Modeling and Computation in the Space of Language: Symbolic and LLM-Based Approaches

Haizhao Yang
Department of Mathematics
Department of Computer Science
University of Maryland College Park

Efficient and Reliable Deep Learning Methods and their Scientific Applications

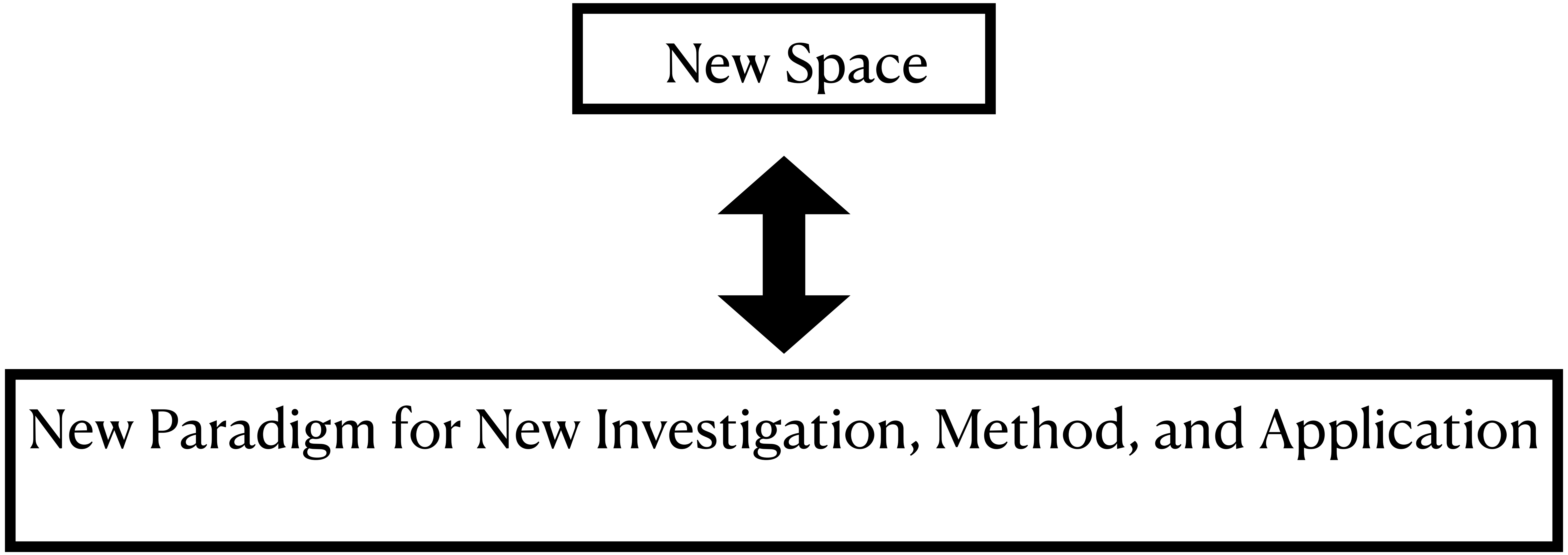
Banff International Research Station

June 24, 2025

Modeling & Computing in the Right Space

- Sparse Grids
  - "Right Space" - the solution can be approximated well by sparse combinations of basis functions
- Low-Rank Tensor Methods
  - "Right Space" - the solution can be approximated by a sum of products of lower-dimensional functions
- Neural Network Methods
  - "Right Space" - the implicit function space learned by the network during training

Modeling & Computing in the Right Space



Modeling & Computing in the Right Space

The space of language: Example 1. math expression and description

A function $f : \mathbb{R}^d \to \mathbb{R}$ is **permutation symmetric** if it is invariant under any permutation of its input components. That is:

$$f(x_1, x_2, \ldots, x_d) = f(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(d)})$$

for any permutation $\pi \in S_d$ (the symmetric group on d elements).

Example: $f(x) = \sum_{i=1}^{d} x_i^2$ is permutation symmetric.

Modeling & Computing in the Right Space

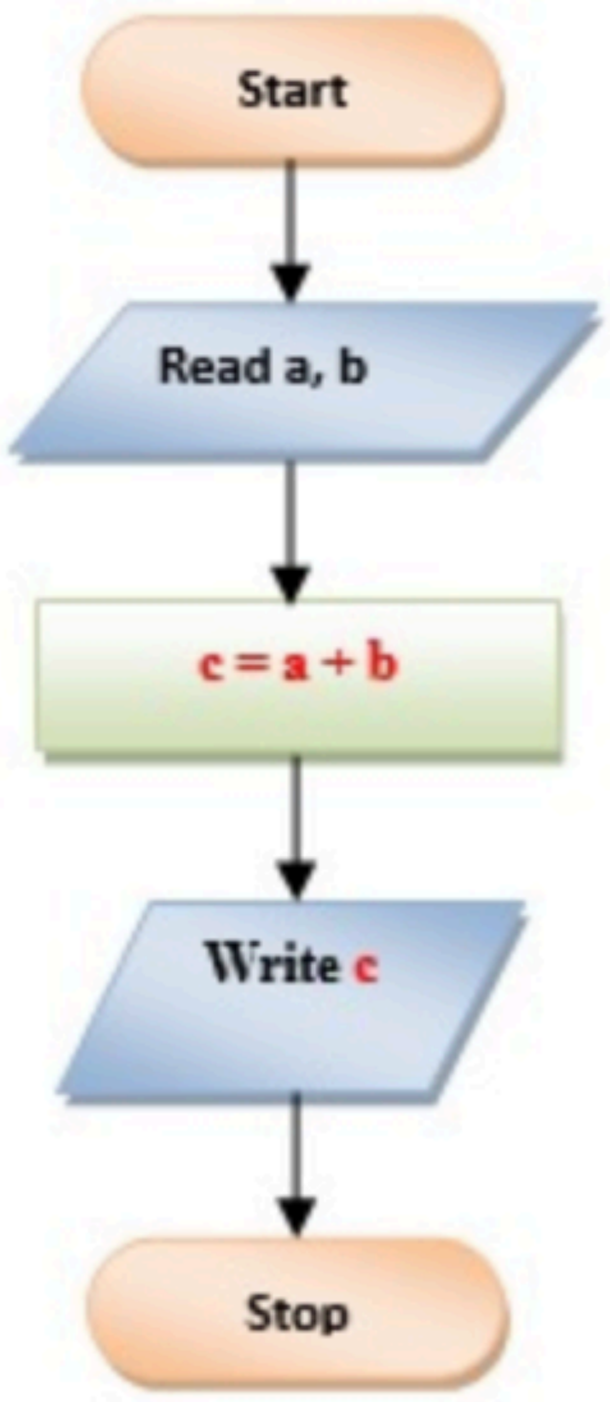
The space of language: Example 2. Algorithm, Flowchart, and code language

