EPFL

A Compositional port-Hamiltonian Approach of Distributed Neural Network Optimal Control with Stability Guarantees

Preprint available

Luca Furieri

Clara Galimberti

Muhammad Zakwan

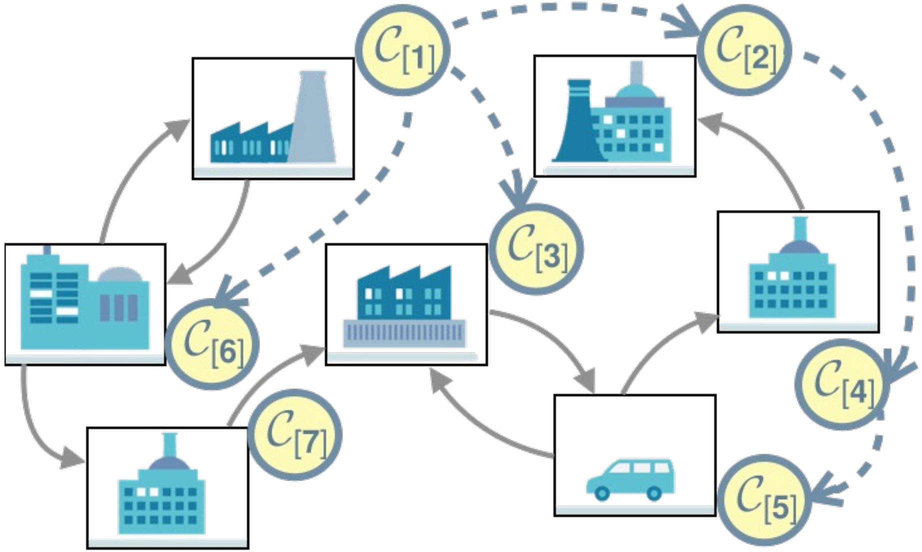
Giancarlo Ferrari Trecate

DECODE group, EPFL

Workshop “Trends on Dissipativity in Systems and Control”, 24.05.2022

Optimal distributed control

- Large scale systems + Optimal distributed control



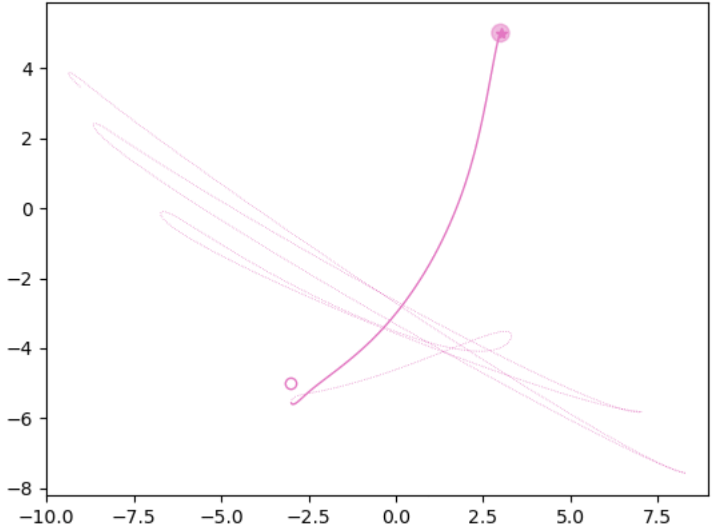
- **Challenge:** non-linear control policies are required
  - Linear systems and quadratic cost (Witsenhausen counterexample)
  - Non-linear systems and/or non-quadratic cost

Deep Neural Networks (DNNs) for parametrizing non-linear policies

Clara Galimberti

Challenges of using DNN policies

- Closed-loop stability guarantees while optimizing transient performance?
  - Not with general multilayer perceptron networks



- Performance optimization over *[0,T]*

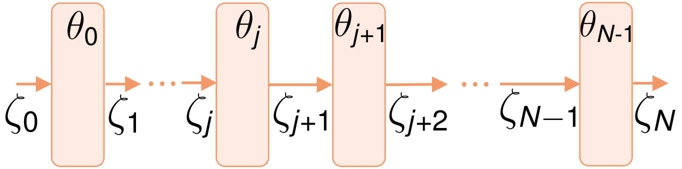
$$\min \int_0^T ||x - x^\star||^2 + ||u||^2 dt$$

s.t. MLP-controller

- Stability if controller applied for $t > T$?

- Stability if optimization stops prematurely?

