DISCOVERING INTERPRETABLE PHYSICAL MODELS USING SR AND DEC

Simone Manti, Alessandro Lucantonio
Department of Mechanical and Production Engineering, Aarhus University

Funded by
the European Union

European Research Council
Established by the European Commission

INTRODUCTION

Problem: use ML to discover new Physics

Why? Still unable to accurately model many physical phenomena (e.g. biological systems)

How? Many directions:

1. Black-box methods (e.g. Neural Networks).

   Pros: many developed and tested models, more solid theory.

   Cons: difficult to interpret, requires large datasets.

2. Symbolic methods (e.g. Symbolic Regression models).

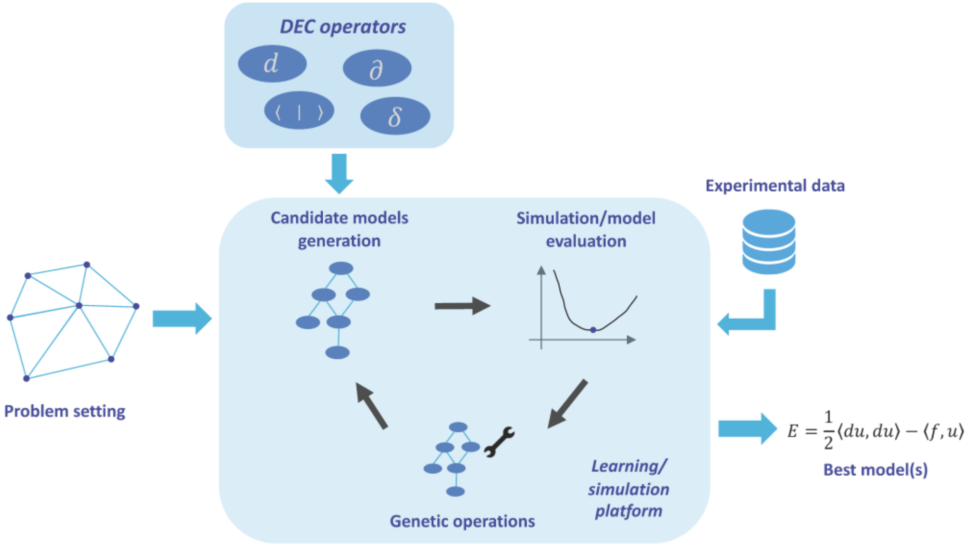
   Pros: interpretability + less data required (more constrained)

   Cons: algebraic equations (majority)

OUR CONTRIBUTION

This work: Develop a new symbolic method combining Symbolic Regression and Discrete Exterior Calculus to discover new physical models starting from data.

- General-purpose method designed for field problems
- The output of the method is an equation -> easy to interpret
- Working to extend it to face real-world and open problems
- Two open-source libraries: *dctkit* (to manage discrete mathematical tools) and *alpine* (to implement the learning strategy)


DEC operators
d
∂
$\langle \mid \rangle$
δ
Experimental data
Candidate models generation
Simulation/model evaluation
Problem setting
$E = \frac{1}{2}\langle du, du\rangle - \langle f, u\rangle$
Best model(s)
Learning/ simulation platform
Genetic operations

dctkit

alpine

STATE-OF-THE-ART COMPARISON

		Ours	PySINDy [1]	EQL [2]	Eureqa [3]	DSR [4]	AI Feynman [5]	PySR [6]
Field Problems		✓	✓	×	×	×	×	×
	Domain source	✓	×	×	×	×	×	×
	Stationary/ Non-stationary	✓/✓	×/✓	×/×	×/×	×/×	×/×	×/×
	Variational/Non-variational	✓/✓	×/✓	×/×	×/×	×/×	×/×	×/×
Dynamical Systems		×	✓	✓	✓	×	×	×
Algebraic Equations		✓	✓	✓	✓	✓	✓	✓

[1] Kaptanoglu A A et al.,PySINDy: A comprehensive Python package for robust sparse system identification, *Journal of Open Source Software*

[2] Sahoo S, Lampert C and Martius G, Learning equations for extrapolation and control, *Int. Conf. on Machine Learning*

[3] Schmidt M and Lipson H, Distilling free-form natural laws from experimental data, *Science*

[4] Petersen B K, Landajuela M, Mundhenk T N, Santiago C P, Kim S K and Kim J T, Deep symbolic regression: recovering mathematical expressions from data via risk-seeking policy gradients (arXiv:1912.04871)

[5] Udrescu S-M and Tegmark M, AI Feynman: a physics-inspired method for symbolic regression, *Sci. Adv*

[6] Cranmer M, Interpretable machine learning for science with PySR and SymbolicRegression.jl (arXiv:2305.01582)

DISCRETE EXTERIOR CALCULUS

- **Why?**
  - Discrete theory: no need for discretization schemes
  - Discrete geometric representation: suitable for field problems
  - Concise + effective set of operators: reduced search space

- **What?**
  - Discrete version of Exterior Calculus (differential forms in a manifold)
  - manifold <-> simplicial complex
  - field <-> form <-> cochain
  - grad, div, lap <-> coboundary (d) and hodge star (★)

