F.L. Lewis
UTA Research Institute (UTARI)
The University of Texas at Arlington, USA

Neural Network Learning Structures for
Real-Time Optimal Control and Dynamic Games

Talk available online at
http://ARRI.uta.edu/acs
He who exerts his mind to the utmost knows nature’s pattern.
The way of learning is none other than finding the lost mind.

Man’s task is to understand patterns in nature and society.
Optimal Control
Reinforcement learning
Integral Reinforcement Learning for Continuous-time systems
Policy Iteration for ZS games
Synchronous Policy Iteration
PI for Solving Dynamic Games online
Force Control

Flexible pointing systems
Simis labs, Inc. and US Army

SBIR Contracts

Vehicle active suspension
Leo Davis Technol.

Learning NN Controller
Relevance - Machine Feedback Control

High-Speed Precision Motion Control with unmodeled dynamics, vibration suppression, disturbance rejection, friction compensation, deadzone/backlash control

Industrial Machines

Vehicle Systems

Vehicle Suspension

Aerospace
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
Neural-adaptive Control During 1990s
Dynamic Neural Networks and Adaptive Control Learning

Extended adaptive controllers to nonlinear-in-parameters systems with Improved performance guarantees under Fewer assumptions


Neural-adaptive Control During 1990s
Dynamic Neural Networks and Adaptive Control Learning

- Bring together Computational Intelligence and Feedback Control
- We want to use powerful Computational Intelligence tools to improve the performance and reliability of feedback controllers
- Brought NN properties into adaptive control to produce a family of much-improved adaptive control systems
- Brought Backpropagation together with Stabilizing terms from Lyapunov Theory
Applications of Our Algorithms to Auto Engine Control


S. Jagannathan
11 US patents

optimal engine controllers based on RL for Caterpillar, Ford (Zetec engine), and GM

8-10% improvement in fuel efficiency and a drastic reduction in NOx (90%), HC (30%) and CO (15%) by operating with adaptive exhaust gas recirculation.

Savings to Caterpillar were over $65,000 per component.
Second Generation Reinforcement Learning for Feedback Control

- Bring together Computational Intelligence and Feedback Control
- We want to use powerful Computational Intelligence tools to improve the performance and reliability of feedback controllers

Paul Werbos- ADP
Cell Homeostasis

The individual cell is a complex feedback control system. It pumps ions across the cell membrane to maintain homeostasis, and has only limited energy to do so.

Permeability control of the cell membrane

Rocket Orbit Injection

Dynamics

\[ \dot{r} = w \]
\[ \dot{\theta} = \frac{v^2}{r} - \frac{\mu}{r^2} + \frac{F}{m} \sin \phi \]
\[ \dot{\phi} = -\frac{wv}{r} + \frac{F}{m} \cos \phi \]
\[ \dot{m} = -F \dot{m} \]

Objectives
- Get to orbit in minimum time
- Use minimum fuel
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
\[ J = \frac{1}{2} \int_0^\infty \left( \tanh(x^T Q x) + \rho \int_0^u \sigma^{-1}(v) dv \right) dt \]
Optimality and Games

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

Multi-player Games Occur in:
- Economics
- Control Theory disturbance rejection
- Team games
- International politics
- Sports strategy

But, optimal control and game solutions are found by
- Offline solution of Matrix Design equations
- A full dynamical model of the system is needed
RL has been developed for Discrete-Time Systems

Discrete-Time Optimal Control

System

\[ x_{k+1} = f(x_k) + g(x_k)u_k \]

Cost Performance Measure

\[ V(x_k) = \sum_{i=k}^{\infty} x_i^T Q x_i + u_i^T R u_i \]

Bellman optimality eq. = DT Hamilton-Jacobi-Bellman equation

\[ V^*(x_k) = \min_{u_k} \left[ x_k^T Q x_k + u_k^T R u_k + V^*(x_{k+1}) \right] \]

\[ = \min_{u_k} \left[ x_k^T Q x_k + u_k^T R u_k + V^* \left( f(x_k) + g(x_k)u_k \right) \right] \]

Minimize wrt \( u_k \)

\[ 2Ru_k + g(x_k)^T \frac{dV^*(x_{k+1})}{dx_{k+1}} = 0 \]

\[ u^*(x_k) = -\frac{1}{2} R^{-1} g(x_k)^T \frac{\partial V^*(x_{k+1})}{\partial x_{k+1}} \]

Offline solution
Difficult to solve
Contains the dynamics
DT Optimal Control – Linear Systems Quadratic cost (LQR)

System

\[ x_{k+1} = Ax_k + Bu_k \]

cost

\[ V(x_k) = \sum_{i=k}^{\infty} x_i^T Q x_i + u_i^T R u_i \]

Fact. The cost is quadratic

\[ V(x_k) = x_k^T P x_k \quad \text{for some symmetric matrix } P \]

HJB = DT Riccati equation

\[ 0 = A^T P A - P + Q - A^T P B (R + B^T P B)^{-1} B^T P A \]

Optimal Control \[ u_k = -L x_k \]

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

Optimal Cost

\[ V^*(x_k) = x_k^T P x_k \]

Off-line solution

Dynamics must be known
Optimal Control- The DT Linear Quadratic Regulator (LQR)

User prescribed optimization criterion

\[ J = (Q, R) \]

\[ 0 = A^T P A - P + Q - A^T P B (R + B^T P B)^{-1} B^T P A \]

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

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
Data-Based Method
Online in real-time
Using adaptive control techniques
Without knowing the full dynamics

For nonlinear systems and general performance indices

1. System dynamics
2. Value/cost function
3. Bellman equation
4. HJ solution equation (Riccati eq.)
5. Policy iteration – gives the structure we need
Discrete-Time Reinforcement Learning

System

\[ x_{k+1} = f(x_k) + g(x_k)u_k \]

cost

\[ V_h(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(x_i, u_i) \quad u_k = h(x_k) = \text{the prescribed control policy} \]

Optimal Control Solution

HJB Equation

\[ V^*(x_k) = \min_{u_k} (r(x_k, u_k) + \gamma V^*(x_{k+1})) \]

Optimal Control

\[ h^*(x_k) = \arg\min_{u_k} (r(x_k, u_k) + \gamma V^*(x_{k+1})) \]

For RL

Bellman eq.

\[ V_h(x_k) = r(x_k, h(x_k)) + \gamma V_h(x_{k+1}) \]

Stationarity Condition

\[ h'(x_k) = \arg\min_{u_k} (r(x_k, u_k) + \gamma V(x_{k+1})) \]

DT Hamiltonian

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

Focus on these two eqs
DT Policy Iteration

Cost for any given control policy \( h(x_k) \) satisfies the recursion

\[
V_h(x_k) = r(x_k, h(x_k)) + \gamma V_h(x_{k+1}) \quad \text{Bellman eq.}
\]

Recursive solution - Actor/Critic Structure

Pick stabilizing initial control

Policy Evaluation

\[
V_{j+1}(x_k) = r(x_k, h_j(x_k)) + \gamma V_{j+1}(x_{k+1}) \quad \text{f(.) and g(.) do not appear}
\]

Policy Improvement

\[
h_{j+1}(x_{k+1}) = \arg\min_{u_k} (r(x_k, u_k) + \gamma V_{j+1}(x_{k+1}))
\]

Temporal difference

\[
e_k = -V_{j+1}(x_k) + r(x_k, h_j(x_k)) + \gamma V_{j+1}(x_{k+1})
\]

Howard (1960) proved convergence for MDP

Bertsekas- rollout algorithms
Approximate Dynamic Programming (ADP)

Four ADP Methods proposed by Paul Werbos

Critic NN to approximate:

- Heuristic dynamic programming
  - Value Iteration
    - Value $V(x_k)$
  - Dual heuristic programming
    - Gradient $\frac{\partial V}{\partial x}$

