Project Koopman Update

Tom Lu

24 April 2020

Control Performance Analysis

Minimum energy budget method

- For assessing following system's closed-loop control performance:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \quad (1)$$

$$\mathbf{u} = K\mathbf{x} \quad (2)$$

- Where $K = -\lambda^{-1}B^T P$ and P be the solution to the Riccati equation:

$$A^T P + PA - \lambda^{-1}PBB^T P + Q = 0 \quad (3)$$

Theoretical total controller input

We can show that the average total controller input is:

$$\int \langle \mathbf{u}^T \mathbf{u} \rangle_{\mathbf{x}_0} dt = \frac{1}{\lambda} \mathrm{Tr}(P - M) \quad (4)$$

where M is the solution through the Lyapunov equation:

$$(A + K)M + M(A + K)^T + I = 0 \quad (5)$$

Supplementary: Minimum energy budget method derivation

- Begin by writing the closed-loop dynamics as:

$$\mathbf{x}(t) = e^{t(A+BK)}\mathbf{x}_0 \tag{6}$$

- Note that the total cost is the trace of Riccati equation solution:

$$C = \int_0^\infty \mathbf{x}^T\mathbf{x} + \lambda\mathbf{u}^T\mathbf{u} = \mathrm{Tr}(P) \tag{7}$$

- The controller cost can therefore be written as:

$$\int_0^\infty \mathbf{u}^T\mathbf{u} = \frac{1}{\lambda}\left(\mathrm{Tr}(P) - \int_0^\infty \mathbf{x}^T\mathbf{x}\right) \tag{8}$$

- Expanding $\int_0^\infty \mathbf{x}^T\mathbf{x}$:

$$\begin{aligned} \int_0^\infty \mathbf{x}^T\mathbf{x} &= \int_0^\infty \mathrm{Tr}(\mathbf{x}\mathbf{x}^T) & (9) \\ &= \mathrm{Tr}\left(\int_0^\infty e^{t(A+BK)}\mathbf{x}_0\mathbf{x}_0^T e^{t(A+BK)^T} dt\right) & (10) \end{aligned}$$

- As we average over all initial conditions, $\langle \mathbf{x}_0 \mathbf{x}_0^T = I \rangle$:

$$\int_0^\infty \langle \mathbf{x}^T \mathbf{x} \rangle_{\mathbf{x}_0} = \mathrm{Tr}\left(\int_0^\infty e^{t(A+BK)} e^{t(A+BK)^T} dt \right) = \mathrm{Tr}(M) \tag{11}$$

- Where M is the solution to the Lyapunov equation:

$$(A + BK)M + M(A + BK)^T + I = 0 \tag{12}$$

- Finally, substituting in equation 8:

$$\int_0^\infty \langle \mathbf{u}^T \mathbf{u} \rangle_{\mathbf{x}_0} = \frac{1}{\lambda} \mathrm{Tr}(P - M) \tag{13}$$

Control Performance Analysis

Minimum energy budget method

- Problem therefore becomes a root-finding problem, i.e. iteratively searching for $\int \mathbf{u}\mathbf{u}^T dt = \gamma$
- Does this method apply to *DENIS*? Where:

$$\dot{\mathbf{x}} = A(\mathbf{x}_0)\mathbf{x} + B\mathbf{u} \tag{14}$$

- Do equations 3 and 12 become:

$$\langle A^T \rangle_{\mathbf{x_0}} P + P\langle A \rangle_{\mathbf{x_0}} - \lambda^{-1} PBB^T P + Q = 0 \tag{15}$$

$$(\langle A \rangle_{\mathbf{x_0}} + BK)M + M(\langle A \rangle_{\mathbf{x_0}} + BK)^T + I = 0 \tag{16}$$

- Is this equivalent to getting a λ for each $\mathbf{x}_0$, simulate, then plot the average of $\int \mathbf{x}^T \mathbf{x} dt$?
- What happens to the analysis when we have:

$$\mathbf{u} = K(\mathbf{x} - \mathbf{x}_{\text{ref}}) \tag{17}$$

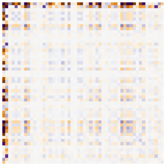
Now we have:

$$\mathbf{x}(t) = \mathbf{x}_0 e^{t(A+BK)} + (I - e^{t(A+BK)})(A + BK)^{-1} BK\mathbf{x}_{\text{ref}} \tag{18}$$

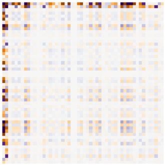
- Recall that:

$$A(\mathbf{x}_0)^T P + PA(\mathbf{x}_0) - \lambda^{-1} PBB^T P + Q = 0 \tag{19}$$

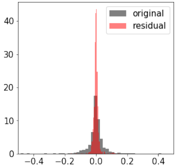
- One problem is in obtaining $\langle P \rangle_{\mathbf{x}_0}$, we can either solve with $\langle A \rangle_{\mathbf{x}_0}$ directly, or we can average over multiple solutions of $P(\mathbf{x}_0)$. Should these two methods should be equal to each other?
- Empirically, they are:



(a) Method I



(b) Method II



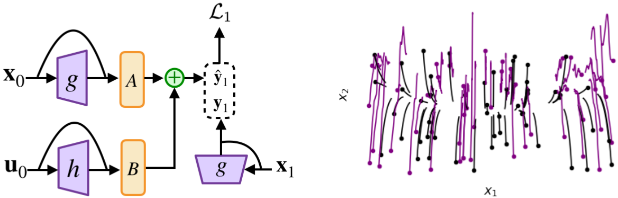
(c) Distributions

Figure: Methods to calculate $\langle P \rangle_{\mathbf{x}_0}$

DEINA: Principled or Not?

Toy example derivation

- From experience, *DEINA* has been a lot harder to train
- Let's start from a toy system for which **we know** the Koopman eigenfunctions, and see whether *DEINA* can discover them correctly
- Recap on **DEINA** architecture:



(a) **DEINA** Architecture (b) Toy trajectories

- We choose the **modified** toy system (purple is driven):

$$x_1 = \mu x_1 \tag{20}$$

$$x_2 = \lambda(x_2 - x_1^2) + u^2 \tag{21}$$

DEINA: Principled or Not?

Toy example derivation

- Denote the input vector as $\mathbf{u} = [0, u]^T$, we can show the **continuous** Koopman transform of this system is:

$$\dot{\mathbf{y}} = \begin{bmatrix} \mu & 0 & 0 \\ 0 & \lambda & -\lambda \\ 0 & 0 & 2\mu \end{bmatrix} \mathbf{y} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ u^2 \end{bmatrix} \tag{22}$$

$$\dot{\mathbf{y}} = A\mathbf{y} + B\tilde{\mathbf{u}} \tag{23}$$

- Where $\mathbf{y} = [x_1, x_2, x_1^2]^T$. Using Euler approximation, equation 23 can be written in **discrete-form** as:

$$\mathbf{y}_{k+1} = (A\delta t + I)\mathbf{y}_k + B\delta t\tilde{\mathbf{u}}_k \tag{24}$$

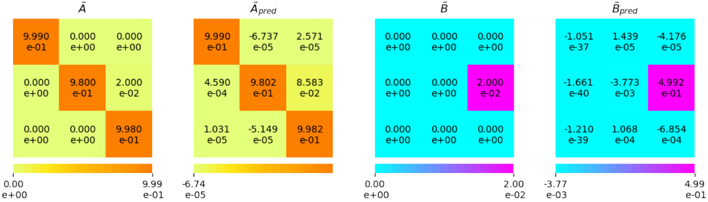
$$= \tilde{A}\mathbf{y}_k + \tilde{B}\tilde{\mathbf{u}}_k \tag{25}$$

- Which means, *DEINA* only needs to learn the $\tilde{A}$, $\tilde{B}$ matricies, **jointly** with the transformations $g(\mathbf{x}) = x_1^2$ and $h(\mathbf{u}) = u^2$

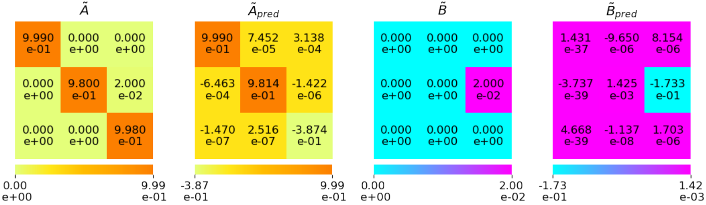
DEINA: Principled or Not?

Toy example results

- We train a **simple** network, where both encoders $g()$ and $h()$ have 2 small hidden-layers, i.e. their architecture is $[2, 50, 50, 1]$. Together with 18 total parameters for $\tilde{A}$ and $\tilde{B}$.
- This is hardly 1000 parameters, can we get **perfect** reconstruction?
- **Sometimes**, but it is **very difficult!**
- Example of near-perfect results ($\mathcal{L} = 1e - 5$):



- It is however, a lot more common for the model to be stuck at local minima ($\mathcal{L} = 6e-4$):



- Compare the **learnt** perfect and imperfect reconstructions:

(c) Perfect eigenfunctions

(d) Imperfect eigenfunctions

DEINA: Principled or Not?

Toy example results

- We see just from the toy example, with less than 1000 parameters, local minima's loss is an order greater than the global minima
- In fact, the "perfect" reconstruction is trained over 800 epochs, and many initializations
- **Hypothesis**: there is a "race" between learning the eigenfunctions and the Koopman matrix, the Koopman matrix is further downstream, therefore gets "fixed" relatively easily
- Potential solution: variable per layer learning rates?
- **TODO**: Keep on investigating control performance analysis, better way to train *DEINA*. *DEINA* clearly has the potential to learn input non-affine systems, it is just a harder network to train.