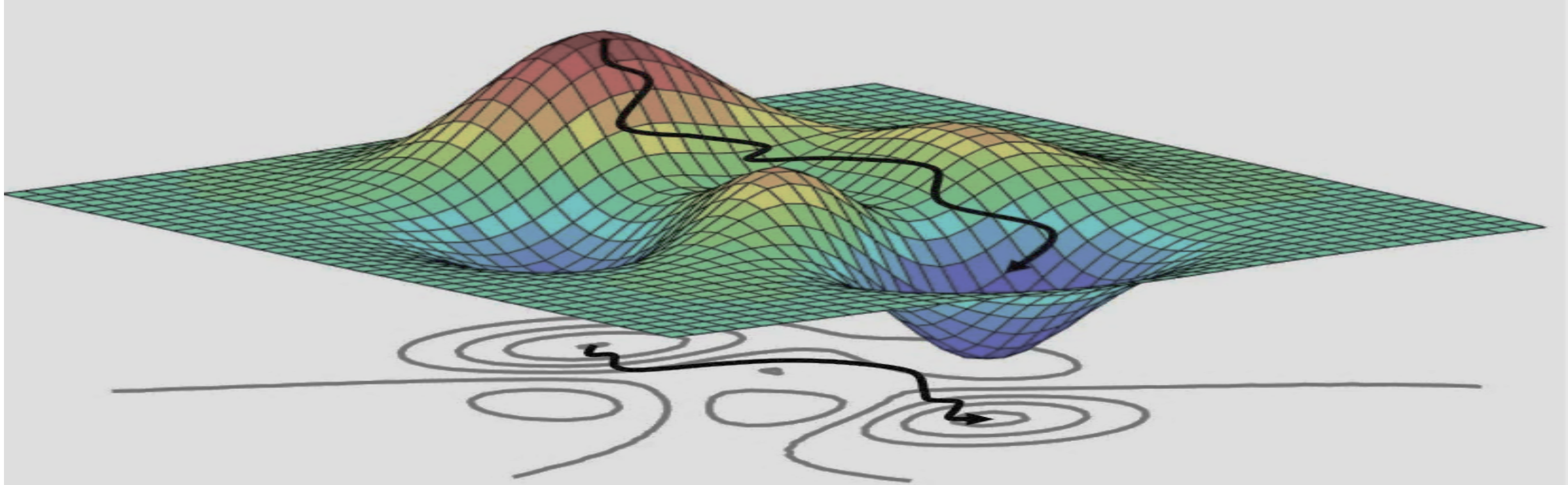


Learning to Optimize with Convergence Guarantees

using nonlinear system theory[1]

Luca Furieri

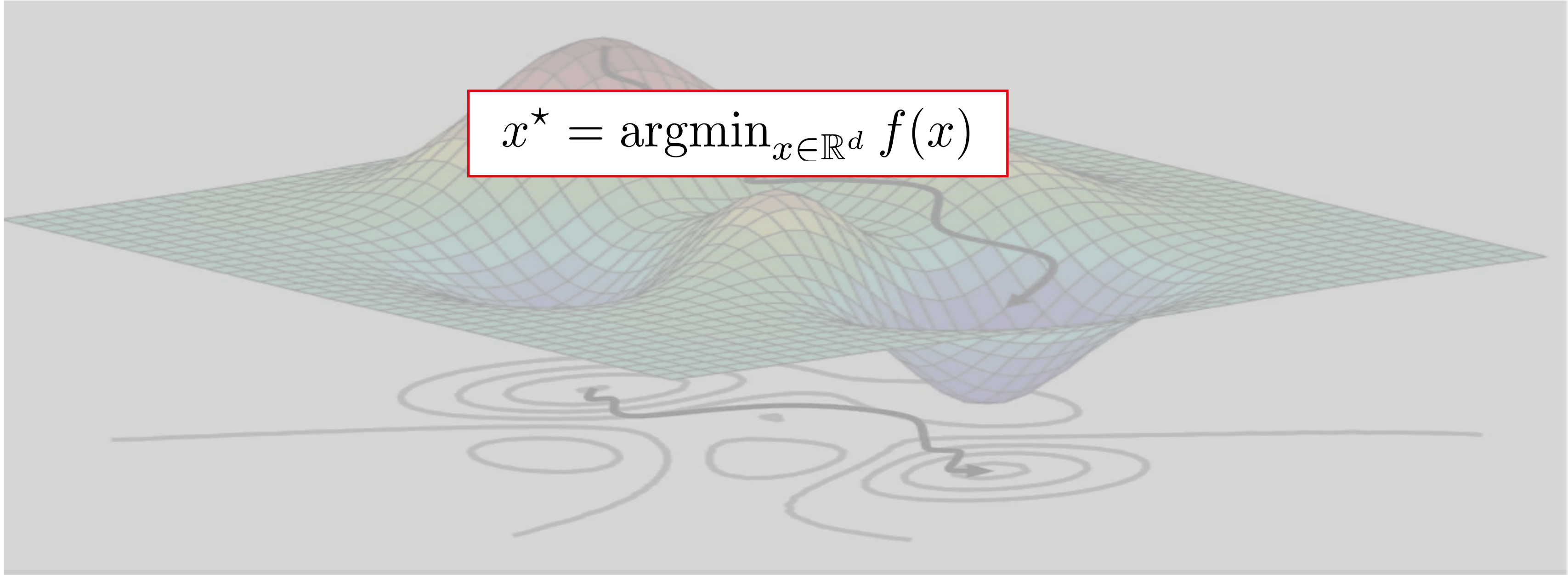
[1] Andrea Martin and Luca Furieri, “Learning to optimize with convergence guarantees using nonlinear system theory”, IEEE Control System Letters, 2024

École polytechnique fédérale de Lausanne

Swiss National Science Foundation

NCCR Automation

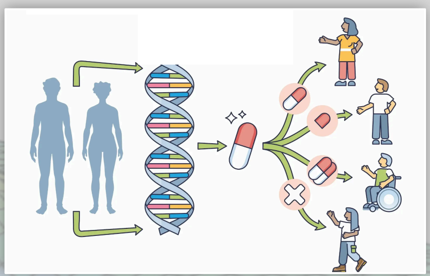
Optimization across disciplines

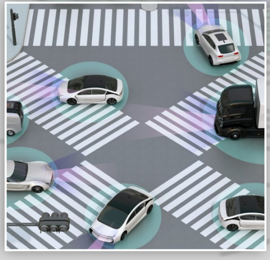


$$x^\star = \text{argmin}_{x \in \mathbb{R}^d} f(x)$$

Optimization across disciplines

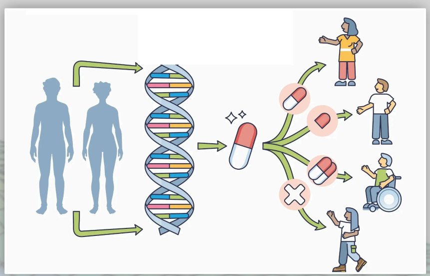
$$x^\star = \operatorname{argmin}_{x \in \mathbb{R}^d} f(x)$$

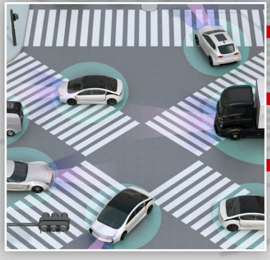




Optimization across disciplines

$$x^\star = \mathrm{argmin}_{x \in \mathbb{R}^d} f(x)$$





Nonconvex

Large scale

No analytical solution

Iterative optimization algorithms

$$x_{t+1} = x_t + \mathrm{update}_t(f, x_t)$$ *…until convergence*

(Hard) questions for the algorithm designer

1. ***Convergence*** as $t \to \infty$? From which initial guesses x_0?
2. ***How many interations*** for the algorithm to converge?
3. **Does the algorithm find a *«good» solution***?

(Hard) questions for the algorithm designer

1. ***Convergence*** as $t \to \infty$? From which initial guesses x_0?
2. ***How many interations*** for the algorithm to converge?
3. **Does the algorithm find a *«good» solution***?

Examples

(Hard) questions for the algorithm designer

1. ***Convergence*** as $t \to \infty$? From which initial guesses x_0?
2. ***How many interations*** for the algorithm to converge?
3. **Does the algorithm find a *«good» solution***?

Examples

- Steepest descent algorithm: $x_{t+1} = x_t - \eta \nabla f(x_t)$

1. **Any x0** 2. **Linear rate.** 3. **Closest stationary point**

(Hard) questions for the algorithm designer

1. ***Convergence*** as $t \to \infty$? From which initial guesses x_0?
2. ***How many interations*** for the algorithm to converge?
3. **Does the algorithm find a *«good» solution*?**

Examples

- Steepest descent algorithm: $x_{t+1} = x_t - \eta \nabla f(x_t)$

  1. Any x0 2. Linear rate. 3. Closest stationary point

- Newton-Raphson algorithm: $x_{t+1} = x_t - \left(\nabla^2 f(x_t)\right)^{-1} \nabla f(x_t)$

  1. (x0 close to x*, global variants) 2. Quadratic rate 3. Stationary point

(Hard) questions for the algorithm designer

1. ***Convergence*** as $t \to \infty$? From which initial gue... $_0$?
2. ***How m...***
3. **Does th...**

Hyper-parameter tuning

- Step-size, 1° and 2° momentum…
- Lack of a general theory beyond convex
  - In ML, best practices and know-how

→ **Design of new algorithms**

- Steepest de...

1. **Any x0** 2...

- Newton-Raph...

1. **(x0 close to** ... **tationary point**

- Adam algorithm *(Kingma et al., 2014)*

1. **May diverge on convex** 2. **Empirical** 3. **Deep learning, empirical**

(Hard) questions for the algorithm designer

1. ***Convergence*** as $t \to \infty$? From which initial gue… $_0$?
2. ***How m…***
3. **Does th…**

Hyper-parameter tuning

- Step-size, 1° and 2° momentum…
- Lack of a general theory beyond convex
  - In ML, best practices and know-how

