LEARNING IN ADAPTIVE SYSTEMS
Learning in Adaptive Systems

Generate real-time estimates of parameters and states using on-line data

The design of a self-tuning controller in the presence of parametric uncertainties

Learning $\equiv$ Parameter Estimation in Real-time
Problem Statement – Adaptive Estimation

Goal: Find $\hat{\theta}$, $\hat{x}$ so that $\hat{\theta} \to \theta$, $\hat{x} \to \Gamma x$

Plant (Dynamic System)

Adaptive Estimator

Model Structure

On-line information

State estimate

Parameter estimate

$u$ $\rightarrow$ $y$

$\dot{x} = f(x, \theta, u)$
$y = g(x, \theta, u)$

$\theta$: parameter, unknown
$x, y, u$: state, output, input

$e$: state error
$\hat{\theta}$: parameter estimate

$\hat{x} = F(\hat{x}, \hat{\theta})$
$\dot{\hat{\theta}} = G(\hat{x}, \hat{\theta})$

$\theta$, $\hat{x}$, $\hat{\theta}$: state, output, input

$\Gamma x$: Eigenvalues
Problem Statement – Adaptive Control

Goal: Find $u, C_1, C_2$ so that regulation and tracking occur.

Adaptive Control

$$u = C_1(\omega, \theta_c(t), e)$$

$$\dot{\theta}_c = C_2(\omega, \theta_c(t), e)$$

$e$: tracking error

$\theta_c$: control parameter estimate

Plant (Dynamic System)

$$\dot{x} = f(x, \theta, u)$$

$$y = g(x, \theta, u)$$

$\theta$: parameter, unknown

$x, y, u$: state, output, input

$\omega$: real-time data
Error Models - Two types of errors

- $e$: Performance error (ex. $\hat{x} - x; x - x_m$)
  - can be measured, needs to be reduced

- $\tilde{\theta}$: Parameter error (ex. $\hat{\theta} - \theta$)
  - Unknown, can be adjusted – Learning Rule

**Goal:** Determine error models and learning rules
Typical Error Models

Adaptive Control

Plant (Dynamic System)

$e$: tracking error, $\tilde{\theta} = \theta_c - \theta^*$

Model Structure

Control input $u$

Parameter estimate $\theta_c$

On-line information

$\omega$: system data

Error Model 1: $e = \tilde{\theta}^T \omega$

Error Model 3: $e = W(s)[\tilde{\theta}^T \omega]$  \hspace{1cm} $W(s)$: Dynamic operator

Goal: Find learning rules for adjusting $\tilde{\theta}$ so that high performance and learning are achieved
Error Model 1

System Model:

\[ y = \omega^T \theta \]

\[ \gamma \]

\[ \omega \]

\[ \theta^T \]

\[ y \]

Build an estimator:

\[ \hat{y} = \omega^T \hat{\theta} \]

Performance error:

\[ e = \hat{y} - y \]

Parameter error:

\[ \tilde{\theta} = \hat{\theta} - \theta \]

Error Model:

\[ \omega \]

\[ \tilde{\theta}^T \]

\[ e \]

Stability Framework:

\[ J = e^2 \]

\[ \dot{\theta} = -\eta \nabla J \]

\[ V = \| \tilde{\theta} \|^2 \]

\[ \dot{V} = 2 \tilde{\theta}^T \dot{\tilde{\theta}} = -2 \eta \tilde{\theta}^T e \omega = -2 \eta e^2 \leq 0 \]

Simplest Gradient Descent
Error Model 3

- Can be extended to higher order systems (in a later lecture)
- Class of $W(s)$ strictly positive real
  \[ \Rightarrow e'(t) \text{ and } e(t) \text{ have same sign most of the time} \]
- $J = e^2, \quad W \neq \frac{\partial J}{\partial \tilde{\theta}} \quad \text{But the adaptive law still works}$
- If $W(s)$ is not strictly positive real, this will not work

\[ W(s) = \frac{1}{s + a} \]
\[ \dot{\theta} = -e\omega \Rightarrow V = \frac{1}{2}(e^2 + \tilde{\theta}^T \tilde{\theta}) \]
\[ \dot{V} \leq 0 \]
Stability Framework

- Performance and Learning are conflicting objectives
- Effect of imperfect learning on performance has to be accommodated.
- Use a stability framework

\[ \dot{\theta} \]

\[ W(s) \]

\[ e \]

Real-time data

Goal: Find learning rules for adjusting \( \tilde{\theta} \) so that \( \dot{\theta} \leq 0 \)
Distinctions: Gradient-descent leads to instability!

\[ e = W(s)[\phi^T \tilde{\theta}] \]

\[ \dot{\tilde{\theta}} = -\nabla(e^2) \]
\[ \rightarrow \text{Instability} \]

Real-time data \( \phi \) \( \rightarrow \) \( \tilde{\theta} \) \( \rightarrow \) \( W(s) \) \( \rightarrow \) Performance error \( e \)

Stable adaptive law

\[ V = e^2 + \|\tilde{\theta}\|^2 \]
\[ \dot{V} = 2\tilde{\theta}^T \dot{\tilde{\theta}} + 2ee\dot{e} \]
\[ \dot{\tilde{\theta}} = -e\phi \]
\[ = -2e^2 \leq 0 \quad \text{(for a class of } W(s)\text{)} \]

* HP Whitaker, An Adaptive System for Control of the Dynamics Performances of Aircraft and Spacecraft, Inst Aeronautical Services, 1959 - MIT Rule
Guarantees with Imperfect Learning

Stable Subspace  \( \theta^* = \theta_1 \)

Unstable Subspace  \( \theta_1 + \Delta \)

Adaptive Control

Open Loop Unstable Short Period Dynamics

Adaptive Controller

Learning  \( \theta \)

Controller  \( u \)

\( \alpha \)

Time, t (s)

Angle of Attack, \( \alpha \) (deg)

Effect of Different Adaptive Parameters

\( \theta_1 + \Delta \)

With \( \theta_1 \)

Performance goal: \( \text{Min}(e) \)

Imperfect learning, yet safe performance
MACHINE LEARNING
Machine Learning

The ability of a computer to learn using on-line data
- Significant success in image & speech recognition, games
- By and large a classification/pattern-recognition tool

Standard Methods
- Linear Regression
- Logistic Regression
- Multi-layered Neural Networks
- Support-Vector Machines

Typical learning methods:
- Supervised learning
- Unsupervised learning
- Reinforcement learning
Neural Networks: A popular learning methodology

A bit of history

• Proposed in 1944 by McCullough and Pitts
• Controversy in the ‘70s: Multilayered Perceptrons, Minsky and Papert
• Resurgence in the 1980s
• A re-resurgence in the 21st century – fast processing power of graphics

(from MIT News)
Fundamentals of ML/Neural Networks

Universal Approximation Theorem

For $\epsilon > 0$, $\exists N, w^*$ s.t.

$$|y_a - y| \leq \epsilon \quad \forall x \in X$$
Activation functions

(from Wikipedia)
(1) $x$: demographic/employment related variable
   $y$: income above or below $50K$

(2) $x$: 15,000 nouns, 500 images from the web
   $y$: ranking based on # occurrences

(3) $x$: 800,000 text articles
   $y$: classify into 4 categories

(4) $x$: pixel images
   $y$: classify into one of 10 digits \{0, ..., 9\}
Deep Learning: Increased layers

(adapted from P. Khargonekar, ACC Workshop on Machine Learning, 2018)