AD Heuristic dynamic programming
  (Watkins Q Learning)
  - Q function $Q(x_k, u_k)$

AD Dual heuristic programming
  - Gradients $\frac{\partial Q}{\partial x}, \frac{\partial Q}{\partial u}$

Action NN to approximate the Control

Bertsekas- Neurodynamic Programming

Barto & Bradtke- Q-learning proof (Imposed a settling time)
ADP for Discrete-Time Systems

Work of Paul Werbos


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

---

**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- The Linear Quadratic Regulator (LQR)

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

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

Minimum energy, minimum control effort

The Solution

Solve the Algebraic Riccati Equation (ARE)
\[ 0 = PA + A^T P + Q - PBR^{-1}B^T P \]

Then the optimal feedback control is
\[ u = -R^{-1}B^T Px = -Kx \]

Result:
Let Q and R be symmetric and positive definite
Then the control \( u=-Kx \) minimizes \( V(x) \)
and makes the system stable and robust

MATLAB function \([K,P]=lqr(A,B,Q,R)\)
Optimal Control- The Linear Quadratic Regulator (LQR)

User prescribed optimization criterion

\[ J = (Q, R) \]

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

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

Control

\[ K \]

System

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

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 adaptive control techniques
Without knowing the full dynamics

For nonlinear systems and general performance indices

Adaptive Control Structures for:
A. Optimal control    B. Zero-sum games    C. Non zero-sum games

1. System dynamics
2. Value/cost function
3. Bellman equation
4. HJ solution equation (Riccati eq.)
5. Policy iteration – gives the structure we need
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 bring together Optimal Control and Adaptive Control!


Books


New Chapters on:
- Reinforcement Learning
- Differential Games

Continuous-Time Optimal Control

Nonlinear System dynamics \( \dot{x} = f(x,u) = f(x) + g(x)u \)

Cost/value \( V(x(t)) = \int_{t_i}^{\infty} r(x,u) \, dt = \int_{t_i}^{\infty} (Q(x) + u^T R u) \, dt \)

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

Off-line solution
HJB hard to solve. May not have smooth solution.
Dynamics must be known
Optimality in Biological Systems

Every living organism improves its control actions based on rewards received from the environment.

The resources available to living organisms are usually meager. Nature uses optimal control.

Reinforcement Learning

1. Apply a control. Evaluate the benefit of that control.
2. Improve the control policy.

RL finds optimal policies by evaluating the effects of suboptimal policies.
Different methods of learning

Reinforcement learning
Ivan Pavlov 1890s

We want OPTIMAL performance
- ADP- Approximate Dynamic Programming

Actor-Critic Learning
Desired performance

Sutton & Barto book

Continuous-Time 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
\[ V(x(t)) = \int_t^\infty r(x, u) \, dt = \int_t^\infty (Q(x) + u^T Ru) \, dt \]

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

To avoid solving HJB equation

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

Find cost for any given admissible $u(x)$

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

Utility

$$r(x, u) = Q(x) + u^T R u$$

CT Bellman equation
Scalar equation

Policy Iteration Solution

Pick stabilizing initial control policy

Policy Evaluation - Find cost, Bellman eq.

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

$$V_j(0) = 0$$

Policy improvement - Update control

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

• 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


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

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

This can be done online without knowing \( f(x) \)

using measurements of \( x(t), u(t) \) along the system trajectories

system \quad \dot{x} = f(x) + g(x)u

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

Key Idea

Lemma 1 – Draguna Vrabie

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

Is equivalent to \quad Integral reinf. form for the CT Bellman eq.

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

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

Allows definition of temporal difference error for CT systems

\[ e(t) = -V(x(t)) + \int_{t}^{t+T} r(x, u) \, d\tau + V(x(t+T)) \]
Lemma 1 - D. Vrabie- LQR case

\[ A_c^T P + P A_c + K^T RK + Q = 0 \]

Lyapunov Matrix equation

\[ A_c = A - BK \]

is equivalent to Integral reinforcement form

\[ x^T(t)P x(t) = \int_{t}^{t+T} x^T(\tau)(Q+K^T RK)x(\tau)d\tau + x^T(t+T)P x(t+T) \]

Scalar equation

Solves Lyapunov equation without knowing A or B
Integral Reinforcement Learning (IRL)- Draguna Vrabie

IRL Policy iteration

Policy evaluation- IRL Bellman Equation
Cost update
\[ V_k(x(t)) = \int_t^{t+T} r(x, u_k) \, dt + V_k(x(t+T)) \]

CT Bellman eq.

\[ f(x) \text{ and } g(x) \text{ do not appear} \]

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

Policy improvement

Control gain update
\[ u_{k+1} = h_{k+1}(x) = -\frac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x} \]

\[ g(x) \text{ needed for control update} \]

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
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

$$x^T(t)P_kx(t) = \int_{t}^{t+T} x^T(\tau)(Q+K_k^TRK_k)x(\tau)\,d\tau + x^T(t+T)P_kx(t+T)$$

$$x^T(t)P_kx(t) - x^T(t+T)P_kx(t+T) = \int_{t}^{t+T} x^T(\tau)(Q+K_k^TRK_k)x(\tau)\,d\tau$$

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

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

$$= \bar{p}_k^T[\bar{x}(t) - \bar{x}(t+T)]$$

Quadratic basis set
Nonlinear Case- Approximate Dynamic Programming

Value Function Approximation (VFA) to Solve Bellman Equation

– Paul Werbos (ADP), Dimitri 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 \( V = W^T \phi(x) \)

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

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

Scalar equation with vector unknowns

regression vector

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

Now use RLS along the trajectory to get new weights \( W_{k+1} \)

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 \([t, t+T)\)

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

Observe \(x(t)\)

Apply \(u^k = L_k x\)

Observe cost integral \(\rho(t, t+T)\)

Update \(P\)

Observe \(x(t+T)\)

Apply \(u^k = L_k x\)

Observe cost integral \(\rho(t+T, t+2T)\)

Update \(P\)

Observe \(x(t+2T)\)

Apply \(u^k = L_k x\)

Observe cost integral \(\rho(t+2T, t+3T)\)

Update \(P\)

Do RLS until convergence to \(P_k\)

Or use batch least-squares

**A is not needed anywhere**

Update control gain

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

This is a data-based approach that uses measurements of \(x(t), u(t)\) instead of the plant dynamical model.
Continuous-time control with discrete gain updates

Gain update (Policy)

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
Implementation

Policy evaluation
Need to solve online

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

Add a new state: Integral Reinforcement

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

This is the controller dynamics or memory
A New Direct Optimal Adaptive Controller

Solves Riccati Equation Online without knowing A matrix
Data-driven Learning in Real Time

Run RLS or use batch L.S.
To identify value of current control

Update FB gain after Critic has converged

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
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

User prescribed optimization criterion

\[ J = (Q, R) \]

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

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

Control \( K \)

System

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

On-line Control Loop

On-line Performance Loop

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

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

\((x(t), \rho(t, t+T), x(t+T))\)
Simulation 1- F-16 aircraft pitch rate controller

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

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

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

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

Select quadratic NN basis set for VFA

Exact solution \[W_1^* = [p_{11} \ 2p_{12} \ 2p_{13} \ p_{22} \ 2p_{23} \ p_{33}]^T\]

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

**A matrix not needed**

![System states graph]

- **Control signal**
  - Time (s): 0, 0.5, 1, 1.5, 2
  - Control signal values: 0, 0.5, 1, 1.5

- **Controller parameters**
  - Time (s): 0, 0.5, 1, 1.5, 2
  - Parameter values: 0, 0.5, 1, 1.5

- **Critic parameters**
  - Time (s): 0, 0.5, 1, 1.5, 2
  - Parameter values: 0, 0.5, 1, 1.5

**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

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

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

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

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

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

IRL period of T= 0.1s.

