F.L. Lewis, NAI

Moncrief-O’Donnell Chair, UTA Research Institute (UTARI)
The University of Texas at Arlington, USA
Foreign Consulting Professor, Guangdong Univ. of Science and Technology
and
Qian Ren Consulting Professor, State Key Laboratory of Synthetical Automation for Process Industries
Northeastern University, Shenyang, China

Data-driven Control and Optimization for Industrial Processes:
Reinforcement Learning and Supervisory Control

Talk available online at
http://www.UTA.edu/UTARI/acs
Invited by Shengli Xie
Guoxu Zhou
I. Online Optimization Using Reinforcement Learning
II. Methods for data-driven control
III. Data-driven Optimization Using RL with Output feedback
Relevance- Industrial Process Control

Precision Process Control with unmodeled dynamics, disturbance rejection, time-varying parameters, deadzone/backlash control

Industrial Machines

XY Table

Chemical Vapor Deposition

Autoclave
Intelligent Operational Control for Complex Industrial Processes

Jinliang Ding

Professor Chai Tianyou

State Key Laboratory of Synthetical Automation for Process Industries
Northeastern University
May 20, 2013
Production line for mineral processing plant – Chai Tianyou

Mineral Processing Plant in Gansu China

the high-intensity magnetic production line (HMPL)

Particle ore

Grinding unit

High-intensity magnetic separation

Concentrated ore

Dewatering unit

Mixed concentrated ore

the low-intensity magnetic production line (LMPL)

Raw ore

Screening of raw ore

Lump ore

Roasting and dry separation

Useful ore

Waste rock

Grinding unit

Low-intensity magnetic separation

Tailing

Dewatering unit

Mixed tailing
Modern Industrial Processes Must be Operated in Optimal Fashion

- Minimum time
- Minimum energy usage
- Minimum resource consumption
- Minimum pollution
- Minimum ...
Rocket Orbit Injection

Dynamics

\[
\begin{align*}
\dot{r} &= w \\
\dot{w} &= \frac{v^2}{r} - \frac{\mu}{r^2} + \frac{F}{m} \sin \phi \\
\dot{v} &= -\frac{w v}{r} + \frac{F}{m} \cos \phi \\
\dot{m} &= -F m
\end{align*}
\]

Objectives

Get to orbit in minimum time
Use minimum fuel

http://microsat.sm.bmstu.ru/e-library/Launch/Dnepr_GEO.pdf
The Power of Optimal Design

Once you can do optimal design that minimizes a performance index, many sorts of designs are immediately possible.

Minimum energy
\[ J = \frac{1}{2} \int_{0}^{\infty} x^T Q x + u^T R u \, dt \]

Minimum fuel
\[ J = \frac{1}{2} \int_{0}^{\infty} x^T Q x + \rho u u \, dt \]

Minimum time
\[ J = \int_{0}^{T} 1 \, dt = T \]

Constrained control inputs
\[ J = \frac{1}{2} \int_{0}^{\infty} \left( Q(x) + \int_{0}^{u} \sigma^{-1}(v) \, dv \right) \, dt \]

Approximate minimum time with smooth control inputs
\[ J = \frac{1}{2} \int_{0}^{\infty} \left( \tanh(x^T Q x) + \rho \int_{0}^{u} \sigma^{-1}(v) \, dv \right) \, dt \]

Also-
- Conservation of resources
- Efficient execution
- Save Money
Optimal Control

Optimal Control is Effective for:
- Industrial Process Control
- Aircraft Autopilots
- Vehicle engine control
- Aerospace Vehicles
- Ship Control

But, optimal control solutions are normally found by
- Offline solution of Matrix Design equations
- A full dynamical model of the system is needed
Optimal Control- The Linear Quadratic Regulator (LQR)

User prescribed optimization criterion

\[ J = (Q, R) \]

\[ 0 = PA + A^T P + Q - PBR^{-1}B^T P \]

\[ K = R^{-1}B^T P \]

An Offline Design Procedure
that requires Knowledge of system dynamics model \((A,B)\)

System modeling is expensive, time consuming, and inaccurate
We want to find optimal control solutions
Online in real-time
Using measured Plant input/output data
Using adaptive control techniques
Without knowing the full dynamics

For nonlinear systems and general performance indices

Data-Driven Online Optimal Control

DDC- Data-driven Control
DDO- Data-driven Optimization
Adaptive Control is online and works for unknown systems. Generally not Optimal

Optimal Control is off-line, and needs to know the system dynamics to solve design eqs.

We want ONLINE DIRECT ADAPTIVE OPTIMAL Control For any performance cost of our own choosing

Reinforcement Learning turns out to be the key to this!


