Automatic Discovery of Analytical Control Laws for Viability Problems

Alberto Tonda, Ph.D., Senior permanent researcher (DR)

UMR 518 Mathématiques et Informatique Appliquées - PS, INRAE, U. Paris-Saclay
UAR 3611 Institut des Systèmes Complexes Paris Ile-de-France

alberto.tonda@inrae.fr

Who am I?

- Career
  - Bachelor and Master in Computer Science Engineering
  - Ph.D. from Politecnico di Torino, Italy, in 2011
  - Permanent researcher in France since late 2012 (INRAE)
  - Senior researcher (DR) since 2023
- Research interests
  - Stochastic multi-objective optimization
  - Machine learning (explainable AI)
  - Biological/agri-food problems

Who do I work for, exactly?

- INRAE pays my salary
  - Institute of research for agriculture, food and the environment
  - ~13,000 people, of which ~4,000 researchers
  - 2nd largest in the world (after US Department of Agriculture)
- MIA-PS (Applied Maths and CS) is my lab
- Associate researcher for the lab Complex Systems Institute
- Université Paris-Saclay is an “umbrella” on top of the rest

The power of ~~friendship~~ collaborative research

Viability theory

Dr. Isabelle Alvarez

Dr. Sophie Martin

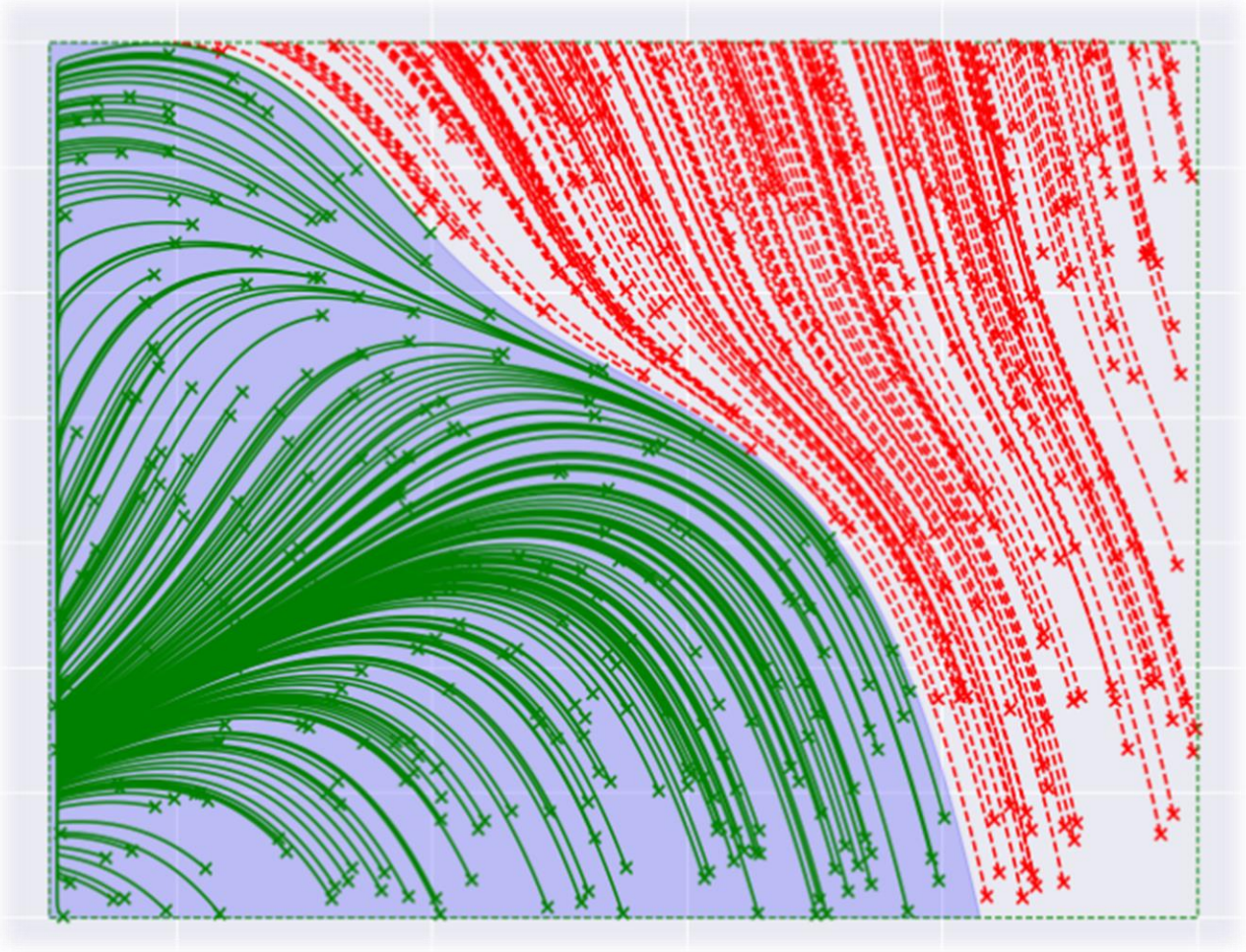
Symbolic regression

Dr. Evelyne Lutton

Prof. Giovanni Squillero

Outline

- Viability theory
- Control laws
- Symbolic regression
- Proposed approach
- Experimental evaluation



Viability theory

- Devised to study the properties of dynamical systems
  - While respecting given **constraints**
  - Problems in ecology, economy, sustainable development...

- Problem description
  - System of ordinary differential equations (ODEs)
  - Control law(s) that can impact trajectories
  - Set of constraints to be respected

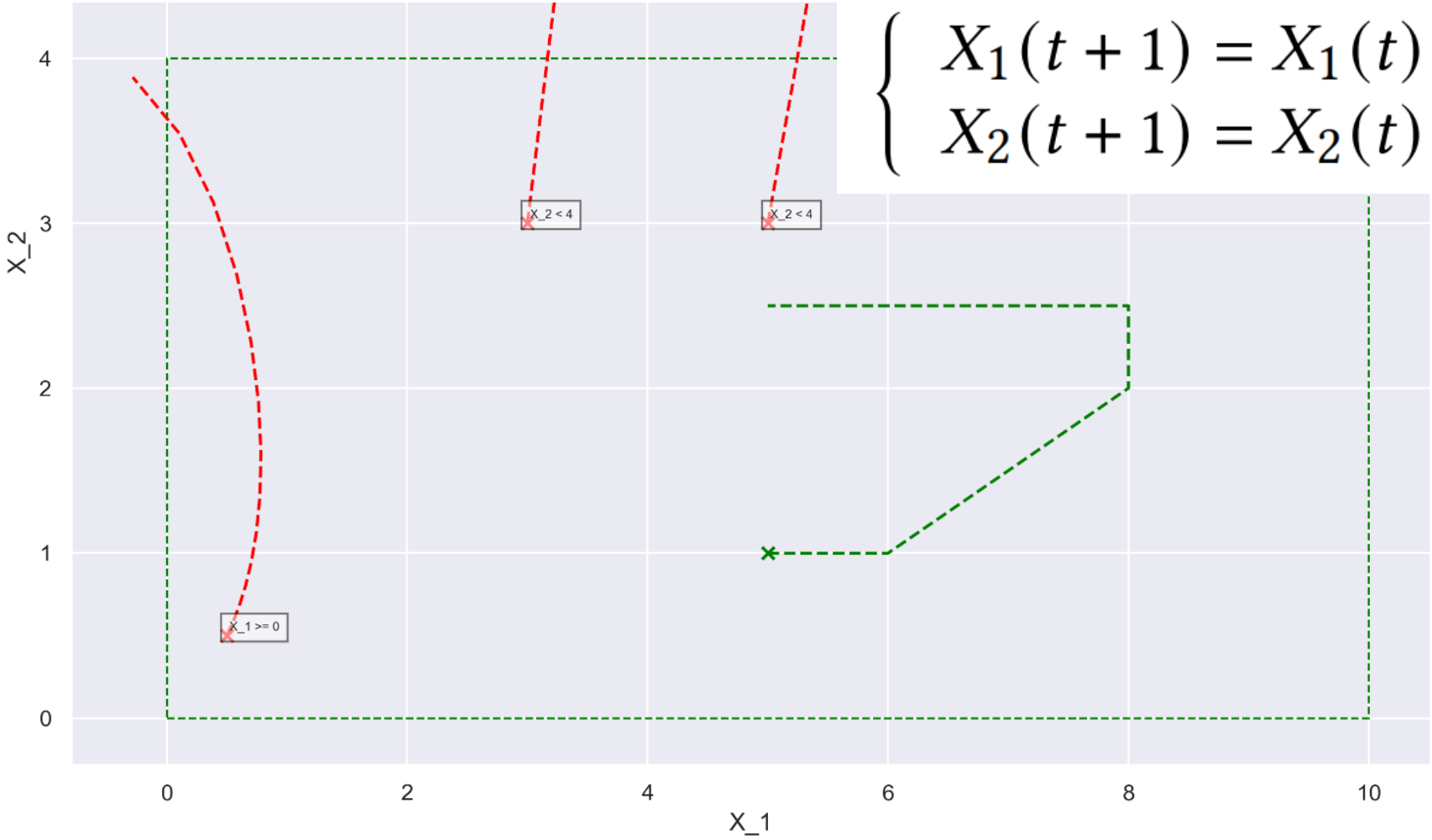
Viability theory

$$\begin{cases} X_1(t+1) = X_1(t) + \mathbf{u(X_1(t), X_2(t), t)} \\ X_2(t+1) = X_2(t) + X_1(t) * X_2(t) \end{cases}$$

$$0 \leq X_1 \leq 10$$
$$0 \leq X_2 \leq 4$$

Alberto TONDA, Team EKINOCS, UMR 518 MIA-PS, INRAE, Université Paris-Saclay
alberto.tonda@inrae.fr

