Can we extrapolate?

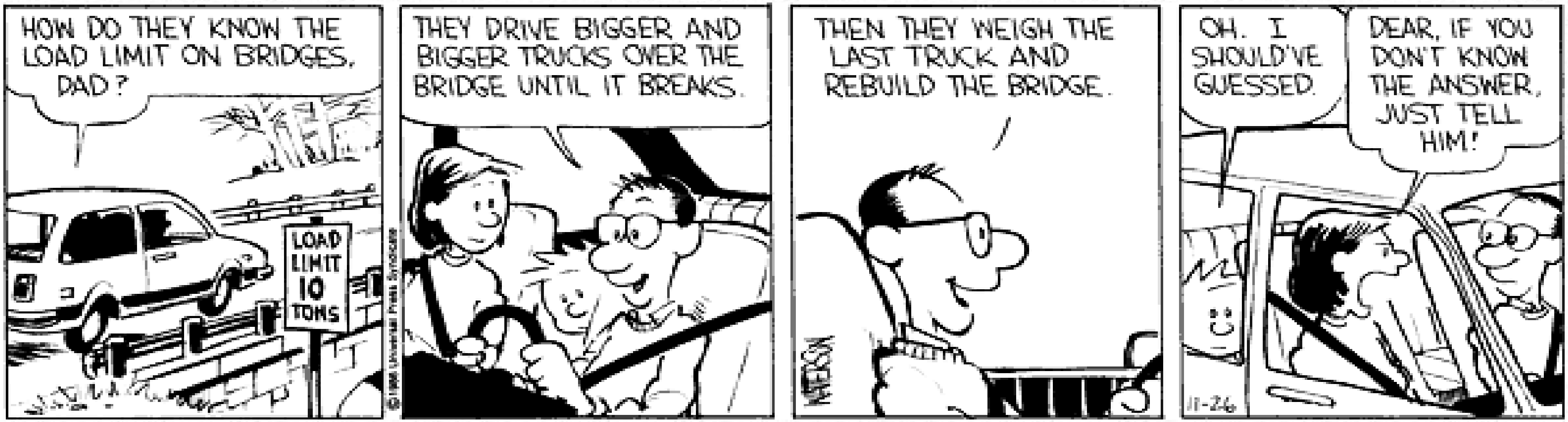
Equation identification for Extrapolation and Control

by Georg Martius

MPI for Intelligent Systems, Tübingen
Autonomous Learning Group

MAX-PLANCK-GESELLSCHAFT

MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS TÜBINGEN

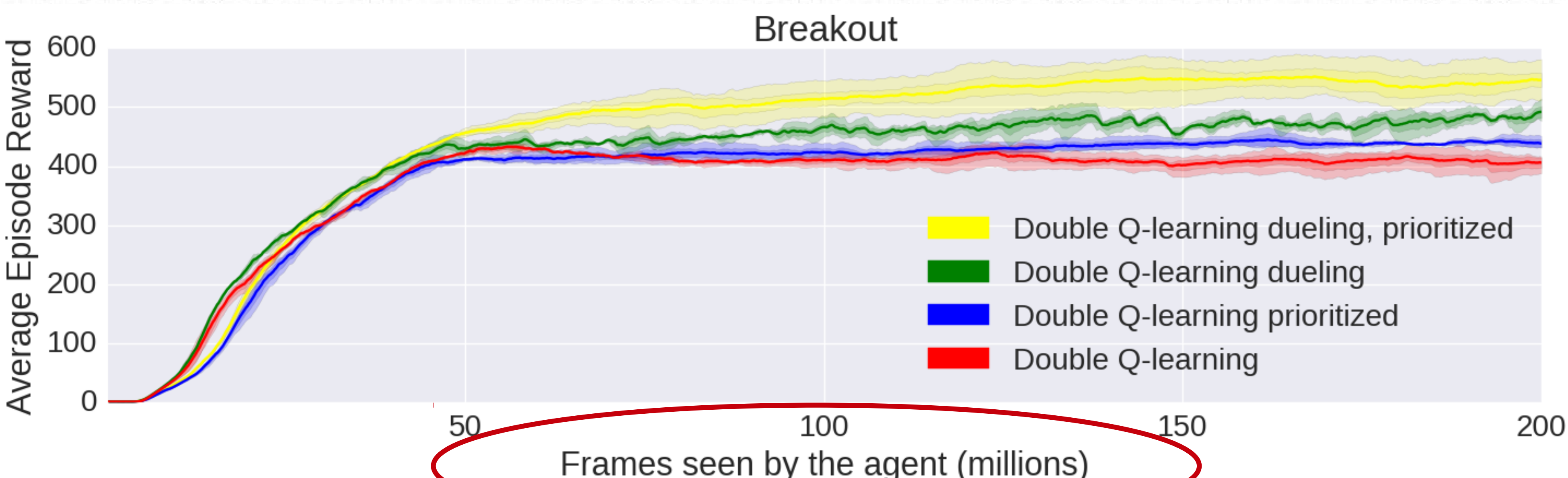


Watterson, B.: Calvin and Hobbes

Want:

- extrapolate to unseen domains
- by identify underlying equations (from observed data)
- use it to efficiently control a robot
- get interpretable models

