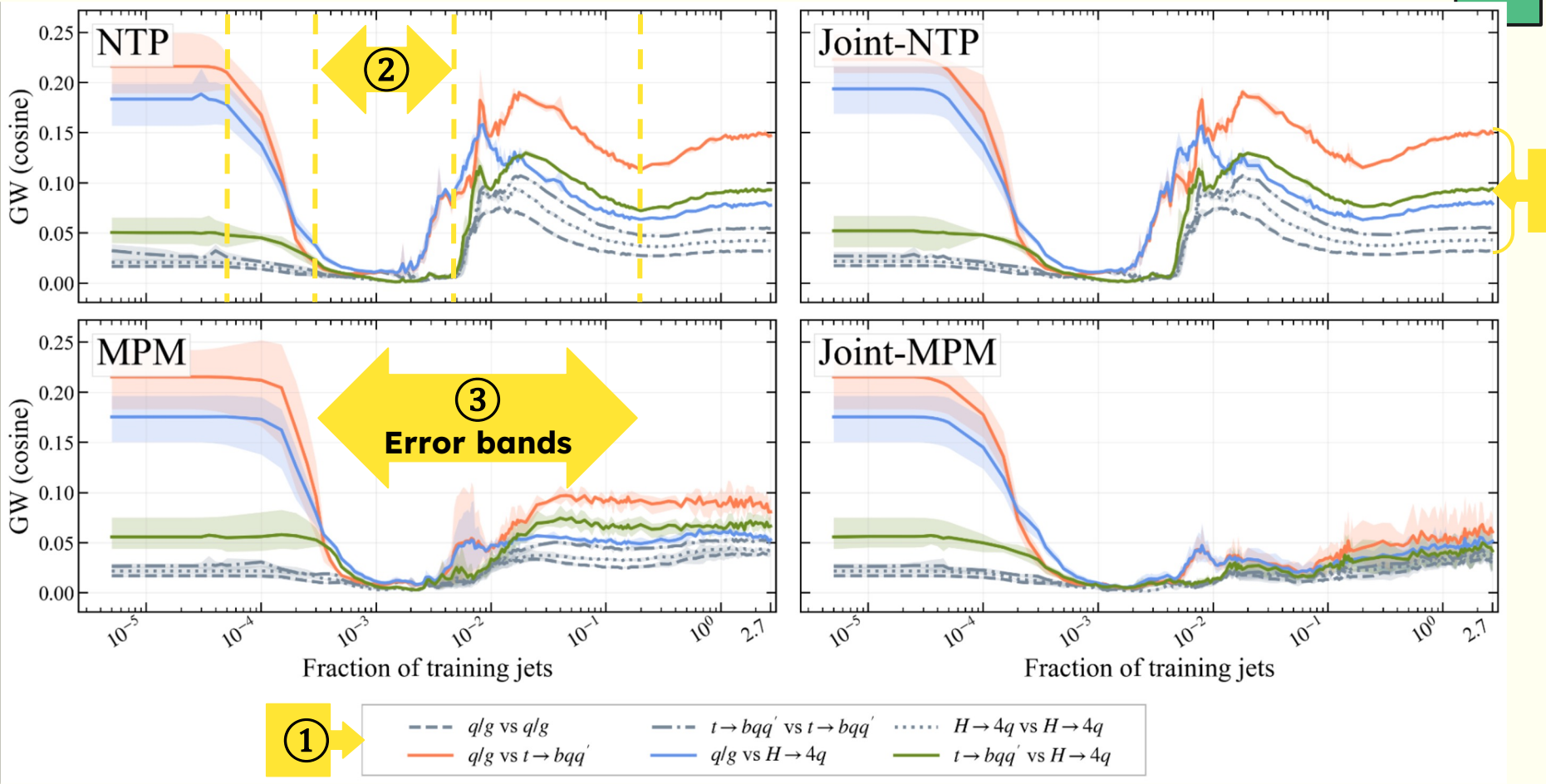


… of Differences between Diff Jet Types $\mathrm{GW}(c_1, c_2)|_{\mathcal{A}_{s_i}, b_i = b_{\mathrm{final}}}$



