Physics Informed Neural Networks for Optimal Intercept Problem

[1]*Enrico Schiassi*,

[2]**Andrea D'Ambrosio**, [1]**Roberto Furfaro**, and [2]**Fabio Curti**,

[1]**University of Arizona, USA**

[2]**Sapienza University of Rome**

IAA/AAS SciTech Forum 2020, Dec. 08-10, 2020, Moscow, Russia

Contents

Space Systems Engineering Laboratory

Introduction: Overview and Motivations

Space Systems
Engineering Laboratory

- Optimal intercept problems represent one of the most useful optimization problem in aerospace engineering.

- Optimal intercept problems are mainly related to missile guidance.
  - It is important to have robust algorithms suitable for real-time applications.

Introduction: Goals

Space Systems Engineering Laboratory

- To employ ***Physics-Informed Neural Networks (PINN)*** to solve Optimal Control Problems (OPCs).
- To develop a new algorithm, suitable for on-board application, based on the newly developed ***Physics-Informed Extreme Theory of Functional Connections*** **(X-TFC)** [Schiassi et al. 2020] to compute optimal trajectories for intercept problems.
  - The focus of this talk is to show the effectiveness of our X-TFC based algorithm in solving optimal control problems, focusing particularly to the solution of the optimal intercept problem.

PINN and OCPs

- **(Data) + Neural Networks + Physics Laws = Physics-Informed Neural Networks (PINN)**
- PINNs are a newly developed framework for solving DEs
  - The physics laws (modeled via DEs), and eventually data, drive the training of the network

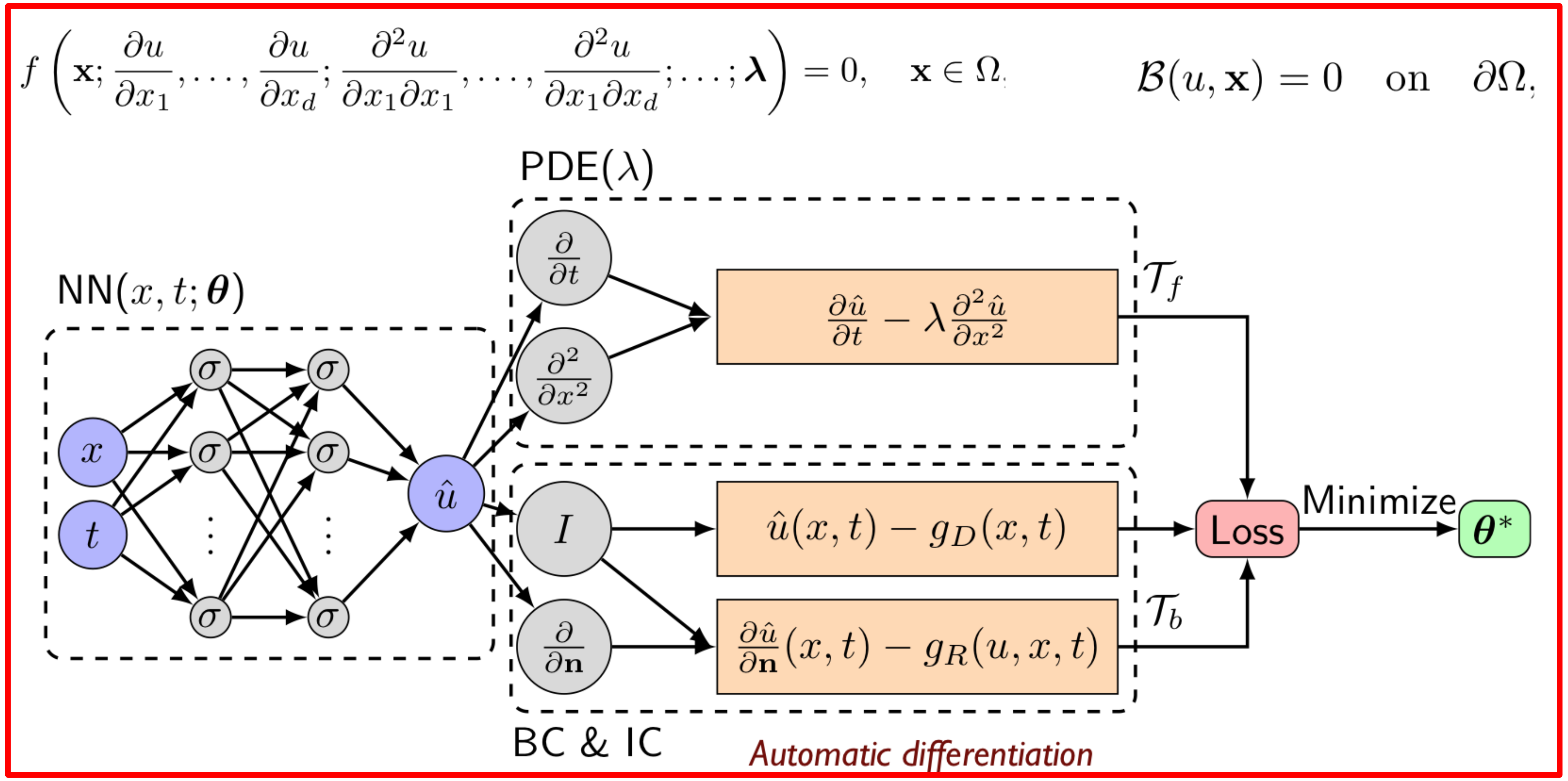


Image taken from: Lu, L., Meng, X., Mao, Z. and Karniadakis, G.E., 2019. DeepXDE: A deep learning library for solving differential equations. *arXiv preprint arXiv:1907.04502*.

