

Report 220311

Suwon Lee[†]

Abstract. 본 보고서에서는 심층신경망을 이용하여 LPV systems에 대한 PID 제어기와 H_∞ 제어기를 튜닝하는 알고리즘을 제안한다. 제안하는 알고리즘을 통해 학습된 심층신경망의 입력은 LPV systems의 scheduling parameter이고 출력은 제어기의 design parameter이다. 심층신경망을 학습하기 위한 손실함수는 제어시스템의 폐루프 시간응답에 대한 성능지수로 설계한다. 즉 손실함수를 통해 PID 제어기 및 H_∞ 제어기 설계과정에서 직접 고려하기 어려운 시간응답에 대한 성능지수를 고려할 수 있으므로, 제안한 알고리즘은 기존 제어기 튜닝과정에서의 한계점을 극복하였다고 할 수 있다. 본 보고서에서는 알고리즘의 설계방법에 대해 다루며, 알고리즘을 구현하고 평가하는 내용은 다음 보고서에서 다룬다.

1. Introduction

- 본 보고서에서는 LPV systems에 대한 PID 제어기와 H_∞ 제어기를 고려한다.
- LPV system 예시로는 모평항공기의 short period-mode dynamics를 사용한다.
- 본 보고서에서 제안하는 알고리즘은 주어진 시뮬레이션 시간응답으로부터 성능지수를 산출하고, 산출된 성능지수를 바탕으로 제어기의 설계변수¹를 튜닝하는 알고리즘이다.
- 따라서 알고리즘을 통해 튜닝된 제어기의 성능은 시간응답에 대한 성능지수를 어떻게 설계하느냐에 직접적인 영향을 받는다.

2. Controller Description

2.1. PID Controller

PID 제어기 $K(s)$ 는 전달함수 형태로 다음과 같이 설계된다.

$$K(s) = k_P + k_I \frac{1}{s} + k_d \frac{s}{T_f s + 1} \quad (1)$$

2.2. H_∞ Controller

H_∞ 제어기는 선형최적제어기의 일종으로, 폐루프 시스템 전달함수의 ∞ -norm이 주어진 $\gamma \geq 0$ 값보다 작거나 같아지도록 보장된다. 다음과 같은 시스템을 고려하자.

$$\begin{aligned} \dot{x} &= Ax + Bu + Ew \\ z &= Cx + D_1u + D_2w \end{aligned} \quad (2)$$

여기서 x 는 플랜트와 loop shaping filter를 포함한 전체 시스템의 상태변수이고, u 는 제어기에서 도출되는 제어명령, w 는 외부에서 주어지는 입력이며, z 는 시스템 출력 y 과 loop shaping output을 모두 포함하는 벡터이다. H_∞ 제어시스템의 폐루프 블록선도는 Fig. 1에 나타내었다.

[†]Kookmin University, Republic of Korea..

Email: suwon.lee@kookmin.ac.kr.

¹PID 제어기의 제어개인, H_∞ 제어기의 loop shaping 시 필요한 parameters 등이 포함된다.

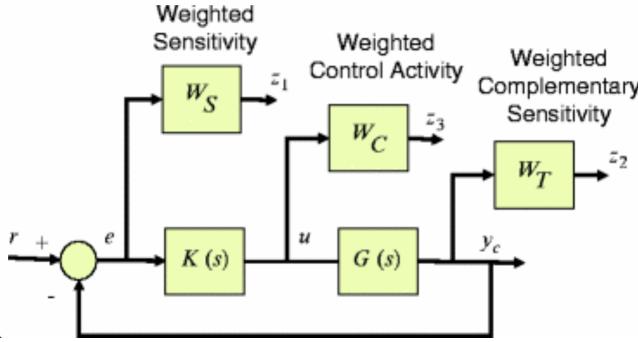


Figure 1. Block diagram of the H_∞ controller with loop shaping

H_∞ 제어기의 비용함수는 다음과 같다.

