Estimating Learning
估計學習理論模型

Joseph Tao-yi Wang (王道一)
Experimetrics Module 7, EE-BGT

Outline: Estimating Learning (Experimetrics, Ch. 18)

1. Directional Learning (DL): Selten and Stoecker (1986)
2. Reinforcement Learning (RL)
3. Belief Learning (BL)
4. EWA Learning: Camerer and Ho (ECMA 1999)
   - Experience-Weighted Attraction – a Hybrid of RL and BL

Directional Learning Theory

- Adjust behavior in response to previous outcome
  - Selten and Stoecker (1986)
  - Finitely Repeated Prisoner's Dilemma (PD)
  - SPE: Always Defect
- Stylized Facts
  - Tacit Cooperation Until Close to End
  - Want to Defect 1st (then Keep Defect)
- Decision: Which Round to Defect

	C	D
C	2, 2	0, 3
D	3, 0	1, 1

Directional Learning Theory

- Play N Supergames with a different opponent each time
  - Adjust next intended deviation period:
- If Deviated First:
  - May gain if deviated later
- If Deviated Later:
  - May gain if deviate early
- If Deviate in the Same Round:
  - May gain if deviate 1 period earlier

	C	D
C	2, 2	0, 3
D	3, 0	1, 1

The Data: Table B1 of Selten and Stoecker (1986)

- n=35 subjects play 25 supergames (of 10-round PD)
  - Play the same opponent within 10 rounds of PD, but
  - Randomly rematch in between: `selten-stoecker.dta`
- Intended Deviation Period of each supergame: `self`
  - `self/other` = 1-10 (period)
  - `self/other` = 11 (later than opponent, but unobserved)
  - `self/other` = 12 (never deviate)
- Deviate `before/same/after` their opponent

Simple Linear Regression

- Predict difference in `self` with `before/same/after`
  - `d.self` - Difference in `self`
  - `l.before` - Lagged `before`
  - `l.same` - Lagged `same`
  - `l.after` - Lagged `after`
- STATA Command:

```
xtset i t
regress d.self l.before l.same l.after, nocon
```

No constant term

Results

```
. xtset i t
       panel variable: i (strongly balanced)
        time variable: t, 1 to 25
                delta: 1 unit

. regress d.self l.before l.same l.after, nocon
      Source |       SS       df       MS              Number of obs = 528
-------------+------------------------------           F( 3, 525) = 29.36
       Model |  93.535929     3   31.178643            Prob > F = 0.0000
    Residual |  557.464071   525  1.06183633           R-squared = 0.1437
-------------+------------------------------           Adj R-squared = 0.1388
       Total |        651   528  1.23295455            Root MSE = 1.0305
                                                       Number of obs = 1000
------------------------------------------------------------------------------
      D.self |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      before |
         L1. |   .3645833   .0743666     4.90   0.000     .2184906    .5106761
             |
        same |
         L1. |  -.1626506   .0799788    -2.03   0.042    -.3197683   -.0055329
             |
       after |
         L1. |  -.6117647   .0790322    -7.74   0.000    -.767023    -.4565064
```

Players deviate 0.36 periods later if before

Players deviate 0.16 periods earlier if same

Players deviate 0.61 periods earlier if after

Pursue-Evade Game (Rosenthal et al. 2003)

- Data: 100 pairs of 50 rounds
  `pursue_evade_sim.dta`
- Payoff Table

	L	R
L	1, -1	0, 0
R	0, 0	2, -2

- Player 1 (Pursuer): **L** (left) or **R** (right)
  - $\mathrm{y1} = 0$ if Pursuer choose **L**; $\mathrm{y1} = 1$ if Pursuer choose **R**
- Player 2 (Evader): **L** (left) or **R** (right)
  - $\mathrm{y2} = 0$ if Evader choose **L**; $\mathrm{y2} = 1$ if Evader choose **R**

Pursue-Evade Game (Rosenthal et al. 2003)

