Designing recurrent cells to enforce stability in real-time dynamical simulations

Louen Pottier, Anders Thorin

Designing recurrent cells to enforce stability in real-time dynamical simulations

POTTIER Louen, THORIN Anders
CEA List, Palaiseau (contact : louen.pottier@cea.fr)

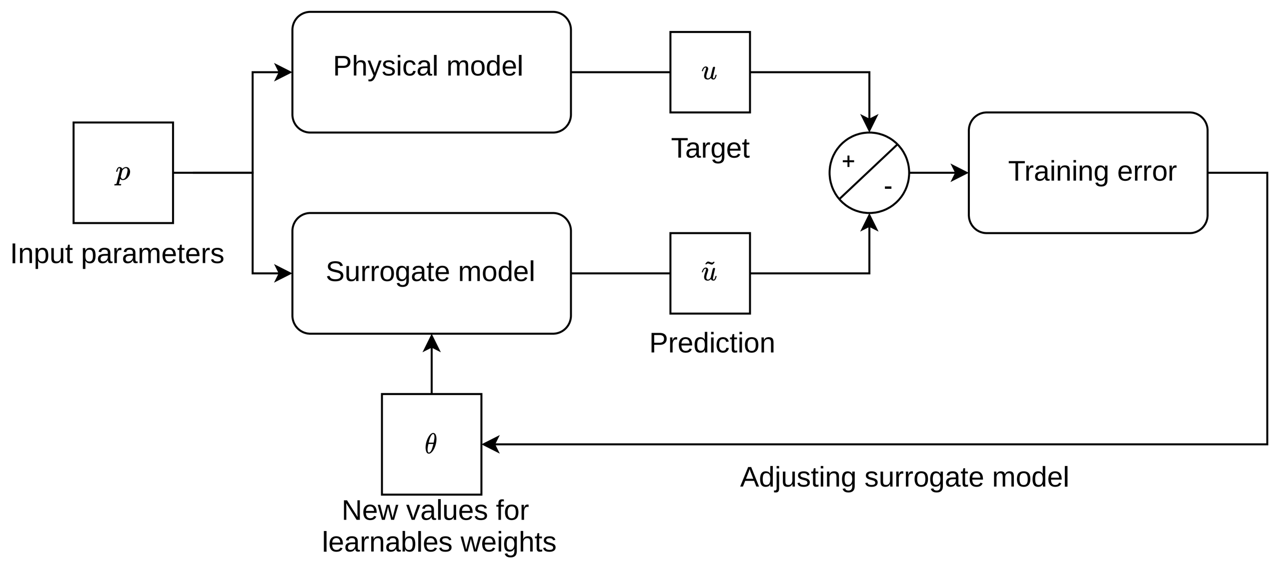
Context

Interactive simulation in structural dynamics

- Interactivity $\Longrightarrow$ evolution of mechanical loading not known in advance
- Need to quickly solve the equations of dynamics

