SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

*Master in Artificial Intelligence and Robotics*

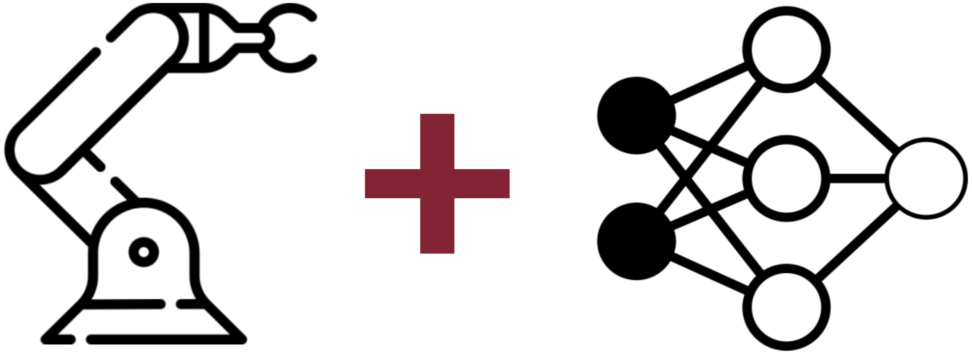
# Data-driven identification of a one-link robot with flexible joint

*Lagrangian Neural Networks applied to elastic robots*

**Course: Robotics 2**
**Professor: Alessandro De Luca**
**Tutor: Pietro Pustina**
**Students: Lorenzo Cirillo 1895955, Claudio Schiavella 1884561**

# INTRODUCTION

- Robot dynamic model
  - Relationship between applied forces and motion
  - D'Alembert, Hamilton
- Euler-Lagrange equations
  - Modeling and control
  - Symbolic form equations of motions
- Modeling systems → Neural networks → Lagrangian NN



SAPIENZA
UNIVERSITÀ DI ROMA

# PARAMETRIC LAGRANGIAN

## Lagrangian

$$\boxed{L(q,\dot{q}) = T(q,\dot{q}) - U(q)}$$

- Kinetic $T(q,\dot{q})$ and potential $U(q)$ energy
- Generalized coordinates $q \in \mathbb{R}^N$
  - $N$ = robot DOF

## Euler-Lagrange equations

$$\boxed{\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = u_i} \qquad i = 1, \dots, N$$

- Non-conservative forces $u_i$
  - Considered null in our application

# PARAMETRIC LAGRANGIAN

**Parametric Lagrangian**

$$\boxed{\ddot{q} = \left(\nabla_{\dot{q}}\nabla_{\dot{q}}^{T}L\right)^{-1}\left[\nabla_{q}L - (\nabla_{q}\nabla_{\dot{q}}^{T}L)\dot{q}\right]}$$

- Lagrangian analytical expression not always known
- Alternative approach
  - Learning L + Automatic differentiation

Sapienza Università di Roma

# RIGID ROBOT

*1R robot with y-axis agreeing with gravity*

**Dynamic Model:** $$\boxed{(I + md^2)\ddot{q} + mg_0 d\sin(q) = 0}$$

**Acceleration:** $$\boxed{\ddot{q} = -\frac{1}{I + md^2} mg_0 d\sin(q)}$$

- Coriolis and Centrifugal term not present
  - One single link
  - Contribute to the joint when it is moving is null

# ELASTIC ROBOT

*1R robot with y-axis agreeing with gravity*

**Dynamic model:**

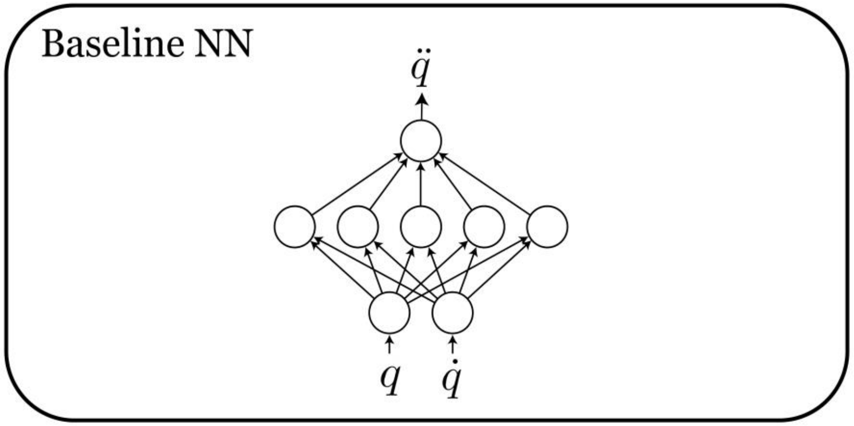
$$\begin{cases} (I + md^2)\ddot{q} + mg_0 d\sin(q) + k(q-\theta) = 0 \\ I_m\ddot{\theta} + k(\theta - q) = 0 \end{cases}$$

**Accelerations:**

$$\begin{cases} \ddot{q} = -\dfrac{1}{I + md^2}\left[mg_0 d\sin(q) + k(q-\theta)\right] \\ \ddot{\theta} = \dfrac{1}{I_m}k(q-\theta) \end{cases}$$

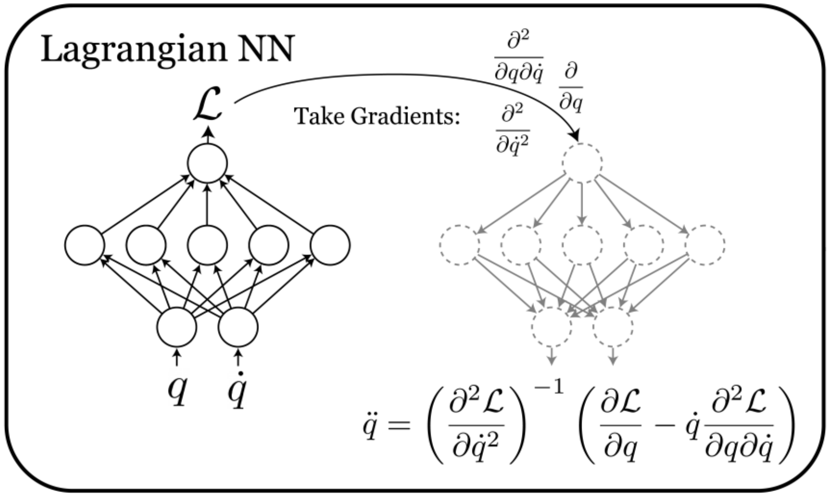
- Again Coriolis and Centrifugal term not present
  - Constant inertia matrix

# BASELINE NN




- Classic feedforward neural network
  - Only fully connected layers
- Learn $\ddot{q}$ directly from $\dot{q}$ and $q$
  - Not considering Lagrangian and relative constraints
- Function of a comparison baseline
  - Same structure of Lagrangian NN to comparison reasons

# LAGRANGIAN NN




- Same baseline structure, completely different approach
  - Very close to robot nature

- Learn Lagrangian
  - Functional programming context
  - Predictions benefit of the Lagrangian approach features

- $\ddot{q}$ obtained from parametric Lagrangian

# SIMULATIONS

- **Type 1:** Observe behavior of the two networks
  - Trajectory tracking
  - Trajectory error
  - Energy conservation
- **Type 2:** Introduce measurament noise
  - Same tests as in type 1
  - Highlight network robustness in not ideal context

# RIGID CASE

## Robot parameters

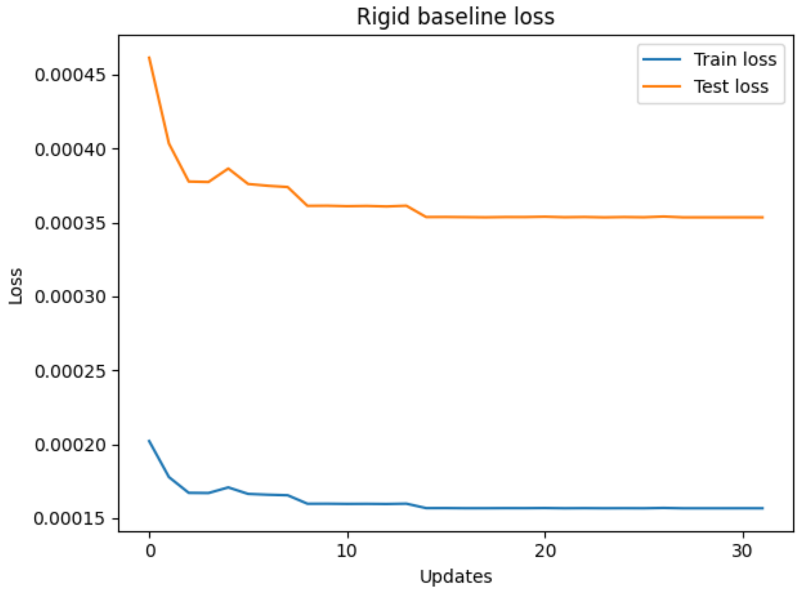
| Parameter | Measure |
|---|---|
| Length | $0.30\ m$ |
| Base radius | $0.01\ m$ |
| CoM distance | $0.15\ m$ |
| Mass | $0.25\ kg$ |
| Inertia | $0.01125\ kg\ m^2$ |

