CS545—Contents XVI

• Adaptive Control
  ▪ Model Reference Adaptive Control
  ▪ Self-Tuning Regulators
  ▪ Linear Regression
  ▪ Recursive Least Squares
  ▪ Gradient Descent
  ▪ Feedback-Error Learning

• Reading Assignment for Next Class
  ▪ See http://www-clmc.usc.edu/~cs545
The Adaptive Control Problem

- Characterize the desired behavior of the closed loop system
- Determine a suitable control law with adjustable parameters
- Find a mechanism for adjusting the parameters
- Implement the control law
Model-Reference Adaptive Control (Direct Learning)

- Performance is given to correspond to a particular reference model
  - E.g. $m\ddot{x} + b\dot{x} + c = u$
- Adjustment of controller is done directly
- E.g., adjust controller parameter by gradient descent

![Diagram of Model-Reference Adaptive Control](image)
Model-Reference Adaptive Control – Example

- Consider the generic control system
  \[ \dot{x} = f(x) + g(x)u \]

- For this example, make this an even simpler system
  \[ \dot{x} = f(x) + u \]

- Assume that \( f \) is unknown and needs to be estimated by a learning process. Thus, we can formulated a control law:
  \[ u = -\hat{f}(x) + \dot{x}_d - k(x - x_d) \]
  \[ = -x \hat{\theta} + \dot{x}_d - k(x - x_d) \]

  - Where we replaced \( f \) with a simple linear function
Model-Reference Adaptive Control – Example

- The goal of model-reference adaptive control is to adjust the open parameter and the control law such that the system is ALWAYS stable.
- The system dynamics are now

\[ \dot{x} = x\, \theta - x\, \hat{\theta} + \dot{x}_d - k(x - x_d) \]

- Define errors

\[ e = x_d - x \]
\[ \tilde{\theta} = \theta - \hat{\theta} \]

- Thus:

\[ \dot{x} = x\, \tilde{\theta} + \dot{x}_d + ke \]
\[ 0 = x\, \tilde{\theta} + \dot{e} + ke \]
\[ \dot{e} = -x\, \tilde{\theta} - ke \]
Model-Reference Adaptive Control – Example

- Define a Lyapunov function

\[ V = \frac{1}{2} e^2 + \frac{1}{2} \tilde{\theta} \Gamma^{-1} \tilde{\theta} \]
\[ \dot{V} = e \dot{e} + \tilde{\theta} \Gamma^{-1} \dot{\tilde{\theta}} \]
\[ = e \dot{e} - \tilde{\theta} \Gamma^{-1} \dot{\tilde{\theta}} \]
\[ = e \left( -x \tilde{\theta} - ke \right) - \tilde{\theta} \Gamma^{-1} \dot{\tilde{\theta}} \]
\[ = -ex\tilde{\theta} - ke^2 - \tilde{\theta} \Gamma^{-1} \dot{\tilde{\theta}} \]

- Thus, choose:

\[ -ex\tilde{\theta} - \tilde{\theta} \Gamma^{-1} \dot{\tilde{\theta}} = 0 \]
Model-Reference Adaptive Control – Example

• Thus:

\[-ex\tilde{\theta} - \tilde{\theta} \Gamma^{-1}\dot{\theta} = 0\]

\[\tilde{\theta}(-ex - \Gamma^{-1}\dot{\theta}) = 0\]

\[\dot{\theta} = -\Gamma ex\]

• I.e., the guaranteed stable parameter adaption law is

\[\dot{\theta} = -\Gamma ex\]
Nonlinear Example
Self-tuning Regulators (Indirect Learning)

- Controller is redesigned based on some estimated parameters
  - “certainty equivalence principle”
  - E.g., LQR controller is redesigned based on estimated model
  - This corresponds to an indirect update of the controller

![Diagram of self-tuning regulators](image-url)
Example: Estimate the Robot Model From Data

- How to obtain data?
  - Try “random” commands $u$, observe the state and change of state
  - Don’t destroy the robot …

- How to estimate the model?
  - Model is nonlinear
    - Need nonlinear estimation techniques (e.g., neural networks)
  - Model is linear
    - Use linear regression or recursive least squares

- Essential ingredients of estimation:
  - A cost criterion:
    - Usually least squares
      $$J = \frac{1}{2} \sum_{i=1}^{N} (t_i - y_i)^2$$
  - Some adjustable parameter (“a data generating model”)
Linear Regression for One Output

- The data generating model
  \[ y = \tilde{w}^T \tilde{x} + w_0 + \epsilon = w^T x + \epsilon \]
  where \( x = [x^T, 1]^T, w = [\tilde{w}, w_0]^T, E\{\epsilon\} = 0 \)