… of Differences between Diff Jet Types

$\mathrm{GW}(c_1, c_2)|_{\mathcal{A}_{s_i}, b_i = b_{\mathrm{final}}}$

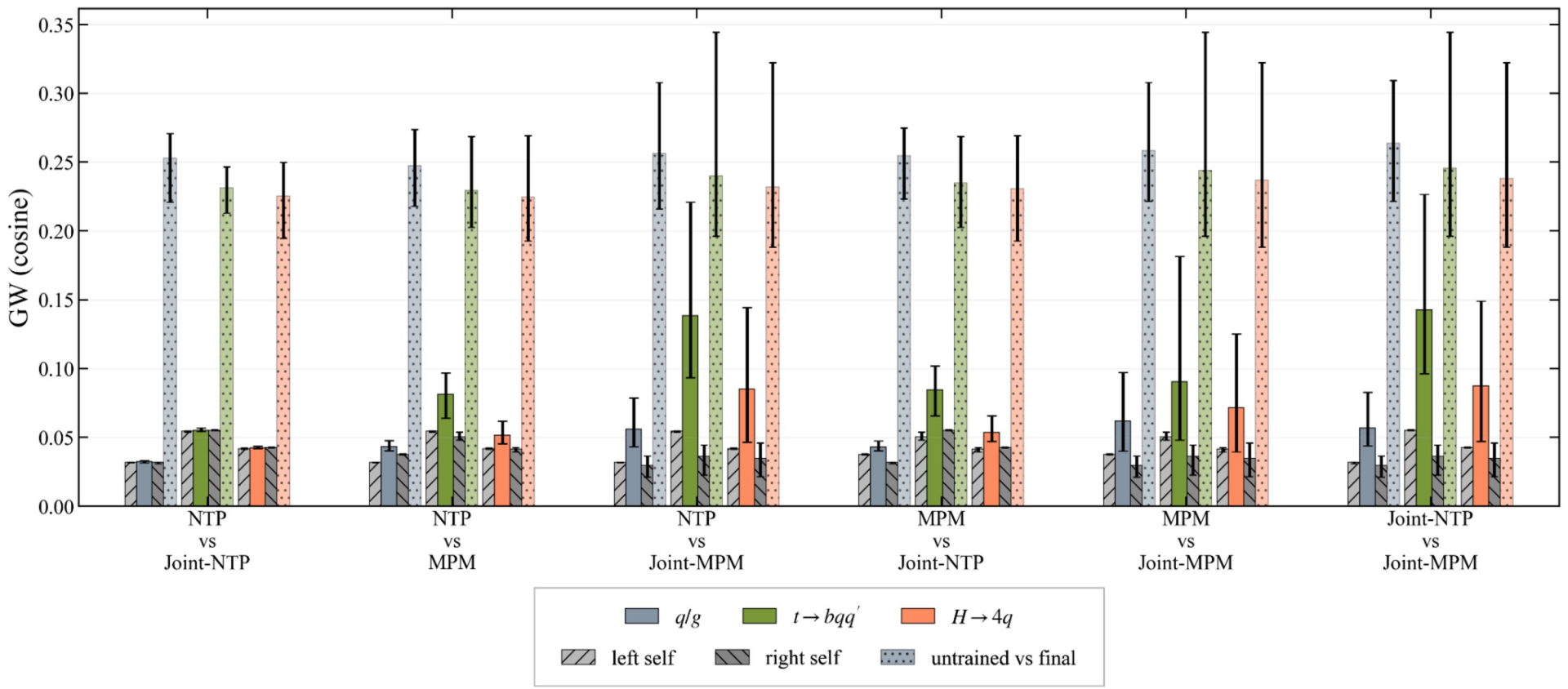


Q3 *How do different models organize the same jet type in their respective rep. spaces?*

$$\mathrm{GW}(\mathcal{A}, \mathcal{B})|_{b_i=\texttt{final}, s=\mathrm{EoT}, c=\mathrm{fix}}$$

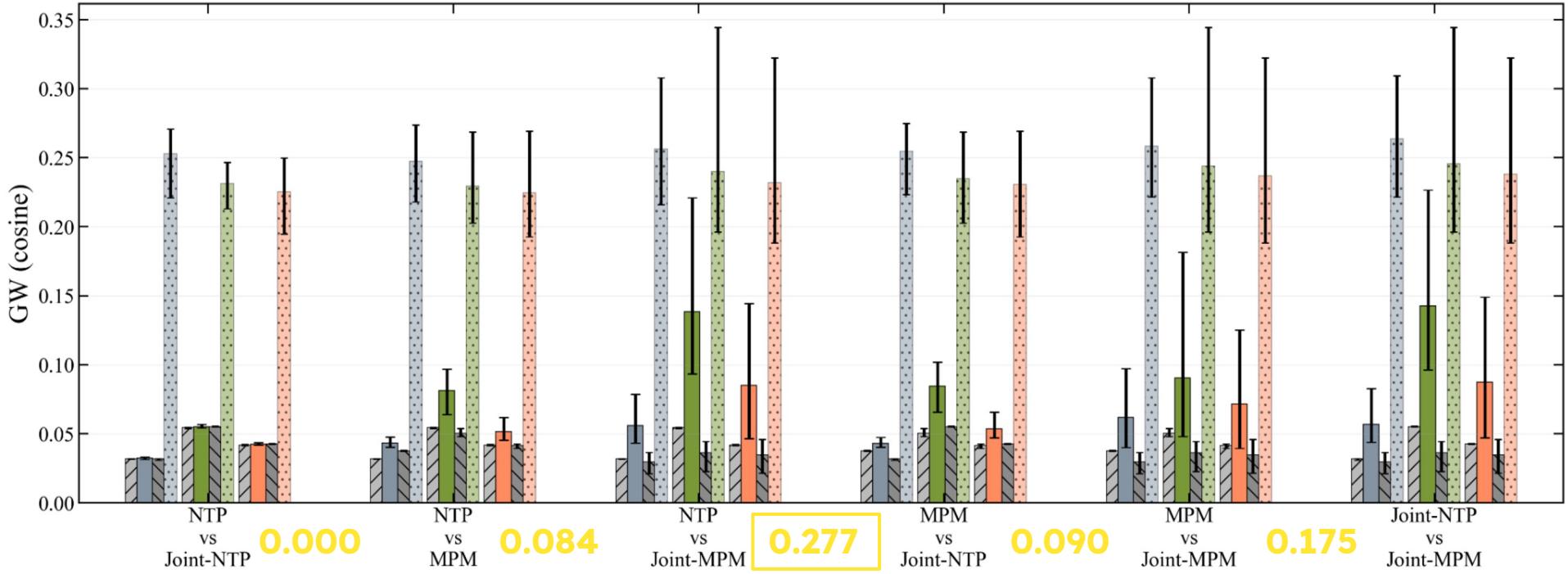
GW Difference between Two Models for Same jet Type

Error bars: min & max of 3×3=9 seed combinations (6×6=36 for "*untrained vs final*").