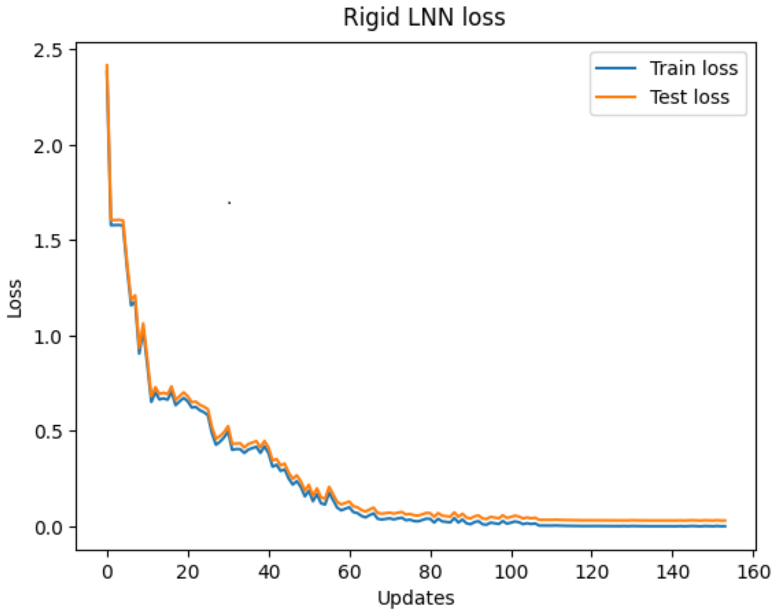
## Network setup

| Characteristic | Value |
|---|---|
| Layers | 4 |
| Neurons | 500 |
| Activation | Softplus |
| Loss | MSE |
| Regularization | L2 penalty |
| Train epochs | 10 000 |