To find sum of two numbers

Algorithm

1. Start
2. Read a, b
3. c = a + b
4. Print or display c
5. Stop

Flowchart



Program

```
#include<stdio.h>

int main()
{
    int a, b, c;

    printf("Enter value of a: ");
    scanf("%d", &a);

    printf("Enter value of b: ");
    scanf("%d", &b);
    c = a+b;

    printf("Sum of given two numbers is: %d", c);

return 0;
}
```

Modeling & Computing in the Space of Natural Language

Question:

- How to model and compute in the space of natural language?

Two Complementary Approaches:

- Symbolic learning (Finite Expression Method)
- Large language model (LLM)

Applications:

- Search for a solution
- Search for a mathematical model
- Search for a computational algorithm
- Search for executable code
- ...

- **Symbolic learning (Finite Expression Method)**
- Large language model (LLM) for modeling and computing assistant

Finite Expression Method (FEX) Methodology

Liang and Y. arXiv:2206.10121

Motivating Problem:

- A **structured** high-dimensional Poisson equation

$$-\Delta u = f \quad \text{for } x \in \Omega, \quad u = g \text{ for } x \in \partial\Omega$$

  with a solution $u(x) = \frac{1}{2}\sum_{i=1}^{d} x_i^2$ of low complexity $O(d)$, i.e., $O(d)$ operators in this expression

Idea:

- Find an explicit expression that approximates the solution of a PDE
- Function space with finite expressions
  - **Mathematical expressions**: a combination of symbols with rules to form a valid function, e.g., $\sin(2x) + 5$
  - k**-finite expression**: a mathematical expression with at most k operators
  - Function space in FEX: $\mathbb{S}_k$ as the set of s-finite expressions with $s \leq k$

Finite Expression Method (FEX) Theory

Liang and Y. arXiv:2206.10121

Advantages in Real Analysis: "No" curse of dimensionality in approximation

Theorem (Liang and Y. 2022) Suppose the function space is $\mathbb{S}_k$ generated with operators including ``+", ``-", ``$\times$", ``/", ``$\max\{0,x\}$", ``sin(x)", and ``2^x". Let $p \in [1, +\infty)$. For any f in the Holder function class $\mathscr{H}_\mu^\alpha([0,1]^d)$ and $\varepsilon > 0$, there exists a k-finite expression ϕ in $\mathbb{S}_k$ such that

$$\|f - \phi\|_{L^p} \leq \varepsilon,$$

if

$$k \geq \mathcal{O}(d^2(\log d + \log \frac{1}{\varepsilon})^2).$$

Finite Expression Method (FEX) Practice

Advantages in Practice:

- Leverage the power of descriptive structures of problems

Question:

- How to do computation with description?

Answers:

- Reinforcement learning
- In-context learning
- Bayesian approach

Finite Expression Method for Solving PDEs

Liang and Y. arXiv:2206.10121

Least square based FEX

- e.g., $\mathscr{D}(u) = f$ in Ω and $\mathscr{B}(u) = g$ on $\partial\Omega$
- A mathematical expression u^* to approximate the PDE solution via

$$u^* = \arg\min_{u \in \mathbb{S}_k} \mathscr{L}(u) := \arg\min_{u \in \mathbb{S}_k} \|\mathscr{D}u - f\|_2^2 + \lambda \|\mathscr{B}u - g\|_2^2$$

- Or numerically

$$u^* = \arg\min_{u \in \mathbb{S}_k} \mathscr{L}(u) := \arg\min_{u \in \mathbb{S}_k} \frac{1}{n} \sum_{i=1}^{n} |\mathscr{D}u(x_i) - f(x_i)|^2 + \lambda \frac{1}{m} \sum_{j=1}^{m} |\mathscr{B}u(x_j) - g(x_j)|^2$$

- Question: how to solve this combinatorial optimization problem? Reinforcement learning

Numerical Comparison

Liang and Y. arXiv:2206.10121

- NN method:
  - Neural networks with a ReLU^2-activation function
  - ResNet with depth 7 and width 50

- FEX method:
  - Depth 3 binary tree
  - Binary set $\mathbb{B} = \{+, -, \times\}$
  - Unary set $\mathbb{U} = \{0, 1, \text{Id}, (\cdot)^2, (\cdot)^3, (\cdot)^4, \exp, \sin, \cos\}$