GW Difference between Two Models for Same jet Type

Error bars: min & max of 3×3=9 seed combinations (6×6=36 for "*untrained vs final*").



$$D_{\mathrm{rel}}(A,B) = \frac{d_{\mathrm{GW}}(A,B) - d_{\mathrm{same}}}{d_{\mathrm{null}} - d_{\mathrm{same}}}$$

Summary & Outlook

(Not Yet Conclusive) Conclusions

- The “final” representation space of all models share similar geometric structure across jet classes. → *The Platonic Representation Hypothesis seems to be working here!* 😈
- Geometric structure of each model is to first order the same for each jet type.
- Different jet types have larger geometric differences compared to the same jet type, with t *vs* qcd consistently having the highest GW.
- GW reveals a striking multi-regime structure during training, which is totally invisible in the usual validation loss curves.
- The penultimate blocks of the models seem to undergo the largest geometric rearrangement.

(Not Yet Conclusive) Conclusions

- The “final” representation space of all models share similar geometric structure across jet classes. → *The Platonic Representation Hypothesis seems to be working here!*🥳
- Geometric structure of each model is to first order the same for each jet type.
- Different jet types have larger geometric differences compared to the same jet type, with t *vs* qcd consistently having the highest GW.
- GW reveals a striking multi-regime structure during training, which is totally invisible in the usual validation loss curves.
- The penultimate blocks of the models seem to undergo the largest geometric rearrangement.

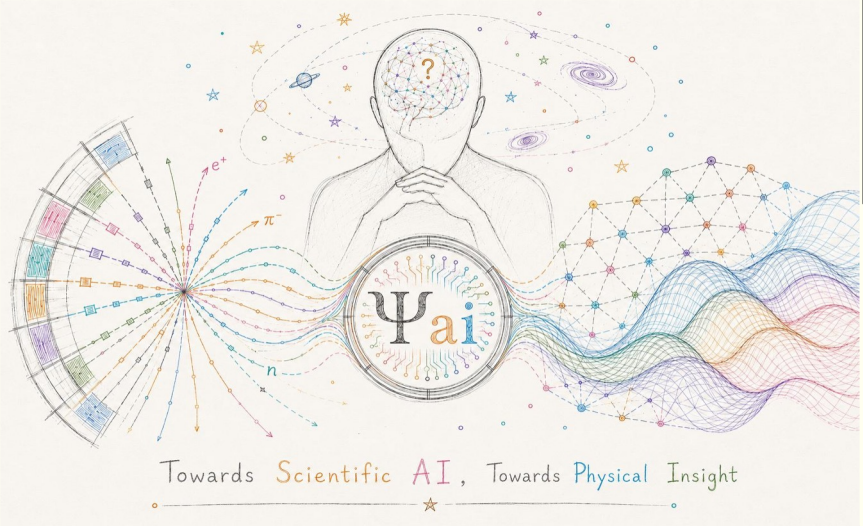
Questions beget questions.
And so many more questions!

We are only picking up a few curious shells on the shore; the ocean remains unexplored…

Contact Emails:
tianji_cai@tongji.edu.cn /
tianjiresearch@gmail.com.

Thanks for your attention!

Q&A?



Backup Slides

Abstract

Foundation models have demonstrated remarkable performance in collider physics, yet little is understood about the geometric principles underlying their neural representations. We propose representation geometry as a new direction for *Scientific AI*, where machine learning models serve not only as powerful predictors but also as “microscopes” for revealing the intrinsic structure of the underlying physical data. In this work, we study the OmniJet family of collider foundation models pre-trained with three different self-supervised objectives on the JetClass benchmark dataset. Using the Gromov–Wasserstein distance, we compare the metric geometry of representation spaces across network blocks, training steps, and pre-training schemes. Our preliminary results uncover highly structured geometric evolution both across network depth and throughout training, while suggesting that learned representations exhibit both common structural features and systematic variations induced by pre-training objectives. These studies establish a framework for investigating whether and to what extent collider foundation models converge toward common geometric structures, laying the groundwork for future empirical tests of the Platonic Representation Hypothesis in a scientific domain whose underlying reality is governed by fundamental physical laws.

Notation & Details

Object	Notation	Examples
Model	Curly capital letters	$\mathcal{A}, \mathcal{B}, \mathcal{C}$, ...
Model Head	H = L for linear, T for transformer in subscript	$\mathcal{A}_{\mathrm{L}}, \mathcal{A}_{\mathrm{T}}$, ...
Model Pre-training	NTP, MPM, NTP+MPM in subscript after head	$\mathcal{A}_{\text{L-NTP}}, \mathcal{A}_{\text{T-MPM}}$, ...
Initializations	Numbers added to the head in subscript	$\mathcal{A}_{\mathrm{L1}}, \mathcal{A}_{\mathrm{L2}}, \mathcal{A}_{\mathrm{T1}}$, ...
Model Block	b_i in the superscript	$\mathcal{A}^{b_1}, \mathcal{A}^{b_2}$...
Training Step	s as subscript (EoT for end of training)	$\mathcal{A}_{s=1000}, \mathcal{A}_{s=10^5}, \mathcal{A}_{s=\mathrm{EoT}}$...
Inference Data	Lower letter c (c_1, c_2, ...)	c = JetClass, QCD, T, ...

