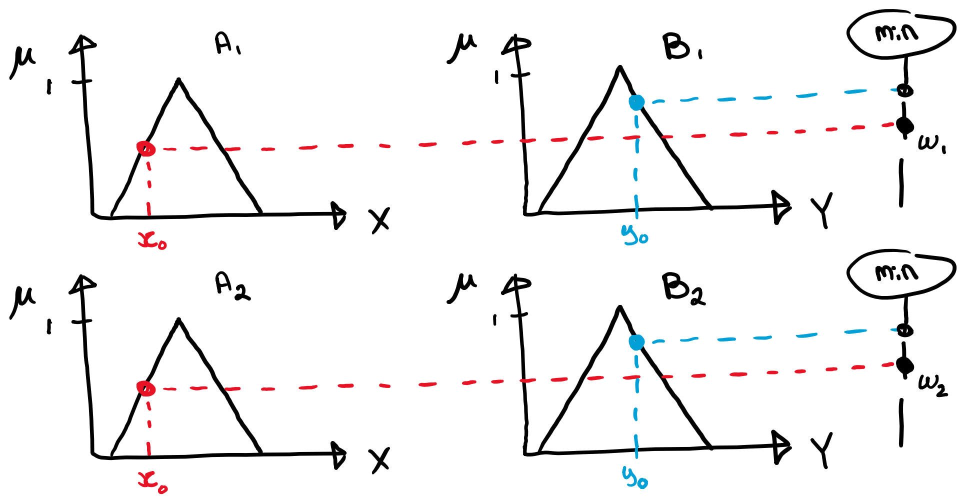
1) Mamdani fuzzy model

$$\frac{\text{max} - \text{min/ product}}{R_1 \cup R_2 \cup \ldots}$$

2) Sugeno fuzzy models

Takagi-Sugeno-Kang (TSK): consequent part of a rule is a polynomial function of inputs.



$z_1 = p_1 x + q_1 y + r_1$

$z_2 = p_2 x + q_2 y + r_2$

Defuzzification:

$$z^* = \frac{w_1 z_1 + w_2 z_2}{w_1 + w_2}$$
$$z^* = \frac{w_1(p_1 x + q_1 y + r_1) + w_2(p_2 x + q_2 y + r_2)}{w_1 + w_2}$$

Type 1: TSK model (1st order)

$$z_1 = p_1 x^1 + q_1 y^1 + r_1$$
$$z_1 = p_1 x + q_1 y + r_1$$

Type 0: (or 0^{th} order TSK)

$$z_1 = p_1 x^0 + q_1 y^0 + r_1$$
$$z_1 = p_1 + q_1 + r_1$$
(consequent part is just a number)
$$z_1 = C_1$$

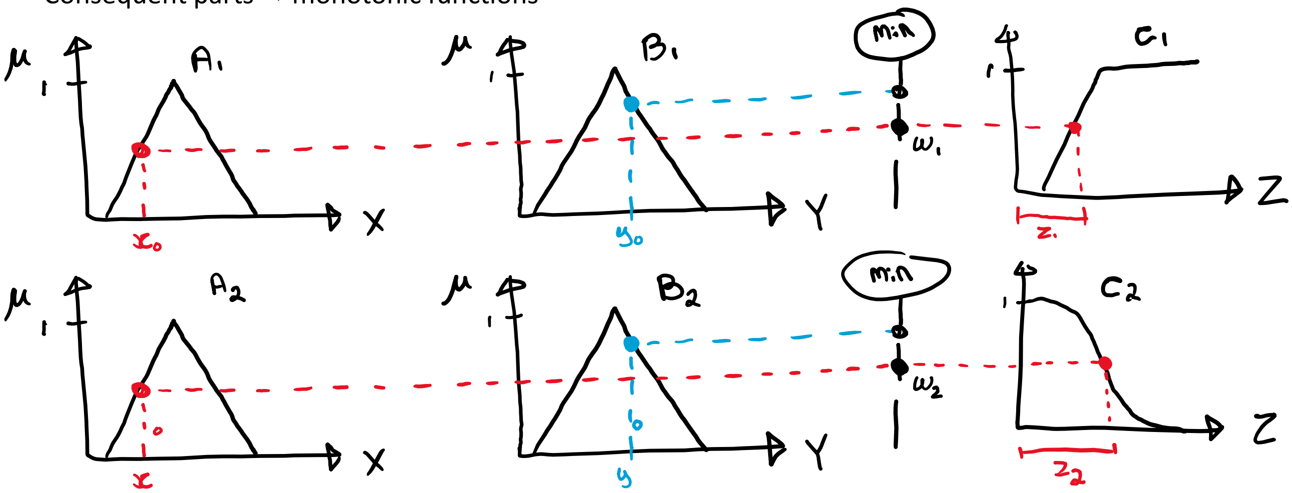
1st order TSK and models are commonly used in modeling (forecasting) applications.