- Two Players: $i = 1, 2$
- Rounds: $t = 1, 2, \ldots, T = 50$
- Two Actions: $s_i^0 = \mathbf{L}$, $s_i^1 = \mathbf{R}$
  - Relabel as
- Actions $j = 0$ (**L**) and $j = 1$ (**R**)
- Strategy of Players i in round t is $s_i(t)$
- Strategy of Players $-i$ in round t is $s_{-i}(t)$
- Players i's Payoff in round t is $\pi_i\left(s_i(t), s_{-i}(t)\right)$

	L	**R**
L	1, -1	0, 0
R	0, 0	2, -2

Learning

- Attraction to action $j = 0, 1$ after round t is $A_i^j(t)$
- Initial Attractions to action $j = 0, 1$ is $A_i^j(0)$
  - Normalize one of initial attractions to 0 for each player
- Choice Probability obtained by logistic transformation

$$P_i^j(t) = \frac{\exp\left[\lambda A_i^j(t-1)\right]}{\exp\left[\lambda A_1^j(t-1)\right] + \exp\left[\lambda A_0^j(t-1)\right]}$$

- Irrelevant ($\lambda = 0$)
- Important (λ large)

  - $i = 1, 2$; $j = 0, 1$; $t = 1, 2, \ldots, T$; $\lambda =$ Sensitivity to attractions

Reinforcement Learning (RL)

- Erev and Roth (1998)
- Update attractions in response to previous payoffs
  - Choices "reinforced" only by previous payoffs

$$\underline{\underline{A_i^j(t)}} = \phi \underline{\underline{A_i^j(t-1)}} + I(s_i(t) = s_i^j)\boxed{\pi_i(s_i^j, s_{-i}(t))}$$

  - $i = 1, 2; j = 0, 1; t = 1, 2,\ldots,T$
- Recency parameter:
  - $\phi = 0$: Only most recent payoff is remembered
  - $\phi = 1$: All past payoffs have equal weight

Reinforcement Learning (RL)

- Normalize Initial Attractions $A_1^1(0) = 0,\ A_2^1(0) = 0$
- Estimate Initial Attractions $A_1^0(0),\ A_2^0(0)$, as well as
- Recency parameter ϕ and Sensitivity parameter λ
  - In STATA using Maximum Likelihood
  - (See code in package)

```
* LIKELIHOOD EVALUATION PROGRAM STARTS HERE
program define reinforcement
* SPECIFY ARGUMENTS: NAMES OF MAXIMAND AND 4 PARAMETERS
args logl phi lam A10_start A20_start
quietly{
```

Max logL with $\phi,\ \lambda, A_1^0(0),\ A_2^0(0)$

```
* INITIALISE ATTRACTION VARIABLES FOR CURRENT LIKELIHOOD EVALUATION:

replace A10=.
replace A11=.
replace A20=.
replace A21=.

* UPDATE ATTRACTIONS BY ADDING PAY-OFFS FROM CHOSEN STRATEGIES
* Aij IS PLAYER i’s ATTRACTION TO STRATEGY j,
* UPDATED BY ACTUAL PAYOFF IN CURRENT PERIOD,
* FIRST GENERATE VALUES OF ATTRACTION VARIABLES IN PERIOD 1
* (USING INITIAL ATTRACTIONS)
* THEN GENERATE VALUES OF ATTRACTION VARIABLES IN SUBSEQUENT PERIODS

by i: replace A10=‘phi’*‘A10_start’+wx10 if _n==1
by i: replace A11=‘phi’*0+wx11 if _n==1
by i: replace A20=‘phi’*‘A20_start’+wx20 if _n==1
by i: replace A21=‘phi’*0+wx21 if _n==1

by i: replace A11=‘phi’*A11[_n-1]+wx11 if A11==.
by i: replace A10=‘phi’*A10[_n-1]+wx10 if A10==.
by i: replace A21=‘phi’*A21[_n-1]+wx21 if A21==.
by i: replace A20=‘phi’*A20[_n-1]+wx20 if A20==.
```

Update Attractions with payoffs of chosen actions

Reinfor...