- The right space: solutions with simple descriptive structures

Solving High-Dimensional PDEs with FEX

Poisson equation

$\Omega = [0,1]^d$

$-\Delta u = f$ for $x \in \Omega$ and $u = g$ on $\partial\Omega$

True solution: $u(\boldsymbol{x}) = \frac{1}{2}\sum_{i=1}^{d} x_i^2$

Linear conservation law

$T\times\Omega = [0,1]\times[-1,1]^d$

$\frac{\pi d}{4} u_t - \sum_{i=1}^{d} u_{x_i} = 0$ and $u(0,x) = \sin\left(\frac{\pi}{4}\sum_{i=1}^{d} x_i\right)$

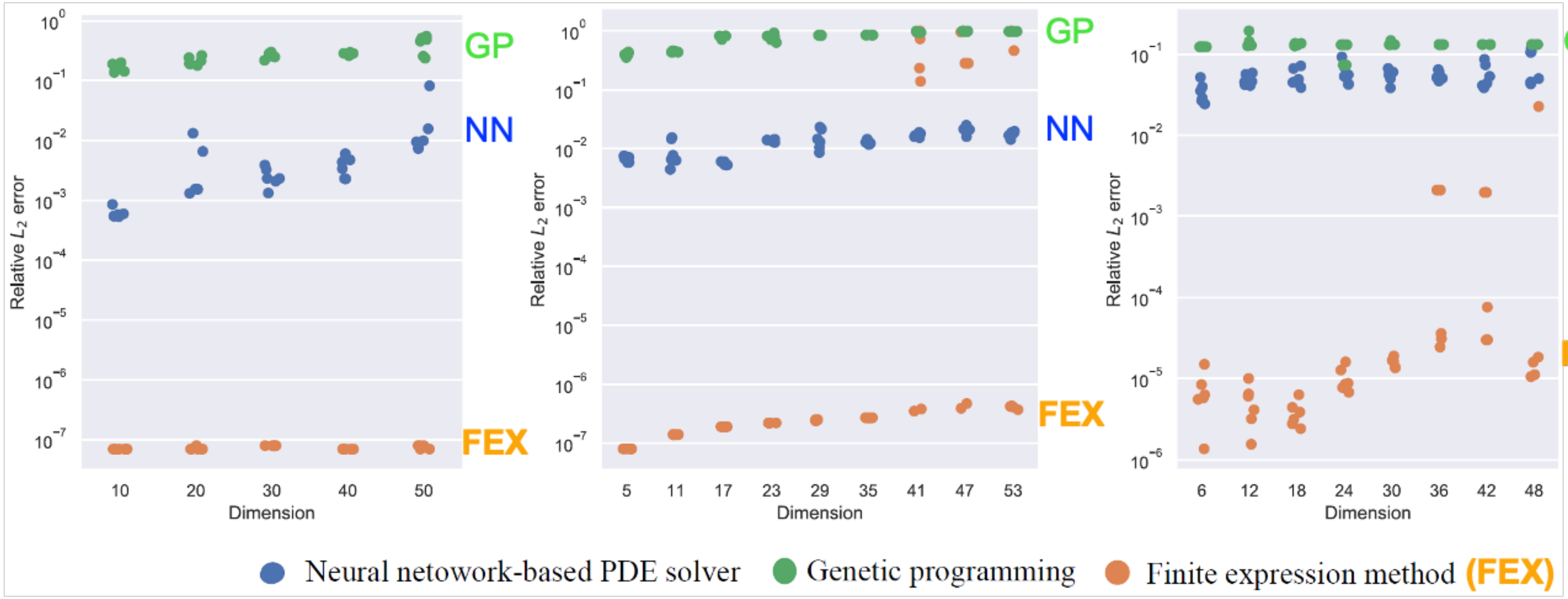
True solution: $u(\boldsymbol{x}) = \sin(t + \frac{\pi}{4}\sum_{i=1}^{d} x_i)$

Nonlinear Schrödinger equation

$\Omega = [-1,1]^d$

$-\Delta u + u^3 + Vu = f$ for $x \in \Omega$

True solution: $u(\boldsymbol{x}) = \exp(\sum_{i=1}^{d} \cos(x_i)/d)$



Liang and Y. arXiv:2206.10121

FEX for Partial Integral Differential Equations

Hardwick, Liang, Y., arxiv:2410.00835

$$\frac{\partial u}{\partial t} + b \cdot \nabla u + \frac{1}{2}\mathrm{Tr}(\sigma\sigma^T H(u)) + \mathrm{A}u + f = 0$$

$$u(T, \cdot\,) = g(\,\cdot\,)$$

$$\mathrm{A}u(t,x) = \int_{\mathbb{R}^n} (u(t, x + G(x,z)) - u(t,x) - G(x,z)\cdot \nabla u(t,x))\nu(dz)$$

$G(x,z) \in \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$, and ν is a Levy measure associated with a Poisson random measure.