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

Hence, the value estimate was updated every 1.5s.

**System states**

**P matrix parameters P(1,1), P(1,3), P(2,4), P(4,4)**
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

$$J = \frac{1}{2} \int_0^\infty \left( \tanh(x^T Q x) + \rho \int_0^u \sigma^{-1}(v) \, dv \right) \, dt$$
Pythagoras 500 BC

Natural Philosophy, Ethics, and Mathematics

Music, Mathematics, Gymnastics, Astronomy, Medicine

The School of Pythagoras  *esoterikoi* and *exoterikoi*

Mathematikoi - *learners*

Translate music to mathematical equations

Ratios in music

Numbers and the harmony of the spheres

Serenity and Self-Possession

Patterns in Nature

Fire, air, water, earth
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
Oscillation is a fundamental property of neural tissue

Brain has multiple adaptive clocks with different timescales

**gamma rhythms** 30-100 Hz, hippocampus and neocortex
  - high cognitive activity.
  - consolidation of memory
  - spatial mapping of the environment – place cells

The high frequency processing is due to the large amounts of sensorial data to be processed

**theta rhythm**, Hippocampus, Thalamus, 4-10 Hz
  - sensory processing, memory and voluntary control of movement.

Spinal cord

Motor control 200 Hz


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
Summary of Motor Control in the Human Nervous System

- Cerebral cortex
- Motor areas
- Thalamus
- Basal ganglia
- Cerebellum
- Brainstem
- Spinal cord
- Exteroceptive receptors
- Interoceptive receptors
- Muscle contraction and movement

Memory functions:
- Long term
- Short term

Limbic System:
- Hippocampus

Reinforcement Learning - dopamine

Hierarchy of multiple parallel loops

Motor control 200 Hz

Gamma rhythms 30-100 Hz

Theta rhythms 4-10 Hz

Supervised learning

Unsupervised learning

Reflex

Work with Dan Levine

picture by E. Stingu D. Vrabie
Adaptive Critic structure

Reinforcement learning

Theta waves 4-8 Hz

Motor control 200 Hz
IRL for Data-driven Online Solution of Differential Games

Zero-sum 2-Player Games and H-infinity Control
Synchronous Real-time Data-driven Optimal Control
Synchronous Solution of 2-Player Zero-sum Games
Synchronous Solution of Multi-player Non Zero-sum Games
H-Infinity Control Using Neural Networks

Disturbance Rejection

System

\[
\begin{align*}
\dot{x} &= f(x) + g(x)u + k(x)d \\
y &= h(x) \\
z &= \begin{bmatrix} y^T & u^T \end{bmatrix}^T \\
u &= l(x)
\end{align*}
\]

Performance output

\[
\int_0^\infty \|z(t)\|^2 dt = \int_0^\infty (h^T h + \|u\|^2) dt \leq \gamma^2
\]

For all \(L_2\) disturbances

And a prescribed gain \(\gamma^2\)

\(L_2\) Gain Problem

Find control \(u(t)\) so that

Zero-Sum differential game

Nature as the opposing player
Online Zero-Sum Differential Games

H-infinity Control

System
\[
\dot{x} = f(x, u) = f(x) + g(x)u + k(x)d \\
y = h(x)
\]

Cost
\[
V(x(t), u, d) = \int_{t}^{\infty} \left( h^T h + u^T R u - \gamma^2 \|d\|^2 \right) dt = \int_{t}^{\infty} r(x, u, d) dt
\]

Differential equivalent is ZS game Bellman equation
\[
0 = r(x, u, d) + \dot{V} = r(x, u, d) + \left( \nabla V \right)^T (f(x) + g(x)u + k(x)d) \equiv H(x, \frac{\partial V}{\partial x}, u, d)
\]

\[
V(0) = 0
\]

Given any stabilizing control and disturbance policies \( u(x), d(x) \)

the cost value is found by solving this nonlinear Lyapunov equation


Define 2-player zero-sum game as

\[
V^*(x(0)) = \min_u \max_d V(x(0), u, d) = \min_u \max_d \int_0^\infty \left( h^T(x) h(x) + u^T R u - \gamma^2 \|d\|^2 \right) dt
\]

The game has a unique value (saddle-point solution) iff the Nash condition holds

\[
\min_u \max_d V(x(0), u, d) = \max_d \min_u V(x(0), u, d)
\]

A necessary condition for this is the Isaacs Condition

\[
\min_u \max_d H(x, \nabla V, u, d) = \max_d \min_u H(x, \nabla V, u, d)
\]

Stationarity Conditions

\[
0 = \frac{\partial H}{\partial u}, \quad 0 = \frac{\partial H}{\partial d}
\]
Game saddle point solution found from Hamiltonian - BELLMAN EQUATION

\[ H(x, \frac{\partial V}{\partial x}, u, d) = h^T h + u^T Ru - \gamma^2 \|d\|^2 + (\nabla V)^T (f(x) + g(x)u + k(x)d) = 0 \]

Optimal control/dist. policies found by stationarity conditions

\[ 0 = \frac{\partial H}{\partial u}, \quad 0 = \frac{\partial H}{\partial d} \]

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

\[ d = \frac{1}{2\gamma^2} k^T (x) \nabla V \]

HJI equation

\[ 0 = H(x, \nabla V, u^*, d^*) \]

\[ = h^T h + \nabla V^T (x)f(x) - \frac{1}{4} \nabla V^T (x)g(x)R^{-1} g^T (x) \nabla V (x) + \frac{1}{4\gamma^2} \nabla V^T (x)kk^T \nabla V (x) \]

\[ V(0) = 0 \]

(‘Nonlinear Game Riccati’ equation)
Linear Quadratic Zero-Sum Games

\[
\dot{x} = Ax + B_1 u_1 + B_2 u_2 \\
y = Cx \\
-J_2(x(t), u_1, u_2) = J_1(x(t), u_1, u_2) = \frac{1}{2} \int_t^\infty (x^T Q x + u_1^T R_{11} u_1 - u_2^T R_{12} u_2) \, d\tau,
\]

Game Algebraic Riccati Equation

\[
0 = A^T P + PA + Q - PB_1 R_{11}^{-1} B_1^T P + PB_2 R_{12}^{-1} B_2^T P \\
u_1 = -K_1 x \equiv -R_{11}^{-1} B_1^T P x, \quad u_2 = K_2 x \equiv R_{12}^{-1} B_2^T P x
\]

Want to Solve online without knowing A matrix
Using data-driven learning
Double Policy Iteration Algorithm to Solve HJI

Add inner loop to solve for available storage

Start with stabilizing initial policy $u_0(x)$

1. For a given control policy $u_j(x)$ solve for the value $V_{j+1}(x(t))$

2. Set $d^0 = 0$. For $i=0,1,...$ solve for $V^i_j(x(t))$, $d^{i+1}$

\[
0 = h^T \gamma + \nabla V^i_j(x)(f + gu_j + kd^i) + u_j^T Ru_j - \gamma^2 \left\| d^i \right\|^2
\]

\[
d^{i+1} = \frac{1}{2\gamma^2}k^T(x)\nabla V^i_j
\]

On convergence set $V_{j+1}(x) = V^i_j(x)$

3. Improve policy:

\[
u_{j+1}(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla V_{j+1}
\]

- Convergence proved by Van der Schaft if can solve nonlinear Lyapunov equation exactly
- Abu Khalaf & Lewis used NN to approximate $V$ for nonlinear systems and proved convergence

Off-line solution

Nonlinear Lyapunov equation must be solved at each step
Online Policy Iteration for 2-player ZS games

Options:
1. Both players learn online (two critics) to optimize their behavior policies
   a) simultaneously
   b) taking turns – while one is learning the other player maintains a fixed policy