I. Online Optimization Using Reinforcement Learning
Books


- New Chapters on:
  - Reinforcement Learning
  - Differential Games

Different methods of learning

Reinforcement learning
Ivan Pavlov 1890s

We want OPTIMAL performance
- ADP- Approximate Dynamic Programming
- Paul Werbos

Actor-Critic Learning

Desired performance

Adaptive Learning system
Actor

Tune actor

Reinforcement signal

Control Inputs

System

outputs

environment

Sutton & Barto book

DDO- Data-Driven Online Optimal Control

1. System dynamics
2. Value/cost function
3. Bellman equation
4. HJB solution equation (Riccati eq.)
5. Reinforcement Learning – gives the structure we need
RL has been developed for Discrete-Time Systems

Discrete-Time System Hamiltonian Function

$$H(x_k, \nabla V(x_k), h) = r(x_k, h(x_k)) + \gamma V_h(x_{k+1}) - V_h(x_k)$$

- Directly leads to temporal difference techniques
- System dynamics does not occur
- Two occurrences of value allow APPROXIMATE DYNAMIC PROGRAMMING methods
  
  Paul Werbos HDP, DHP, ADHDP, ADDHP

Continuous-Time System Hamiltonian Function

$$H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + r(x, u) = \left( \frac{\partial V}{\partial x} \right)^T \dot{x} + r(x, u) = \left( \frac{\partial V}{\partial x} \right)^T f(x, u) + r(x, u)$$

Leads to off-line solutions if system dynamics is known
Hard to do on-line learning

- How to define temporal difference?
- System dynamics DOES occur
- Only ONE occurrence of value gradient

How can one do Policy Iteration for Unknown Continuous-Time Systems?
What is Value Iteration for Continuous-Time systems?
How can one do ADP for CT Systems?
Optimal Control: Linear Quadratic Regulator

System
\[ \dot{x} = Ax + Bu \]

Cost
\[ V(x(t)) = \int_{t}^{\infty} (x^T Q x + u^T Ru) \, d\tau = x^T (t) Px(t) \]

Differential equivalent is the Bellman equation
\[ 0 = H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + x^T Q x + u^T Ru = 2 \left( \frac{\partial V}{\partial x} \right)^T \dot{x} + x^T Q x + u^T Ru = 2x^T P(Ax + Bu) + x^T Q x + u^T Ru \]

Given any stabilizing FB policy \( u = -Kx \)

The cost value is found by solving Lyapunov equation = Bellman equation
\[ 0 = (A - BK)^T P + P(A - BK) + Q + K^T RK \]

Optimal Control is
\[ u = -R^{-1} B^T Px = -Kx \]

Algebraic Riccati equation
\[ 0 = PA + A^T P + Q - PBR^{-1} B^T P \]

Full system dynamics must be known
Off-line solution
Continuous-Time Nonlinear Optimal Control

To find online methods for optimal control

Focus on these two equations

System dynamics

\[ \dot{x} = f(x, u) = f(x) + g(x)u \]

Cost/value

\[ V(x(t)) = \int_t^\infty r(x, u) \, dt = \int_t^\infty (Q(x) + u^T R u) \, dt \]

Bellman Equation, in terms of the Hamiltonian function

\[ H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + r(x, u) = \left( \frac{\partial V}{\partial x} \right)^T \dot{x} + r(x, u) = \left( \frac{\partial V}{\partial x} \right)^T (f(x) + g(x)u) + r(x, u) = 0 \]

Stationarity condition

\[ \frac{\partial H}{\partial u} = 0 \]

Stationary Control Policy

\[ u = h(x) = -\frac{1}{2} R^{-1} g^T (x) \frac{\partial V}{\partial x} \]

HJB equation

\[ 0 = \left( \frac{dV^*}{dx} \right)^T f + Q(x) - \frac{1}{4} \left( \frac{dV^*}{dx} \right)^T g R^{-1} g^T \frac{dV^*}{dx} \quad , \quad V(0) = 0 \]
CT Policy Iteration – a Reinforcement Learning Technique

Given any admissible policy \( u(x) = h(x) \)

The cost is given by solving the CT Bellman equation

\[
0 = \left( \frac{\partial V}{\partial x} \right)^T f(x, u) + r(x, u) \equiv H(x, \frac{\partial V}{\partial x}, u)
\]

Utility \( r(x, u) = Q(x) + u^T Ru \)

Policy Iteration Solution

Pick stabilizing initial control policy \( h_0(x) \)

Policy Evaluation - Find cost, Bellman eq.

\[
0 = \left( \frac{\partial V_j}{\partial x} \right)^T f(x, h_j(x)) + r(x, h_j(x)) \quad V_j(0) = 0
\]

Policy improvement - Update control

\[
h_{j+1}(x) = -\frac{1}{2} R^{-1} g^T(x) \frac{\partial V_j}{\partial x}
\]

Converges to solution of HJB

\[
0 = \left( \frac{dV^*}{dx} \right)^T f + Q(x) - \frac{1}{4} \left( \frac{dV^*}{dx} \right)^T g R^{-1} g^T \frac{dV^*}{dx}
\]

