F.L. Lewis
National Academy of Inventors

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

New Developments in Integral Reinforcement Learning:
Continuous-time Optimal Control and Games

Supported by:
ONR
US NSF

Talk available online at
http://www.UTA.edu/UTARI/acs

Supported by:
China NNSF
China Project 111
Data-driven Online Solution of Differential Games
Synchronous Solution of Multi-player Non Zero-sum Games
Multi-player Game Solutions
IEEE Control Systems Magazine, Dec 2017
Multi-player Differential Games

500 BC

Sun Tzu's

THE ART OF WAR

孫子兵法

Sun Tzu bin fa
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

Each process has its own dynamics
\[ \dot{\delta}_i = A\delta_i + (d_i + g_i)B_iu_i - \sum_{j \in N_i} e_{ij}B_ju_j \]

And cost function
\[ J_i(\delta_i(0),u_i,u_{-i}) = \frac{1}{2} \int_0^\infty (\delta_i^TQ_{ii}\delta_i + u_i^TR_{ii}u_i + \sum_{j \in N_i} u_j^TR_{ij}u_j) \, dt \]

Each process helps other processes achieve optimality and efficiency
Multi-Player Nonlinear Systems
\[ \dot{x} = f(x) + \sum_{j=1}^{N} g_j(x)u_j \]
Continuous-time, \( N \) players

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

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

Requires Offline solution of coupled Hamilton-Jacobi–Bellman eqs.
\[ 0 = (\nabla V_i)^T \left( f(x) - \frac{1}{2} \sum_{j=1}^{N} g_j(x) R_{jj}^{-1} g_j^T(x) \nabla V_i \right) + Q_i(x) + \frac{1}{4} \sum_{j=1}^{N} \nabla V_j^T g_j(x) R_{ij} R_{jj}^{-1} g_j^T(x) \nabla V_j, \quad V_i(0) = 0 \]

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
\[ 0 = P_i A_c + A_c^T P_i + Q_i + \sum_{j=1}^{N} P_j B_j R_{jj}^{-1} R_{ij} R_{jj}^{-1} B_j^T P_j, \quad i \in N \]

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:

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

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

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

For N-players

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

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

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
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

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

N Actor Neural Networks

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

On-Line Learning – for Player 1:

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

\[ \dot{\hat{W}}_3 = -\alpha_3 \{\hat{F}_2 \hat{W}_3 - \hat{F}_1 \sigma_3^T \hat{W}_1\} - \frac{1}{4} \nabla \phi_1 g(x) R_{11}^{-1} R_{21} R_{11}^{-1} g^T(x) \nabla \phi_1^T \hat{W}_3 m_2^T \hat{W}_2 - \frac{1}{4} D_1(x) \hat{W}_3 m_1^T \hat{W}_1 \]

Learns Bellman eq. solution

Learns control policy

Convergence is proven using Lyapunov functions

Multi-Agent Learning Guaranteed Convergence Proof

Lyapunov energy-based Proof:

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

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

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

Guarantees stability

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

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

\[ W_1 = \text{Unknown LS solution to Bellman equation for given N} \]

\[ H(x, W_1, u) = W_1^T \nabla \phi_1(f + gu) + Q(x) + u^T Ru = \varepsilon_H \]
Multi-player 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

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

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

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

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

Optimal Value s

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

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

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

Select VFA basis set

\[
\varphi_1(x) = \varphi_2(x) \equiv [x_1^2 \quad x_1x_2 \quad x_2^2]
\]

Algorithm converges to

\[
\hat{W}_1(t_f) = [0.5015 \quad 0.0007 \quad 1.0001]^T = \hat{W}_3(t_f)
\]

\[
\hat{W}_2(t_f) = [0.2514 \quad 0.0006 \quad 0.5001]^T = \hat{W}_4(t_f)
\]

\[
\hat{u}(x) = -\frac{1}{2} R_{11}^{-1} \begin{bmatrix}
0 \\
\cos(2x_1) + 2
\end{bmatrix}^T \begin{bmatrix}
2x_1 \\
x_2 \quad x_1 \\
0 \quad 2x_2
\end{bmatrix} \begin{bmatrix}
0.5015 \\
0.0007 \\
1.0001
\end{bmatrix}
\]

\[
\hat{d}(x) = -\frac{1}{2} R_{22}^{-1} \begin{bmatrix}
0 \\
\sin(4x_1^2) + 2
\end{bmatrix}^T \begin{bmatrix}
2x_1 \\
x_2 \quad x_1 \\
0 \quad 2x_2
\end{bmatrix} \begin{bmatrix}
0.2514 \\
0.0006 \\
0.5001
\end{bmatrix}
\]
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.
Games on Communication Graphs

500 BC

Sun Tz bin fa

Sun Tzu's
THE ART OF WAR

500 BC
Key Point

Lyapunov Functions and Performance Indices Must depend on graph topology


Hongwei Zhang, F.L. Lewis, and Abhijit Das
Graphical Games
Synchronization- Cooperative Tracker Problem

Node dynamics \[ \dot{x}_i = Ax_i + B_i u_i, \quad x_i(t) \in \mathbb{R}^n, \quad u_i(t) \in \mathbb{R}^{m_i} \]

Target generator dynamics \[ \dot{x}_0 = Ax_0 \]

Synchronization problem \[ x_i(t) \to x_0(t), \forall i \]

Local neighborhood tracking error (Lihua Xie)
\[ \delta_i = \sum_{j \in N_i} e_{ij}(x_i - x_j) + g_i(x_i - x_0), \]