Challenges of using DNN policies

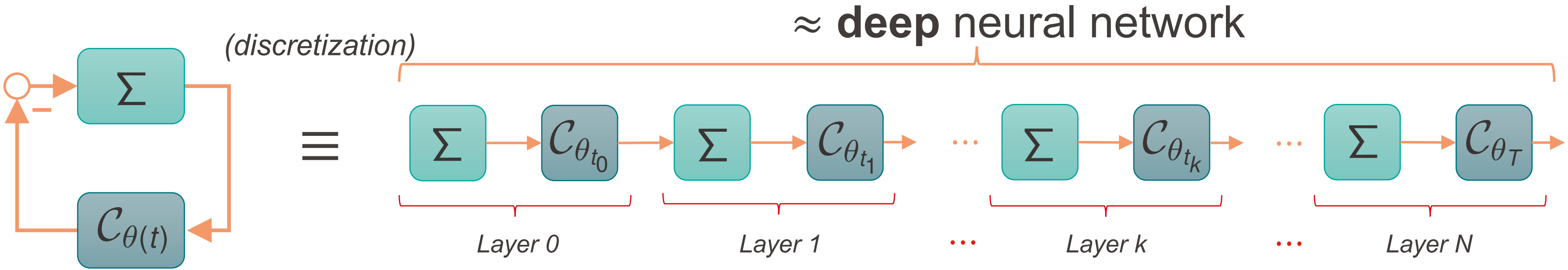


- Vanishing gradients during optimization
  - Backpropagation → Gradient descent

$$\frac{\partial \mathcal{L}}{\partial \theta_{i,j}} = \frac{\partial \zeta_{j+1}}{\partial \theta_{i,j}} \underbrace{\prod_{\ell=j+1}^{N-1} \frac{\partial \zeta_{\ell+1}}{\partial \zeta_{\ell}}}_{\text{Backward sensitivity matrix (BSM)}} \frac{\partial \mathcal{L}}{\partial \zeta_N}$$

  - If $\frac{\partial \mathcal{L}}{\partial \theta_{i,j}} \approx 0 \rightarrow$ Local minima? ☺ / BSM small? ☹

- Optimization for a long control horizon $\equiv$ optimization of a DNN



Challenges of using DNN policies

- Distributed NN control architectures

Graph Neural Networks for Distributed Linear-Quadratic Control

Fernando Gama* FGAMA@BERKELEY.EDU and **Somayeh Sojoudi** SOJOUDI@BERKELEY.EDU
Electrical Engineering and Computer Sciences Dept., University of California, Berkeley, CA 94709, USA

The linear-quadratic co
solution is a linear cont

Communication Topology Co-Design in Graph Recurrent Neural Network based Distributed Control

Fengjun Yang† and Nikolai Matni*

Abstract—**When designing large-scale distributed con-** enjoy approximation guarantees, see for example [1]–[4

  - *A posteriori* analysis of the closed-loop stability

- **Question:** Distributed NN controllers guaranteeing closed-loop stability *by design*?

Our contributions

For port-Hamiltonian systems:

- NN model-based controllers guaranteeing closed-loop stability
  - Optimization of an arbitrary cost over a finite horizon

$$\min_{\theta(t)} \quad \int_0^T \ell(\mathbf{x}(t), \mathbf{u}(t), \theta(t))\, dt$$
$$\text{s.t.} \quad \text{closed-loop stability}$$

- Additional requirement:
  Non-vanishing gradients during optimization

- … even in a distributed setting!

Outline

- Optimal control problem with pH-NN controllers
  - Port Hamiltonian systems
  - pH-NN controller architecture
- Distributed implementations of pH-NN controllers
- Numerical validations
  - pH-NN controllers for robots navigation task
- Conclusions

Outline

- **Optimal control problem with pH-NN controllers**
  - **Port Hamiltonian systems**
  - **pH-NN controller architecture**
- Distributed implementations of pH-NN controllers
- Numerical validations
  - pH-NN controllers for robots navigation task
- Conclusions

EPFL

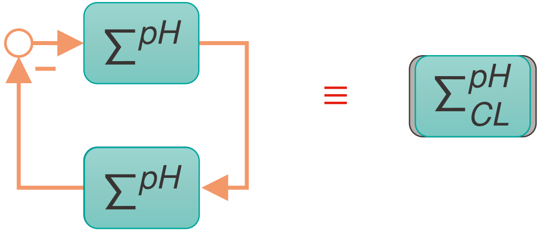
Port-Hamiltonian (pH) systems[1]

$$\dot{\mathbf{x}}(t) = (\boldsymbol{\Omega} - \mathbf{R})\frac{\partial V(\mathbf{x}(t))}{\partial \mathbf{x}} + \mathbf{G}^{\top}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{G}\frac{\partial V(\mathbf{x}(t))}{\partial \mathbf{x}}$$

- $\boldsymbol{\Omega}$ skew-symmetric
- $\mathbf{R} \succeq 0$

- V: Hamiltonian function
  - Continuously differentiable
  - Radially unbounded
- Attractivity: $\lim_{t\to\infty} \mathbf{x}(t) \in \left\{ \mathbf{R}\frac{\partial V}{\partial \mathbf{x}} = 0 \right\}$
- Compositionality:



[1] A. van der Schaft and D. Jeltsema. "Port-Hamiltonian systems theory: An introductory overview." *Foundations and Trends in Systems and Control* 1.2-3 (2014): 173-378.