Viability theory



$$\begin{cases} X_1(t+1) = X_1(t) + \mathbf{u(X_1(t), X_2(t), t)} \\ X_2(t+1) = X_2(t) + X_1(t) * X_2(t) \end{cases}$$

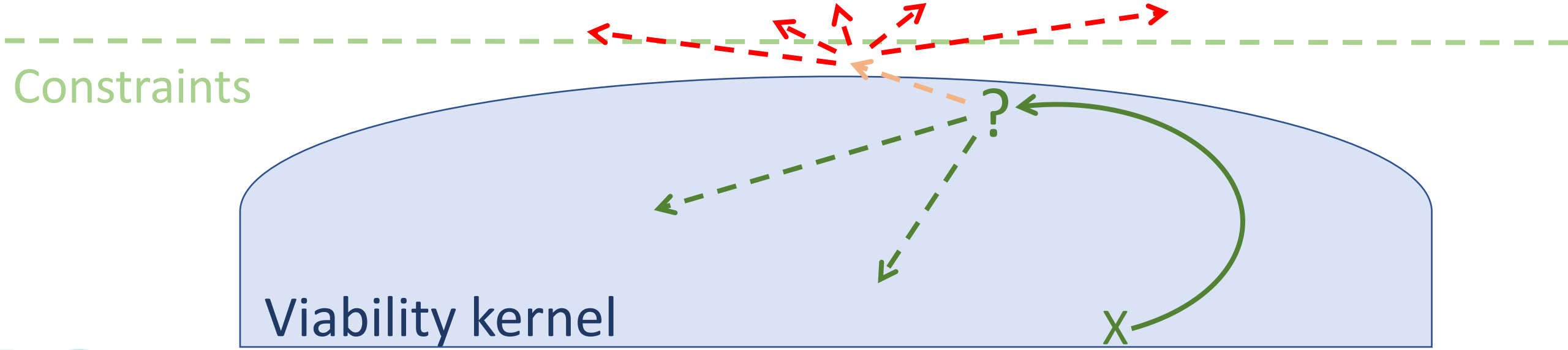
$$0 \leq X_1 \leq 10$$

$$0 \leq X_2 \leq 4$$

Alberto TONDA, Team EKINOCS, UMR 518 MIA-PS, INRAE, Université Paris-Saclay
alberto.tonda@inrae.fr

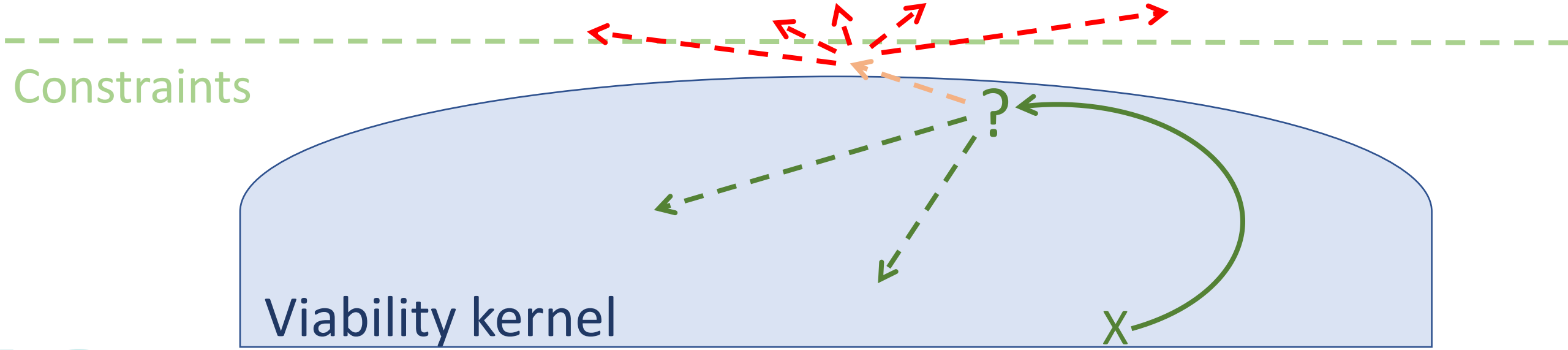
Viability kernel

- Subset of the state space within constraints
- Where it's <u>always possible</u> to control trajectory to stay viable



Limitations of existing approaches

- Current SOTA approach to control laws is <u>not explicit</u>
  - At each time step, user can take a decision
  - Aim is to stay within constraints, but this is not enough
  - Decision requires knowing the **viability kernel**



Limitations of existing approaches

- Viability kernel is not generally known
  - Finding its boundaries is computationally expensive
  - It might be even impossible for a high number of state variables
- It would be desirable to express control with an **equation**
- But how to automatically infer an equation?
  - The equation is not necessarily linear, quadratic, or cubic
  - Learning free-form equations?

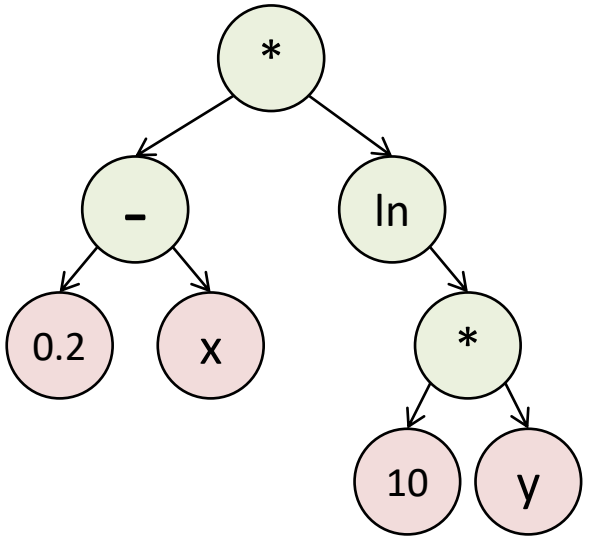
Learning equations as an optimization problem

- Every problem can be framed as an **optimization problem**
  - **Encoding** to describe a candidate solution
  - **Variators** to move between (neighboring) solutions
  - **Evaluation** to assess quality of a solution
  - Defining a **search space**
- **Optimization algorithms** can explore the search space and find high-quality candidate solutions

Alberto TONDA, Team EKINOCS, UMR 518 MIA-PS, INRAE, Université Paris-Saclay
alberto.tonda@inrae.fr