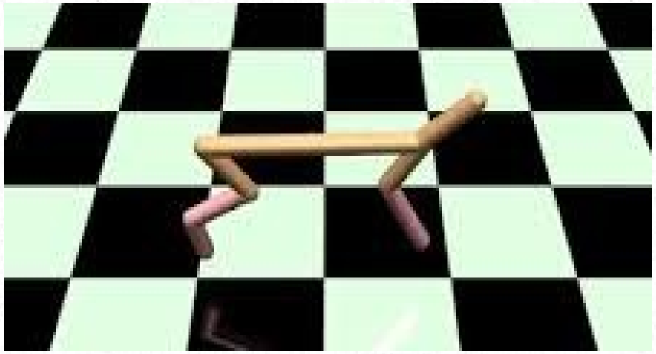
Reinforcement Learning today

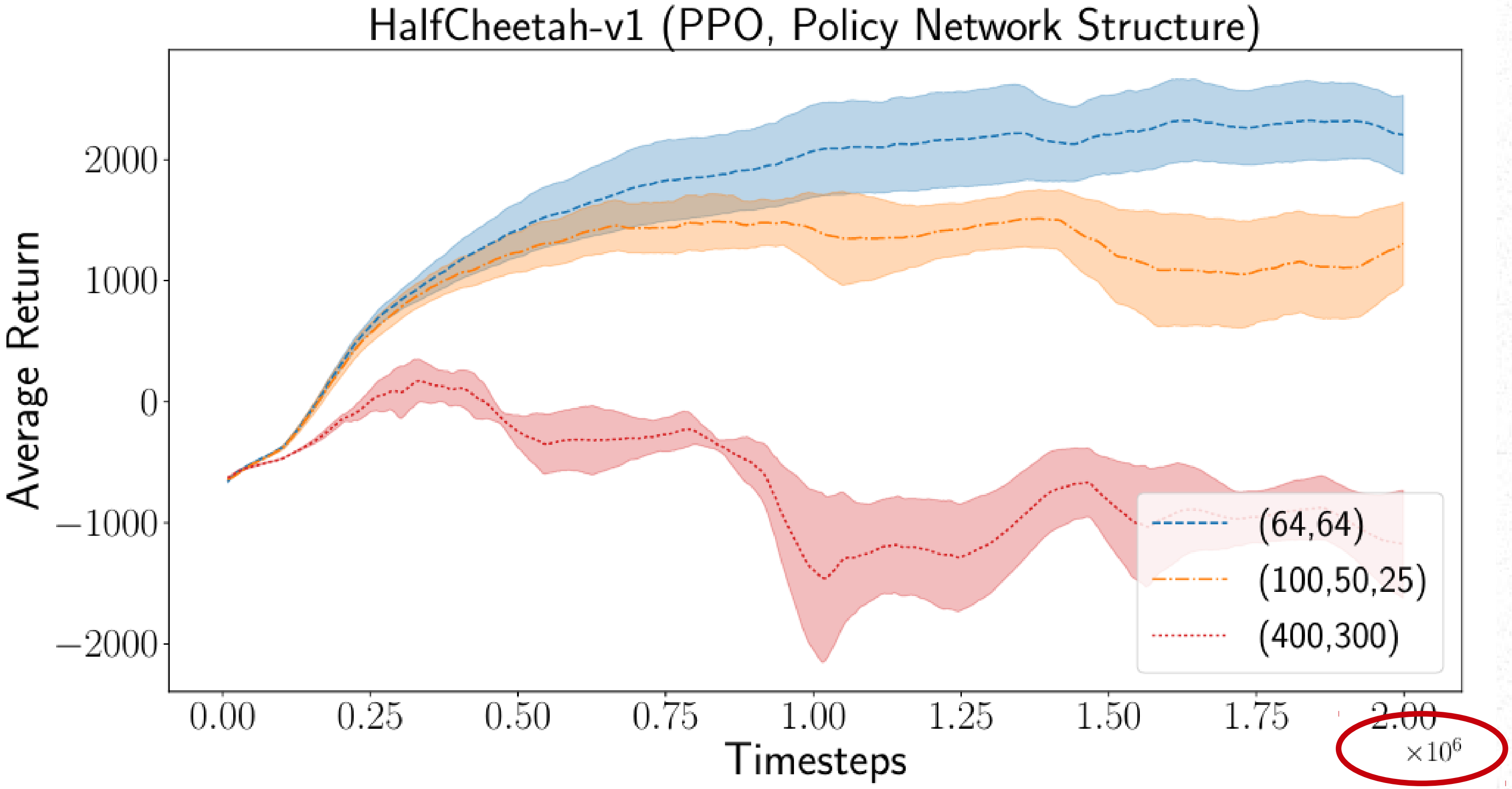


By SZYMON SIDOR & JOHN SCHULMAN
(openai.com)

Reinforcement Learning today

OpenAi Gym Half Cheetah environment

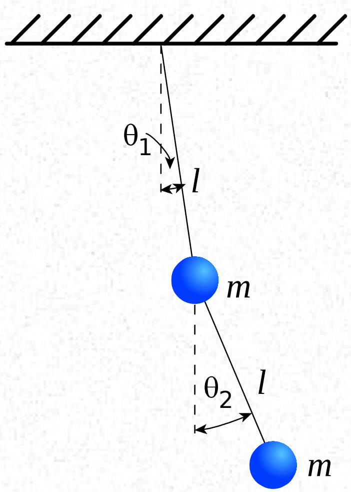


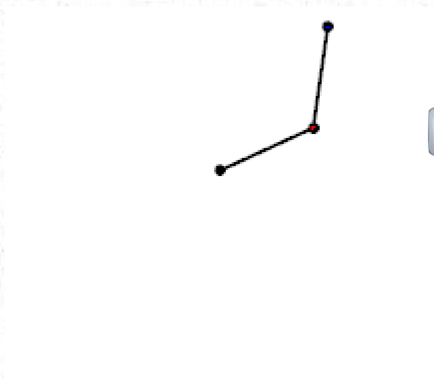


Henderson et al 2017

Learning the physics of the world

Example: Learn equation of double pendulum from interaction





$$\dot{\omega}_1 = \frac{g\sin(\theta_1 - 2\theta_2) + 3g\sin(\theta_1) + l\omega_1^2\sin(2\theta_1 - 2\theta_2) + \cdots}{2l(\cos(\theta_1 - \theta_2)^2 - 2)}$$

$$\dot{\omega}_2 = \frac{(-g\sin(2\theta_1 - \theta_2) + g\sin(\theta_2) - 2l\omega_1^2\sin(\theta_1 - \theta_2) - \cdots}{l(\cos(\theta_1 - \theta_2)^2 - 2)}$$

Want:

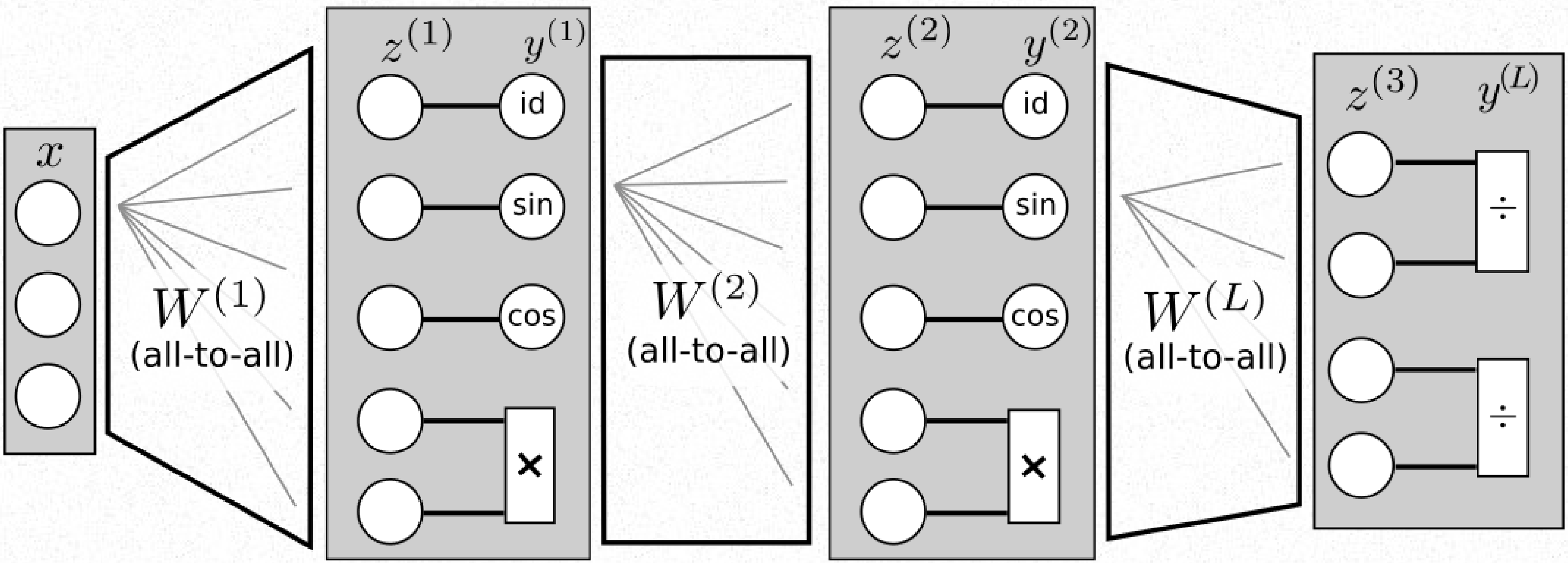
- identify underlying equations
- extrapolate to unseen domains

Martius, Lampert arXiv 2016

Differentiable Architecture for Equation Learning

Data: $\{(x_1, y_1), (x_2, y_2), \cdots\}$

Assumption: y = f(x) + noise   f is in the model class



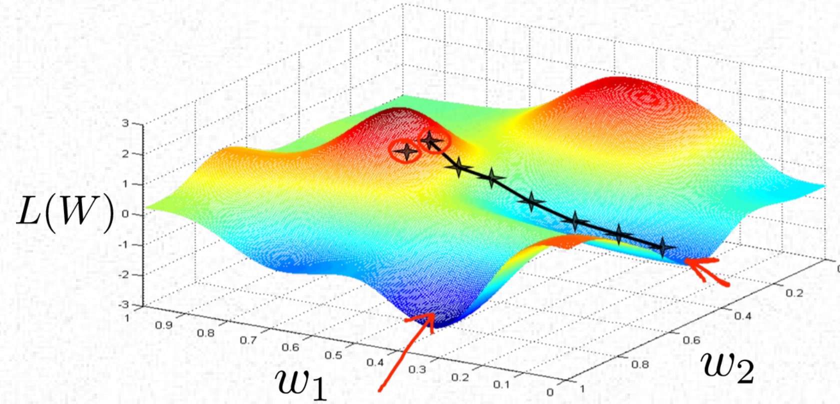
Replace standard units of NN

by id, sin, cos, multiplication and division.