Port-Hamiltonian (pH) systems[1]

$$\dot{\mathbf{x}}(t) = (\boldsymbol{\Omega} - \mathbf{R})\frac{\partial V(\mathbf{x}(t))}{\partial \mathbf{x}} + \mathbf{G}^{\top}\mathbf{u}(t)$$

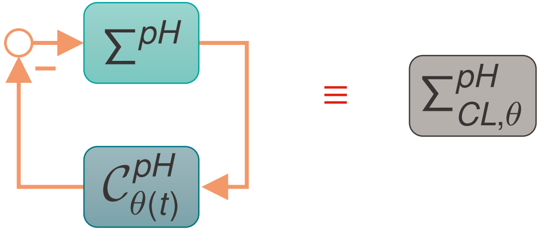
$$\mathbf{y}(t) = \mathbf{G}\frac{\partial V(\mathbf{x}(t))}{\partial \mathbf{x}}$$

- $\boldsymbol{\Omega}$ skew-symmetric
- $\mathbf{R} \succeq 0$

- V: Hamiltonian function
  - Continuously differentiable
  - Radially unbounded

- Attractivity: $\lim_{t\to\infty} \mathbf{x}(t) \in \left\{ \mathbf{R}\frac{\partial V}{\partial \mathbf{x}} = 0 \right\}$

- Compositionality:



[1] A. van der Schaft and D. Jeltsema. "Port-Hamiltonian systems theory: An introductory overview." *Foundations and Trends in Systems and Control* 1.2-3 (2014): 173-378.

NN controllers

$$\dot{\xi}(t) = (\mathbf{J}_c - \mathbf{R}_c)\frac{\partial \Phi(\xi(t), t)}{\partial \xi} + \mathbf{G}_c{}^\top \mathbf{y}(t)$$

$$\mathbf{u}(t) = \mathbf{G}_c \frac{\partial \Phi(\xi(t), t)}{\partial \xi}$$

- $\mathbf{J}_c$ skew-symmetric
- $\mathbf{R}_c \succeq 0$

- Elements in green are free to be chosen!
- Φ: time-varying Hamiltonian function
  - Continuously differentiable
  - Radially unbounded

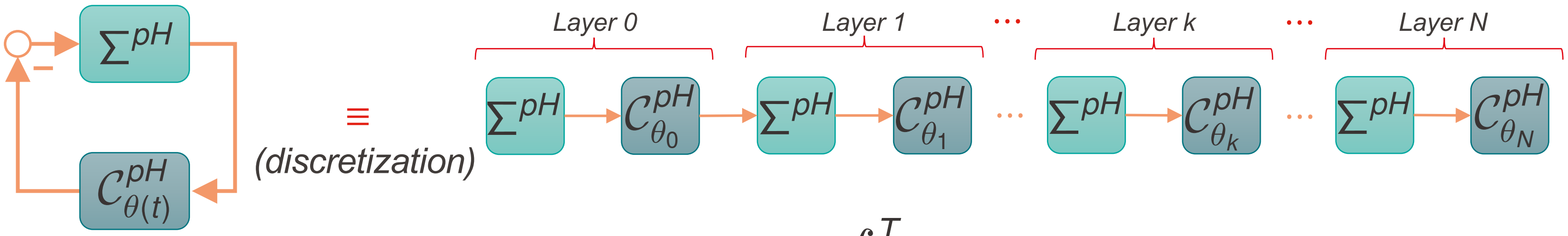
Use a (deep) NN for parametrizing Φ

$$\Phi(\xi(t), t) = \Phi(\xi(t), \theta_\phi(t))$$

- pH NN controllers $\subset$ NeuralODEs[1]

[1] R.T. Chen, Y. Rubanova, J. Bettencourt, and D.K. Duvenaud, “Neural ordinary differential equations”, *Advances in neural information processing systems*, vol *31*, 2018.