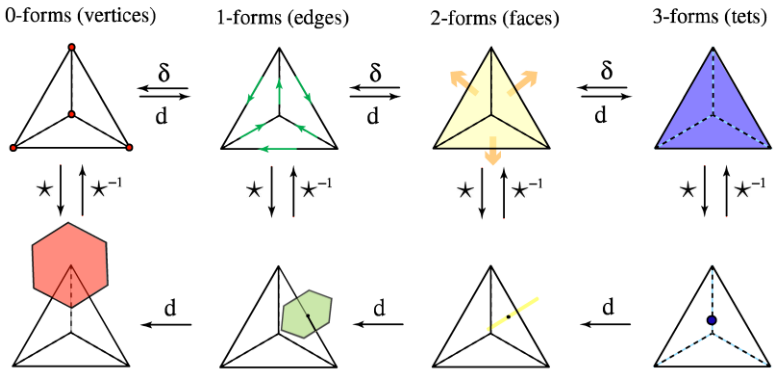


Figure from Desbrun et al., *Discrete Differential forms for Computational Modeling*

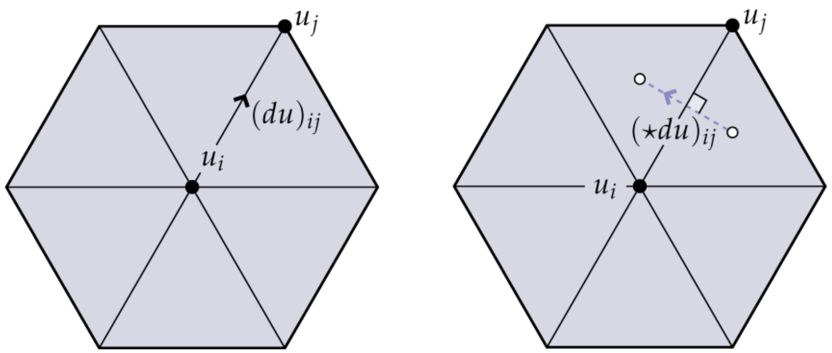
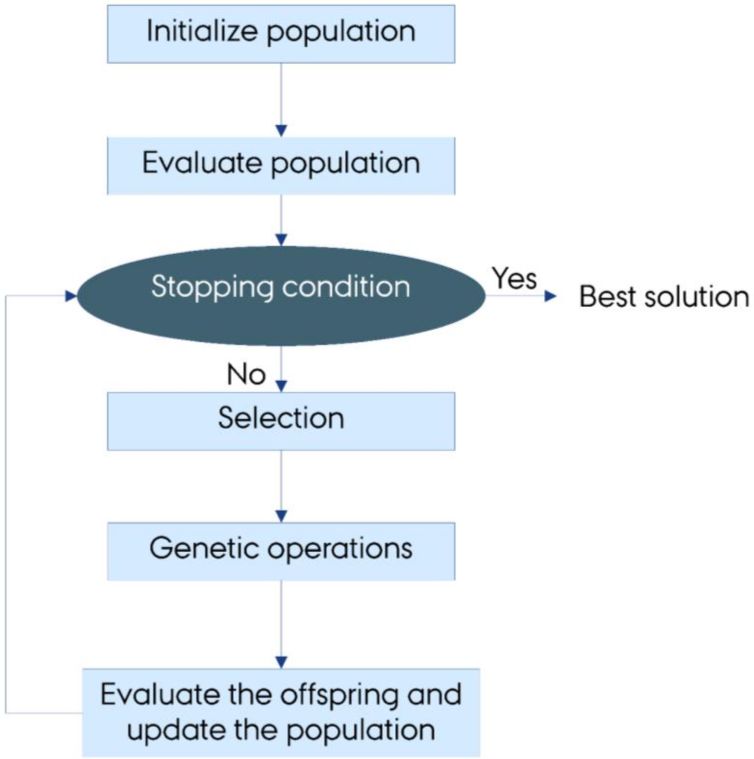


Figure from K.Crane, *Discrete Differential Geometry: An applied introduction*

SYMBOLIC REGRESSION

- **Why?**
  - Generate and manipulate candidate equations describing a given physical system;
  - interpretability;
  - use small dataset for training, validation and test.

- **What?**
  - Symbolic Regression -> find equations given data;
  - Genetic Programming -> evolutionary strategy that explores the space updating an initial population through genetic operations. The goal is to maximise a proper fitness function.



THE METHOD

1. Individual: potential energy or residual

2. Pure GP does not work -> we cannot sum e.g. a cochain with a scalar.
   Sol: Strongly-Typed Genetic Programming -> type consistent trees

3. Dimensionless variables -> every generated expression is physically meaningful

4. Each individual is minimized (for the residual-> its norm) according to initial and boundary conditions. Then, we compare the solution with the true data, maximizing

$$F(I) = -(\alpha \mathrm{MSE}(I) + \eta R(I))$$

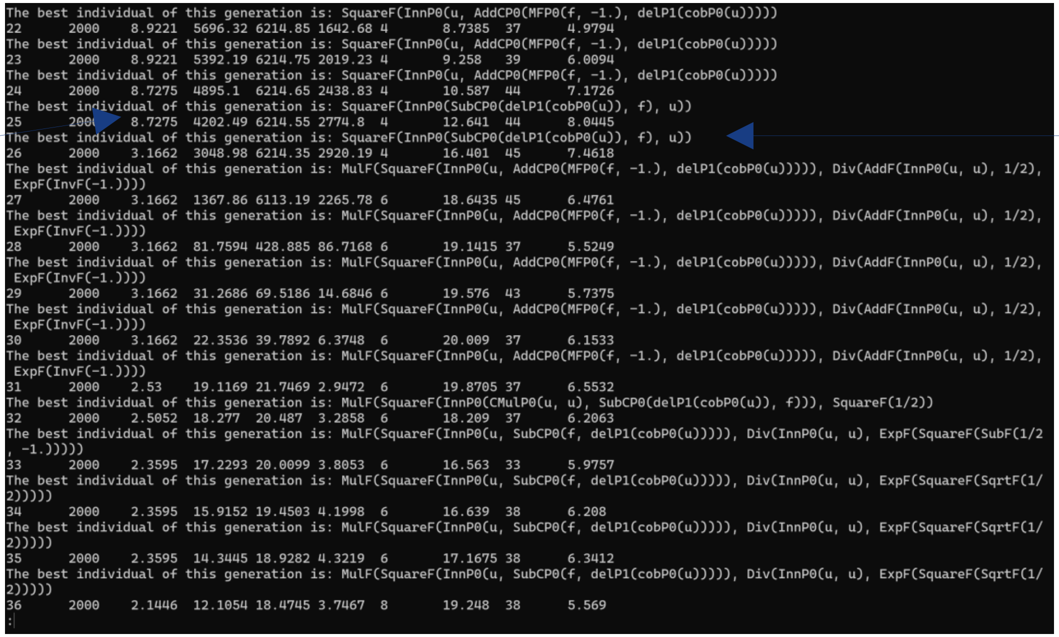
fixed scaling factor (problem dependent)