Symbolic regression

- **Encoding** of a candidate equation: binary tree



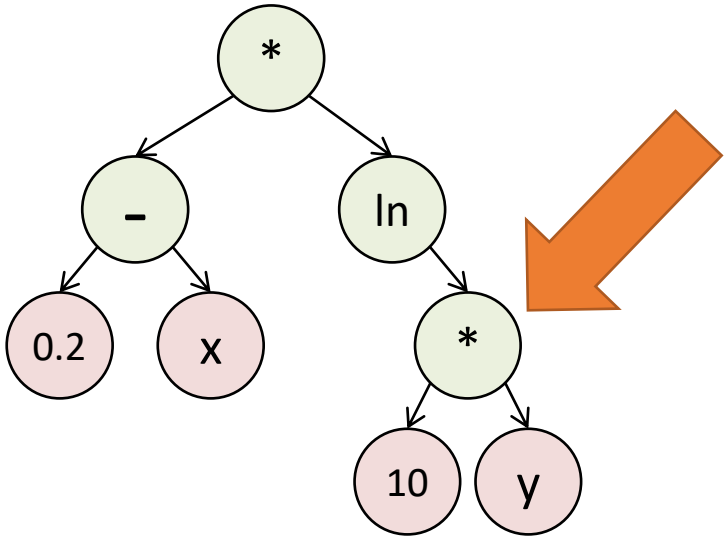
Operators: +, -, *, /, ln…

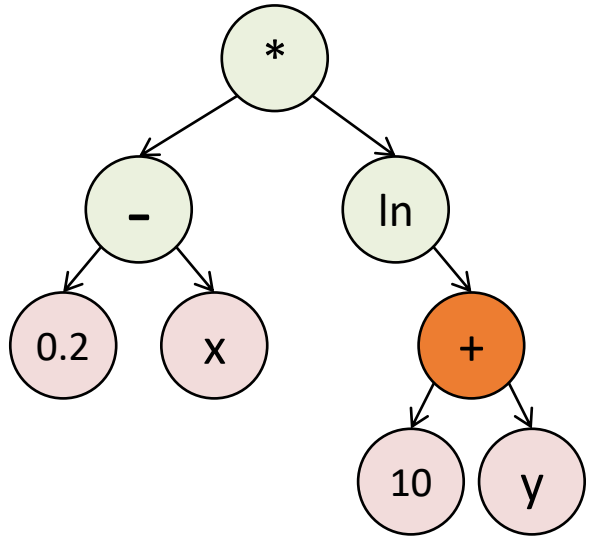
Terminals: reals, integers, variables

$$f(x, y) = (0.2 - x) * \ln(10 * y)$$

Symbolic regression

- **Variators** to go from a candidate equation to another



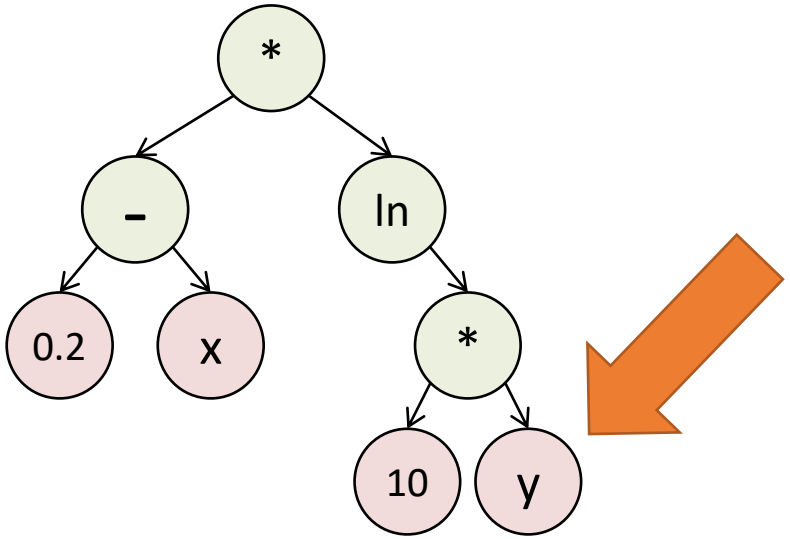


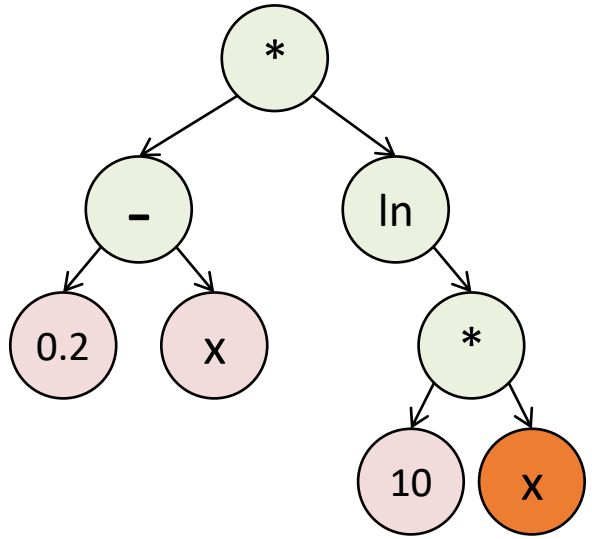
$$f(x,y) = (0.2 - x) * \ln(10 * y)$$

$$f(x,y) = (0.2 - x) * \ln(10 + y)$$

Symbolic regression

- **Variators** to go from a candidate equation to another



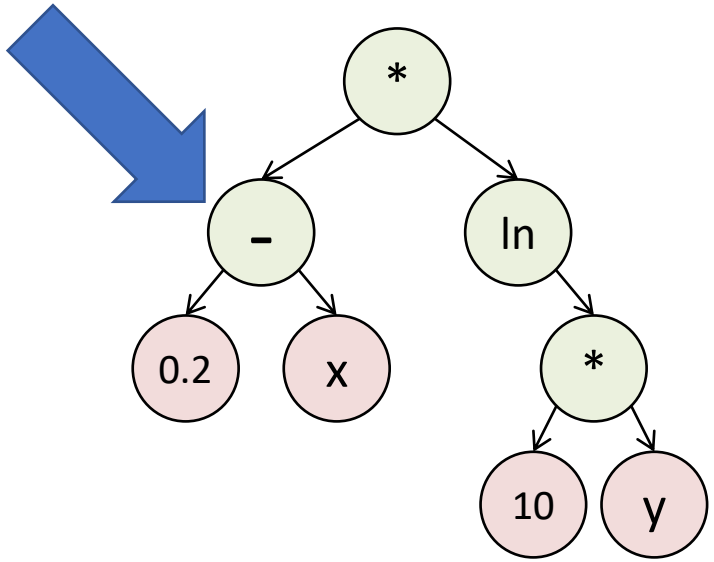


$$f(x,y) = (0.2 - x) * \ln(10 * y)$$

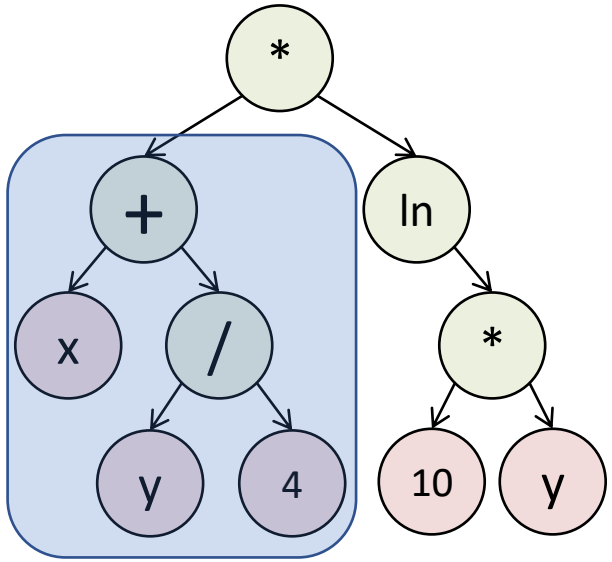
$$f(x,y) = (0.2 - x) * \ln(10 + x)$$

Symbolic regression

- **Variators** to go from a candidate equation to another



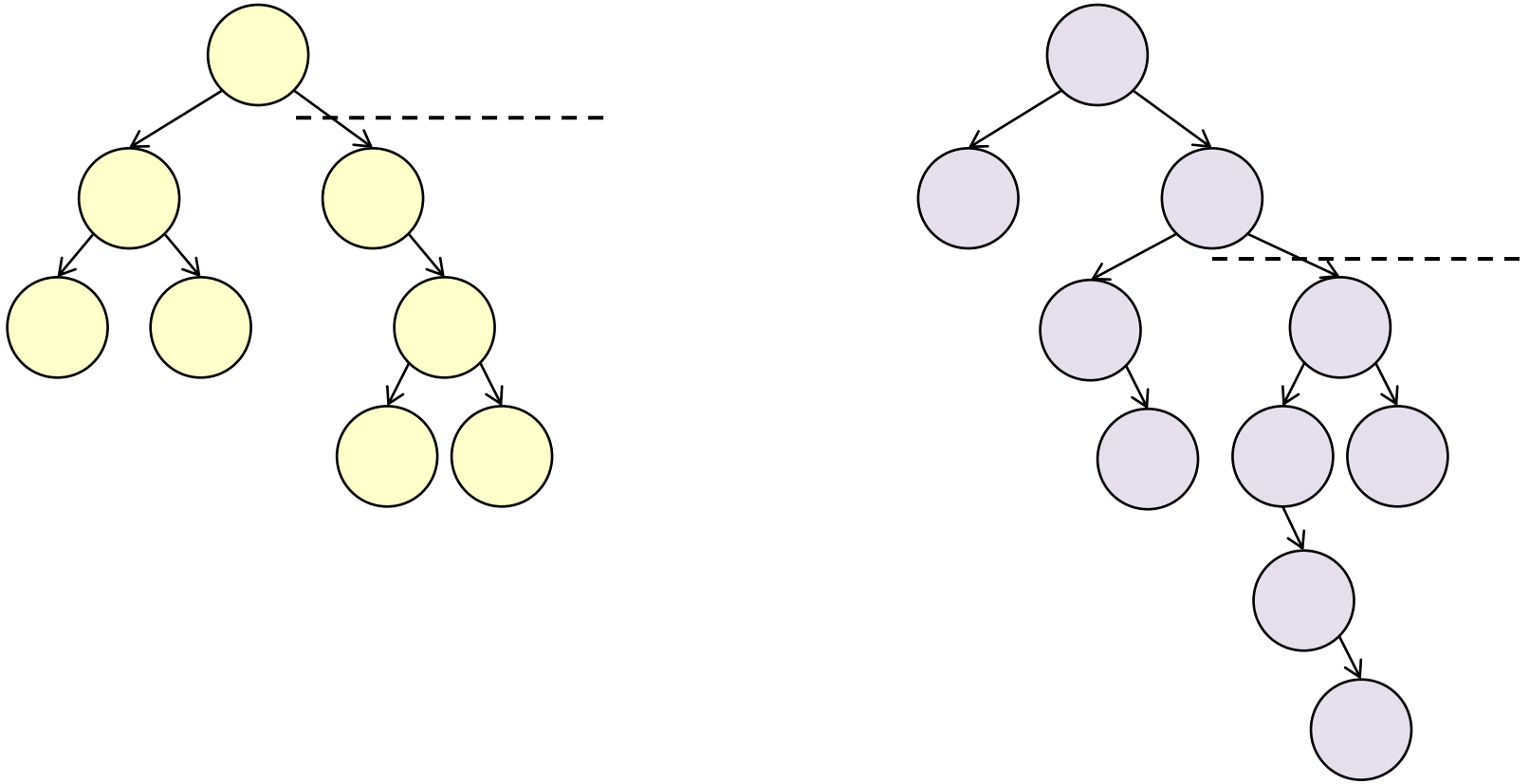