- Convergence proved by Leake and Liu 1967, Saridis 1979 if Lyapunov eq. solved exactly
- Beard & Saridis used Galerkin Integrals to solve Lyapunov eq.
- Abu Khalaf & Lewis used NN to approx. \( V \) for nonlinear systems and proved convergence

Full system dynamics must be known

Off-line solution

LQR Policy iteration = Kleinman algorithm (1960s)

1. For a given control policy $u = -K_j x$ solve for the cost:

$$0 = A_j^T P_j + P_j A_j + Q + K_j^T R K_j$$

Bellman eq. = Lyapunov eq.

$$A_j = A - BK_j$$

Matrix equation

2. Improve policy:

$$K_{j+1} = R^{-1} B^T P_j$$

- If started with a stabilizing control policy $K_0$ the matrix $P_j$ monotonically converges to the unique positive definite solution of the Riccati equation.
- Every iteration step will return a stabilizing controller.
- The system has to be known.

- **OFF-LINE DESIGN**
- **MUST SOLVE LYAPUNOV EQUATION AT EACH STEP.**

Kleinman 1968
Integral Reinforcement Learning

Work of Draguna Vrabie

\[ \dot{x} = f(x) + g(x)u \]

Can Avoid knowledge of drift term \( f(x) \)

Policy iteration requires repeated solution of the CT Bellman equation

\[ 0 = \dot{V} + r(x,u(x)) = \left( \frac{\partial V}{\partial x} \right)^T \dot{x} + r(x,u(x)) = \left( \frac{\partial V}{\partial x} \right)^T (f(x) + g(x)u) + Q(x) + u^T Ru \equiv H(x, \frac{\partial V}{\partial x}, u(x)) \]

This can be done online without knowing \( f(x) \) using measurements of \( x(t), u(t) \) along the system trajectories
value \[ V(x(t)) = \int_t^\infty r(x,u) \, d\tau \]  

**Key Idea**

Chop off the tail

\[ V(x(t)) = \int_t^{t+T} r(x,u) \, d\tau + \int_t^\infty r(x,u) \, d\tau \]

**Lemma 1 – Draguna Vrabie**

\[ 0 = \left( \frac{\partial V}{\partial x} \right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u), \quad V(0) = 0 \]

Bad Bellman eq

Is equivalent to

Integral reinf. form for the CT Bellman eq.

\[ V(x(t)) = \int_t^{t+T} r(x,u) \, d\tau + V(x(t+T)), \quad V(0) = 0 \]

Good Bellman eq

Solves Bellman equation without knowing \( f(x,u) \)

**Allow definition of temporal difference error for CT systems**

\[ e(t) = -V(x(t)) + \int_t^{t+T} r(x,u) \, d\tau + V(x(t+T)) \]
Integral Reinforcement Learning (IRL)- Draguna Vrabie

**IRL Policy iteration**

<table>
<thead>
<tr>
<th>Policy evaluation- IRL Bellman Equation</th>
<th>Cost update</th>
<th>CT Bellman eq.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_k(x(t)) = \int_t^{t+T} r(x,u_k) , dt + V_k(x(t+T))$</td>
<td>$f(x)$ and $g(x)$ do not appear</td>
<td></td>
</tr>
</tbody>
</table>

Equivalent to

$$0 = \left( \frac{\partial V}{\partial x} \right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u)$$

Solves Bellman eq. (nonlinear Lyapunov eq.) without knowing system dynamics

<table>
<thead>
<tr>
<th>Policy improvement</th>
<th>Control gain update</th>
<th>$g(x)$ needed for control update</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_{k+1} = h_{k+1}(x) = -\frac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$</td>
<td>$g(x)$ needed for control update</td>
<td></td>
</tr>
</tbody>
</table>

Initial stabilizing control is needed

Converges to solution to HJB eq.

$$0 = \left( \frac{dV^*}{dx} \right)^T f + Q(x) - \frac{1}{4} \left( \frac{dV^*}{dx} \right)^T g R^{-1} g^T \frac{dV^*}{dx}$$

D. Vrabie proved convergence to the optimal value and control
Integral Reinforcement Learning (IRL) - Draguna Vrabie

CT LQR Case

Value function is quadratic

\[ V(x(t)) = x^T(t)Px(t) \quad u_k(t) = -L_kx(t) \]

CT Bellman eq.

\[ x^T(t)P_kx(t) = \int_t^{t+T} x^T(\tau)(Q + K_k^TRK_k)x(\tau)d\tau + x^T(t+T)P_kx(t+T) \]

is equivalent to

\[ A_k^TP_k + P_kA_k + K_k^TRK_k + Q = 0 \]

\[ A_k = A - BK_k \]

\[ K_{k+1} = R^{-1}B^TP_k \]

Converges to solution to ARE

\[ 0 = PA + A^TP + Q - PBR^{-1}B^TP \]