# RIGID CASE – NETWORK TRAINING

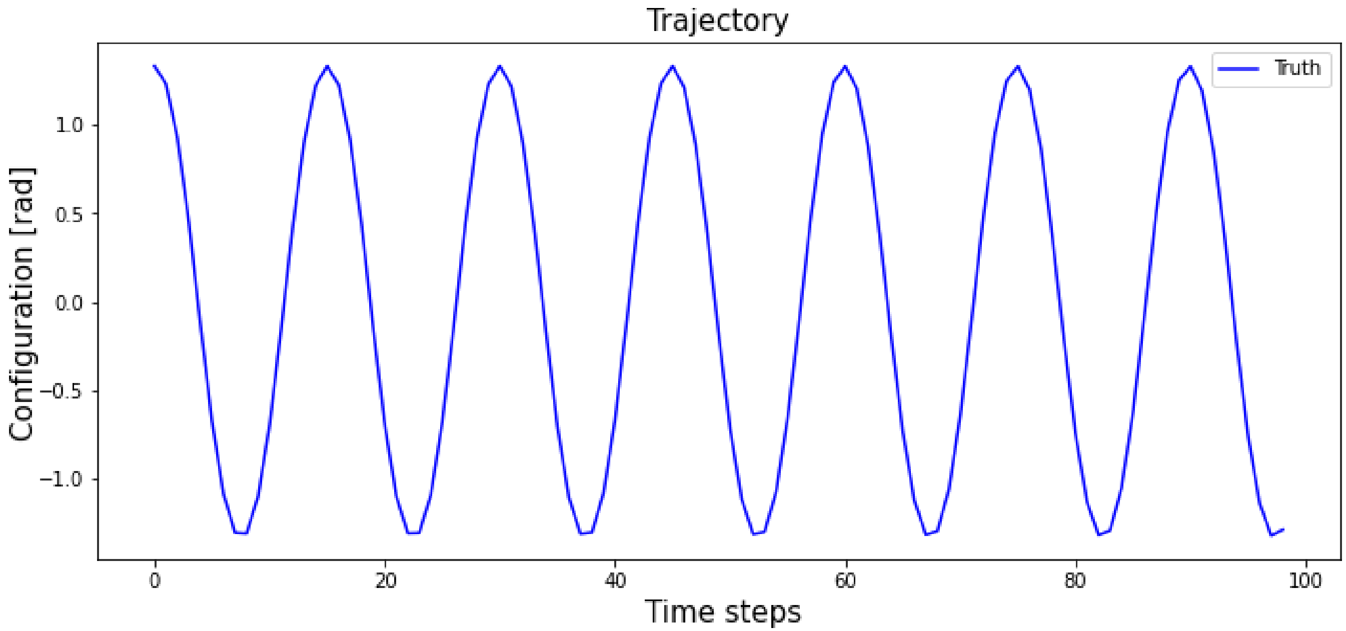
## Train and test losses



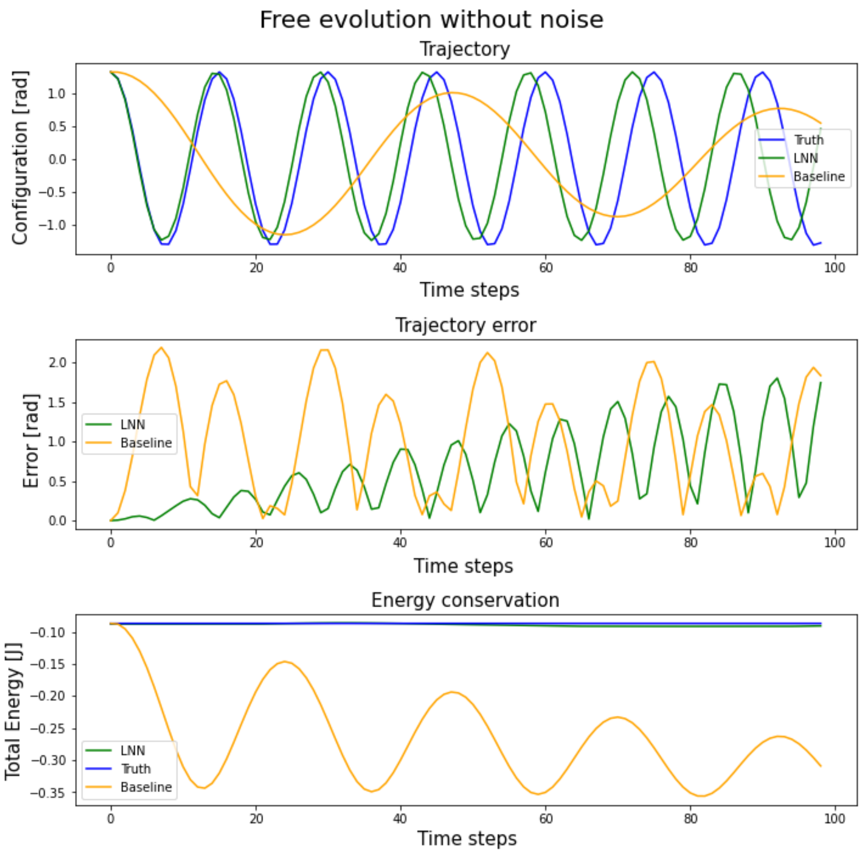



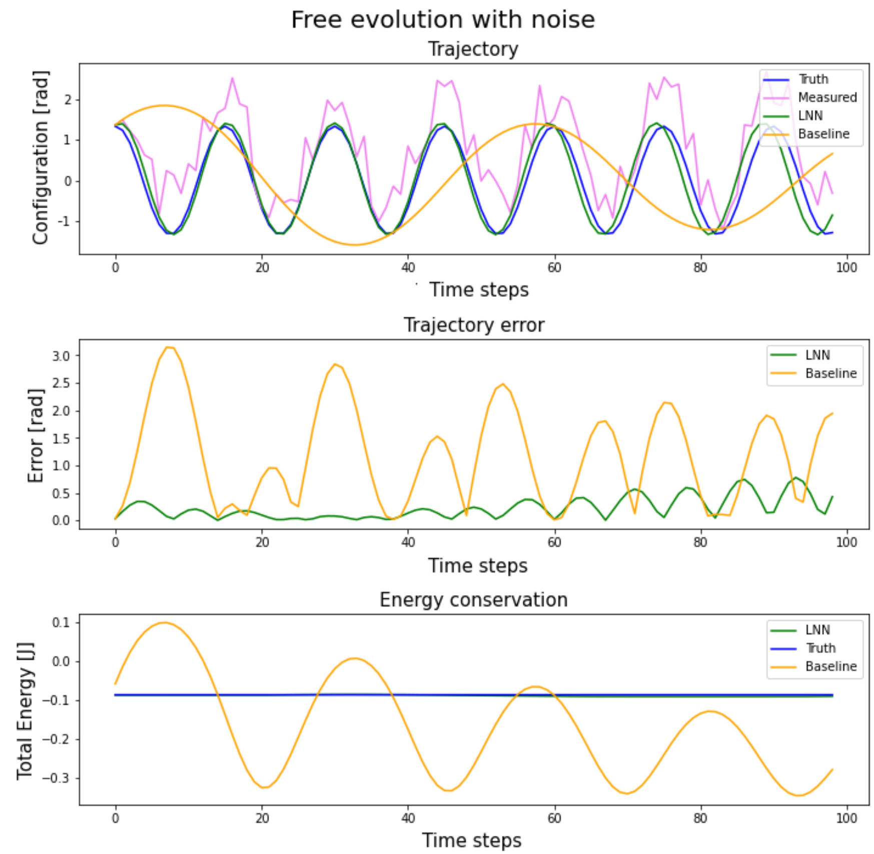
# RIGID CASE – SIMULATION 1

**Initial conditions:** $q(0) = 1{,}33\,rad \quad \dot{q}(0) = 0{,}2\,\dfrac{rad}{s}$



# RIGID CASE – SIMULATION 1

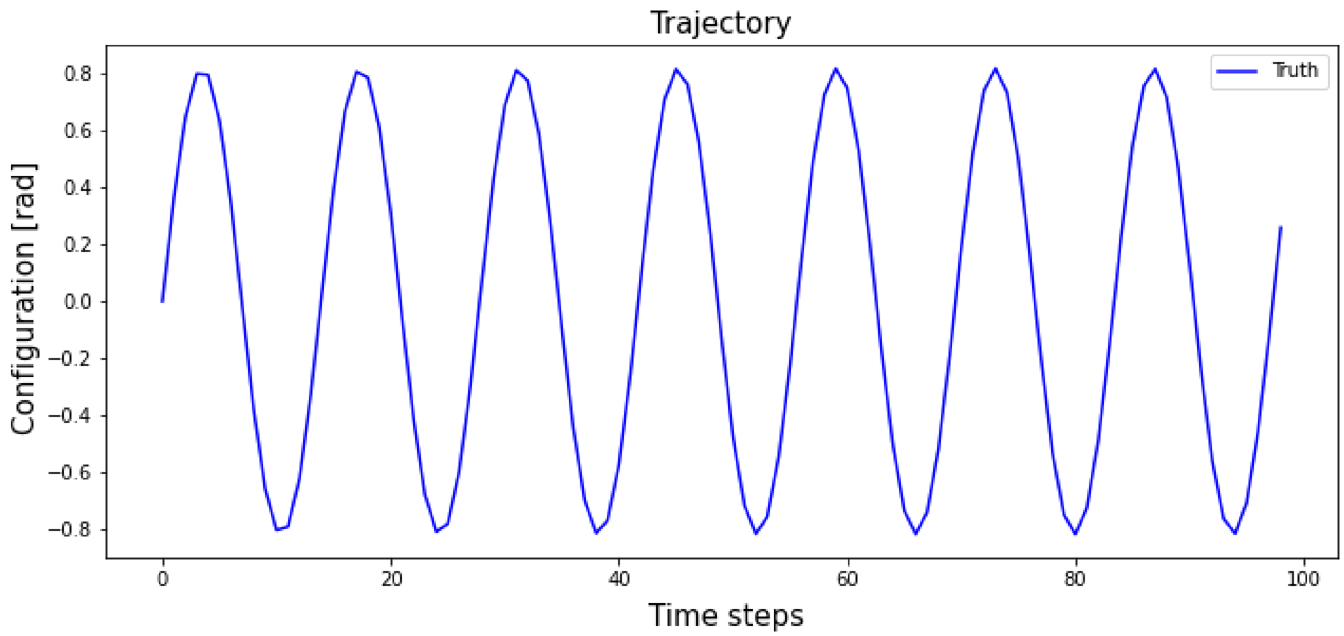


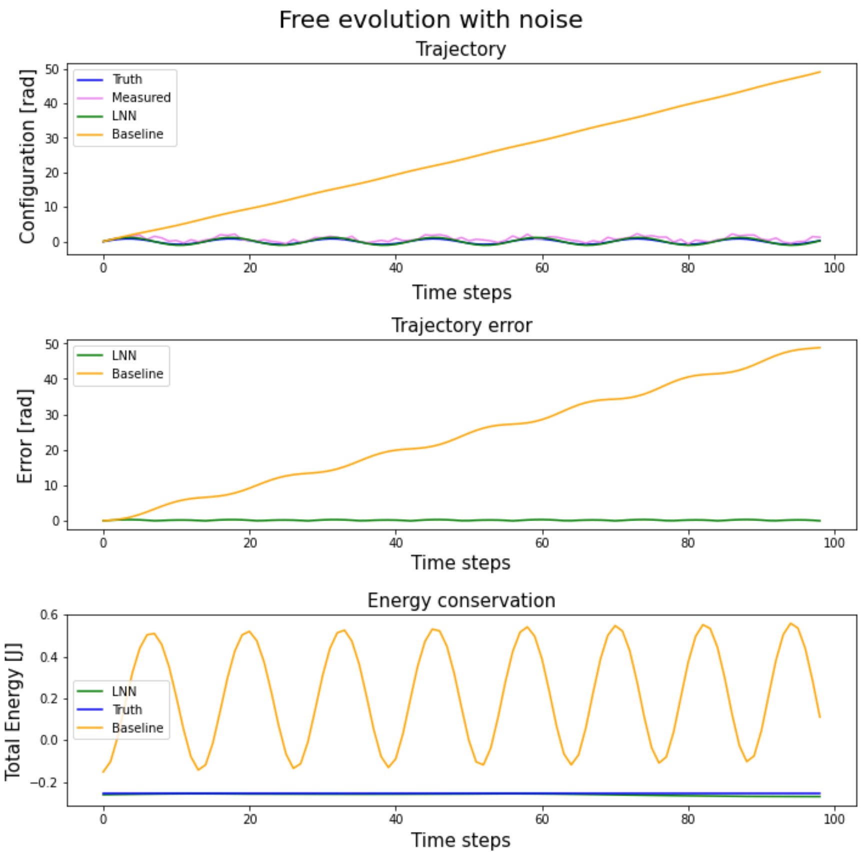
# RIGID CASE – SIMULATION 2

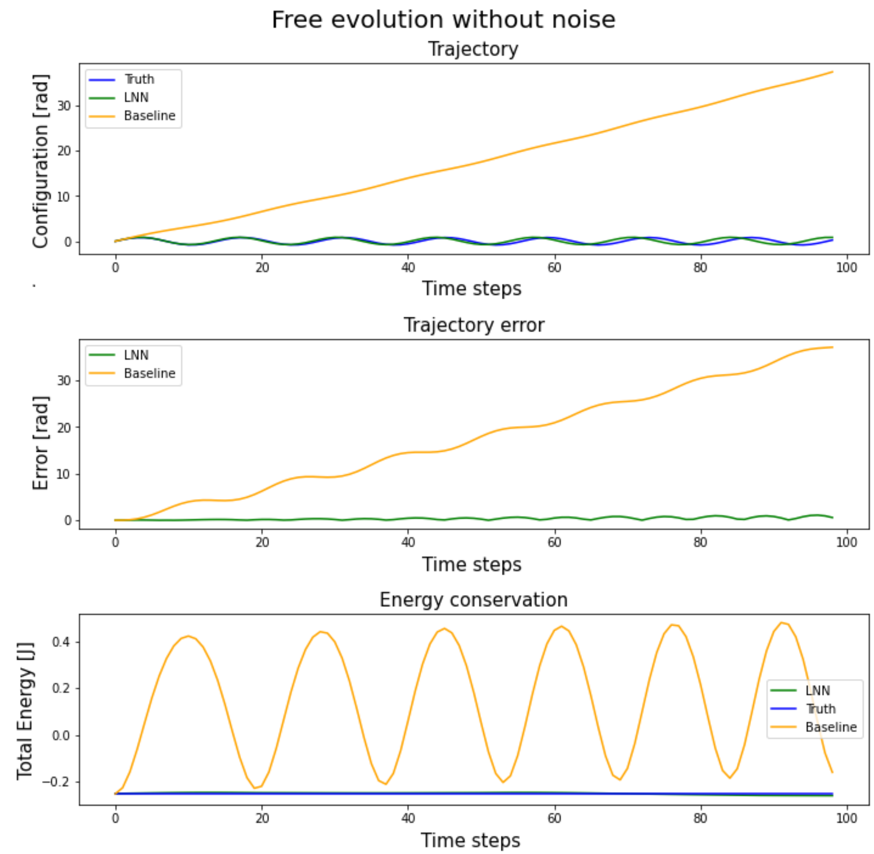
**Initial conditions:**

$$q(0) = 0\,rad \quad \dot{q}(0) = 4\,\frac{rad}{s}$$



# RIGID CASE – SIMULATION 2

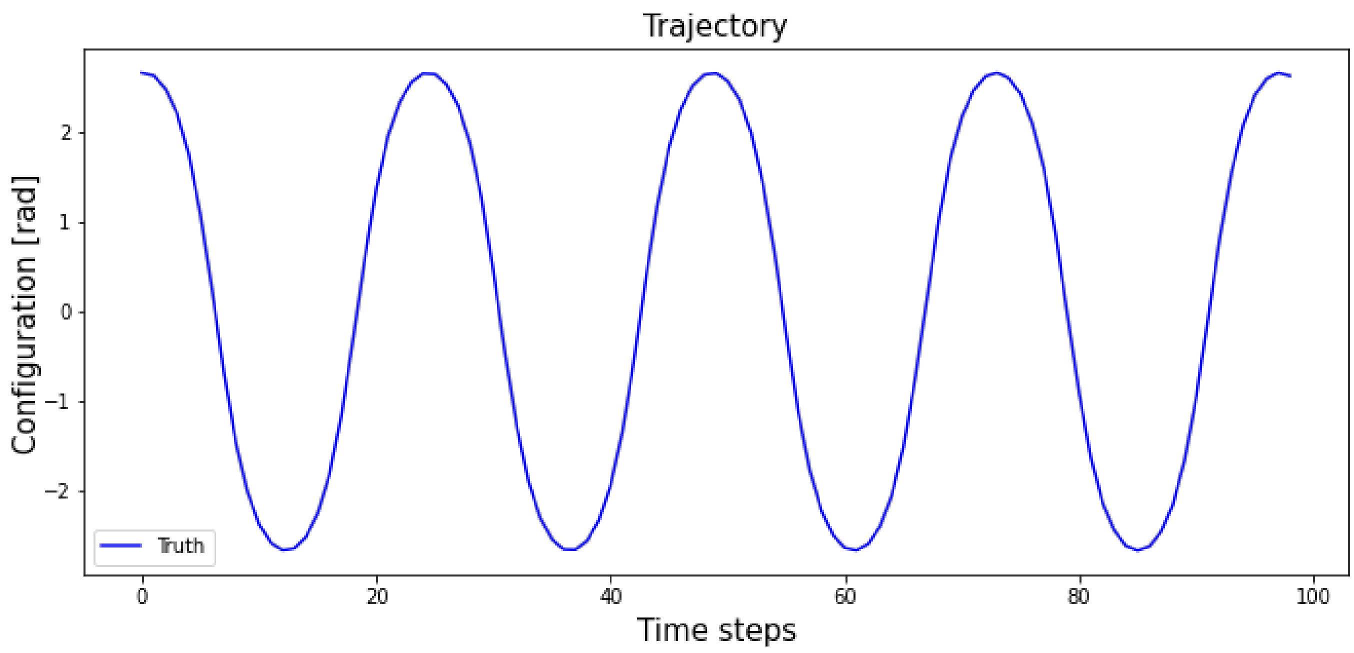