→ **Design of new algorithms**

- Steepest de…

1. **Any x0** 2…

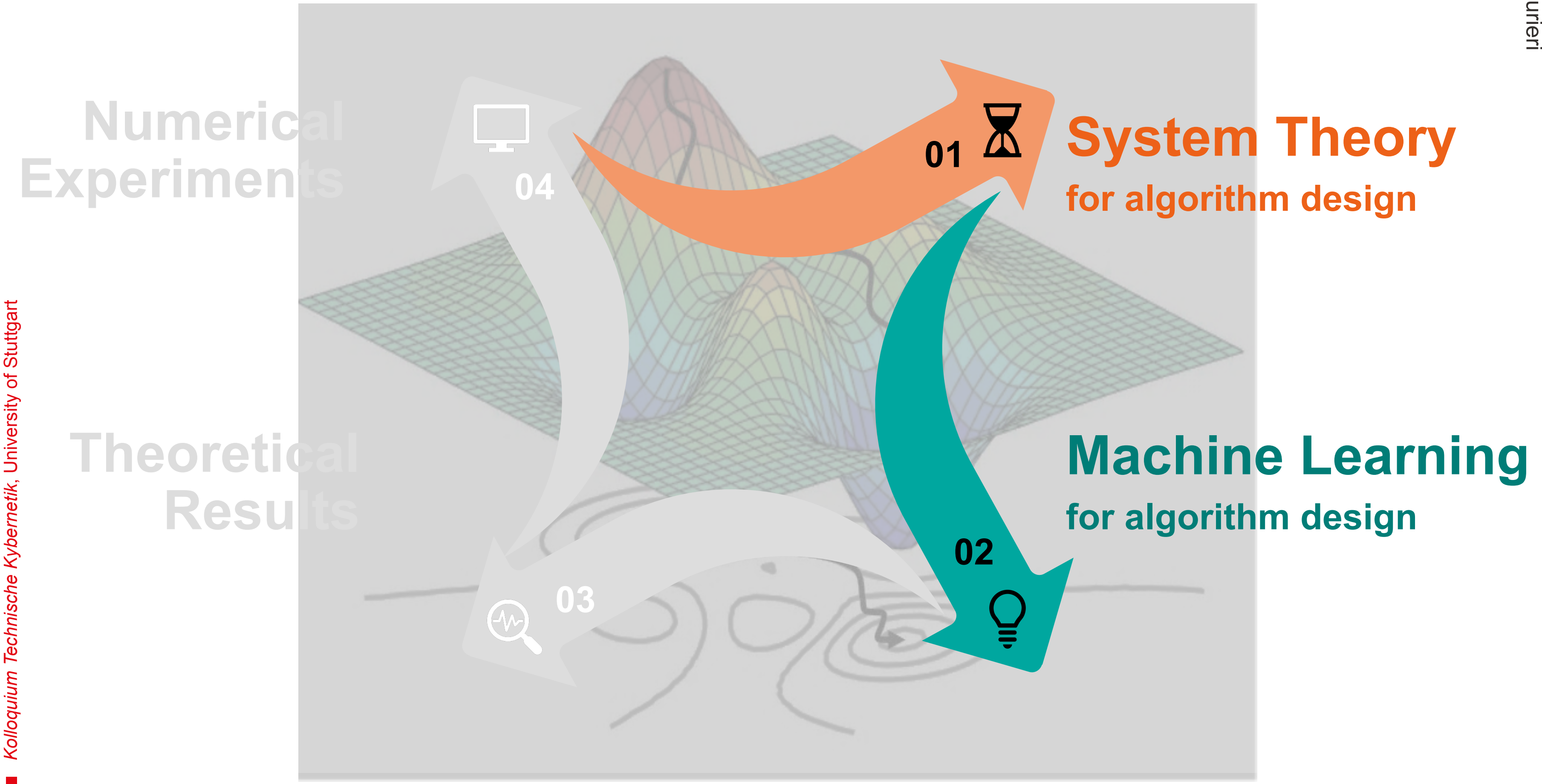
- Newton-Raph…

1. **(x0 close to** … **tationary point**

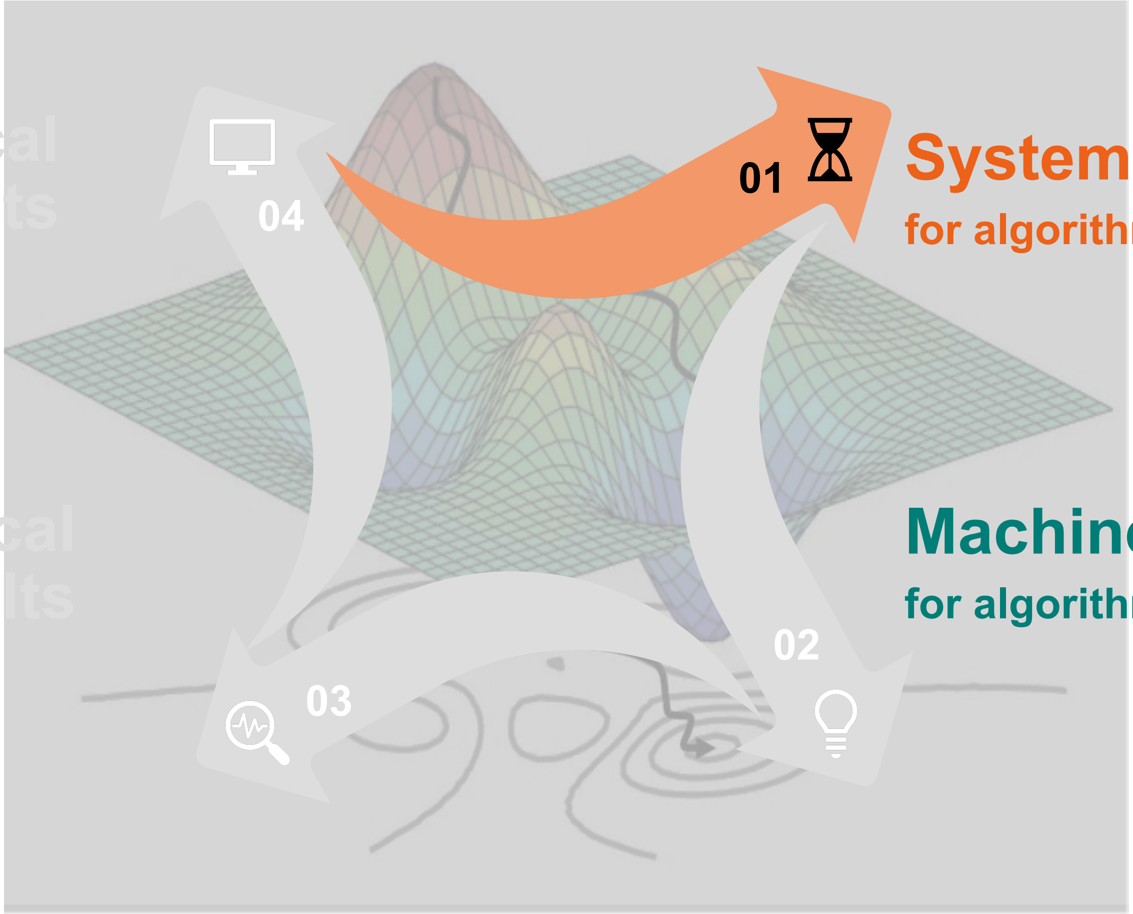
- Adam algorithm *(Kingma et al., 2014)*

1. **May diverge on convex** 2. **Empirical** 3. **Deep learning, empirical**

Design of new optimization algorithms



Design of new optimization algorithms



Numerical Experiments

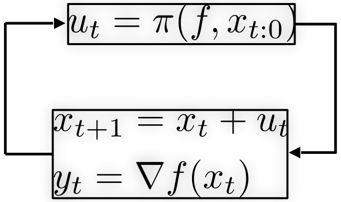
Theoretical Results

System Theory
for algorithm design

Machine Learning
for algorithm design

Success stories from system theory

Classical Algorithms $\equiv$ Control policies[1],[2]



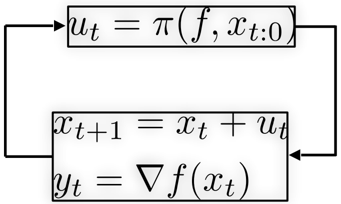
$$
u_t = \begin{cases} -\eta \nabla f(x_t) & \text{(gradient descent)} \\ -\eta \nabla f(x_t) + \beta(x_t - x_{t-1}) & \text{(heavy ball)}^{[1]} \\ -\left(\nabla^2 f(x_t)\right)^{-1} \nabla f(x_t) & \text{(Newton-Raphson)} \\ \dots \end{cases}
$$

[1] B. Polyak. *Introduction to Optimization*. Optimization Software, Inc., New York, 1987

[2] J. Wang and N. Elia. *A control perspective for centralized and distributed convex optimization*. In IEEE Conference on Decision and Control and European Control Conference Orlando, FL, USA. IEEE, 2011

Success stories from system theory

Classical Algorithms $\equiv$ Control policies[1],[2]



$$u_t = \begin{cases} -\eta \nabla f(x_t) & \text{(gradient descent)} \\ -\eta \nabla f(x_t) + \beta(x_t - x_{t-1}) & \text{(heavy ball)}^{[1]} \\ -\left(\nabla^2 f(x_t)\right)^{-1} \nabla f(x_t) & \text{(Newton-Raphson)} \\ \dots & \end{cases}$$

Algorithm design $\equiv$ *Robust* control problem

For every cost function f in a class $\mathcal{F}$, designed algo must achieve:

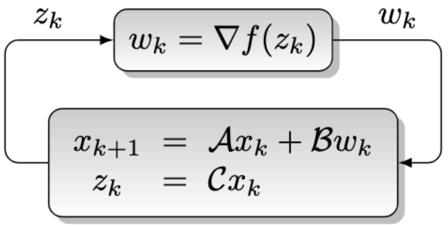
1. ***Convergence:*** regulate $y_t = \nabla f(x_t)$ to zero
2. ***Speed:*** $\mathcal{H}_2/\mathcal{H}_\infty$ guarantees, exponential stability…
3. ***Solution quality:*** e.g., performance of equilibria

[1] B. Polyak. *Introduction to Optimization*. Optimization Software, Inc., New York, 1987

[2] J. Wang and N. Elia. *A control perspective for centralized and distributed convex optimization*. In IEEE Conference on Decision and Control and European Control Conference Orlando, FL, USA. IEEE, 2011

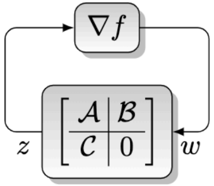
Success stories from system theory

- Classical algorithms equivalent to uncertain LTIs



$$\left(\begin{array}{c|c} \mathcal{A} & \mathcal{B} \\ \hline \mathcal{C} & 0 \end{array}\right) = \left(\begin{array}{cc|c} (1+\beta)I_d & -\beta I_d & -\alpha I_d \\ I_d & 0 & 0 \\ \hline (1+\gamma)I_d & -\gamma I_d & 0 \end{array}\right)$$

- ball algorithm by Polyak:



- … **is a Lure's system**[1,2]! We know a lot about them…

[1] Lessard, L., Recht, B., & Packard, A. (2016). Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, *26*(1), 57-95.

[2] Scherer, C., & Ebenbauer, C. (2021). Convex synthesis of accelerated gradient algorithms. *SIAM Journal on Control and Optimization*, *59*(6), 4615-4645.

Success stories from system theory

- **Breakthrough[1,2]:** convex design of algorithm $\mathcal{A}, \mathcal{B}, \mathcal{C}$ such that

$$||x_k - x^\star|| \leq K\rho^k ||x_0 - x^\star|| \qquad K > 0,\ \ \rho \in (0,1)$$

for all cost functions that are **(strongly)-convex and smooth**

- Integral Quadratic Constraints (IQCs) for optimal worst-case convergence rates[1,2]
- Extrememum–seeking control, algorithms with delayed info…[2]

[1] Lessard, L., Recht, B., & Packard, A. (2016). Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, *26*(1), 57-95.

[2] Scherer, C., & Ebenbauer, C. (2021). Convex synthesis of accelerated gradient algorithms. *SIAM Journal on Control and Optimization*, *59*(6), 4615-4645.

Success stories from system theory

- **Breakthrough[1,2]:** convex design of algorithm $\mathcal{A}, \mathcal{B}, \mathcal{C}$ such that

$$||x_k - x^\star|| \leq K\rho^k ||x_0 - x^\star|| \qquad K > 0,\ \ \rho \in (0,1)$$

for all cost functions that are **(strongly)-convex and smooth**

- Integral Quadratic Constraints (IQCs) for optimal worst-case convergence rates[1,2]
- Extrememum–seeking control, algorithms with delayed info…[2]

A main take-away

Crucial role of nonlinear, robust system theory

for the next generation of optimization algorithms

[1] Lessard, L., Recht, B., & Packard, A. (2016). Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, *26*(1), 57-95.

[2] Scherer, C., & Ebenbauer, C. (2021). Convex synthesis of accelerated gradient algorithms. *SIAM Journal on Control and Optimization*, *59*(6), 4615-4645.

EPFL

Success stories from system theory

- **Breakthrough[1,2]:** convex design of algorithm $\mathcal{A}, \mathcal{B}, \mathcal{C}$ such that

  $\in (0, 1)$

  for all cost functio

- Integral Quadratic … gence rates[1,2]
- Extrememum–seek…

Open challenges

- Algorithms for **non-convex** cost functions
- Scaling to very large problems
- Beyond worst-case rates? (e.g. solution quality)

<u>A main take-away</u>

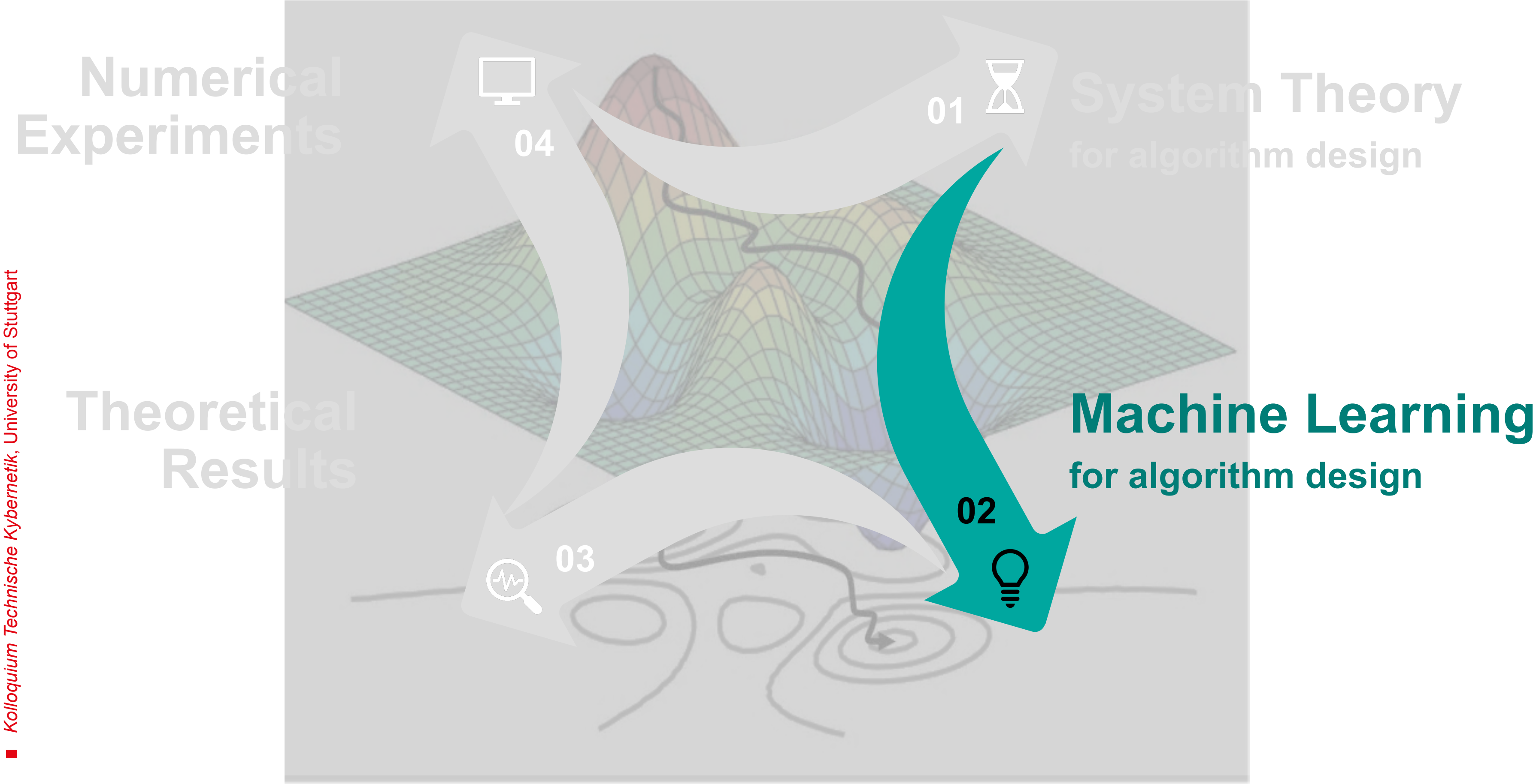
Crucial role of nonlinear, robust system theory

for the next generation of optimization algorithms

[1] Lessard, L., Recht, B., & Packard, A. (2016). Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, *26*(1), 57-95.