Dimension	2	4	6	8	10	20	30
FEX-PG	2.99e-7	3.17e-7	5.16e-7	7.26e-7	2.05e-7	8.02e-7	4.49e-7
TD-NN [23]	0.00954	0.00251	0.00025	0.00671	0.01895	0.00702	0.01221

Dimension	40	50	60	70	80	90	100
FEX-PG	9.05e-7	4.27e-7	4.55e-7	3.54e-7	5.89e-7	6.44e-7	5.64e-7
TD-NN [23]	0.00956	0.00219	0.00944	0.00044	0.00277	0.00460	0.00548

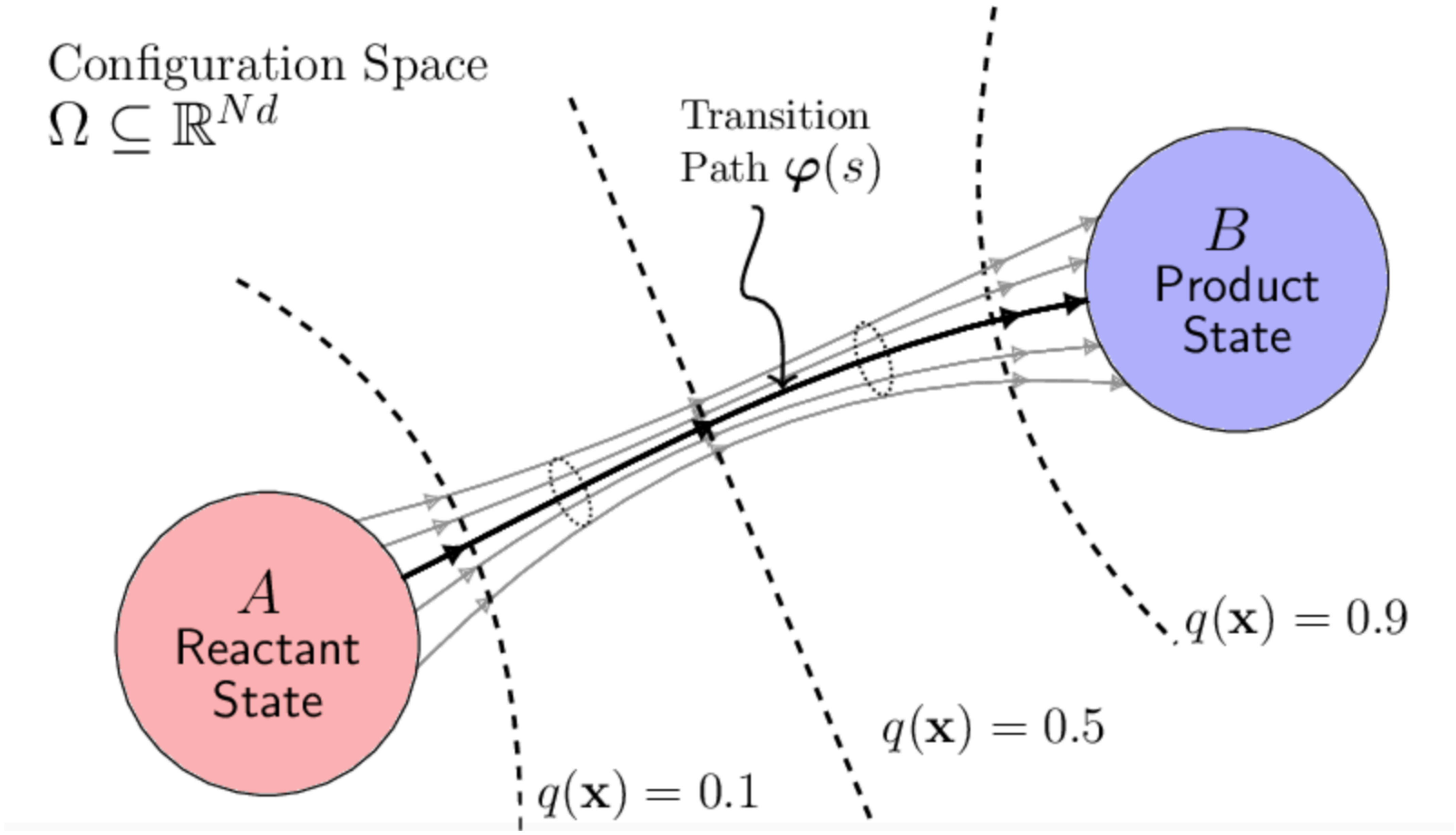
Committor Function for Rare Events

Song, Cameron, Yang arXiv:2306.12268, SISC, 2025

$$\begin{cases} \beta^{-1}\Delta q + \nabla V \cdot \nabla q = 0 & \text{for } \boldsymbol{x} \notin A \cup B \\ q(\mathbf{x}) = 0 & \text{for } \boldsymbol{x} \in A \\ q(\mathbf{x}) = 1 & \text{for } \boldsymbol{x} \in B\,. \end{cases}$$

where β is temperature and $q(\boldsymbol{x})$ is the probability

for starting from $\boldsymbol{x}$, visiting B before visiting A.



Committor Function for Rare Events

Song, Cameron, Yang arXiv:2306.12268, SISC, 2025

Difficulty

- dimension $\propto$ number of atoms

Physical Structure

- Low-dimensional structure: a small number of collective variables

Machine Learning

- Identify a low-dimensional structure
- High-dimension $\rightarrow$ low-dimension

Committor Function for Rare Events

Song, Cameron, Yang arXiv:2306.12268, SISC, 2025

Example: Double-Well potential

$$V(\mathbf{x}) = \underbrace{\left(x_1{}^2 - 1\right)^2}_{\text{collective variable}} + 0.3 \sum_{i=2}^{d} x_i^2$$

with $$A = \left\{x \in \mathbb{R}^d \mid x_1 \le -1\right\}, \quad B = \left\{x \in \mathbb{R}^d \mid x_1 \ge 1\right\}$$

The ground truth solution is

$$q(\mathbf{x}) = f(x_1)$$

where $f(x_1)$ solves a 1D BVP:

$$\frac{d^2 f\left(x_1\right)}{dx_1^2} - 4x_1\left(x_1^2 - 1\right)\frac{df\left(x_1\right)}{dx_1} = 0, \quad f(-1) = 0, \quad f(1) = 1$$