Regularization hyperparameter

Regularization function

RESULTS - PRELIMINARY INFO

- 3 different benchmarks in variational form: *Poisson, Elastica, Linear Elasticity* equations
- Data are split in training, validation and test (double hold-out).
- 50 final model discovery runs to compute recovery rate or MSE mean ± std



```
The best individual of this generation is: SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u)))))
22      2000    8.9221  5696.32 6214.85 1642.68 4       8.7385  37      4.9794
The best individual of this generation is: SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u)))))
23      2000    8.9221  5392.19 6214.75 2019.23 4       9.258   39      6.0094
The best individual of this generation is: SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u)))))
24      2000    8.7275  4895.1  6214.65 2438.83 4       10.587  44      7.1726
The best individual of this generation is: SquareF(InnP0(SubCP0(delP1(cobP0(u)), f), u))
25      2000    8.7275  4202.49 6214.55 2774.8  4       12.641  44      8.0445
The best individual of this generation is: SquareF(InnP0(SubCP0(delP1(cobP0(u)), f), u))
26      2000    3.1662  3048.98 6214.35 2920.19 4       16.401  45      7.4618
The best individual of this generation is: MulF(SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u))))), Div(AddF(InnP0(u, u), 1/2),
 ExpF(InvF(-1.))))
27      2000    3.1662  1367.86 6113.19 2265.78 6       18.6435 45      6.4761
The best individual of this generation is: MulF(SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u))))), Div(AddF(InnP0(u, u), 1/2),
 ExpF(InvF(-1.))))
28      2000    3.1662  81.7594 428.885 86.7168 6       19.1415 37      5.5249
The best individual of this generation is: MulF(SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u))))), Div(AddF(InnP0(u, u), 1/2),
 ExpF(InvF(-1.))))
29      2000    3.1662  31.2686 69.5186 14.6846 6       19.576  43      5.7375
The best individual of this generation is: MulF(SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u))))), Div(AddF(InnP0(u, u), 1/2),
 ExpF(InvF(-1.))))
30      2000    3.1662  22.3536 39.7892 6.3748  6       20.009  37      6.1533
The best individual of this generation is: MulF(SquareF(InnP0(u, AddCP0(MFP0(f, -1.), delP1(cobP0(u))))), Div(AddF(InnP0(u, u), 1/2),
 ExpF(InvF(-1.))))
31      2000    2.53    19.1169 21.7469 2.9472  6       19.8705 37      6.5532
The best individual of this generation is: MulF(SquareF(InnP0(CMulP0(u, u), SubCP0(delP1(cobP0(u)), f))), SquareF(1/2))
32      2000    2.5052  18.277  20.487  3.2858  6       18.209  37      6.2063
The best individual of this generation is: MulF(SquareF(InnP0(u, SubCP0(f, delP1(cobP0(u))))), Div(InnP0(u, u), ExpF(SquareF(SubF(1/2
, -1.)))))
33      2000    2.3595  17.2293 20.0099 3.8053  6       16.563  33      5.9757
The best individual of this generation is: MulF(SquareF(InnP0(u, SubCP0(f, delP1(cobP0(u))))), Div(InnP0(u, u), ExpF(SquareF(SqrtF(1/
2)))))
34      2000    2.3595  15.9152 19.4503 4.1998  6       16.639  38      6.208
The best individual of this generation is: MulF(SquareF(InnP0(u, SubCP0(f, delP1(cobP0(u))))), Div(InnP0(u, u), ExpF(SquareF(SqrtF(1/
2)))))
35      2000    2.3595  14.3445 18.9282 4.3219  6       17.1675 38      6.3412
The best individual of this generation is: MulF(SquareF(InnP0(u, SubCP0(f, delP1(cobP0(u))))), Div(InnP0(u, u), ExpF(SquareF(SqrtF(1/
2)))))
36      2000    2.1446  12.1054 18.4745 3.7467  8       19.248  38      5.569
:
```