- No...
- Esti...
- Rec...
  - In...
  - (S...

```
* GENERATE PROBABILITY OF PLAYER i CHOOSING STRATEGY j (pij)
* USING _PREVIOUS_ PERIOD'S ATTRACTIONS

replace p11=.
replace p21=.

by i: replace p11=exp(`lam'*0)/(exp(`lam'*0)+exp(`lam'*`A10_start')) if _n==1
by i: replace p21=exp(`lam'*0)/(exp(`lam'*0)+exp(`lam'*`A20_start')) if _n==1

by i: replace p11=exp(`lam'*A11[_n-1])/(exp(`lam'*A11[_n-1]) ///
+exp(`lam'*A10[_n-1])) if p11==.
by i: replace p21=exp(`lam'*A21[_n-1])/(exp(`lam'*A21[_n-1]) ///
+exp(`lam'*A20[_n-1])) if p21==.

* GENERATE LOG-LIKELIHOOD CONTRIBUTION AS THE PRODUCT OF THE PROBABILITIES
* OF THE CHOICES OF THE TWO PLAYERS

quietly replace `logl'=ln((p11*y1+(1-p11)*(1-y1))*(p21*y2+(1-p21)*(1-y2)))
}
end

* LIKELIHOOD EVALUATION PROGRAM ENDS HERE
```

Compute choice probabilities with Attractions

Compute log-Likelihood

```
* READ DATA

use "pursue_evade_sim.dta", clear

* GENERATE AMOUNT EACH PLAYER _WOULD_ RECEIVE BY PLAYING EACH STRATEGY,
* _GIVEN_ THE STRATEGY CHOSEN BY THE OTHER PLAYER
* x_ij IS PAYOFF PLAYER i (i=1,2) WOULD RECEIVE BY PLAYING STRATEGY j
* j=0,1; 0=LEFT; 1=RIGHT).

gen int x11= 2*(y2==1)+0*(y2==0)
gen int x10= 0*(y2==1)+1*(y2==0)

gen int x21= (-2)*(y1==1)+0*(y1==0)
gen int x20= 0*(y1==1)+(-1)*(y1==0)

* GENERATE AMOUNT EACH PLAYER RECIEVES BY PLAYING THE STRATEGY THEY CHOOSE;
* ZERO FOR THE UNCHOSEN STRATEGY
* wx_ij IS AMOUNT RECEIEVED BY i CHOOSING j. wx_ij = 0 IF j NOT CHOSEN.

gen int wx11= (y1)*x11
gen int wx10= (1-y1)*x10

gen int wx21= (y2)*x21
gen int wx20= (1-y2)*x20
```

Read simulated data

Compute payoffs of possible action

Compute weighted payoffs with actual realization

Reinfor

- No
- Esti
- Rec
  - In
  - (Se

Reinfor 98)

- No
- Esti as
- Rec
  - In
  - (Se

```
* INITIALISE ATTRACTION VARIABLES, AND CHOICE PROBABILITY VARIABLES

gen double A10=.
gen double A11=.
gen double A20=.
gen double A21=.

gen double p11=.
gen double p21=.

* SET STARTING VALUES:

mat start=( 0.95,0.20,0.0,0.0)

*RUN ML

ml model lf reinforcement /phi /lam /A10_start /A20_start
ml init start, copy

ml max, trace search(norescale)
```

Set initial values

Maximum Likelihood Estimation

Results of Reinforcement Learning (RL)

Number of obs = 5000
Wald chi2(0) = .
Log likelihood = -6863.0929
Prob > chi2 = .

