Lecture 27

Modeling (2):
Control and System Identification

Outline

- ANN based Nonlinear Control
  - Problem Formulation
  - Network Inversion
  - Reinforcement Learning

System Identification Problem

Consider an unknown system (Plant) with output y(t) which depends on current and past input u(t).

- System Identification Problem –

  Given: input u(t) and output y(t), $0 \le t \le t_{max}$,

  Find T[•]

  such that $\hat{y}(t) = T[u(t)] \rightarrow y(t)$

Control Problem

Given: desired output $y^*(t)$, $t_1 \le t \le t_2$

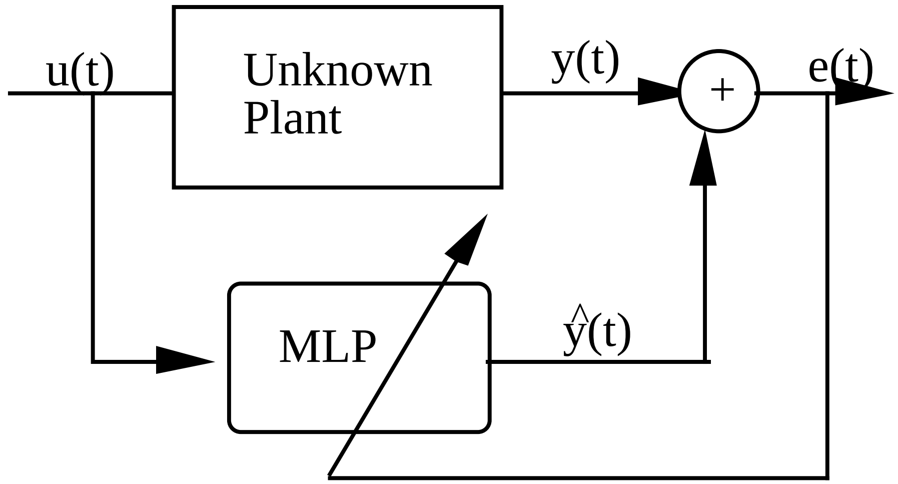
Find: input $u(t)$, $t_0 \le t \le t_2$ $(t_0 \le t_1)$

such that $y(t) \longrightarrow y^*(t)$ for $t_1 \le t \le t_2$

- <u>Path-Following Control Problem</u> – Entire tragectory of the desired output sequence is specified ($t_1 \sim t_0$)
- <u>Reinforcement Learning Problem</u> – Only the destination is given. The intermediate path is not specified ($t_1 >> t_0$).

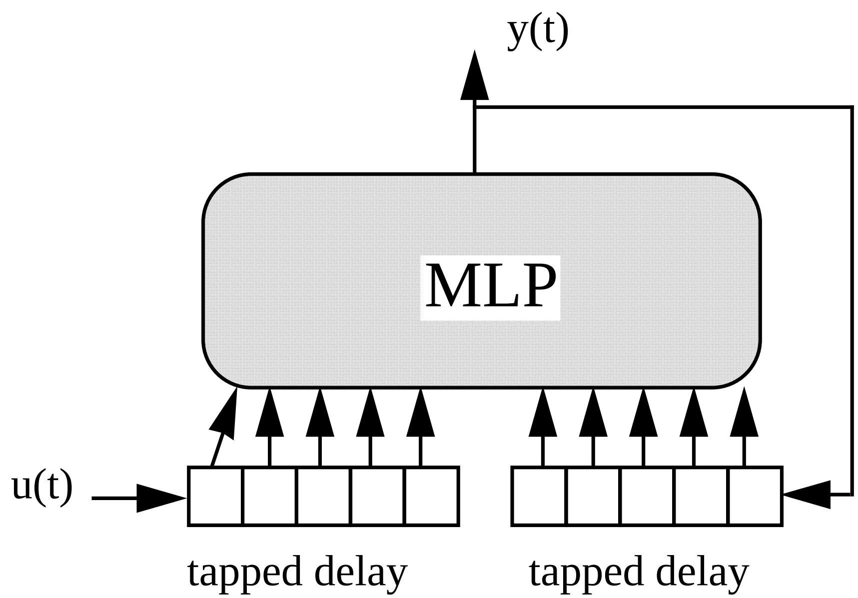
System Identification

- With the same input {u(t)}, find a mathematical model, in this case, a MLP, which will best approximate the output sequence.



- Essentially, a function approximation problem. Due to the particular dynamics of the plant, recurrent ANN are often considered.

MLP for System Identification

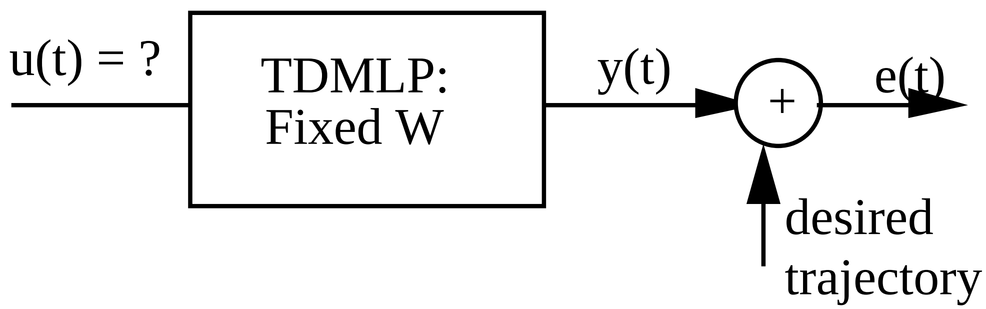


y(t) = F(y(t-1), • • •, y(t–M), u(t), u(t–1), • • •, u(t–N))

- Past outputs are used as "states".