$$J(u, w) = \frac{1}{2} \int_{t_0}^T (z^T z - \gamma^2 w^T w) d\tau \quad (3)$$

비용함수의 목적은 minimizing control u^* 와 maximizing disturbance w^* 를 찾는 것이다. H_∞ 제어기의 제어명령은 다음과 같이 설계된다.

$$u = - \begin{bmatrix} I_{n_u} & 0 \end{bmatrix} R^{-1} (B^T P + S^T) x = -K_\infty x \quad (4)$$

$$S = \begin{bmatrix} C^T D_1 & C^T D_2 \end{bmatrix}, \quad R = \begin{bmatrix} D_1^T D_1 & D_1^T D_2 \\ D_2^T D_1 & D_2^T D_2 - \gamma^2 I \end{bmatrix} \quad (5)$$

where $D_2^T D_2 < \gamma^2 I$ and there exists a $P \geq 0$ that solves the following ARE:

$$PA + A^T P + C^T C - \begin{bmatrix} B^T P + D_1^T C \\ E^T P + D_2^T C \end{bmatrix}^T \begin{bmatrix} D_1^T D_1 & D_1^T D_2 \\ D_2^T D_1 & D_2^T D_2 - \gamma^2 I \end{bmatrix}^{-1} \begin{bmatrix} B^T P + D_1^T C \\ E^T P + D_2^T C \end{bmatrix} = 0 \quad (6)$$

이제 플랜트와 loop shaping filter로부터 전체 시스템 동역학 식 (2)을 도출하자. 먼저 플랜트 동역학은 다음과 같이 쓸 수 있다.

$$\begin{aligned} \dot{x}_p &= A_p x_p + B_p u \\ y_p &= C_p x_p + D_p u \end{aligned} \quad (7)$$

구동기 동역학을 1차 시스템으로 가정하면 다음과 같이 쓸 수 있다.

$$\dot{u} = -\frac{1}{\tau_a} u + \frac{1}{\tau_a} u_c \quad (8)$$

여기서 τ_a 는 구동기 동역학 시정수이고, u_c 는 구동기 제어명령이다. 구동기 동역학을 포함한 시스템 동역학 방정식은 다음과 같다.

$$\begin{aligned} \dot{x} &= Ax + Bu_c \\ y &= Cx + Du_c \end{aligned} \quad (9)$$

where

$$\begin{aligned} x &= \begin{bmatrix} x_p \\ u \end{bmatrix}, \quad y = \begin{bmatrix} y_p \\ y_a \end{bmatrix} \end{aligned} \quad (10)$$

$$A = \begin{bmatrix} A_p & B_p \\ 0 & -1/\tau_a \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1/\tau_a \end{bmatrix}, \quad C = \begin{bmatrix} C_p & D_p \\ C_a & D_a \end{bmatrix}, \quad D = \begin{bmatrix} 0 \\ E_a \end{bmatrix} \quad (11)$$

Sensitivity weighting filter W_S 의 상태변수방정식은 다음과 같이 표현할 수 있다.

$$\begin{aligned} \dot{x}_s &= A_S x_S + B_S(y - r) \\ z_1 &= C_S x_S + D_S(y - r) \end{aligned} \quad (12)$$

여기서 r 은 추종하고자 하는 신호 명령이고, z_1 은 weighted sensitivity이다. Sensitivity weighting filter의 전달함수 $W_S(s)$ 는 다음과 같이 쓸 수 있다.

$$\begin{aligned} W_S(s) &= \frac{K_S(\tau_S s + 1)}{s} \\ \tau_S &= \frac{1}{2\pi\omega_S} \end{aligned} \quad (13)$$

여기서 K_S 는 계인, ω_S 는 desired loop gain crossover frequency으로, 튜닝 파라미터이다.