Model Architecture

MPM

#	Name	Type	Params
0	backbone	BackboneTransformer	1.6 M
1	backbone.input_projection	ProjectAdd	640
2	backbone.input_projection.embed_part	Linear	512
3	backbone.input_projection.embed_jet	Linear	128
4	backbone.transformer	Transformer	1.6 M
5	backbone.transformer.blocks	ModuleList	1.6 M
6	backbone.transformer.blocks.0	EncoderBlock	198 K
9	backbone.transformer.blocks.1	EncoderBlock	198 K
12	backbone.transformer.blocks.2	EncoderBlock	198 K
15	backbone.transformer.blocks.3	EncoderBlock	198 K
18	backbone.transformer.blocks.4	EncoderBlock	198 K
21	backbone.transformer.blocks.5	EncoderBlock	198 K
24	backbone.transformer.blocks.6	EncoderBlock	198 K
27	backbone.transformer.blocks.7	EncoderBlock	198 K
30	backbone.transformer.final_norm	LayerNorm	256
31	head_mpm	TokenPredictionHead	1.3 M
32	head_mpm.transformer_blocks	ModuleList	265 K
34	head_mpm.transformer_blocks.blocks	ModuleList	265 K
35	head_mpm.transformer_blocks.blocks.0	EncoderBlock	132 K
36	head_mpm.transformer_blocks.blocks.1	EncoderBlock	132 K
37	head_mpm.transformer_blocks.final_norm	LayerNorm	256
38	head_mpm.unembedding_mlp	Identity	0
39	head_mpm.unembedding_matrix	Linear	1.1 M

NTP

#	Name	Type	Params
0	backbone	BackboneTransformer	1.6 M
1	backbone.input_projection	ProjectAdd	640
2	backbone.input_projection.embed_part	Linear	512
3	backbone.input_projection.embed_jet	Linear	128
4	backbone.transformer	Transformer	1.6 M
5	backbone.transformer.blocks	ModuleList	1.6 M
6	backbone.transformer.blocks.0	EncoderBlock	198 K
9	backbone.transformer.blocks.1	EncoderBlock	198 K
12	backbone.transformer.blocks.2	EncoderBlock	198 K
15	backbone.transformer.blocks.3	EncoderBlock	198 K
18	backbone.transformer.blocks.4	EncoderBlock	198 K
21	backbone.transformer.blocks.5	EncoderBlock	198 K
24	backbone.transformer.blocks.6	EncoderBlock	198 K
27	backbone.transformer.blocks.7	EncoderBlock	198 K
30	backbone.transformer.final_norm	LayerNorm	256
31	head_gen	TokenPredictionHead	1.3 M
32	head_gen.transformer_blocks	ModuleList	265 K
34	head_gen.transformer_blocks.blocks	ModuleList	265 K
35	head_gen.transformer_blocks.blocks.0	EncoderBlock	132 K
36	head_gen.transformer_blocks.blocks.1	EncoderBlock	132 K
37	head_gen.transformer_blocks.final_norm	LayerNorm	256
38	head_gen.unembedding_mlp	Identity	0
39	head_gen.unembedding_matrix	Linear	1.1 M