Neuro fuzzy models (NF) are fuzzy model – but they are different from conventional fuzzy systems. They can use machine learning algorithms to update parameters.

3) Tsukamoto fuzzy models

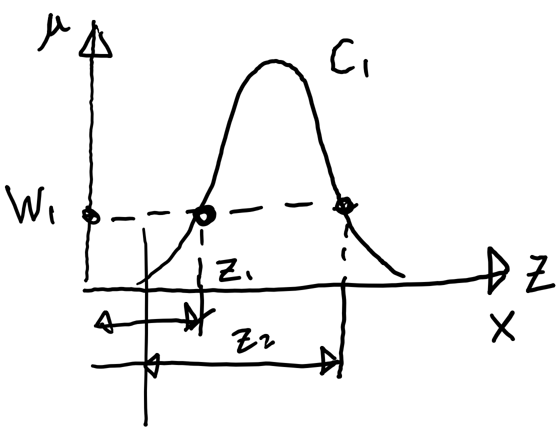
Premise parts → same

Consequent parts → monotonic functions



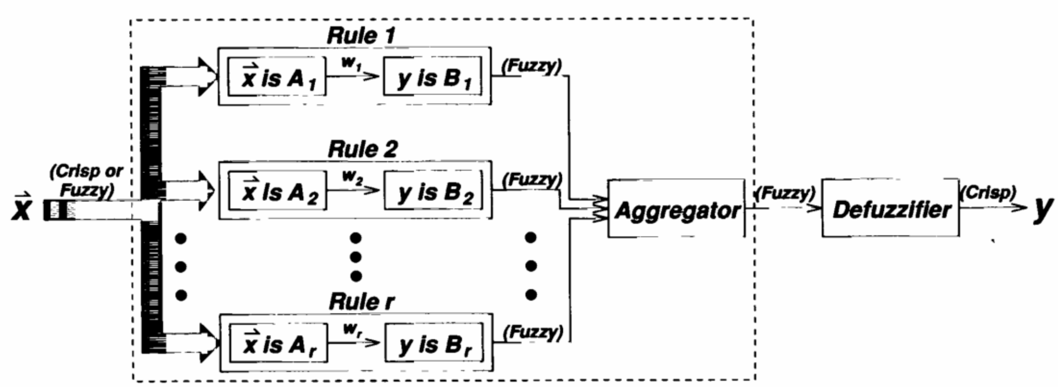
Defuzzification (output):

$$z^* = \frac{w_1 z_1 + w_2 z_2}{w_1 + w_1}$$



(This is why we use monotonic functions - otherwise there is two different results at a single firing strength)

(Read by yourself for more info – Section 4.5, Book 2)



Example 3.1

Consider the room comfort control system schematically shown in Figure 3.3. The temperature (T) and humidity (H) are the process variables that are measured. These sensor signals are provided to the fuzzy logic controller, which determines the cooling rate (C) that should be generated by the air conditioning unit. The objective is to maintain a particular comfort level inside the room.

A simplified fuzzy rule base of the comfort controller is graphically presented in Figure 3.4. The temperature level can assume one of two fuzzy states (HG, LW), which denote high and low, respectively, with the corresponding membership functions. Similarly, the humidity level can assume two other fuzzy states (HG, LW) with associated membership functions. Note that the membership functions of T are quite different from those of H, even though the same nomenclature is used. There are four rules, as given in Figure 3.4. The rule base is:

Rule 1:		If	T	is	HG	and	H	is	HG	then	C	is	PH
Rule 2:	else	if	T	is	HG	and	H	is	LW	then	C	is	PL
Rule 3:	else	if	T	is	LW	and	H	is	HG	then	C	is	NL
Rule 4:	else	if	T	is	LW	and	H	is	LW	then	C	is	NH
	end	if											

The nomenclature used for the fuzzy states is as follows:

Temperature (T)	*Humidity* (H)	*Change in cooling rate* (C)
HG = High	HG = High	PH = Positive high
LW = Low	LW = Low	PL = Positive low
		NH = Negative high
		NL = Negative low