- Optimal Control Problems (OCPs) are generally hard and computationally expensive. In general, Open-Loop solutions can be found in two ways
  - **Direct Method**: Transform a continuous problem in a finite NLP problems and find the minimum
  - **Indirect Method**: Apply Pontryagin Minimum Principle (PMP) to derive the necessary conditions
    - The solution of the OC reduces to the solution of a Two Point Boundary Value Problem (TPBVP) that is a systems of ODEs

New Methods for Solving DEs

Space Systems Engineering Laboratory

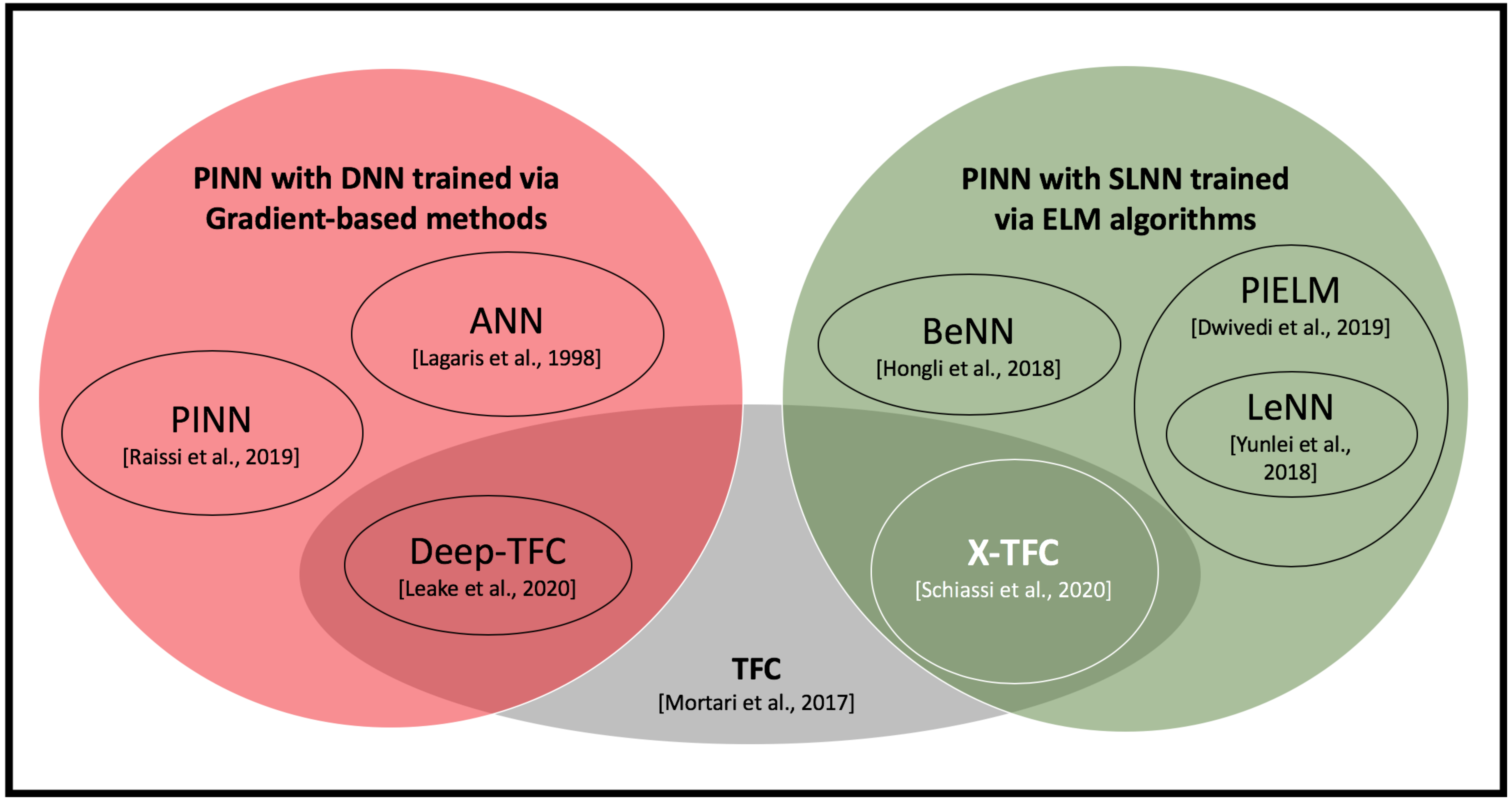
- The **Theory of Functional Connections (TFC)** [Mortari, 2017] is a recently developed framework for functional interpolation
  - The functions are approximated via a **constrained expression**
    - Sum of a **free-chosen** function and a functional that **analytically** satisfies the constraints
  - **TFC is applied to solve parametric DEs**
    - The free-chosen function is an expansion of Chebyshev polynomials
    - The constraints are the Initial/Boundary Conditions (IC or BC)
- The **Physics-Informed Neural Network (PINN) Methods** are a novel approach, coming from the Machine Learning community
  - The DE latent solutions are approximated via a (Deep) Neural Network (NN), and the DEs drive the NN training (i.e. it acts as regulator)

	Pros	Cons
TFC (w/Cheb. Pol.)	• ICs/BCs always analytically satisfied • Accurate Solutions • Low Computational Time	• Non-linear problems can be very sensitive to the initial guesses • It suffers of the curse of dimensionality when solving ODE/PDE problems
PINN	• Expanding the latent solution via NN allows to apply this method to solve high order PDEs (e.g. no curse of dimensionality)	• Many training points required for high accuracy (ICs/BCs not analytically satisfied) • Computational expensive when gradient-based methods are used to train the NN

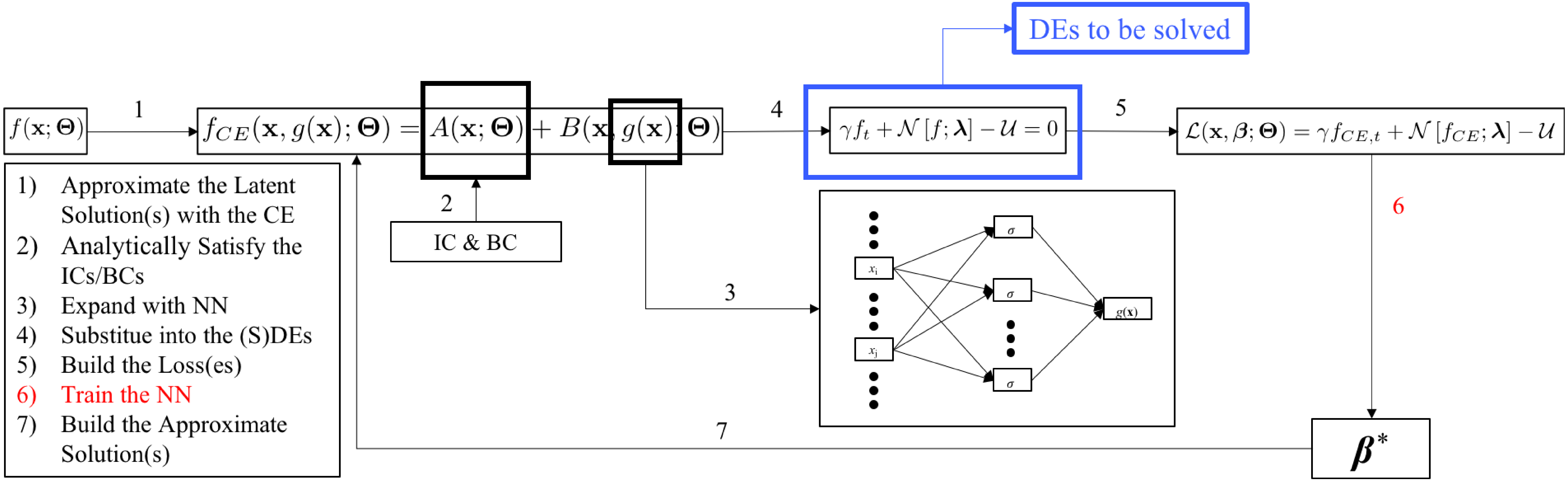
New Methods for Solving DEs (cont'd)