Sahoo, Lampert, Martius, ICML 2018

Regression with sparsity regularization

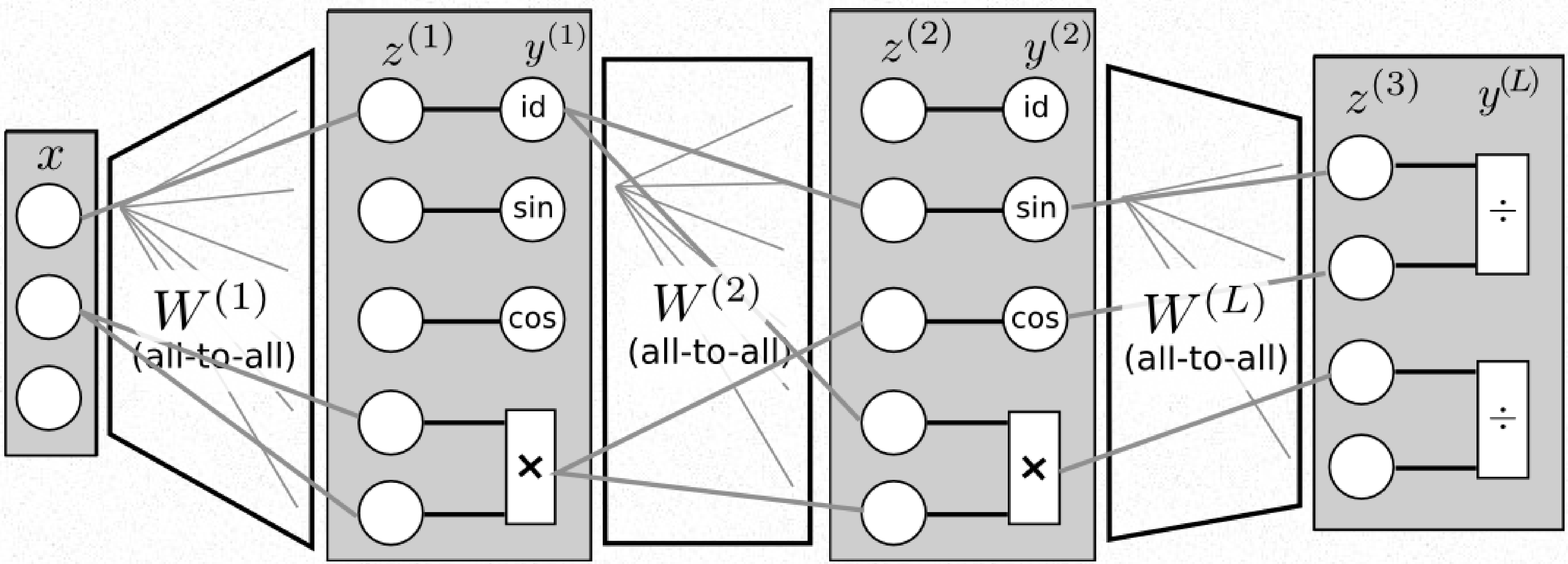
$$E = \sum_{i=1}^{n} |f(x_i, W) - y_i|^2 + \lambda |W|^1$$



Training by gradient descent

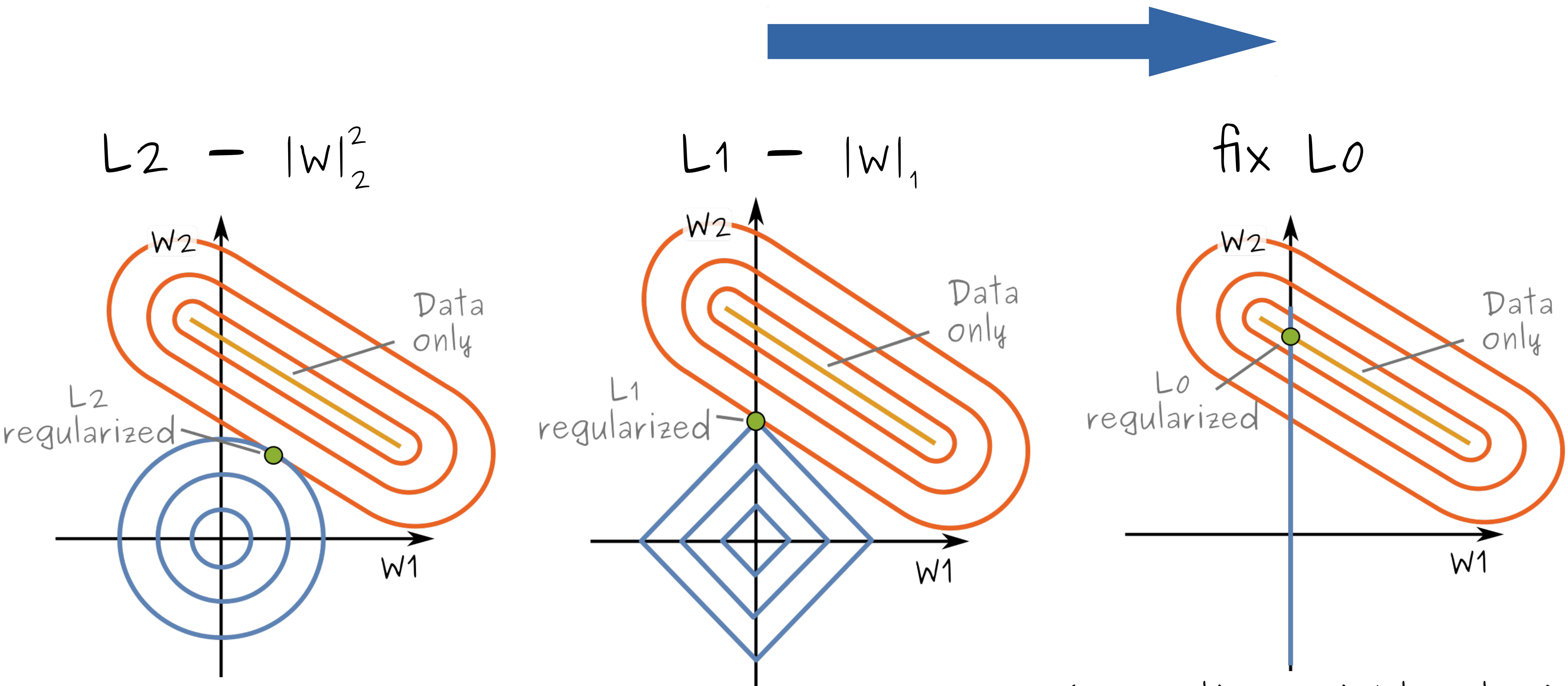
$$\Delta W \propto -\frac{\partial E}{\partial W}$$