Committor Function for Rare Events

Song, Cameron, Yang arXiv:2306.12268, SISC, 2025

FEX identifies the following representation

Eqn 1: $\alpha_{1,1}x_1 + \ldots + \alpha_{1,10}x_{10} + \beta_1$

Eqn 2: $\alpha_{2,1}\tanh(x_1) + \ldots + \alpha_{2,10}\tanh(x_{10}) + \beta_2$

$\mathcal{J}(\mathbf{x}) = \alpha_3 \tanh(\text{Eqn 1} + \text{Eqn 2}) + \beta_3$

where $\alpha_3 = 0.5,\ \beta_3 = 0.5$

	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9	α_{10}	β
Eqn 1	**1.6798**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Eqn 2	**1.9039**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

FEX discovers $q(\mathbf{x}) = f(x_1)$ and high-dimensional → low-dimension

FEX for Learning Physical Laws

2D Burgers equation with periodic boundary conditions on $(x, y, t) \in [0,2\pi]^2 \times [0,10]$:

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} + \nu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2})$$

$$\frac{\partial v}{\partial t} = -u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y} + \nu(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2})$$

$$u(x, y, 0) = u_0(x, y)$$

$$v(x, y, 0) = v_0(x, y)$$

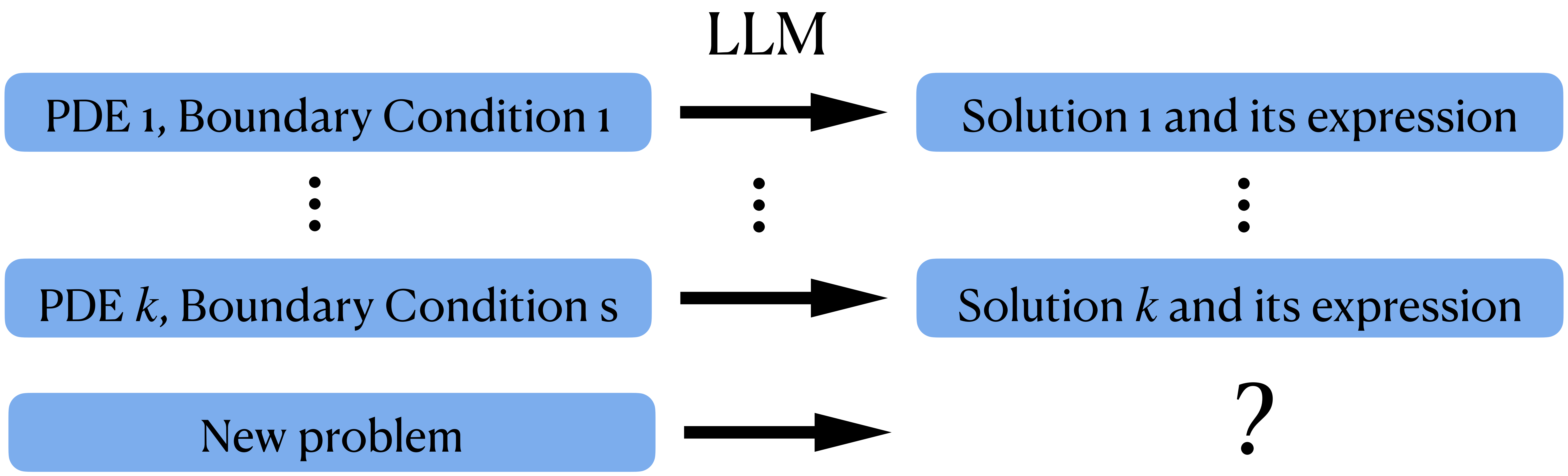
$$\nu = 0.1$$

	PDE-Net 2.0	SINDy	GP	SPL	FEX
Mean Absolute Error	1.086×10^{-3}	3.239×10^{-1}	4.973×10^{-1}	2.1×10^{-1}	2.021×10^{-4}

Unraveling Symbolic Structures in FEX with LLMs

Bhatnagar, Liang, Patel, Y., arXiv:2503.09986

- Methodology: Fine-tune LLM & in-context learning



- Advantages: faster convergence & higher accuracy

Unraveling Symbolic Structures in FEX with LLMs

Bhatnagar, Liang, Patel, Y., arXiv:2503.09986

Operator information preserved in solution from condition

Theorem $-\Delta u = f$ in Ω and $u = g$ on $\partial\Omega$. Reasonable smoothness for f and g. $\forall \delta \in (0,1)$, $\exists \bar{u} : \mathbb{R}^d \to \mathbb{R}$ such that

- $\|u - \bar{u}\|_{L^2(\Omega)} \leq \delta$
- $\bar{u}$ only has operators $+$, $\times$, $(\,\cdot\,)^2$, and those in f, g, and $\mathrm{dist}(x, \partial\Omega)$
- $\#\mathrm{ops}(\bar{u}) = (\#\mathrm{ops}(f) + \#\mathrm{ops}(g)) \times \mathrm{poly}(\frac{1}{\delta})$

Results generalize to time-dependent problems.