Space Systems Engineering Laboratory

- The Physics-Informed **Extreme Theory of Functional Connections (X-TFC)** is a synergy of the TFC and the standard PINN methods that helps to overcome their limitations for solving DEs
  - X-TFC uses the TFC constrained expression where the free-chosen function is a Single Layer Feedforward NN (SLNN) trained via Extreme Learning Machine (ELM) Algorithm [Huang et al., 2006]



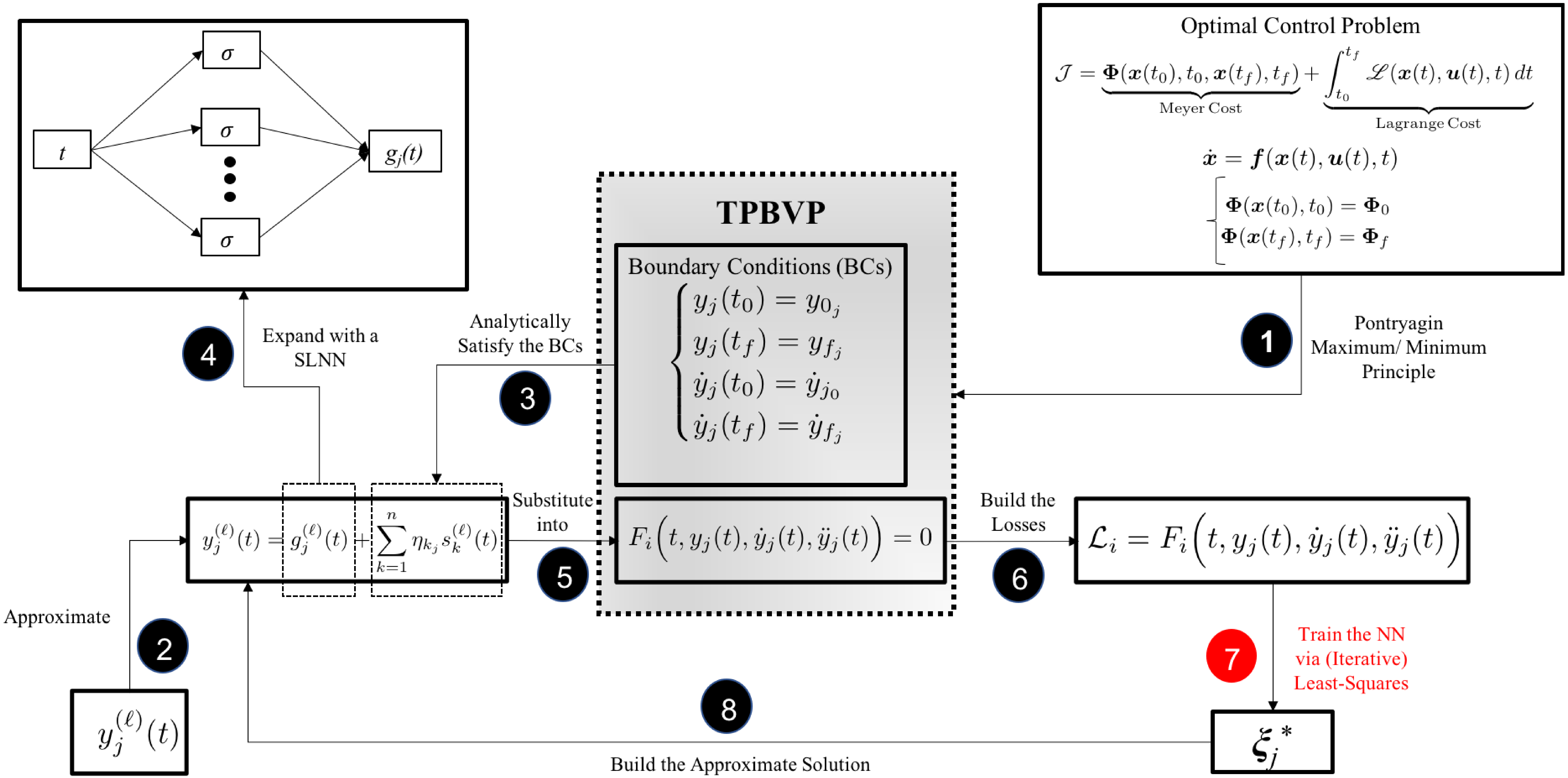
X-TFC approach to solving generic DEs



1) Approximate the Latent Solution(s) with the CE
2) Analytically Satisfy the ICs/BCs
3) Expand with NN
4) Substitue into the (S)DEs
5) Build the Loss(es)
6) Train the NN
7) Build the Approximate Solution(s)