Joint

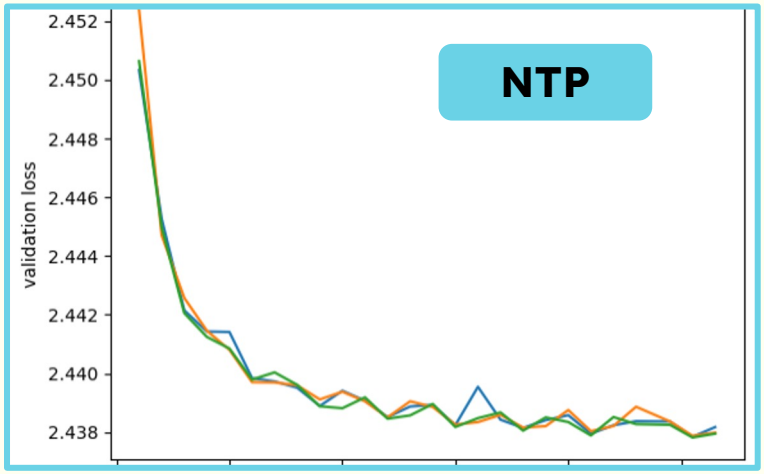
shared

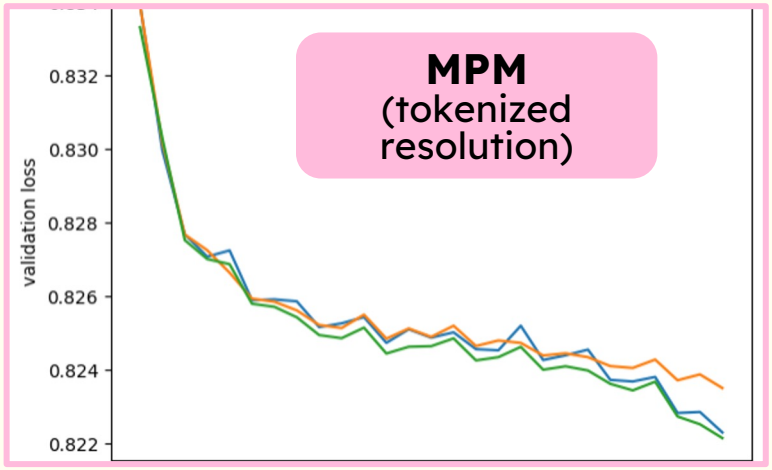
separate

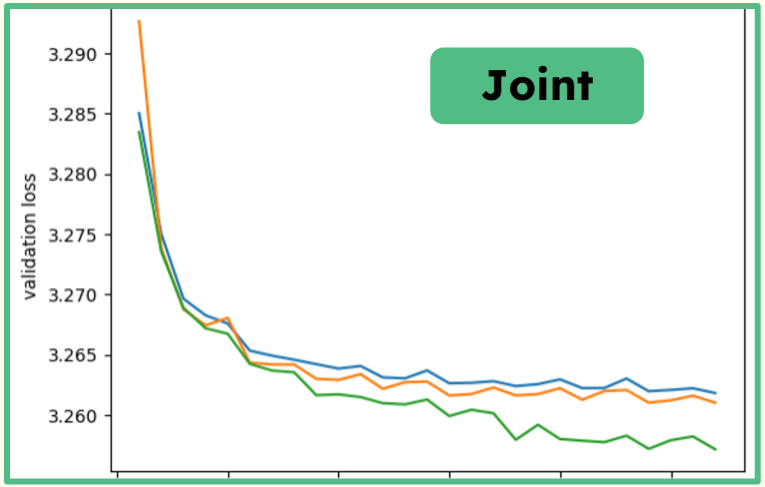
separate

#	Name	Type	Params
0	backbone	BackboneTransformer	1.6 M
1	backbone.input_projection	ProjectAdd	640
2	backbone.input_projection.embed_part	Linear	512
3	backbone.input_projection.embed_jet	Linear	128
4	backbone.transformer	Transformer	1.6 M
5	backbone.transformer.blocks	ModuleList	1.6 M
6	backbone.transformer.blocks.0	EncoderBlock	198 K
9	backbone.transformer.blocks.1	EncoderBlock	198 K
12	backbone.transformer.blocks.2	EncoderBlock	198 K
15	backbone.transformer.blocks.3	EncoderBlock	198 K
18	backbone.transformer.blocks.4	EncoderBlock	198 K
21	backbone.transformer.blocks.5	EncoderBlock	198 K
24	backbone.transformer.blocks.6	EncoderBlock	198 K
27	backbone.transformer.blocks.7	EncoderBlock	198 K
30	backbone.transformer.final_norm	LayerNorm	256
31	head_mpm	TokenPredictionHead	1.3 M
32	head_mpm.transformer_blocks	ModuleList	265 K
34	head_mpm.transformer_blocks.blocks	ModuleList	265 K
35	head_mpm.transformer_blocks.blocks.0	EncoderBlock	132 K
36	head_mpm.transformer_blocks.blocks.1	EncoderBlock	132 K
37	head_mpm.transformer_blocks.final_norm	LayerNorm	256
38	head_mpm.unembedding_mlp	Identity	0
39	head_mpm.unembedding_matrix	Linear	1.1 M
40	head_gen	TokenPredictionHead	1.3 M
41	head_gen.transformer_blocks	ModuleList	265 K
43	head_gen.transformer_blocks.blocks	ModuleList	265 K
44	head_gen.transformer_blocks.blocks.0	EncoderBlock	132 K
45	head_gen.transformer_blocks.blocks.1	EncoderBlock	132 K
46	head_gen.transformer_blocks.final_norm	LayerNorm	256
47	head_gen.unembedding_mlp	Identity	0
48	head_gen.unembedding_matrix	Linear	1.1 M