Unraveling Symbolic Structures in FEX with LLMs

Bhatnagar, Liang, Patel, Y., arXiv:2503.09986

PDE Type	Metric	Uninformed FEX	LLM-Informed FEX	Speedup
Conservation	Avg. Iter	28.52	5.23	5.45×
	Avg. Time (m)	23.67	3.95	6.00×
Poisson	Avg. Iter	28.73	6.47	4.44×
	Avg. Time (m)	62.07	14.09	4.41×

$u(x)$	Method	Binary Size	Unary Size	Iters	Time [m]	Error
$4\cos\big(4x^2\cos(x_0)\big)$	**LLM-informed**	1	2	4.25	8.25	0
	Uninformed	3	9	167	340	0
$4\,x_1^3+4\,x_1^2+2\cos\big(4\,x_1^3\cos(x_0)\big)$	**LLM-informed**	2	4	102	286	10^{-8}
	Uninformed[6]	3	9	2000+	2400+	N/A

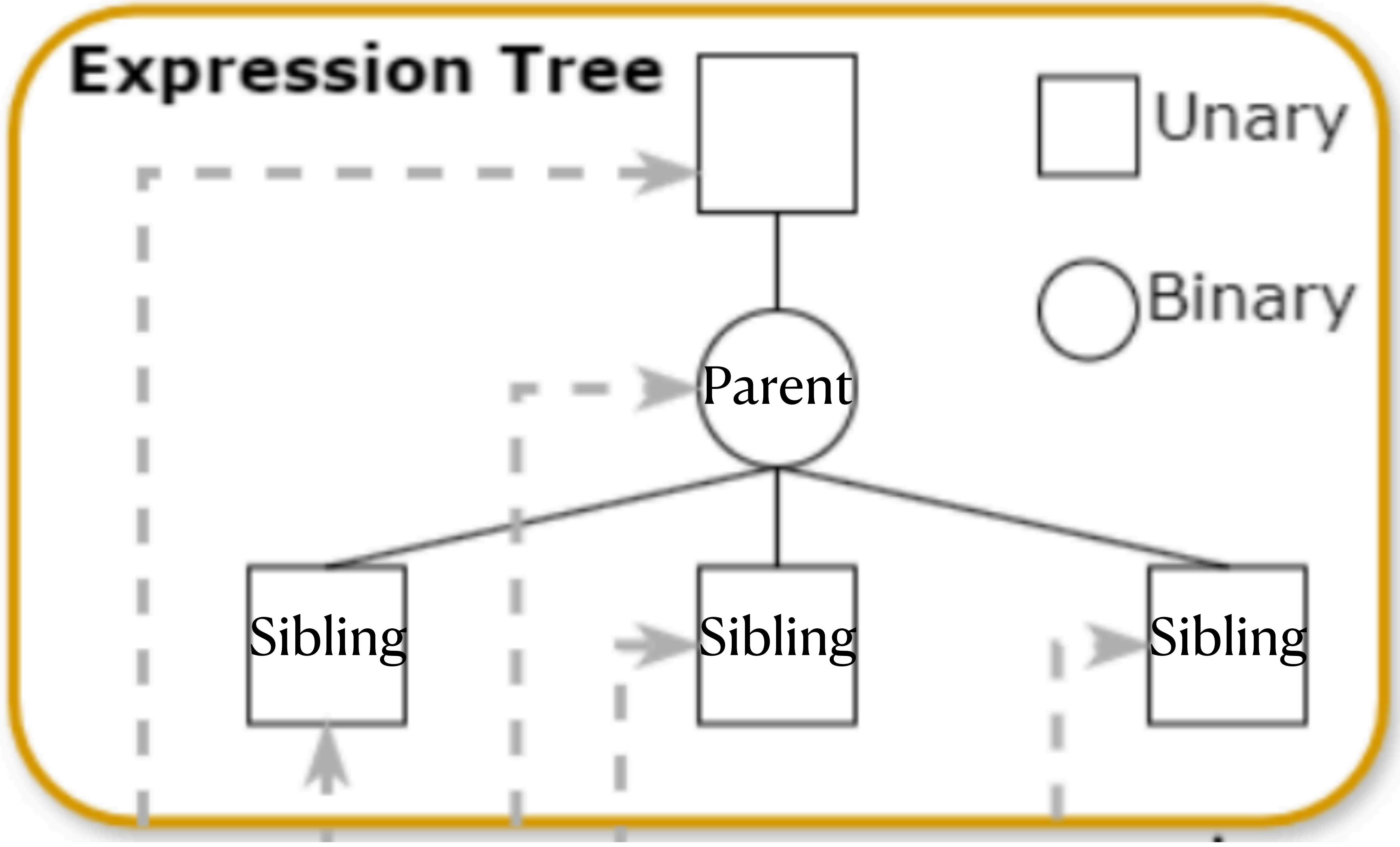
Bayesian Symbolic Learning

Huang, Wen, Adusumilli, Choudhary, Y., arXiv:2503.09592

Scientific Paper → Domain Specific Expressions → Domain Specific Priors → Optimization

Domain Specific Priors
- Pr(·| Parent)
- Pr(·| Parent, siblings)
- Formulation rules