Local nbhd. tracking error dynamics
\[ \dot{\delta}_i = A\delta_i + (d_i + g_i)B_iu_i - \sum_{j \in N_i} e_{ij}B_ju_j \]

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


New Differential Graphical Game

Control action of player $i$

State dynamics of agent $i$

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

Value function of player $i$

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

Central Dynamics

Local Value Function depends on ALL other control actions

Central Dynamics

\[ \dot{z} = Az + \sum_{i=1}^{N} B_i u_i \]

Value function of player \( i \)

\[ J_i(z(0), u_i, u_{-i}) = \frac{1}{2} \int_{0}^{\infty} (z^T Q z + \sum_{j=1}^{N} u_j^T R_{ij} u_j) \, dt \]

Control action of player \( i \)
Problems with Nash Equilibrium Definition on Graphical Games

Game objective

\[
V_i^*(\delta_i(t)) = \min_{u_i} \int_1^\infty \left( \frac{1}{2} (\delta_i^T Q_i \delta_i + u_i^T R_i u_i + \sum_{j \in N_i} u_j^T R_i u_j) \right) dt
\]

Define \( u_{-i}(t) = \{ u_j : j \in N - i \} \) Neighbors of node \( i \)

\( u_{G-i} = \{ u_j : j \in N, j \neq i \} \) All other nodes in graph

Def: Nash equilibrium

\[ \{ u_1^*, u_2^*, ..., u_N^* \} \] are in Nash equilibrium if

\[
J_i^* = J_i(u_i, u_{G-i}^*) \leq J_i(u_i, u_{G-i}) \quad \forall i \in N
\]

Counterexample. Disconnected graph

Then, each agent’s cost does not depend on any other agent

\[
J_i(u_i) = J_i(u_i, u_{G-i}) = J_i(u_i, u'_{G-i}) \quad \forall i
\]

Let each node play his optimal control

\[
J_i^* = J_i(u_i^*)
\]

Then all agents are in Nash equilibrium

Note- this Nash is also coalition-proof
New Definition of Nash Equilibrium for Graphical Games

To restore symmetry of Nash Equilibrium

Def: Interactive Nash equilibrium

\[ \{u_1^*, u_2^*, \ldots, u_N^*\} \] are in Interactive Nash equilibrium if

1. \[ J_i^* \triangleq J_i (u_i^*, u_{G-i}^*) \leq J_i (u_i, u_{G-i}^*), \quad \forall i \in N \]
2. There exists a policy \( u_j \) such that

\[ J_i (u_j, u_{G-j}^*) \neq J_i (u_j^*, u_{G-j}^*), \quad \forall i, j \in N \]

That is, every player can find a policy that changes the value of every other player.

Theorem 3. Let \( (A, B_i) \) be reachable for all \( i \).
Let agent \( i \) be in local best response

\[ J_i (u_i^*, u_{-i}) \leq J_i (u_i, u_{-i}), \quad \forall i \]

Then \( \{u_1^*, u_2^*, \ldots, u_N^*\} \) are in global Interactive Nash iff the graph is strongly connected.
Graphical Game Solution Equations

Value function

\[ V_i(\delta_i(t)) = \frac{1}{2} \int_t^\infty (\delta_i^T Q_i \delta_i + u_i^T R_i u_i + \sum_{j \in N_i} u_j^T R_{ij} u_j) \, dt \]

Differential equivalent (Leibniz formula) is Bellman’s Equation

\[ H_i(\delta_i, \frac{\partial V_i}{\partial \delta_i}, u_i, u_{-i}) = \partial V_i^T \left( A \delta_i + (d_i + g_i) B_i u_i - \sum_{j \in N_i} e_{ij} B_j u_j \right) + \frac{1}{2} \delta_i^T Q_i \delta_i + \frac{1}{2} u_i^T R_i u_i + \frac{1}{2} \sum_{j \in N_i} u_j^T R_{ij} u_j = 0 \]

Stationarity Condition

\[ 0 = \frac{\partial H_i}{\partial u_i} \quad \Rightarrow \quad u_i = -(d_i + g_i) R_i^{-1} B_i^T \frac{\partial V_i}{\partial \delta_i} \]

1. Coupled HJ equations

\[ \frac{\partial V_i^T}{\partial \delta_i} A_i^c + \frac{1}{2} \delta_i^T Q_i \delta_i + \frac{1}{2} (d_i + g_i)^2 \frac{\partial V_i^T}{\partial \delta_i} B_i R_i^{-1} B_i^T \frac{\partial V_i}{\partial \delta_i} + \frac{1}{2} \sum_{j \in N_i} (d_j + g_j)^2 \frac{\partial V_j^T}{\partial \delta_j} B_j R_j^{-1} R_{ij} B_j^T \frac{\partial V_j}{\partial \delta_j} = 0, \quad i \in N \]

\[ H_i(\delta_i, \frac{\partial V_i}{\partial \delta_i}, u_i^*, u_{-i}^*) = 0 \]

where

\[ A_i^c = A \delta_i - (d_i + g_i)^2 B_i R_i^{-1} B_i^T \frac{\partial V_i}{\partial \delta_i} + \sum_{j \in N_i} e_{ij} (d_j + g_j) B_j R_j^{-1} B_j^T \frac{\partial V_j}{\partial \delta_j}, \quad i \in N \]

Now use Synchronous PI to learn optimal Nash policies online in real-time as players interact

Distributed Multi-Agent Learning Proofs
Online Solution of Graphical Games

Multi-agent Learning Convergence proofs

Use Reinforcement Learning

POLICY ITERATION

Algorithm 1. Policy Iteration (PI) Solution for N-player distributed games.