if right formula is learned →
great extrapolation



Regularization Phases

- Want: sparse solution

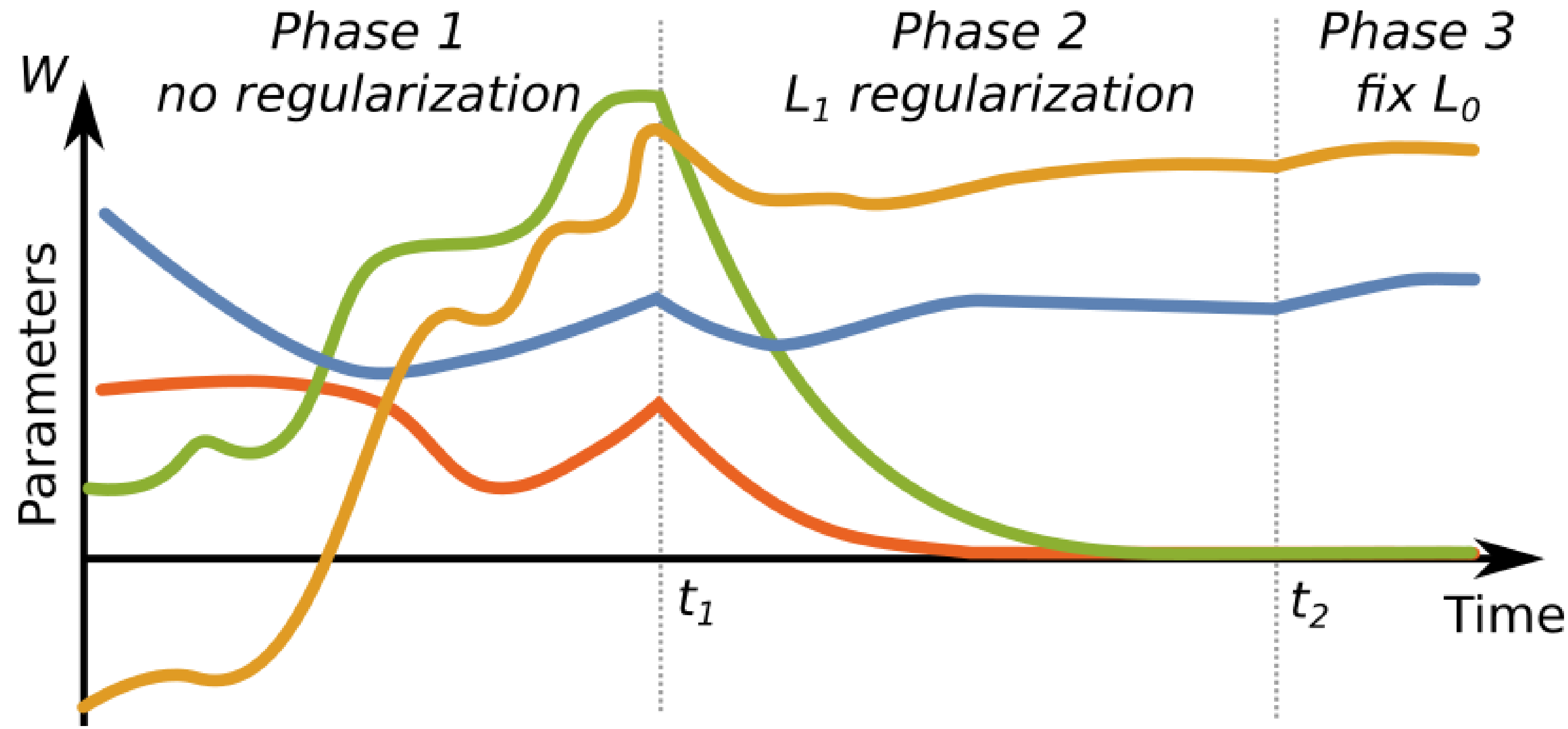


- (keep tiny weights at 0)

- sparse solution without tradeoff

Sahoo, Lampert, Martius, ICML 2018

Regularization Phases

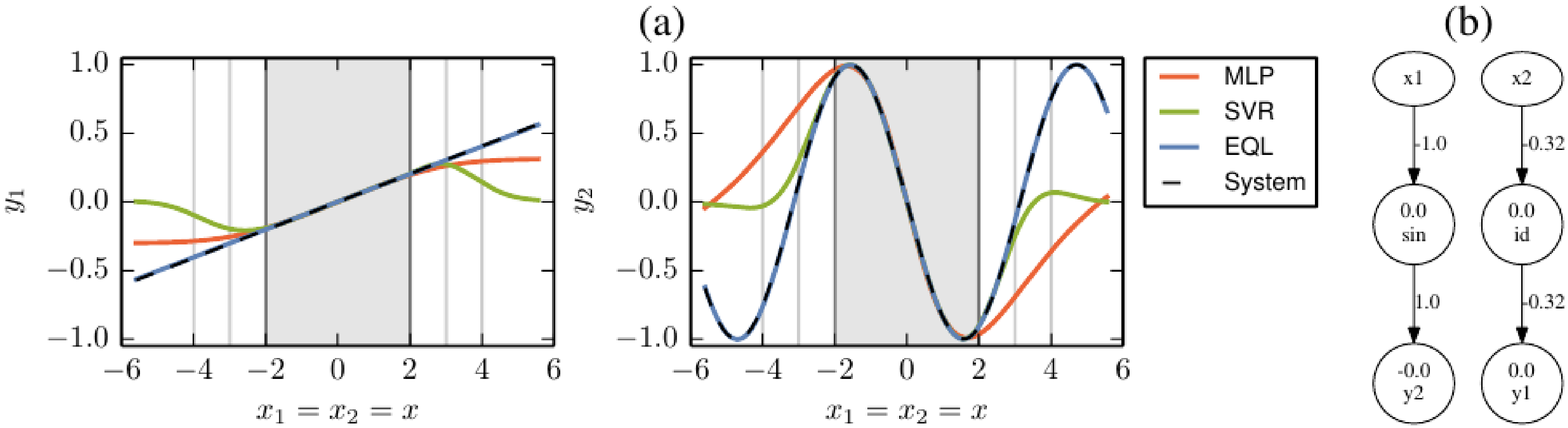


New ways to achieve sparsity: Bayesian compression/learned dropout
ICLR 2018, ArXiv: 1712.01312 by
Christos Louizos, Max Welling, Diederik P. Kingma

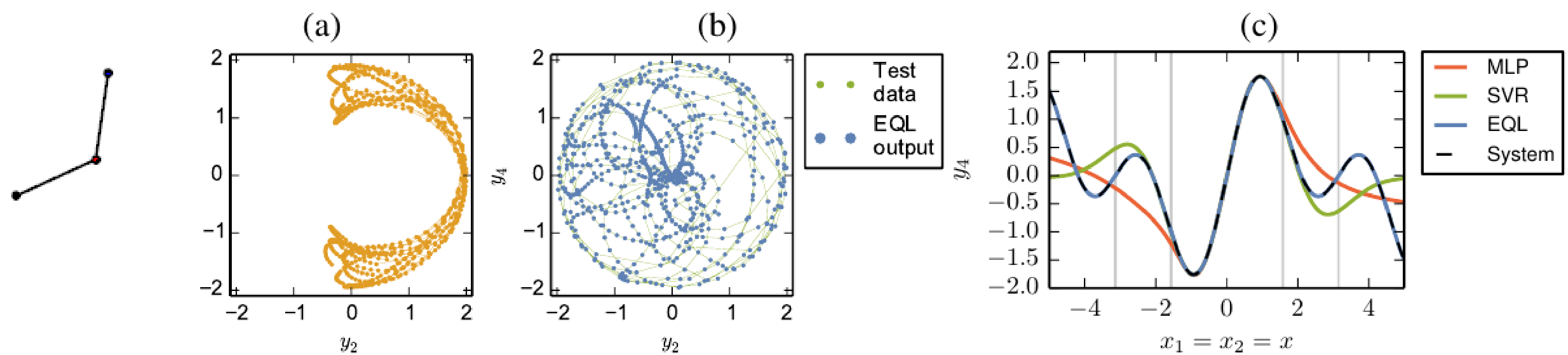
Martius, Lampert arXiv 2016

Pendulum Dynamics

$$\dot{x}_1 = x_2 \quad \text{and} \quad \dot{x}_2 = -g \sin(x_1)\,,$$



Double Pendulum Kinematics



(d)

	EQL	MLP	SVR
extrapolation error	0.0003 ± 0.00003	0.58 ± 0.03	0.25

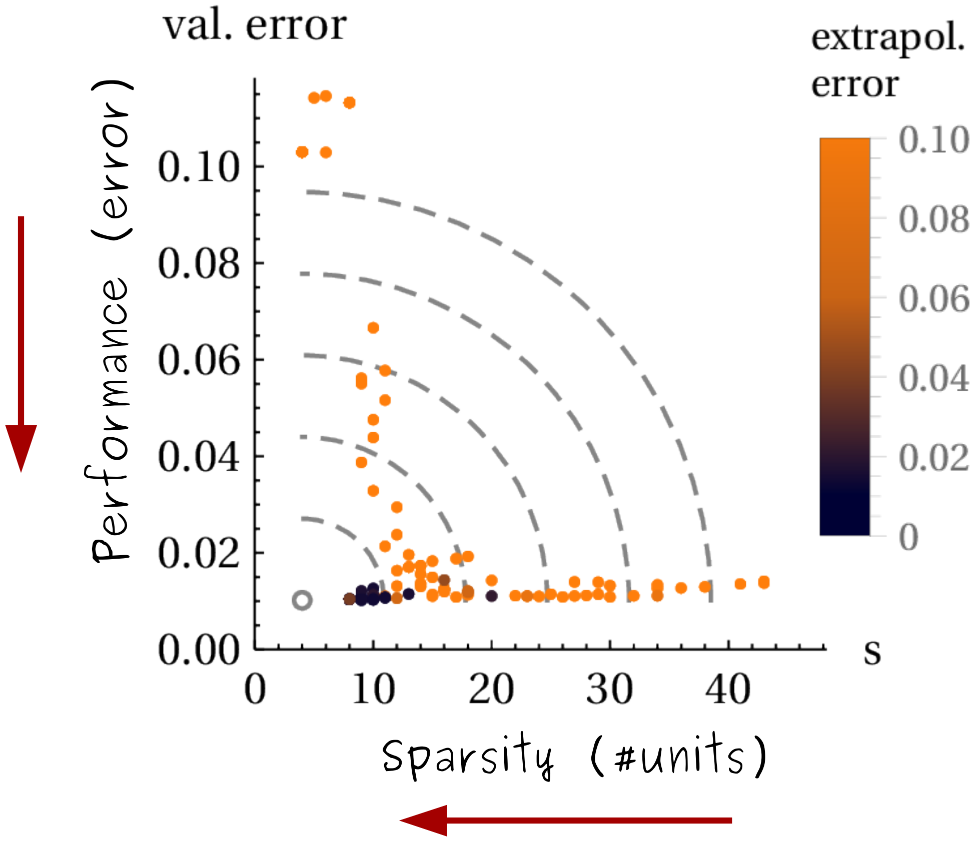
Martius, Lampert arXiv 2016

Model Selection

Occams Razor: Most simple formula is most likely the right one.