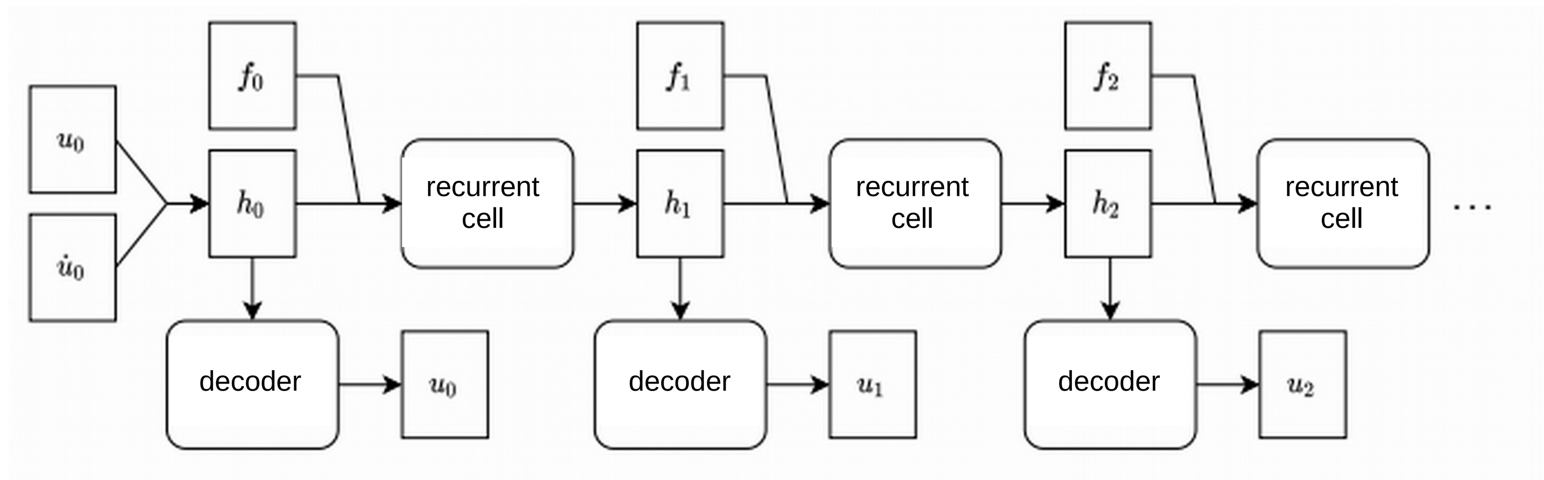
Surrogate model

- Substitution of the slow physical model by a less expensive model trained using some datas generated with the physical model
- « Black box » model, loss of physical sense in favor of speed



Recurrent Neural Networks as a surrogate model

- Encoding the initial conditions in a low-dimensional latent space
- Time recurrence in the latent space
- Decoding the latent variable at each time step