2. Only one player learns online => single critic
   - the other player uses a fixed policy and only updates it at discrete moments based on information on the policy of his opponent

\[ u = -B^T P^i_u x \]
\[ \dot{x} = Ax + B_2 u + B_1 w; \quad x_0 \]
\[ w = B_1^T P^i_w x \]

Parameters that define the policies of the players
Online Nash equilibrium Learning

The game is played as follows:
1. The game starts while Player 2 (the disturbance) does not play.
2. Player 1
   a. plays the game without opponent and
   b. uses reinforcement learning to find the optimal behavior which
      minimizes its value;
   c. then informs Player 2 on his new optimized value fn.
3. Player 2 starts playing using the value fn. of his opponent.
4. Player 1
   a. corrects iteratively his own behavior using reinforcement learning such
      that its value is again minimized;
   b. then informs Player 2 on his new optimized value fn.
5. Go to step 3 until the two policies are characterized by the
   same parameter values.
Policy Iteration for Online Zero-Sum Games

The game is played as follows:

1. \( i = 1; \quad P_u^{i-1} = P_u^0 = 0; \quad w_1 = B_1^T P_u^0 x = 0 \)

2. Player 1 solves online, using HDP, the Riccati equation

\[
P_u^1 A + A^T P_u^1 - P_u^1 B_2 B_2^T P_u^1 + C^T C = 0
\]

\[
u_1 = -B_2^T P_u^1 x
\]

then informs Player 2 on \( P_u^1 \)

3. Player 2 uses the value \( P_u^i \) of Player 1. Computes his policy \( w_i = B_1^T P_u^i x \)

4. Player 1 solves online, using HDP, the Riccati equation;

\[
Z_u^i A_u^{i-1} + A_u^{i-1T} Z_u^i - Z_u^i B_2 B_2^T Z_u^i + Z_u^{i-1} B_1 B_1^T Z_u^{i-1} = 0
\]

\[
P_u^i = Z_u^i + P_u^{i-1}
\]

\[
u_i = -B_2^T P_u^i x
\]

then informs Player 2 on \( P_u^i \)

5. Set \( i = i + 1 \). Go to step 3 until the two policies are characterized by the same parameter values.

Riccati equations can be solved using HDP without knowledge of the A matrix.

Convergence proven by Lanzon, Feng, Brian Anderson 2009

Draguna Vrabie
Integral Reinforcement Learning (IRL) to solve ARE- Draguna Vrabie

CT Bellman eq.

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

Solves Lyapunov equation without knowing A or B

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

Only B is needed

Converges to solution to ARE

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

This is a data-based approach that uses measurements of x(t), u(t)
Instead of the plant dynamical model.
Actor-Critic structure - three time scales

\[ P_u^{i(k)} = P_u^{i-1} + Z_u^{i(k)} \]

\[ P_u^{i(k-1)} = P_u^{i-1} + Z_u^{i(k-1)} \]

Controller/Player 1:
\[ u = -B^T_i P_u^{i(k-1)} x \]

System:
\[ \dot{x} = Ax + B_2 u + B_1 w; x_0 \]

Disturbance/Player 2:
\[ w = B_1^T P_w^{i-1} x \]

Critic Learning procedure:
\[ \dot{V} = x^T C^T C x + \hat{u}^T \hat{u}, \text{ if } i=1 \]
\[ \dot{V} = w^T \hat{w} + \hat{u}^T \hat{u}, \text{ if } i>1 \]
Simulation- H-inf control for Electric Power Plant- LFC

\[ \dot{x} = Ax + B_2u + B_1d \]

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

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

\[
B_2 = \begin{bmatrix}
0 \\
0 \\
1/T_G \\
0 \\
\end{bmatrix}
\]

\[
B_1 = \begin{bmatrix}
-8 \\
0 \\
0 \\
0 \\
\end{bmatrix}
\]

\[
0 = A^T P + PA + C^T C - P(B_2B_2^T - B_1B_1^T)P
\]

A is unknown
B_1, B_2 are known

Frequency
Generator output
Governor position
Integral control
Load disturbance
Simulation result – Electric Power Plant LFC

• System – Power plant - internally stable system;
  – system state \( x = [\Delta f(t) \; \Delta P_g(t) \; \Delta X_g(t) \; \Delta E(t)] \)
  (incremental changes of: frequency deviation, generator output, governor position and integral control)
  – Player 1 - controller; Player 2 – load disturbance

• Nash equilibrium solution

\[
P_u^\infty = \Pi = \begin{bmatrix}
0.6036 & 0.7398 & 0.0609 & 0.5877 \\
0.7398 & 1.5438 & 0.1702 & 0.5978 \\
0.0609 & 0.1702 & 0.0502 & 0.0357 \\
0.5877 & 0.5978 & 0.0357 & 2.3307
\end{bmatrix}
\]

• Online learned solution using ADP – after 5 updates of the parameters of Player 2

\[
P_u^5 = \begin{bmatrix}
0.6036 & 0.7399 & 0.0609 & 0.5877 \\
0.7399 & 1.5440 & 0.1702 & 0.5979 \\
0.0609 & 0.1702 & 0.0502 & 0.0357 \\
0.5877 & 0.5979 & 0.0357 & 2.3307
\end{bmatrix}
\]

Solves GARE online without knowing \( A \)

\[
0 = A^T P + PA + C^T C - P(B_2 B_2^T - B_1 B_1^T)P
\]
Parameters of the critic – ARE Solution elements

- Parameters of the cost function of the game

- Cost function learning using least squares
- Sampling integration time $T=0.1$ s
- The policy of Player 1 is updated every 2.5 s
- The policy of Player 2 is updated only when the policy of Player 1 has converged
- Number of updates of Player 1 before an update of Player 2

moments when Player 2 is updated
Synchronous Real-time Data-driven Optimal Control
Synchronous Solution of 2-Player Zero-sum Games
Synchronous Solution of Multi-player Non Zero-sum Games
Synchronous
Online Solution of Optimal Control for Nonlinear Systems

Optimal Adaptive Control

Policy Iteration gives the structure needed for online optimal solution

A new structure of adaptive controllers
CT Policy Iteration – a Reinforcement Learning Technique

To avoid solving HJB equation

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

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

Cost for any given admissible \( u(x) \)

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

CT Bellman equation

Policy Iteration Solution

Pick stabilizing initial control policy

**Policy Evaluation** - Find cost, Bellman eq.

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

\[ V_j(0) = 0 \]

**Policy improvement** - Update control

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

- 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
Synchronous
Online Solution of Optimal Control for Nonlinear Systems

Optimal Adaptive Control

Policy Iteration gives the structure needed for online optimal solution

Need to solve online:

Bellman eq. for Value

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

Control update

\[ h(x) = -\frac{1}{2} R^{-1} g^T (x) \nabla V \]
Solve by parameterizing value $V(x)$

*Value Function Approximation (VFA)*—*Paul Werbos*

converts Bellman PDE into algebraic equation

**Critic NN**

Take VFA as

$$V(x) = W_1^T \phi_1(x) + \varepsilon(x), \quad \nabla V(x) = \nabla \phi_1^T W_1$$

Then Bellman eq

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

becomes

$$H(x, W_1, u) = W_1^T \nabla \phi_1 (f + gu) + Q(x) + u^T Ru = \varepsilon_H$$

$W_1 =$ LS solution to this eq for given $N$. Unknown.