**Theorem - D. Vrabie**

**This algorithm converges and is equivalent to Kleinman’s Algorithm**

This is a data-based approach that uses measurements of \(x(t), u(t)\)
Instead of the plant dynamical model.
CT Policy Iteration – How to implement online?
Linear Systems Quadratic Cost- LQR

Value function is quadratic \( V(x(t)) = x^T(t)Px(t) \)

Policy evaluation- solve IRL Bellman Equation

\[
x^T(t)P_k x(t) = \int_{t}^{t+T} x^T(\tau)(Q+L_k^T RL_k) x(\tau) \, d\tau + x^T(t+T)P_k x(t+T)
\]

\[
x^T(t)P_k x(t) - x^T(t+T)P_k x(t+T) = \int_{t}^{t+T} x^T(\tau)(Q+L_k^T RL_k) x(\tau) \, d\tau
\]

\[
\begin{bmatrix}
  x^1(t) \\
  x^2(t)
\end{bmatrix}
\begin{bmatrix}
  p_{11} & p_{12} \\
  p_{12} & p_{22}
\end{bmatrix}
\begin{bmatrix}
  x^1(t) \\
  x^2(t)
\end{bmatrix}
-
\begin{bmatrix}
  x^1(t+T) \\
  x^2(t+T)
\end{bmatrix}
\begin{bmatrix}
  p_{11} & p_{12} \\
  p_{12} & p_{22}
\end{bmatrix}
\begin{bmatrix}
  x^1(t+T) \\
  x^2(t+T)
\end{bmatrix}
\]

\[
= p_{11} \begin{bmatrix}
  (x^1)^2 \\
  2x^1x^2 \\
  (x^2)^2
\end{bmatrix}_{(t)} - p_{11} \begin{bmatrix}
  (x^1)^2 \\
  2x^1x^2 \\
  (x^2)^2
\end{bmatrix}_{(t+T)}
\]

\[
= p_{11}^T [\bar{x}(t) - \bar{x}(t+T)]
\]

Quadratic basis set= regression vector
Unknown parameter vector
Nonlinear Case- Approximate Dynamic Programming

Value Function Approximation (VFA) to Solve Bellman Equation
– Paul Werbos (ADP), Dmitri Bertsekas (NDP)

\[ V_k(x(t)) = \int_{t}^{t+T} \left( Q(x) + u_k^T R u_k \right) dt + V_k(x(t+T)) \]

Approximate value by Weierstrass Approximator Network \( V = W^T \phi(x) \)

\[ W_k^T \phi(x(t)) = \int_{t}^{t+T} \left( Q(x) + u_k^T R u_k \right) dt + W_k^T \phi(x(t+T)) \]

\[ W_k^T \left[ \phi(x(t)) - \phi(x(t+T)) \right] = \int_{t}^{t+T} \left( Q(x) + u_k^T R u_k \right) dt \]

Solve using Adaptive Control

Scalar equation with vector unknowns

Regression vector

Reinforcement on time interval \([t, t+T]\)

Unknown parameter vector

Now use RLS or batch LS along the trajectory to get new weights \( W_k \)

Then find updated FB

\[ u_{k+1} = h_{k+1}(x) = -\frac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x} = -\frac{1}{2} R^{-1} g^T(x) \left[ \frac{\partial \phi(x(t))}{\partial x(t)} \right]^T W_k \]

Direct Optimal Adaptive Control for Partially Unknown CT Systems
Solving the IRL Bellman Equation

Solve for value function parameters

\[
W^T = \begin{bmatrix}
p_{11} & p_{12} \\
p_{12} & p_{22}
\end{bmatrix}
\]

Need data from 3 time intervals to get 3 equations to solve for 3 unknowns

\[
W_k^T \left[ \phi(x(t)) - \phi(x(t+T)) \right] = \int_t^{t+T} \left( Q(x) + u_k^T R u_k \right) dt
\]

\[
W_k^T \left[ \phi(x(t+T)) - \phi(x(t+2T)) \right] = \int_{t+T}^{t+2T} \left( Q(x) + u_k^T R u_k \right) dt
\]

\[
W_k^T \left[ \phi(x(t+2T)) - \phi(x(t+3T)) \right] = \int_{t+2T}^{t+3T} \left( Q(x) + u_k^T R u_k \right) dt
\]

Now solve by Batch least-squares
Integral Reinforcement Learning (IRL)

Solve Bellman Equation - Solves Lyapunov eq. without knowing dynamics

\[ W_k^T \left[ \phi(x(t)) - \phi(x(t+T)) \right] = \int_t^{t+T} x(\tau)^T (Q + K_k^T R K_k) x(\tau) d\tau = \rho(t, t+T) \]

Data set at time \([t, t+T)\)

\[(x(t), \rho(t, t+T), x(t+T))\]

This is a data-based approach that uses measurements of \(x(t), u(t)\)
Instead of the plant dynamical model.