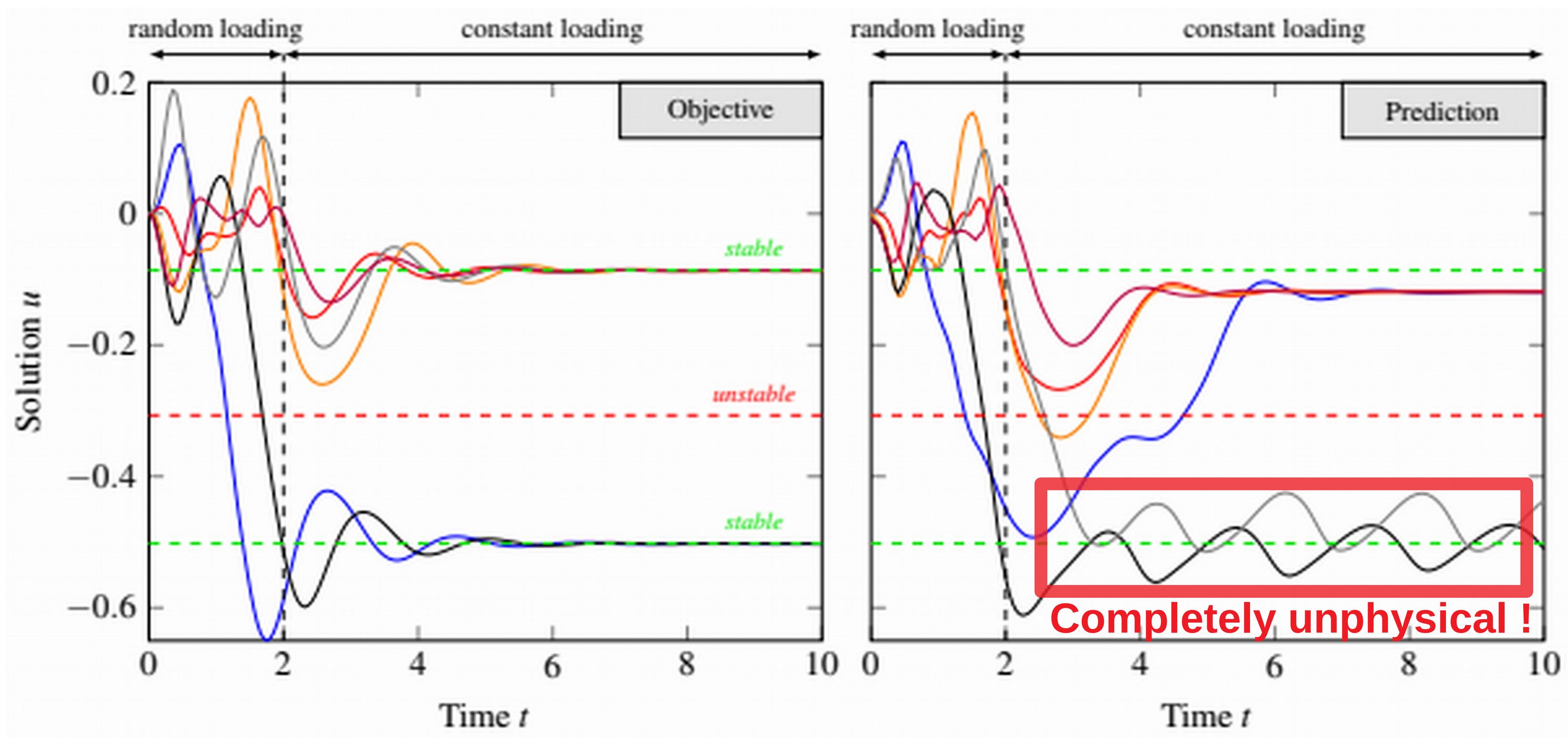
Recurrent cells

- Classical cells : LSTM, GRU, ResNet …
- Must be sufficiently permissive while being conducive to gradient training

Limitations

Stable orbits sometimes learned unintentionally

- For a dissipative dynamical system: no stable orbits, a converging solicitation must lead to a converging solution



Example of RNN having learned stable orbits instead of one of the two equilibrium points

Proposed solution

Recurrent cell architecture structurally incapable of producing stable orbits

- We built a recurrent cell with a learnable internal energy that can never grow without variation of the external solicitation

Continuous cell equation:

$$\begin{cases} E(h,f) = V(h) - h^\top W_f f \\ \dot{h} = A(h)\partial_h E \\ \text{under condition } h_2^\top A(h_1) h_2 \leq 0 \forall h_1, h_2 \end{cases}$$

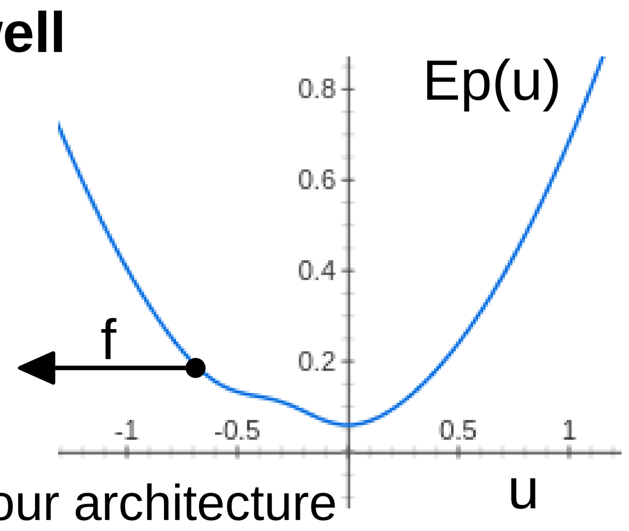
with (for example) :

$$\begin{cases} M(h) = \sum_{i=1}^{n_{pgd}} \mathrm{rn}_i^1(h)\mathrm{rn}_i^2(h)^\top \text{ where } \mathrm{rn}_i^1(h) \text{ and } \mathrm{rn}_i^2(h) \text{ are arbitrary neural networks} \\ A(h) = -\left(M(h) + M^\top(h)\right)^2 + M(h) - M^\top(h) \\ V(h) = \sqrt[n_0]{\prod_{j=1}^{n_0} V_j(h) + 1} - 1 \text{ where } V_1, \ldots, V_{n_0} \text{ are learnable quadratics functions} \end{cases}$$