Step 0: Start with admissible initial policies \( u_i^0 \), \( \forall i \).

Step 1: (Policy Evaluation) Solve for \( V_i^k \) using (14)
\[
H_i(\delta_i, \frac{\partial V_i^k}{\partial \delta_i}, u_i^k, u_{-i}^k) = 0, \forall i = 1, \ldots, N
\]
(38)

Step 2: (Policy Improvement) Update the N-tuple of control policies using
\[
u_i^{k+1} = \text{arg min}_{u_i} H_i(\delta_i, \frac{\partial V_i^k}{\partial \delta_i}, u_i, u_{-i}^k), \forall i = 1, \ldots, N
\]
which explicitly is
\[
u_i^{k+1} = -(d_i + g_i)R_{ii}^{-1}B_i^T\frac{\partial V_i^k}{\partial \delta_i}, \forall i = 1, \ldots, N.
\]
(39)

Go to step 1.

On convergence End

Convergence Results

Theorem 3. Convergence of Policy Iteration algorithm when only \( i \)-th agent updates its policy and all players \( u_{-i} \) in the neighborhood do not change. Given fixed neighbors policies \( u_{-i} \), assume there exists an admissible policy \( u_i \). Assume that agent \( i \) performs Algorithm 1 and the its neighbors do not update their control policies. Then the algorithm converges to the best response \( u_i \) to policies \( u_{-i} \) of the neighbors and to the solution \( V_i \) to the best response HJ equation (36).

The next result concerns the case where all nodes update their policies at each step of the algorithm. Define the relative control weighting as \( \rho_{ij} = \sigma(R_{ij}^{-1}R_{ij}) \), where \( \sigma(R_{ij}^{-1}R_{ij}) \) is the maximum singular value of \( R_{ij}^{-1}R_{ij} \).

Theorem 4. Convergence of Policy Iteration algorithm when all agents update their policies. Assume all nodes \( i \) update their policies at each iteration of PI. Then for small enough edge weights \( a_{ij} \) and \( \rho_{ij} \), \( \mu_i \) converges to the global Nash equilibrium and for all \( i \), and the values converge to the optimal game values \( V_i^k \rightarrow V_i^* \).
Data-driven Online Solution of Differential Games
Zero-sum 2-Player Games and H-infinity Control
H-Infinity Control Using Reinforcement Learning

Disturbance Rejection

System

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

Performance output

\[u = l(x)\]

disturbance

\[d \rightarrow u \rightarrow \text{control}\]

L₂ Gain Problem

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

\[
\begin{align*}
\int_0^\infty \|z(t)\|^2 dt &= \int_0^\infty (h^T h + \|u\|^2) dt \\
\int_0^\infty \|d(t)\|^2 dt &= \int_0^\infty \|d(t)\|^2 dt
\end{align*}
\]

For all L₂ disturbances

\[
\gamma^2 \leq \gamma^2
\]

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

Zero-Sum differential game - Nature as the opposing player

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

\[ 0 = \frac{\partial H}{\partial u} , \quad 0 = \frac{\partial H}{\partial d} \]
Linear Quadratic Zero-Sum Games

\[
\begin{align*}
\dot{x} &= Ax + B_1 u_1 + B_2 u_2 \\
y &= Cx
\end{align*}
\]

\[-J_2(x(t), u_1, u_2) = J_1(x(t), u_1, u_2) = \frac{1}{2} \int_t^\infty (x^T Q x + u_1^T R_{11} u_1 - u_2^T R_{12} u_2) \, d\tau, \quad Q = C^T C\]

Game Algebraic Riccati Equation

\[
0 = A^T P + PA + Q - PB_1 R_{11}^{-1} B_1^T P + PB_2 R_{12}^{-1} B_2^T P
\]

\[
u_1 = -K_1 x \equiv -R_{11}^{-1} B_1^T P x, \quad u_2 = K_2 x \equiv R_{12}^{-1} B_2^T P x
\]
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
\[ V(x(t), u, d) = \int_{t}^{\infty} \left( h^T h + u^T Ru - \gamma^2 \|d\|^2 \right) dt \equiv \int_{t}^{\infty} r(x, u, d) dt \]

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

Given any stabilizing control and disturbance policies \( u(x), d(x) \)
the cost value is found by solving this nonlinear Lyapunov equation


Game saddle point solution found from Hamiltonian - ZS Game 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)
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 \( V^i_j(x(t)), d^{i+1} \)

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

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

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

ONLINE SOLUTION- Use IRL to solve the ZS Bellman eq. and update the actors at each step
Actor-Critic structure - three time scales

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

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

\[ \dot{V} = x^T C^T C x + \dot{u}^T \hat{u}, \text{ if } i = 1 \]

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

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

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

\[ A = \begin{bmatrix} -1/T_p & K_p/T_p & 0 & 0 \\ 0 & -1/T_r & 1/T_r & 0 \\ -1/RT_G & 0 & -1/T_G & -1/T_G \\ K_E & 0 & 0 & 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 \\ 0 \\ 1/T_G \\ 0 \end{bmatrix} \]

\[ A = \begin{bmatrix} -0.0665 & 8 & 0 & 0 \\ 0 & -3.663 & 3.663 & 0 \\ -6.86 & 0 & -13.736 & -13.736 \\ 0.6 & 0 & 0 & 0 \end{bmatrix}, \quad B = [0 \quad 0 \quad 13.7355 \quad 0]^T, \quad B_1 = [-8 \quad 0 \quad 0 \quad 0]^T \]