Training fitness

Best individual so far

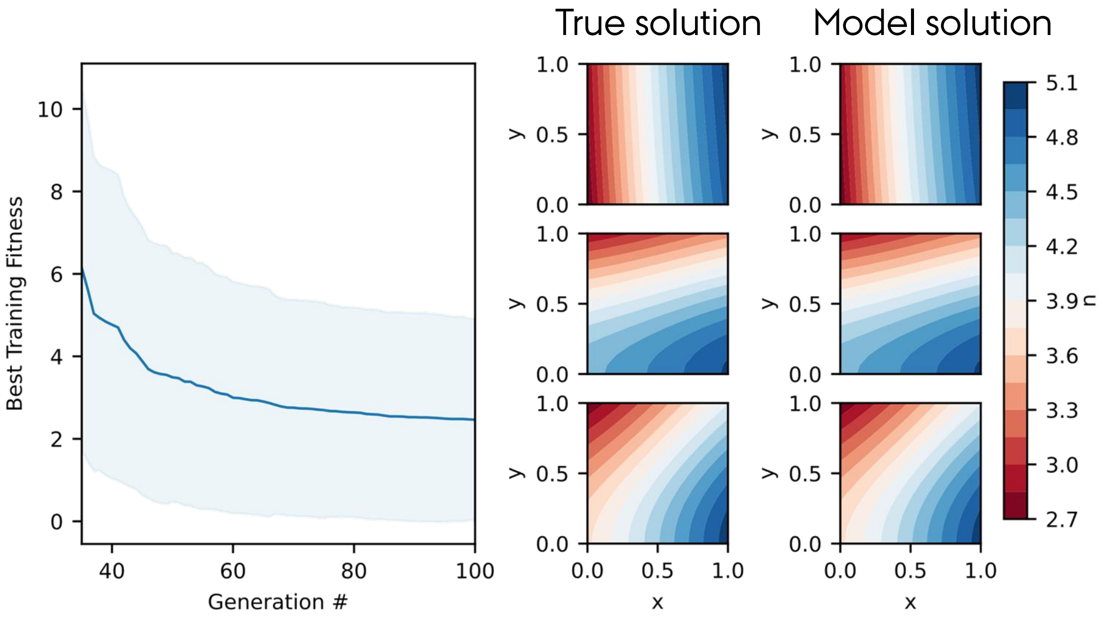
A look inside *alpine*

RESULTS – 2D POISSON

Dirichlet energy using DEC operators:

$$\mathcal{E}_{\mathrm{p}}(u) := \frac{1}{2}\langle du, du\rangle - \langle u, f\rangle = \frac{1}{2}\langle \delta du, u\rangle - \langle u, f\rangle.$$

- Task: learn the correct energy
- Full dataset (training + validation + test): 12 (u,f) pairs without noise
- Dirichlet BCs
- Recovery rate: 66% (best hyperparameters)



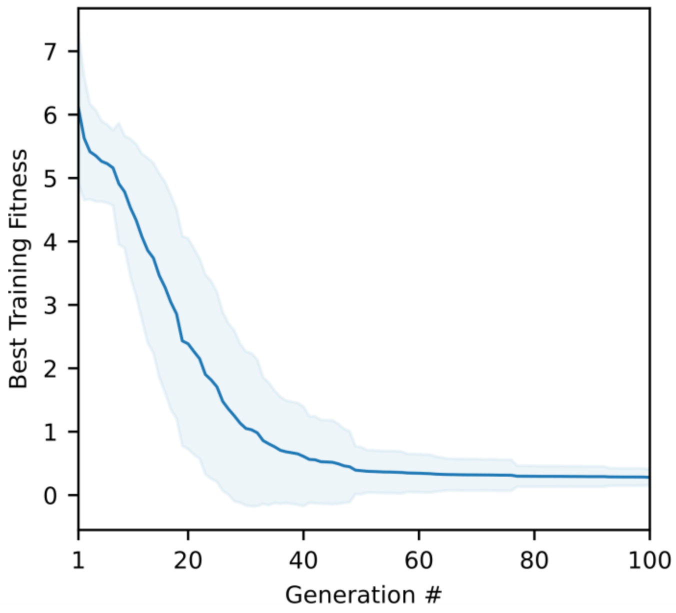
#	Energy	Training Fit.	Test Fit.	Test MSE
1	$\langle u, \delta(d(u/2) - f)\rangle$	0.9	0.9	$9.8 \cdot 10^{-10}$
2	$\langle u, \delta(du) - 2f\rangle$	0.9	0.9	$9.8 \cdot 10^{-10}$
3	$\langle \star(\star u), \delta(du) - 2f\rangle$	1.1	1.1	$9.8 \cdot 10^{-10}$
4	$\langle du, du\rangle - \langle f, 2u\rangle$	1.1	1.1	$9.8 \cdot 10^{-10}$

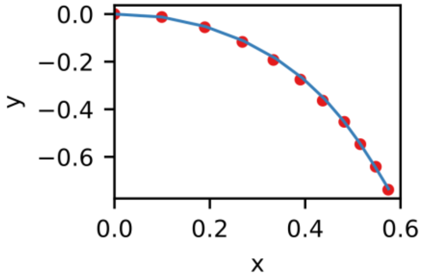
RESULTS – EULER'S ELASTICA

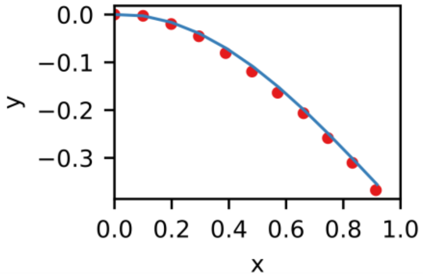
$$\mathcal{E}_{\text{el}}(u) := \frac{1}{2}\langle \mathbb{1}_{\text{int}} \odot \star d^\star u, \mathbb{1}_{\text{int}} \odot \star d^\star u\rangle - \langle f\mathbb{1}, \sin u\rangle$$

- Task: learn the correct energy and the best related B
- f = PL^2/B, where P is the vertical component of the load and B is the *bending stiffness* of the rod
- Full dataset: 10 pairs (u, PL^2) perturbed with uniform noise
- To autotune the constant B we solve at each time

$$\min_{f\geq 0} \quad \|u_f - \bar{u}\|^2 \qquad \text{s.t. } u_f \in \underset{u\,:\,u(0)=0}{\arg\min}\ \mathcal{E}(u, f)$$