[2] Scherer, C., & Ebenbauer, C. (2021). Convex synthesis of accelerated gradient algorithms. *SIAM Journal on Control and Optimization*, *59*(6), 4615-4645.

Design of new optimization algorithms



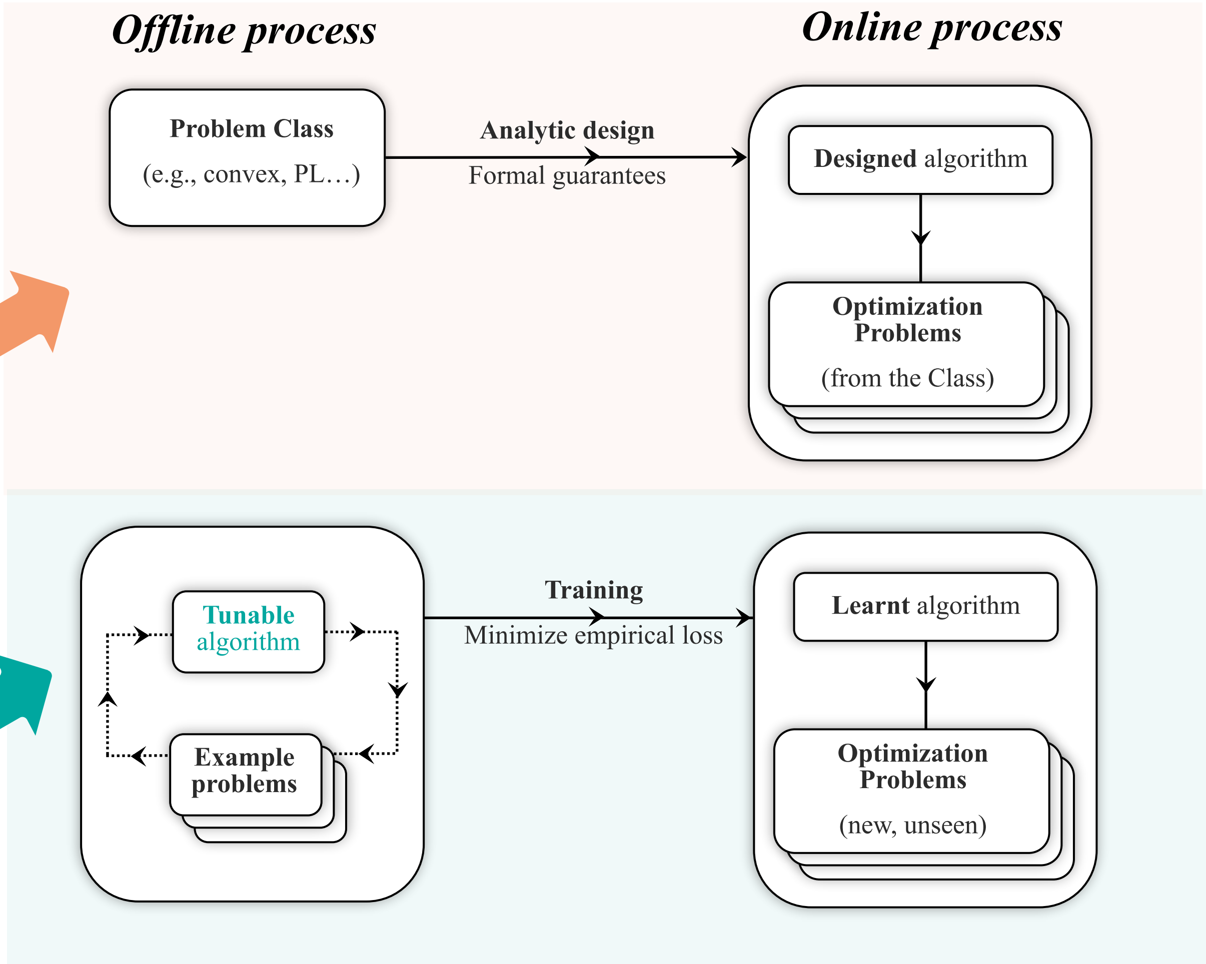
Different philosophies for algorithm design

	Offline process	*Online process*
System theory point-of-view		
Machine learning point-of-view		

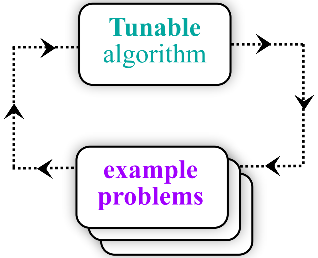
Different philosophies for algorithm design

System theory point-of-view

Machine learning point-of-view

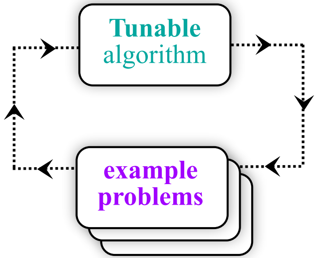


«Learning To Optimize» (L2O) for optimal control



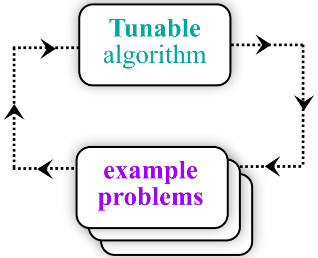
«Learning To Optimize» (L2O) for optimal control

L. Furieri



- **Crucial**: «distribution» of **example problems**
  - *generalizability* VS *performance*
- Rule of thumb
  - *effective when* ***example problems*** *are representative of future instances*

«Learning To Optimize» (L2O) for optimal control



- **Crucial**: «distribution» of **example problems**
  - *generalizability* VS *performance*
- Rule of thumb
  - *effective when **example problems** are representative of future instances*

Prime example: Nonlinear Model Predictive Control (NMPC)

Repeatedly solve…

$$u^\star(x(t)) = \arg\min_{\{u_k\}_{k=0}^{N-1}} \sum_{k=0}^{N-1} l(x_k, u_k) + l_f(x_N)$$

$$\text{subject to} \quad x_0 = x(t), \quad x_{k+1} = f(x_k, u_k)$$

$$x_k \in \mathcal{X}, \; u_k \in \mathcal{U}, \; x_N \in \mathcal{X}_f$$

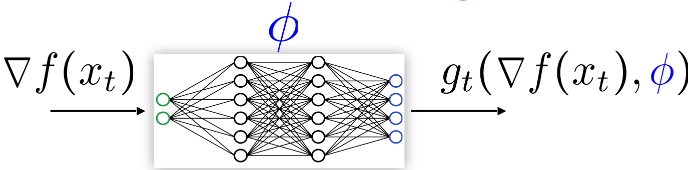
"examples"

A learnt algorithm can discover shortcuts unknown to standard interior-point solvers

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:

$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

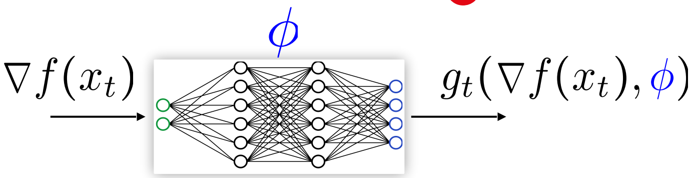


[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016
[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:

$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$



- **Measure performance** of an algorithm over T steps:

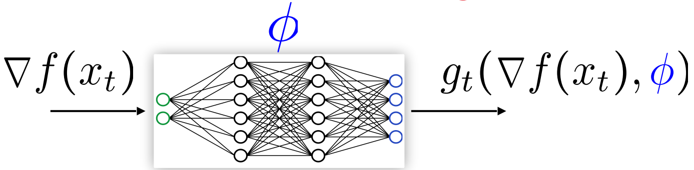
$$L(\phi) = \mathbb{E}_{f \sim \mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016
[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:



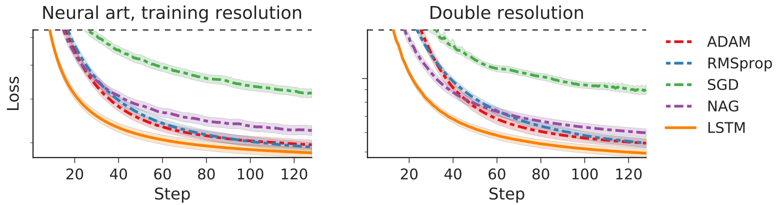
$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f \sim \mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems

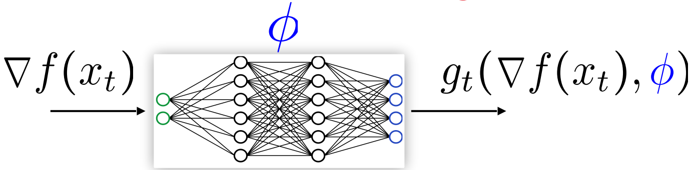


[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016

[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:



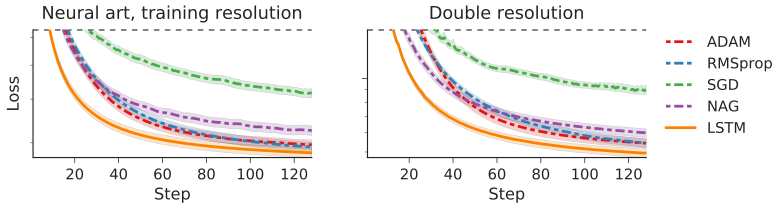
$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f \sim \mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

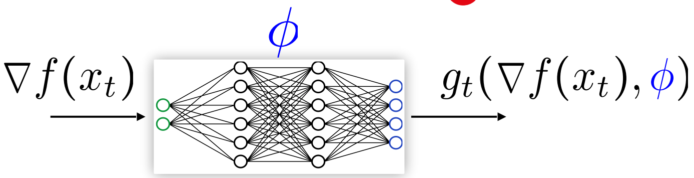
- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems



[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016

[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]



- **Parametrize** algorithms as follows:

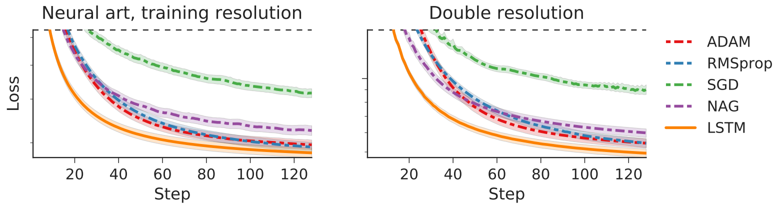
$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f \sim \mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

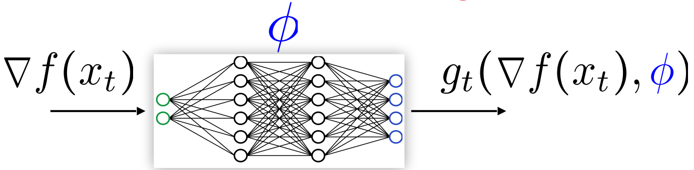
- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems



[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016
[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:



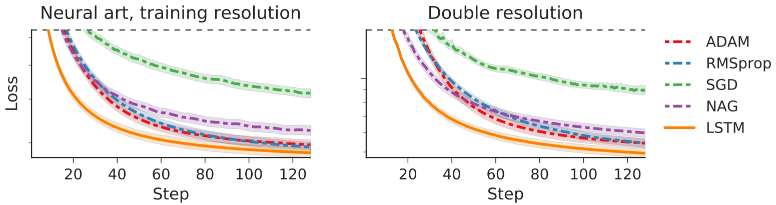
$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f\sim\mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