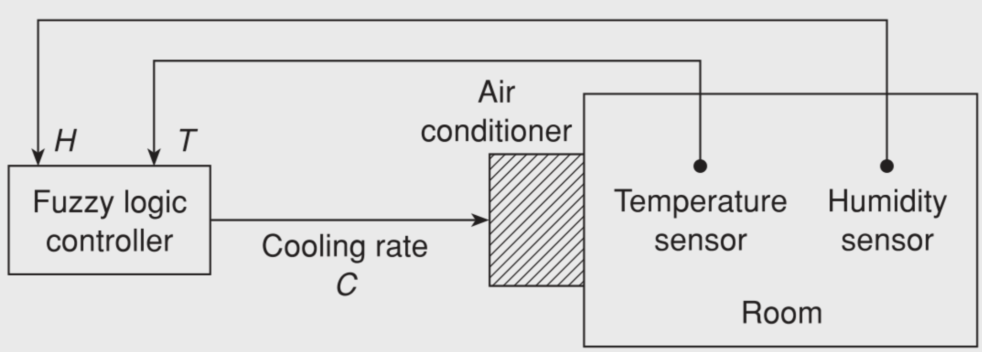
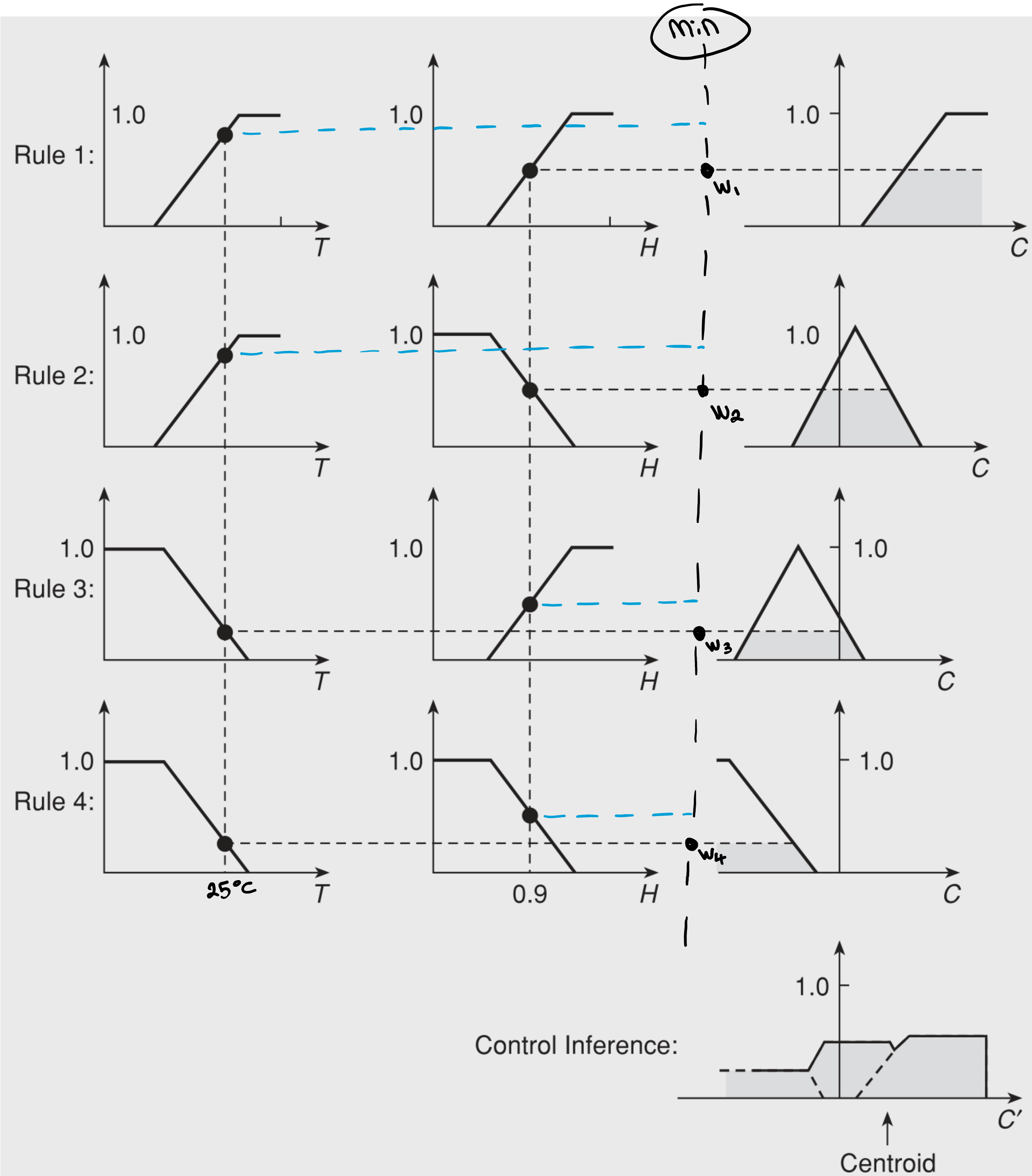