**Action NN for Control Approximation**

$$u(x) = -\frac{1}{2} R^{-1} g^T (x) \nabla \phi_1^T \hat{W}_2,$$

Comes from

$$u(x) = -\frac{1}{2} R^{-1} g^T (x) \nabla V$$

$$\nabla V(x) = \nabla \phi_1^T W_1$$
Online Synchronous Policy Iteration

Theorem (Kyriakos Vamvoudakis)- Online Learning of Nonlinear Optimal Control

Let $\sigma_1 \equiv \nabla \phi_1(f + gu)$ be PE. Tune critic NN weights as

$$\dot{\hat{W}}_1 = -a_1 \frac{\partial E_1}{\partial \hat{W}_1} = -a_1 \frac{\sigma_1}{(\sigma_1 \sigma_1 + 1)^2} [\sigma_1^T \hat{W}_1 + Q(x) + u^T R u]$$

Learning the Value

Tune actor NN weights as

$$\dot{\hat{W}}_2 = -\alpha_2 \{(F_2 \hat{W}_2 - F_1 \sigma_1^T \hat{W}_1) - \frac{1}{4} \overline{D}_1(x) \hat{W}_2 m^T(x) \hat{W}_1\}$$

Learning the control policy

where $\overline{D}_1(x) \equiv \nabla \phi_1(x) g(x) R^{-1} g^T(x) \nabla \phi_1^T(x)$, $m \equiv \frac{\sigma_1}{(\sigma_1 \sigma_1 + 1)^2}$

Then there exists an $N_0$ such that, for the number of hidden layer units $N > N_0$

the closed-loop system state, the critic NN error $\tilde{W}_1 = W_1 - \hat{W}_1$

and the actor NN error $\tilde{W}_2 = W_1 - \hat{W}_2$ are UUB bounded.
Summary Nota Bene

Control policy

\[ u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1^T \hat{W}_2 \]

Tune critic NN weights as

\[ \dot{\hat{W}}_1 = -a_1 \frac{\partial E_1}{\partial \hat{W}_1} = -a_1 \frac{\sigma_1}{(\sigma_1^T \sigma_1 + 1)^2} [\sigma_1^T \hat{W}_1 + Q(x) + u^T Ru] \]

Tune actor NN weights as

\[ \dot{\hat{W}}_2 = -\alpha_2 \left\{ (F_2 \hat{W}_2 - F_1 \sigma_1^T \hat{W}_1) - \frac{1}{4} D_1(x) \hat{W}_2 m^T(x) \hat{W}_1 \right\} \]

Note, it does not work to simply set

\[ u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1^T \hat{W}_1 \]

Must have TWO NNs
Lyapunov energy-based Proof:

\[
L(t) = V(x) + \frac{1}{2} \text{tr}(\bar{W}_1^T a_1^{-1} \bar{W}_1) + \frac{1}{2} \text{tr}(\bar{W}_2^T a_2^{-1} \bar{W}_2).
\]

\(V(x) = \text{Unknown solution to HJB eq.}\)

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

Guarantees stability

\[
\bar{W}_1 = W_1 - \hat{W}_1
\]

\[
\bar{W}_2 = W_1 - \hat{W}_2
\]

\(W_1 = \text{Unknown LS solution to Bellman equation for given } N\)

\[
H(x,W_1,u) = W_1^T \nabla \phi_1(f + gu) + Q(x) + u^T Ru = \epsilon_H
\]
ONLINE solution
Does not require solution of HJB or nonlinear Lyapunov eq.

Does require system dynamics to be known

Finds approximate local smooth solution to NONLINEAR HJB equation online

An optimal adaptive controller
‘indirect’ because it identifies parameters for VFA
‘direct’ because control is directly found from value function
A New Adaptive Control Structure with Multiple Tuned Loops

Adaptive Critics

The Adaptive Critic Architecture

Value update - solve Bellman eq.

\[ V(x) = W_1^T \phi_1(x) \]

Control policy update

\[ u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1^T \hat{W}_2 \]

Critic and Actor tuned simultaneously
Leads to ONLINE FORWARD-IN-TIME implementation of optimal control

Optimal Adaptive Control
Adaptive Control

Identify the performance value - Optimal Adaptive

Identify the system model - Indirect Adaptive

Identify the Controller - Direct Adaptive

\[ V(x) = W^T \varphi(x) \]
Simulation 1 - F-16 aircraft pitch rate controller

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

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

Select quadratic NN basis set for VFA

Exact solution \[W_1^* = [p_{11} \quad 2p_{12} \quad 2p_{13} \quad p_{22} \quad 2p_{23} \quad p_{33}]^T\]

\[= [1.4245 \quad 1.1682 \quad -0.1352 \quad 1.4349 \quad -0.1501 \quad 0.4329]^T\]

Must add probing noise to get PE

\[u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1^T \hat{W}_2 + n(t) \quad \text{(exponentially decay } n(t))\]

Algorithm converges to

\[\hat{W}_1(t_f) = [1.4279 \quad 1.1612 \quad -0.1366 \quad 1.4462 \quad -0.1480 \quad 0.4317]^T.\]

\[\hat{W}_2(t_f) = [1.4279 \quad 1.1612 \quad -0.1366 \quad 1.4462 \quad -0.1480 \quad 0.4317]^T\]

Stevens and Lewis 2003

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

Solves ARE online

\[0 = PA + A^T P + Q - PBR^{-1}B^T P\]
Critic NN parameters- Converge to ARE solution

System states
Simulation 2. – Nonlinear System

\[ \dot{x} = f(x) + g(x)u, \quad x \in \mathbb{R}^2 \]

\[ f(x) = \begin{bmatrix} -x_1 + x_2 \\ -0.5x_1 - 0.5x_2 (1 - (\cos(2x_1) + 2)^2) \end{bmatrix} \]

\[ g(x) = \begin{bmatrix} 0 \\ \cos(2x_1) + 2 \end{bmatrix}. \]

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

Optimal Value \[ V^*(x) = \frac{1}{2} x_1^2 + x_2^2 \]

Optimal control \[ u^*(x) = -(\cos(2x_1) + 2)x_2. \]

Select VFA basis set \[ \phi_1(x) = [x_1^2 \quad x_1 \quad x_2^2]^T, \]

Algorithm converges to

\[ \hat{W}_1(t_f) = [0.5017 \quad -0.0020 \quad 1.0008]^T. \]

\[ \hat{W}_2(t_f) = [0.5017 \quad -0.0020 \quad 1.0008]^T. \]

\[ \hat{u}_2(x) = -\frac{1}{2} R^{-1} \begin{bmatrix} 0 \\ \cos(2x_1) + 2 \end{bmatrix} \begin{bmatrix} 2x_1 \quad 0 \\ x_2 \quad x_1 \end{bmatrix} \begin{bmatrix} 0.5017 \\ -0.0020 \\ 1.0008 \end{bmatrix}. \]

Converse optimal

Solves HJB equation online

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

System States

Optimal value fn.

Value fn. approx. error

Control approx error
Can avoid knowledge of drift term $f(x)$ by using Integral Reinforcement Learning (IRL)

Draguna Vrabie

Replace CT Bellman Equation

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

By the Integral reinforcement form for the CT Bellman eq.

$$V(x(t-T)) = \int_{t-T}^{t} r(x,u) \, d\tau + V(x(t)), \quad V(0) = 0$$
Data-driven Online Synchronous Policy Iteration using IRL

Does not need to know \( f(x) \)

Replace \( \sigma_1 \equiv \nabla \phi_1 (f + gu) \) by \( \Delta \phi(x(t)) \equiv \phi(x(t)) - \phi(x(t-T)) \)

**Theorem (Vamvoudakis & Vrabie)- Online Learning of Nonlinear Optimal Control**

Let \( \Delta \phi(x(t)) \equiv \phi(x(t)) - \phi(x(t-T)) \) be PE. Tune critic NN weights as

\[
\dot{\hat{W}}_1 = -a_1 \frac{\Delta \phi(x(t))^T}{\left(1 + \Delta \phi(x(t))^T \Delta \phi(x(t))\right)^2} \left(\Delta \phi(x(t))^T \hat{W}_1 + \int_{t-T}^{t} \left(Q(x) + \frac{1}{4} \hat{W}_2^T \bar{D}_1 \hat{W}_2\right) d\tau\right)
\]

Learning the Value

Tune actor NN weights as

\[
\dot{\hat{W}}_2 = -a_2 \left(F_2 \hat{W}_2 - F_1 \Delta \phi(x(t))^T \hat{W}_1\right) - \frac{1}{4} a_2 \bar{D}_1(x) \hat{W}_2 \frac{\Delta \phi(x(t))^T}{\left(1 + \Delta \phi(x(t))^T \Delta \phi(x(t))\right)^2} \hat{W}_1
\]

Learning the control policy

Then there exists an \( N_0 \) such that, for the number of hidden layer units \( N > N_0 \)

the closed-loop system state, the critic NN error \( \hat{W}_1 = W_1 - \hat{W}_1 \)

and the actor NN error \( \hat{W}_2 = W_1 - \hat{W}_2 \) are UUB bounded.