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

A is unknown
B_1, B_2 are known
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

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

- Online learned solution using ADP – after 5 updates of the parameters

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

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

- Cost function learning using least squares
- Sampling integration time $T=0.1\text{ s}$
- 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
New Developments in IRL for CT Systems
Q Learning for CT Systems
Experience Replay
Off-Policy IRL
Q learning for CT Systems

\[ Q^\mu (x(t), \tilde{u}[t : t + T]) = \int_t^{t+T} e^{-\gamma (t-\tau)} \left( x^T S x + \mu^T R \mu \right) d\tau + e^{-\gamma T} V^\mu (x(t + T)) \]

\[ Z_k(t) = \begin{bmatrix} x(t) \\ u(t) \\ \vdots \\ u^{(k)}(t) \end{bmatrix} \]

\[ \int_t^{t+T} e^{-\gamma (t-\tau)} \left( x(\tau)^T S x(\tau) + u(\tau)^T R u(\tau) \right) d\tau = Z_{k-1}^T(t) \left( \tilde{M}_{k-1}^T S_{k-1} \tilde{M}_{k-1} + \tilde{R}_k \right) Z_{k-1}(t) + E_k^0 (t + T) \]


CT Q function is quadratic in x, u, and the derivatives of u -
IRL with Experience Replay
Humans use memories of past experiences to tune current policies

\[
x(t) = f(x(t)) + g(x(t))u(t)
\]

Value
\[
V(x(t)) = \int_{t}^{\infty} \left( Q(x(\tau)) + 2 \int_{0}^{u} (\lambda \tanh^{-1}(v/\lambda))^T R \, dv \right) d\tau
\]

Bellman Equation
\[
Q(x) + 2 \int_{0}^{u} (\lambda \tanh^{-1}(v/\lambda))^T R \, dv + \nabla V^T(x) (f(x) + g(x) u) = 0, \quad V(0) = 0
\]

IRL Bellman Equation
\[
V(x(t - T)) = \int_{t-T}^{t} \left( Q(x(\tau)) + 2 \int_{0}^{u} (\lambda \tanh^{-1}(v/\lambda))^T R \, dv \right) d\tau + V(x(t))
\]

Action Update
\[
u^* = -\lambda \tanh \left( (1/2\lambda)R^{-1}g^T(x) \nabla V^*(x) \right)
\]

VFA- Value Function Approximation
\[
\hat{V}(x) = \hat{W}_1^T \phi(x)
\]

Bellman Eq gives Linear Equation for Weights
\[
\int_{t-T}^{t} \left( Q(x(\tau)) + 2 \int_{0}^{u} (\lambda \tanh^{-1}(v/\lambda))^T R \, dv \right) d\tau + W_1^T \Delta \phi(x(t)) \equiv \varepsilon_B(t)
\]

i/o Data Measurements
\[
\Delta \phi(x(t)) = \phi(x(t)) - \phi(x(t - T))
\]
\[
p(t) = \int_{t-T}^{t} Q \left( + 2 \int_{0}^{u} (\lambda \tanh^{-1}(v/\lambda))^T R \, dv \right) d\tau
\]

Standard Critic Weight Tuning
\[
\hat{W}_1(t) = -\alpha_1 \frac{\Delta \phi(t)}{(1 + \Delta \phi(t)^T \Delta \phi(t))^2} \left( p(t) + \Delta \phi(t)^T \hat{W}_1(t) \right)
\]
IRL with Experience Replay

Humans use memories of past experiences to tune current policies

VFA- Value Function Approximation

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

i/o Data Measurements

\[ \Delta \phi(x(t)) = \phi(x(t)) - \phi(x(t - T)) \]

\[ p(t) = \int_{t-T}^{t} \left( Q + 2 \int_{0}^{u} \left( \lambda \tanh^{-1}(v/\lambda) \right)^T R \, dv \right) \, d\tau \]

Data from Previous time intervals

The samples are stored in a history stack. To collect data in the history stack, consider \( \Delta \phi_j \) and \( p_j \) as evaluated values of \( \Delta \phi(t) \) and \( p(t) \) (see (17) and (26)) at the recorded time \( t_j \). That is,

\[ \Delta \phi_j = \Delta \phi(t_j) = \phi(x(t_j)) - \phi(x(t_j - T)) \tag{27} \]

and

\[ p_j = p(t_j) = \int_{t_j-T}^{t_j} \left( Q + 2 \int_{0}^{u} \left( \lambda \tanh^{-1}(v/\lambda) \right)^T R \, dv \right) \, d\tau \tag{28} \]

NN weight tuning uses past samples

\[
\hat{W}_1(t) = -\alpha_1 \frac{\Delta \phi(t)}{(1 + \Delta \phi(t)^T \Delta \phi(t))^2} \left( p(t) + \Delta \phi(t)^T \hat{W}_1(t) \right) - \alpha_1 \sum_{j=1}^{1} \frac{\Delta \phi_j}{(1 + \Delta \phi_j^T \Delta \phi_j)^2} \left( p_j + \Delta \phi_j^T \hat{W}_1(t) \right)
\]

Improvements

1. Speeds up convergence
2. PE condition is milder

New Principles

Off-Policy Learning
Off-Policy Reinforcement Learning

Humans can learn optimal policies while actually playing suboptimal policies

On-policy RL

Target policy: The policy that we are learning about.
Behavior policy: The policy that generates actions and behavior

Target policy and behavior policy are the same

Sutton and Barto Book
Humans can learn optimal policies while actually applying suboptimal policies.