Figure 3.3: Comfort control system of a room


min
Rule 1:
1.0
T
1.0
H
w_1
1.0
C
Rule 2:
1.0
T
1.0
H
w_2
1.0
C
Rule 3:
1.0
T
1.0
H
w_3
1.0
C
Rule 4:
1.0
25°C
T
1.0
0.9
H
w_4
1.0
C
Control Inference:
1.0
C′
Centroid

Example 3.2

A schematic diagram of a simplified system for controlling the liquid level in a tank is shown in Figure 3.8(a). In the control system, the error (actually, correction) is given by

$$e = \text{Desired level} - \text{Actual level}.$$

The change in error is denoted by Δe. The control action is denoted by u, where $u > 0$ corresponds to opening the inflow valve and $u < 0$ corresponds to opening the outflow valve. A low-level direct fuzzy controller is used in this control system, with the control rule base as given in Figure 3.8(b).

The membership functions for E, ΔE, and U are given in Figure 3.8(c). Note that the error measurements are limited to the interval $[-3a, 3a]$ and the Δerror measurements to $[-3b, 3b]$. The control actions are in the range $[-4c, 4c]$.

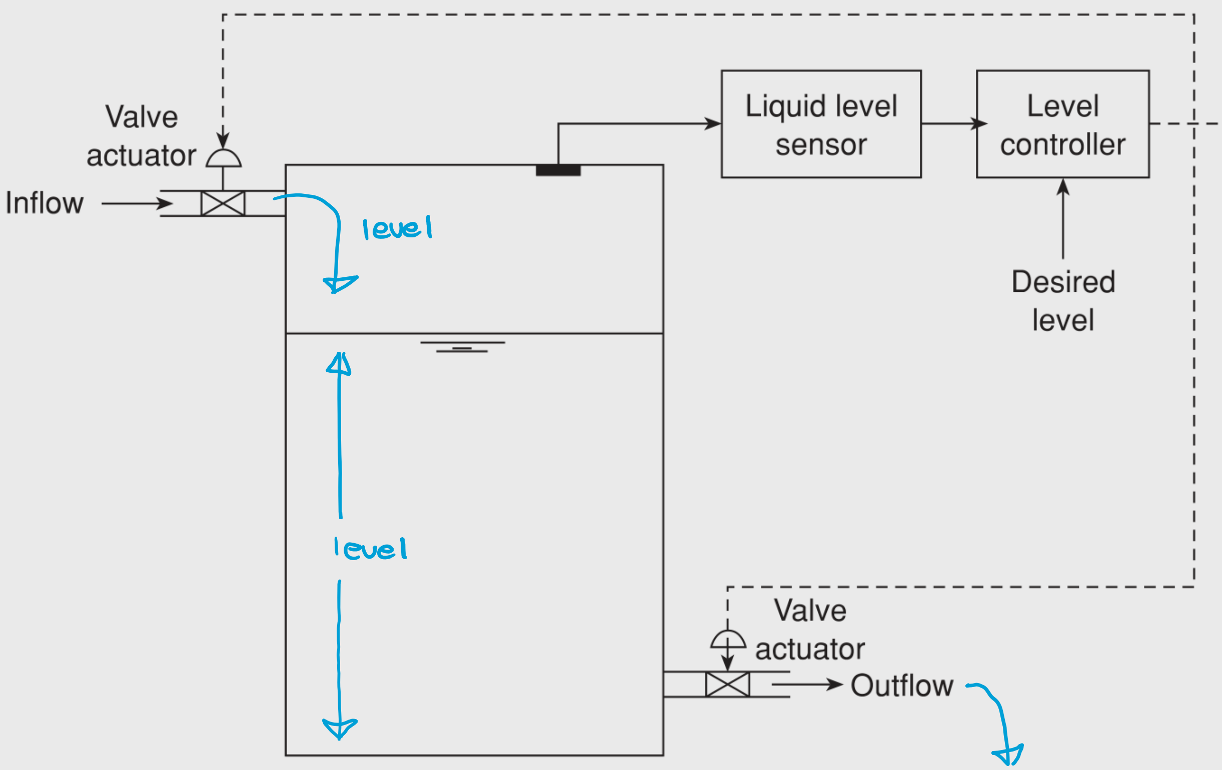


Figure 3.8 (a): Liquid level control system

E \ ΔE	NL	NS	ZO	PS	PL
NL	NL	NL	NM	NS	ZO
NS	NL	NM	NS	ZO	PS
ZO	NM	NS	ZO	PS	PM
PS	NS	ZO	PS	PM	PL
PL	ZO	PS	PM	PL	PL

Figure 3.8 (b): The control rule base

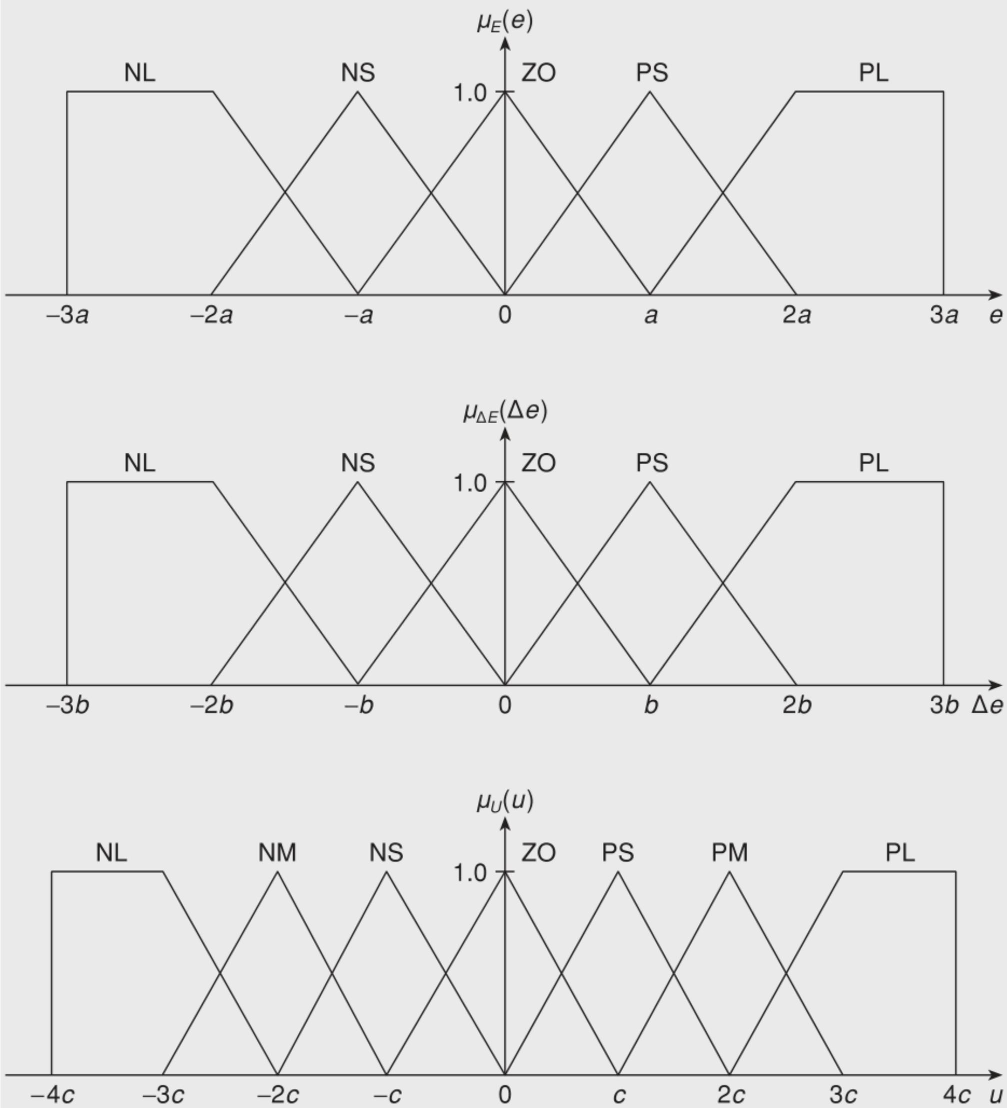


Figure 3.8 (c): The membership functions of error, change in error, and control action

Chapter 4: System Training

The difference between a fuzzy system and a neuro fuzzy system is that we can implement the fuzzy system like a neural network, then we can train system parameters.

We can use machine learning or training algorithms to optimize membership function parameters. This includes the TSK model (the consequent part parameters) and system reasoning structures. Parameters can be linear or nonlinear.