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

\[\left(x(t), \rho(t-T,t), x(t-T)\right)\]
Synchronous Online Zero-Sum Differential Games

Disturbance Rejection - H-infinity Control

System\[
\dot{x} = f(x, u) = f(x) + g(x)u + k(x)d
\]
\[y = h(x)\]

Cost\[
V(x(t), u, d) = \int_{t}^{\infty} \left( h^T h + u^T Ru - \gamma^2 \|d\|^2 \right) dt = \int_{t}^{\infty} r(x, u, d) dt
\]

Differential equivalent is ZS game Bellman equation

\[0 = r(x, u, d) + \dot{V} = r(x, u, d) + (\nabla V)^T (f(x) + g(x)u + k(x)d) \equiv H(x, \frac{\partial V}{\partial x}, u, d)\]

\[V(0) = 0\]

Given any stabilizing control and disturbance policies \(u(x), d(x)\)

the cost value is found by solving this nonlinear Lyapunov equation

Double Policy Iteration Algorithm to Solve HJI

Add inner loop to solve for available storage
Start with stabilizing initial policy \( u_0(x) \)

1. For a given control policy \( u_j(x) \) solve for the value \( V_{j+1}(x(t)) \)

2. Set \( d^0 = 0 \). For \( i=0,1,... \) solve for \( V^i_j(x(t)), d^{i+1} \)

\[
0 = h^T h + \nabla V^i_j(x)(f + gu_j + kd^i) + u_j^T Ru_j - \gamma^2 \|d^i\|^2
\]

\[
d^{i+1} = \frac{1}{2\gamma^2} k^T(x)\nabla V^i_j
\]

On convergence set \( V_{j+1}(x) = V^i_j(x) \)

3. Improve policy:

\[
u_{j+1}(x) = -\frac{1}{2} R^{-1} g^T(x)\nabla V_{j+1}
\]

- Convergence proved by Van der Schaft if can solve nonlinear Lyapunov equation exactly
- Abu Khalaf & Lewis used NN to approximate \( V \) for nonlinear systems and proved convergence

Off-line solution
Nonlinear Lyapunov equation must be solved at each step
Online Solution of ZS Games for Nonlinear Systems

Optimal (Game) Adaptive Control

Policy Iteration gives the structure needed for online solution

Need to solve online these 3 equations:

ZS game Bellman eq. for Value

\[ 0 = h^T h + \nabla V^T (x)(f + gu + kd) + u^T Ru - \gamma^2 \|d\|^2 \]

Disturbance update

\[ d = \frac{1}{2\gamma^2} k^T (x) \nabla V \]

Control update

\[ u(x) = -\frac{1}{2} R^{-1} g^T (x) \nabla V \]
Use Three Neural Networks

Critic NN for VFA

\[ \hat{V}(x) = \hat{W}_1^T \phi_1(x) \]

Bellman eq becomes algebraic eq.

\[ H(x, \hat{W}_1, u) = \hat{W}_1^T \nabla \phi_1 (f + gu + kd) + h^T h + u^T Ru - \gamma^2 \| d \|^2 = e_1 \]

Control Actor NN

\[ u(x) = -\frac{1}{2} R^{-1} g^T (x) \nabla \phi_1^T \hat{W}_2 \]

Comes from

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

Disturbance actor NN

\[ d(x) = \frac{1}{2\gamma^2} k^T (x) \nabla \phi_1^T \hat{W}_3, \]

Comes from

\[ d = \frac{1}{2\gamma^2} k^T (x) \nabla V \]

Simultaneously:

a. Solve Bellman eq.

and

b. update \( u(x), d(x) \)
Online Synchronous Policy Iteration for ZS games

Theorem (Kyriakos Vamvoudakis)- Online Gaming

Let \( \sigma_2 = \nabla \phi_1 (f \, g\theta + k\theta) \) be PE. Tune critic NN weights as

\[
\dot{\hat{W}}_1 = -a_1 \frac{\sigma_2}{(\sigma_2^2 + 1)^2} [\sigma_2^T \hat{W}_1 + h^T Qh - \gamma^2 \|d\|^2 + u^T Ru]
\]

Learning the Value

Tune actor NN weights as

\[
\begin{align*}
\dot{\hat{W}}_2 &= -\alpha_2 \{(F_2 \hat{W}_2 - F_1 \sigma_2^T \hat{W}_1) - \frac{1}{4} \overline{D}_1(x) \hat{W}_2 m^T (x) \hat{W}_1\} \\
\dot{\hat{W}}_3 &= -\alpha_3 \left\{ (F_4 \hat{W}_3 - F_3 \sigma_2^T \hat{W}_1) + \frac{1}{4 \gamma^2} \overline{E}_1(x) \hat{W}_3 m^T \hat{W}_1 \right\}
\end{align*}
\]

Learning the control policies

where

\[
\begin{align*}
\overline{D}_1(x) &= \nabla \phi_1(x) g(x) R^{-1} g^T(x) \nabla \phi_1^T(x), \\
\overline{E}_1(x) &= \nabla \phi_1(x) kk^T \nabla \phi_1^T(x), \\
m &= \frac{\sigma_2}{(\sigma_2^2 + 1)^2}
\end{align*}
\]

Then there exists an \( N_0 \) such that, for the number of hidden layer units

the closed-loop system state, the critic NN error

\( \tilde{W}_1 = W_1 - \hat{W}_1 \)

and the actor NN errors

\( \tilde{W}_2 = W_2 - \hat{W}_2, \quad \tilde{W}_3 = W_3 - \hat{W}_3 \)

are UUB bounded.
Actor-Critic structure –
A New Adaptive Controller with three tuned loops

\[ \dot{V} = h^T h + u^T \frac{R}{T} u - \frac{\gamma^2}{T} \|d\|^2 \]

System:
\[ \dot{x} = f(x) + g(x)u + k(x)d \]

Controller state-memory

Critic NN
Learning procedure

Controller/Player 1

Disturbance/Player 2

A novel form of Hybrid Controller
ONLINE solution
Does not require solution of HJI eq, HJ eq, or nonlinear Lyapunov eq.