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
phi						
_cons	.7676348	.0845556	9.08	0.000	.601909	.9333607
lam						
_cons	.1095563	.0156976	6.98	0.000	.0787895	.1403231
A10_start						
_cons	1.389702	1.457448	0.95	0.340	-1.466843	4.246247
A20_start						
_cons	-8.310659	2.286308	-3.63	0.000	-12.79174	-3.829578

Recency ϕ far from 1: Forget past quickly, significant

Sensitivity λ > 0: Attractions matter, significant

$A_1^0(0) > 0$: Pursuers start at **L**, not significant

$A_2^0(0) < 0$: Evaders start at **R**, significant

Simple Belief Learning (BL): Cournot Learning

- Cournot Learning: Attractions increase by action-corresponding payoffs given opponent actions
  - BR to opponent action in previous round

$$\underline{\underline{A_i^j(t)}} = \underline{\underline{A_i^j(t-1)}} + \boxed{\pi_i\left(s_i^j, s_{-i}(t)\right)}$$

  - $i = 1,\ 2;\ j = 0,\ 1;\ t = 1,\ 2,\ldots,T$
  - Normalize Initial Attractions $A_1^1(0) = 0,\ A_2^1(0) = 0$
  - Only need to estimate Initial Attractions $A_1^0(0),\ A_2^0(0)$ and λ using Maximum Likelihood (Too simple?!)

Belief Learning (BL): Standard Fictitious Play

- Standard Fictitious Play: Attractions is action-corresponding average payoffs
  - Counting cards and BR to opponent actions from all rounds
- All Initial Attractions are zero: $A_i^j(0) = 0$, $j = 0, 1$

$$A_i^j(1) = \pi_i\left(s_i^j, s_{-i}(1)\right), A_i^j(2) = \frac{1}{2}\left[\pi_i\left(s_i^j, s_{-i}(1)\right) + \pi_i\left(s_i^j, s_{-i}(2)\right)\right]$$

$$A_i^j(3) = \frac{1}{3}\left[\pi_i\left(s_i^j, s_{-i}(1)\right) + \pi_i\left(s_i^j, s_{-i}(2)\right) + \pi_i\left(s_i^j, s_{-i}(3)\right)\right]$$

- ..., $A_i^j(t) = \frac{1}{t}\sum_{\tau=1}^{t} \pi_i\left(s_i^j, s_{-i}(\tau)\right)$

Belief Learning (BL): Experience Weight

- Express Attractions based on Experience $N(t)$
  - Observation Equivalents: Experience accumulated up to t
- Initial Experience is zero: $N(0) = 0$
- Iteratively define $N(t) = N(t-1) + 1,\ t = 1, \ldots, T$
- All Initial Attractions are zero: $A_i^j(0) = 0\ ,\ j = 0,\ 1$
- Iteratively define (for $j = 0,\ 1;\ t = 1, \ldots, T$)

$$A_i^j(t) = \frac{1}{N(t)} \left[N(t-1) A_i^j(t-1) + \pi_i\left(s_i^j, s_{-i}(t)\right) \right]$$

  - Special Case of $N(t) = t$ is Standard Fictitious Play!

Belief Learning (BL): Weighted Fictitious Play

- Another Special Case is Weighted Fictitious Play
  - With Recency parameter ϕ
- Initial Experience is zero: $N(0) = 0$
- Iteratively define $N(t) = \phi N(t-1) + 1,\ t = 1, \cdots, T$
- All Initial Attractions are zero: $A_i^j(0) = 0$, $j = 0,\ 1$
- Iteratively define (for $j = 0,\ 1;\ t = 1,\ldots, T$)

$$A_i^j(t) = \frac{1}{N(t)} \left[\phi N(t-1) A_i^j(t-1) + \pi_i(s_i^j, s_{-i}(t)) \right]$$

  - Weights are $1,\ \phi,\ \phi^2,\ \phi^3$, etc.