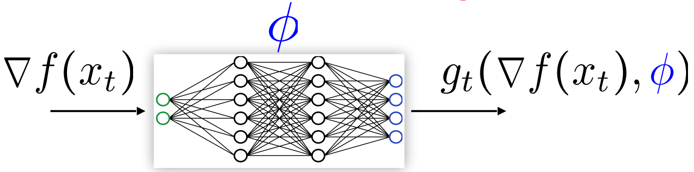
- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems



[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016
[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:



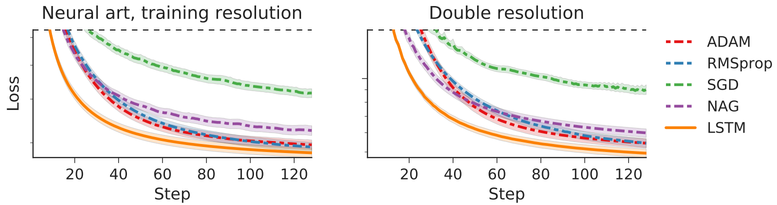
$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f\sim\mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems



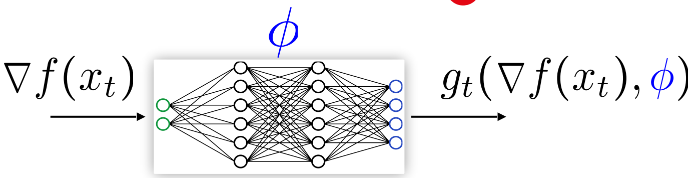
[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016

[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:

$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

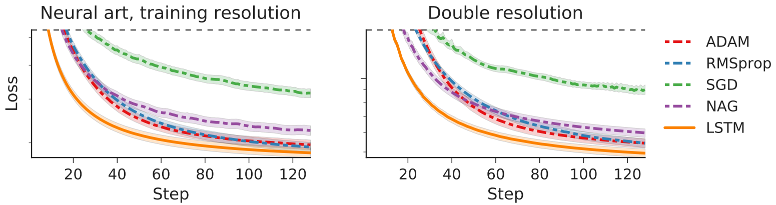


- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f \sim \mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems



[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016

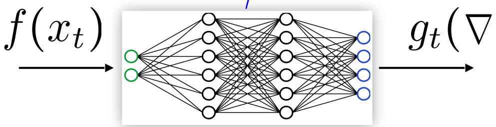
[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:

$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

$\nabla f(x_t)$ → ϕ → $g_t(\nabla f(x_t), \phi)$

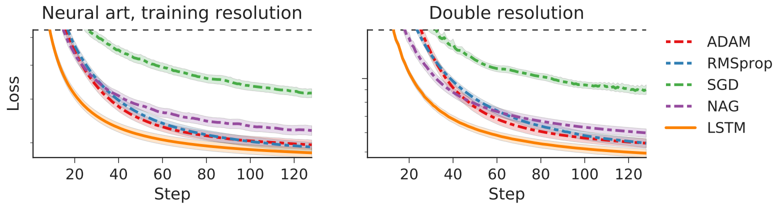


- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f\sim\mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems



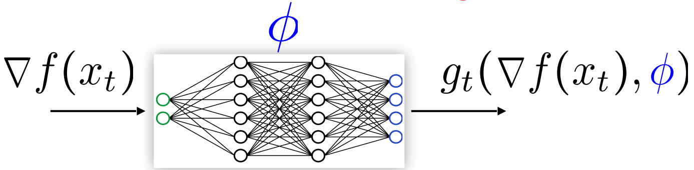
[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016

[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:

$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

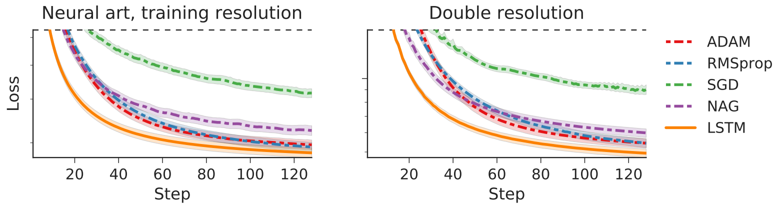


- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f \sim \mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems

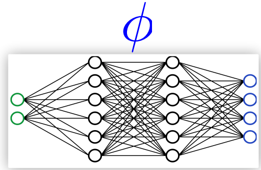


[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016

[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

Success of L2O in machine learning[1,2,3]

- **Parametrize** algorithms as follows:

$\nabla f(x_t) \rightarrow$  $\rightarrow g_t(\nabla f(x_t), \phi)$

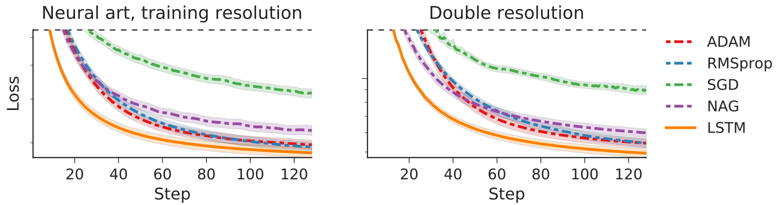
$$x_{t+1} = x_t + g_t(\nabla f(x_t), \phi)$$

- **Measure performance** of an algorithm over T steps:

$$L(\phi) = \mathbb{E}_{f \sim \mathcal{F}, x_0 \sim X_0} \sum_{t=0}^{T} f(x_t)$$

where $\mathcal{F}$ is the set of «*example problems*»

- **Generalization capabilities**! Train over MNIST problems, test on NEURAL ART problems



[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NeurIPS, 2016
[2] Li, Ke, and Jitendra Malik. "Learning to optimize." *ICLR, 2017*

EPFL

Open challenges for L2O

Generalization: no guarantees!

- Performance on unseen problem?
- As hard as a *transfer-learning* problem

Open challenges for L2O

Generalization: no guarantees!

- Performance on unseen problem?
- As hard as a *transfer-learning* problem

Scalability: expensive training

- Repeatedly roll-out algorithm on example problems… **before** improving parameters

Open challenges for L2O

Generalization: no guarantees!

- Performance on unseen problem?
- As hard as a *transfer-learning* problem

Scalability: expensive training

- Repeatedly roll-out algorithm on example problems… **before** improving parameters

Convergence: no guarantees![1]

- Requires conservative **fallback mechanisms**… …even for the convex case[2]

[1] Harrison, James, Luke Metz, and Jascha Sohl-Dickstein. "*A closer look at learned optimization: Stability, robustness, and inductive biases*." NeurIPS, 2022

- [2] Heaton, Howard, et al. "*Safeguarded learned convex optimization*." AAAI Conference, 2023.

Open challenges for L2O

Generalization: no guarantees!

- Performance on unseen problem?
- As hard as a *transfer-learning* problem

Scalability: expensive training

- Repeatedly roll-out algorithm on example problems… **before** improving parameters

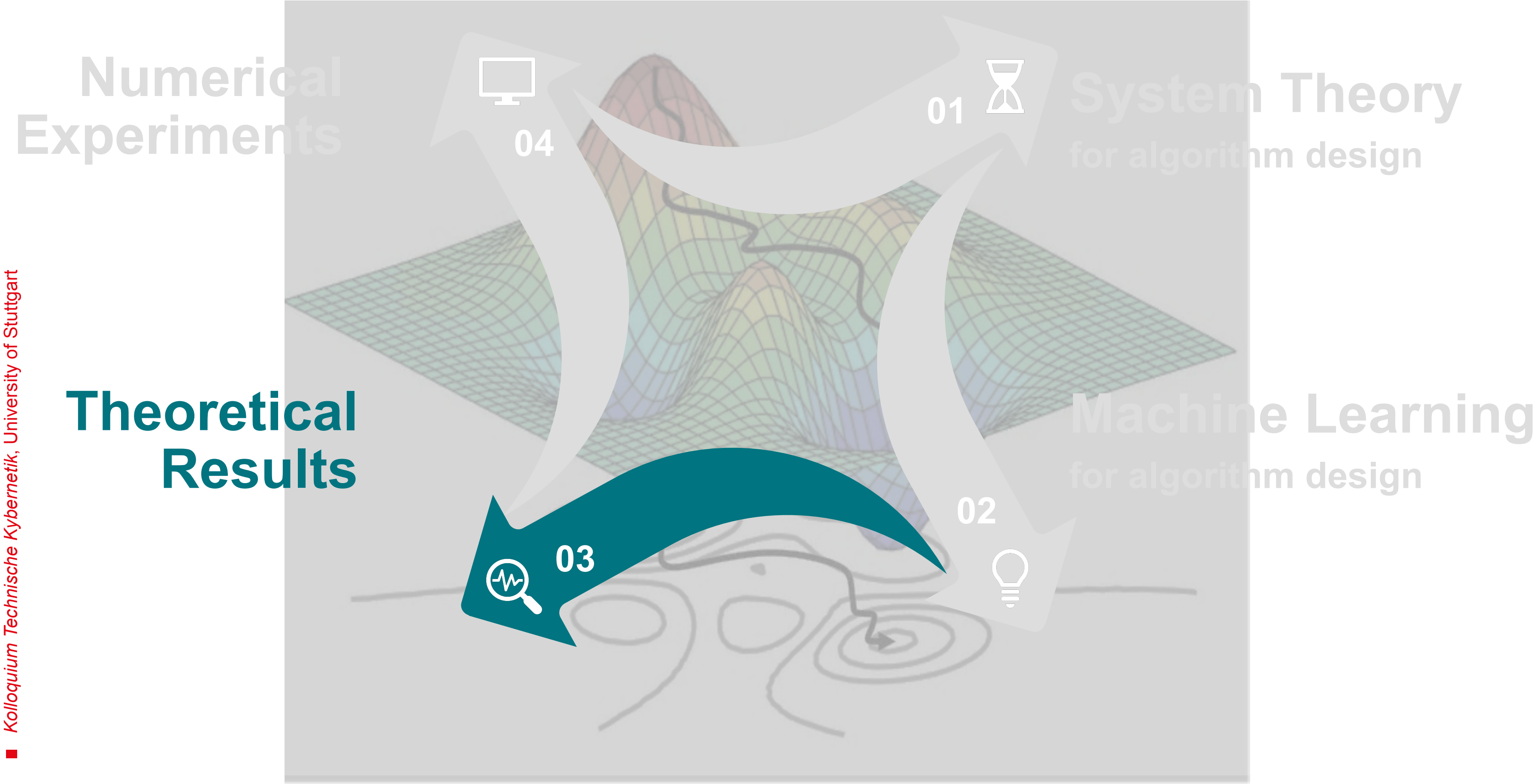
Focus of this work

- **Learn convergent algorithms** for smooth non-convex functions
  - …beyond conservative heuristics[1,2]

[1] Harrison, James, Luke Metz, and Jascha Sohl-Dickstein. "*A closer look at learned optimization: Stability, robustness, and inductive biases*." NeurIPS, 2022

- [2] Heaton, Howard, et al. "*Safeguarded learned convex optimization*." AAAI Conference, 2023.