Target policy and behavior policy are different.


**Off-policy IRL**

Humans can learn optimal policies while actually applying suboptimal policies

System

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

Value

\[ J(x) = \int_t^\infty r(x(\tau), u(\tau)) \, d\tau = \int_t^\infty (Q(x) + u^T R u) \, d\tau \]

**On-policy IRL**

\[
\begin{align*}
J^{[i]}(x(t)) - J^{[i]}(x(t-T)) &= -\int_{t-T}^t Q(x) \, d\tau - \int_{t-T}^t u^{[i]T} R u^{[i]} \, d\tau \\
u^{[i+1]} &= -\frac{1}{2} R^{-1} g^T J_x^{[i]} \\
\end{align*}
\]

Must know \( g(x) \)

**Off-policy IRL**

\[
\begin{align*}
\dot{x} &= f + g u^{[i]} + g(u - u^{[i]}) \\
J^{[i]}(x(t)) - J^{[i]}(x(t-T)) &= -\int_{t-T}^t Q(x) \, d\tau - \int_{t-T}^t u^{[i]T} R u^{[i]} \, d\tau + 2\int_{t-T}^t u^{[i+1]T} R (u^{[i]} - u) \, d\tau \\
\end{align*}
\]

**DDO**

This is a linear equation for \( J^{[i]} \) and \( u^{[i+1]} \)

They can be found simultaneously online using measured data using Kronecker product and VFA

1. Completely unknown system dynamics
2. Can use applied \( u(t) \) for –
Off-policy for Multi-player NZS Games

\[
\dot{x} = f(x) + \sum_{j=1}^{N} g(x)u_j
\]

\[
V_i(x(t)) = \int_{t}^{\infty} (r_j(x,u_1,u_2,\ldots,u_N))d\tau = \int_{t}^{\infty} (Q_i(x) + \sum_{j=1}^{N} u_j^TR_ju_j)d\tau
\]

On-policy

Off-policy

\[
\dot{x} = f(x) + \sum_{j=1}^{N} g(x)u_j^{[k]} + \sum_{j=1}^{N} g(x)(u_j - u_j^{[k]})
\]

Algorithm 1:

Step 1: Start with stabilizing initial policies \(u_1^{[0]}, u_2^{[0]}, \ldots, u_N^{[0]}\)

Step 2: Given the N-tuple of policies \(u_1^{[k]}, u_2^{[k]}, \ldots, u_N^{[k]}\), solve for the N-tuple of costs \(V_1^{[k]}(x(t)), V_2^{[k]}(x(t)), \ldots, V_N^{[k]}(x(t))\) using

\[
0 = \nabla V_i^{[k]}(f(x) + \sum_{j=1}^{N} g(x)u_j^{[k]}) + r_i(x,u_1^{[k]},u_2^{[k]},\ldots,u_N^{[k]})
\]

with \(V_i^{[k]}(0) = 0\).

Step 3: Update the N-tuple of control policies using:

\[
u_i^{[k+1]} = \arg\min_{u_i} [H_i(x,\nabla V_i^{[k]},u_1,\ldots,u_N)]
\]

which explicitly is

\[
u_i^{[k+1]} = -\frac{1}{2}R_i^{-1}g(x)\nabla V_i^{[k]}
\]

\[
V_i^{[k]}(x(t+T)) - V_i^{[k]}(x(t)) = -\int_{t}^{t+T} Q_i(x)d\tau - \int_{t}^{t+T} \sum_{j=1}^{N} u_j^{[k]}^TR_ju_j^{[k]}d\tau - 2\int_{t}^{t+T} u_i^{[k+1]}^TR_i\sum_{j=1}^{N} (u_j - u_j^{[k]})d\tau
\]

DDO

1. Solve online using measured data for \(V_i^{[k]}, u_i^{[k+1]}\)
2. Completely unknown dynamics
3. Add exploring noise with no bias

Off-policy IRL for ZS Games – H-infinity Control

Optimal tracker

Extended state \( X(t) = [e_d(t)^T \ r(t)^T]^T \in \mathbb{R}^{2n} \)
Including ref. traj. dynamics
\[
\dot{X}(t) = F(X(t)) + G(X(t))u(t) + K(X(t))d(t)
\]

\[
V(u,d) = \int_t^\infty e^{-\alpha(t-\tau)}(X^T \ Q \ X + u^T \ R \ u - \gamma^2 d^T d) \ d\tau
\]

Off-policy

\[
\dot{X} = F + G \ u_i + K \ d_i + G(u - u_i) + K(d - d_i)
\]

DDO
1. Solve online using data for \( V_i, u_{i+1}, d_{i+1} \)
2. Completely unknown dynamics
3. Disturbance does not need to be specified
4. Add exploring noise with no bias

H. Li, Derong Liu, D. Wang, IEEE TASE 2015


Algorithm 2. Online Off-policy RL algorithm for solving tracking HJI equation
1. Initialization: Start with a control policy \( u_0 \)
2. Solve the following Bellman equation for \( V_i \) and \( u_{i+1} \) simultaneously
\[
e^{-\alpha(T-t)}V_i(X(t+T)) - V_i(X(t)) =
\int_t^{t+T} e^{-\alpha(\tau-t)}(-X^T Q \ X - u_i^T R u_i + \gamma^2 d_i^T d_i) \ d\tau
+ \int_t^{t+T} e^{-\alpha(\tau-t)}(-2u_{i+1}^T R(u-u_i) + 2\gamma^2 d_{i+1}^T (d-d_i)) \ d\tau
\] (67)