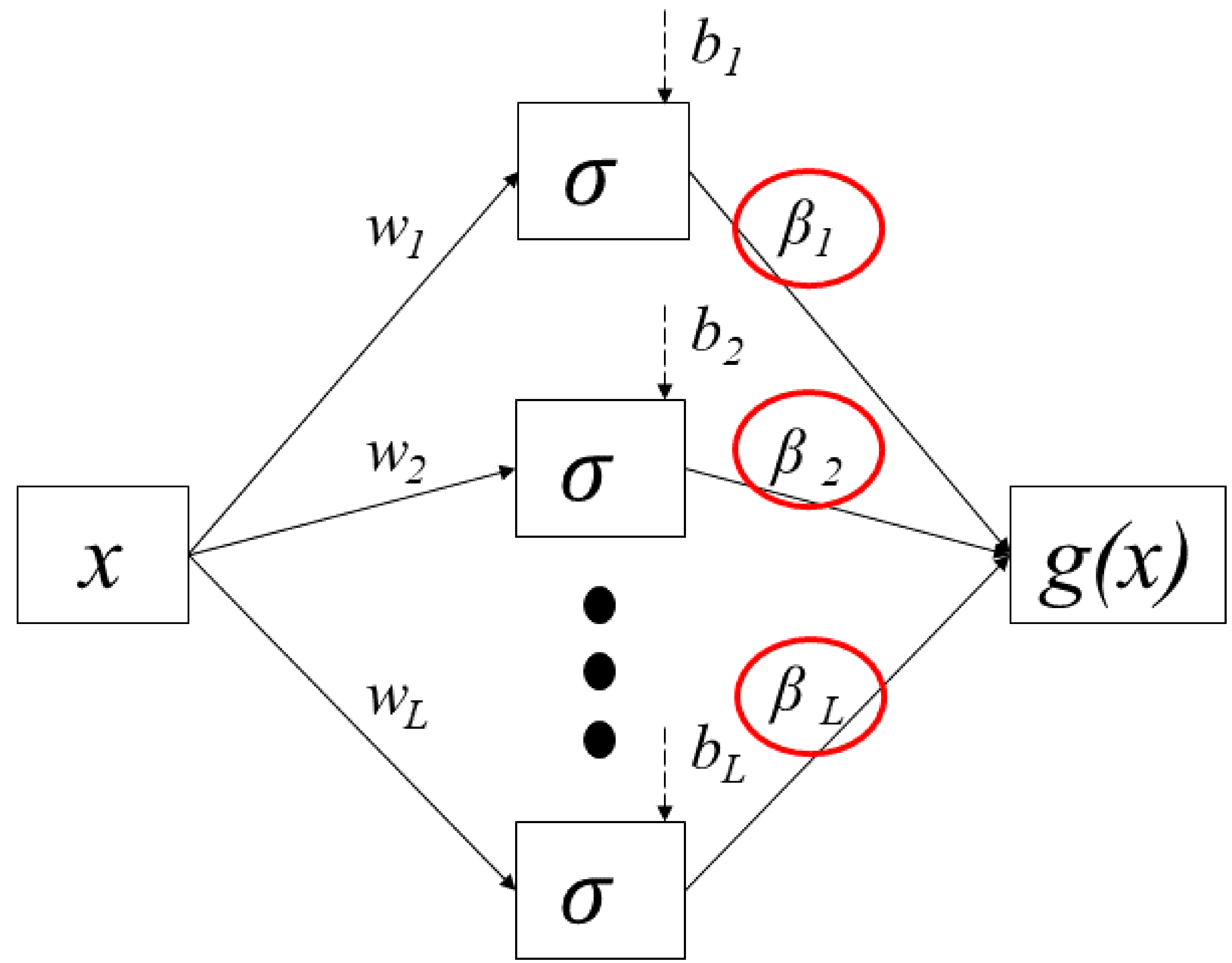
X-TFC approach to solving generic OCPs

Space Systems Engineering Laboratory



ELM algorithm

- ELM is a training algorithm for SLNN that randomly selects input weights and bias, and computes the output weights via least-square
  - Input weights and bias are not tuned during the training
- The convergence of the ELM algorithm is proved by Huang et al. [2006]
  - The convergence is guaranteed for any input weights and bias randomly chosen to any continuous probability distribution



Feldbaum Problem: formulation

Space Systems Engineering Laboratory

- The Feldbaum Problem is a generic OCP that we have chosen as it has analytical solution. This allowed us to perform sensitivity analysis to check the accuracy and the robustness of the proposed physics-informed algorithm. The OCP is posed as following:

$$\min \quad \mathcal{J} = \frac{1}{2}\int_0^1 (f^2 + u^2)\, dt$$

subject to

$$\dot f = \frac{df}{dt} = -f + u$$
$$0 \le t \le 1$$
$$f(0) = 1$$

- Applying the PMP the TPBVP that we will solve via X-TFC is:

$$\begin{cases} \dot f &= \dfrac{\partial H}{\partial \lambda} = -f - \lambda \\ \dot \lambda &= -\dfrac{\partial H}{\partial x} = \lambda - f \end{cases} \quad \text{s.t.} \quad \begin{cases} f(0) = f_0 = 1 \\ \lambda(1) = \lambda_f = 0 \ \text{(transversality condition)} \end{cases}$$

- The CEs and their derivatives are:

$$\begin{aligned} f &= (\boldsymbol{\sigma} - \Omega_1 \boldsymbol{h}_0)^{\mathrm{T}} \boldsymbol{\beta}_f + \Omega_1 f_0 & \dot f &= b^2 \left[(\boldsymbol{\sigma}' - \Omega_1' \boldsymbol{h}_0)^{\mathrm{T}} \boldsymbol{\beta}_f + \Omega_1' f_0 \right] \\ \lambda &= (\boldsymbol{\sigma} - \Omega_1 \boldsymbol{h}_f)^{\mathrm{T}} \boldsymbol{\beta}_\lambda + \Omega_1 \lambda_f & \dot \lambda &= b^2 \left[(\boldsymbol{\sigma}' - \Omega_1' \boldsymbol{h}_f)^{\mathrm{T}} \boldsymbol{\beta}_\lambda + \Omega_1' \lambda_f \right] \end{aligned}$$

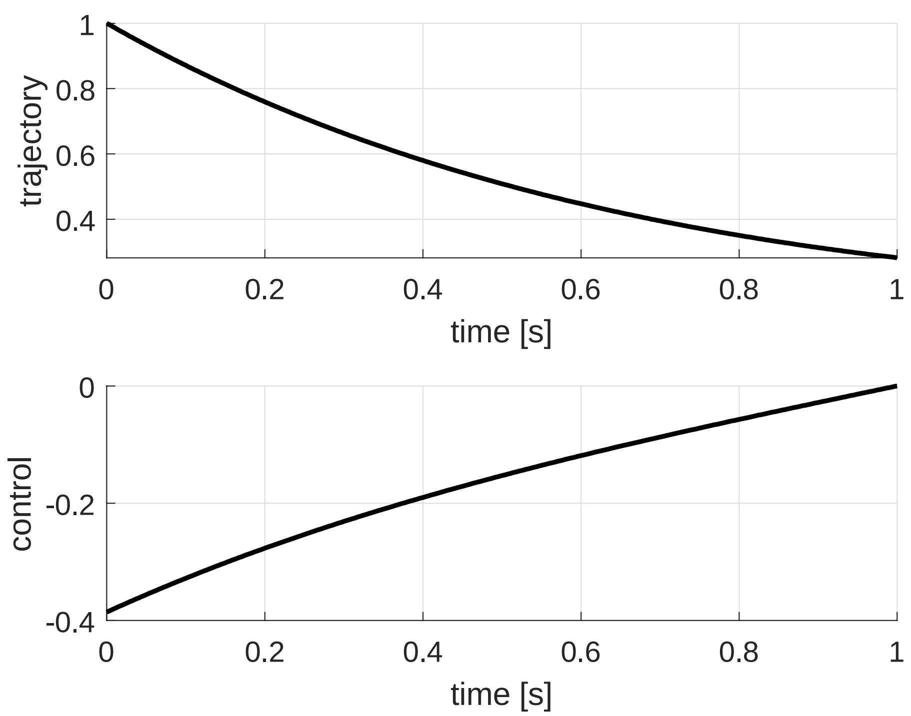
- The unknowns and the losses are:

$$\boldsymbol{\beta} = \{\boldsymbol{\beta}_f \quad \boldsymbol{\beta}_\lambda\}^{\mathrm{T}}$$