Validation Loss Curves During Training







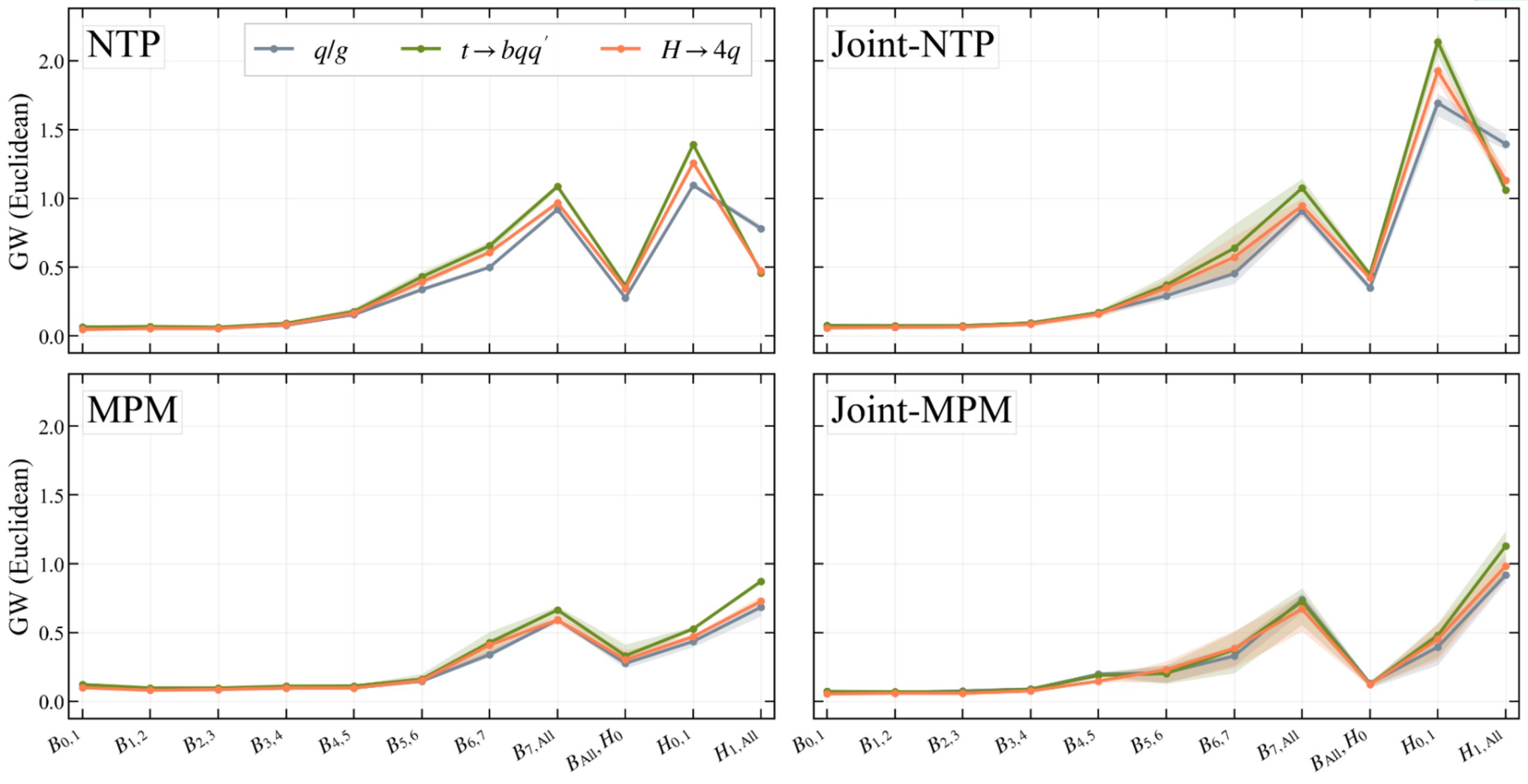
Adaptive Checkpointing Schedule:

- Step 0-10 → Every 1 step: 10 ckpts, 500 jets/ckpt.
- Step 10-200 → Every 10 steps: 20 ckpts, 5k jets/ckpt.
- Step 200-600 → Every 20 steps: 20 ckpts, 10k jets/ckpt.
- Step 600-1600 → Every 50 steps: 20 ckpts; 20k jets/ckpt.
- Step 1600-3600 → Every 100 steps: 20 ckpts; 50k jets/ckpt.
- Step 3600-33600 → Every 1000 steps: 30 ckpts; 500k jets/ckpt.
- Step 33600-333.6k → Every 10k steps: 30 ckpts; 5M jets/ckpt.
- Step 333.6k-540k → Every 20k steps: 10M jets/ckpt

540k steps means 266,705,000 jets. JetClass has 100M training jets, meaning that each jet is seen ~2.7 times.