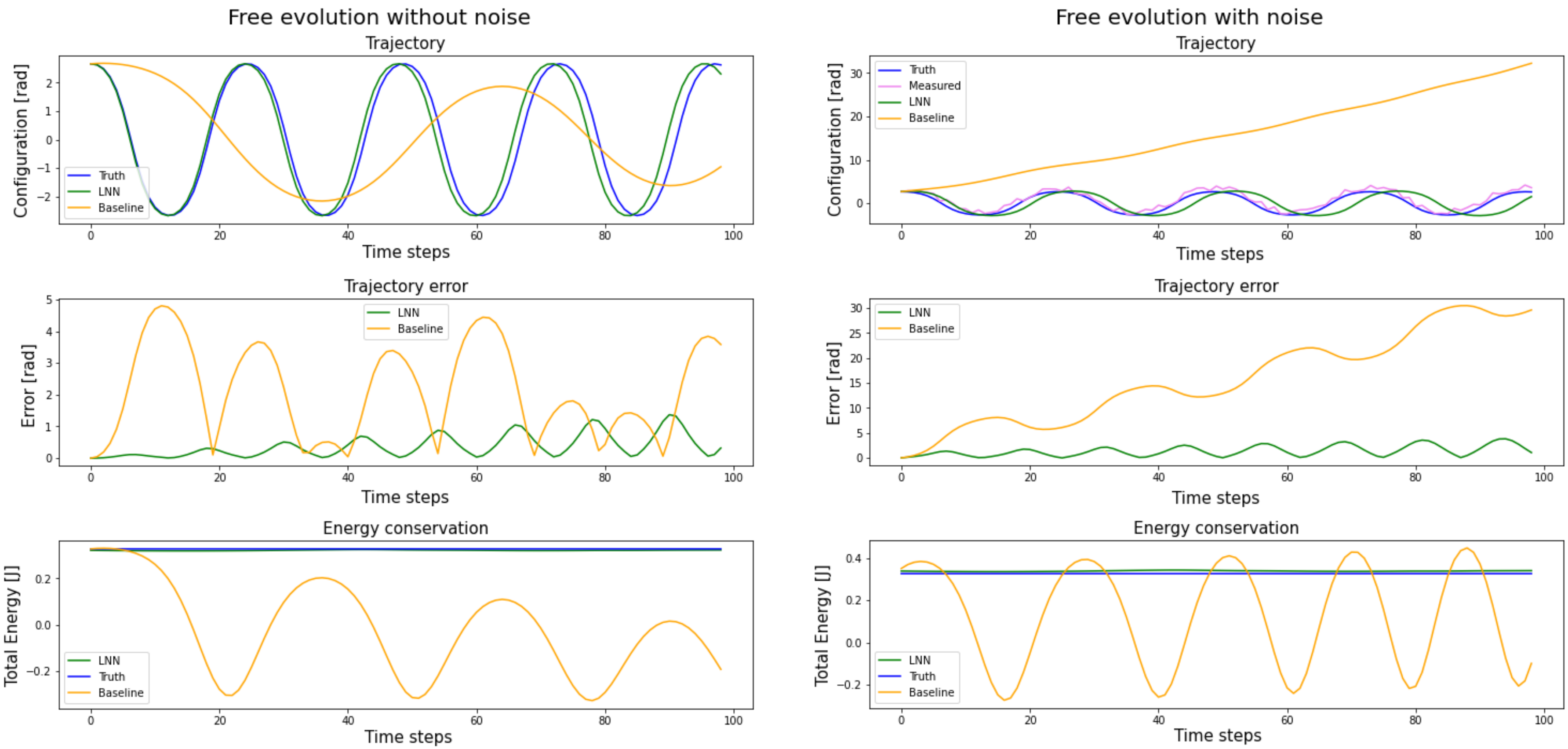
Sapienza Università di Roma

# RIGID CASE – SIMULATION 3

**Initial conditions:** $q(0) = 2{,}66\, rad \quad \dot{q}(0) = 0{,}5\, \frac{rad}{s}$



# RIGID CASE – SIMULATION 3



# ELASTIC CASE

## Motor parameters

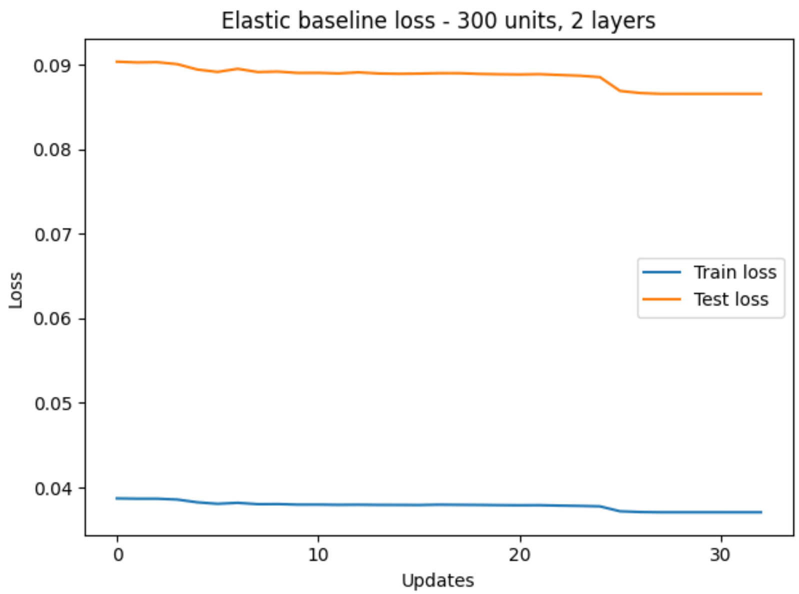
| Parameter | Measure |
|---|---|
| Height | $0.04\ m$ |
| Base radius | $0.02\ m$ |
| Mass | $0.4\ kg$ |
| Reduction ratio | $160$ |
| Inertia | $0.0128\ kg\ m^2$ |
| Stiffness constant | $50\ Nm/rad$ |

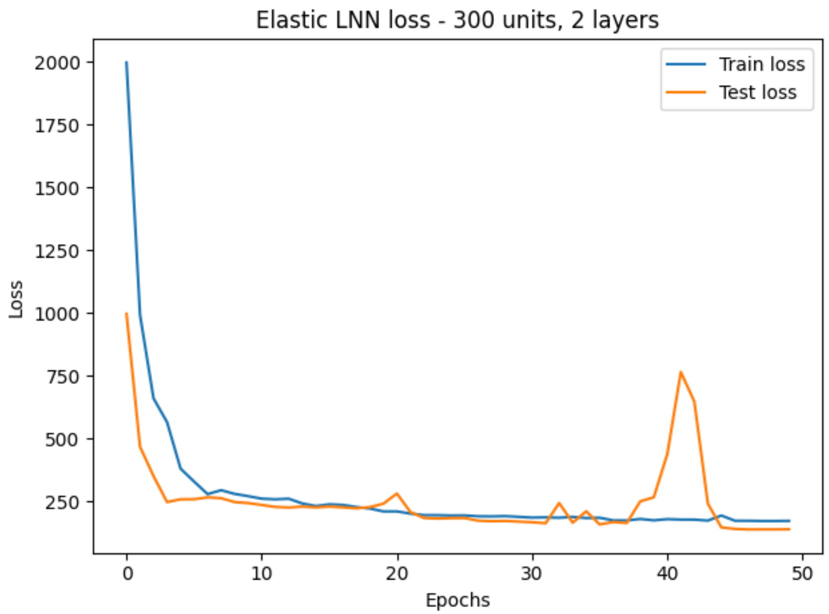
## Network setup

| Configuration | Units | Layers |
|---|---|---|
| A | 300 | 2 |
| B | 500 | 4 |
| C | 700 | 5 |