Or use batch least-squares

\[ A / f(x) \text{ is not needed anywhere} \]

update control gain

\[ K_{k+1} = R^{-1} B^T P_k \]
Continuous-time control with discrete gain updates

$K_k$ changes at discrete intervals $k$.

Control

$u_k(t) = -K_k x(t)$

Reinforcement Intervals $T$ need not be the same. They can be selected on-line in real time.
Persistence of Excitation

\[ W_k^T [\phi(x(t)) - \phi(x(t+T))] = \int_{t}^{t+T} x(\tau)^T (Q + K_k^T R K_k) x(\tau) d\tau = \rho(t, t+T) \]

Regression vector must be PE

Relates to choice of reinforcement interval \( T \)
Implementation

Policy evaluation
Need to solve online

\[ W_k^T [\phi(x(t)) - \phi(x(t+T))] = \int_t^{t+T} x(\tau)^T (Q + K_k^T R K_k) x(\tau) d\tau = \rho(t, t+T) \]

Add a new state = Integral Reinforcement

\[ \dot{\rho} = x^T Q x + u^T R u \]

This is the controller dynamics or memory
Direct Optimal Adaptive Controller

Solves Riccati Equation Online without knowing A matrix

A hybrid continuous/discrete dynamic controller
whose internal state is the observed cost over the interval

Reinforcement interval T can be selected on line on the fly – can change
Actor-Critic Structure

Reinforcement learning

Theta waves 4-8 Hz

A 2-timescale supervisory controller

Motor control 200 Hz

Paul Werbos
Figure 1. Learning-oriented specialization of the cerebellum, the basal ganglia, and the cerebral cortex [1], [2]. The cerebellum is specialized for supervised learning based on the error signal encoded in the climbing fibers from the inferior olive. The basal ganglia are specialized for reinforcement learning based on the reward signal encoded in the dopaminergic fibers from the substantia nigra. The cerebral cortex is specialized for unsupervised learning based on the statistical properties of the input signal.

Doya, Kimura, Kawato 2001
The Bottom Line about Integral Reinforcement Learning

**Off-line ARE Solution**

The Optimal Control Solution

\[ u = -R^{-1}B^TPx = -Kx \]

Algebraic Riccati equation

\[ 0 = PA + A^TP + Q - PBR^{-1}B^TP \]

Full system dynamics must be known

Off-line solution

**Data-driven On-line ARE Solution**

ARE solution can be found online in real-time by using the IRL Algorithm without knowing A matrix

Iterate for \( k=0,1,2,\ldots \)

CT Bellman eq.

\[ x^T(t)P_kx(t) = \int_t^{t+T} x^T(\tau)(Q + K_k^TRK_k)x(\tau) \, d\tau + x^T(t+T)P_kx(t+T) \]

\[ K_{k+1} = R^{-1}B^TP_k \]

Only B is needed

On line solution in real time

Uses data measurements along system trajectory
Optimal Control- The Linear Quadratic Regulator (LQR)

User prescribed optimization criterion

\[ J = (Q, R) \]

\[ 0 = PA + A^T P + Q - PBR^{-1} B^T P \]

\[ K = R^{-1} B^T P \]

An Offline design Procedure
that requires Knowledge of system dynamics model\((A,B)\)

System modeling is expensive, time consuming, and inaccurate
Data-driven Online Adaptive Optimal Control

Data-driven Optimization (DDO)

User prescribed optimization criterion

\[ J = (Q, R) \]

\[ x^T(t) P_k x(t) = \int_t^{t+T} x^T(\tau) (Q + K_k^T R K_k) x(\tau) d\tau + x^T(t+T) P_k x(t+T) \]

\[ K_{k+1} = R^{-1} B^T P_k \]

On-line Control Loop

Control \( K \)

\[ \dot{x} = A x + B u \]

On-line Performance Loop

System

\[ (x(t), \rho(t, t+T), x(t+T)) \]

An Online Supervisory Control Procedure

that requires no Knowledge of system dynamics model A

Automatically tunes the control gains in real time to optimize a user given cost function

Uses measured data \((u(t), x(t))\) along system trajectories
Simulation 1- F-16 aircraft pitch rate controller

\[ \dot{x} = Ax + Bu = \begin{bmatrix} -1.01887 & 0.90506 & -0.00215 \\ 0.82225 & -1.07741 & -0.17555 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ 0 \\ u \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ u \end{bmatrix} \]

\[ x = [\alpha \quad q \quad \delta_e] \]

\[ Q = I, \quad R = I \]

\[ ARE \quad 0 = PA + A^T P + Q - PBR^{-1}B^T P \]

Select quadratic NN basis set for VFA

Exact solution

\[ \bar{W}_1^* = [p_{11} \quad 2p_{12} \quad 2p_{13} \quad p_{22} \quad 2p_{23} \quad p_{33}]^T \]

\[ = [1.4245 \quad 1.1682 \quad -0.1352 \quad 1.4349 \quad -0.1501 \quad 0.4329]^T \]
Simulations on: F-16 autopilot