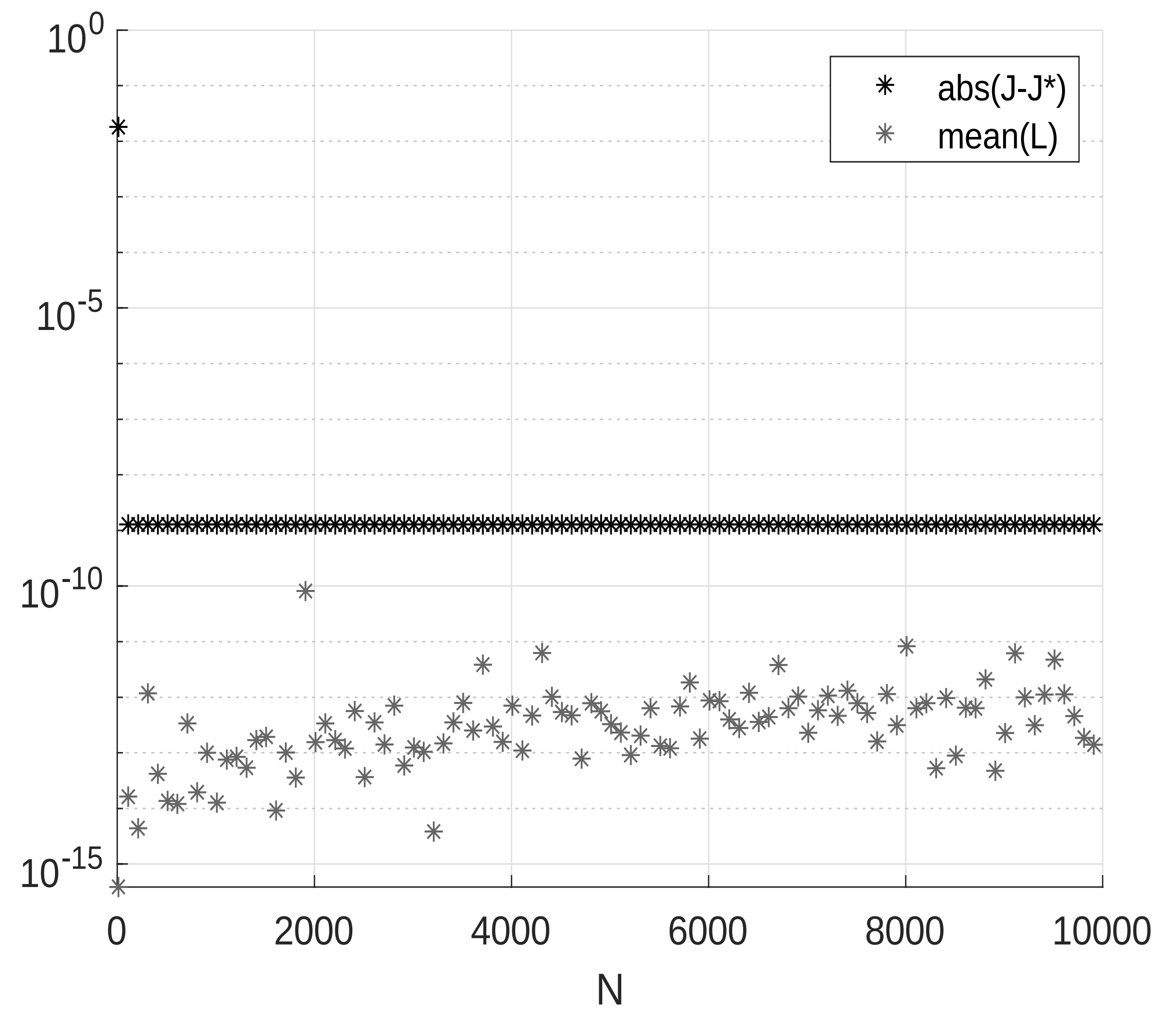
$$\begin{cases} \mathcal{L}_f = \dot f + f + \lambda \\ \mathcal{L}_\lambda = \dot\lambda - \lambda + f \end{cases}$$