Design of new optimization algorithms



Problem Formulation (1)

- Denote as $\mathcal{S}_\beta$ the set of **non-convex** cost functions with β- Lipschitz gradients:

$$f \in \mathcal{S}_\beta \iff |\nabla f(x) - \nabla f(y)| \leq \beta |x - y|$$

Problem Formulation (1)

- Denote as $\mathcal{S}_\beta$ the set of **non-convex** cost functions with β- Lipschitz gradients:

$$f \in \mathcal{S}_\beta \iff |\nabla f(x) - \nabla f(y)| \leq \beta |x - y|$$

- Signal space notation:

$$\begin{cases} x_{t+1} = x_t + \pi_t(f, x_t) \\ x_0 = x_0 \end{cases} \iff z\mathbf{x} = \mathbf{x} + \boldsymbol{\pi}(f, \mathbf{x}) + z\boldsymbol{\delta}_0$$

$z \equiv$ one-time-step shift

$\mathbf{x} = (x_0, x_1, x_2, \ldots)$

$\boldsymbol{\delta}_0 = (x_0, 0, 0, \ldots)$

$\boldsymbol{\pi}(f, \mathbf{x}) \equiv$ algorithm

Problem Formulation (1)

- Denote as $\mathcal{S}_\beta$ the set of **non-convex** cost functions with β- Lipschitz gradients:

$$f \in \mathcal{S}_\beta \iff |\nabla f(x) - \nabla f(y)| \leq \beta |x - y|$$

- Signal space notation:

$$\begin{cases} x_{t+1} = x_t + \pi_t(f, x_t) \\ x_0 = x_0 \end{cases} \iff z\mathbf{x} = \mathbf{x} + \boldsymbol{\pi}(f, \mathbf{x}) + z\boldsymbol{\delta}_0$$

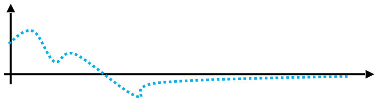
$z \equiv$ one-time-step shift

$\mathbf{x} = (x_0, x_1, x_2, \dots)$

$\boldsymbol{\delta}_0 = (x_0, 0, 0, \dots)$

$\boldsymbol{\pi}(f, \mathbf{x}) \equiv$ algorithm

- ℓ_2-stable *signals* defined as $\mathbf{x} \in \ell_2 \iff \sum_{t=0}^{\infty} |x_t|_2^2 < \infty$



- $\mathcal{L}_2$-stable operators: $\mathbf{A}(\mathbf{x}) \in \ell_2, \ \forall \mathbf{x} \in \ell_2$

Problem Formulation (1)

- Denote as $\mathcal{S}_\beta$ the set of **non-convex** cost functions with β- Lipschitz gradients:

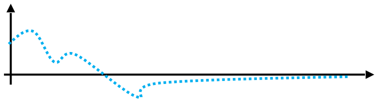
$$f \in \mathcal{S}_\beta \iff |\nabla f(x) - \nabla f(y)| \leq \beta |x - y|$$

- Signal space notation:

$$\begin{cases} x_{t+1} = x_t + \pi_t(f, x_t) \\ x_0 = x_0 \end{cases} \iff z\mathbf{x} = \mathbf{x} + \boldsymbol{\pi}(f, \mathbf{x}) + z\boldsymbol{\delta}_0$$

$z \equiv$ one-time-step shift
$\mathbf{x} = (x_0, x_1, x_2, \dots)$
$\boldsymbol{\delta}_0 = (x_0, 0, 0, \dots)$
$\boldsymbol{\pi}(f, \mathbf{x}) \equiv$ algorithm

- ℓ_2-stable *signals* defined as $\mathbf{x} \in \ell_2 \iff \sum_{t=0}^{\infty} |x_t|_2^2 < \infty$



- $\mathcal{L}_2$-stable operators: $\mathbf{A}(\mathbf{x}) \in \ell_2, \ \forall \mathbf{x} \in \ell_2$

- **Algorithms that convergence for a function** $f \in \mathcal{S}_\beta$

> "$\boldsymbol{\pi}$ *is **sum-square convergent** for f*" $\iff$ $\|\nabla f(\boldsymbol{x})\|^2 < \infty \quad \|\boldsymbol{\pi}(f, \boldsymbol{x})\|^2 < \infty,$
> "$\boldsymbol{\pi} \in \Sigma(f)$" $\left(= \sqrt{\sum_{t=0}^{\infty} |\nabla f(x_t)|^2}\right)$

Problem Formulation (2)

- **Problem**: *design an optimal robustly convergent algorithm*

$$\begin{aligned} \min_{\boldsymbol{\pi}} \quad & \mathbb{E}_{f\sim\text{Examples}, x_0 \sim \mathcal{X}_0} \left[\texttt{AlgoPerf}(f, \mathbf{x})\right] \\ \text{s.t.} \quad & x_{t+1} = x_t + \pi_t(f, x_{t:0}), \\ & \boldsymbol{\pi}(f, \mathbf{x}) \in \Sigma(f), \ \forall f \in \mathcal{S}_\beta \,, \end{aligned}$$

- [1] D. P. Bertsekas and J. N. Tsitsiklis, “Gradient convergence in gradient methods with errors,” SIAM Journal on Optimization, vol. 10, no. 3, pp. 627–642, 2000.

EPFL

Problem Formulation (2)

- **Problem**: *design an optimal robustly convergent algorithm*

$$\min_{\boldsymbol{\pi}} \quad \mathbb{E}_{f\sim \text{Examples}, x_0 \sim \mathcal{X}_0} [\texttt{AlgoPerf}(f, \mathbf{x})]$$
$$\text{s.t.} \quad x_{t+1} = x_t + \pi_t(f, x_{t:0}),$$
$$\boldsymbol{\pi}(f, \mathbf{x}) \in \Sigma(f), \; \forall f \in \mathcal{S}_\beta \,,$$

- **Constraints**: limit the search to algorithms that converge for all smooth functions
  - E.g.: $\pi(f, x_t) = -\eta \nabla f(x_t)$ belongs to $\Sigma(f)$ for every $f \in \mathcal{S}_\beta$, if $\eta < \beta^{-1}$[1]
  - Beyond gradient descent, case-by-case mathematical analysis

[1] D. P. Bertsekas and J. N. Tsitsiklis, “Gradient convergence in gradient methods with errors,” SIAM Journal on Optimization, vol. 10, no. 3, pp. 627–642, 2000.

Problem Formulation (2)

- **Problem**: *design an optimal robustly convergent algorithm*

$$\min_{\boldsymbol{\pi}} \quad \mathbb{E}_{f\sim\text{Examples},x_0\sim\mathcal{X}_0}\left[\texttt{AlgoPerf}(f,\mathbf{x})\right]$$
$$\text{s.t.} \quad x_{t+1} = x_t + \pi_t(f, x_{t:0}),$$
$$\boldsymbol{\pi}(f,\mathbf{x}) \in \Sigma(f),\ \forall f \in \mathcal{S}_\beta\,,$$

- **Constraints**: limit the search to algorithms that converge for all smooth functions
  - E.g.: $\pi(f,x_t) = -\eta\nabla f(x_t)$ belongs to $\Sigma(f)$ for every $f \in \mathcal{S}_\beta$, if $\eta < \beta^{-1}$[1]
  - Beyond gradient descent, case-by-case mathematical analysis

- **Performance metric**:

$$\texttt{AlgoPerf}(f,\mathbf{x}) = \sum_{t=0}^{T} \alpha_t |\nabla f(x_t)|^2 + \gamma_t f(x_t)\,,$$

to balance fast convergence and solution quality

[1] D. P. Bertsekas and J. N. Tsitsiklis, “Gradient convergence in gradient methods with errors,” SIAM Journal on Optimization, vol. 10, no. 3, pp. 627–642, 2000.

Main result 1: a separation principle

- Consider the following algorithm:

$$\boldsymbol{\pi}(f, \mathbf{x}) = -\eta \nabla f(\mathbf{x}) + \mathbf{v}$$

- **Key idea:** separate algorithms into two addends

[1] Andrea Martin, LF, "Learning to optimize with convergence guarantees using nonlinear system theory", IEEE Control System Letters, 2024, to appear

Main result 1: a separation principle

- Consider the following algorithm:

$$\boldsymbol{\pi}(f, \mathbf{x}) = \boxed{-\eta \nabla f(\mathbf{x})} + \boxed{\mathbf{v}}$$

- **Key idea:** separate algorithms into two addends
  1. Gradient descent, aiming to ensure convergence on $\mathcal{S}_\beta$
  2. $\mathbf{v}$ is a performance-boosting term to be designed, *without ruining convergence*

- [1] Andrea Martin, LF, “Learning to optimize with convergence guarantees using nonlinear system theory”, IEEE Control System Letters, 2024, to appear

Main result 1: a separation principle

- Consider the following algorithm:

$$\boldsymbol{\pi}(f, \mathbf{x}) = \boxed{-\eta \nabla f(\mathbf{x})} + \boxed{\mathbf{v}}$$

- **Key idea:** separate algorithms into two addends
  1. Gradient descent, aiming to ensure convergence on $\mathcal{S}_\beta$
  2. $\mathbf{v}$ is a performance-boosting term to be designed, *without ruining convergence*

Result: The algorithm above is sum-square convergent for all $f \in \mathcal{S}_\beta$ if:

- $\eta < \beta^{-1}$
- $\mathbf{v}$ is a bounded-energy ℓ_2 signal, that is, $\sum_{t=0}^{\infty} ||v_t||^2 \leq \infty$

[1] Andrea Martin, LF, "Learning to optimize with convergence guarantees using nonlinear system theory", IEEE Control System Letters, 2024, to appear

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \eta \nabla f(x_t) + v_t$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \boxed{\eta \nabla f(x_t)} + \boxed{v_t}$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \eta \nabla f(x_t) + v_t$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^\mathsf{T}(\underbrace{x_{t+1} - x_t}_{-\eta\nabla_t + v_t}) + \frac{\beta}{2}|x_{t+1} - x_t|$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \boxed{\eta \nabla f(x_t)} + \boxed{v_t}$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^{\mathsf{T}}(\underbrace{x_{t+1} - x_t}_{=\,-\eta\nabla_t + v_t}) + \frac{\beta}{2}|x_{t+1} - x_t|$
- Highlight the role of $\mathbf{v}$: $\frac{\overbrace{2\eta\epsilon(1-\beta\eta)-1}^{>0}}{2\epsilon}|\nabla_t|^2 \leq f(x_t) - f(x_{t+1}) + \left(\frac{\epsilon}{2} + \beta\right)|v_t|^2 .$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \boxed{\eta \nabla f(x_t)} + \boxed{v_t}$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^\top (\overbrace{x_{t+1} - x_t}^{-\eta\nabla_t + v_t}) + \frac{\beta}{2}|x_{t+1} - x_t|$
- Highlight the role of $\mathbf{v}$: $\frac{\overbrace{2\eta\epsilon(1-\beta\eta)-1}^{>0}}{2\epsilon} |\nabla_t|^2 \leq f(x_t) - f(x_{t+1}) + \left(\frac{\epsilon}{2} + \beta\right) |v_t|^2 .$
- Summing over t: $\sum_{t=0}^{\infty} |\nabla_t|^2 \leq M \left(f(x_0) - \inf_{x\in\mathbb{R}^d} f(x)\right) + N \sum_{t=0}^{\infty} |v_t|^2 .$

$$\boxed{\implies ||\nabla f(\mathbf{x})||_2^2 \leq K_1 + K_2 ||\mathbf{v}||_2^2 \leq \infty}$$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \boxed{\eta \nabla f(x_t)} + \boxed{v_t}$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^\top (\overset{-\eta\nabla_t + v_t}{\overline{x_{t+1} - x_t}}) + \frac{\beta}{2}|x_{t+1} - x_t|$
- Highlight the role of $\mathbf{v}$: $\frac{\overbrace{2\eta\epsilon(1-\beta\eta)-1}^{>0}}{2\epsilon} |\nabla_t|^2 \leq f(x_t) - f(x_{t+1}) + \left(\frac{\epsilon}{2} + \beta\right) |v_t|^2 .$
- Summing over t: $\sum_{t=0}^{\infty} |\nabla_t|^2 \leq M\left(f(x_0) - \inf_{x\in\mathbb{R}^d} f(x)\right) + N \sum_{t=0}^{\infty} |v_t|^2 .$

$$\boxed{\implies ||\nabla f(\mathbf{x})||_2^2 \leq K_1 + K_2 ||\mathbf{v}||_2^2 \leq \infty}$$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \boxed{\eta \nabla f(x_t)} + \boxed{v_t}$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^\top (\overbrace{x_{t+1} - x_t}^{-\eta\nabla_t + v_t}) + \frac{\beta}{2}|x_{t+1} - x_t|$
- Highlight the role of $\mathbf{v}$: $\frac{\overbrace{2\eta\epsilon(1-\beta\eta)-1}^{>0}}{2\epsilon} |\nabla_t|^2 \leq f(x_t) - f(x_{t+1}) + \left(\frac{\epsilon}{2} + \beta\right)|v_t|^2 .$
- Summing over t: $\sum_{t=0}^{\infty} |\nabla_t|^2 \leq M\left(f(x_0) - \inf_{x\in\mathbb{R}^d} f(x)\right) + N\sum_{t=0}^{\infty} |v_t|^2 .$

$$\boxed{\Longrightarrow \ ||\nabla f(\mathbf{x})||_2^2 \leq K_1 + K_2||\mathbf{v}||_2^2 \leq \infty}$$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \boxed{\eta \nabla f(x_t)} + \boxed{v_t}$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^{\top}(\overbrace{x_{t+1} - x_t}^{-\eta\nabla_t + v_t}) + \frac{\beta}{2}|x_{t+1} - x_t|$
- Highlight the role of $\mathbf{v}$: $\frac{\overbrace{2\eta\epsilon(1-\beta\eta)-1}^{>0}}{2\epsilon} |\nabla_t|^2 \leq f(x_t) - f(x_{t+1}) + \left(\frac{\epsilon}{2} + \beta\right) |v_t|^2 .$
- Summing over t: $\sum_{t=0}^{\infty} |\nabla_t|^2 \leq M\left(f(x_0) - \inf_{x \in \mathbb{R}^d} f(x)\right) + N \sum_{t=0}^{\infty} |v_t|^2 .$

$$\implies ||\nabla f(\mathbf{x})||_2^2 \leq K_1 + K_2 ||\mathbf{v}||_2^2 \leq \infty$$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory*, IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \eta \nabla f(x_t) + v_t$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^{\mathsf{T}} (\underbrace{x_{t+1} - x_t}_{-\eta\nabla_t + v_t}) + \frac{\beta}{2} |x_{t+1} - x_t|$
- Highlight the role of $\mathbf{v}$: $\frac{\overbrace{2\eta\epsilon(1-\beta\eta)-1}^{>0}}{2\epsilon} |\nabla_t|^2 \leq f(x_t) - f(x_{t+1}) + \left(\frac{\epsilon}{2} + \beta\right) |v_t|^2.$
- Summing over t: $\sum_{t=0}^{\infty} |\nabla_t|^2 \leq M\left(f(x_0) - \inf_{x\in\mathbb{R}^d} f(x)\right) + N \sum_{t=0}^{\infty} |v_t|^2.$

$$\implies ||\nabla f(\mathbf{x})||_2^2 \leq K_1 + K_2 ||\mathbf{v}||_2^2 \leq \infty$$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory*, IEEE Control System Letters, 2024