But too simple can also be wrong!

Multiobjective: Simple and good performance



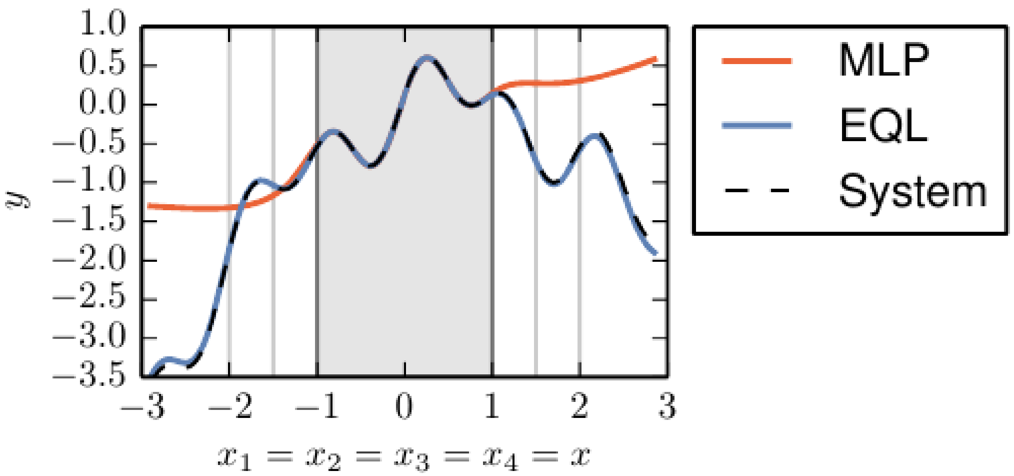
$$\arg\min_{\phi}[\tilde{v}(\phi)^2 + \tilde{s}(\phi)^2]$$

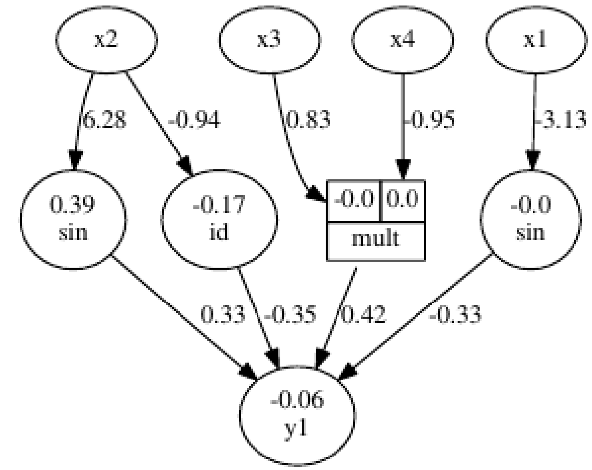
normalized values

Sahoo, Lampert, Martius, ICML 2018

some formulas

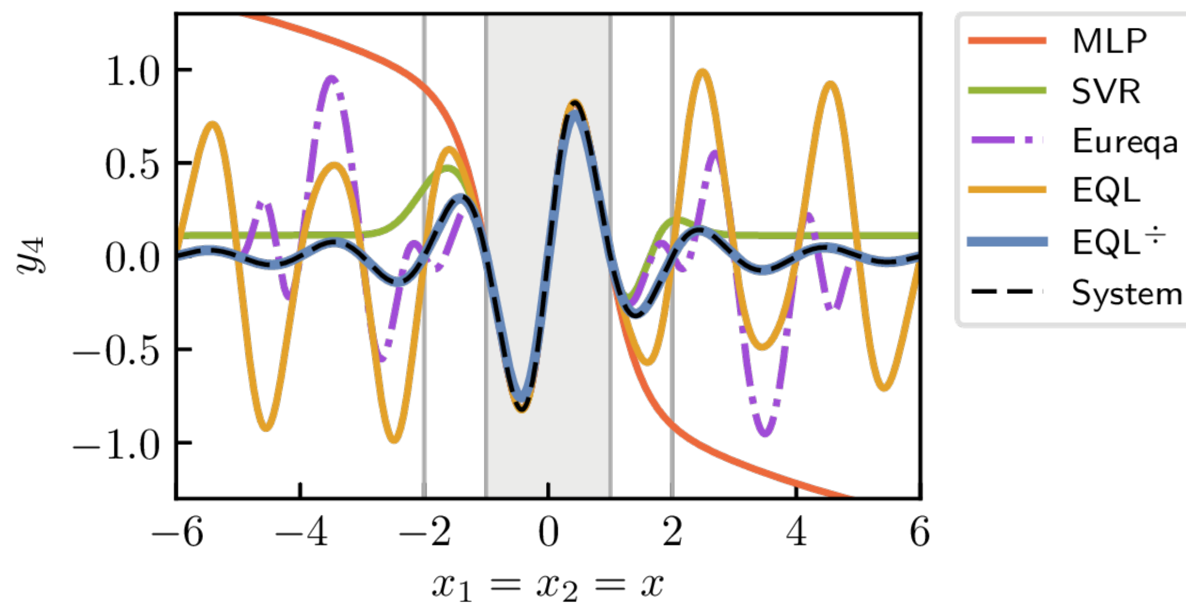
$$y = {}^{1}\!/_{3}\left(\sin(\pi x_1) + \sin\left(2\pi x_2 + {}^{\pi}\!/_{8}\right) + x_2 - x_3 x_4\right)$$





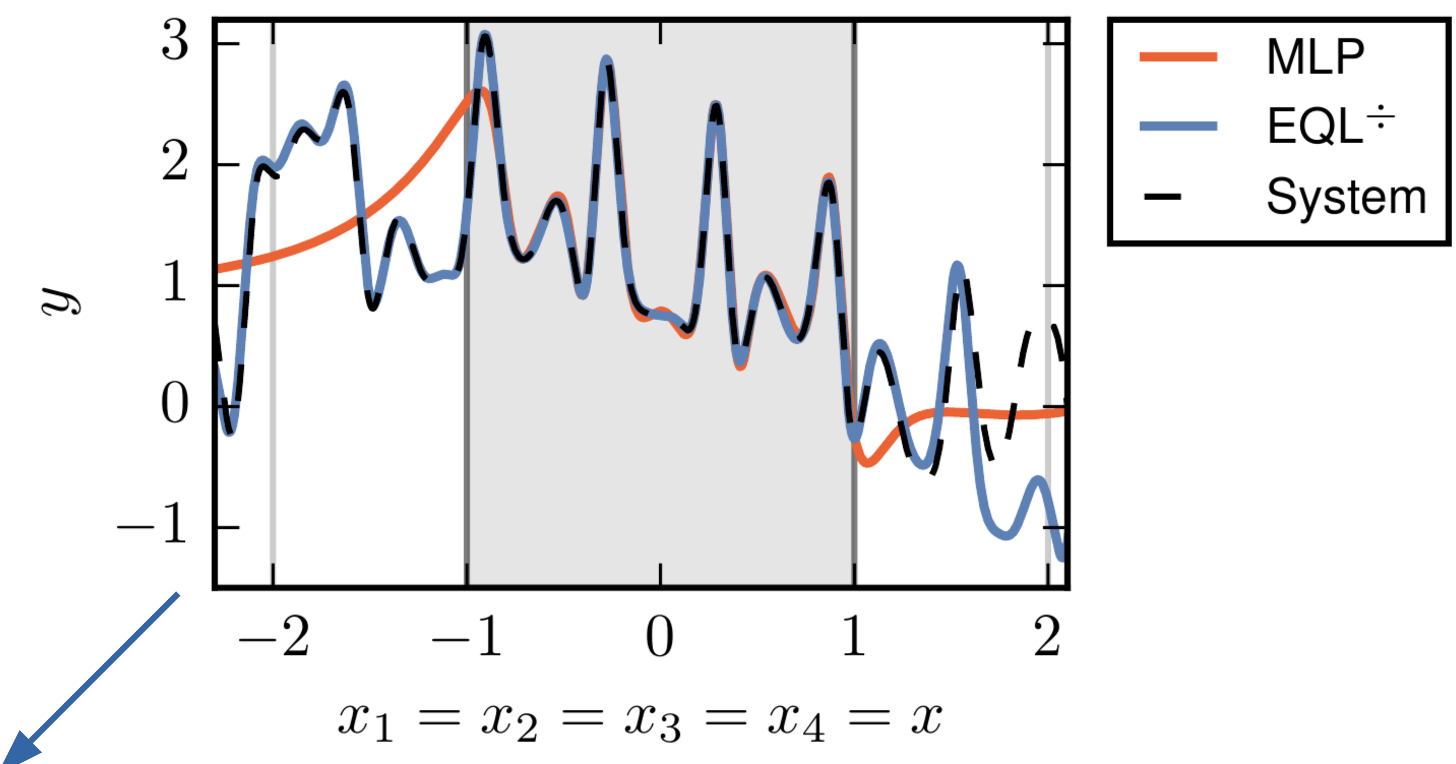
learned formula: $-0.33\sin(-3.13x_1) + 0.33\sin(6.28x_2 + 0.39) + 0.33x_2 - 0.056 - 0.33x_3x_4$

