a Neural Network Approach to Learn Delay Differential Equations via Pseudospectral Collocation

Dimitri Breda, Muhammad Tanveer*

CDLab – Computational Dynamics Laboratory
Department of Mathematics, Computer Science and Physics – University of Udine
*PhD in Mathematical and Physical Sciences (PNRR)

“MOdellistica Numerica e Data-driven per l’Innovazione sostenibile – MONDI”
(CUP: G25F21003390007)

PINN-PAD
February 22, 2024 @ Padova
introducing DDEs [3, 5, 6]

• let $\tau > 0$, $X := C([-\tau, 0]; \mathbb{R})$ and consider

$$x'(t) = f(x_t)$$

for smooth $f : X \to \mathbb{R}$ and $x_t(\theta) := x(t + \theta)$, $\theta \in [-\tau, 0]$

• e.g., Mackey-Glass [9]:

$$f(\varphi) := \frac{\beta \varphi(-\tau)}{1 + \varphi(-\tau)^n} - \gamma \varphi(0) \implies x'(t) = \frac{\beta x(t - \tau)}{1 + x(t - \tau)^n} - \gamma x(t)$$

(in all the experiments: $\beta = 4, \gamma = 2, n = 9.6, \tau = 1$)

(full references at the end)
the abstract Cauchy problem [4]

• the IVP in $\mathbb{R}$

\[
\begin{align*}
    x'(t) &= f(x_t), \quad t \geq 0 \\
    x(\theta) &= \varphi(\theta), \quad \theta \in [-\tau, 0],
\end{align*}
\]

is equivalent to the IVP in $X$

\[
\begin{align*}
    u'(t) &= A(u(t)), \quad t \geq 0, \\
    u(0) &= \varphi
\end{align*}
\]

through $u(t) = x_t$ for $\varphi \in D(A)$, where $A : D(A) \subseteq X \to X$ given by

\[A(\psi) = \psi', \quad D(A) = \{\psi \in X : \psi' \in X \text{ and } \psi'(0) = f(\psi)\}\]

generates the associated semigroup $\{T(t)\}_{t \geq 0}$ of solution operators

\[T(t) : X \to X, \quad T(t)\varphi := x_t\]

• DDEs give rise to dynamical systems on $X$, hence $\infty$-dimensional:
  - a general approach: reduce (1) to a system of ODEs + tools for ODEs
  - [7] uses Euler in view of applying NODEs and TDNNs to learn DDEs

PseudoSpectral Collocation (PSC)

- let $-\tau = \theta_M < \cdots < \theta_1 < \theta_0 = 0$ be Chebyshev extrema
- for $X_M := \mathbb{R}^{M+1}$ set
  - $R_M : X \rightarrow X_M$, $R_M \psi := (\psi(\theta_0), \psi(\theta_1), \ldots, \psi(\theta_M))$
  - $P_M : X_M \rightarrow X$, $(P_M \psi)(\theta) := \sum_{j=0}^{M} \ell_j(\theta) \psi_j$, $\theta \in [-\tau, 0]$
- discretize $\mathcal{A} : \mathcal{D}(\mathcal{A}) \subseteq X \rightarrow X$
  \[ \mathcal{A} \psi = \psi', \quad \mathcal{D}(\mathcal{A}) = \{ \psi \in X : \psi' \in X \text{ and } \psi'(0) = f(\psi) \} \]
  with $\mathcal{A}_M : X_M \rightarrow X_M$
  \[ [\mathcal{A}_M(\Psi)]_{i=1,\ldots,M} = [R_M(P_M \Psi)']_{i=1,\ldots,M}, \quad [\mathcal{A}_M(\Psi)]_0 = f(P_M \Psi) \]

[1] B., Diekmann, Gyllenberg, Scarabel, Vermiglio – SIADS 2016  \rightarrow f \text{ nonlinear}
from DDEs to ODEs

- let
  \[ U(t) := (U_0(t), U_1(t), \ldots, U_M(t))^T \in \mathcal{X}_M, \quad U_i(t) \approx u(t)(\theta_i) \]

- the abstract Cauchy problem

\[
\begin{cases}
  u'(t) = A(u(t)), & t \geq 0, \\
  u(0) = \varphi
\end{cases}
\]

with

\[ A\psi = \psi', \quad \mathcal{D}(A) = \{ \psi \in \mathcal{X} : \psi' \in \mathcal{X} \text{ and } \psi'(0) = f(\psi) \} \]

is reduced to the \( M + 1 \) ODEs

\[
\begin{cases}
  U_0'(t) = f(P_M U(t)) \\
  U_i'(t) = [D_M U(t)]_i, & i = 1, \ldots, M, \\
  U(0) = (\varphi(\theta_0), \varphi(\theta_1), \ldots, \varphi(\theta_M))^T
\end{cases}
\]

for \([D_M]_{i,j} \coloneqq \ell'_j(\theta_i),  i = 1, \ldots, M, j = 0, 1, \ldots, M\]

- the RHS \( f \) affects only the first ODE: easy to code [8]
• the logistic DDE

\[ x'(t) = rx(t)[1 - x(t - 1)] \]

has equilibrium \( \bar{x} = 1 \) for all \( r \):