$$f(x,y) = (0.2 - x) * \ln(10 * y)$$



$$f(x,y) = (x + (y/4)) * \ln(10 + x)$$

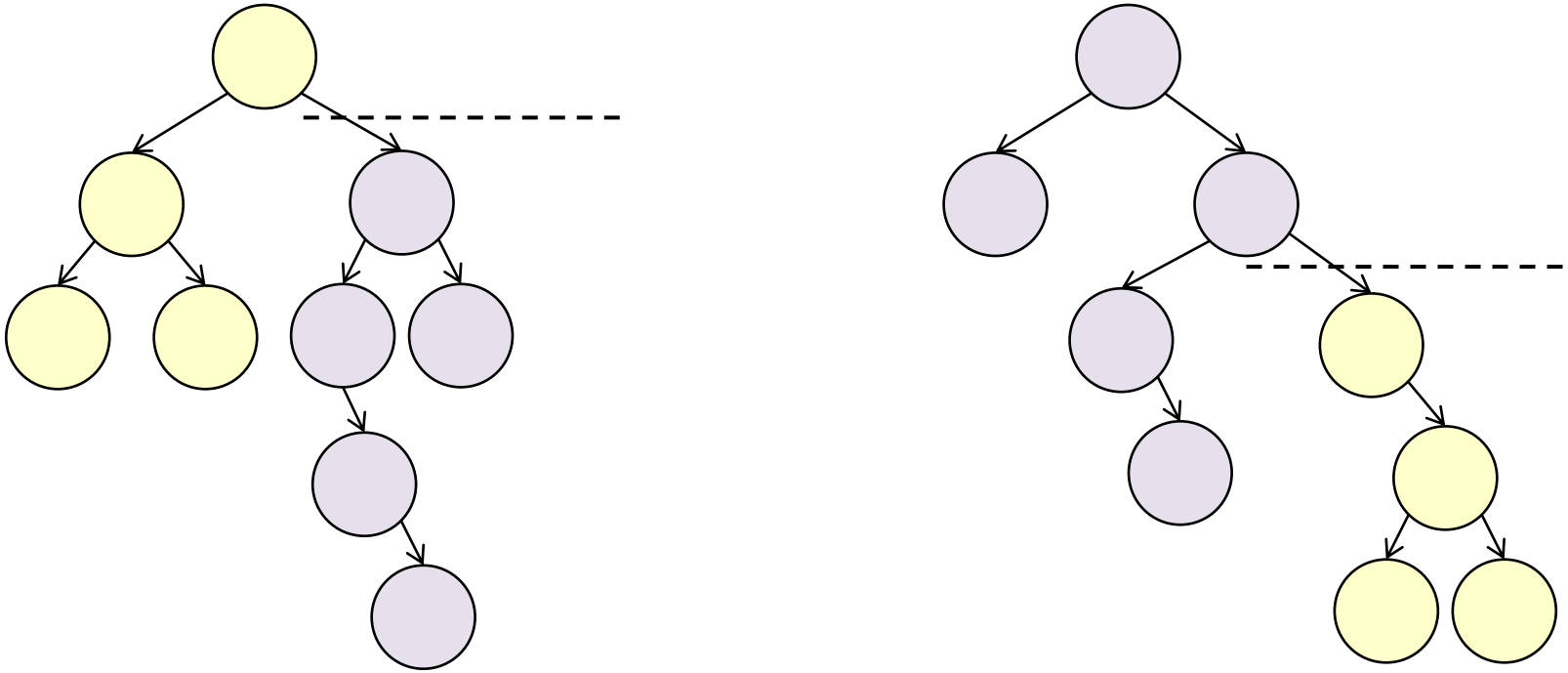
Symbolic regression

- **Variators** to go from a candidate equation to another



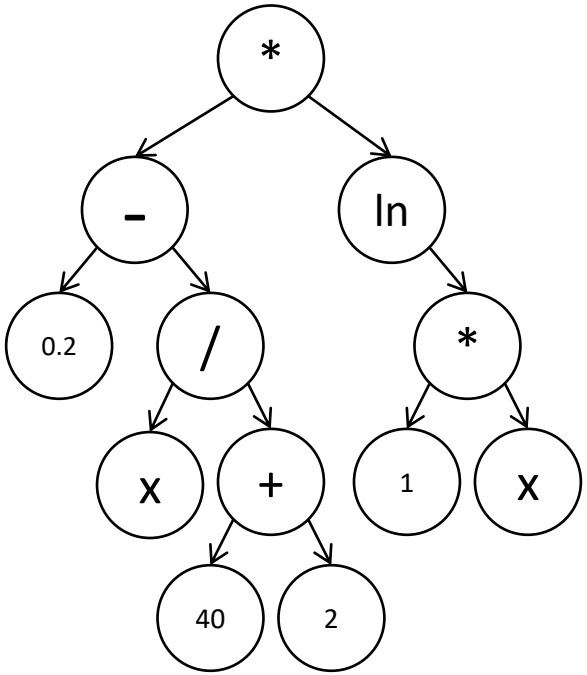
Symbolic regression

- **Variators** to go from a candidate equation to another

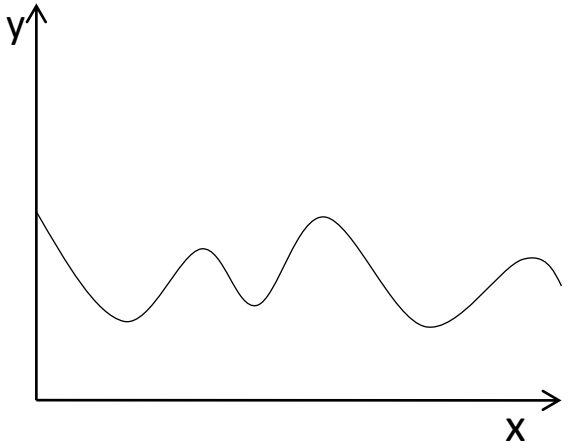


Symbolic regression

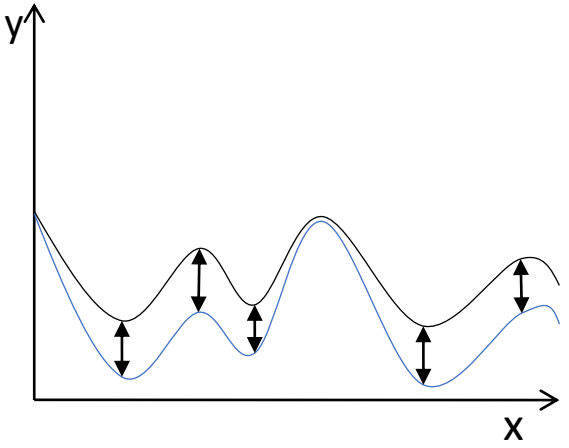
- **Evaluator** to assess quality of a solution



$$f(x) = \left(0.2 - \frac{x}{42}\right) * \log(x)$$



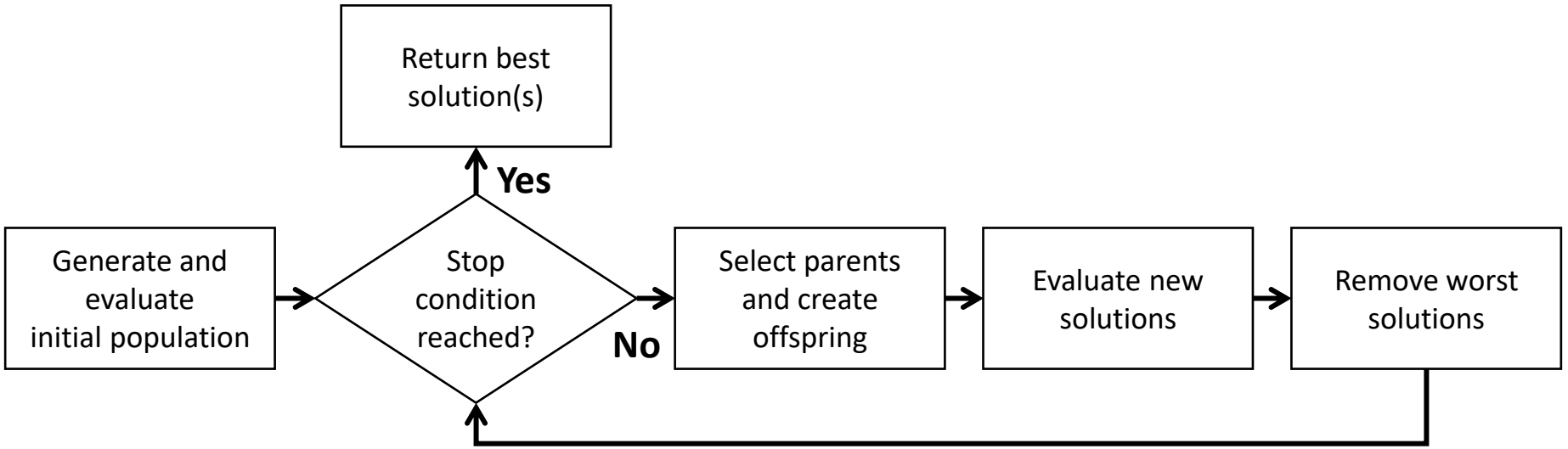
$$E(f) = \sum_{i=0}^{N} |f(x_i) - g(x_i)|$$



Error (to be minimized)

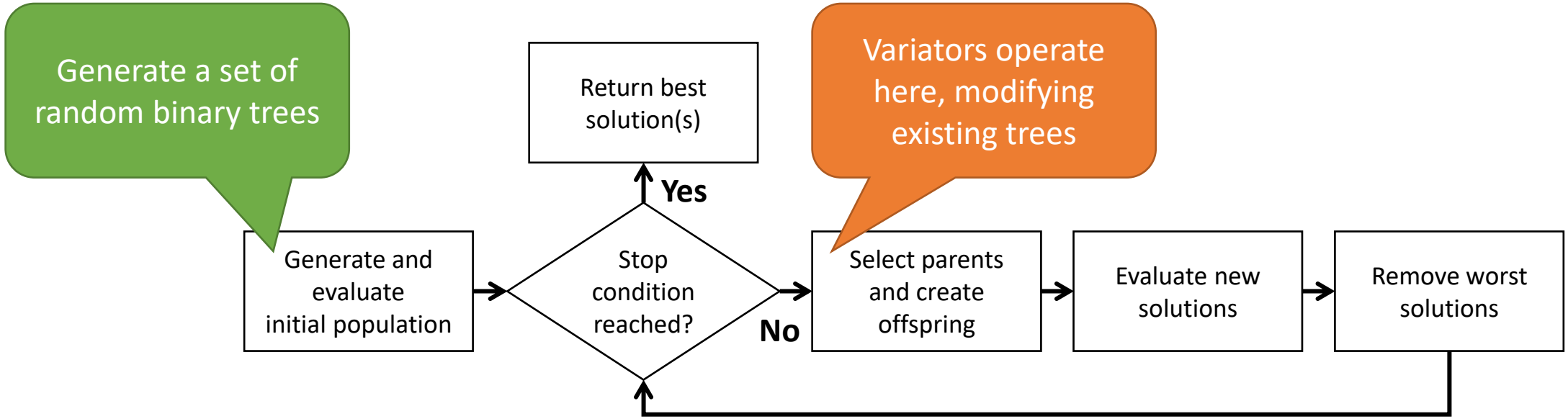
Symbolic regression

- Search space of all possible equations is **too vast**
  - Impossible to explore exhaustively
  - Solution: **stochastic optimization** (evolutionary algorithms)