Observe played policy of disturbance
Compute his worst case malicious attack should he choose to play it
Off-Policy Learning for Estimating Malicious Adversaries’ Hidden True Intent

MA Systems
\[ \dot{x}_i = Ax_i + Bu_i + Dv_i \]

Two Opposing Teams

MAS H-infinity control

Cost
\[ V_i(x_i(t)) = \frac{1}{2} \int_t^\infty \left( x_i^T Q_i x_i + u_i^T R_i u_i + \sum_{j \in N_i} u_j^T R_{ij} u_j - \gamma^2 v_i^T T_i v_i - \gamma^2 \sum_{j \in N_i} v_j^T T_{ij} v_j \right) dt \]

Off-policy IRL
\[ \dot{x}_i = Ax_i + Bu_i^k + Dv_i^k + B(u_i - u_i^k) + D(v_i - v_i^k) \]

Off-Policy Bellman Eq.
\[ V_i^k(x_i(t)) - V_i^k(x_i(t+T)) = \frac{1}{2} \int_t^{t+T} r_i(x_i, u_i^k, v_i^k) dt - \int_t^{t+T} (u_i^k)^T R_i (u_i - u_i^k) - \gamma^2 (v_i^k)^T R_i (v_i - v_i^k) dt \]
New Principles

3. Off-Policy LQ Tracker for Continuous-time Systems
LQ Tracker for Continuous-time Systems – Follow a Leader

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

Assume the reference trajectory is generated by
\[ \dot{y}_d = Fy_d \]

Augmented system state
\[ X(t) = \begin{bmatrix} x(t) \\ y_d(t) \end{bmatrix} \]

Augmented system
\[ \dot{X} = \begin{bmatrix} A & 0 \\ 0 & F \end{bmatrix} X + \begin{bmatrix} B \\ 0 \end{bmatrix} u \equiv TX + Bu \]

Value function
\[ V(X(t)) = \frac{1}{2} \int_{t}^{\infty} e^{-\gamma(t-\tau)} \left[ X^T \dot{Q}_T X + u^T R u \right] d\tau = \frac{1}{2} X(t)^T P X(t) \]

\[ Q_T = C_1^T Q C_1, \quad C_1 = [C - I] \]

Optimal Control Solution

Optimal Tracker Solution by Reinforcement Learning

Tracker ARE

\[ T^T P + TP - \gamma P + C_1^TQC_1 - PB_1 R^{-1}B_1^T P = 0 \]

\[ K = [K_1, K_2] = -R^{-1}B_1^T P \]

Control

\[ u = K X = K_1 x + K_2 y_d \]
Online solution to the CT LQT ARE: Off-policy IRL

\[ \dot{X} = TX + B_1 u = T_i X + B_1 (K^i X + u). \quad T_i = T - B_1 K^i \]

\[ e^{-\gamma T} X(t + T)^T P^i X(t + T) - X(t)^T P^i X(t) = -\int_t^{t+T} \frac{d}{d\tau} (e^{-\gamma (\tau-t)} X^T P^i X) d\tau \]

\[ = -\int_t^{t+T} e^{-\gamma (\tau-t)} X^T (Q_T + K^i R K^i) X d\tau + 2\int_t^{t+T} e^{-\gamma (\tau-t)} (u + K^i X)^T B_1^T P^i X d\tau \]

Off-policy IRL Bellman equation

**Algorithm. Online Off-policy IRL algorithm for LQT**

**Online step:** Apply a fixed control input and collect some data

**Offline step:** Policy evaluation and improvement using LS on collected data

\[ e^{-\gamma T} X(t + T)^T P^i X(t + T) - X(t)^T P^i X(t) = -\int_t^{t+T} e^{-\gamma (\tau-t)} X^T Q_i X d\tau + 2\int_t^{t+T} e^{-\gamma (\tau-t)} (u + K^i X)^T R K^i X d\tau \]

No knowledge of the dynamics is required
Output Synchronization of Heterogeneous MAS
Output Synchronization of Heterogeneous MAS

Heterogeneous Multi-Agents

\[
\begin{align*}
\dot{x}_i &= A_i x_i + B_i u_i \\
y_i &= C_i x_i
\end{align*}
\]

Leader

\[
\begin{align*}
\dot{\zeta}_0 &= S \zeta_0 \\
y_0 &= R \zeta_0
\end{align*}
\]

Output regulation error

\[\eta_i(t) = y_i(t) - y_0(t) \to 0\]

Output regulator equations

\[
\begin{align*}
A_i \Pi_i + B_i \Gamma_i &= \Pi_i S \\
C_i \Pi_i &= R
\end{align*}
\]

Dynamics are different, state dimensions can be different
o/p reg eqs capture the common core of all the agents dynamics
And define a synchronization manifold
Standard Solution

MAS \[ \dot{x}_i = A_i x_i + B_i u_i \]
\[ y_i = C_i x_i \]

Leader \[ \dot{\zeta}_0 = S \zeta_0 \]
\[ y_0 = R \zeta_0 \]

o/p regulation \[ y_i(t) - y_0(t) \to 0, \forall i \]