Linear: $z = 3x + 5y + 2$
Non-linear: $z^* = 2x^2 + 3y^3 + x + 2$

4.1 Least Squares Estimator (LSE)

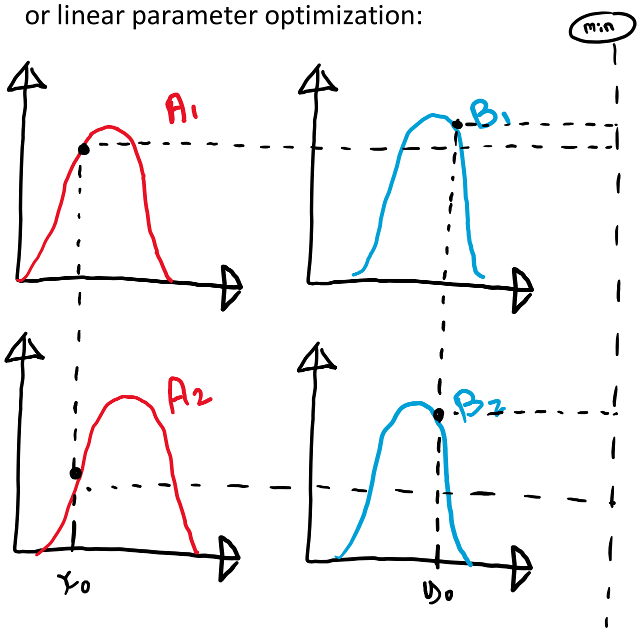
For linear parameter optimization:

$$y = \theta_1 f(\vec{u}_1) + \theta_2 f(\vec{u}_2) \dots \theta_n f(\vec{u}_n)$$

$\text{Parameters} = \{\theta_1 \quad \theta_2 \quad \dots \quad \theta_n\}$
$\text{Output} = y$
$\text{Input vectors} = \vec{\mathrm{u}}_1, \vec{\mathrm{u}}_2, \dots, \vec{\mathrm{u}}_\mathrm{n}$
(Because $\vec{\mathrm{u}} = \{\vec{\mathrm{u}}_1 \quad \vec{\mathrm{u}}_2 \quad \dots \quad \vec{\mathrm{u}}_\mathrm{n}\}$)

4.1 Least Squares Estimator

or linear parameter optimization:



$z_1 = p_1 x + q_1 y + r_1$

$z_2 = p_2 x + q_2 y + r_2$

$$z^* = \frac{w_1 z_1 + w_2 z_2}{w_1 + w_2}$$

$$z^* = \frac{w_1(p_1 x + q_1 y + r_1) + w_2(p_2 x + q_2 y + r_2)}{w_1 + w_2}$$

Linear parameters: $p_1, q_1, r_1, p_2, q_2, r_2$

$$\mu_{A_2} = e^{-\frac{(x-a)^2}{b^2}} \quad ; \quad w_2 = e^{-\left(\frac{x_0 - a}{b}\right)^2}$$

Nonlinear: MF (membership function) parameters

$$y = \theta_1 f_1(\vec{u}) + \theta_2 f_2(\vec{u}) + \cdots + \theta_n f_n(\vec{u})$$

$$\vec{u} = \{x_1, x_2, \dots, x_n\}^T \quad \text{inputs}$$

$$\vec{\theta} = \{\theta_1, \theta_2, \dots, \theta_n\}^T \quad \text{unknown}$$

Linear parameters:

$$\{\vec{u}_1, y_1\}, \{\vec{u}_2, y_2\}, \dots, \{\vec{u}_m, y_m\}$$

General representation:

$$\{\vec{u}_i, y_i\} \quad ; \quad i = 1, 2, \dots, m$$

$$\begin{gathered}
f_1(\vec{u}_1)\theta_1 + f_2(\vec{u}_1)\theta_2 + \cdots + f_n(\vec{u}_1)\theta_n = y_1 \\
f_1(\vec{u}_2)\theta_1 + f_2(\vec{u}_2)\theta_2 + \cdots + f_n(\vec{u}_2)\theta_n = y_2 \\
\vdots \\
f_1(\vec{u}_m)\theta_1 + f_2(\vec{u}_m)\theta_2 + \cdots + f_n(\vec{u}_m)\theta_n = y_m
\end{gathered}$$

Matrix representation:

$$\begin{array}{c} \vec{a}_1^T \\ \vec{a}_2^T \\ \vdots \\ \vec{a}_i^T \\ \vdots \\ \vec{a}_n^T \end{array} \underbrace{\begin{bmatrix} f_1(\vec{u}_1) & f_2(\vec{u}_1) & \cdots & f_n(\vec{u}_1) \\ f_1(\vec{u}_2) & f_2(\vec{u}_2) & \cdots & f_n(\vec{u}_2) \\ \vdots & \vdots & \cdots & \vdots \\ f_n(\vec{u}_i) & f_n(\vec{u}_i) & \cdots & f_n(\vec{u}_i) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(\vec{u}_m) & f_2(\vec{u}_m) & \cdots & f_n(\vec{u}_m) \end{bmatrix}}_{A} \underset{\vec{\theta}}{\begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}} = \underset{\vec{y}}{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}$$