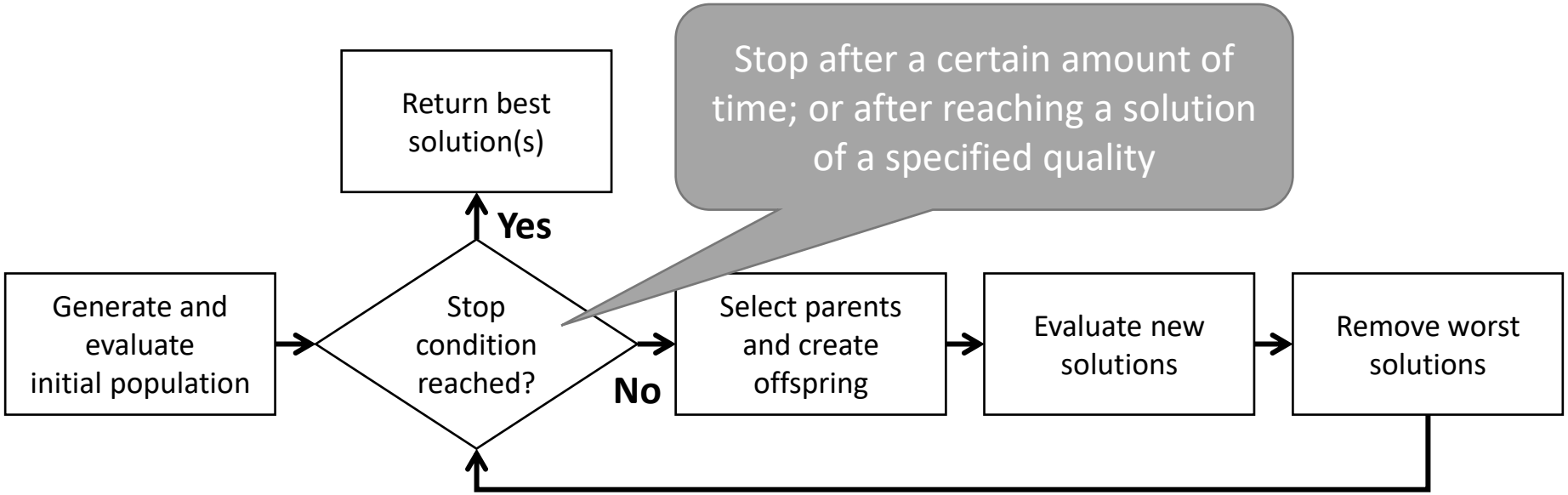
Symbolic regression

- Search space of all possible equations is **too vast**
  - Impossible to explore exhaustively
  - Solution: **stochastic optimization** (evolutionary algorithms)



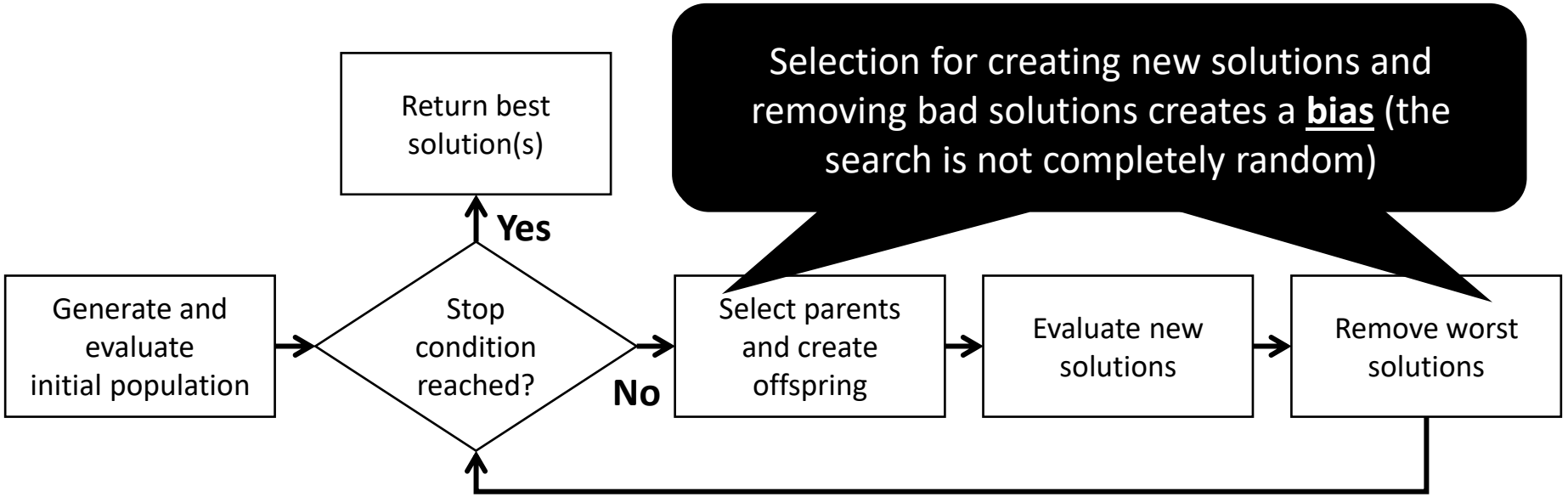
Symbolic regression

- Search space of all possible equations is **too vast**
  - Impossible to explore exhaustively
  - Solution: **stochastic optimization** (evolutionary algorithms)



Symbolic regression

- Search space of all possible equations is **too vast**
  - Impossible to explore exhaustively
  - Solution: **stochastic optimization** (evolutionary algorithms)



Symbolic regression

Symbolic regression for viability theory

- Evolve (effective) explicit control laws
  - Candidate solution is a set of SR trees encoding equations
  - **One tree for each control law** in the viability problem
  - Classic SR encoding and variators

$$u(x_1, \ldots, x_n) \begin{cases} u_1 = G_1(x_1, \ldots, x_n) \\ u_2 = G_2(x_1, \ldots, x_n) \\ \ldots \\ u_n = G_n(x_1, \ldots, x_n) \end{cases}$$

Symbolic regression for viability theory

- Evaluation (noisy/stochastic)
  - Randomly draw a set of initial conditions (**i.c.**) inside constraints
  - Evaluate number of trajectories that stay inside constraints
  - Even trajectories that go outside, count number of steps inside
  - Candidate solutions in same iteration evaluated on same set of i.c.

$$F(I) = \frac{\sum_{i=1}^{n_{ic}} r(i)}{n_{ic}}$$

$$r(i) = \begin{cases} 1.0 & \text{if trajectory for initial conditions } i \text{ is viable} \\ \frac{v_i}{t_{max}/\Delta_t} & \text{otherwise} \end{cases}$$

Symbolic regression for viability theory

- Code developed in Python
  - **gplearn** for SR classes and operators, **inspyred** for the basic evolutionary loop, **scipy** for solvers of ordinary differential equations, **sympy** for symbolic computation
- Hyperparameters of the evolutionary algorithm
  - $\mu = 100,\ \lambda = 100, \tau = 2, (\mu + \lambda)$, max evaluations = 10,000
  - Function set: $\{+, -, *, /, \log, \sqrt{}, \sin, \cos\}$
  - Initialization: *half and half*, $2 \leq d \leq 4$
  - $p_c = 0.4, p_h = 0.1, p_s = 0.1, p_p = 0.1, p_i = 0.3$
  - $n_{ic} = 100, \Delta_t = 0.1, t_{max} = 100$