Output regulator equations
\[ A_i \Pi_i + B_i \Gamma_i = \Pi_i S \]
\[ C_i \Pi_i = R \]

Control
\[ \dot{\zeta}_i = S \zeta_i + c \left[ \sum_{j=1}^{N} a_{ij} (\zeta_j - \zeta_i) + g_i (\zeta_0 - \zeta_i) \right] \]
\[ u_i = K_{1i} (x_i - \Pi_i \zeta_i) + \Gamma_i \zeta_i = K_{1i} x_i + (\Gamma_i - K_{2i} \Pi_i) \zeta_i \equiv K_{1i} x_i + K_{2i} \zeta_i \]

Must know your dynamics \( A_i, B_i \)
And leader’s dynamics \( S, R \)
And solve o/p regulator equations

Pioneered by Jie Huang
Optimal Output Synchronization of Heterogeneous MAS Using Off-policy IRL

\[
\begin{align*}
\dot{x}_i &= A_i x_i + B_i u_i \\
y_i &= C_i x_i \\
\dot{\zeta}_0 &= S \zeta_0 \\
y_0 &= R \zeta_0
\end{align*}
\]

**Optimal Tracker Problem**

Augmented Systems

\[
X(t) = \begin{bmatrix} x_i(t)^T & \zeta_0^T \end{bmatrix}^T \in \mathbb{R}^{n_i+p}
\]

\[
\dot{X}_i = T_i X_i + B_{ij} u_i
\]

\[
T_i = \begin{bmatrix} A_i & 0 \\ 0 & S \end{bmatrix}, B_{ij} = \begin{bmatrix} B_i \\ 0 \end{bmatrix}
\]

Performance index

\[
V(X_i(t)) = \int_t^\infty e^{-\gamma_i(t-\tau)} X_i^T (C_{ij}^T Q_i C_{ij} + K_i^T W_i K_i) X_i \, d\tau
\]

\[
= X_i(t)^T P_i X_i(t)
\]

Control

\[
u_i = K_{1i} x_i + K_{2i} \zeta_0 = K_i X_i
\]

Optimal Tracker Solution by Reinforcement Learning

\[ K_i = [K_{i1}, K_{i2}] = -W_i^{-1}B_{i1}^T P_i \]

Tracker ARE

\[ T_i^T P_i + T_i P_i - \gamma_i P_i + C_{i1}^T Q_i C_{i1} - P_i B_{i1} W_i^{-1} B_{i1}^T P_i = 0 \]

---

**Algorithm 1.** *On-policy IRL State-feedback algorithm*

Policy Evaluation- Solve IRL Bellman equation

\[ e^{-\gamma \delta t} X_i (t + \delta t) P_i^\kappa X_i (t + \delta t) - X_i (t)^T P_i^\kappa X_i (t) = - \int_t^{t+\delta t} e^{-\gamma (\tau-t)} (y_i - y_0)^T Q_i (y_i - y_0) d\tau \]

Policy Update-

\[ K_i^{\kappa+1} = [K_{i1}^{\kappa+1}, K_{i2}^{\kappa+1}] = -W_i^{-1} B_{i1}^T P_i^\kappa \]

---

Theorem- Algorithm 1 converges to the solution to the ARE

Bellman equation is solved using RLS or batch LS

It requires a Persistence of Excitation (PE) condition that may be hard to satisfy

Must know \( B_{ii} \)

**Control** \( u_i = K_{i1} x_i + K_{i2} \zeta_0 = K_i x_i \)
Off-Policy RL

Tracker dynamics

\[ \dot{X}_i = T_i X_i + B_{i1} u_i \]

Rewrite as

\[ \dot{X}_i = (T_i + B_{i1} K_i^\kappa)X_i + B_{i1} (u_i - K_i^\kappa X_i) \equiv T_i X_i + B_{i1} (u_i - K_i^\kappa X_i) \]

Now the Bellman equation becomes

\[
e^{-\gamma \delta t} X_i (t + \delta t)^T P_i^\kappa X_i (t + \delta t) - X_i (t)^T P_i^\kappa X_i (t) = -\int_t^{t+\delta t} e^{-\gamma (\tau-t)} (y_i - y_0)^T Q_i (y_i - y_0) d\tau \]
\[+ \quad 2\int_t^{t+\delta t} e^{-\gamma (\tau-t)} (u_i - K_i^\kappa X_i)^T W_i K_i^\kappa X_i d\tau \]

Extra term containing \( K_i^{\kappa+1} \)

Algorithm 2. Off-policy IRL Data-based algorithm

Iterate on this equation and solve for \( P_i^\kappa, K_i^{\kappa+1} \) simultaneously at each step.

Note about probing noise

If \( u_i = K_i^\kappa X_i + e \) then \( (u_i - K_i^\kappa X_i) = e \)

Do not have to know any dynamics

\[
\begin{align*}
\dot{x}_i &= A_i x_i + B_i u_i \\
y_i &= C_i x_i \\
\dot{z}_0 &= S \zeta_0 \\
y_0 &= R \zeta_0
\end{align*}
\]
Theorem- Off-policy Algorithm 2 converges to the solution to the ARE

\[ T_i^T P_i + T_i P_i - \gamma_i P_i + C_i^T Q_i C_i - P_i B_i W_i^{-1} B_i^T P_i = 0 \]

Theorem- o/p reg eq solution

Let \( P_i = \begin{bmatrix} P_{i11}^i & P_{i12}^i \\ P_{i21}^i & P_{i22}^i \end{bmatrix} \)