Main result 1: sketch of the proof

- Nonlinear dynamics: $x_{t+1} = x_t - \boxed{\eta \nabla f(x_t)} + \boxed{v_t}$
- Exponential convergence if $v_t = 0$ … What if $\mathbf{v} \in \ell_2$?
  - **Nonlinear system theory**: check **Input-to-state (ISS) stability** property! [1]
  - Non-trivial: multiple equilibria, stronger than Bounder-Input-Bounded-Output…

Proof sketch[2]

- β- smoothness of the gradients: $f(x_{t+1}) \leq f(x_t) + \nabla_t^\top (\overset{-\eta\nabla_t + v_t}{\overline{x_{t+1} - x_t}}) + \frac{\beta}{2}|x_{t+1} - x_t|$
- Highlight the role of $\mathbf{v}$: $\frac{\overbrace{2\eta\epsilon(1-\beta\eta)-1}^{>0}}{2\epsilon} |\nabla_t|^2 \leq f(x_t) - f(x_{t+1}) + \left(\frac{\epsilon}{2} + \beta\right)|v_t|^2 .$
- Summing over t: $\sum_{t=0}^{\infty} |\nabla_t|^2 \leq M\left(f(x_0) - \inf_{x\in\mathbb{R}^d} f(x)\right) + N\sum_{t=0}^{\infty}|v_t|^2 .$

$$\boxed{\implies ||\nabla f(\mathbf{x})||_2^2 \leq K_1 + K_2||\mathbf{v}||_2^2 \leq \infty}$$

[1] Khalil, Hassan K. *Control of nonlinear systems*. Prentice Hall, New York, NY, 2002.
[2] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 2: completeness

(Crucial) question: can *any* algorithm that robustly converges on $\mathcal{S}_\beta$ be expressed as

$$\boldsymbol{\pi}(f, \mathbf{x}) = -\eta \nabla f(\mathbf{x}) + \mathbf{v} \quad ?$$

[1] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 2: completeness

(Crucial) question: can *any* algorithm that robustly converges on $\mathcal{S}_\beta$ be expressed as

$$\boldsymbol{\pi}(f, \mathbf{x}) = -\eta \nabla f(\mathbf{x}) + \mathbf{v} \quad ?$$

Result: Let $\boldsymbol{\pi}$ be any robustly convergent algorithm:

$$\boldsymbol{\pi}(f, \mathbf{x}) \in \Sigma(f), \ \forall f \in \mathcal{S}_\beta \, .$$

Then, there exists an $\mathcal{L}_2$ operator $\mathbf{V}$ such that the algorithm

$$\boldsymbol{\pi}'(f, \mathbf{x}) = -\eta \nabla f(\mathbf{x}) + \mathbf{V}(x_0)$$

yields the same trajectories as $\boldsymbol{\pi}$, for every x_0 and every $f \in \mathcal{S}_\beta$.

[1] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Main result 2: completeness

(Crucial) question: can *any* algorithm that robustly converges on $\mathcal{S}_\beta$ be expressed as

$$\boldsymbol{\pi}(f,\mathbf{x}) = -\eta\nabla f(\mathbf{x}) + \mathbf{v} \quad ?$$

Result: Let $\boldsymbol{\pi}$ be any robustly convergent algorithm:

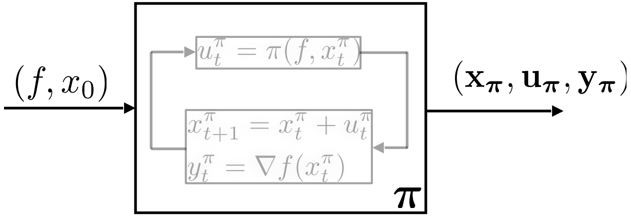
$$\boldsymbol{\pi}(f,\mathbf{x}) \in \Sigma(f),\ \forall f \in \mathcal{S}_\beta\,.$$

Then, there exists an $\mathcal{L}_2$ operator $\mathbf{V}$ such that the algorithm

$$\boldsymbol{\pi}'(f,\mathbf{x}) = -\eta\nabla f(\mathbf{x}) + \mathbf{V}\,(x_0)$$

yields the same trajectories as $\boldsymbol{\pi}$, for every x_0 and every $f \in \mathcal{S}_\beta$.

Proof sketch:



is an $\mathcal{L}_2$ operator.

Then $\mathbf{V}(x_0) = \eta\nabla f(\mathbf{x}_{\boldsymbol{\pi}}(x_0)) + \mathbf{u}_{\boldsymbol{\pi}}(x_0)$ achieves desired behavior, and also lies in $\mathcal{L}_2$

[1] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory,* IEEE Control System Letters, 2024

Importance of main results 1 and 2

One-to-one parametrization

$$-\eta\nabla f(\mathbf{x}) + \mathbf{V} \in \mathcal{L}_2 \equiv \boldsymbol{\pi}(f,\mathbf{x}) \in \Sigma(f),\ \forall f \in \mathcal{S}_\beta\,.$$

Importance of main results 1 and 2

One-to-one parametrization

$$-\eta \nabla f(\mathbf{x}) + \mathbf{V} \in \mathcal{L}_2 \equiv \boldsymbol{\pi}(f, \mathbf{x}) \in \Sigma(f), \ \forall f \in \mathcal{S}_\beta \,.$$

- Unconstrained learning over all converging algorithms

$$\min_{\mathbf{V} \in \mathcal{L}_2} \ \mathbb{E}_{f \sim \text{Examples}, x_0 \sim \mathcal{X}_0} \left[\texttt{AlgoPerf}(f, \mathbf{x}) \right] \quad \text{s.t.} \quad x_{t+1} = x_t - \eta \nabla f(x_t) + V_t(x_0),$$

$$\equiv$$

$$\min_{\boldsymbol{\pi}} \ \mathbb{E}_{f \sim \text{Examples}, x_0 \sim \mathcal{X}_0} \left[\texttt{AlgoPerf}(f, \mathbf{x}) \right] \quad \text{s.t.} \quad x_{t+1} = x_t + \pi_t(f, x_{t:0}), \quad \cancel{\boldsymbol{\pi}(f, \mathbf{x}) \in \Sigma(f), \ \forall f \in \mathcal{S}_\beta} \,,$$

EPFL

Importance of main results 1 and 2

One-to-one parametrization

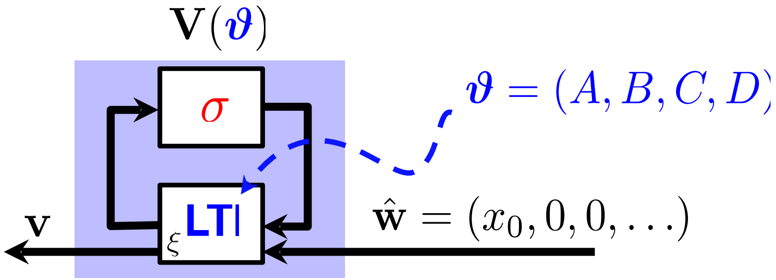
$$-\eta\nabla f(\mathbf{x}) + \mathbf{V} \in \mathcal{L}_2 \equiv \boldsymbol{\pi}(f,\mathbf{x}) \in \Sigma(f),\ \forall f \in \mathcal{S}_\beta\,.$$

- Unconstrained learning over all converging algorithms

$$\begin{aligned}\min_{\mathbf{V}\in\mathcal{L}_2}\ & \mathbb{E}_{f\sim \text{Examples}, x_0\sim\mathcal{X}_0}[\texttt{AlgoPerf}(f,\mathbf{x})]\\ \text{s.t.}\ & x_{t+1} = x_t - \eta\nabla f(x_t) + V_t(x_0),\end{aligned} \equiv \begin{aligned}\min_{\boldsymbol{\pi}}\ & \mathbb{E}_{f\sim \text{Examples}, x_0\sim\mathcal{X}_0}[\texttt{AlgoPerf}(f,\mathbf{x})]\\ \text{s.t.}\ & x_{t+1} = x_t + \pi_t(f, x_{t:0}),\\ & \boldsymbol{\pi}(f,\mathbf{x}) \in \Sigma(f),\ \forall f \in \mathcal{S}_\beta\,,\end{aligned}$$

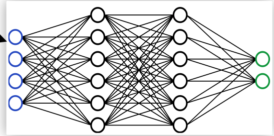
How do we search over these operators?

Unconstrained parametrizations of $\mathbf{V} \in \mathcal{L}_2$



- Expressive models including

$$\xi_t = \hat{A}\xi_{t-1} + \hat{B}\,\mathrm{NN}(\xi_{t-1}, \hat{\mathbf{w}}_t)$$
$$u_t = \hat{C}\xi_t + \hat{D}\,\mathrm{NN}(\xi_t, \hat{\mathbf{w}}_t)$$

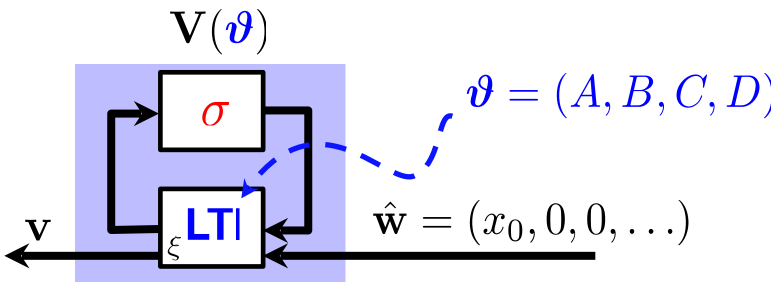


- $\mathbf{V}(\vartheta) \in \mathcal{L}_2$ if there is a storage function $S(\xi) = \xi^\mathsf{T} P \xi$ verifying

$$S(\xi_{t+1}) - S(\xi_t) \leq \gamma^2 |x_0| - |u_t|$$

[1] Kim, K. K., E. Ríos Patrón, and R. D. Braatz. *Standard representation and unified stability analysis for dynamic artificial neural network models,* Transactions on Neural Networks, 2018
[2] Revay, M, R. Wang, and I.R. Manchester. *Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness*. IEEE TAC 2023

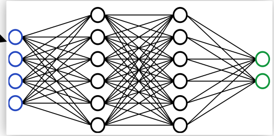
Unconstrained parametrizations of $\mathbf{V} \in \mathcal{L}_2$



- Expressive models including

$$\xi_t = \hat{A}\xi_{t-1} + \hat{B}\,\mathrm{NN}(\xi_{t-1}, \hat{\mathbf{w}}_t)$$

$$u_t = \hat{C}\xi_t + \hat{D}\,\mathrm{NN}(\xi_t, \hat{\mathbf{w}}_t)$$



- $\mathbf{V}(\vartheta) \in \mathcal{L}_2$ if there is a storage function $S(\xi) = \xi^\top P \xi$ verifying

$$S(\xi_{t+1}) - S(\xi_t) \leq \gamma^2 |x_0| - |u_t|$$

Free parametrization[2]: explicit map $\boldsymbol{\Theta} \rightarrow (\vartheta, P)$ such that $\mathbf{V}(\vartheta) \in \mathcal{L}_2$, for any $\boldsymbol{\Theta} \in \mathbb{R}^d$

[1] Kim, K. K., E. Ríos Patrón, and R. D. Braatz. *Standard representation and unified stability analysis for dynamic artificial neural network models,* Transactions on Neural Networks, 2018

[2] Revay, M, R. Wang, and I.R. Manchester. *Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness*. IEEE TAC 2023

Unconstrained parametrizations of $\mathbf{V} \in \mathcal{L}_2$

Free parametrization[2]: explicit map $\boldsymbol{\Theta} \rightarrow (\boldsymbol{\vartheta}, P)$ such that $\mathbf{V}(\boldsymbol{\vartheta}) \in \mathcal{L}_2$, for any $\boldsymbol{\Theta} \in \mathbb{R}^d$

$$\min_{\boldsymbol{\Theta} \in \mathbb{R}^d} \quad \mathbb{E}_{f \sim \text{Examples}, x_0 \sim \mathcal{X}_0} \left[\texttt{AlgoPerf}(f, \mathbf{x}) \right]$$

$$\text{s.t.} \quad x_{t+1} = x_t - \eta \nabla f(x_t) + V_t(x_0, \boldsymbol{\Theta})$$