$\vec{\theta}^T = \{\theta_1, \theta_2, \dots, \theta_n\}$

Summary:

- Vectors "→" (column representation, typically)
- Matrix $\underline{A}$
- Scalar

$$\underbrace{\vec{a}_i^T = \{f_1(\vec{u}_1), f_2(\vec{u}_2), \dots, f_n(\vec{u}_i)\}}_{\{\vec{u}_i;\, y_i\}}$$

$\underline{A}\,\vec{\theta} = \vec{y}$

If $\underline{A}$ is non-singular ($\det \neq 0$)

$\underbrace{\underline{A}^{-1}\underline{A}}_{\text{Identity Matrix}}\,\vec{\theta} = \underline{A}^{-1}\,\vec{y}$

$\vec{\theta} = \underline{A}^{-1}\vec{y}$

$m \to n$

$m = \#$ of training data points

$n = \#$ of linear paramerers to be optimized

"In general, the training data points should be 5-times the number of linear data points to be optimized"

- Noise in experiments

Unavoidable (always present)

$$\underbrace{\underline{A}\,\vec{\theta}}_{\text{theoretical}} + \overset{\text{error}}{\vec{e}} = \underbrace{\vec{y}}_{\text{measured}}$$

Error vector:

$$\vec{e} = \vec{y} - \underline{A}\,\vec{\theta}$$

Objective function:

$$E(\vec{\theta}) = \left(y_1 - \vec{a}_1^T\vec{\theta}\right)^2 + \left(y_2 - \vec{a}_2^T\vec{\theta}\right)^2 + \cdots + \left(y_i - \vec{a}_i^T\vec{\theta}\right)^2 + \cdots + \left(y_m - \vec{a}_m^T\vec{\theta}\right)^2$$

$$E(\vec{\theta}) = \sum_{i=1} \left(y_i - \vec{a}_i^T\vec{\theta}\right)^2$$

$$\vec{e}_i = y_i - \vec{a}_i^T \vec{\theta} \quad ; \quad \text{Where } i = 1, 2, 3, \dots, m$$

$$E(\vec{\theta}) = \vec{e}_1^T \vec{e}_1 + \vec{e}_2^T \vec{e}_2 + \cdots + \vec{e}_i^T \vec{e}_i + \cdots + \vec{e}_m^T \vec{e}_m$$

$$E(\vec{\theta}) = \sum_{i=1}^{m} \vec{e}_i^T \vec{e}_i$$

Consider:

$\vec{e}^T \vec{e} = (\vec{y} - \underline{A}\vec{\theta})^T (\vec{y} - \underline{A}\vec{\theta})$

$= [\vec{y}^T - (\underline{A}\vec{\theta})^T](\vec{y} - \underline{A}\vec{\theta})$

$= [\vec{y}^T - \vec{\theta}^T \underline{A}^T](\vec{y} - \underline{A}\vec{\theta})$

$= \vec{y}^T \vec{y} - \vec{y}^T \underline{A}\vec{\theta} - \vec{\theta}^T \underline{A}^T \vec{y} + \vec{\theta}^T \underline{A}^T \underline{A}\vec{\theta}$

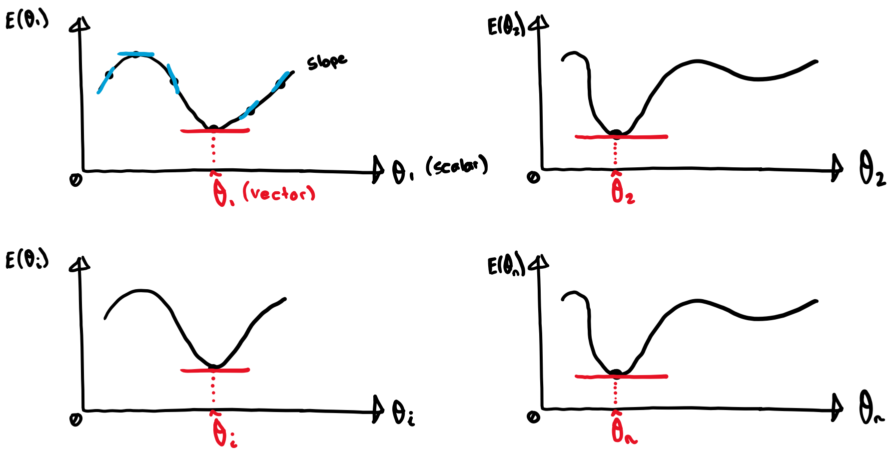
$\boxed{\vec{y} = \underline{A}\vec{\theta}}$