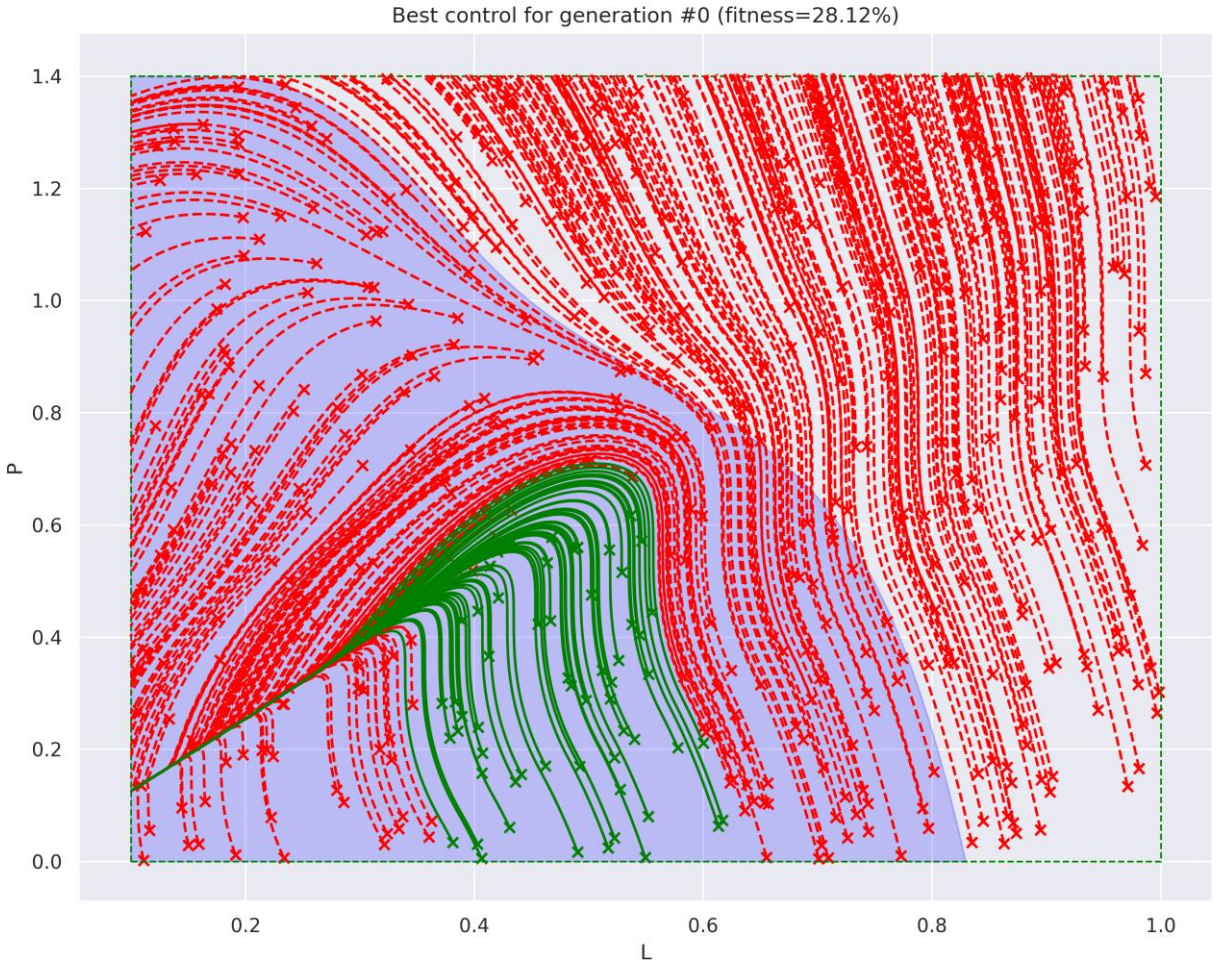
Experiment 1: lake eutrophication

- Equations describing the eutrophication of a lake
  - $L(t)$, input of phosphorus
  - $P(t)$, concentration of phosphorus
  - $u(L, P)$, control law, variation in the input of phosphorus
- For this benchmark, the **viability kernel** is known

$$\begin{cases} L(t+1) = L(t) + u \\ P(t+1) = P(t) - b \cdot P(t) + L(t) + r \cdot \frac{P(t)^q}{P(t)^q + m^q} \end{cases} \quad \text{with } u \in [u_{min}, u_{max}]$$

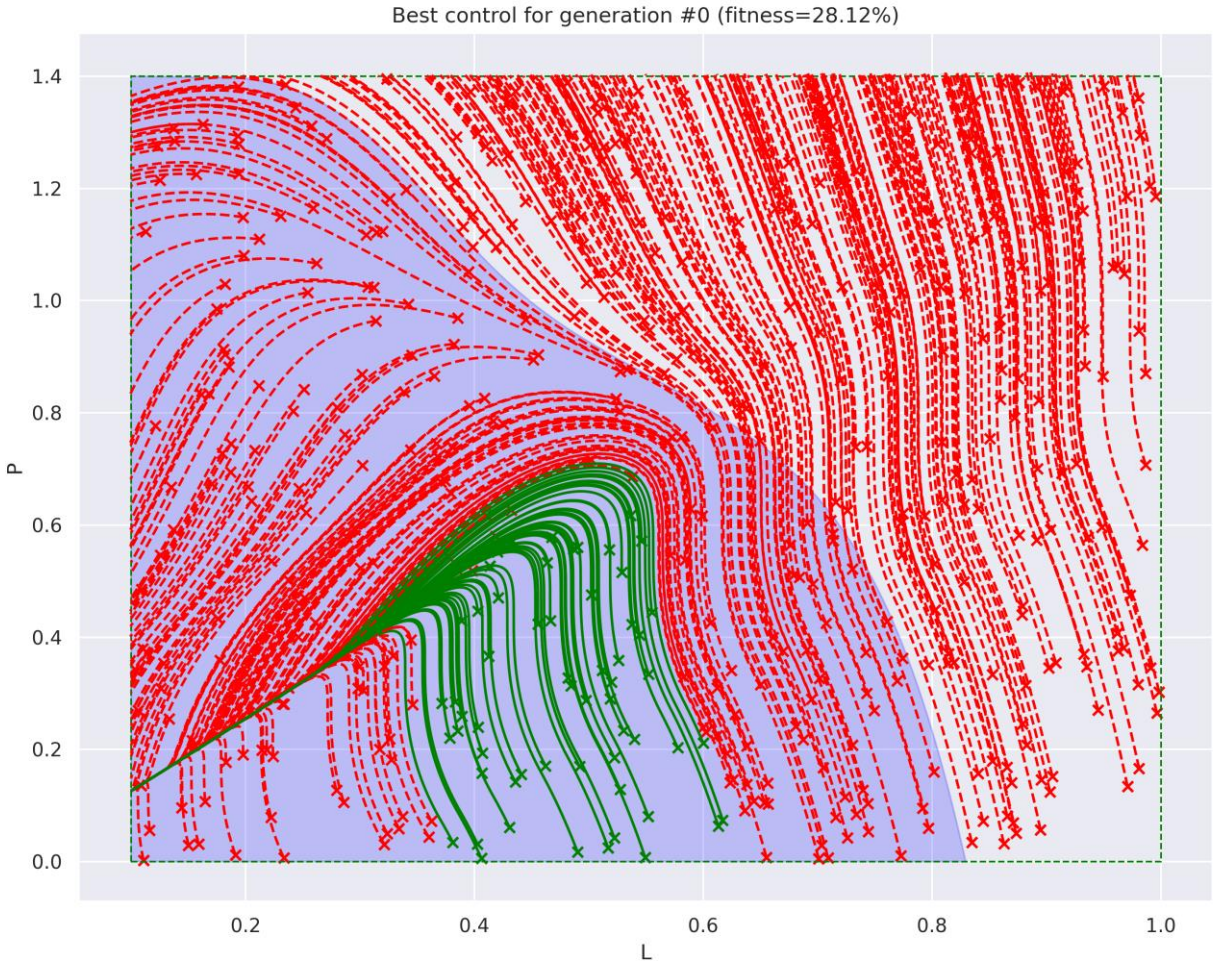
$$u(L,P) = \begin{cases} u_{min} & \text{if } u < u_{min} \\ G(L,P) & \text{if } u_{min} \leq u \leq u_{max} \\ u_{max} & \text{if } u > u_{max} \end{cases}$$

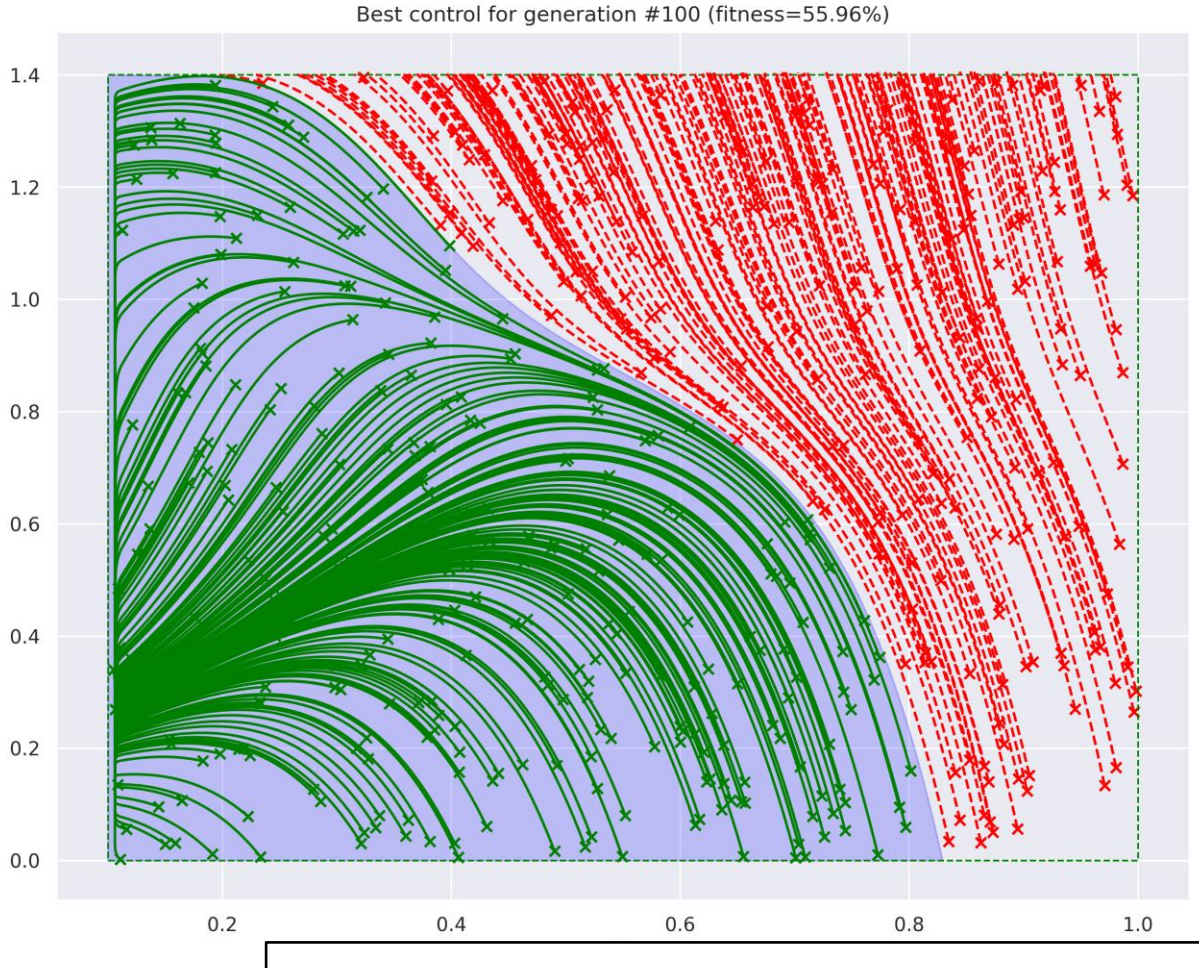
Experiment 1: lake eutrophication



Alberto TONDA, Team EKINOCS, UMR 518 MIA-PS, INRAE, Université Paris-Saclay
alberto.tonda@inrae.fr