A matrix not needed

System states

Control signal

Controller parameters

Critic parameters

Converge to SS Riccati equation soln

Solves ARE online without knowing $A$

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$
Simulation 2: Load Frequency Control of Electric Power system

\[ \dot{x} = Ax + Bu \]

\[ x(t) = [\Delta f(t), \Delta P_g(t), \Delta X_g(t), \Delta E(t)]^T \]

\[ A = \begin{bmatrix}
-1/T_p & K_p/T_p & 0 & 0 \\
0 & -1/T_r & 1/T_r & 0 \\
-1/RT_G & 0 & -1/T_G & -1/T_G \\
K_E & 0 & 0 & 0 \\
\end{bmatrix}, \quad B = \begin{bmatrix}
0 \\
0 \\
1/T_G \\
0 \\
\end{bmatrix} \]

ARE

\[ 0 = PA + A^T P + Q - PBR^{-1}B^T P \]

ARE solution using full dynamics model (A,B)

\[ P_{ARE} = \begin{bmatrix}
0.4750 & 0.4766 & 0.0601 & 0.4751 \\
0.4766 & 0.7831 & 0.1237 & 0.3829 \\
0.0601 & 0.1237 & 0.0513 & 0.0298 \\
0.4751 & 0.3829 & 0.0298 & 2.3370 \\
\end{bmatrix}. \]
$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

Solves ARE online without knowing $A$

$$P_{ARE} = \begin{bmatrix}
0.4750 & 0.4766 & 0.0601 & 0.4751 \\
0.4766 & 0.7831 & 0.1237 & 0.3829 \\
0.0601 & 0.1237 & 0.0513 & 0.0298 \\
0.4751 & 0.3829 & 0.0298 & 2.3370
\end{bmatrix}.$$ 

IRL period of $T = 0.1$s.

Fifteen data points $(x(t), x(t+T), \rho(t:t+T))$

Hence, the value estimate was updated every 1.5s.
Optimal Control Design Allows a Lot of Design Freedom

\[ V(x(t)) = \int_{t}^{\infty} r(x, u) \, d\tau \]

Tailor controls design by choosing utility \( r(x, u) \)
The Power of Optimal Design

Once you can do optimal design that minimizes a performance index, many sorts of designs are immediately possible.

Minimum energy

\[ J = \frac{1}{2} \int_{0}^{\infty} x^T Q x + u^T R u \, dt \]

Minimum fuel

\[ J = \frac{1}{2} \int_{0}^{\infty} x^T Q x + \rho |u| \, dt \]

Minimum time

\[ J = \int_{0}^{T} 1 \, dt = T \]

Constrained control inputs

\[ J = \frac{1}{2} \int_{0}^{\infty} \left( Q(x) + \int_{0}^{u} \sigma^{-1}(v) \, dv \right) \, dt \]

Approximate minimum time with smooth control inputs

\[ J = \frac{1}{2} \int_{0}^{\infty} \left( \tanh(x^T Q x) + \rho \int_{0}^{u} \sigma^{-1}(v) \, dv \right) \, dt \]

Conservation of resources
Efficient execution
Save Money

Also-
Can CHANGE the Performance Specs
In real time

\[ \begin{array}{c}
\text{tanh}(p) \\
1 \\
-1 \\
p
\end{array} \]
Optimal Control for Constrained Input Systems

Control constrained by saturation function $\sigma(\cdot)$

Encode constraint into Value function

\[
J(u, d) = \int_0^\infty \left( Q(x) + 2 \int_0^u \sigma^{-T}(v) dv \right) dt
\]

\[
\|u\|_q^2 = 2 \int_0^u \sigma^{-T}(v) dv
\]

(Used by Lyshevsky for $H_2$ control)

This is a quasi-norm

Weaker than a norm –
homogeneity property is replaced by the weaker symmetry property

\[
\|x\|_q = \|-x\|_q
\]

Then

\[
u = -\sigma \left( R^{-1} g(x)^T \frac{\partial V}{\partial x} \right)
\]

Is BOUNDED
Kung Tz  500 BC
Confucius

Tian xia da tong
Harmony under heaven

Archery
Chariot driving

Music
Rites and Rituals

Poetry
Mathematics

Man’s relations to
Family
Friends
Society
Nation
Emperor
Ancestors

124 BC - Han Imperial University in Chang-an
II. Data-Driven Control

Autotuning
Iterative Learning Control
Improving Existing Built-in Process Controllers
Structured Process Controllers - XY-Table Example
Data-Driven Control

What is Supervisory Process Control?