Then the solution to the output regulator equations

\[ A_i \Pi_i + B_i \Gamma_i = \Pi_i S \]
\[ C_i \Pi_i = R \]

Is given by

\[ \Pi_i = -(P_{11}^i)^{-1} P_{12}^i \]
\[ \Gamma_i = K_{2i} - K_{1i} (P_{11}^i)^{-1} P_{12}^i \]

Do not have to know the Agent dynamics or the leader’s dynamics (S,R)
To avoid knowledge of leader’s state in

\[ u_i = K_{1i} x_i + K_{2i} \dot{\zeta}_0 = K_i X_i \]

Use adaptive observer for leader’s state

\[ \dot{\zeta}_i = \hat{S}_i \zeta_i + c \left[ \sum_{j=1}^{N} a_{ij} (\zeta_j - \zeta_i) + g_i (\zeta_0 - \zeta_i) \right] \]

\[ \hat{S}_{vec_i} = -\Gamma_{si} (I_q \otimes \zeta_i) \left[ \sum_{j=1}^{N} a_{ij} (\zeta_j - \zeta_i) + g_i (\zeta_0 - \zeta_i) \right] \]

Then use control

\[ u_i = K_{1i} x_i + K_{2i} \dot{\zeta}_i \equiv K_i \dot{X}_i \equiv K_i \begin{bmatrix} x_i \\ \zeta_i \end{bmatrix} \]

Note that \( \hat{S}_i \) may not converge to actual leader’s matrix \( S \)

Do not have to know the leader’s dynamics \((S,R)\)
New Principles

There Appear to be Multiple Reinforcement Learning Loops in the Brain

Multiple Actor-Critic Learning Structures

Narendra MMAC - Multiple Model Adaptive Control
Recent work in Cognitive Neuropsychology reveals the interactions of brain regions in fast decision-making.

New research by Dan Levine

Limbic System -
Amygdala and Orbitofrontal Cortex (OFC)
Anterior Cingulate Cortex (ACC) and
dorsolateral prefrontal cortex (DLPFC)
Dan Levine, Neural Dynamics of affect, gist, probability, and choice, Cognitive Sys Research, 2012

The human brain operates with MULTIPLE REINFORCEMENT LOOPS

**ACC and DLPFC**
Deliberative decision & control based on more real-time data

If there is mismatch or dissonance
ACC sends reset to OFC to form new category

If there is risk or stress, ACC recruits DLPFC
For deliberative decision & control based on real-time data
Perhaps optimality?

Hippocampus is invoked for detailed task knowledge

**Basal ganglia-Dopamine neurons**
Basal ganglia select actual actions taken

**Amygdala and OFC**
Intuition and Limited data
Fast skilled response based on stored memories using gist and SATISFICING

**Hippocampus-Spatial maps & Task context**

**DLPFC Deliberative Decision**

**Longer-term Memory**

**STM**

**Thalamus**

**LTM**

Dan Levine, Neural Dynamics of affect, gist, probability, and choice, Cognitive Sys Research, 2012
There appear to be multiple reinforcement learning loops in the brain.

Multiple Time scales

theta rhythms 4-10 Hz

Limbic System

Known work by Doya

Reinforcement Learning

Basal ganglia-Dopamine neurons

Cerebral Cortex

DLPFC

ACC

OFC

gamma rhythms 30-100 Hz

Fast skilled response using gist

Work by Dan Levine and others

Motor control 200 Hz

Timing pulses

New actor-critic Reinforcement Learning loop

Cerebellum-direct and inverse models

Brainstem

Spinal cord

Exteroceptive receptors

Interoceptive receptors

Muscle contraction and movement

There appears to be multiple reinforcement learning loops in the brain. Known work by Doya. Fast skilled response using gist. Work by Dan Levine and others. Motor control 200 Hz.
New Fast Decision Structure Using Shunting Inhibition and Multiple Actor-Critic Learning

Shunting Inhibition NN

\[
s_j = \frac{g_h (w_j^T x + w_{j0}) + b_j}{a_j + f_h (c_j^T x + c_{j0})}
\]

Standard Value Function approximation is based only on excitatory NN synapses

\[
\hat{V}_i(x) = \hat{W}_i^T \phi_i(x) = \sum w_{i\ell} \phi_{i\ell}(x)
\]

We use shunting inhibition - it is faster

New Value Function Approximation (VFA) Structures Can Encode Risk, Gist, and Emotional Salience

New Fast Decision Structure Using Adaptive Self-organizing Map and Multiple Actor-Critic Learning

1. Adaptive Self-Organizing Map

Self-organizing map (SOM) neural network is a well-known network for data clustering which has been widely used in many applications such as pattern recognition, biological modeling, signal processing, and data mining. This is known as the Kohonen learning algorithm [32]-[33]. It uses an unsupervised learning approach that divides a set of given input data into groups or clusters. In the SOM, not only are

2. New Actor-Critic ADP structure

Standard NN approximation is based on excitatory NN synapses

\[ \hat{V}_i(x) = \hat{W}_i^T \phi_i(x) = \sum w_{i\ell} \phi_{i\ell}(x) \]

Our New VFA structure is

\[ V(X(k)) = \sum_{j=1}^{J} I_j(k) \sum_{i=1}^{I} (w_{ij} \phi_i(X(k))) \]

Indicator function depends on ASOM Classification

New Value Function Approximation (VFA) Structures
Can Encode Risk, Gist, and Emotional Salience