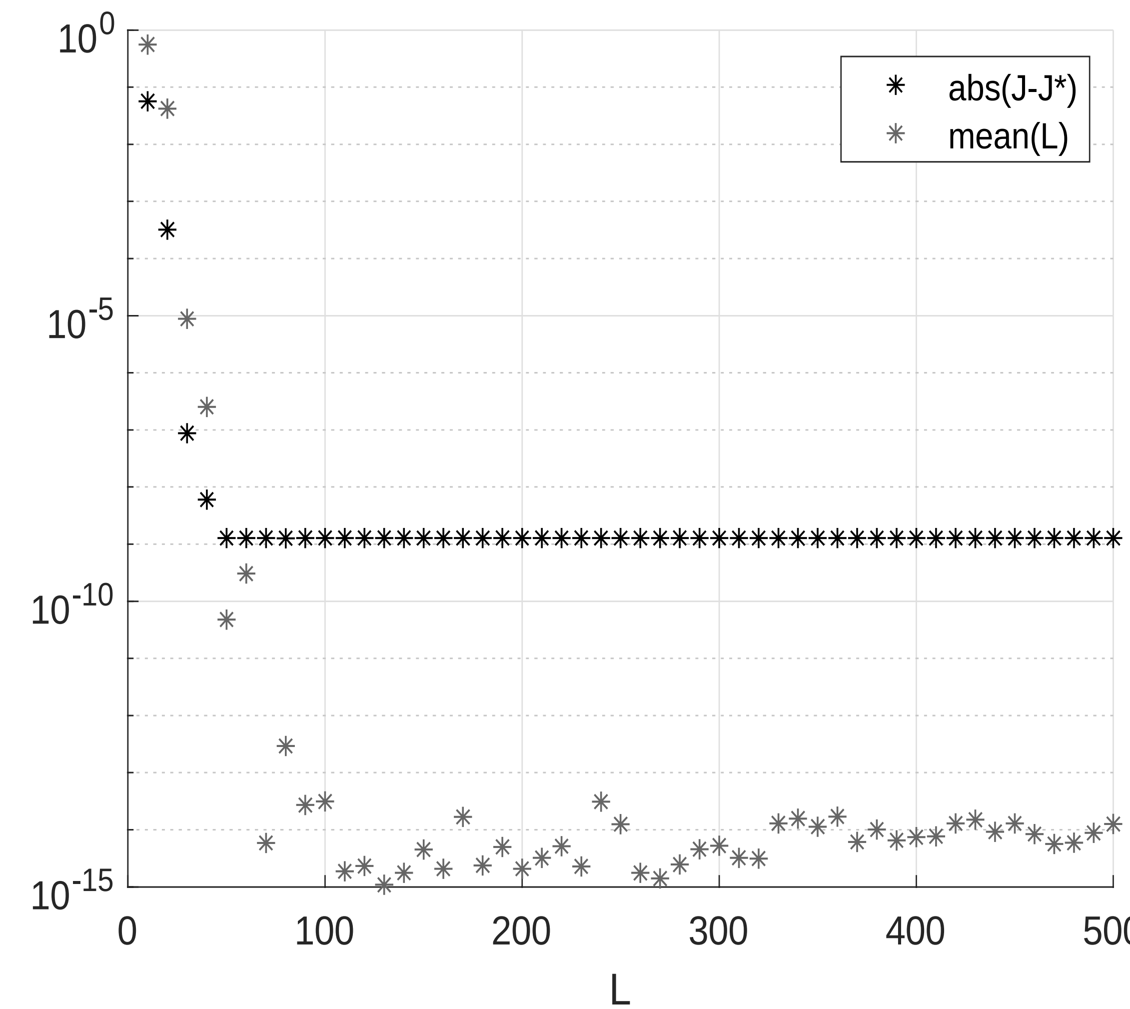
Feldbaum Problem: results

Time evolution of the state and control



Sensitivity analysis: fixed L = 100 (left), fixed N = 100 (right)





Performances analysis

N	L	CPU time [s]	mean($\mathbb{L}$)	$\lvert\mathcal{J}-\mathcal{J}^*\rvert$
10	50	0.0007	4.9×10^{-16}	5.5×10^{-2}
10	90	0.0008	5.1×10^{-16}	6.5×10^{-3}
20	90	0.0008	5.6×10^{-16}	1.5×10^{-3}
50	90	0.001	5.6×10^{-16}	7.3×10^{-8}
200	90	0.004	8.3×10^{-15}	1.3×10^{-9}
500	90	0.006	8.7×10^{-14}	1.3×10^{-9}
100	20	0.0009	1.9×10^{-1}	1.7×10^{-2}
100	50	0.003	1.5×10^{-6}	8.9×10^{-9}
100	100	0.004	1.4×10^{-15}	1.3×10^{-9}
100	150	0.005	8.1×10^{-15}	1.3×10^{-9}

Activation Function: Gaussian
Input weights and bias sampled from: unif [-10,10]

Minimum Time - Energy Optimal Intercept: formulation

Space Systems Engineering Laboratory

- The minimum time-energy optimal intercept problem is posed as following:

$$\min \quad \mathcal{J} = \Gamma t_f + \frac{1}{2}\int_{t_0}^{t_f} (\boldsymbol{a}_M^T \boldsymbol{a}_M)\, dt$$

$$\text{subject to} \quad \begin{cases} \dot{\boldsymbol{r}} = \boldsymbol{v} \\ \dot{\boldsymbol{v}} = \boldsymbol{a}_T - \boldsymbol{a}_M \\ t_0 \le t \le t_f \\ \boldsymbol{r}(t_0) = \boldsymbol{r}_0 \\ \boldsymbol{v}(t_0) = \boldsymbol{v}_0 \\ \boldsymbol{r}(t_f) = \boldsymbol{0} \end{cases}$$

where $\mathbf{r}$ and $\mathbf{v}$ are the relative position and velocity vectors between the target and the interceptor, $\mathbf{a}_T$ and $\mathbf{a}_M$ are the commanded acceleration of the target and the interceptor, respectively.

- Applying the PMP (plus transversality conditions), we get the following TPBVP:

$$\begin{aligned} \dot{\boldsymbol{r}} &= \frac{\partial H}{\partial \boldsymbol{\lambda}_r} &&= \boldsymbol{v} \\ \dot{\boldsymbol{v}} &= -\frac{\partial H}{\partial \boldsymbol{\lambda}_v} &&= \boldsymbol{a}_T - \boldsymbol{\lambda}_v \\ \dot{\boldsymbol{\lambda}}_r &= -\frac{\partial H}{\partial \boldsymbol{r}} &&= \boldsymbol{0} \\ \dot{\boldsymbol{\lambda}}_v &= -\frac{\partial H}{\partial \boldsymbol{v}} &&= -\boldsymbol{\lambda}_r \\ H(t_f) + \Gamma &= 0 \end{aligned} \qquad \text{s.t.} \quad \begin{cases} \boldsymbol{r}(t_0) = \boldsymbol{r}_0 \\ \boldsymbol{v}(t_0) = \boldsymbol{v}_0 \\ \boldsymbol{r}(t_f) = \boldsymbol{0} \\ \boldsymbol{\lambda}_v(t_f) = \boldsymbol{0} \end{cases}$$

- The TPBVP will be solved via X-TFC.