Complementary sensitivity weighting filter W_T 의 상태변수방정식은 다음과 같이 표현할 수 있다.

$$\begin{aligned} \dot{x}_T &= A_T x_T + B_T y \\ z_2 &= C_T x_T + D_T y \end{aligned} \quad (14)$$

여기서 z_2 는 weighted complementary sensitivity이다. Complementary sensitivity weighting filter의 전달함수 $W_T(s)$ 는 다음과 같이 쓸 수 있다.

$$\begin{aligned} W_T(s) &= \frac{K_T(\tau_N s + 1)}{\tau_D s + 1} \\ \tau_N &= \frac{1}{2\pi\omega_T} \end{aligned} \quad (15)$$

여기서 K_T 는 계인, ω_T 는 gain crossover frequency이며 τ_D 는 시정수로 튜닝 파라미터이다.

다음으로 D_1 을 injective로 만들고 제어명령의 동역학 u 를 제한하기 위해 다음과 같이 z_3 를 정의한다.

$$z_3 = W_C C_C \dot{u} = W_C C_C(Ax + Bu_c) \quad (16)$$

전체 시스템을 합쳐서 쓰면 다음과 같이 식 (2)와 같은 형태로 쓸 수 있다.

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \dot{x}_S \\ \dot{x}_T \end{bmatrix} &= \begin{bmatrix} A & 0 & 0 \\ B_S C & A_S & 0 \\ B_T C & 0 & A_T \end{bmatrix} \begin{bmatrix} x \\ x_S \\ x_T \end{bmatrix} + \begin{bmatrix} B \\ B_S D \\ B_T D \end{bmatrix} u + \begin{bmatrix} 0 \\ -B_S \\ 0 \end{bmatrix} r \\ \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} &= \begin{bmatrix} -D_S C & C_S & 0 \\ D_T C & 0 & C_T \\ W_C C_C A & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ x_S \\ x_T \end{bmatrix} + \begin{bmatrix} -D_S D \\ D_T D \\ W_C C_C B \end{bmatrix} u + \begin{bmatrix} D_S \\ 0 \\ 0 \end{bmatrix} r \end{aligned} \quad (17)$$

H_∞ 제어기는 설계자에 의해 loop shaping parameter가 주어지고 γ 가 도출되면 최적제어기가 결정되는 형태를 갖는 것을 확인할 수 있다. 따라서 loop shaping filter를 어떻게 설계하느냐에 따라 제어기의 성능이 달라질 수 있다. 그러나 식 (3)와 같이 특정한 형태의 최적화 비용함수만을 사용할 수 있다는 한계점이 존재한다. 한편 최소의 γ 를 도출하는 알고리즘은 일반적으로 γ -iteration과 같은 line search 기법을 사용할 수 있다. 이 때 식 (6)의 Riccati equation을 만족하는지 여부와 폐루프 시스템의 특성방정식의 근이 안정한지 ($\text{Re}(\lambda(A - BK_\infty)) < 0$) 여부를 확인하여 적합한 γ 를 도출해야 한다.

본 보고서에서 제안하는 알고리즘에 의해 학습된 심층신경망은 주어진 scheduling parameter에 대해 최적의 sensitivity parameter, complementary sensitivity parameters, controller weight를 튜닝하게 된다. 이때 사용자가 원하는 형태로 손실함수를 설계하여 심층신경망을 학습할 수 있으므로, 기존의 H_∞ 제어기 설계기법의 장점을 유지하면서 추가적인 최적성능을 도모할 수 있다는 장점이 있다.

3. Deep Neural Network Design

각 제어기에 대한 심층신경망의 입출력 구조는 Table 1과 같다. PID 제어기에 대한 심층신경망 $\mathcal{N}_1(\eta)$ 은 주어진 scheduling parameter에 대해 제어계인과 filter constant를 튜닝한다. H_∞ 제어기에 대한 심층신경망 $\mathcal{N}_2(\eta)$ 은 주어진 scheduling parameter에 대해 loop shaping parameters를 튜닝하며, H_∞ 제어기 설계기법에 의해 튜닝된 loop shaping parameters에 대한 최적의 γ 가 도출된다.