Experiment 1: lake eutrophication





$$G(L,P) = \sin(\log(L)) - (\log(L) - 1.4617)$$

Alberto TONDA, Team EKINOCS, UMR 518 MIA-PS, INRAE, Université Paris-Saclay
alberto.tonda@inrae.fr

Experiment 2: 3D sphere

- Equations describing a 3D sphere
  - **Viability kernel** known (the sphere's volume)
  - **Encoding**: three SR trees describing equations

$$\begin{cases} x(t+1) = x(t) + a \cdot u_x \\ y(t+1) = y(t) + a \cdot u_y \\ z(t+1) = z(t) + a \cdot u_z \end{cases}$$

$$\text{with } u_x^2 + u_y^2 + u_z^2 \leq 1$$

$$x^2 + y^2 + z^2 \leq r$$

Experiment 2: 3D sphere

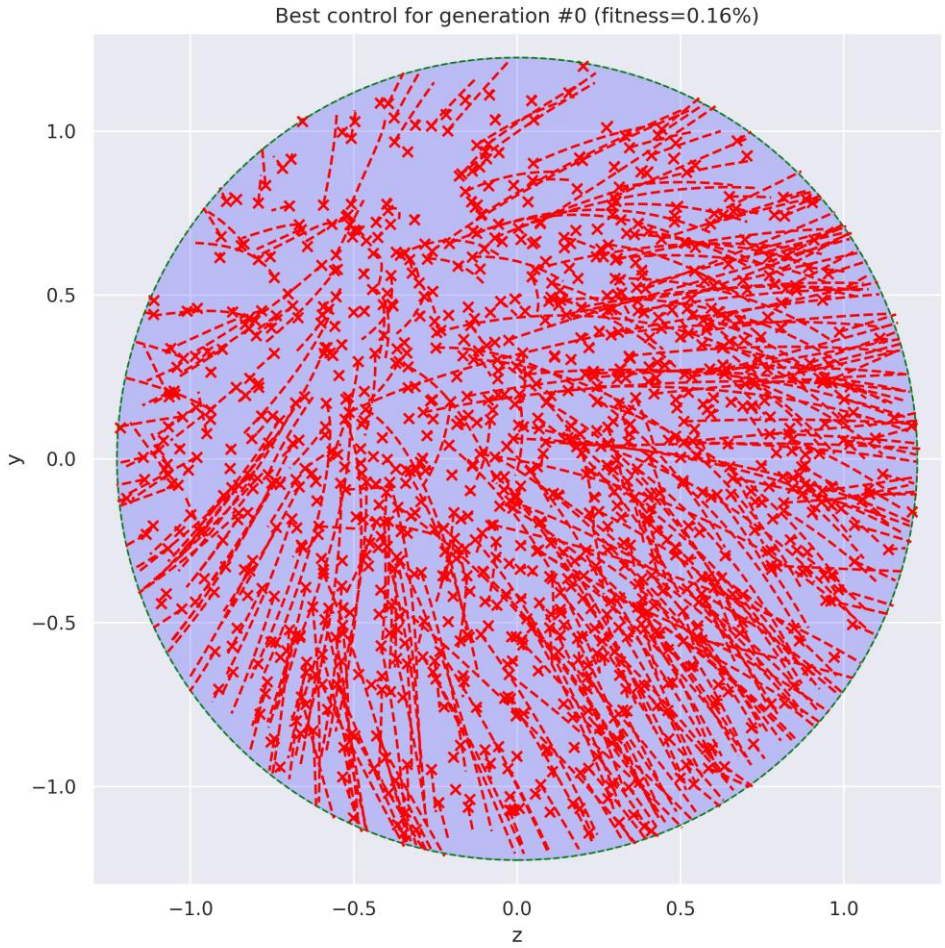
- Equations describing a 3D sphere
  - **Viability kernel** known (the sphere's volume)
  - **Encoding**: three SR trees describing equations

$$\begin{cases} x(t+1) = x(t) + a \cdot u_x \\ y(t+1) = y(t) + a \cdot u_y \\ z(t+1) = z(t) + a \cdot u_z \end{cases}$$

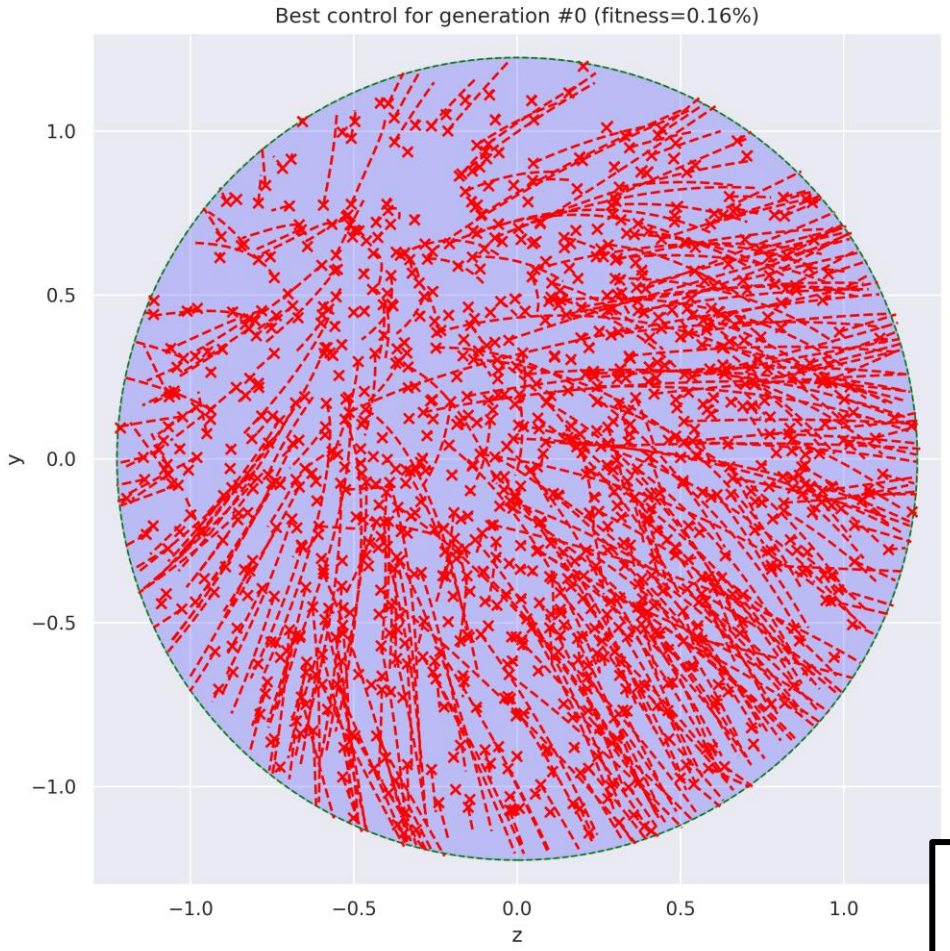
$$\text{with } u_x^2 + u_y^2 + u_z^2 \leq 1$$