1. Tuning of Controller
2. Computing Setpoints for an Inner-loop Controller
Data-driven Online Adaptive Optimal Control

1. Tuning of Controller

User Input
User prescribed optimization criterion

\[ J = (Q, R) \]

On-line Performance Loop

\[ x^T(t)P_kx(t) = \int_{t}^{t+T} x^T(\tau)(Q + K_k^TRK_k)x(\tau)d\tau + \int_{t}^{t+T} P_k(t+T)P_k(t+T) \]

\[ K_{k+1} = R^{-1}B^TP_k \]

An Online Supervisory Control Procedure
that requires no Knowledge of system dynamics model \( A \)

Based on 2 timescales

Automatically tunes the control gains in real time to optimize a user given cost function
Uses measured data \((u(t), x(t))\) along system trajectories

Relate to Supervisory Process Controllers
Autotuning

1. Tuning of Controller

Online tuning of PID gains in real time for unknown plants

\[ u = K \left( e + \frac{1}{T_i} \int_0^t e \, d\tau + T_d \frac{de}{dt} \right) \]

Tune control parameters \( K, T_i, T_d \) online using real time measurements from Plant.
Autotuning

1. Apply Relay feedback to find limit cycle

\[
T = \frac{2\pi}{\omega}
\]

\[
K_a = \frac{4d}{\pi a}
\]

\(\omega\) = oscillation frequency
Autotuning

2. Compute PID gains using Zeigler-Nichols

\[ u = K \left( e + \frac{1}{T_i} \int_0^t e \, d\tau + T_d \frac{de}{dt} \right) \]

<table>
<thead>
<tr>
<th></th>
<th>( K )</th>
<th>( T_i )</th>
<th>( T_d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>0.5( K_a )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PI</td>
<td>0.4( K_a )</td>
<td>0.8( T )</td>
<td></td>
</tr>
<tr>
<td>PID</td>
<td>0.6( K_a )</td>
<td>0.5( T )</td>
<td>0.12( T )</td>
</tr>
</tbody>
</table>

3. Switch in PID controller and run process
Result using PI gains from Autotuning

PI controller

System

\[ K_p \left( 1 + \frac{K_i}{s} \right) \]

\[ \frac{b}{s + a} \]

\[ \frac{1}{s} \]

\[ \frac{1}{0.005s + 1} \]

Control Input

Output

Time, seconds
Data-Driven Control

What is Supervisory Process Control?

2. Computing Setpoints for an Inner-loop Controller
Iterative Learning Control (ILC) 
Run-to-Run Control

Control oven temperature to give desired thickness of wafer thin film deposition

What goes here?

Desired thickness $y^D(t)$

Run-run update $e_k(t)$

Desired temp $u_k(t)$

Temp Ctrlr

Deposition thickness $y_k(t)$

Fixed Inner-loop temp ctrlr

Plant
Run-to-Run Control (ILC)  

CVD Thickness Control Problem

Desired thickness $y^D(t)$  

Desired temp $e_k(t)$  

Run-run update $u_k(t)$  

Temp Ctrlr  

Plant  

Fixed Inner-loop temp ctrlr

Temp. Control is constant

Can only measure end-point dep. error

Select control at $(k+1)$-st run to make error smaller

$y_k(t)$ thickness

temp

$u_k(t)$

$e_k(t)$

$u_{k+1}(t)$

$e_{k+1}(t)$
Run-to-Run Control (ILC)
Improving the Control at the Next Run

Plant

\[ y_k = P(z)u_k = z^{-d} P(z^{-1})u_k \]

\( d = \) system delay

Error

\[ e_k(t) = y^D(t) - y_k(t) \]

Arimoto Update Rule

\[ u_{k+1}(t) = u_k(t) + \gamma e_k(t + d) \]

Conditions on the plant \( P(z) \) for convergence of this rule
Fixing an Existing Poor Process Controller

2. Computing Setpoints for an Inner-loop Controller

Industrial Processes have built-in controllers that may not have good performance. Add a outer-loop controller that fixes the problem.

What goes here?

Desired thickness $y^D(t)$ → Error $e(t)$ → Desired temp → New ctrlr To fix it → Existing Ctrlr → Plant

Example- steady-state error is not zero
Add integral action to make Type I system

$$u(t) = K \int_0^t e \, d\tau$$
$$\dot{u} = Ke$$

Euler’s approximation with sample period $T$

$$\frac{u_{k+1} - u_k}{T} = Ke_k$$
$$u_{k+1} = u_k + KTe_k$$

Same as Arimoto Update Rule!

$$u_{k+1}(t) = u_k(t) + \gamma e_k(t + d)$$

$k$ is the run number.
Higher-Level Supervisory Control

Supervisory control as an outer feedback decision loop

An outer FB loop on a SLOWER TIME SCALE
This is like a managing Decision Loop

\[ u_{k+1} = u_k + KTe_k \]

Sends setpoints To inner ctrl
Shi nian shu mu
Bai nian shu ren

Keshi-
Wu nian su xuesheng
III. Data-driven Optimization
Using RL with Output feedback
Let's look at the Structure of an Industry Process Controller

**XY Table Motion Control**

Shankar Abhinav and Motab Almousa

$$G(S) = \frac{X(S)}{U(S)} = \frac{b}{s+a} = \frac{\text{displacement}}{\text{current}}$$