- Least Squares Cost Function
  \[ J = \frac{1}{2} (t - y)^T (t - y) = \frac{1}{2} (t - Xw)^T (t - Xw) \]
  where: \( t = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_n \end{bmatrix}, X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} \)

- Minimize Cost
  \[ \frac{\partial J}{\partial w} = 0 = \frac{\partial J}{\partial w} \left( \frac{1}{2} (t - Xw)^T (t - Xw) \right) = -(t - Xw)^T X \]
  \[ = -t^T X + (Xw)^T X = -t^T X + w^T X^T X \]
  thus: \( t^T X = w^T X^T X \) or \( X^T t = X^T X w \)
  \[ \text{result: } w = (X^T X)^{-1} X^T t \]
Remarks on Least Squares

- the Pseudo-Inverse
  \[ X^\# = (X^T X)^{-1} X^T \]
  - pseudo inverses are a special solution to an infinite set of solutions of a non-unique inverse problem
  - the matrix inversion above may still be ill-defined if \( X^T X \) is close to singular and so-called Ridge regression needs to be applied
  - more about inverse problems in dedicated class

- Ridge Regression
  \[ X^\# = (X^T X + \gamma I)^{-1} X^T \] where \( \gamma << 1 \)

- Multiple Outputs: just like multiple single output regressions
  \[ W = (X^T X)^{-1} X^T Y \]
Geometrical Interpretation of Least Squares

Subspace $S$ spanned by the columns of $X$

Vector of residual errors (orthogonal to $y$)

$t - y = t - Xw = t - \sum_i [X]_i w_i$

- $y$ is the optimal reconstruction of $t$ in the range of $X$

$y$ is an orthogonal Projection of $t$ on $S$
Physical Interpretation of Least Squares

- all springs have the same spring constant
- points far away generate more “force” (danger of outliers)
- springs are vertical
- solution is the minimum energy solution achieved by the springs
Recursive Least Squares

- The Sherman-Morrison-Woodbury Theorem
  \[(A - zz^T)^{-1} = A^{-1} + \frac{A^{-1}zz^TA^{-1}}{1 - z^TA^{-1}z}\]

- More General: The Matrix Inversion Theorem
  \[(A - BC)^{-1} = A^{-1} + A^{-1}B(I + C^{-1}A)^{-1}CA^{-1}\]

- Recursive Least Squares Update

  Initialize:  \[P^n = \frac{I}{\gamma}\] where \(\gamma << 1\) (note \(P = (X^TX)^{-1}\))

  For every new data point \((x,t)\)
  (note that \(x\) includes the bias):

  \[P^{n+1} = \frac{1}{\lambda} \left( P^n - \frac{P^nxx^TP^n}{\lambda + x^TP^nx} \right)\] where \(\lambda = \begin{cases} 1 & \text{if no forgetting} \\ < 1 & \text{if forgetting} \end{cases}\)

  \[W^{n+1} = W^n + P^{n+1}x(t - W^nxx^T)\]
Recursive Least Squares (cont’ d)

- Some amazing facts about recursive least squares
  - Results for $W$ are EXACTLY the same as for normal least squares update (batch update) after every data point was added once! (no iterations)
  - NO matrix inversion necessary anymore
  - NO learning rate necessary
  - Guaranteed convergence to optimal $W$ (linear regression is an optimal estimator under many conditions)
  - Forgetting factor $\lambda$ allows to forget data in case of changing target functions
  - Computational load is larger than batch version of linear regression
  - But don’t get fooled: if data is singular, you still will have problems!
Feedback Error Learning

- Feedback Error Learning is a special adaptive control architecture to learn an inverse dynamics feedforward controller.

How would we change recursive least squares to deal with an error signal instead of an absolute teacher signal?
Gradient Descent
(a.k.a. LMS, Delta rule, Widrow-Hoff rule, Adaline rule)

- Gradient descent can be used even if the model is nonlinear in the parameters
- Idea: Change parameters incrementally to reduce the least squares cost iteratively
- Stochastic Update

Given: \( J_i = \frac{1}{2}(t_i - y_i)^2 \) and \( a_i = w^T x_i \)

\[
\frac{\partial J_i}{\partial w} = -(t_i - y_i) \frac{\partial y_i}{\partial w} = -(t_i - y_i) \frac{\partial f(a_i)}{\partial w}
\]

\[
= -(t_i - y_i) \frac{\partial f(a_i)}{\partial a_i} \frac{\partial a_i}{\partial w} = -(t_i - y_i) f'(a_i) x_i^T
\]

Update:

\[
w^{n+1} = w^n - \alpha \left( \frac{\partial J_i}{\partial w} \right)^T = w^n + \alpha (t_i - y_i) f'(a_i) x_i^T
\]

\[
w^{n+1} = w^n + \alpha \delta_i f'(a_i) x_i^T
\]

where \( \alpha \) is the learning rate