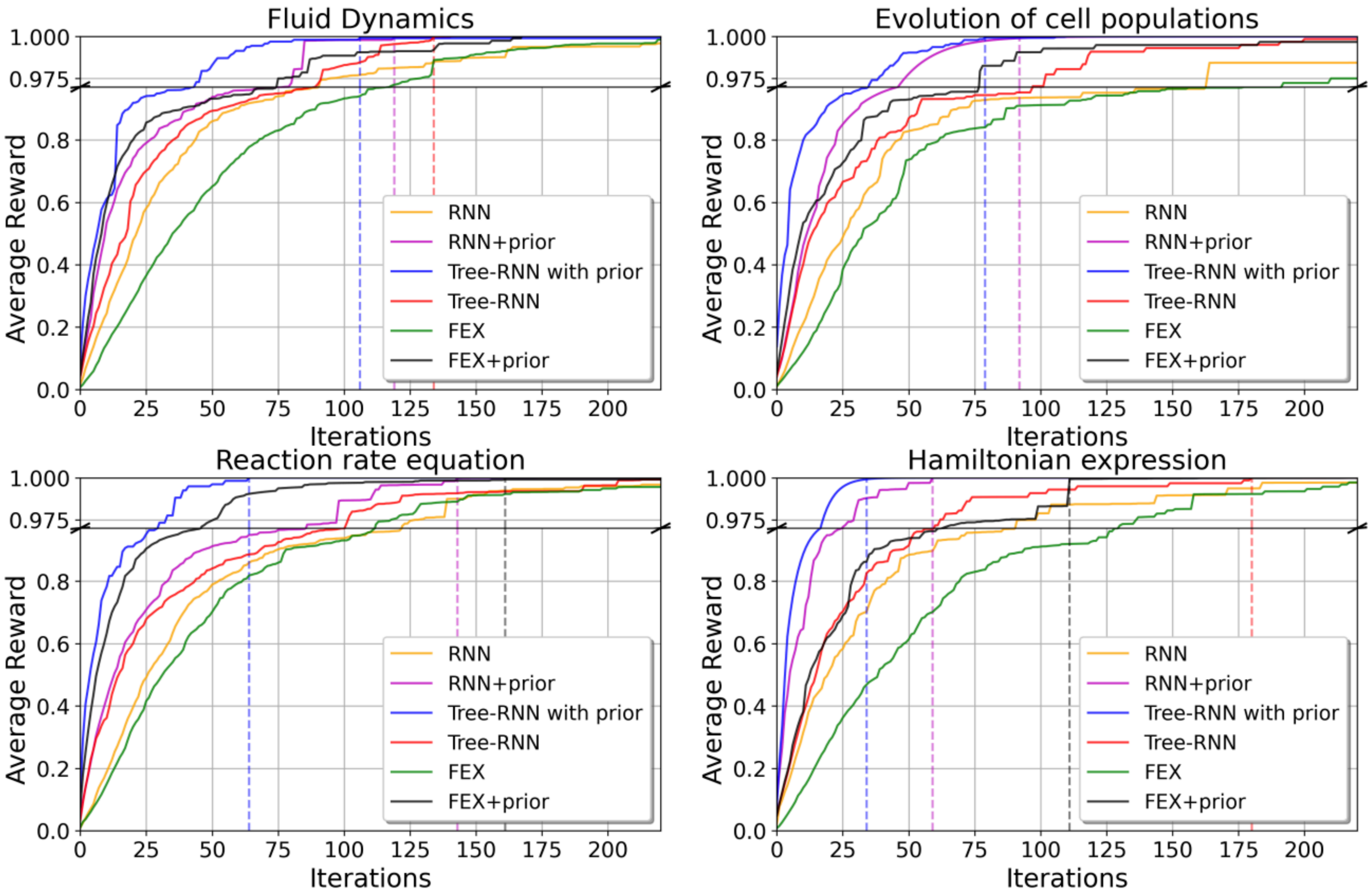
Optimization
- Better samples
- Faster convergence
- Eliminate artifacts