Table 1. Input and outputs of deep neural network for each controllers.

CONTROLLER TYPE	INPUT	OUTPUTS
PID	η (scheduling parameter)	k_P, k_I, k_D (gains), T_f (filter time constant)
H_∞	η (scheduling parameter)	ω_S, K_S (sensitivity parameters), ω_T, K_T, τ_D (complementary sensitivity parameters), W_c (controller weight), γ (upper bound of system norm)

한편, 각 튜닝 파라미터는 그 물리적인 의미에 따라 각각 특정 범위의 값을 갖도록 제한해야 할 필요성이 있다. 이를테면 T_f 는 filter time constant 이므로 $[0, 1]$ 의 범위의 값을 갖도록 할 수 있다. 또한 gain crossover frequency ω_S, ω_T 등은 제어대상 시스템의 물리적 특성과 설계조건에 따라 원하는 주파수 값을 갖도록 제한해야 한다. 이처럼 각 출력변수마다에 대한 범위 제약조건은 해당 변수를 출력하는 심층신경망에서 sigmoid 활성화함수를 사용해 다음과 같이 구현할 수 있다.

$$h = \sigma(y) \times (\bar{h} - \underline{h}) + \underline{h} \quad (18)$$

여기서 $\sigma(\cdot)$ 는 시그모이드 함수, \bar{h}, \underline{h} 는 각각 출력 h 의 상한과 하한이다.

3.1. DNN which returns fixed values for some of its outputs

심층신경망을 이용하여 제어기에 대한 최적화된 튜닝 파라미터를 학습하도록 함으로써 사용자가 직접 시행착오를 거쳐 튜닝하는 번거로움을 줄일 수 있다. 그러나 일부 튜닝 파라미터는 사용자가 특정 값으로 고정하고 해당 파라미터에 대해서 다른 튜닝 파라미터만을 최적화된 값으로 사용하고 싶은 경우가 있을 수 있다.

그러한 경우에는 심층신경망의 입출력 구조가 변경되어야만 한다. 즉, 고정된 값을 갖는 튜닝 파라미터를 입력으로 취급하고 나머지 튜닝 파라미터만을 출력으로 사용하게 된다.

...(중략)

4. Loss Function Design

손실함수는 페루프 제어시스템의 시간응답을 바탕으로 설계된 성능지수이다. 일반적으로 전달함수 형태로 주어지는 제어시스템은 그 특성방정식을 분석하는 방법을 통해 주파수영역에서 안정성 및 성능을 설계할 수 있다. 그러나 주파수영역에서의 분석기법만으로는 시간영역에서의 성능을 직접 조절하는 것이 어렵다. 이를테면 주파수영역 분석기법에서는 특성방정식의 근을 복소평면상의 좌반면에 위치시킴으로써 페루프 시스템의 안정성을 확보할 수 있고, Bode plot 등을 통해 이득여유와 위상여유를 확인함으로써 강건성을 확보할 수 있으며, 최종값정리를 통해 DC gain을 확인할 수 있다. 하지만 주파수영역 분석기법만으로는 제어명령의 상/하한 제약조건, 출력의 overshoot, rise time 그리고 settling time 등과 같은 시간응답에서의 성능지수를 직접 조정하기 어려워 시간영역에서의 분석을 통해 확인하여야만 한다.

본 보고서에서는 수치 시뮬레이션을 통해 도출된 시간영역에서의 페루프 시스템 응답을 평가하는 손실함수를 설계하여 심층신경망을 학습하는 데에 사용한다.

$$J(s) = \alpha_p J_{\text{overshoot}} + \alpha_r J_{\text{risetime}} + \alpha_s J_{\text{settlingtime}} \quad (19)$$

where

$$\begin{aligned} J_{\text{overshoot}} &= \|s.\text{Overshoot}\| \\ J_{\text{risetime}} &= \|s.\text{RiseTime} - T_r^d\| \\ J_{\text{settlingtime}} &= \|s.\text{SettlingTime} - T_s^d\| \end{aligned} \quad (20)$$

여기서 s 는 시뮬레이션을 통해 얻어진 시계열 데이터 및 응답특성을 가지고 있는 데이터이며, T_r^d 는 rise time의 목표값, T_s^d 는 settling time의 목표값이다. $\alpha_1, \alpha_2, \alpha_3$ 는 각 손실함수 성분에 대한 가중치이다. 응답특성들을 계산하는 알고리즘은 Table 2와 같다.