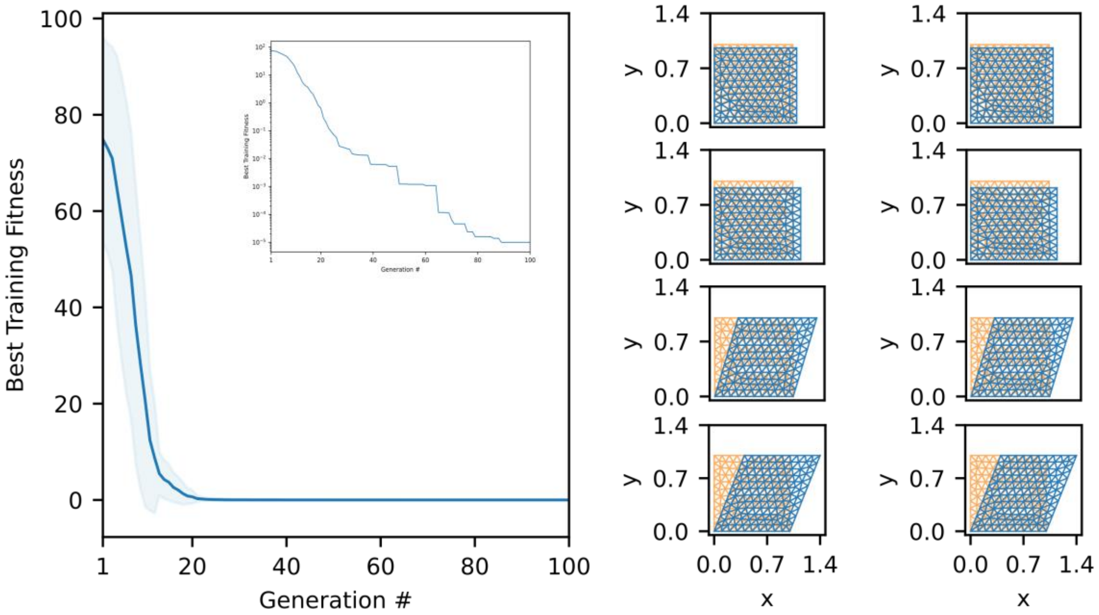


#	Energy	Training Fit.	Test Fit.	Test MSE	B
1	$\langle \star\mathbb{1}_{\text{int}}, (d^\star u)^2\rangle - \langle \sin u, f\mathbb{1}\rangle$	0.196	0.1939	0.0084	37.0312 Nm^2
2	$\langle \arccos(-1)\sin u + \delta^\star(\sin(\sin(d^\star u))), u - f\mathbb{1}\rangle$	0.2123	0.2247	0.0075	7.1779 Nm^2
3	$\langle u - f\mathbb{1}, \delta^\star(\sin(\sin(d^\star u))) + 1/\exp(-1)\sin u\rangle$	0.214	0.2168	0.0067	6.8262 Nm^2
4	$\langle \star\mathbb{1}_{\text{int}}, (d^\star u)^2\rangle - \langle f\mathbb{1}, \sin u\rangle + 1/2$	0.216	0.2139	0.0084	37.0312 Nm^2

RESULTS – 2D LINEAR ELASTICITY

$$\mathcal{E}_{LE}(\boldsymbol{F}) := \frac{1}{2}\langle \boldsymbol{F} + \boldsymbol{F}^T - 2\boldsymbol{I} + \frac{\lambda_-}{\mu_-}\,\mathrm{tr}\left(\frac{1}{2}(\boldsymbol{F} + \boldsymbol{F}^T) - \boldsymbol{I}\right)\boldsymbol{I}, \frac{1}{2}(\boldsymbol{F} + \boldsymbol{F}^T) - \boldsymbol{I}\rangle$$

- Task: learn the energy as a function of **F**
- Full dataset: 20 node coordinates in the deformed configuration (homogeneous deformations)
- Implemented an internal filter (during training) for frame indifference
- Recovery rate: 92% with the best hyperparameter set
- Some of the learned energies are equivalent to the correct one only if **F** is constant