$$y = \frac{\sin(\pi x_1)}{(x_2^2 + 1)}$$



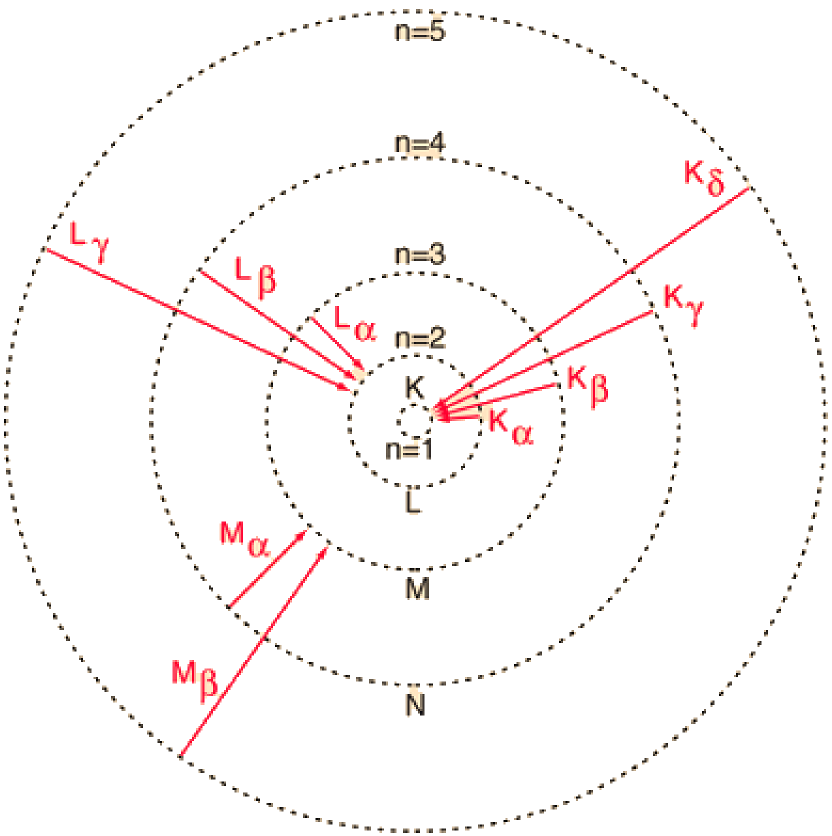
Random formulas

random formula
(RE2-2)

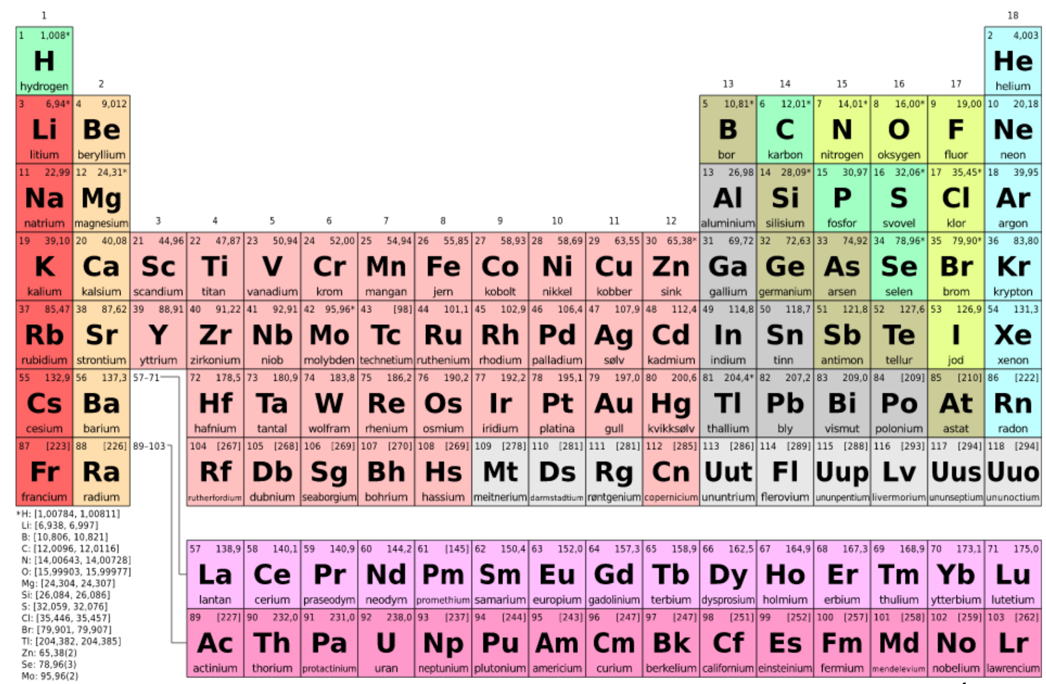


	RE2-1	RE2-2	RE2-3 ✗	RE2-4	RE3-1 ✗	RE3-2	RE3-3	RE3-4
EQL$^{\div}$ V^{ex}-S	0.05±0.05	0.07±0.02	0.70±0.12	0.01±0.00	0.89±0.56	0.47±0.45	0.10±0.12	0.53±0.32
EQL$^{\div}$ V^{int}-S	0.17±0.14	0.27±0.11	3.21±5.02	0.01±0.00	4.66±11.50	1.62±1.67	0.35±0.43	1.41±2.42
MLP V^{ex}	1.55±0.07	1.04±0.04	1.03±0.25	0.97±0.10	1.11±0.20	1.89±0.20	0.70±0.42	1.77±0.25
MLP V^{int}	1.57±0.07	1.05±0.03	1.45±0.15	1.00±0.09	1.32±0.17	2.03±0.16	1.30±0.47	1.86±0.17
SVR V^{ex}	1.15	1.06	0.59	1.51	0.75	1.81	0.37	1.23
SVR V^{int}	1.20	2.12	17.72	13.89	11.79	11.28	0.37	17.67
Const 0	6.73	2.57	0.50	5.36	1.65	72.26	17.67	3.15

X-Ray transition energies



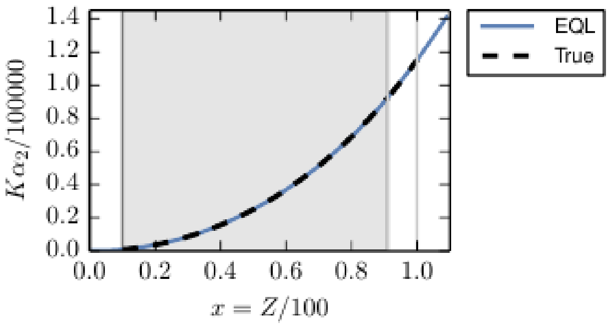
http://hyperphysics.phy-astr.gsu.edu



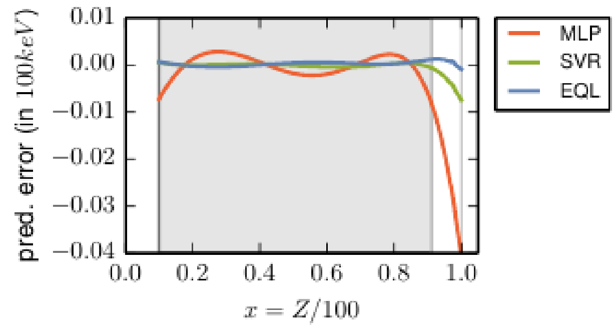
wikimedia

X-Ray transition energies

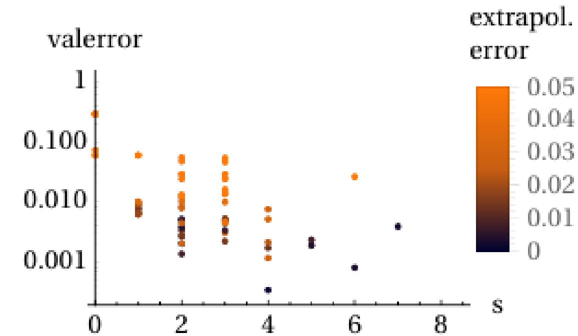
(a)