Experimentations

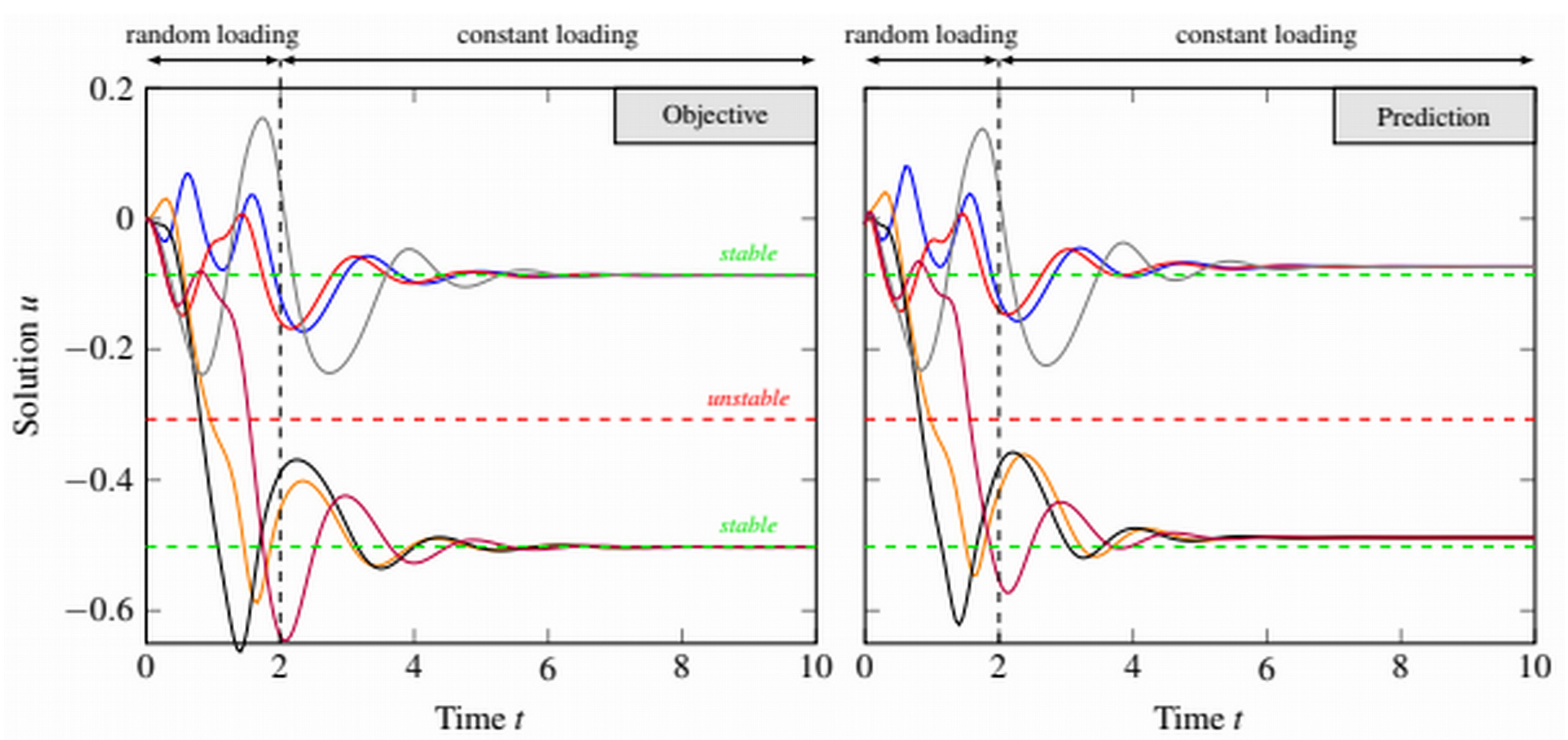
Material point in a non-convex potential well

- Simple 1 dof test case
- We aim to predict the horizontal displacement u(t) given oscillating horizontal force f(t)
- One or two possible equilibrium points depending on the horizontal effort applied



Results

- Both equilibriums can be properly learned by our architecture
- Unlike LSTM and GRU it can never produce stable orbits