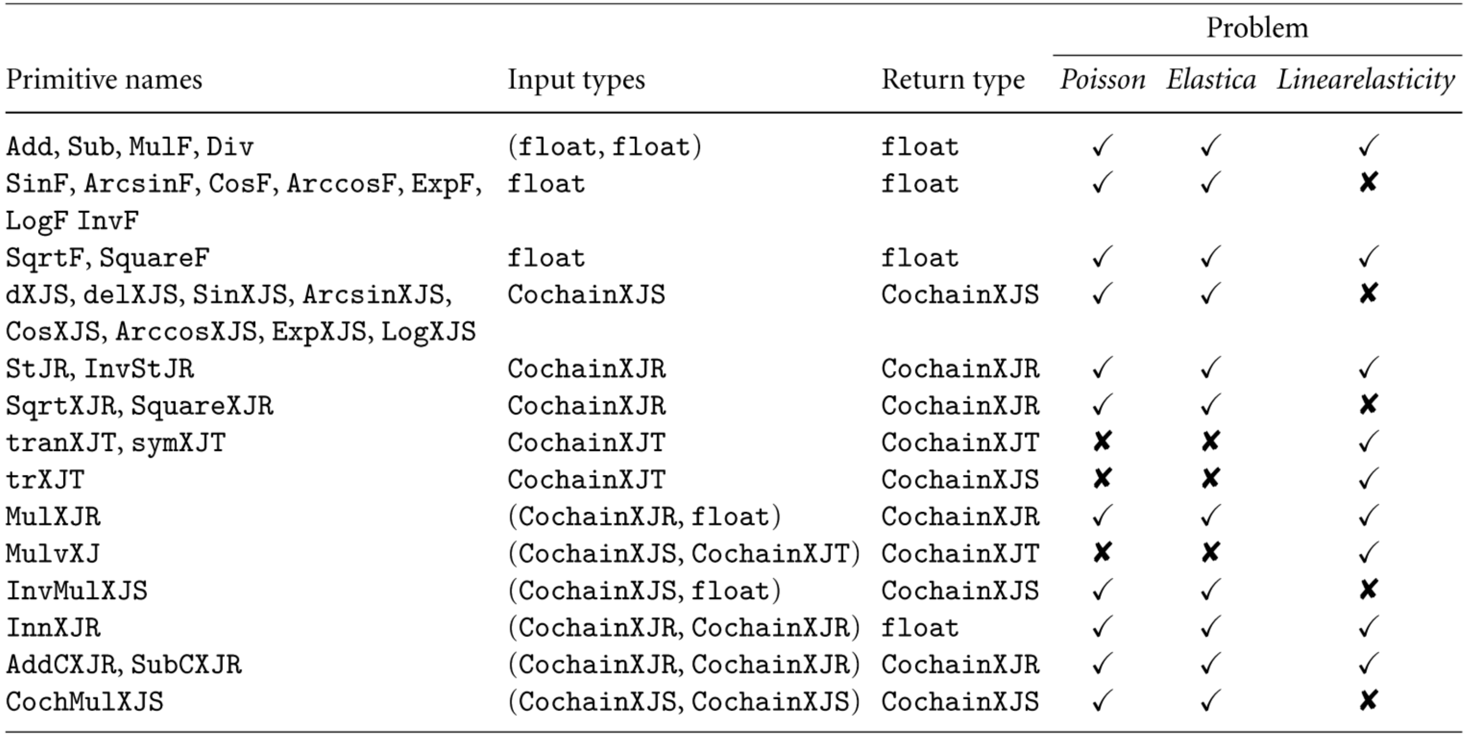


#	Energy	Training MSE	Test MSE
1	$\langle \boldsymbol{I} - \boldsymbol{F}^T, \boldsymbol{I}\rangle^2 - 0.1(\langle \boldsymbol{I} - \boldsymbol{F}^T, \mathrm{sym}(\boldsymbol{F}) - \boldsymbol{I}\rangle + \langle \boldsymbol{I} - \mathrm{sym}(\boldsymbol{F}), \mathrm{sym}(\boldsymbol{F}) - \boldsymbol{I}\rangle)$	0	$1.1672 \cdot 10^{-16}$
2	$\langle -0.5\boldsymbol{I} + 0.5\,\mathrm{sym}(\boldsymbol{F}), \mathrm{sym}(\boldsymbol{F}) - \boldsymbol{I} + (\mathrm{tr}(\boldsymbol{F}) - \mathrm{tr}(\boldsymbol{I}))(2\mathrm{tr}(\boldsymbol{I})\boldsymbol{I} + \boldsymbol{I})\rangle$	0	$2.0785 \cdot 10^{-16}$
3	$\langle \boldsymbol{I} - \boldsymbol{F}^T, \boldsymbol{I} - \mathrm{sym}(\boldsymbol{F}) + 5\langle \boldsymbol{I} - \boldsymbol{F}, \boldsymbol{I}\rangle\boldsymbol{I}\rangle$	0	$3.7309 \cdot 10^{-16}$
4	$\langle \mathrm{sym}(\boldsymbol{F}) - \boldsymbol{I}, \mathrm{sym}(\boldsymbol{F}) - \boldsymbol{I}\rangle + 5\langle \boldsymbol{I} - \boldsymbol{F}, \boldsymbol{I}\rangle^2$	0	$8.7033 \cdot 10^{-16}$

Thanks for your attention!