Table 2. Step response characteristics descriptions

CHARACTERISTIC	DESCRIPTION
Overshoot	Percentage overshoot. Relative to the normalized response $y_{\text{norm}} = (y(t) - y_{\text{init}})/(y_{\text{final}} - y_{\text{init}})$, the overshoot is the larger of zero and $100 \times \max(y_{\text{norm}}(t) - 1)$.
RiseTime	Time it takes for the response to rise from 10% to 90% of the way from y_{init} to y_{final} .
SettlingTime	The first time T such that the error $ y(t) - y_{\text{final}} \leq 0.02 \times e_{\text{max}}$ for $t \geq T$, where e_{max} is the maximum error $ y(t) - y_{\text{final}} $ for $t \geq 0$.

5. Dataset Acquisition

심층신경망 학습을 위한 데이터셋은 수치시뮬레이션을 통해 얻는다. 일반적으로 다수의 시뮬레이션을 통해 원하는 조건을 달성하는 매개변수 조합을 찾기 위해서는 몬테카를로 시뮬레이션 방법을 주로 사용한다. 몬테카를로 시뮬레이션 방법에서는 학습 매개변수의 범위를 설정하고, 해당 범위를 일정 간격으로 분할해 매개변수 벡터공간(샘플공간)에서 균일하게 분포한 샘플포인트를 취해 수치시뮬레이션을 수행하거나, 샘플공간에서의 무작위 샘플링을 통해 수치시뮬레이션을 수행하게 된다. 몬테카를로 시뮬레이션 방법은 샘플공간을 균일하게 탐색하므로 샘플공간차원의 개수가 클수록, 매개변수의 범위가 넓을수록 필요한 시뮬레이션 횟수가 늘어난다는 단점을 갖는다. 따라서, 본 연구에서는 능동샘플링 기법을 활용하여 샘플공간을 효율적으로 탐색함으로써 불필요한 시뮬레이션을 피하고 심층신경망을 학습하는데 효율적인 데이터를 우선적으로 취득하도록 한다.

심층신경망 학습을 위한 데이터셋 \mathcal{D} 의 원소 $d_i \in \mathcal{D}$ 를 $d_i = (\eta_i, y_i, h_i)$ 의 순서쌍으로 정의하자. 여기서 η_i 는 LPV systems의 parameter로, scheduling parameter이다. y_i 는 제어기 튜닝 파라미터이고, h_i 는 η_i, y_i 에 대해 수행한 수치 시뮬레이션 결과이다. 데이터셋에서 심층신경망의 입력을 η , 출력을 y 로 사용하고 h 로부터 손실함수를 계산할 수 있다.

$$y = \mathcal{N}(\eta), \quad h = \mathcal{S}(\eta, y) \quad (21)$$

여기서 \mathcal{N} 과 \mathcal{S} 는 각각 심층신경망과 수치 시뮬레이션을 수행하는 오퍼레이터이다. 데이터셋은 (η, y) 벡터 공간을 샘플공간으로 하여 데이터를 수집한다. 이 때 샘플공간에서의 무작위 샘플링을 수행하여 데이터셋을 구성하면 몬테카를로 방법을 사용하는 것이다. 본 보고서에서는 Algorithm 1와 같이 능동샘플링 기법을 사용하여 데이터셋을 구성하면서 동시에 심층신경망 \mathcal{N} 을 학습하는 방법을 사용한다.