# ELASTIC CASE – CONFIGURATION A

## Train and test losses

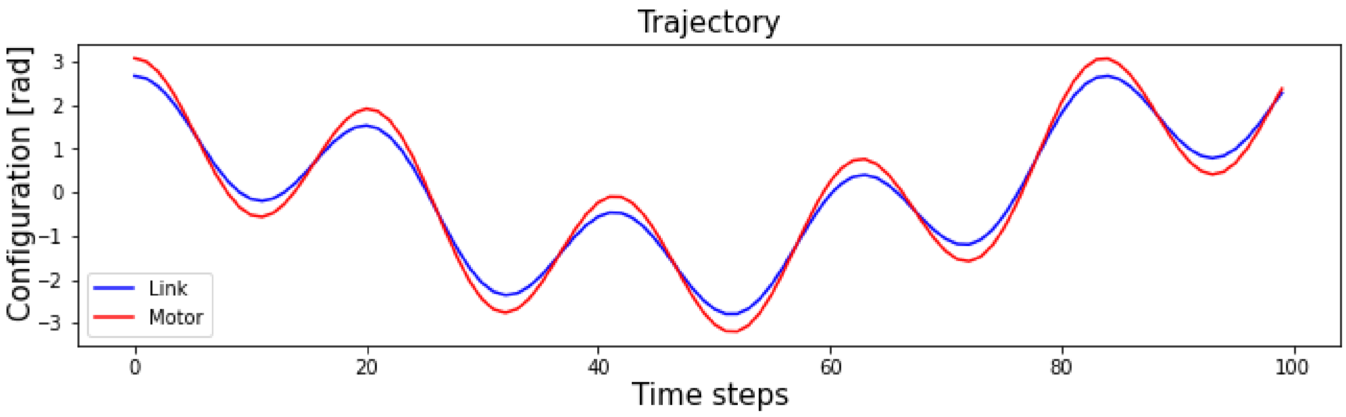


# ELASTIC CASE – SIMULATION 1

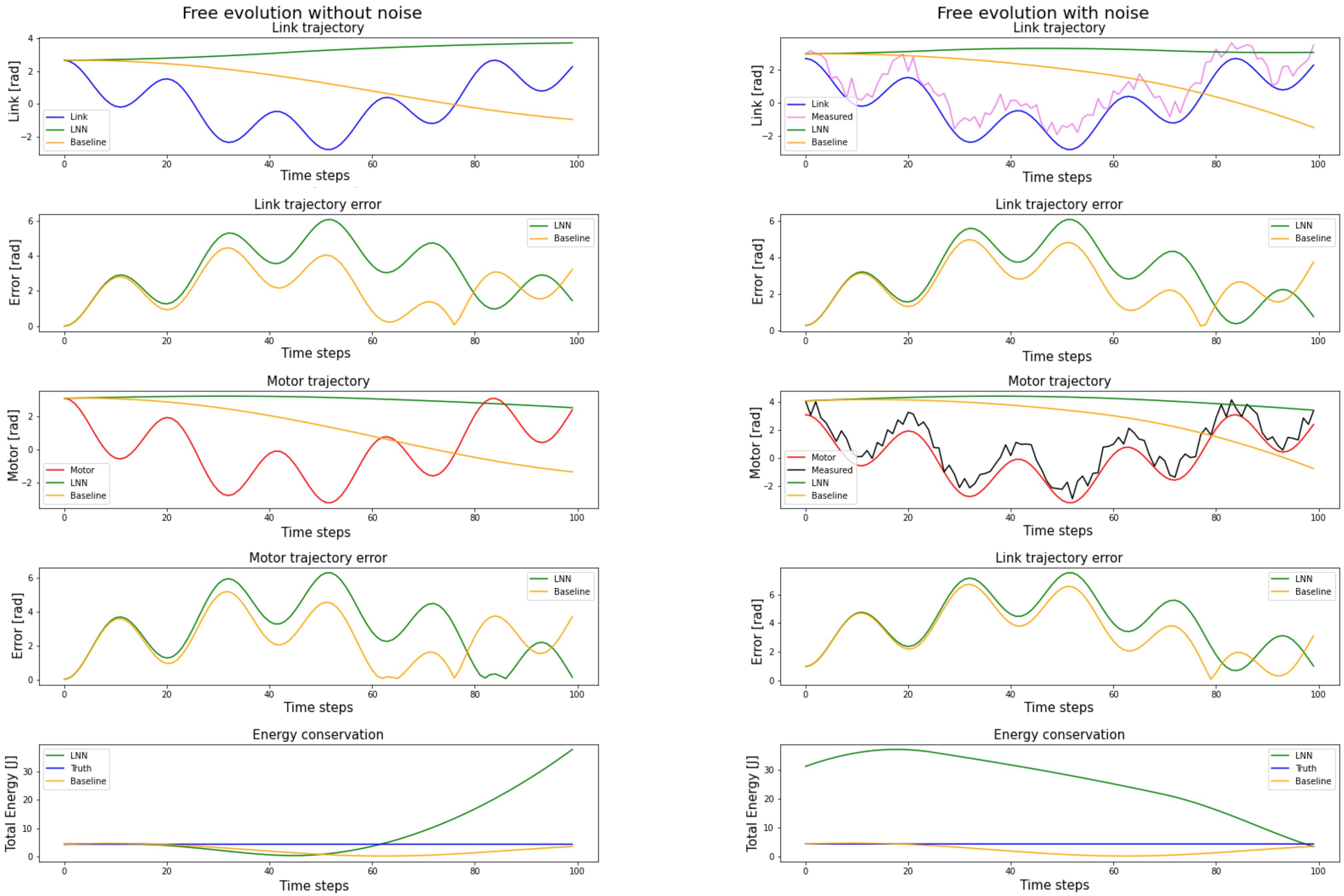
**Initial conditions:**

$$\begin{cases} q(0) = 2{,}66\, rad \quad \dot{q}(0) = 0{,}5 \dfrac{rad}{s} \\ \theta(0) = 3{,}06\, rad \quad \dot{\theta}(0) = 1 \dfrac{rad}{s} \end{cases}$$