Does require system dynamics to be known

Finds approximate local smooth solution to NONLINEAR HJI equation online

An optimal adaptive controller
   ‘indirect’ because it identifies parameters for VFA
   ‘direct’ because control is directly found from value function
Simulation 1- F-16 aircraft pitch rate controller

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

\[y = C^T x\]

\[Q = C^T C = I, \quad R = I\]

Solves GARE online

\[A^T P + PA + Q - PBR^{-1} B^T P + \frac{1}{\gamma^2} PKK^T P = 0\]

Exact solution

\[W_1 = \begin{bmatrix}
p_{11} & 2p_{12} & \cdots & 2p_{13} & p_{22} & \cdots & 2p_{23} & p_{33}
\end{bmatrix}^T\]

\[= \begin{bmatrix}
1.6573 & 1.3954 & -0.1661 & 1.6573 & -0.1804 & 0.4371
\end{bmatrix}^T\]

Must add probing noise to \(u(x)\) and \(d(x)\) to get PE

\[u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1^T \hat{W}_2 + n(t)\]

(exponentially decay \(n(t)\))

Algorithm converges to

\[
\hat{W}_1(t_f) = \begin{bmatrix}
1.7090 & 1.3303 & -0.1629 & 1.7354 & -0.1730 & 0.4468
\end{bmatrix}^T.
\]

\[
\hat{W}_2(t_f) = \hat{W}_3(t_f) = \hat{W}_4(t_f)
\]

\[\hat{d}(x) = \frac{1}{2\gamma^2} \begin{bmatrix}
x_2 & x_1 & 0 & 0 & 0 & \gamma
\end{bmatrix}^T\]

\[= \begin{bmatrix}
1.7090 \\
1.3303 \\
-0.1629 \\
1.7354 \\
-0.1730 \\
0.4468
\end{bmatrix}\]
Critic NN parameters

System states
F-16 aircraft pitch rate controller

Critic NN parameters
With disturbance

Critic NN parameters
Without disturbance

Converges FASTER with an opponent
One learns faster with an adversary
Simulation 3. – Nonlinear System

\[
\dot{x} = f(x) + g(x)u + k(x)d, \quad x \in \mathbb{R}^2
\]

\[
f(x) = \begin{bmatrix}
-x_1 + x_2 \\
-x_1^3 - x_2^3 + 0.25x_2\cos(2x_1) + 2 - 0.25x_2 \frac{1}{\gamma^2}\sin(4x_1) + 2^2
\end{bmatrix}
\]

\[
g(x) = \begin{bmatrix}
0 \\
\cos(2x_1) + 2
\end{bmatrix}, \quad k(x) = \begin{bmatrix}
0 \\
\sin(4x_1) + 2
\end{bmatrix}.
\]

\[
Q = I, \quad R = I, \quad \gamma = 8
\]

Optimal Value \[ V^*(x) = \frac{1}{4}x_1^4 + \frac{1}{2}x_2^2 \]

Saddle point solution \[ u^*(x) = -(\cos(2x_1) + 2)x_2, \quad d^*(x) = \frac{1}{\gamma^2}(\sin(4x_1) + 2)x_2 \]

Solves HJI eq. online \[ 0 = h^T h + \nabla V^T (x)f(x) - \frac{1}{4}\nabla V^T (x)g(x)R^{-1}g^T(x)\nabla V(x) + \frac{1}{4\gamma^2}\nabla V^T (x)kk^T \nabla V(x) \]

Select VFA basis set \[ \varphi_1(x) = [x_1^2, x_2^2, x_1^4, x_2^4] \]

Algorithm converges to \[ \hat{W}_1(t_f) = [0.0008, 0.4999, 0.2429, 0.0032]^T \]
\[ \hat{W}_2(t_f) = \hat{W}_3(t_f) = \hat{W}_1(t_f) \]

\[
\hat{u}_2(x) = -\frac{1}{2}R^{-1}\begin{bmatrix}
0 \\
\cos(2x_1) + 2 \\
4x_1^3 \\
0
\end{bmatrix}^T \begin{bmatrix}
2x_1 & 0 \\
0 & 2x_2 \\
4x_1^3 & 0 \\
0 & 4x_2^3
\end{bmatrix} \begin{bmatrix}
0.0008 \\
0.4999 \\
0.2429 \\
0.0032
\end{bmatrix}
\]

\[
d(x) = \frac{1}{2\gamma^2}\begin{bmatrix}
0 \\
\sin(4x_1) + 2 \\
4x_1^3 \\
0
\end{bmatrix}^T \begin{bmatrix}
2x_1 & 0 \\
0 & 2x_2 \\
4x_1^3 & 0 \\
0 & 4x_2^3
\end{bmatrix} \begin{bmatrix}
0.0008 \\
0.4999 \\
0.2429 \\
0.0032
\end{bmatrix}
\]
Manufacturing as the Interactions of Multiple Agents

Each machine has its own dynamics and cost function.

Neighboring machines influence each other most strongly.

There are local optimization requirements as well as global necessities.
Real-Time Solution of Multi-Player NZS Games

Kyriakos Vamvoudakis

Multi-Player Nonlinear Systems
\[ \dot{x} = f(x) + \sum_{j=1}^{N} g_j(x)u_j \]
Continuous-time, \( N \) players

Optimal control
\[ V_i^*(x(0), \mu_1, \mu_2, \ldots, \mu_N) = \min_{\mu_i} \int_{0}^{\infty} \left( Q_i(x) + \sum_{j=1}^{N} \mu_i^{T}R_{ij}\mu_i \right) dt; \quad i \in N \]

Nash equilibrium
\[ V_i^* \equiv V_i(\mu_1^*, \mu_2^*, \ldots, \mu_N^*) \leq V_1(\mu_1^*, \mu_2^*, \ldots, \mu_N^*), \quad i \in N \]

Requires Offline solution of coupled Hamilton-Jacobi –Bellman eqs.

\[ 0 = (\nabla V_i)^T \left( f(x) - \frac{1}{2} \sum_{j=1}^{N} g_j(x)R_{jj}^{-1}g_j^T(x)\nabla V_j \right) + Q_i(x) + \frac{1}{4} \sum_{j=1}^{N} \nabla V_j^T g_j(x)R_{jj}^{-1}R_{ij}R_{jj}^{-1}g_j^T(x)\nabla V_j, \quad V_i(0) = 0 \]

Control policies
\[ \mu_i(x) = -\frac{1}{2} R_{ii}^{-1}g_i^T(x)\nabla V_i, \quad i \in N \]

Linear Quadratic Regulator Case- coupled AREs

\[ 0 = P_iA_c + A_c^TP_i + Q_i + \sum_{j=1}^{N} P_jB_jR_{jj}^{-1}R_{ij}R_{jj}^{-1}B_j^TP_j, \quad i \in N \]

These are hard to solve
In the nonlinear case, HJB generally cannot be solved
Team Interest vs. Self Interest

The objective functions of each player can be written as a *team average* term plus a *conflict of interest* term:

\[
J_1 = \frac{1}{3} (J_1 + J_2 + J_3) + \frac{1}{3} (J_1 - J_2) + \frac{1}{3} (J_1 - J_3) \equiv J_{\text{team}} + J_1^{\text{coi}}
\]

\[
J_2 = \frac{1}{3} (J_1 + J_2 + J_3) + \frac{1}{3} (J_2 - J_1) + \frac{1}{3} (J_2 - J_3) \equiv J_{\text{team}} + J_2^{\text{coi}}
\]

\[
J_3 = \frac{1}{3} (J_1 + J_2 + J_3) + \frac{1}{3} (J_3 - J_1) + \frac{1}{3} (J_3 - J_2) \equiv J_{\text{team}} + J_3^{\text{coi}}
\]

For N-players

\[
J_i = \frac{1}{N} \sum_{j=1}^{N} J_j + \frac{1}{N} \sum_{j=1}^{N} (J_i - J_j) \equiv J_{\text{team}} + J_i^{\text{coi}}, \quad i = 1, N
\]

For *N-player zero-sum games*, the first term is zero, i.e. the players have no goals in common.
Real-Time Solution of Multi-Player Games

Non-Zero Sum Games – Synchronous Policy Iteration

Kyriakos Vamvoudakis

Value functions

\[ V_i(x(0), \mu_1, \mu_2, \ldots \mu_N) = \int_0^\infty (Q_i(x) + \sum_{j=1}^N \mu_i^T R_{ij} \mu_j) \, dt; \quad i \in N \]

Differential equivalent gives coupled Bellman eqs.