Algorithm 1 Training DNN with active sampling

Input: available number of simulations n_T , number of initial random samples n_i , number of active samples per iteration n_a .

1: Initial random sampling on (η, y) space, add n_i samples (η_i, y_i) into the sample set \mathcal{X} .

2: Evaluate samples: $h_i = \mathcal{S}(\eta_i, y_i) \in \mathcal{H}$, attain dataset $\mathcal{D} = (\mathcal{X}, \mathcal{H})$, s.t. $(\eta_i, y_i, h_i) \in \mathcal{D}$.

3: iteration count $m \leftarrow 0$

repeat

4: Train neural network $\mathcal{N}(\eta) = y^*$ for the loss function J s.t. $y^* = \underset{y}{\operatorname{argmin}} J(h(\eta, y))$

5: Conduct random sampling in η space to get n_a samples, evaluate $\mathcal{N}(\eta_j) = y_j$

6: Evaluate $h_j = \mathcal{S}(\eta_j, y_j)$ and add new sample to the dataset $\mathcal{D} \leftarrow (\eta_j, y_j, h_j)$.

7: $m \leftarrow m + 1$

until $n_i + (m + 1)n_a > n_T$

Algorithm 1에 의해 초기에 무작위로 η, y 를 샘플링하여 시뮬레이션을 수행, h 를 도출하고 데이터셋 \mathcal{D} 를 구성한다. 초기 \mathcal{D} 를 사용하여 DNN \mathcal{N} 를 학습하면 그 다음부터는 \mathcal{N} 으로부터 능동샘플링과 DNN 학습을 반복한다. 매 반복에서 \mathcal{N} 은 입력으로 주어지는 η 에 대해 최적의 손실함수값을 갖는 시뮬레이션 h 를 도출할 것으로 예상되는 y^* 를 예측하여 출력한다. 각 반복 단계에서 η 에 대한 샘플링은 무작위 샘플링을 수행하고, y 에 대한 샘플링은 DNN \mathcal{N} 을 기반으로 하는 능동샘플링을 수행하게 된다. 이는 η 의 경우 scheduling parameter 이므로 전 영역에서 균일하게 샘플링해야 하는 반면, y 는 튜닝 파라미터이므로 좋은 손실함수 성능을 낼 것으

로 예측되는 값 근처 영역에서 샘플링하는 것이 좋고, 그렇지 않은 영역에서의 값은 불필요한 데이터라고 볼 수 있기 때문이다. 반복 횟수가 늘어날수록 전체 데이터셋 \mathcal{D} 의 원소의 개수가 늘어나므로 DNN \mathcal{N} 의 성능이 점차 좋아진다. 반복에 의해 수행한 총 시뮬레이션 수행횟수 $n_i + mn_a$ 가 횟수제한 n_T 에 도달하기 직전에 반복을 멈추고 최종적으로 학습된 심층신경망 \mathcal{N} 을 제어기 튜너로서 얻는다.

6. Conclusions

본 보고서에서는 LPV시스템에 대한 PID 제어기 및 H_∞ 제어기 튜너를 심층신경망으로 학습하는 알고리즘에 대해 다루었다. 심층신경망의 입력으로는 LPV시스템의 scheduling parameter를, 출력으로는 제어기 튜닝 파라미터를 학습하도록 하였다. 손실함수는 폐루프 제어시스템의 시간응답에 대한 성능지수를 사용하였다. 한편 심층신경망 학습을 위한 데이터셋을 효율적으로 확보하기 위해 능동샘플링 기법을 활용하였다. 제안한 알고리즘에 의해 학습된 심층신경망 기반 제어기 튜너는 기존의 제어기 설계방법에서 직접 고려하기 어려운 시간응답에 대한 성능을 직접 고려할 수 있고, 심층신경망을 통해 제어기 파라미터를 자동으로 튜닝함으로써 성능을 최적화할 수 있다는 점에서 전문가가 각 파라미터를 직접 튜닝해야만 했던 기존의 한계점을 극복하였다.