$\vec{y}^T = (\underline{A}\vec{\theta})^T$

$\boxed{\vec{y}^T = \vec{\theta}^T \underline{A}^T}$

$= \vec{y}^T \vec{y} - \vec{y}^T \underline{A}\vec{\theta} - \vec{y}^T \underline{A}\vec{\theta} + \vec{\theta}^T \underline{A}^T \underline{A}\vec{\theta}$

$= \vec{y}^T \vec{y} - 2\vec{y}^T \underline{A}\vec{\theta} + \vec{\theta}^T \underline{A}^T \underline{A}\vec{\theta}$



$$\vec{\theta} = \{\theta_1, \theta_2, \dots, \theta_n\}^T$$

$$\frac{\partial E(\vec{\theta})}{\partial \vec{\theta}} = \overset{=0}{\frac{\partial (\vec{y}^T \vec{y})}{\partial \vec{\theta}}} - 2(\vec{y}^T \underline{A})^T + \left[(\underline{A}^T \underline{A}) + (\underline{A}^T \underline{A})^T\right] \vec{\theta}$$

Let:

$$\frac{\partial E(\vec{\theta})}{\partial \vec{\theta}} = 0 \quad ; \quad \vec{\theta} = \hat{\vec{\theta}}$$

Consider:

$$\frac{\partial (\vec{y}^T \underline{A}\vec{x})}{\partial \vec{x}} = A^T \vec{y}$$

$$= 0 - 2\underline{A}^T\vec{y} + \left[\underline{A}^T A + \underline{A}^T\left(\underline{A}^T\right)^T\right]\hat{\vec{\theta}}$$

$$-2\underline{A}^T\vec{y} + 2\underline{A}^T\underline{A}\hat{\vec{\theta}} = 0$$

$$\underline{A}^T\underline{A}\hat{\vec{\theta}} = \underline{A}^T\vec{y}$$

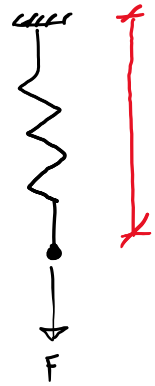
$$\left(\underline{A}^T\underline{A}\right)^{-1}\left(\underline{A}^T\underline{A}\right)\hat{\vec{\theta}} = \left(\underline{A}^T\underline{A}\right)^{-1}\underline{A}^T\vec{y}$$

$$\hat{\vec{\theta}} = \left(\underline{A}^T\underline{A}\right)^{-1}\underline{A}^T\vec{y}$$

$$\boxed{\hat{\vec{\theta}} = \frac{\underline{A}^T\vec{y}}{\underline{A}^T\underline{A}}}$$

Example 3.1 (Jang's Book)

$m = 7$



Experiment	Force (Newtons)	Length of Spring (inches)
1	1.1	1.5
2	1.9	2.1
3	3.2	2.5
4	4.4	3.3
5	5.9	4.1
6	7.4	4.6
7	9.2	5.0

$$L = k_0 + k_1 F$$

$$\begin{cases} k_0 + 1.1k_1 = 1.5 \\ k_0 + 1.9k_1 = 2.1 \\ \cdots \\ k_0 + 9.2k_1 = 5.0 \end{cases}$$

$$\underline{A}\hat{\vec{\theta}} = \vec{y} - \vec{e}$$

Since $\vec{e} = \vec{y} - \underline{A}\vec{\theta}$

$$\begin{bmatrix} 1 & 1.5 \\ 1 & 2.1 \\ \vdots & \vdots \\ 1 & 5.0 \end{bmatrix} \begin{bmatrix} k_0 \\ k_1 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_7 \end{bmatrix} - \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_7 \end{bmatrix}$$

$$\hat{\vec{\theta}} = \begin{bmatrix} k_0 \\ k_1 \end{bmatrix} = \frac{\underline{A}^T \vec{y}}{\underline{A}^T \underline{A}}$$ (2×2 matrix)

Use MATLAB (inv and $.*$ operators)

Or, manually (since it is a 2x2 matrix) via:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\hat{\vec{\theta}} = \begin{bmatrix} k_0 \\ k_1 \end{bmatrix} = \begin{bmatrix} 1.20 \\ 0.44 \end{bmatrix}$$

$$\boxed{L = 1.20 + 0.40F}$$