Minimum Time - Energy Optimal Intercept: formulation (cont'd)

Space Systems Engineering Laboratory

- The CEs and their derivatives are:

$$
\begin{aligned}
r_j &= \left(\boldsymbol{\sigma} - \Omega_1\boldsymbol{\sigma}_0 - \Omega_2\boldsymbol{\sigma}_f - \Omega_3\boldsymbol{\sigma}_0'\right)^{\mathrm{T}}\beta_j + \Omega_1 r_{0_j} + \Omega_2 r_{f_j} + \frac{\Omega_3 v_{0_j}}{b^2} \\
v_j &= b^2\left[\left(\boldsymbol{\sigma}' - \Omega_1'\boldsymbol{\sigma}_0 - \Omega_2'\boldsymbol{\sigma}_f - \Omega_3'\boldsymbol{\sigma}_0'\right)^{\mathrm{T}}\beta_j + \Omega_1' r_{0_j} + \Omega_2' r_{f_j} + \frac{\Omega_3' v_{0_j}}{b^2}\right] \\
a_j &= b^4\left[\left(\boldsymbol{\sigma}'' - \Omega_1''\boldsymbol{\sigma}_0 - \Omega_2''\boldsymbol{\sigma}_f - \Omega_3''\boldsymbol{\sigma}_0'\right)^{\mathrm{T}}\beta_j + \Omega_1'' r_{0_j} + \Omega_2'' r_{f_j} + \frac{\Omega_3'' v_{0_j}}{b^2}\right] \\
\lambda_{r,j} &= \boldsymbol{\sigma}^{\mathrm{T}}\beta_{r,j} \\
\dot{\lambda}_{r,j} &= b^2\boldsymbol{\sigma}'^T\beta_{r,j} \\
\lambda_{v,j} &= \left(\boldsymbol{\sigma} - \boldsymbol{\sigma}_f\right)^{\mathrm{T}}\beta_{v,j} + \lambda_{v_{f,j}} \\
\dot{\lambda}_{v,j} &= b^2\boldsymbol{\sigma}'^T\beta_{v,j}
\end{aligned}
$$

The $\boldsymbol{\sigma}$ are the activation functions of the SLNN that is trained via ELM, where $\boldsymbol{\beta}$'s are the *output weights* of the network.

The Ω's are called switching functions, and their expression can be found in the manuscript.

- The unknowns and the losses are:

$$
\Xi = \{\beta_{r,1} \quad \beta_{r,2} \quad \beta_{r,3} \quad \beta_{\lambda_{r,1}} \quad \beta_{\lambda_{r,2}} \quad \beta_{\lambda_{r,3}} \quad \beta_{\lambda_{v,1}} \quad \beta_{\lambda_{v,2}} \quad \beta_{\lambda_{v,3}} \quad b\}^{\mathrm{T}}
$$

$$
\begin{cases}
\mathcal{L}_{a,j} = a_j - a_{T,j} + \lambda_{v,j} \\
\mathcal{L}_{\lambda_{r,j}} = \dot{\lambda}_{r,j} \\
\mathcal{L}_{\lambda_{v,j}} = \dot{\lambda}_{v,j} + \lambda_{r,j} \\
\mathcal{L}_{\lambda_H} = \sum_{j=1}^{3} (\lambda_{r,j} v_j) + \Gamma
\end{cases}
$$

$$
\boxed{b^2 = c = \frac{z_f - z_0}{t_f - t_0}}
$$

Mapping coefficient from t in $[t_0; t_f]$ to z in $[z_0; z_f]$

Minimum Time – Energy Optimal Intercept: results

Space Systems Engineering Laboratory

Performances analysis (Γ=1)

$\boldsymbol{r}_0$ = [500, -600, -500] m
$\boldsymbol{v}_0$ = [-50, 60, 5] m/s
$\boldsymbol{a}_T$ = [1, -2, 0,1] m/s^2 (assumed constant)

N	L	# of iterations	CPU time [s]	mean($\mathbb{L}$)	mean($H+\Gamma$)	$H(t_f)+\Gamma$	t_f [s]	$\mathcal{J}$
20	8	8	0.005	2.2×10^{-6}	2.5×10^{-6}	1.3×10^{-9}	45.54	65.30
20	12	5	0.006	1.7×10^{-7}	3.9×10^{-5}	6.6×10^{-12}	45.54	65.30
20	16	9	0.01	1.3×10^{-9}	2.5×10^{-8}	7.3×10^{-12}	45.54	65.30
30	16	9	0.02	2.6×10^{-9}	2.1×10^{-7}	6.2×10^{-13}	45.54	65.30
30	30	7	0.03	2.0×10^{-10}	7.4×10^{-8}	2.8×10^{-10}	45.54	65.30
30	30	7	0.03	2.0×10^{-10}	7.4×10^{-8}	2.8×10^{-10}	45.54	65.30

Activation Function: hyperbolic tangent
Input weights and bias sampled from: unif [-1,1]

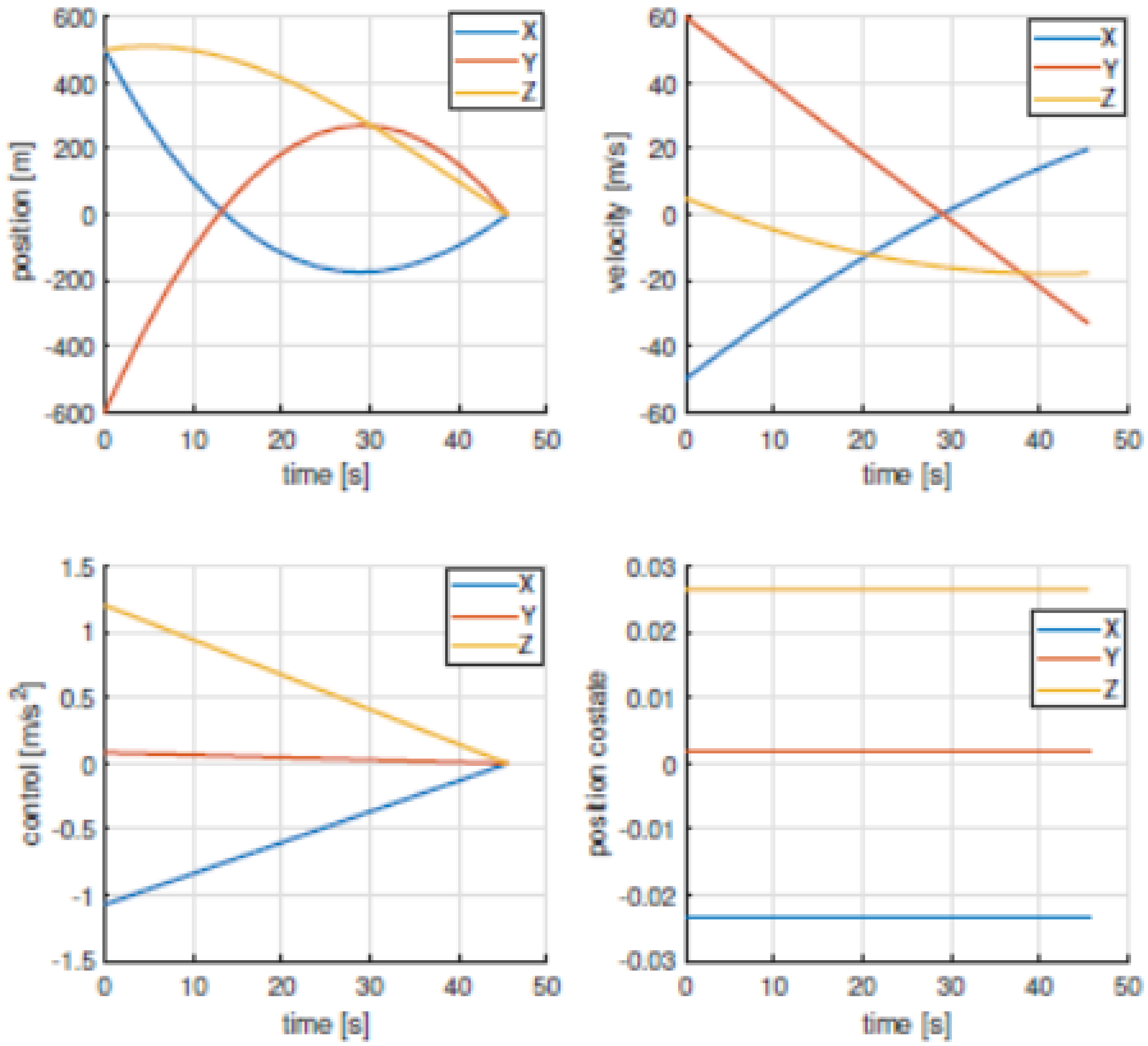
We compare our results with GPOPS. The results obtained with GPOPS were the following: tf = 45.54, H(tf) + **Γ** = 2.35 x 10^{-6}, with a CPU time ~ 1.48 [s]