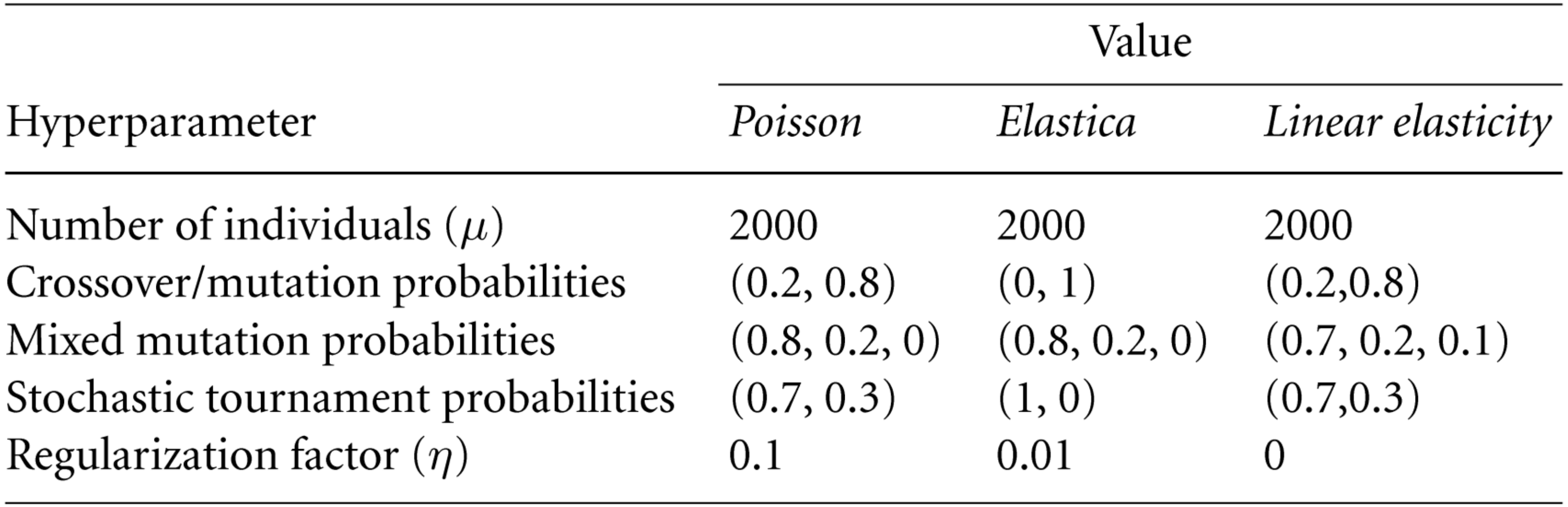
Scan for the full paper

PRIMITIVES



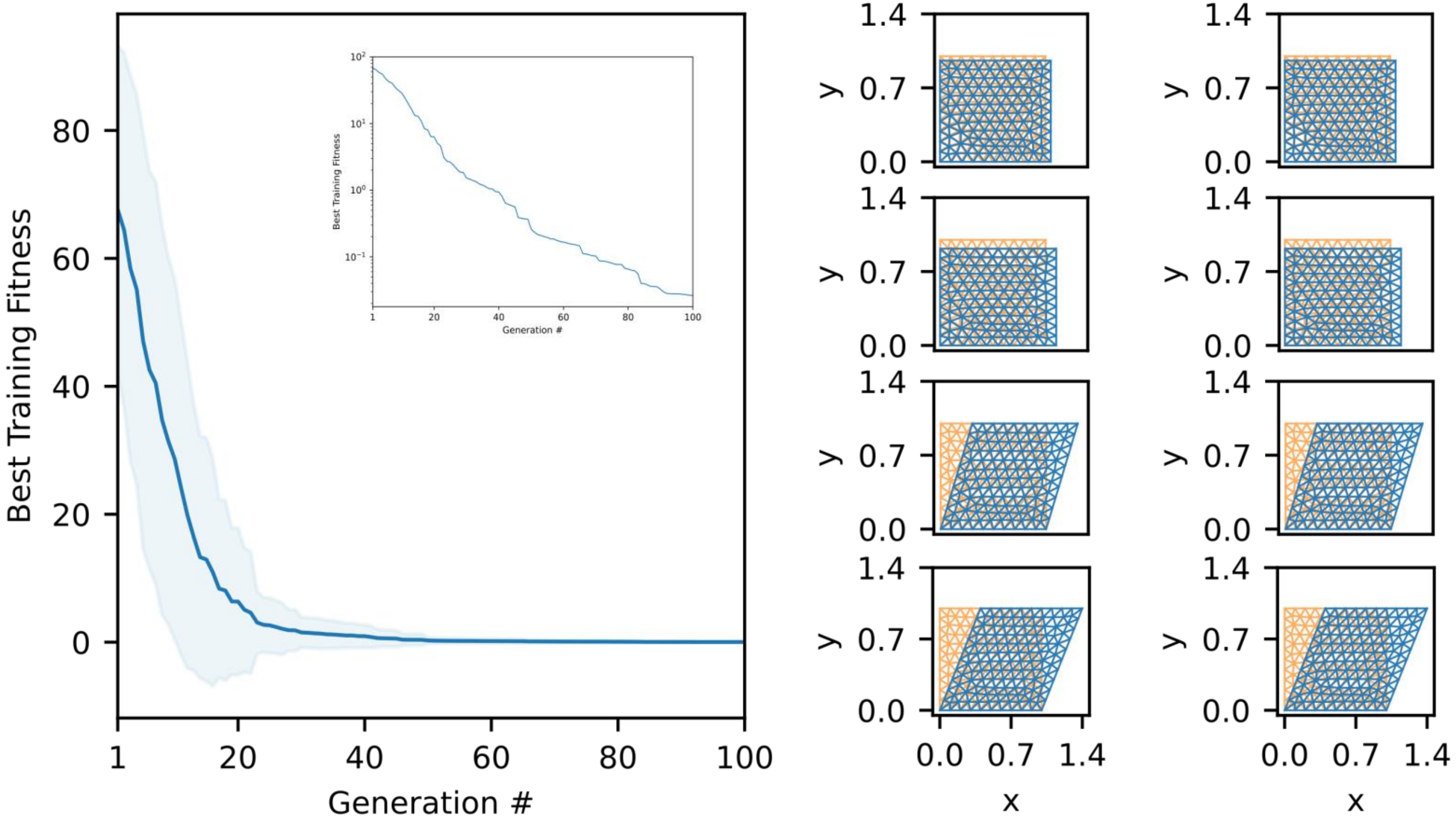
			Problem		
Primitive names	Input types	Return type	*Poisson*	*Elastica*	*Linearelasticity*
Add, Sub, MulF, Div	(float, float)	float	✓	✓	✓
SinF, ArcsinF, CosF, ArccosF, ExpF, LogF InvF	float	float	✓	✓	✘
SqrtF, SquareF	float	float	✓	✓	✓
dXJS, delXJS, SinXJS, ArcsinXJS, CosXJS, ArccosXJS, ExpXJS, LogXJS	CochainXJS	CochainXJS	✓	✓	✘
StJR, InvStJR	CochainXJR	CochainXJR	✓	✓	✓
SqrtXJR, SquareXJR	CochainXJR	CochainXJR	✓	✓	✘
tranXJT, symXJT	CochainXJT	CochainXJT	✘	✘	✓
trXJT	CochainXJT	CochainXJS	✘	✘	✓
MulXJR	(CochainXJR, float)	CochainXJR	✓	✓	✓
MulvXJ	(CochainXJS, CochainXJT)	CochainXJT	✘	✘	✓
InvMulXJS	(CochainXJS, float)	CochainXJS	✓	✓	✘
InnXJR	(CochainXJR, CochainXJR)	float	✓	✓	✓
AddCXJR, SubCXJR	(CochainXJR, CochainXJR)	CochainXJR	✓	✓	✓
CochMulXJS	(CochainXJS, CochainXJS)	CochainXJS	✓	✓	✘