Fill in Operators

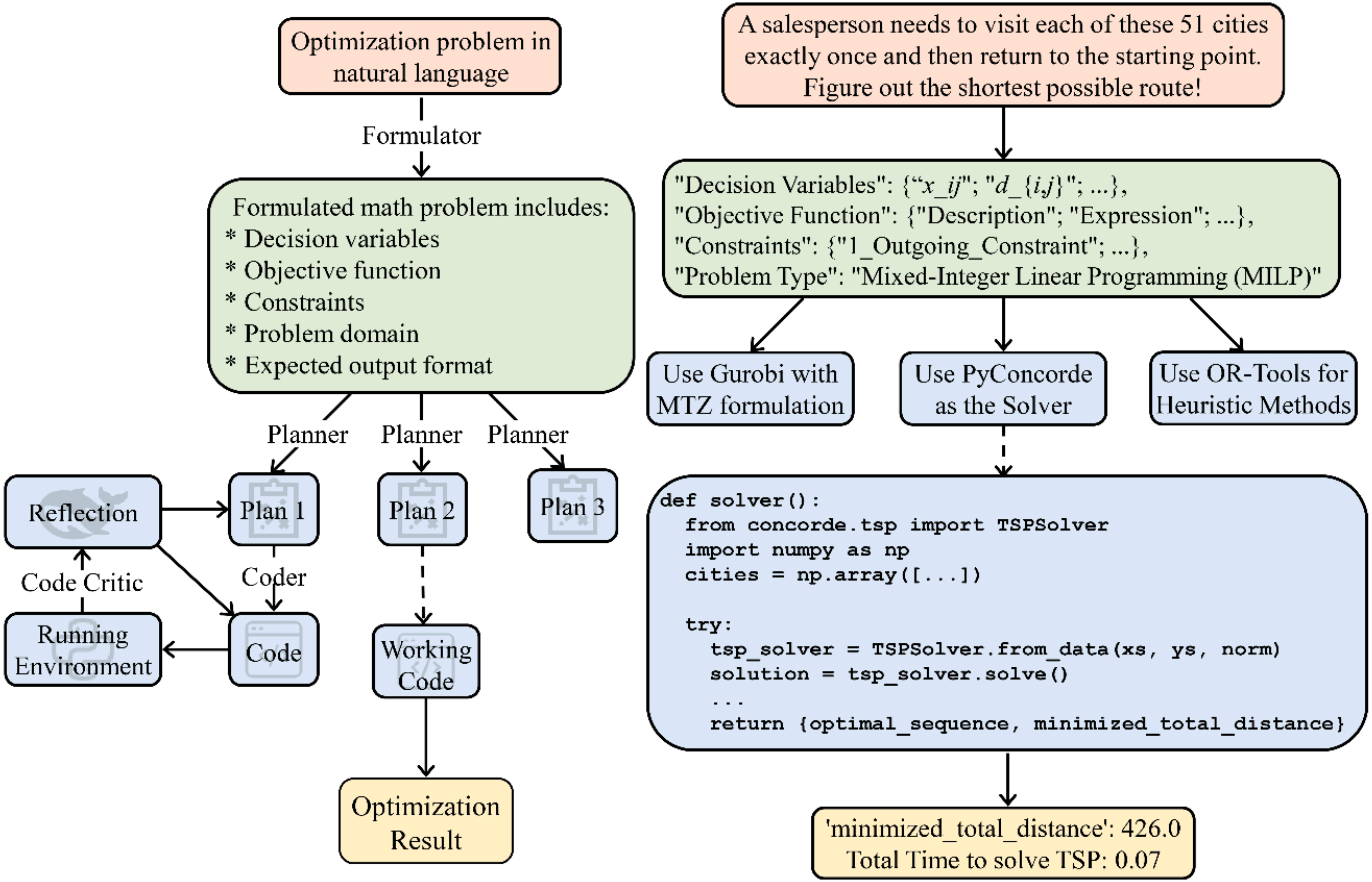
Bayesian Symbolic Learning

Huang, Wen, Adusumilli, Choudhary, Y., arXiv:2503.09592



- Symbolic learning (Finite Expression Method)
- **Large language model (LLM) for modeling and computing assistant**
  - **Example**: automatic optimization modeling, solving, and testing

Overview and New Features of OptimAI



LLM Agents for Modeling & Computing from Natural Language

OptimAI: Thind, Sun, Liang, Y., arXiv:2504.16918

Table 1: Comparison of Functional Capabilities between OptimAI and Prior Methods.

Functional Capabilities	OptiMUS	Optibench	CoE	OptimAI
Natural language input	✗	✓	✓	✓
Planning before coding	✗	✗	✓	✓
Multi-solver support	✗	✗	✗	✓
Switching between plans	✗	✗	✗	✓
Code generation	✓	✓	✓	✓
Distinct LLM collaboration	✗	✗	✗	✓

LLM Agents for Modeling & Computing from Natural Language

OptimAI: Thind, Sun, Liang, Y., arXiv:2504.16918

Table 2: Previous work on using LLMs for optimization.

Work	Dataset Proposed	Size	Problem Type(s)
NL4Opt Competition [8]	NL4Opt	289	LP
Chain-of-Experts (CoE) [9]	ComplexOR	37	LP, MILP
OptiMUS [10, 11, 12]	NLP4LP	67	LP, MILP
Optibench [13]	Optibench	605	LP, NLP, MILP, MINLP
OR-LLM-Agent[14]	OR-LLM-Agent	83	LP, MILP

Abbreviations: LP - Linear Programming, NLP - Nonlinear Programming, MI - Mixed-Integer.

LLM Agents for Modeling & Computing from Natural Language

OptimAI: Thind, Sun, Liang, Y., arXiv:2504.16918

Table 3: Accuracy comparison between OptimAI and state-of-the-art methods.

Agent \ Dataset	NLP4LP	Optibench Linear w/o Tab.	Optibench Linear w/ Tab.	Optibench Nonlin. w/o Tab.	Optibench Nonlin. w/ Tab.
OptiMUS [11]	71.6%	-	-	-	-
Optibench [13]	-	75.4%	62.5%	42.1%	32.0%
Ours w/ GPT-4o	79.1%	81.2%	73.8%	72.0%	48.0%
Ours w/ GPT-4o+o1-mini	**88.1%**	84.2%	**80.0%**	77.3%	56.0%
Ours w/ QwQ (by Qwen)	79.1%	86.2%	77.5%	**81.6%**	50.0%
Ours w/ DeepSeek-R1	82.1%	**87.4%**	78.8%	79.5%	**60.0%**

All evaluations were conducted under a zero-shot prompting setting. GPT-4o+o1-mini refers to using o1-mini as the planner while employing GPT-4o for all other roles.

LLM Agents for Modeling & Computing from Natural Language

OptimAI: Thind, Sun, Liang, Y., arXiv:2504.16918

Table 4: Generalization of OptimAI across NP-hard combinatorial optimization problems.

	Math Programming	TSP	JSP	Set Covering
OptimAI	✓	✓	✓	✓
OptiMUS	✓	✗	✗	✗
Optibench	✓	✗	✗	✗

Traveling salesman problem (TSP), job shop scheduling problem (JSP), and set covering problem.

LLM Agents for Modeling & Computing from Natural Language

OptimAI: Thind, Sun, Liang, Y., arXiv:2504.16918

Table 5: Synergistic effects of combining heterogeneous LLMs.

Planner \ Remaining Roles	Llama 3.3 70B	DeepSeek-R1 14B	Gemma 2 27B
Llama 3.3 70B	59%	54%	54%
DeepSeek-R1 14B	**68%**	50%	41%
Gemma 2 27B	**77%**	**59%**	54%

LLM Agents for Modeling & Computing from Natural Language

OptimAI: Thind, Sun, Liang, Y., arXiv:2504.16918

Table 6: Ablation study of OptimAI design.

Formulator	Planner	Code Critic	Revisions	Executability	Productivity
✓	✓	✓	**1.7**	**3.6**	**6.8**
✗	✓	✓	2.0	3.2	6.3
✓	✗	✓	7.8	3.1	1.2
✓	✓	✗	6.2	3.3	2.2

Take Home Messages

- Modeling and Computing **in the language space**
- **Descriptive structures** ease challenges and form a right space
- Language space admits **automatic big** search