\[ 0 = Q_i(x) + \sum_{j=1}^N u_j^T R_{ij} u_j + (\nabla V_i)^T (f(x) + \sum_{j=1}^N g_j(x) u_j) \equiv H_i(x, \nabla V_i, u_1, \ldots, u_N), \quad i \in N \]

Policy Iteration Solution:

Solve Bellman eq.

\[ 0 = r(x, \mu_1^k, \ldots, \mu_N^k) + (\nabla V_i^k)^T \left( f(x) + \sum_{j=1}^N g_j(x) \mu_j^k \right), \quad V_i^k(0) = 0 \quad i \in N \]

Policy Update

\[ \mu_i^{k+1}(x) = -\frac{1}{2} R_{ii}^{-1} g_i^T(x) \nabla V_i^k, \quad i \in N \]

Convergence has not been proven
Hard to solve Hamiltonian equation
But this gives the structure we need for online Synchronous PI Solution
Policy Iteration gives the structure needed for online solution

Need to solve online:

**Coupled Bellman eqs.**

\[
0 = Q_i(x) + \sum_{j=1}^{N} u_j^T R_{ij} u_j + (\nabla V_i)^T (f(x) + \sum_{j=1}^{N} g_j(x)u_j) = H_i(x, \nabla V_i, u_1, ..., u_N), \quad i \in N
\]

**Control policies**

\[
\mu_i(x) = -\frac{1}{2} R_{ii}^{-1} g_i^T(x) \nabla V_i, \quad i \in N
\]

Each player needs 2 NN – a Critic and an Actor
Real-Time Solution of Multi-Player Games

Kyriakos Vamvoudakis

Online Synchronous PI Solution for Multi-Player Games

Each player needs 2 NN – a Critic and an Actor

2-player case

Player 1

\[ \hat{V}_1(x) = \hat{W}_1^T \phi_1(x), \]

Player 2

\[ \hat{V}_2(x) = \hat{W}_2^T \phi_2(x). \]

N Critic Neural Networks for VFA

\[ u_1(x) = -\frac{1}{2} R_{11}^{-1} g_1^T(x) \nabla \phi_1^T \hat{W}_3, \]

\[ u_2(x) = -\frac{1}{2} R_{22}^{-1} g_2^T(x) \nabla \phi_2^T \hat{W}_4. \]

N Actor Neural Networks

On-Line Learning – for Player 1:

\[ \dot{\hat{W}}_1 = -a_1 \sigma_1 \frac{\sigma_1}{(\sigma_1^T \sigma_1 + 1)^2} [\sigma_1^T \hat{W}_1 + Q_1(x) + u_1^T R_{11} u_1 + u_2^T R_{12} u_2] \]

Learns Bellman eq. solution

\[ \dot{\hat{W}}_3 = -\alpha_3 \{(F_2 \hat{W}_3 - F_1 \sigma_3^T \hat{W}_1) - \frac{1}{4} \nabla \phi_1 g(x) R_{11}^{-1} R_{21} R_{11}^{-1} g^T(x) \nabla \phi_1^T \hat{W}_3 m_2^T \hat{W}_2 - \frac{1}{4} \bar{D}_1(x) \hat{W}_3 m_1^T \hat{W}_1} \]

Learns control policy
Lyapunov energy-based Proof:

\[ L(t) = V(x) + \frac{1}{2} \text{tr}(\tilde{W}_1^T a_1^{-1} \tilde{W}_1) + \frac{1}{2} \text{tr}(\tilde{W}_2^T a_2^{-1} \tilde{W}_2). \]

V(x) = Unknown solution to HJB eq.

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

Guarantees stability

\[ \tilde{W}_1 = W_1 - \hat{W}_1 \]
\[ \tilde{W}_2 = W_1 - \hat{W}_2 \]

\( W_1 = \) Unknown LS solution to Bellman equation for given N

\[ H(x, W_1, u) = W_1^T \nabla \phi_1(f + gu) + Q(x) + u^T Ru = \epsilon_H \]
Manufacturing as the Interactions of Multiple Agents

Each machine has its own dynamics and cost function
Neighboring machines influence each other most strongly
There are local optimization requirements as well as global necessities

\[
\dot{\delta}_i = A\delta_i + (d_i + g_i)B_i u_i - \sum_{j \in N_i} e_{ij} B_j u_j
\]

And cost function

\[
J_i(\delta_i(0), u_i, u_{-i}) = \frac{1}{2} \int_0^\infty (\delta_i^T Q_{ii} \delta_i + u_i^T R_{ii} u_i + \sum_{j \in N_i} u_j^T R_{ij} u_j) \, dt
\]

Each process helps other processes achieve optimality and efficiency
Graphical Games for Multi-Process Optimal Control

Optimal Performance of Each Process Depends on the Control of its Neighbor Processes

Control Policy of Each Process Depends on the Performance of its Neighbor Processes
Simulation. – Nonlinear System – 2-player game

\[ \dot{x} = f(x) + g(x)u + k(x)d, \quad x \in \mathbb{R}^2 \]

\[
\begin{align*}
  f(x) &= \begin{bmatrix} x_2 \\ -x_2 - \frac{1}{3} x_1 + \frac{1}{4} x_2 (\cos(2x_1) + 2)^2 + \frac{1}{4} x_2 (\sin(4x_1^2) + 2)^2 \end{bmatrix} \\
  g(x) &= \begin{bmatrix} 0 \\ \cos(2x_1) + 2 \end{bmatrix}, \quad k(x) = \begin{bmatrix} 0 \\ (\sin(4x_1) + 2) \end{bmatrix}.
\end{align*}
\]

\[ Q_1 = 2Q_2 = 2I, \quad R_{11} = 2R_{22} = 2I, \quad R_{12} = 2R_{21} = 2I \]

Optimal Value
\[ V_1^*(x) = \frac{1}{2} x_1^2 + x_2^2 \quad V_2^*(x) = \frac{1}{4} x_1^2 + \frac{1}{2} x_2^2 \]

Optimal Policies
\[ u^*(x) = -2(\cos(2x_1) + 2)x_2 \quad d^*(x) = -(\sin(4x_1^2) + 2)x_2 \]

Solves HJB equations online

\[
0 = (\nabla V_i)^T \left( f(x) - \frac{1}{2} \sum_{j=1}^N g_j(x) R_{ij}^{-1} g_j^T(x) \nabla V_j \right) + Q_i(x) + \frac{1}{4} \sum_{j=1}^N \nabla V_j^T g_j(x) R_{ij}^{-1} g_j(x) \nabla V_j, \quad V_i(0) = 0
\]

Select VFA basis set
\[ \varphi_1(x) = \varphi_2(x) = [x_1^2 \ x_1x_2 \ x_2^2] \]

Algorithm converges to
\[ \hat{W}_1(t_f) = [0.5015 \ 0.0007 \ 1.0001]^T = \hat{W}_2(t_f) \]
\[ \hat{W}_2(t_f) = [0.2514 \ 0.0006 \ 0.5001]^T = \hat{W}_4(t_f) \]

\[
\begin{align*}
  \hat{u}(x) &= -\frac{1}{2} R_{11}^{-1} \begin{bmatrix} 0 \\ \cos(2x_1) + 2 \end{bmatrix} \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \end{bmatrix}^T [0.5015 \\ 0.0007 \ 1.0001] \\
  \hat{d}(x) &= -\frac{1}{2} R_{22}^{-1} \begin{bmatrix} 0 \\ \sin(4x_1^2) + 2 \end{bmatrix} \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \end{bmatrix}^T [0.2514 \\ 0.0006 \ 0.5001]
\end{align*}
\]
Critic 1 NN parameters  

Critic 2 NN parameters  

Evolution of the States  

3D approximation error value for player 1.  

3D approximation error of control for player 1.