(b)



(c)



(d)

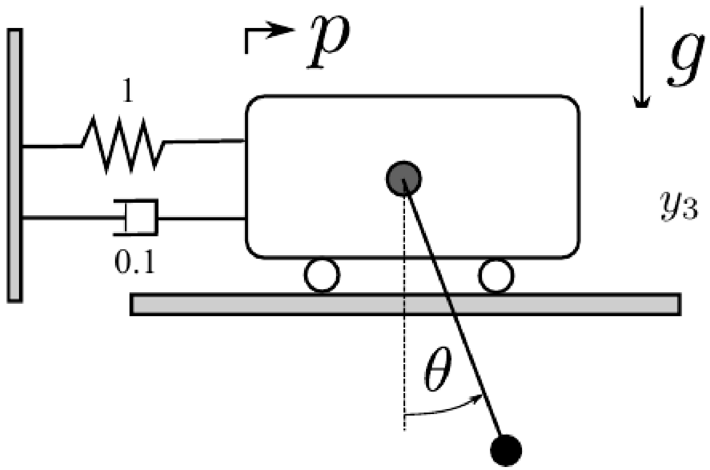
	interpolation	extrapolation
EQL	0.00042	0.0061 ± 0.0038
MLP	0.002	0.0180 ± 0.0024
SVR	0.00067	0.0057 ± 0.0014

(e)

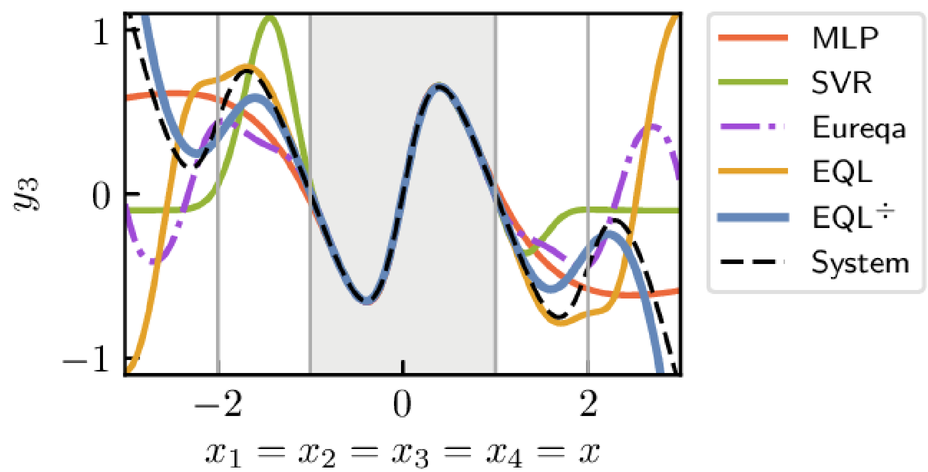
s	formula
1	$y = 1.28x^2 - 0.183x + 0.026$
2	$y = 1.98x^2 - 1.42x + 0.618 - 1.45\text{sigm}(-3.65x - 0.3)$
3	$y = -0.38z + 2.47\text{sigm}(-2.25z - 2.77) + 0.38$ with $z = \cos(2.32x - 0.08)$
4	$y = 0.221z + 0.42\text{sigm}(0.75z - 3.73)$ with $z = 4.65x^2 - 0.229x$

Martius, Lampert arXiv 2016

Cart-Pendulum dynamics



$$y_3 = \frac{-x_1 - 0.01x_3 + x_4^2 \sin(x_2) + 0.1x_4 \cos(x_2) + 9.81 \sin(x_2)\cos(x_2)}{\sin^2(x_2) + 1},$$