Network Inversion

- Assume y(t) = g(W, u(t), •••, u(t-p),y(t-1), •••, y(t-q)), given d(t+1), and fix W, what should be u(t+1)?



Since

d(t+1) = g(W, u(t+1), •••, u(t-p+1),y(t), •••, y(t-q+1))

Network Inversion (Cont'd)

- We use a gradient descent method to find u(t+1):
- Initially, u(t+1,0) = u(t), compute $\hat{y}(t+1,0)$

  Update u(t+1,m) iteratively using the formula

$$u(t+1,m+1) = u(t+1,m) - \eta \frac{\partial \mathbf{E}}{\partial u(t+1,m)}$$

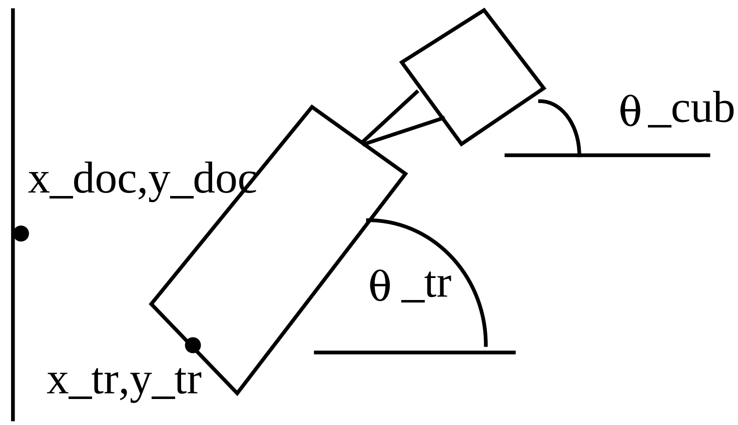
- This method is called Network inversion because it finds the input for given output.
- Applications: Robot arm manipulation, query learning.

Reinforcement Learning

- No teacher to show how to proceed or what was wrong.
- Often only a "success" or "failure" indicator is available after a long sequence of control steps.
- Examples: Game playing, Trailer loading duck backing, multiple-step time series prediction
- Credit Assignment Problem:
  - Which step is to blame?
  - How the strategy should be changed?

RL Example

Example. (Nguyen and Widraw)

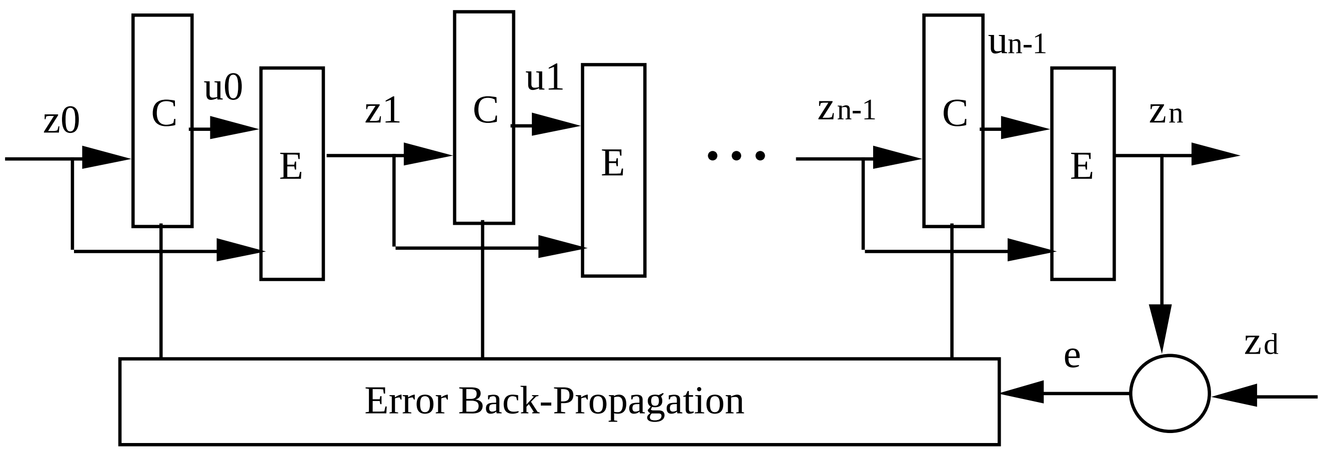


Min. $J = E\{a_1(x_{\_doc} - x_{\_tr})^2 + a_2(y_{\_doc} – y_{\_tr})^2 + a_3\ q_{\_tr}{}^2\}$

Starting from arbitrary position, back trailer to the loading duck, match the two dots.

Reinforcement Learning (2)

- Usually a recurrent MLP structure is used for reinforcement learning problems.
- Truck-backing controller structure:



C: controller, E: emulator. zi: state i. Only one copy of C and E exisits. Error back-propagation performed only at the last stage when the iteration completed.