**XY Table model along one axis**

\[
\begin{align*}
A &= \begin{bmatrix} 0 & 1 \\ 0 & -11 \end{bmatrix}, & B &= \begin{bmatrix} 0 \\ 1 \end{bmatrix}, & C &= \begin{bmatrix} 1 & 0 \end{bmatrix}
\end{align*}
\]

**XY Table with Data filter for velocity measurements**

\[
\begin{align*}
A &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & -11 & 0 \\ 0 & 200 & -200 \end{bmatrix}, & B &= \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, & C &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{align*}
\]
Complete model

\[
\begin{align*}
Ax + Bu &= 0 \\
K_v \left( 1 + \frac{K_v}{s} \right) &= v
\end{align*}
\]

PI controller

Dynamics model

\[
\dot{x} = Ax + Bu
\]

YM table

Data filter

Control integrator

\[
x = \begin{bmatrix} x & v & v_f & \int e \, dt \end{bmatrix}
\]

Data Filter

Model includes the system, data filters, and control structure

\[
A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -11 & 0 & 0 \\ 0 & 200 & -200 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}
\]

\[
B = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}
\]

\[
C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\]

OPFB Controller

\[
u = -Ky
\]

\[
K = \begin{bmatrix} K_v & K & K_w \end{bmatrix}
\]
Output Feedback (OPFB)

Not state variable feedback (SVFB)

In practical industry process designs:
- All states are not measurable
- Only a reduced set of information is available – the outputs
- The controller has a structure that cannot be changed
- Only some control gains can be tuned

Only 3 control gains can be selected
Optimal Control- Linear Quadratic Regulator (LQR) with OUTPUT FEEDBACK (OPFB)

More realistic and practical

System dynamics \[ \dot{x} = Ax + Bu \]

Performance Index \[ V(x(t)) = \int_{t}^{\infty} (x^TQx + u^Tr u) \, d\tau = x^T(t)Px(t) \]

Minimum energy, minimum control effort

The Solution

Solve the OPFB Algebraic Riccati Equation (ARE)

\[ 0 = PA + A^TP + Q - PB R^{-1} B^TP + L^TR^{-1}L \]

For some design parameter matrix \( L \) such that

\[ KC = R^{-1}(B^TP + L) \]

Then the optimal OPFB feedback control is

\[ u = -Ky \]

OPFB LQR Solution Algorithm

Algorithm 2 (Offline PI Solution for Output-Feedback Control).

1) Start with an admissible control policy $K^0$ and $L = 0$.
2) (Policy evaluation) given a control input gain $K^i$, find the $P^i$ using the equation

\[
(A - BK^iC)^T P^i + P^i (A - BK^iC) + Q + C^T (K^i)^T R(K^i) C = 0. \tag{29}
\]

3) (Policy improvement) update the control policy and the matrix $L$ using

\[
K^{i+1} = R^{-1}(B^T P^i + L^i) C^T (C C^T)^{-1}
\]

\[
L^{i+1} = RK^{i+1} C - B^T P^i. \tag{30}
\]

OPFB Bellman equation

An Offline Design Procedure
That requires Full Knowledge of system dynamics model
System modeling is expensive, time consuming, and inaccurate
OPFB LQR Solution Algorithm

Algorithm 2 (Offline PI Solution for Output-Feedback Control).

1) Start with an admissible control policy $K^0$ and $L = 0$.
2) (Policy evaluation) given a control input gain $K^i$, find

$$x_t^T P^i x_t = \int_{t}^{t+T} x_T^T (Q + (K^i)^T R(K^i)) x_T dT$$

$$+ x_{t+T}^T P^i x_{t+T}.$$  \hspace{1cm} (29)

3) (Policy improvement) update the control policy and the matrix $L$ using

$$K^{i+1} = R^{-1}(B^T P^i + L^i)C^T (CC^T)^{-1}$$

$$L^{i+1} = RK^{i+1} C - B^T P^i.$$  \hspace{1cm} (30)

A Real-time design Procedure that uses measured data requires no knowledge of system dynamics matrix $A$
Finally! We can supervise a process and tune the allowable gains to get Optimal Performance!

User Input

User prescribed optimization criterion

\[ J = (Q, R) \]

An Online Supervisory Control Procedure that requires no Knowledge of system dynamics model A

Automatically tunes the OPFB control gains in real time to optimize a user given cost function

Uses measured data \((u(t), x(t))\) along system trajectories
An Online Supervisory Control Procedure that requires no Knowledge of system dynamics model A

Uses 2 timescales

Automatically tunes the OPFB control gains in real time to optimize a user given cost function

Uses measured data \((u(t), x(t))\) along system trajectories
Using the output feedback algorithm we get the optimal gains for the control of the XY table motion. The gains are as follows:

\[ K_p = 0.0887 \]

\[ K = 1.0435 \]

\[ K = 0.9528 \]