11 - POD NEURAL NETWORKS ⟶ reference paper: Non-intrusive reduced order modeling of non-linear problems using neural networks, Hesthaven & Ubbiali - 2018 JcP

The goal of the paper is to provide a fast tool to deal with non-linear problems
↳ with a non-affine structure

↓ How to build this tool?
They avoid projection online!

Problem formulation for non-linear problems.

Given a parameter $\mu \in \mathcal{P}$, find $u(\mu) \in V$ s.t. $g(u(\mu); \mu) = 0$ (1) strong version of the equation
(parametric problem)

i.e. Navie-Stocks $-\Delta u + (u \cdot \nabla u) + \nabla p = f$ (non-linear problem)

when we discretize ⟶ $Au + C(u)u + Bp = f$
there is this dependence that represents the non-linearity

↓

In weak formulation, we have $g(u(\mu), v; \mu) = 0 \quad \forall v \in V$ (2)
(assuming $f=0$, but that in the left-part)

Considering the discretized problem
⟶ $V_\delta \subset V$ and given the parameter $\mu \in \mathcal{P}$, find $u_\delta(\mu) \in V_\delta$ s.t. $g(u_\delta(\mu), v; \mu) = 0 \quad \forall v \in V_\delta$

Algebraically we can build the residual vector $G_\delta(u_\delta(\mu); \mu) = 0 \in \mathbb{R}^{N_\delta}$ where $\dim(V_\delta) = N_\delta$

How G is build? $G_\delta(\vec{u}_\delta(\mu); \mu)_i = g(u_\delta(\mu), \varphi_i; \mu)$ where $\{\varphi_i\}_{i=1}^{N_\delta}$ is a FOM basis

End of FOM level

What should I do at the ROM level?

1. Build the space with POD:
   - Collect snapshots
   - Solve the generalized eigevalue problem over covariance matrix
   - Take eigenvectors related to the N-largest eigenvalues
   - Build the basis matrix
   - Project the matrices & the vectors ⟶ Attention! To build $\mathbb{R}^N \ni G_N(u_N(\mu); \mu) = B^T G_\delta(B u_N(\mu); \mu)$ ISSUE: for each μ I have to assamble again the matrix

Here, we cannot separare the variable, so we cannot perfor this

2. Solve the reduce systhem is NOT fast in N_δ ⟶ Not good

IDEA: can I avoid 2. using NL? We want to avoid the 2. step. How to do that?

POD-NN is an algorithm that exploits the direct projection of the snapshots in the reduce space to creare a map Π^{NN} (Neural Network) s.t.

$\mu^* \longmapsto \Pi^{NN}(\mu^*) = u_N(\mu^*) \in \mathbb{R}^N$

I am learning the reduced coefficients

To build the ner we need a geometric interpreration of a reduced problem

- ROMs build a reduced basis matrix $B \in \mathbb{R}^{N_\delta \times N}$ where $B = [\varphi_1 \mid \dots \mid \varphi_N]$ ⟶ φ_i: basis function
- Let us consider a linear problem: the simplest way to define direct projection

i.e. $\begin{cases} -\Delta u = f & \forall v \in H_0^1(\Omega) \\ \text{boundary condition} \end{cases}$ ⟶ weak formulation $\int_\Omega \rho \nabla u \nabla v \, dx = \int f u \, dy$

This can be seen as minimization problem ↓ FOM $Au = f$ where $A_{ij} = \int_\Omega \nabla\varphi_i \nabla\varphi_j \, dx$ for $\{\varphi_i\}_{i=1}^{N_\delta}$ basis of the FOM

matrix related to the grad-grad operator

↳ H_0^1-semi norm equivalent to H_0^1 norm

Considering, $A = X_\delta \in \mathbb{R}^{N_\delta, N_\delta}$ this is the inner product matrix, that during the labs we called sniffness matrix

↓

A small version of my matrix & vector can be written like that: $B^T A B = A_N$, $B^T f = f_N$

11.1 Algebraic properties of the Galerkin-ROM

The goal here is to find u_N without solving the system

↓

Hence, let us define the vector representation of the error $\boxed{e_\delta(\mu) = u_\delta(\mu) - B^T u_N(\mu)}$

The FOM residual, on the contrary, can be written as $\boxed{r_\delta(\mu) = f - AB\, u_N(\mu)}$ ⟶ take the small vector and project back

↳ depends on u_N and μ

The following hold

1. $Ae_\delta(\mu) = Au_\delta(\mu) - AB u_N(\mu) = f - AB u_N(\mu) = r_\delta(u_N;\mu)$ ⟶ The error goes to zero if $r_\delta(\mu)$ of the FOM solution is zero
2. $B^T A u_\delta(\mu) = B^T f = f_N$
3. $B^T r_\delta(u_N;\mu) = B^T f - B^T AB u_N = f_N - A_N u_N$ ⟶ $B^T r_\delta(u_N,\mu) \to 0$ if B is a "good" representation

$f_N - A_N u_N \to 0$ is the reduced system (RPS)

I do NOT want to solve $f_N - A_N u_N = 0$, but I want to express u_N in terms of $u_\delta(\mu)$ using the properties 1.,2.,3.

Let us take the FOM problem evaluated in u_N in the FOM space

$AB\, u_N(\mu) = f = Au_\delta(\mu)$

(if B is good; by FOM definition)

↓

Now we project, everything, in the reduced space

$B^T AB u_N(\mu) = B^T f = B^T A u_\delta(\mu)$

⟶ Then, $\boxed{u_N(\mu) = (B^T AB)^{-1} B^T A u_\delta(\mu)}$ ⟶ if we have $u_\delta(\mu)$, we have a direct way to compute u_N

↓ IP

and IP is the direct projection of $u_\delta(\mu)$ in the ROM space

Finally, we can train our net, because we can define the loss

$\mu^* \xrightarrow{\pi^{NN}} u_N(\mu^*)$ BUT How to train it?

The input is μ

The output is $u_N(\mu) \in \mathbb{R}^N$

The loss could be $\mathcal{L} = \sum_\mu \frac{1}{N} \| \pi^{NN}(\mu) - u_N(\mu) \| = \sum_\mu \frac{1}{N} \| \pi^{NN} - IP\, u_\delta(\mu) \| = \sum_\mu \frac{1}{N} \| \pi^{NN} - (B^T X_\delta B)^{-1} B^T X_\delta\, u_\delta(\mu) \|$

↳ number of the snapshots; X_δ is the stiffness matrix; basis function matrix related to POD

Hence, now we have all we need!

OFFLINE

- Snapshots collection
- POD
- Train the net ⟶ You're loosing the theorem (including NON linearity) that tells us that the projection is the best thing

\+ I exploit POD

\- The offline phase is longer

ONLINE

- Evaluation of the net: $\mu^* \longmapsto \pi(\mu^*)$

\+ Very fast: give me the parameter μ, I'll give you the representation

\- No control on the accuracy