Minimum Time – Energy Optimal Intercept: results (cont'd)

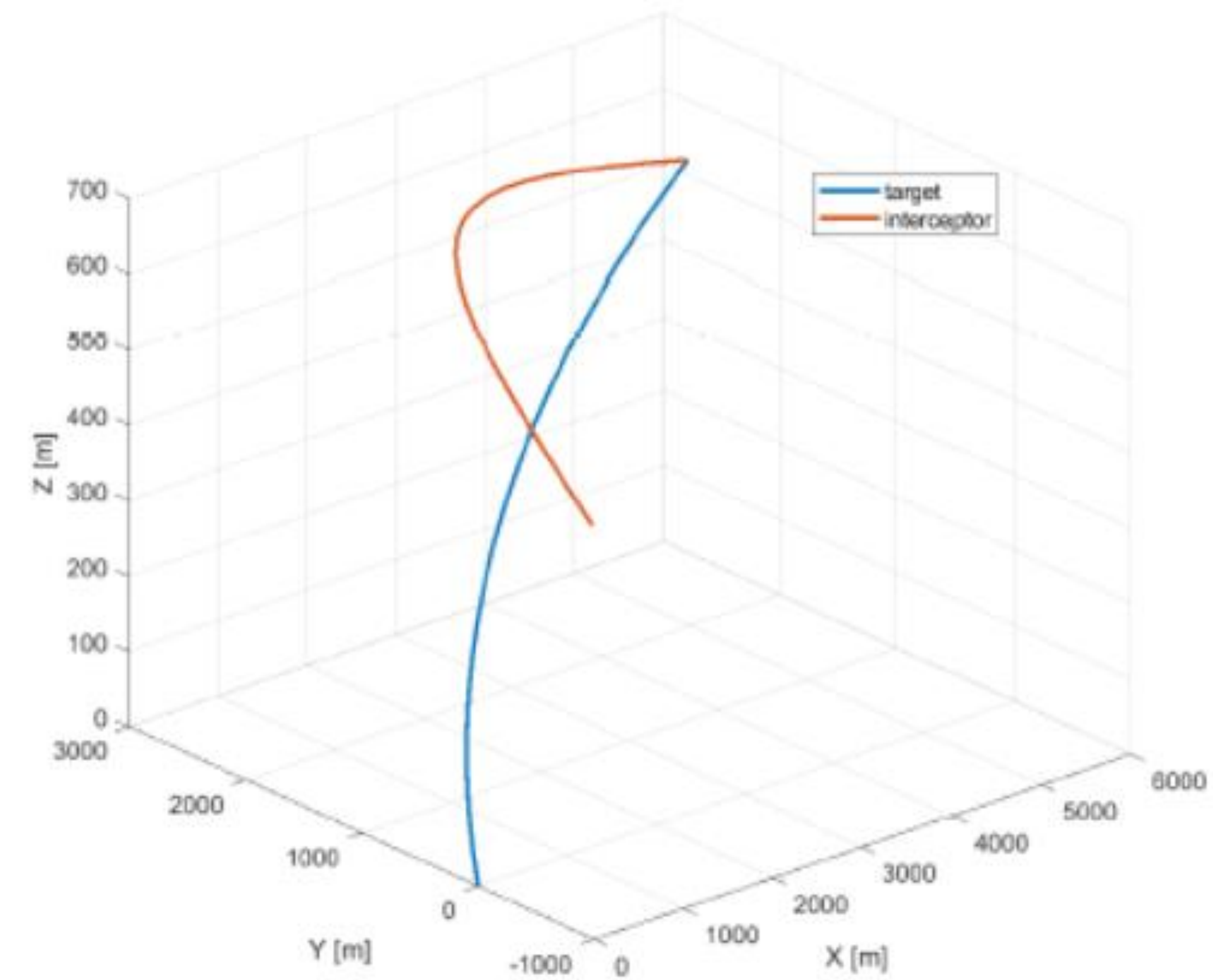
Space Systems Engineering Laboratory

Time evolution of the states and control

$L_2 = 1.3 \times 10^{-9}$



Optimal Trajectories



Activation Function: hyperbolic tangent
Input weights and bias sampled from: unif [-1,1]
Number of points: 20
Number of neurons: 16

Conclusions and Outlooks

- We presented a new algorithm based on the newly developed Physics-Informed X-TFC for solving general OPCs.
  - The physics-informed X-TFC framework is used to solve the TPBVP arising from the application of the PMP.
- The algorithm was tested in designing minimum time – energy optimal intercept trajectories.
  - The CPU time, in order of milliseconds, makes the proposed algorithm suitable for on board applications.
  - The performances are comparable with the state-of-the-art software such as GPOPS II.
- Works are in progress to:
  - Employing the physics-informed X-TFC based algorithm to tackle a wide variety of OPCs (especially OPCs for space guidance, navigation, and control).
  - Use the ability of the X-TFC framework in solving PDEs with high accuracy with a low CPU time to perform real-time computation of closed-loop optimal control via the direct solution of the HJB equation

Thanks for watching =)

Space Systems Engineering Laboratory