SAPIENZA UNIVERSITÀ DI ROMA

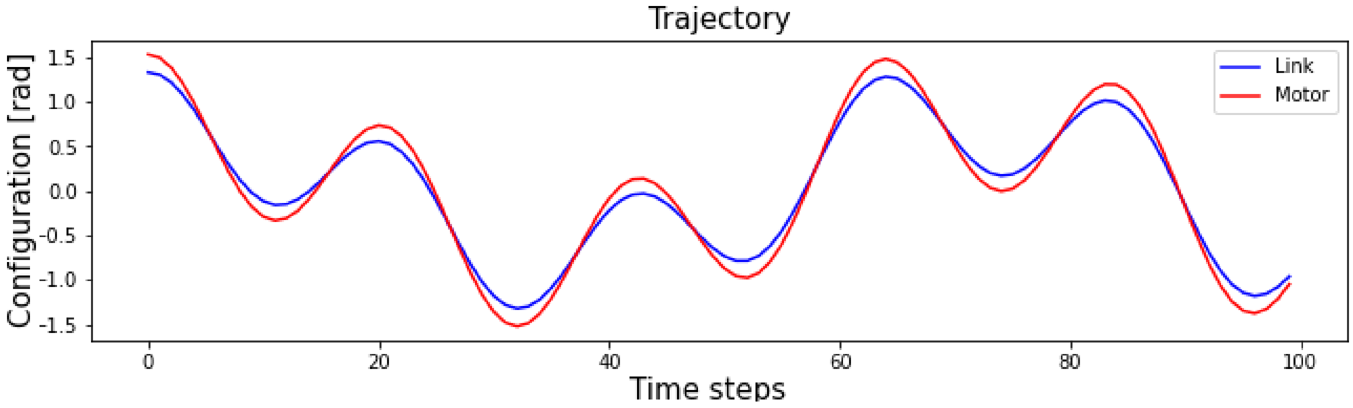
# ELASTIC CASE – SIMULATION 1



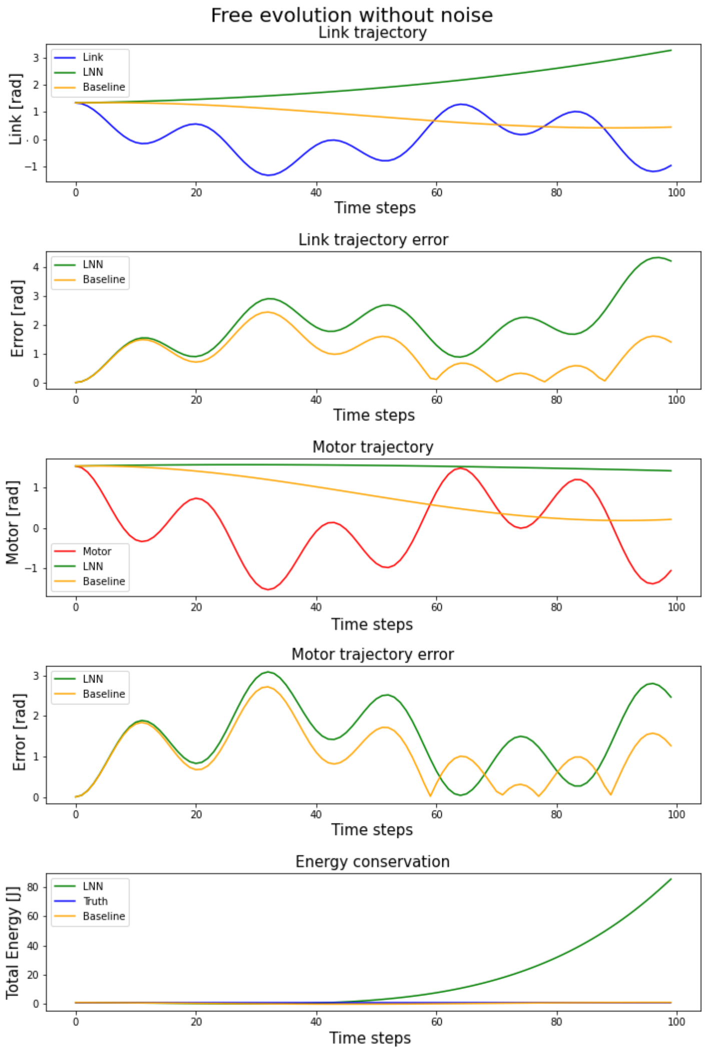
# ELASTIC CASE – SIMULATION 2

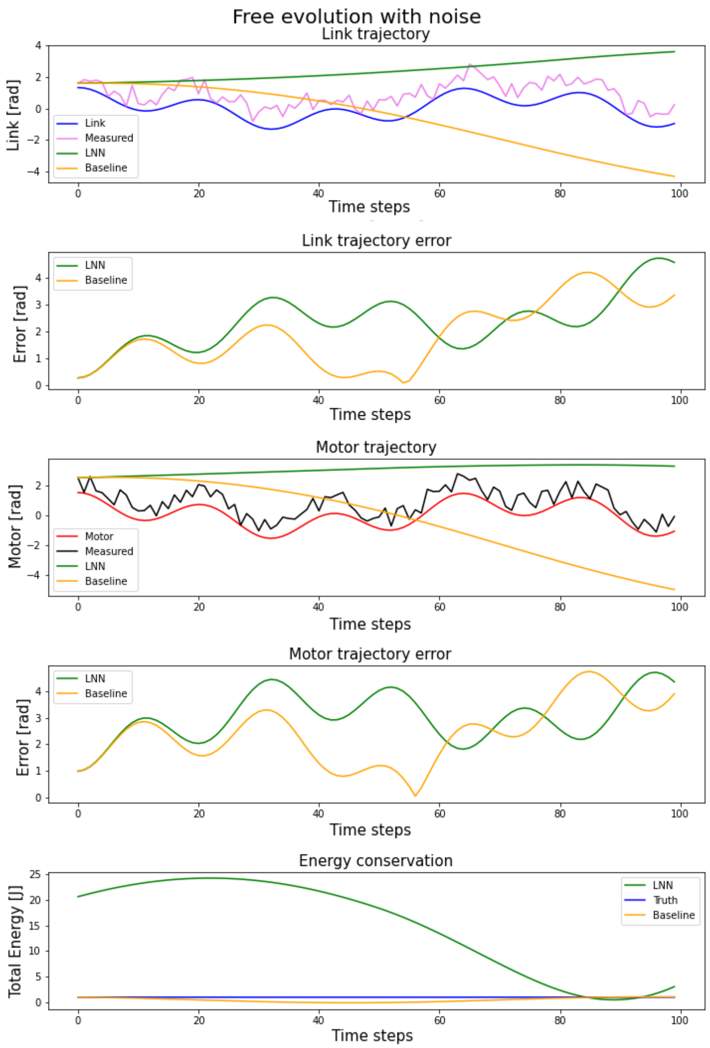
**Initial conditions:**

$$\begin{cases} q(0) = 1{,}33\,rad \quad \dot{q}(0) = 1\,\dfrac{rad}{s} \\ \theta(0) = 1{,}51\,rad \quad \dot{\theta}(0) = 0{,}5\,\dfrac{rad}{s} \end{cases}$$