BEST HYPERPARAMETERS



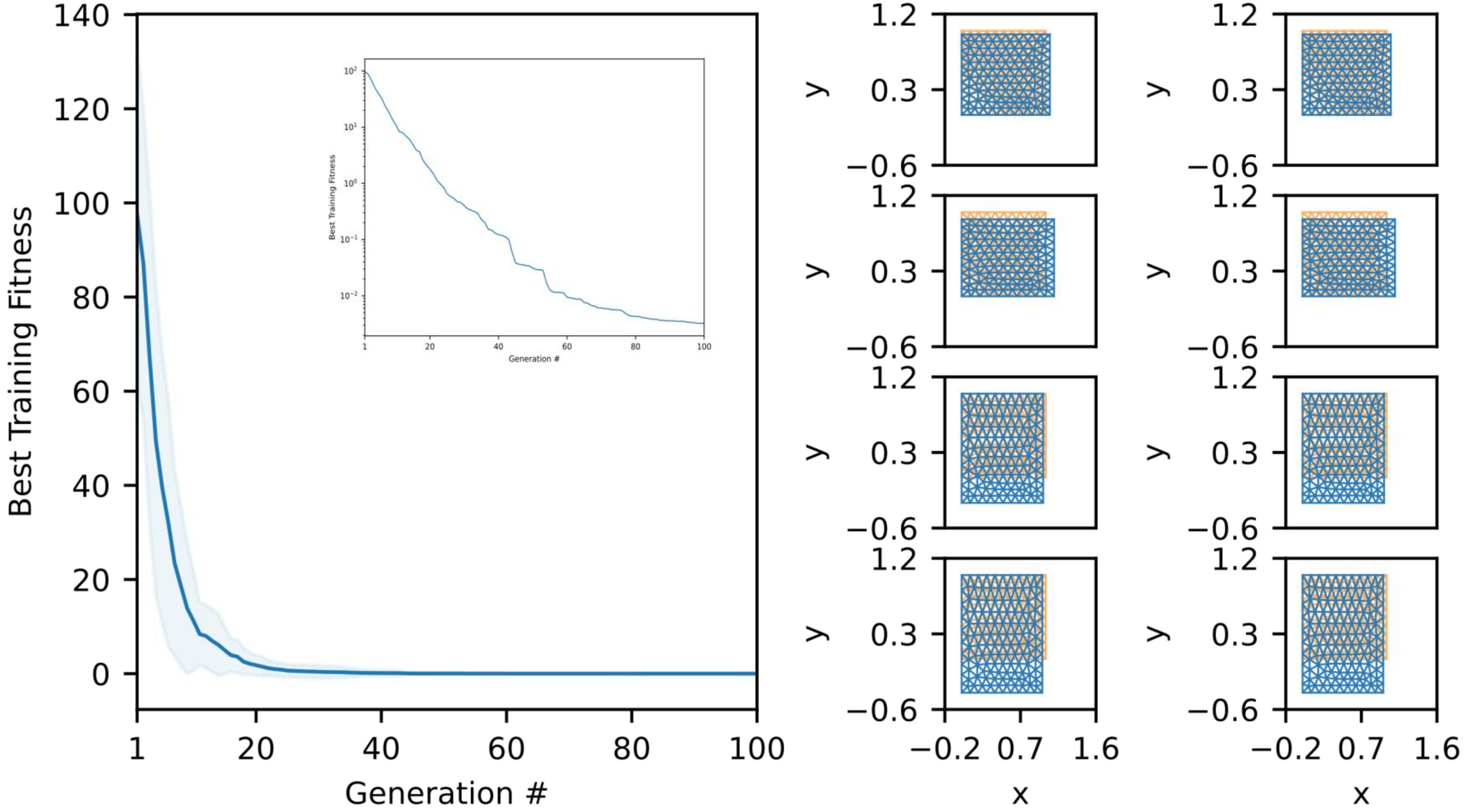
	Value		
Hyperparameter	*Poisson*	*Elastica*	*Linear elasticity*
Number of individuals (μ)	2000	2000	2000
Crossover/mutation probabilities	(0.2, 0.8)	(0, 1)	(0.2,0.8)
Mixed mutation probabilities	(0.8, 0.2, 0)	(0.8, 0.2, 0)	(0.7, 0.2, 0.1)
Stochastic tournament probabilities	(0.7, 0.3)	(1, 0)	(0.7,0.3)
Regularization factor (η)	0.1	0.01	0

2D LINEAR ELASTICITY – STANDARD GP



4% recovery rate

2D LINEAR ELASTICITY – NON HOMOG.


Best Training Fitness
Generation #
10^2
10^1
10^0
10^{-1}
10^{-2}
Best Training Fitness
Generation #
y
x