NN controllers



- Choose θ minimizing the cost: $\mathcal{L} = \int_0^T \ell(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\theta}(t))\, dt$
- Analogous to NN training!
  - Number of layers: N
  - Discretization step size: $h = T/N$

$$\mathcal{L}_{DT} = \sum_{k=0}^{N} \ell_{DT}(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\theta}_k)$$

- Closed-loop stability → For an arbitrary $T > 0$, i.e. arbitrary network depth

  → When θ^* is a local minima

  … but also before convergence! → I.e. for θ such that $\nabla_\theta \mathcal{L} \neq 0$

Non-vanishing gradients during training

Theorem[1]**.** If no dissipation in the loop, then

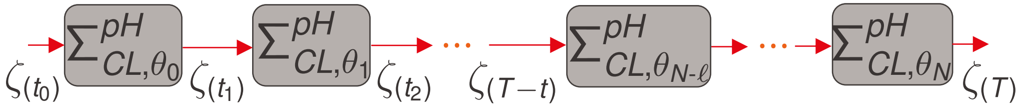
$$\left\| \frac{\partial \zeta(T)}{\partial \zeta(T-t)} \right\| \geq 1$$

$$\zeta = \begin{bmatrix} \text{system state} \\ \text{controller state} \end{bmatrix}$$

- Why?

$$\frac{\partial \zeta(T)}{\partial \zeta(T-t)} \text{ is symplectic, i.e. } \boldsymbol{\Psi} = \left(\frac{\partial \zeta(T)}{\partial \zeta(T-t)} \right)^\top \boldsymbol{\Psi} \frac{\partial \zeta(T)}{\partial \zeta(T-t)}$$

Interconnection matrix of the closed-loop system



[1] L. Furieri, C. Galimberti, M. Zakwan and G. Ferrari Trecate, “Distributed neural network control with dependability guarantees: a compositional port-Hamiltonian approach”, *arXiv preprint arXiv:2112.09046*, 2021.

Outline

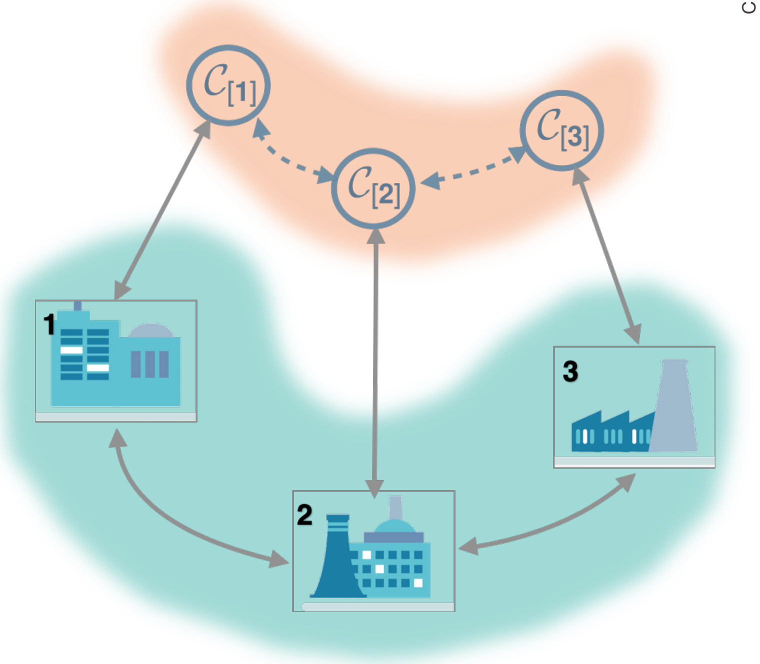
- Optimal control problem with pH-NN controllers
  - Port Hamiltonian systems
  - pH-NN controller architecture
- **Distributed implementations of pH-NN controllers**
- Numerical validations
  - pH-NN controllers for robots navigation task
- Conclusions

Distributed pH NN controllers

- : Network of pH systems
  - power-preserving interconnections
-  : Network of pH controllers?
  - Local energy: $\Phi_i(\xi_i, \text{neighbors}(\xi_i))$
    e.g. $\Phi_1(\xi_1, \xi_2)$
  - pH if $\dot{\xi}_i$ depends on $\frac{\partial \Phi_i}{\partial \xi_i}$ only?