– asymptotically stable for \( r \in (0, \pi/2) \)

– unstable for \( r > \pi/2 \)

– Hopf bifurcation at \( r^* := \pi/2 \)

• error in approximating \( r^* \) via MatCont on approximating ODE:
Neural ODEs [10]

NODEs are deep learning operations whose output is defined by the (numerical) solution of an associated ODE:

\[ y' = g(t, y, \theta) \]

Implementation: We use MATLAB tools for DL (e.g., dlarray, dlide45)

[10] Chen, Bettencourt, Rubanova, Duvenaud – NeurIPS 2018
ODEs from DDEs via Euler

grey: ground truth DDE, red: ground truth ODE, blue: NODE
CPU times:
Euler: 9 minutes for 11 nodes, 12 m for 31 nodes
ODEs from DDEs via PSC

grey: ground truth DDE, red: ground truth ODE, blue: NODE

CPU times:
PSC: 9 minutes for 6 nodes, 9m for 11 nodes

TDNNs are capable of handling delayed input sequences.

\[
\begin{align*}
\text{Time} & \quad \text{Sequence} \\
\text{Data} & \quad \text{Learning} \\
\text{NODE} & \quad \text{Extraction} \\
U' = \begin{bmatrix} \text{net}(PU) \\ D_M U \end{bmatrix} \\
\dot{x}(t) & = \text{net}(x_t)
\end{align*}
\]

Assume \(d \ll M\) trainable delays

\[
\begin{align*}
x(t) \\
x(t - \tau_1) \\
\vdots \\
x(t - \tau_d)
\end{align*}
\]

Cost function:

\[
L = \sum_{j=1}^{N} \left\| \hat{x}(j) - x(j) \right\|_1
\]

\[
\frac{\partial L}{\partial \tau_i} = \frac{\partial L}{\partial x(-\tau_i)} \frac{\partial x(-\tau_i)}{\partial \tau_i} = \frac{\partial L}{\partial x(-\tau_i)}(-\dot{x}(-\tau_i))
\]

NODE:

\[
\begin{align*}
\dot{X}(t) & = \hat{A}_M(U(t)) \\
\hat{A}_M(U) & = \begin{bmatrix} \text{net}(PU) \\ D_M U \end{bmatrix}
\end{align*}
\]

NDDE:

\[
\dot{x}(t) = \text{net}(x(t), x(t - \tau_1), \ldots, x(t - \tau_d))
\]

Resume

**ODE:**
\[ u'(t) = \left[ \frac{\beta u_M(t)}{1+(u_M(t))^n} - \gamma u_0(t) \right] D_M u(t) \]

**NODE:**
\[ u'(t) = \left[ W_3 \tanh(W_2 \tanh(W_1 P u(t) + b_1) + b_2) \right] D_M u(t) \]

**NDDE:**
\[ x'(t) = W_3 \tanh(W_2 \tanh(W_1 \begin{bmatrix} x(t) \\ x(t-\tau_1) \\ \vdots \\ x(t-\tau_d) \end{bmatrix} + b_1) + b_2, \]

Here \( P : \mathbb{R}^{M+1} \to \mathbb{R}^{d+1} \) transforms state collocated samples \( (x(t+\theta_0), \ldots, x(t+\theta_M)) \) into delayed states \( (x(t), x(t-\tau_1), \ldots, x(t-\tau_d)) \) and is computed through barycentric interpolation [12] on Chebyshev extrema.

**Info:** simulation horizon = 0.5, batch size = 1000, and update parameters using adaptive moment estimation method

TDNN Results

CPU times:
- Euler: 16 minutes for 21 nodes, 17m for 31 nodes
- PSC: 8m for 8 nodes, 10m for 11 nodes
SINDy [13] for DDEs

- Collect measurement data \( \mathbf{x}(t) \) and (approximate) \( \mathbf{x}'(t) \)
- Construct a library \( \Theta(\mathbf{x}) \) of candidate functions
- Solve \( \mathbf{x}' = \Theta(\mathbf{x})\Xi \) by sparse regression promoting the sparsity of \( \Xi \)

"Expert" SINDy for DDEs [14, 15]

- direct on the DDE
- via library extension to the delayed data-set \( \mathbf{x}(t - \tau_i) \)
- delays assumed to be known

"Pragmatic" SINDy for DDEs

- on ODEs via PSC of the DDE
- via library extension to the collocated data-set \( \mathbf{x}(t + \theta_i) \)
- delays assumed not to be known

[14] Breda, Demo, Pecile, Rozza – IFAC TDS 2022
Expert SINDy

grey: ground truth, black: SINDy, red: samples
Pragmatic SINDy

50 uniform samples M=5

50 random samples, M=5
Concluding remarks

• data-driven methods for DDEs
• Need to explore and understand more
• Challenge: handling of delay terms

ongoing and future:

• SINDy for stochastic DDEs, with R. D’Ambrosio (L’Aquila), D. Conte and I. Santaniello (Salerno)
• extend to structured population models (PNRR)
18th IFAC Workshop on Time Delay Systems
September 24–27, 2024, Udine, Italy

https://tds2024.uniud.it
https://tds2024.uniud.it

Thank you