[1] Kim, K. K., E. Ríos Patrón, and R. D. Braatz. *Standard representation and unified stability analysis for dynamic artificial neural network models,* Transactions on Neural Networks, 2018

[2] Revay, M, R. Wang, and I.R. Manchester. *Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness*. IEEE TAC 2023

Unconstrained parametrizations of $\mathbf{V} \in \mathcal{L}_2$

Free parametrization[2]: explicit map $\boldsymbol{\Theta} \rightarrow (\boldsymbol{\vartheta}, P)$ such that $\mathbf{V}(\boldsymbol{\vartheta}) \in \mathcal{L}_2$, for any $\boldsymbol{\Theta} \in \mathbb{R}^d$

⇩

$$\min_{\boldsymbol{\Theta} \in \mathbb{R}^d} \quad \mathbb{E}_{f \sim \text{Examples}, x_0 \sim \mathcal{X}_0} \left[\texttt{AlgoPerf}(f, \mathbf{x})\right]$$

$$\text{s.t.} \quad x_{t+1} = x_t - \eta \nabla f(x_t) + V_t(x_0, \boldsymbol{\Theta})$$

Challenges for effective L2O

1. **Insufficient features**
   - Learning to map $x_0 \in \mathbb{R}^d \rightarrow \mathbf{v}^\star \in \mathbb{R}^\infty$ is hard
2. **Full gradient measurements** for convergence

[1] Kim, K. K., E. Ríos Patrón, and R. D. Braatz. S… *network models,* Transactions on Neural Networks…
[2] Revay, M, R. Wang, and I.R. Manchester. *Recu… …dynamic models with guaranteed stability and robustness*. IEEE TAC 2023

1) Enriching the input features with $[I, f, \nabla f](x_t)$

- Parametrize additional signal with rich input features (**not in** ℓ_2):

$$\boldsymbol{\omega}(\boldsymbol{\varphi}) = \boldsymbol{\Omega}(\mathbf{x}, f(\mathbf{x}), \nabla f(\mathbf{x}), \boldsymbol{\varphi})$$

- Construct a combined signal $\mathbf{z}(\boldsymbol{\varphi}, \boldsymbol{\Theta})$ <u>which is ℓ_2 by construction</u>:

$$z_t(f, \nabla f, x_{t:0}, \boldsymbol{\varphi}, \boldsymbol{\Theta}) = \frac{\omega_t(f, \nabla f, x_{t:0}, \boldsymbol{\varphi})}{|\omega_t(f, \nabla f, x_{t:0}, \boldsymbol{\varphi})|} |v_t(x_0, \boldsymbol{\Theta})|$$

1) Enriching the input features with $[I, f, \nabla f](x_t)$

- Parametrize additional signal with rich input features (**not in** ℓ_2):

$$\boldsymbol{\omega}(\boldsymbol{\varphi}) = \boldsymbol{\Omega}(\mathbf{x}, f(\mathbf{x}), \nabla f(\mathbf{x}), \boldsymbol{\varphi})$$

- Construct a combined signal $\mathbf{z}(\boldsymbol{\varphi}, \boldsymbol{\Theta})$ <u>which is ℓ_2 by construction</u>:

$$z_t(f, \nabla f, x_{t:0}, \boldsymbol{\varphi}, \boldsymbol{\Theta}) = \frac{\omega_t(f, \nabla f, x_{t:0}, \boldsymbol{\varphi})}{|\omega_t(f, \nabla f, x_{t:0}, \boldsymbol{\varphi})|} |v_t(x_0, \boldsymbol{\Theta})|$$

Preserves one-to-one parametrization!

$$-\eta \nabla f(\mathbf{x}) + \frac{\boldsymbol{\Omega}(f, \nabla f, \mathbf{x})}{|\boldsymbol{\Omega}(f, \nabla f, \mathbf{x})|} |\mathbf{V}(x_0)| \equiv \boldsymbol{\pi}(f, \mathbf{x}) \in \Sigma(f), \ \forall f \in \mathcal{S}_\beta \,.$$

Proof of sufficiency: by construction

Proof of necessity: reduces to previous case by choosing $\boldsymbol{\Omega}(f, \nabla f, \mathbf{x}) = \mathbf{V}(x_0)$

2) Convergence with incomplete gradients

- Typical scenario in machine learning

$$f(x) = \sum_{i \in \text{batches}} f_i(x) \quad \Rightarrow \quad \underline{\text{access to } \nabla f_i(x) \text{ only}}$$

[1] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory*, IEEE Control System Letters, 2024

2) Convergence with incomplete gradients

- Typical scenario in machine learning

$$f(x) = \sum_{i \in \text{batches}} f_i(x) \quad \Rightarrow \quad \underline{\text{access to } \nabla f_i(x) \text{ only}}$$

Result: Convergence (*asymptotic*) is preserved using the update rule

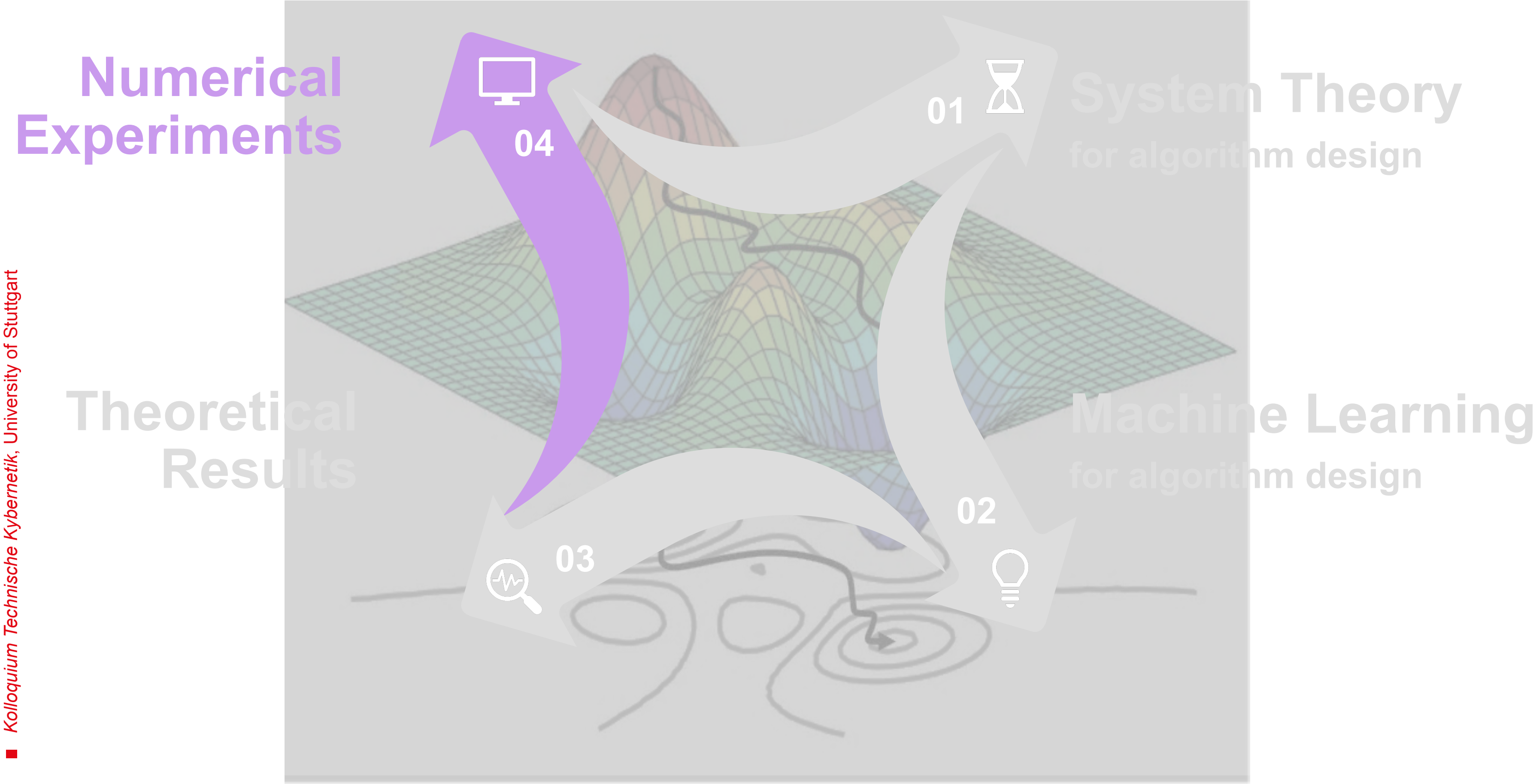
$$\boldsymbol{\pi}_t(x_t) = -\eta_t \nabla f_t(x_t) + v_t$$

as long as $\boldsymbol{\eta} \in \ell_2$ and $|v_t| \leq C\eta_t |\nabla f(x_t)|$.

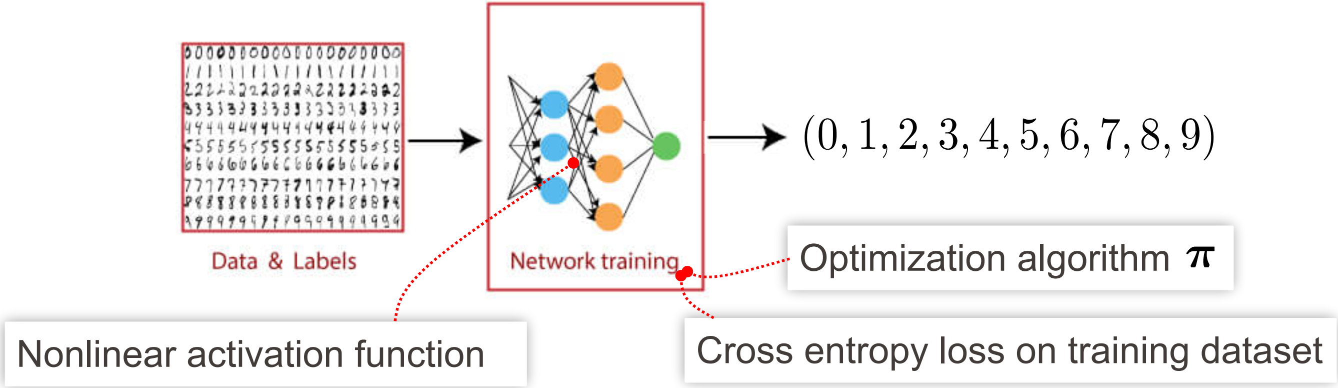
- **Convergence** guarantees of L2O algorithms **compatible with ML tasks**!
- **Only sufficient, only asymptotic convergence**
  - *necessity* relies on full gradient information
  - *asymptotic convergence*: consistent with stochastic gradient descent results

[1] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory*, IEEE Control System Letters, 2024

Design of new optimization algorithms

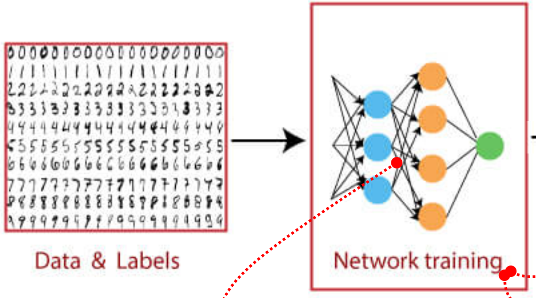


Numerical experiments: image classification[1]



[1] LeCun, Y., Cortes, C. and Burges, C.J.C. (1998) The MNIST Database of Handwritten Digits. New York, USA. http://yann.lecun.com/exdb/mnist/

Numerical experiments: image classification[1]



$\longrightarrow (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)$

Optimization algorithm π

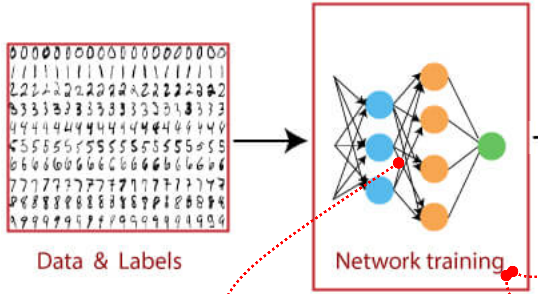
Nonlinear activation function

Cross entropy loss on training dataset

- **Algorithm training**: repeatedly optimize from uniformly distributed initial weights
  - $\mathrm{AlgoPerf}$: average algorithm performance (speed and solution quality)
  - We fix $\tanh$ activation function on

[1] LeCun, Y., Cortes, C. and Burges, C.J.C. (1998) The MNIST Database of Handwritten Digits. New York, USA. http://yann.lecun.com/exdb/mnist/

Numerical experiments: image classification[1]