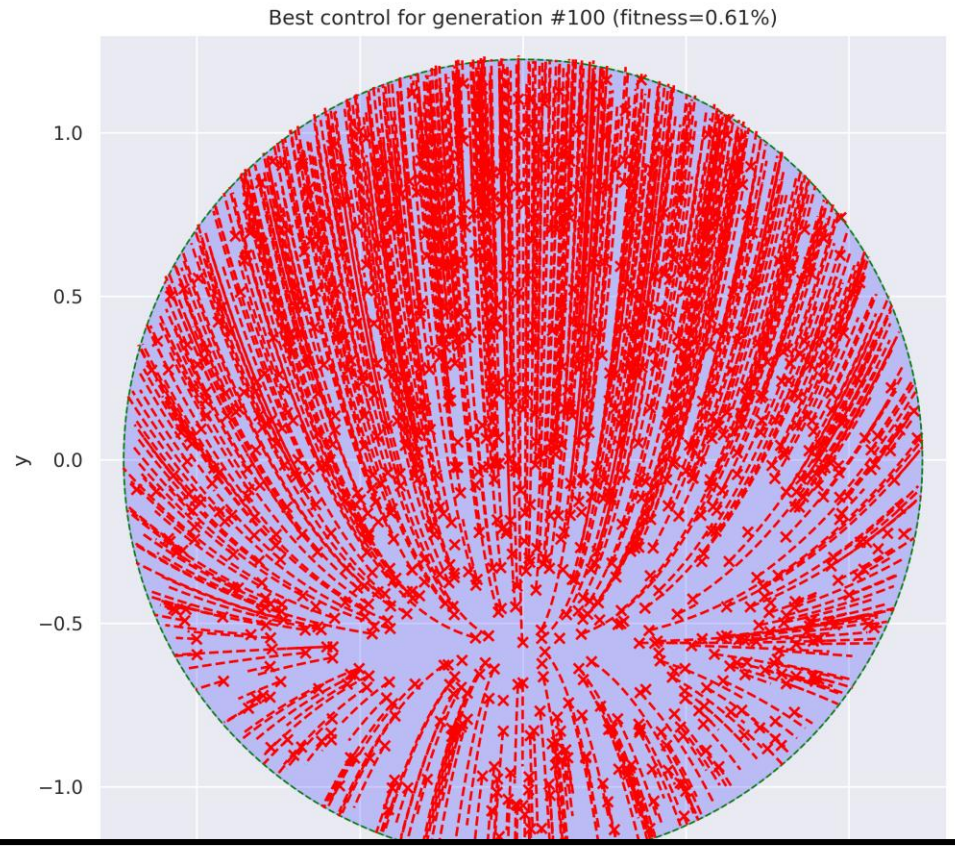
$$x^2 + y^2 + z^2 \leq r$$

Experiment 2: 3D sphere



Experiment 2: 3D sphere





$$\begin{cases} u_x(x,y,z) = y \cdot \dfrac{-0.9655 \cdot x}{y} \\ u_y(x,y,z) = \cos(\cos((y + \log(\cos(-0.4505 \cdot z))) \cdot 0.3767)) \\ u_z(x,y,z) = z \cdot \sin(-0.6640) \end{cases}$$

Experiment 2: 3D sphere

- Hypothesis: formulation of the problem matters
  - Same problem, in polar coordinates
  - Constraints are easier to manage for an EA

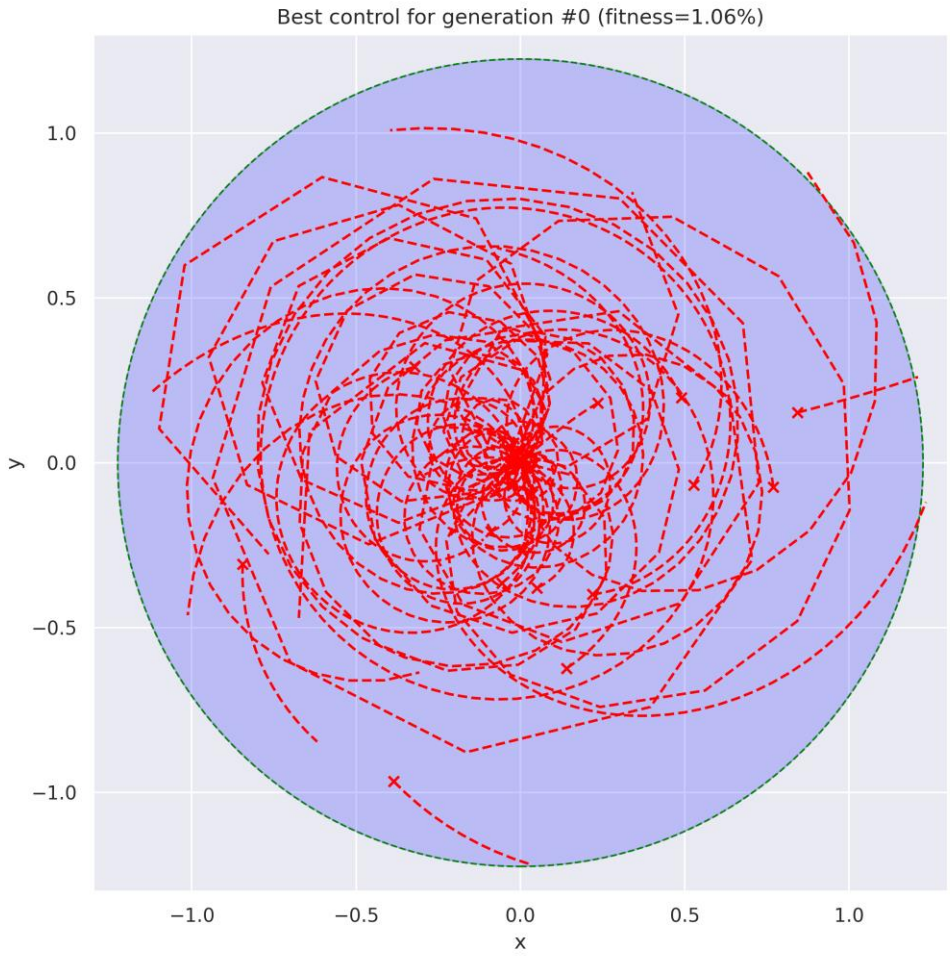
$$\begin{cases} x = \rho \cdot \sin(\theta) \cdot \cos(\phi) \\ y = \rho \cdot \sin(\theta) \cdot \sin(\phi) \\ z = \rho \cdot \cos(\theta) \end{cases}$$

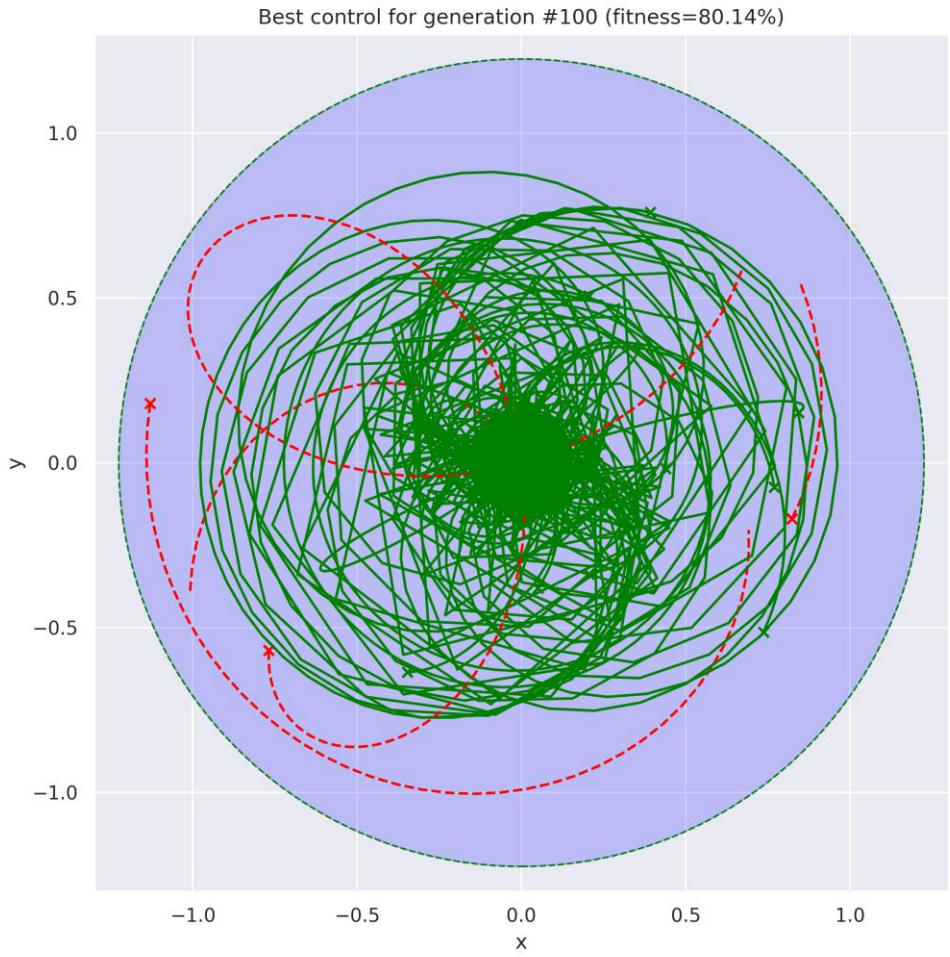
$$\begin{cases} \rho(t+1) = \rho(t) + a \cdot u_\rho \\ \theta(t+1) = \theta(t) + a \cdot u_\theta \\ \phi(t+1) = \phi(t) + a \cdot u_\phi \end{cases}$$

$$\begin{gathered} 0 \leq \rho \leq \sqrt{r} \\ 0 \leq \theta \leq \pi \\ 0 \leq \phi \leq 2\pi \end{gathered}$$

$$U(\rho, \theta, \phi) = \begin{cases} u_\rho(\rho, \theta, \phi) = \min(\max(G_\rho(\rho, \theta, \phi), -1.0), 1.0) \\ u_\theta(\rho, \theta, \phi) = \min(\max(G_\theta(\rho, \theta, \phi), 0.0), \pi) \\ u_\phi(\rho, \theta, \phi) = \min(\max(G_\phi(\rho, \theta, \phi), 0.0), 2\pi) \end{cases}$$

Experiment 2b: 3D sphere (polar coordinates)





Conclusions

- Proposed approach seems to *partially* work
  - **Evaluation function** can likely be **improved**
  - **Formulation** of the problem **matters** for efficacy
  - Initialization: **default values** for constants (constraint values)

- Limitations
  - Computing the viability kernel **guarantees viability** “forever”
  - Our approach only **guarantees viability up to time** t_{max}
  - However, **computing kernel is unfeasible** for real-world cases

Future works

- Viability theory offers **promising leads** for EAs
- Interpretable control laws
  - Explicit control law equations easier to understand
  - Drive/start evolutionary search towards/from existing strategies?
- *Tyches* in viability theory are non-controllable factors
  - Co-evolution to optimize *tyches* (worst-case scenarios)
  - Competing against control laws
- Computing (approximate) viability kernels

Thank you for your time! Questions?

https://github.com/albertotonda/evolutionary-viability-theory

alberto.tonda@inrae.fr