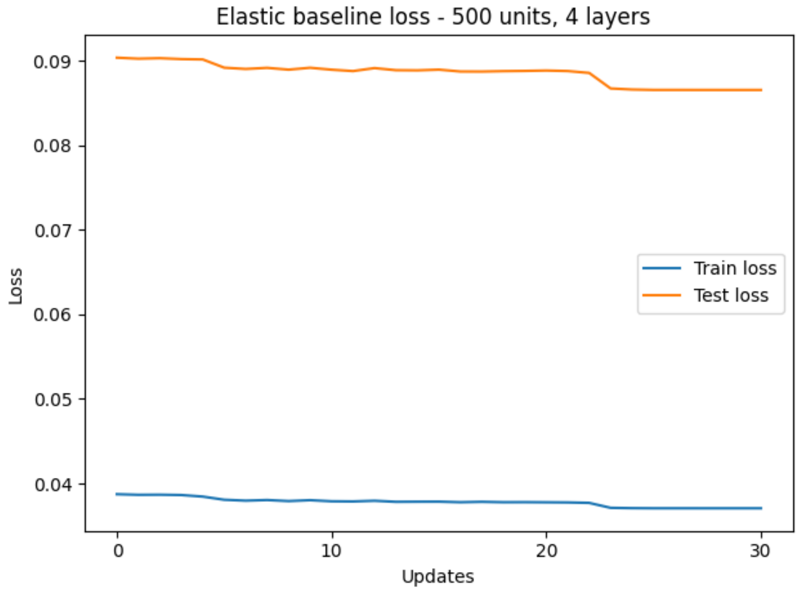
# ELASTIC CASE – SIMULATION 2

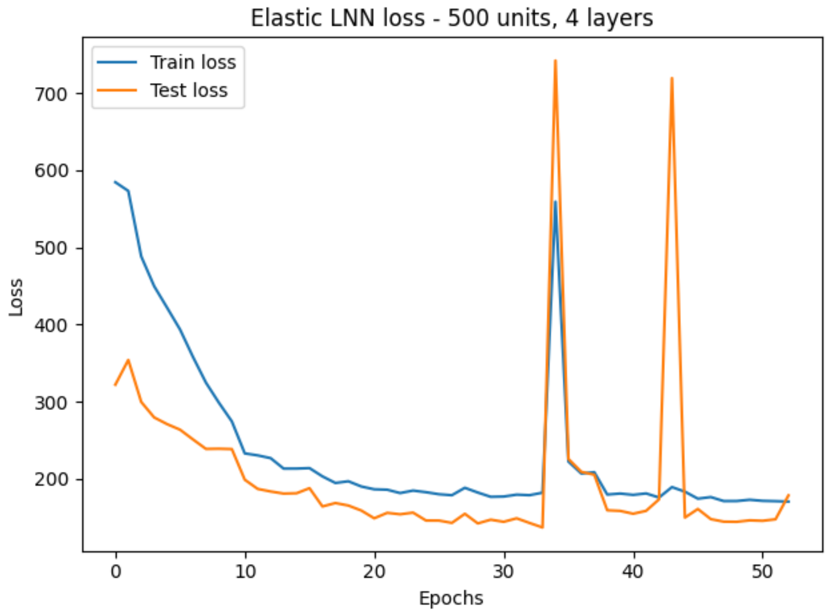


# ELASTIC CASE – CONFIGURATION B

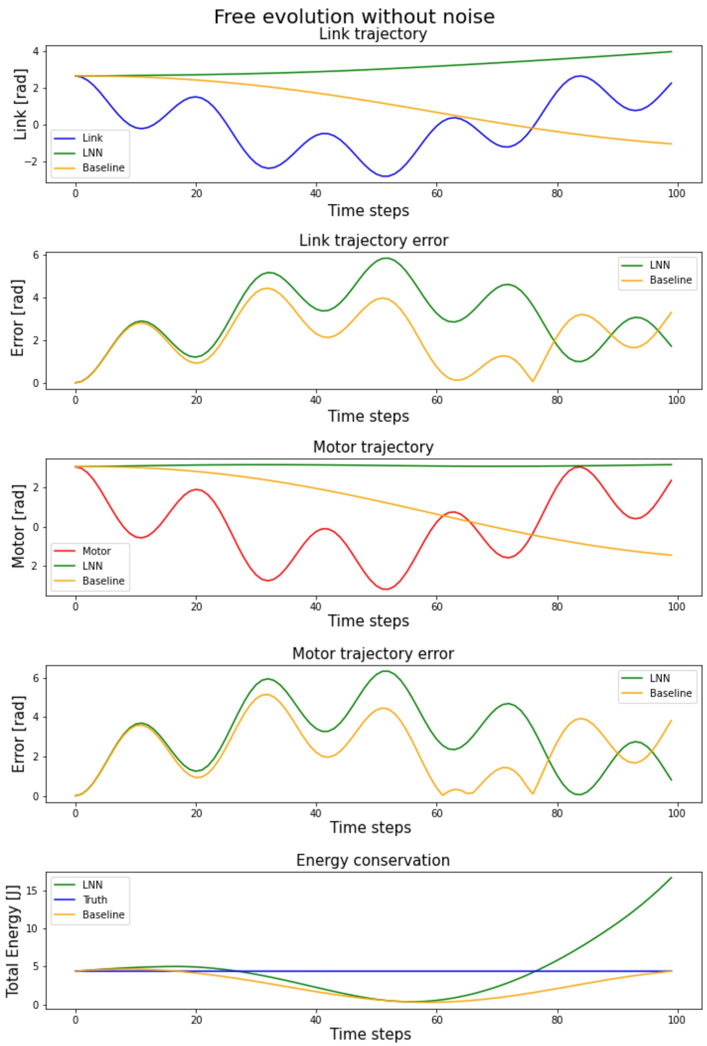
## Train and test losses

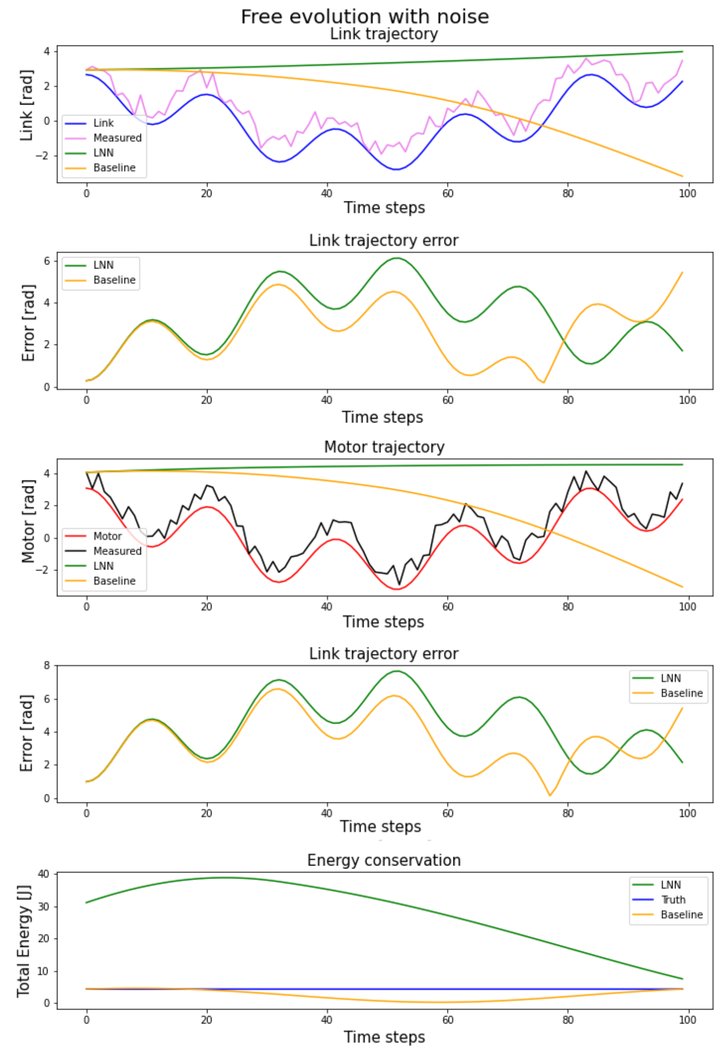


SAPIENZA
UNIVERSITÀ DI ROMA

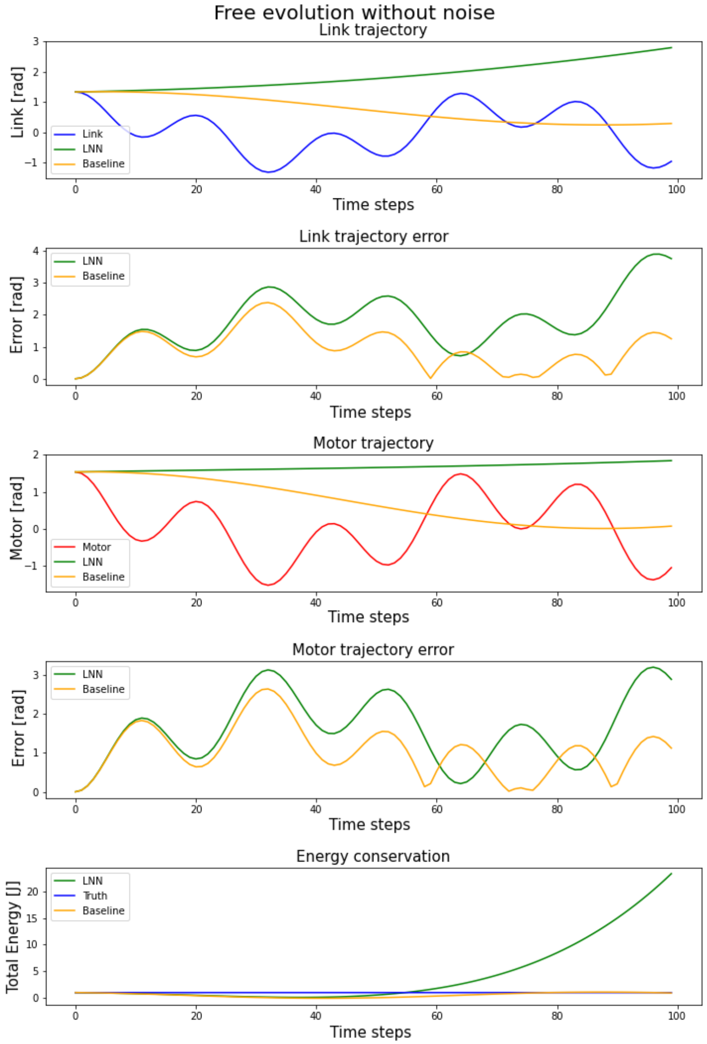
# ELASTIC CASE – SIMULATION 3

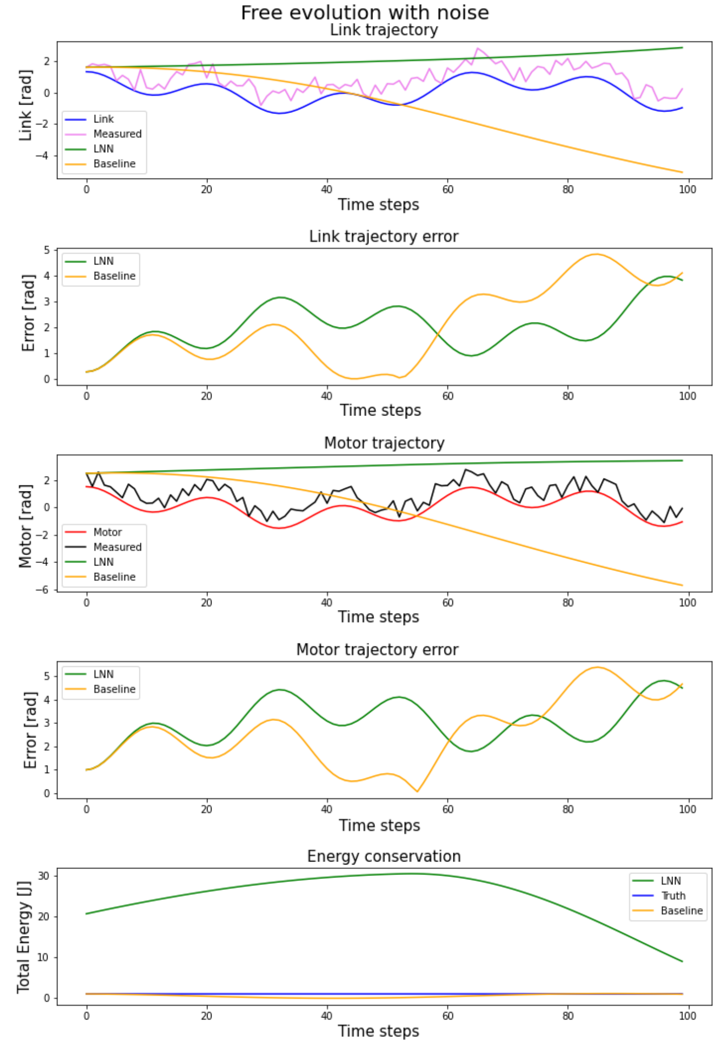


SAPIENZA
UNIVERSITÀ DI ROMA

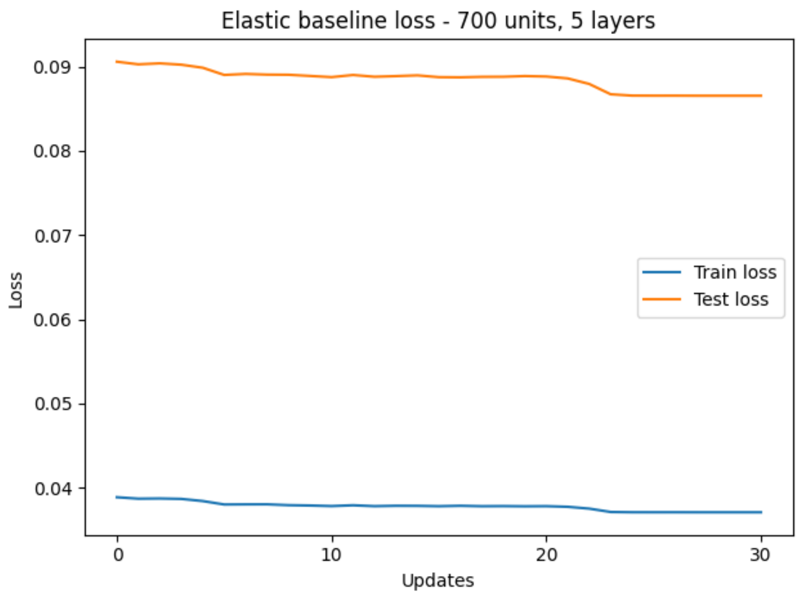
# ELASTIC CASE – SIMULATION 4

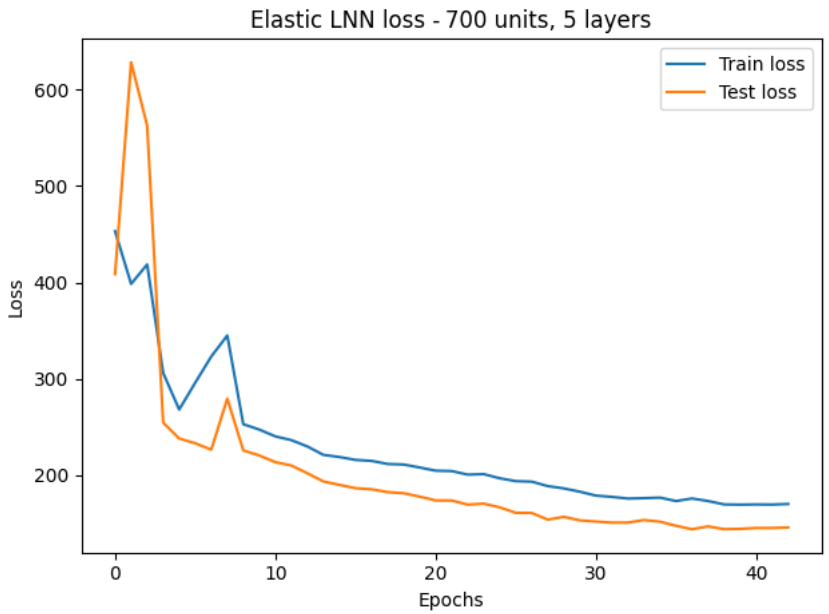


# ELASTIC CASE – CONFIGURATION C

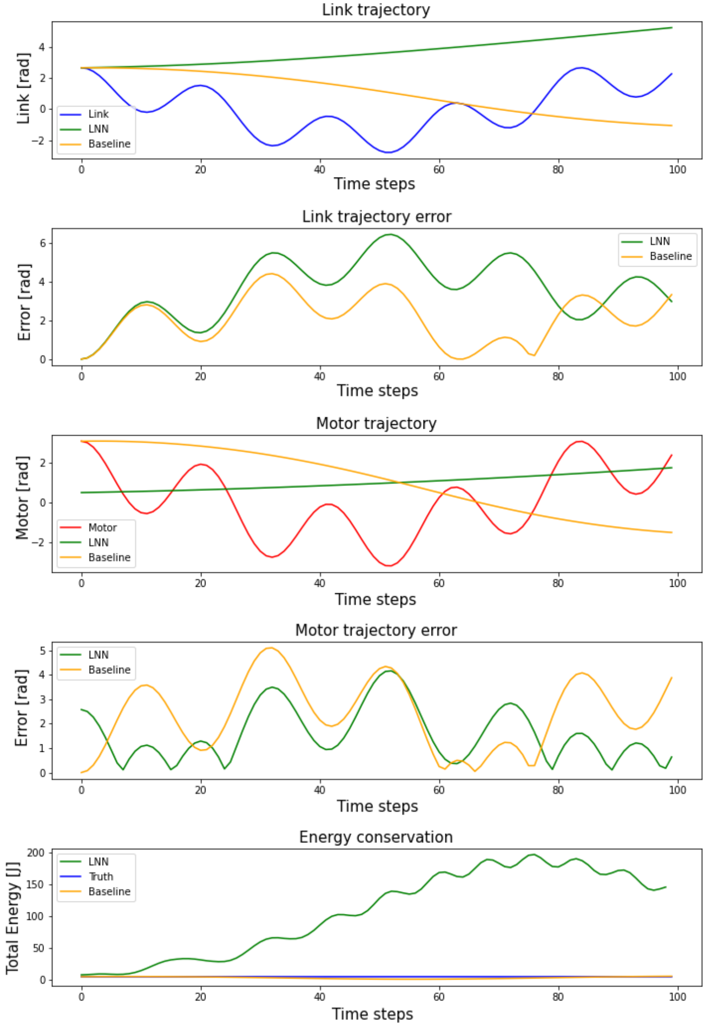
## Train and test losses



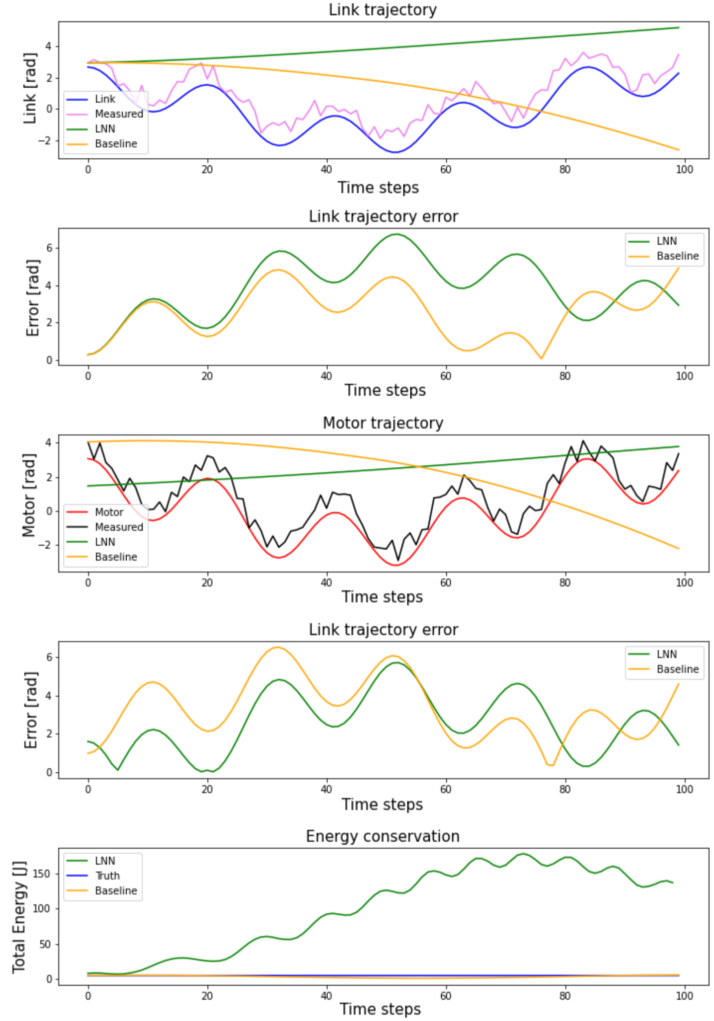