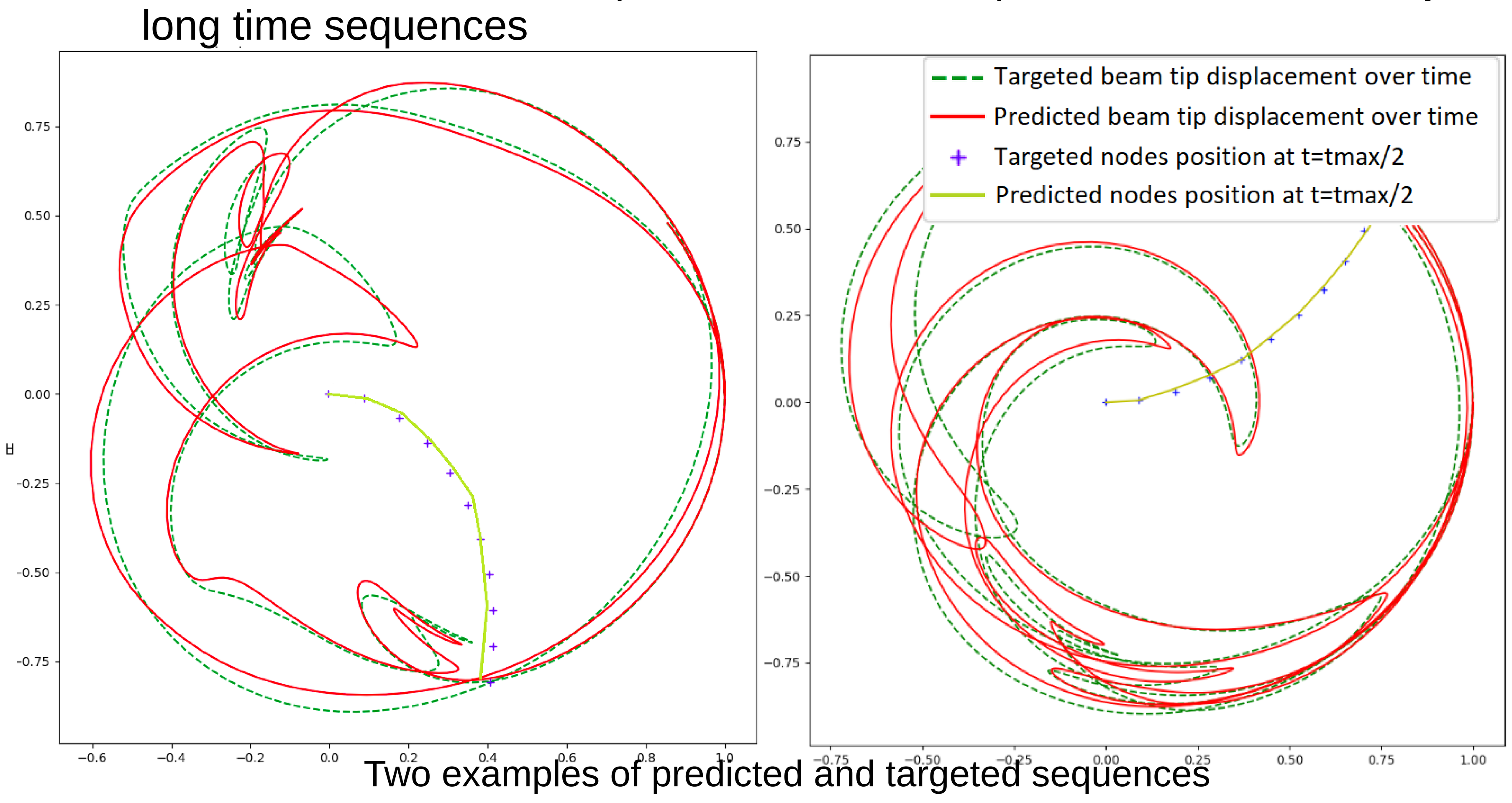
Example of RNN that has correctly learned the two equilibrium points

Hyperelastic beam in finite transformation

- FEM simulation 30 dofs
- We aim to predict the beam displacement over time given a time sequence of high amplitude oscillating force at beam tip

Results

- Our architecture is able to predict the beam displacement even for very long time sequences



Two examples of predicted and targeted sequences

Conclusion and perspectives

- Our architecture has an universal approximator property for hyperelastic structures and a mathematically guaranteed absence of stable orbits
- We were able to train it satisfactorily on two test cases even if the mean error obtained with GRU cells were lower most of the time
- The proposed structure for A(h) and V(h) must be improved to increase the representation power of our architecture