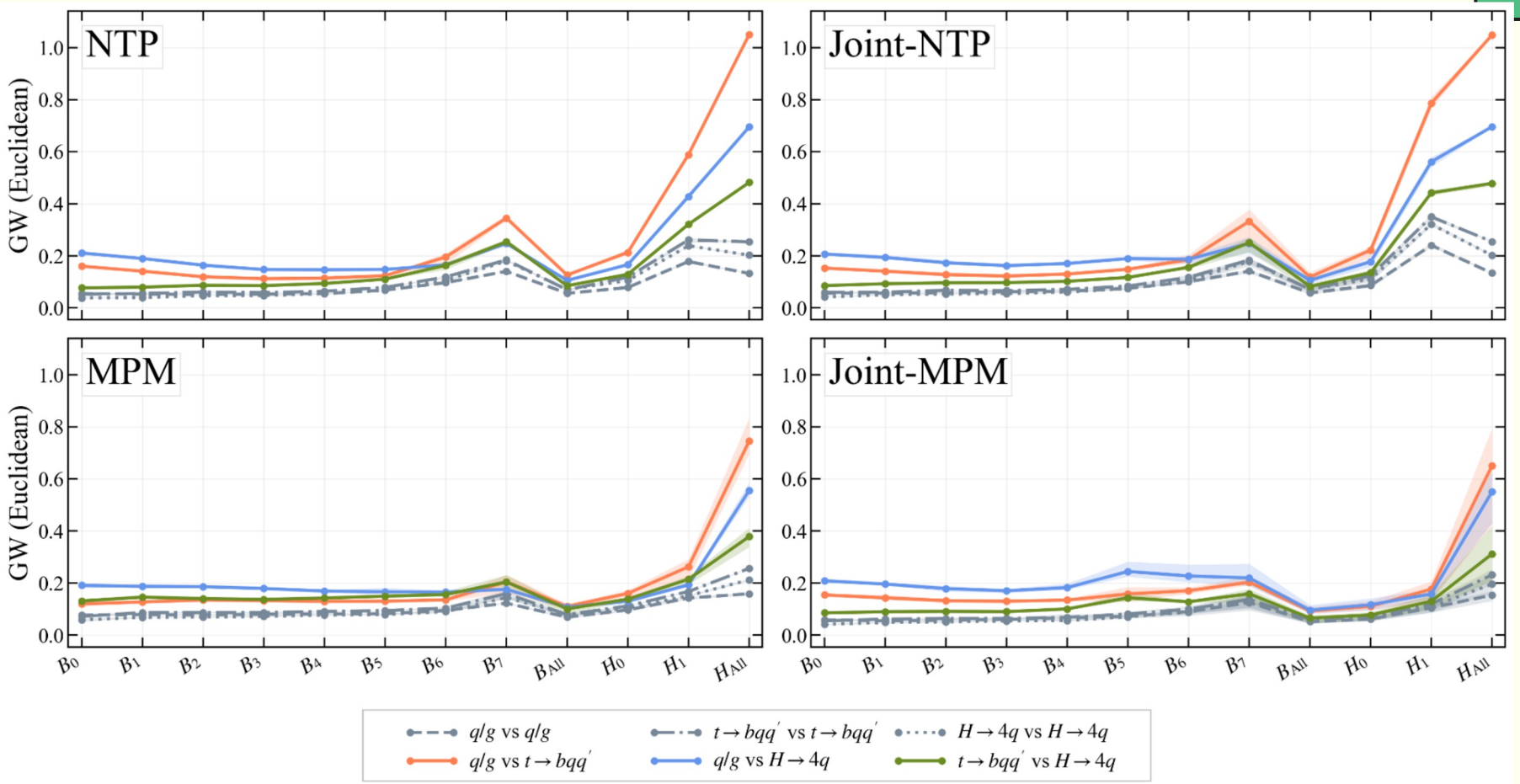
Block Evolution of Same Jet Type

$$\mathrm{GW}(\mathcal{A}^{b_i}, \mathcal{A}^{b_{i+1}})|_{s=\mathrm{EoT}, c=\mathrm{fix}}$$