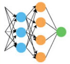


$$(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)$$

Optimization algorithm π

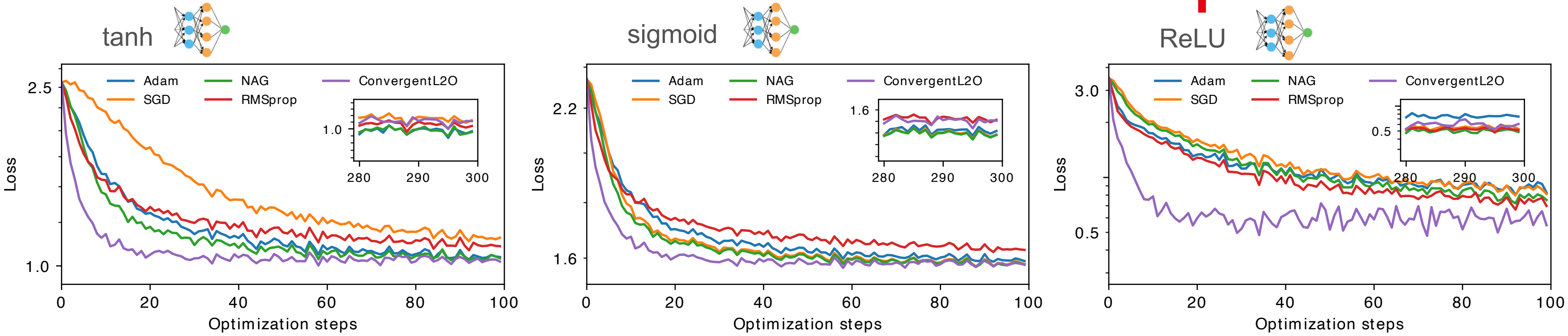
Nonlinear activation function

Cross entropy loss on training dataset

- **Algorithm training**: repeatedly optimize  from uniformly distributed initial weights
  - $\mathrm{AlgoPerf}$: average algorithm performance (speed and solution quality)
  - We fix $\tanh$ activation function on 
- **Algorithm testing**: compare with classical fine-tuned optimizers
  - Generalization to different activation functions on 
  - Convergence guaranteed by design

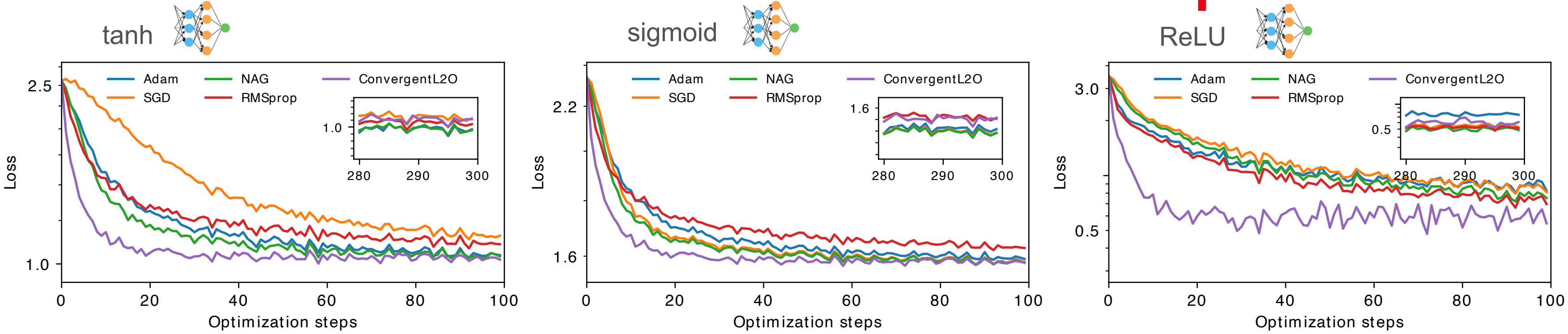
[1] LeCun, Y., Cortes, C. and Burges, C.J.C. (1998) The MNIST Database of Handwritten Digits. New York, USA. http://yann.lecun.com/exdb/mnist/

Numerical Experiments



[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NIPS 2016

Numerical Experiments



Classification accuracy on test data

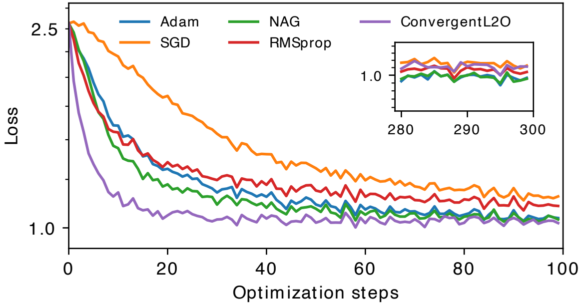
Step $t = 20$	tanh	sigmoid	ReLU
Adam	$71.7 \pm 5.1\%$	$76.1 \pm 3.1\%$	$52.7 \pm 11.1\%$
SGD	$44.9 \pm 4.2\%$	$79.7 \pm 1.9\%$	$49.8 \pm 9.3\%$
NAG	$79.7 \pm 1.4\%$	$81.1 \pm 1.5\%$	$52.7 \pm 10.2\%$
RMSprop	$69.4 \pm 2.9\%$	$72.8 \pm 2.3\%$	$61.1 \pm 8.9\%$
ConvergentL2O	**$87.0 \pm 0.5\%$**	**$86.8 \pm 0.6\%$**	$86.3 \pm 0.6\%$
LSTM	$82.2 \pm 0.1\%$	$83.3 \pm 0.1\%$	**$88.3 \pm 0.0\%$**

Step $t = 300$	tanh	sigmoid	ReLU
Adam	**$89.5 \pm 0.5\%$**	**$89.6 \pm 0.3\%$**	$70.3 \pm 12.2\%$
SGD	$87.4 \pm 0.4\%$	$89.3 \pm 0.3\%$	$80.6 \pm 8.1\%$
NAG	$89.4 \pm 0.2\%$	$89.4 \pm 0.2\%$	$82.2 \pm 7.6\%$
RMSprop	$87.6 \pm 2.1\%$	$88.5 \pm 0.4\%$	$81.5 \pm 7.5\%$
ConvergentL2O	$88.5 \pm 0.2\%$	$88.4 \pm 0.3\%$	$87.7 \pm 0.2\%$
LSTM	$81.4 \pm 0.0\%$	$81.4 \pm 0.0\%$	**$88.3 \pm 0.0\%$**

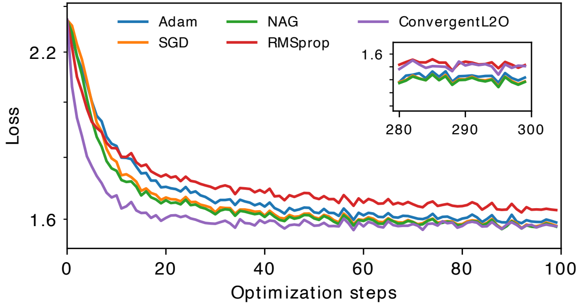
[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NIPS 2016

Numerical Experiments

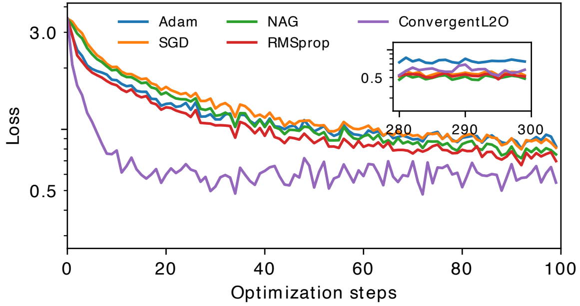
tanh



sigmoid



ReLU

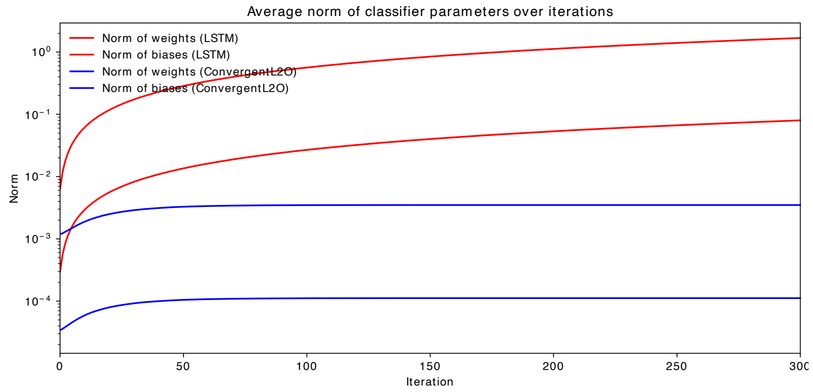


Classification accuracy on test data

Step $t = 20$	tanh	sigmoid	ReLU
Adam	$71.7 \pm 5.1\%$	$76.1 \pm 3.1\%$	$52.7 \pm 11.1\%$
SGD	$44.9 \pm 4.2\%$	$79.7 \pm 1.9\%$	$49.8 \pm 9.3\%$
NAG	$79.7 \pm 1.4\%$	$81.1 \pm 1.5\%$	$52.7 \pm 10.2\%$
RMSprop	$69.4 \pm 2.9\%$	$72.8 \pm 2.3\%$	$61.1 \pm 8.9\%$
ConvergentL2O	**$87.0 \pm 0.5\%$**	**$86.8 \pm 0.6\%$**	$86.3 \pm 0.6\%$
LSTM	$82.2 \pm 0.1\%$	$83.3 \pm 0.1\%$	**$88.3 \pm 0.0\%$**

Step $t = 300$	tanh	sigmoid	ReLU
Adam	**$89.5 \pm 0.5\%$**	**$89.6 \pm 0.3\%$**	$70.3 \pm 12.2\%$
SGD	$87.4 \pm 0.4\%$	$89.3 \pm 0.3\%$	$80.6 \pm 8.1\%$
NAG	$89.4 \pm 0.2\%$	$89.4 \pm 0.2\%$	$82.2 \pm 7.6\%$
RMSprop	$87.6 \pm 2.1\%$	$88.5 \pm 0.4\%$	$81.5 \pm 7.5\%$
ConvergentL2O	$88.5 \pm 0.2\%$	$88.4 \pm 0.3\%$	$87.7 \pm 0.2\%$
LSTM	$81.4 \pm 0.0\%$	$81.4 \pm 0.0\%$	**$88.3 \pm 0.0\%$**

LSTM from [1] diverges!



[1] Andrychowicz, Marcin, et al. "Learning to learn by gradient descent by gradient descent." NIPS 2016

Conclusions

- **Automated synthesis of high-performance optimization algorithms**
  - With theoretical guarantees by design
  - For non-convex functions

Nonlinear system theory

L2O

- **What's next?**
  - *Analyzing generalization capabilities (robust performance)*
  - Online and distributed optimization
  - Stronger convergence guarantees for classes of non-convex functions
  - Applications in optimal control, e.g., NMPC

Conclusions

- **Automated synthesis of high-performance optimization algorithms**
  - With theoretical guarantees by design
  - For non-convex functions

Nonlinear system theory

L2O

- **What's next?**
  - *Analyzing generalization capabilities (robust performance)*
  - Online and distributed optimization
  - Stronger convergence guarantees for classes of non-convex functions
  - Applications in optimal control, e.g., NMPC

Check out the paper:

Thank you! Questions?

$$\min_{\mathbf{\Omega}, \mathbf{V} \in \mathcal{L}_2} \quad \mathbb{E}_{f \sim \text{Examples}, x_0 \sim \mathcal{X}_0} \left[\texttt{AlgoPerf}(f, \mathbf{x}) \right]$$

$$\text{s.t.} \quad x_{t+1} = x_t - \eta \nabla f(x_t) + \frac{\omega_t(f, \nabla f, x_t)}{|\omega_t(f, \nabla, x_t)|} |v_t(x_0)|,$$

[1] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory*, IEEE Control System Letters, 2024

Thank you! Questions?

Check out the paper:

$$\min_{\boldsymbol{\Omega}, \mathbf{V} \in \mathcal{L}_2} \mathbb{E}_{f\sim \text{Examples}, x_0\sim\mathcal{X}_0}\left[\texttt{AlgoPerf}(f,\mathbf{x})\right]$$
$$\text{s.t.} \quad x_{t+1} = x_t - \eta\nabla f(x_t) + \frac{\omega_t(f,\nabla f, x_t)}{|\omega_t(f,\nabla, x_t)|}|v_t(x_0)|,$$

$$\equiv$$

$$\min_{\boldsymbol{\pi}} \mathbb{E}_{f\sim \text{Examples}, x_0\sim\mathcal{X}_0}\left[\texttt{AlgoPerf}(f,\mathbf{x})\right]$$
$$\text{s.t.} \quad x_{t+1} = x_t + \pi_t(f, x_{t:0}),$$
$$\boldsymbol{\pi}(f,\mathbf{x}) \in \Sigma(f), \ \forall f \in \mathcal{S}_\beta,$$

[1] Andrea Martin and Luca Furieri, *Learning to optimize with convergence guarantees using nonlinear system theory*, IEEE Control System Letters, 2024