# ELASTIC CASE – SIMULATION 5

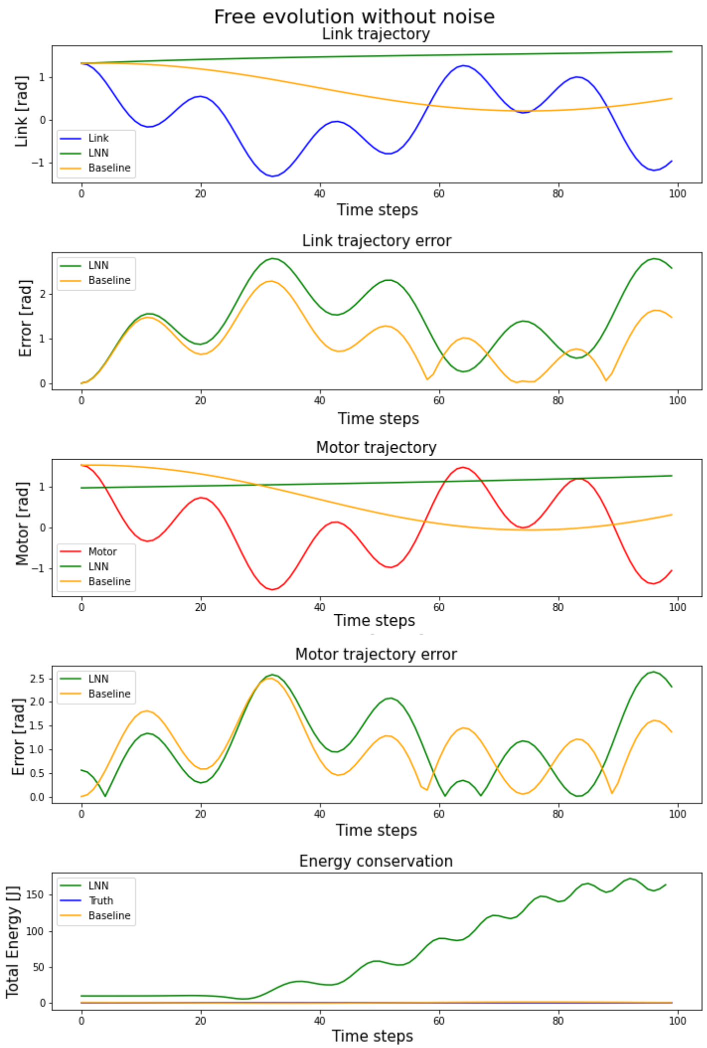
Free evolution without noise

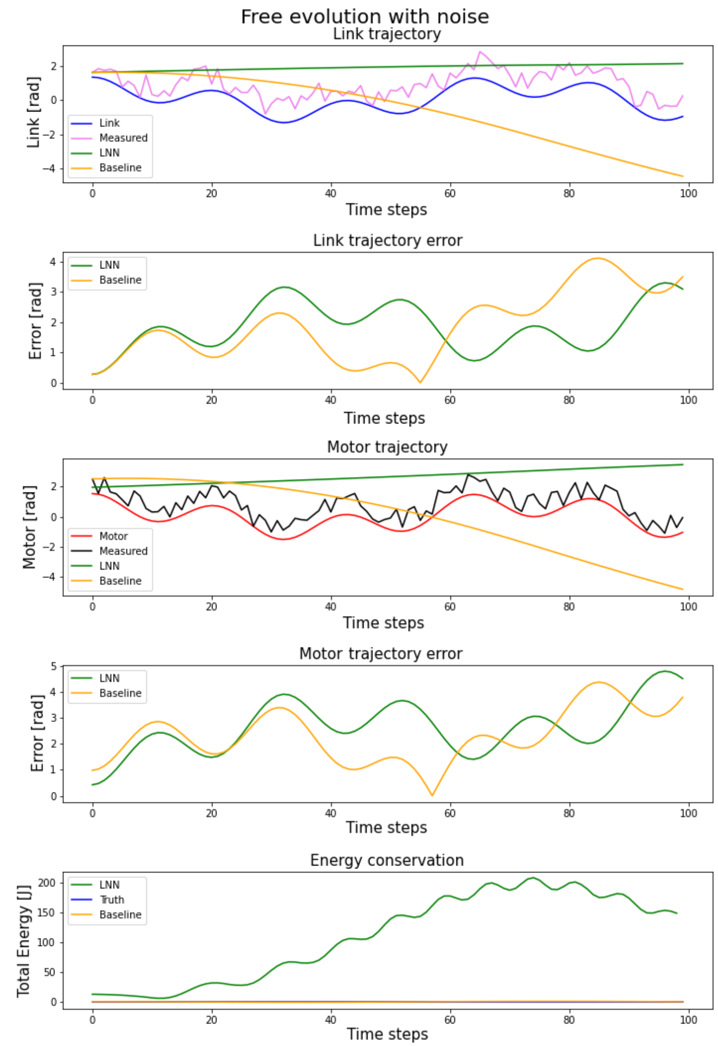



Free evolution with noise



# ELASTIC CASE – SIMULATION 6



# CONCLUSIONS

- At least in the rigid case, LNN performances are very good
  - Both in trajectory error and energy conservation
- Unfortunately, bad results in elastic case
- Elastic LNN seems to improve as network complexity increase
  - More wide/deep network and more initial conditions
  - High computational resources needed
- Paper LNN approach works on unitary configurations
  - We reject this in preference to real robot parameters

SAPIENZA
UNIVERSITÀ DI ROMA

**THANK YOU FOR THE ATTENTION**