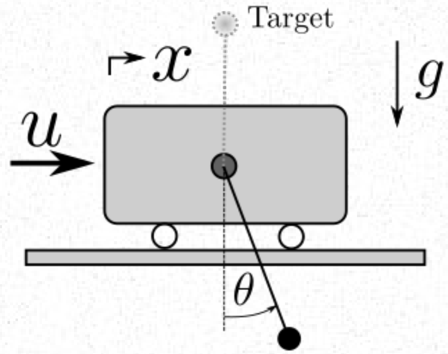


Able to learn dynamics equations

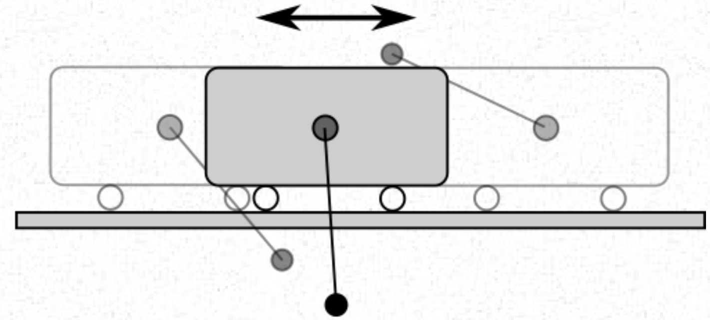
Sahoo, Lampert, Martius, ICML 2018

Learning Cart-Pole swingup

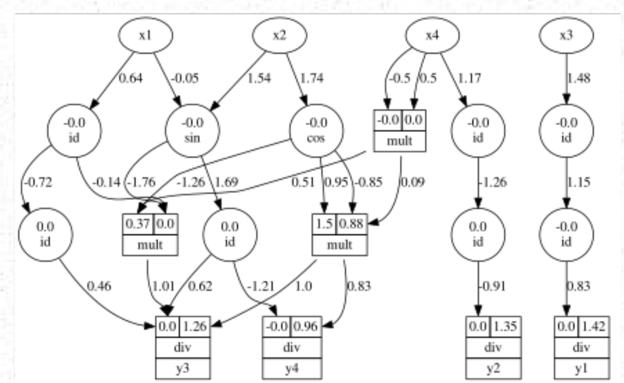
Robot



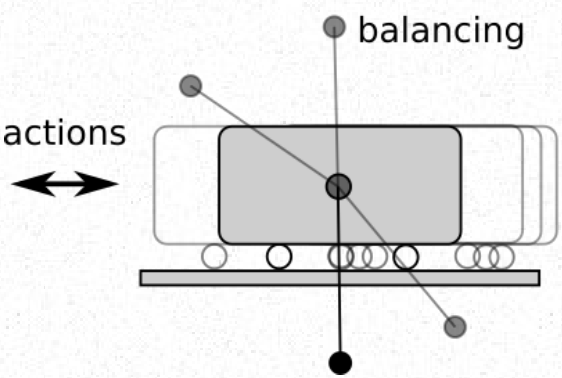
random movements



Learning Equation



Control

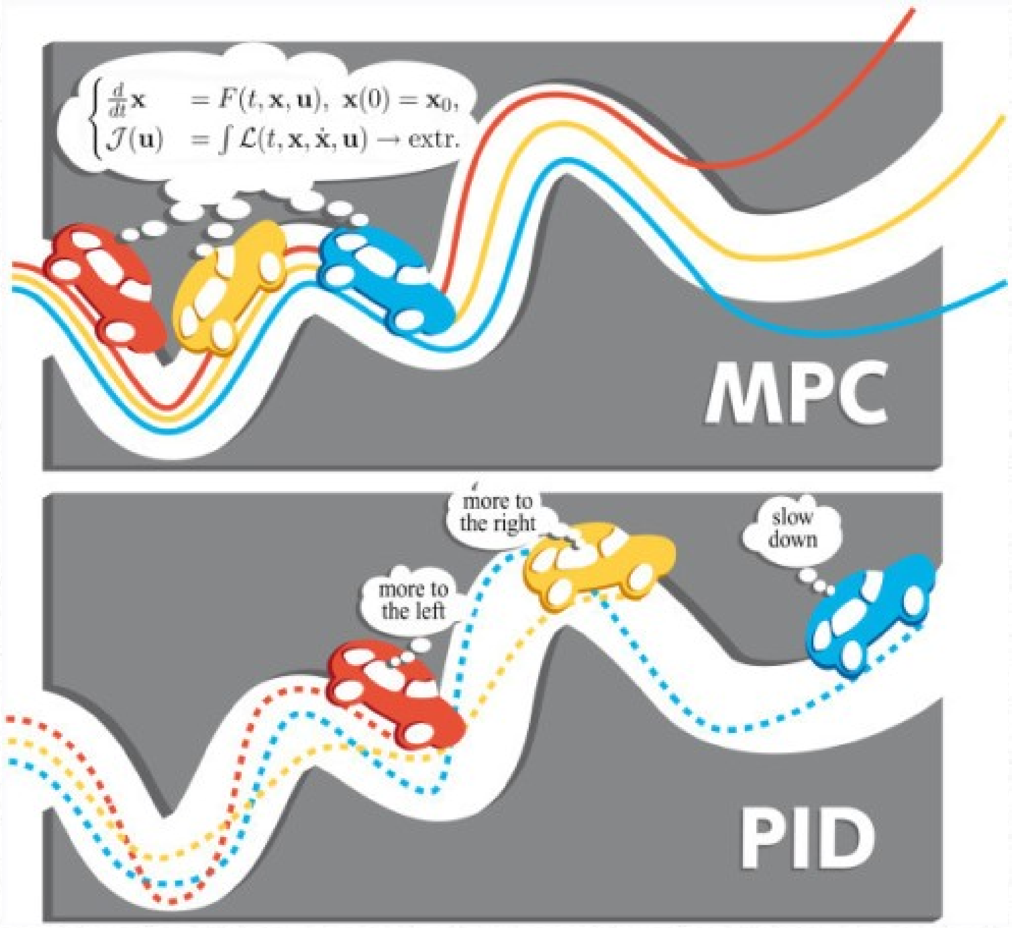


Model predictive Control,
random shooting method

Sahoo, Lampert, Martius, ICML 2018

Model Predictive Control

- plan ahead with model
- take best action
- replan



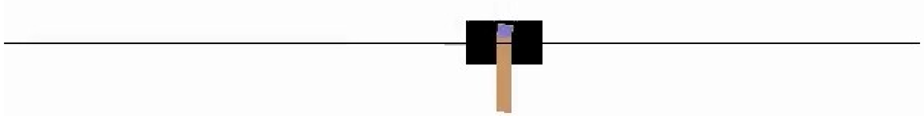
(openi.nlm.nih.gov)

here: planning = many random rollouts

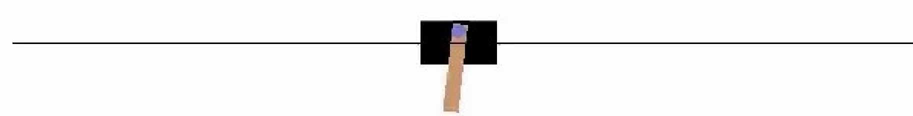
Learning Equations for Extrapolation and Control

by S.S.Sahoo, C.H.Lampert and G.Martius, ICML 2018

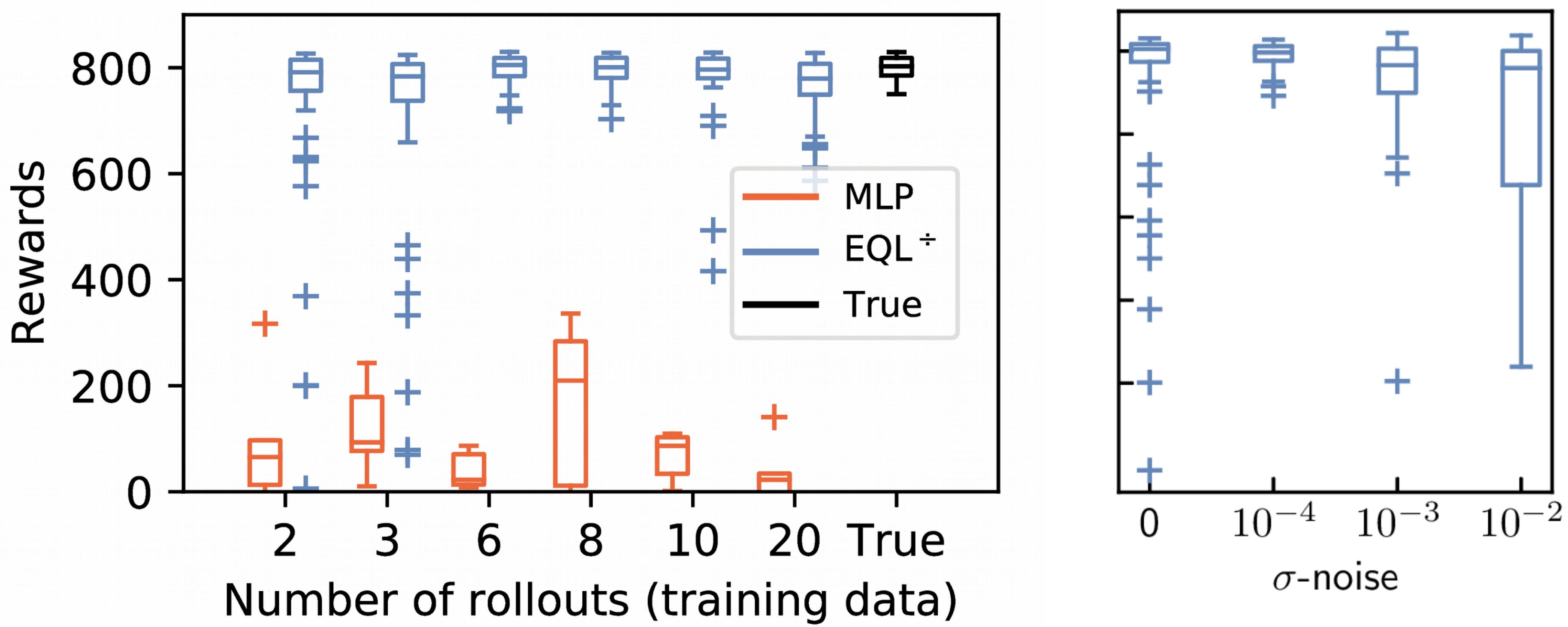
Training
1 Random rollout



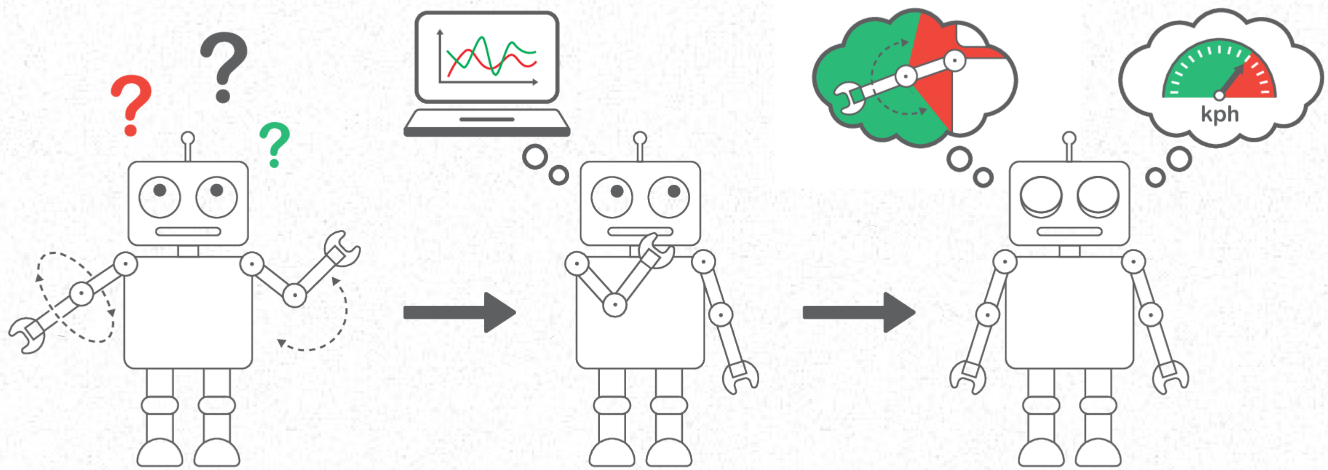
Validation
1 Random rollout
(stronger actions)



Cart-pole Swingup



Sahoo, Lampert, Martius, ICML 2018



- Robots need good learned models to become efficient
- Learning Equation from data

→ exquisite extrapolation capabilties

With

Code on github.com/martius-lab

Subham S Sahoo
IIT, India

Christoph Lampert
IST Austria