↓

NO



Distributed pH NN controllers

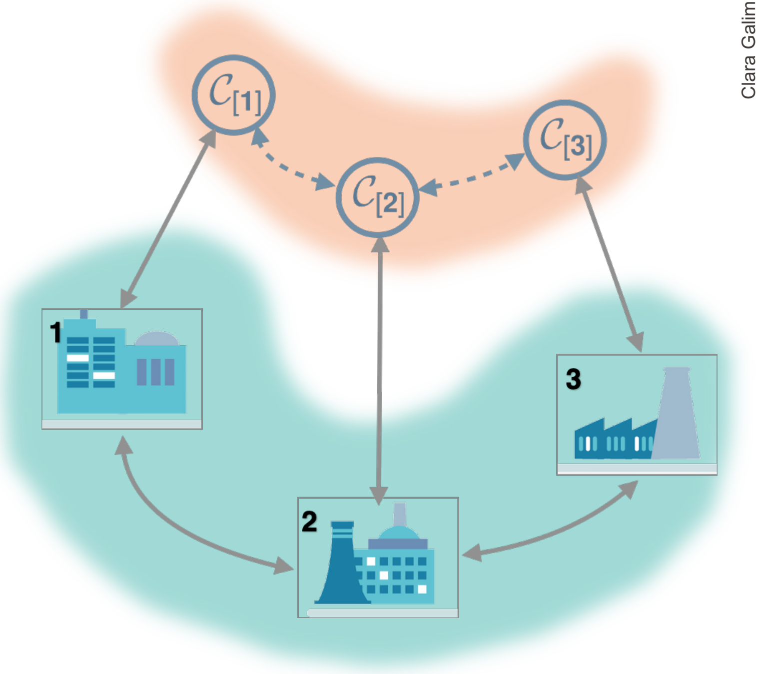
- Solution:
  - Define a global energy $\Phi = \sum \Phi_i$
  - Make $\dot{\xi}_i$ depend on $\frac{\partial \Phi}{\partial \xi_i}$

- Communication requirements? (set $i = 1$)

$$\frac{\partial \Phi}{\partial \xi_1} = \frac{\partial \Phi_1(\xi_1, \xi_2)}{\partial \xi_1} + \frac{\partial \Phi_2(\xi_1, \xi_2, \xi_3)}{\partial \xi_1}$$

ξ_3 is needed in location 1!

The given network is not enough!



- **Problem:** on which controller states should Φ_i depend upon?
  - for the prescribed communication network
  - while guaranteeing to be pH

Distributed pH NN controllers

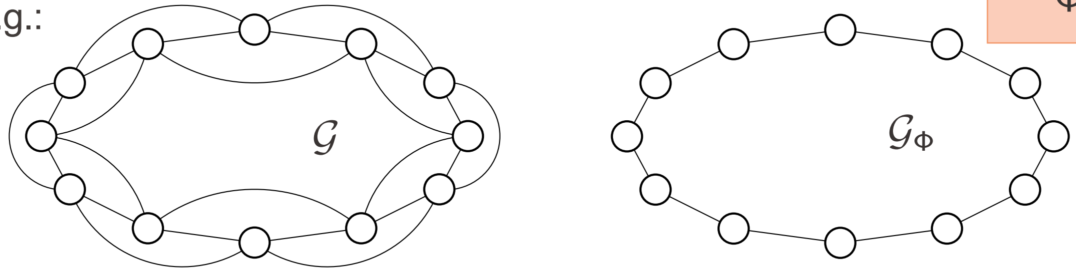
- $\mathcal{G}$ → prescribed communication network
- $\mathcal{G}_\Phi$ → which local energy depends upon which state

Theorem[1]**.** Let $\mathcal{G}_\phi$ be the communication graph describing the state dependencies of the local energies. Then, the NN control policies are distributed according to a prescribed interconnection network $\mathcal{G}$ if

$$\mathcal{G}_\phi^2 \subseteq \mathcal{G}\,.$$

Trivial solution:
Local energies only depend on local states, i.e.
$\Phi_i(\xi_i)\,,\quad \forall i = 1, \dots, M$

E.g.:



- Extension including input-output coupling between controllers can be found in [1]

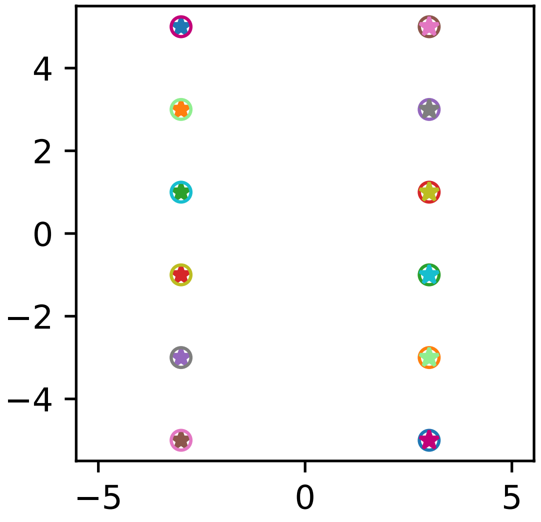
[1] L. Furieri, C. Galimberti, M. Zakwan and G. Ferrari Trecate, “Distributed neural network control with dependability guarantees: a compositional port-Hamiltonian approach”, *arXiv preprint arXiv:2112.09046*, 2021.