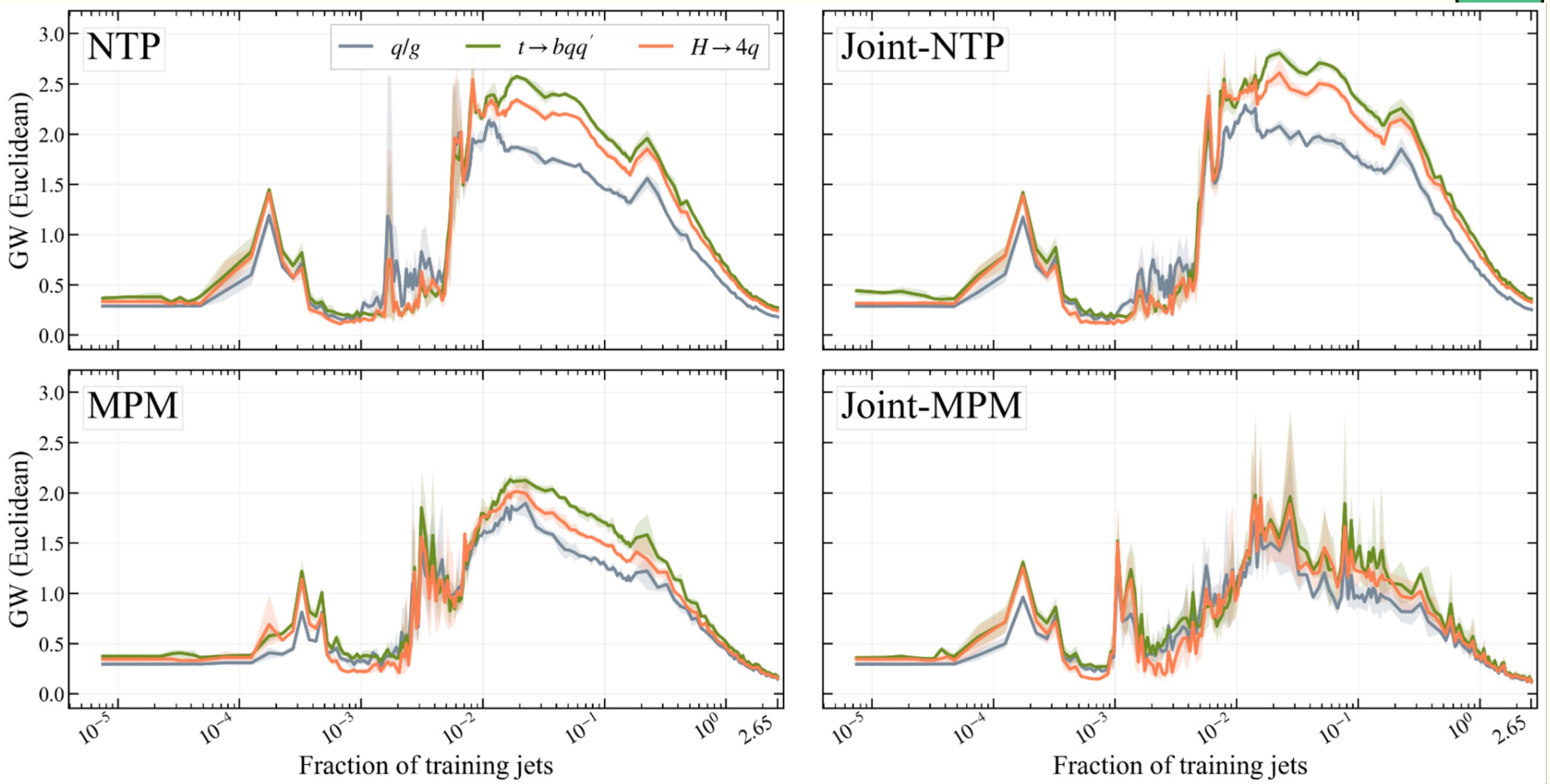


… of Differences between Diff Jet Types

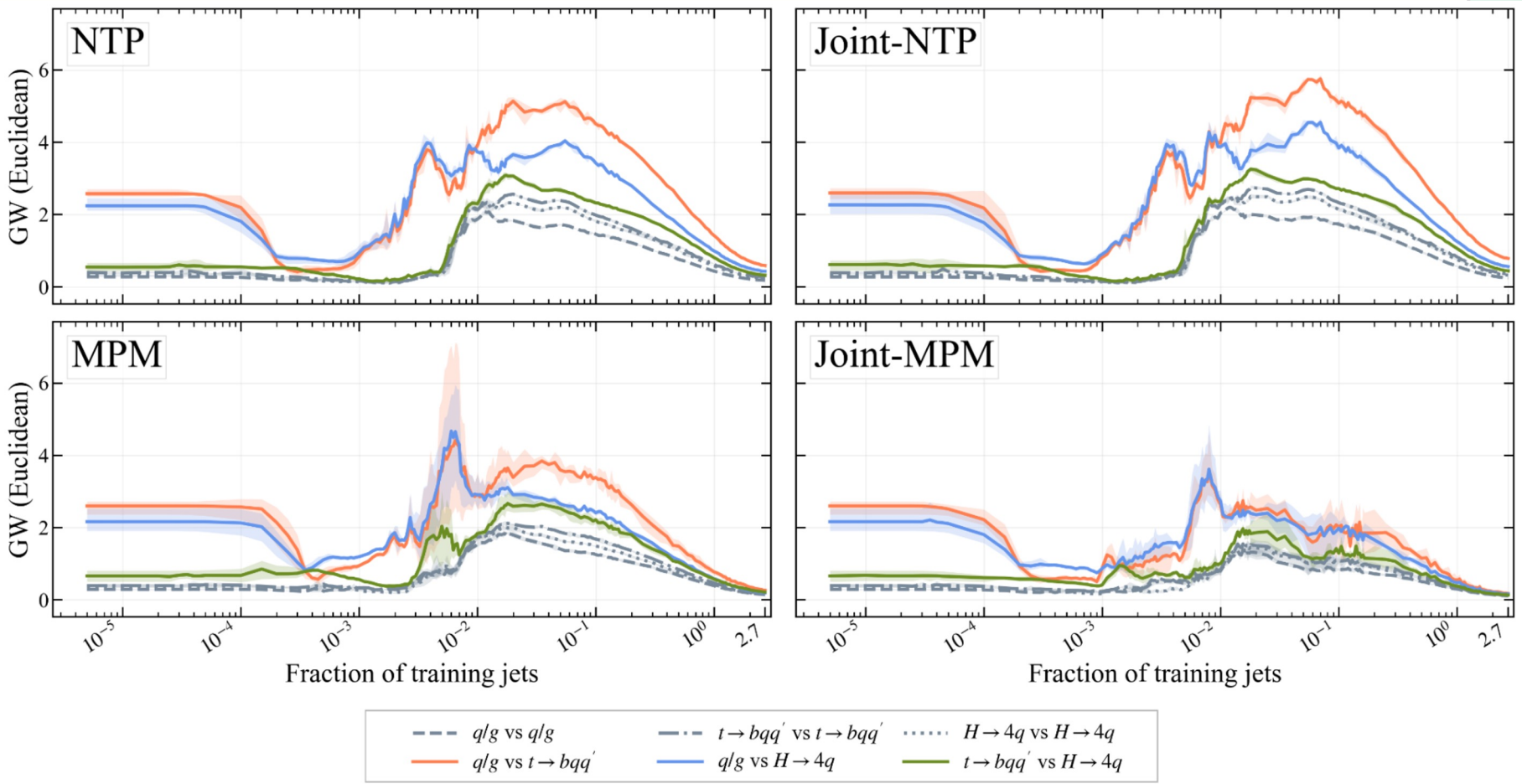
$\mathrm{GW}(c_1, c_2)|_{\mathcal{A}^{bi}, s=\mathrm{EoT}}$



Training Evolution of Same Jet Type $\mathrm{GW}(\mathcal{A}_{s_i}, \mathcal{A}_{s_{i+1}})|_{b_i=\texttt{final},c=\mathrm{fix}}$



… of Differences between Diff Jet Types $\mathrm{GW}(c_1, c_2)|_{\mathcal{A}^{bi}, s=\mathrm{EoT}}$



GW Difference between Two Models for Same jet Type