Outline

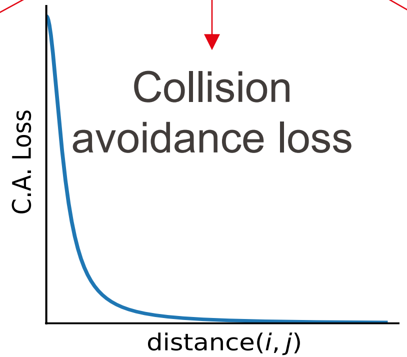
- Optimal control problem with pH-NN controllers
  - Port Hamiltonian systems
  - pH-NN controller architecture
- Distributed implementations of pH-NN controllers
- **Numerical validations**
  - **pH-NN controllers for robots navigation task**
- Conclusions

Navigation task using pH NN distributed controllers

- 12 mobile robots in *xy*-plane
  - Modelled by linear point masses
  - **Objective:** navigation ($\star \rightarrow \circ$) within a given time T **+** collision avoidance
  - Prestabilized dynamics around $\bar{\mathbf{x}}$ ($\circ$)
  - **Finite horizon stage cost:** $\ell_Q + \ell_{CA} + \ell_R$



Quadratic loss penalizing:
- Distance to target point
- Non zero velocity
- Input magnitude



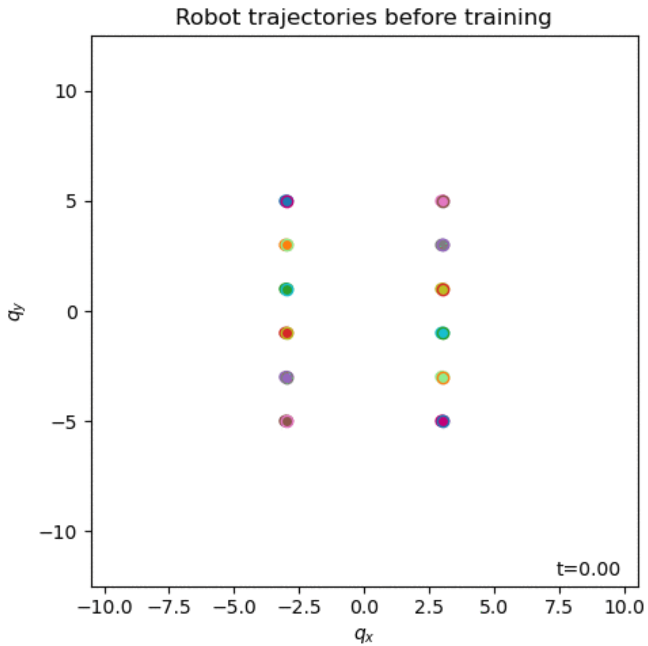
Collision avoidance loss

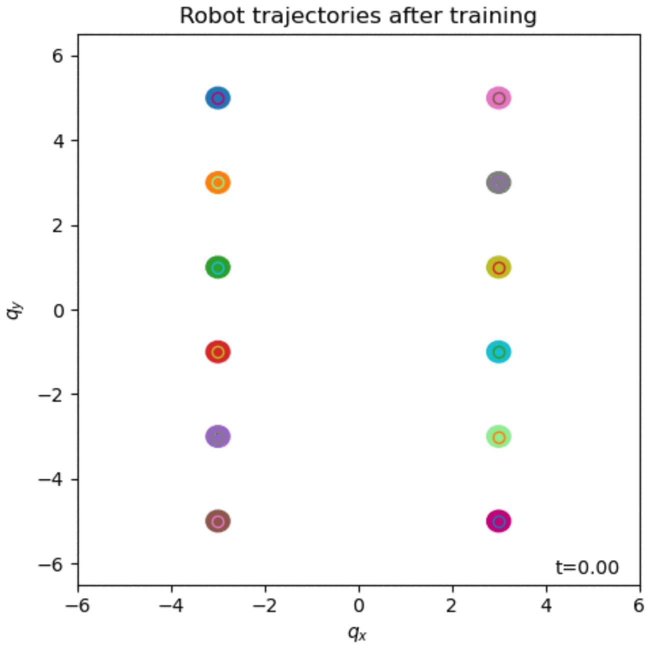
Regularization loss for pH-DNNs

Smoothing parameters across layers

Navigation task using pH NN distributed controllers

Clara Galimberti





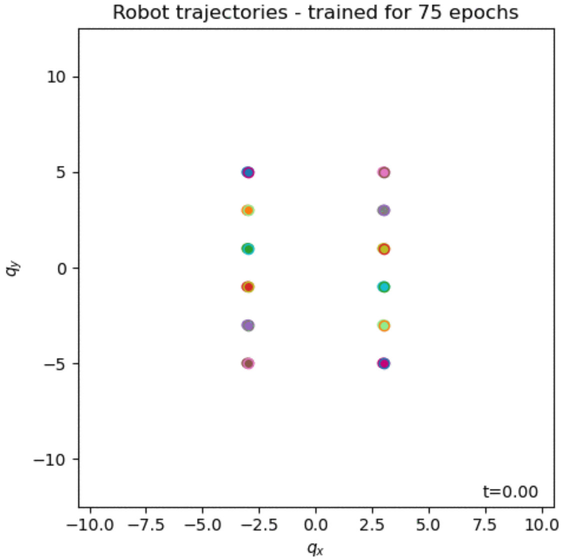
Zero collisions after training

Stability is guaranteed *by design*

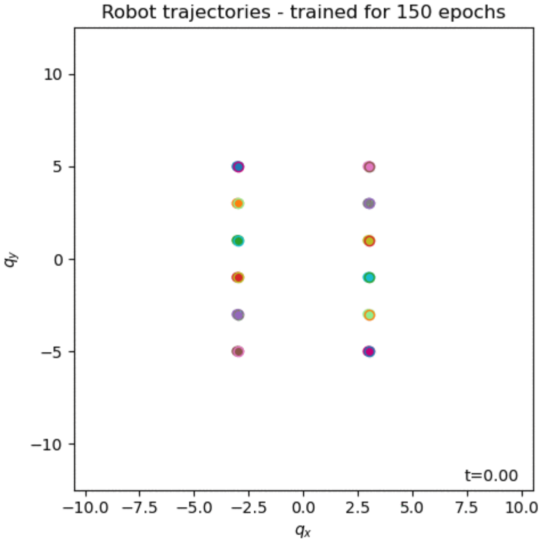
Gifs can be found in our GitHub repository:
https://github.com/DecodEPFL/DeepDisCoPH

Navigation task using pH NN distributed controllers

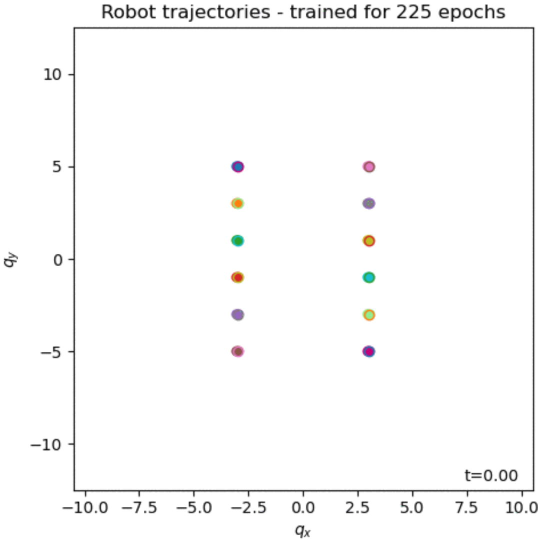
- Early stopping of the training:



25% of the training



50% of the training



75% of the training

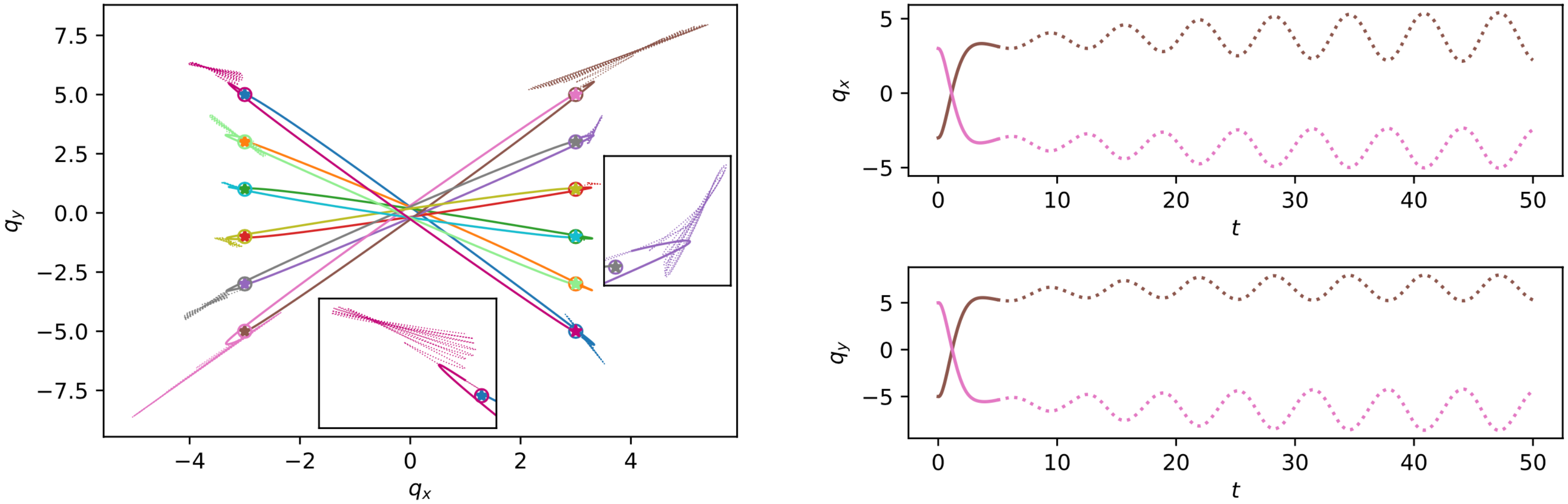
Stability is always guaranteed

Gifs can be found in our GitHub repository:
https://github.com/DecodEPFL/DeepDisCoPH

Navigation task using pH NN distributed controllers

- Replacing neural port-Hamiltonian controllers with MLP networks

Results after training → even when not considering collision avoidance



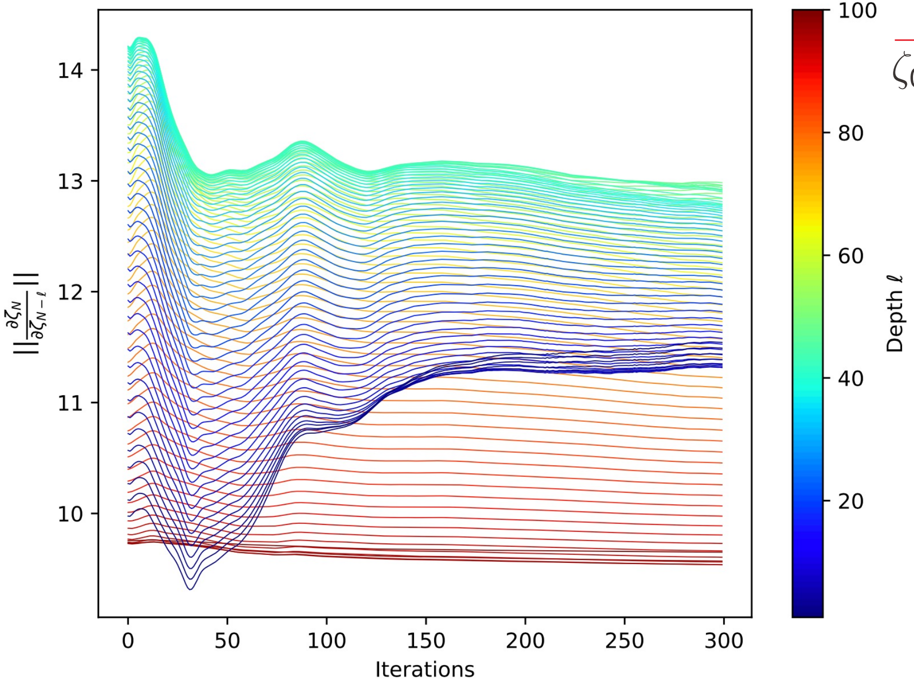
No stability guarantees

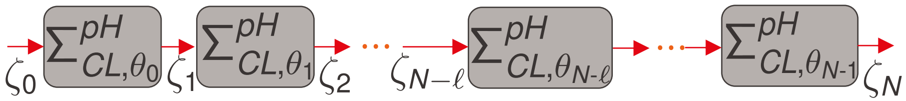
EPFL

Navigation task using pH NN distributed controllers

- Gradients during training

Backward sensitivity matrix norm





$$\left\|\frac{\partial \zeta_N}{\partial \zeta_{N-\ell}}\right\| \geq 1$$

validates the absence of vanishing gradients during training

Conclusions

- pH NN control policies:
  - Stability of the closed loop *by-design,* i.e. for arbitrary parameters
  - Non-vanishing gradients during training
  - Distributed implementations complying with pH structure

- Next steps?
  - Stable NN controllers by design beyond pH? $\longrightarrow$ [2]
  - Going data-driven: How to incorporate uncertainties in the system modelling?

[2] L. Furieri, C. Galimberti and G. Ferrari Trecate, “Neural System Level Synthesis: Learning over All Stabilizing Policies for Nonlinear Systems”, *arXiv preprint arXiv: 2203.11812*, 2022.

Thank you for your attention!