Belief Learning (BL): Weighted Fictitious Play

- Weighted Fictitious Play: Attractions is action-corresponding average payoffs weighted by recency (exponentially discounted)
- All Initial Attractions are zero: $A_i^j(0) = 0$, $j = 0, 1$

$$A_i^j(1) = \pi_i(s_i^j, s_{-i}(1)),$$

$$A_i^j(2) = \frac{1}{\phi + 1}\left[\phi\pi_i(s_i^j, s_{-i}(1)) + \pi_i(s_i^j, s_{-i}(2))\right]$$

$$A_i^j(3) = \frac{\phi^2\pi_i(s_i^j, s_{-i}(1)) + \phi\pi_i(s_i^j, s_{-i}(2)) + \pi_i(s_i^j, s_{-i}(3))}{\phi^2 + \phi + 1}, \text{ etc.}$$

Belief Learning (BL): Weighted Fictitious Play

- In general, initial attractions and $N(0)$ need not be zero
  - Normalize Initial Attractions $A_1^1(0) = 0$, $A_2^1(0) = 0$
  - And:
- Estimate Initial Attractions $A_1^0(0)$, $A_2^0(0)$, $N(0)$ as well as Recency parameter ϕ and Sensitivity parameter λ
  - In STATA using Maximum Likelihood (See code in package)
- Standard Fictitious Play if $\phi = 1$
- Cournot Learning if $\phi = 0$

Belief Learnin

- In general, ... o
  - Normalize
  - And:
- Estimate In
  as Recency
  - In STATA
- Standard F
- Cournot Le

```
* LIKELIHOOD EVALUATION PROGRAM STARTS HERE

program define belief
args logl phi lam A10_start A20_start N_start

quietly{

replace A10=.
replace A11=.
replace A20=.
replace A21=.

replace N=.

by i: replace N=`phi'*`N_start'+1 if _n==1
by i: replace N=`phi'*N[_n-1]+1 if N==.

by i: replace A10=(`phi'*`N_start'*`A10_start'+x10)/N if _n==1
by i: replace A11=(`phi'*`N_start'*0+x11)/N if _n==1
by i: replace A20=(`phi'*`N_start'*`A20_start'+x20)/N if _n==1
by i: replace A21=(`phi'*`N_start'*0+x21)/N if _n==1
```

Max logL with ϕ, $\lambda, A_1^0(0)$, $A_2^0(0)$, $N(0)$

Compute Experience $N(t)$

Compute initial Attractions with Experience and payoffs of chosen actions

```
* Aij is the attraction, updated by payoff (either actual or hypothetical) in t,
* to be used to determine choice probs in t+1

by i: replace A11=('phi'*N[_n-1]*A11[_n-1]+x11)/N if A11==.
by i: replace A10=('phi'*N[_n-1]*A10[_n-1]+x10)/N if A10==.
by i: replace A21=('phi'*N[_n-1]*A21[_n-1]+x21)/N if A21==.
by i: replace A20=('phi'*N[_n-1]*A20[_n-1]+x20)/N if A20==.

*pij are the probabilities player i choosing strategy j

replace p11=.
replace p21=.

by i: replace p11=exp('lam'*0)/(exp('lam'*0)+exp('lam'*'A10_start')) if _n==1
by i: replace p21=exp('lam'*0)/(exp('lam'*0)+exp('lam'*'A20_start')) if _n==1

by i: replace p11=exp('lam'*A11[_n-1])/(exp('lam'*A11[_n-1]) ///
+exp('lam'*A10[_n-1])) if p11==.
by i: replace p21=exp('lam'*A21[_n-1])/(exp('lam'*A21[_n-1]) ///
+exp('lam'*A20[_n-1])) if p21==.

replace 'logl'=ln((p11*y1+(1-p11)*(1-y1))*(p21*y2+(1-p21)*(1-y2)))
}
```

Update Attractions with Experience and payoffs of chosen actions

Compute choice probabilities from Attractions

Compute log-Likelihood

```
end

* LIKELIHOOD EVALUATION PROGRAM ENDS HERE

* READ DATA

use "pursue_evade_sim.dta", clear

* GENERATE AMOUNT EACH PLAYER _WOULD_ RECEIVE BY PLAYING EACH STRATEGY,
* _GIVEN_ THE STRATEGY CHOSEN BY THE OTHER PLAYER
* x_ij IS PAYOFF PLAYER i (i=1,2) RECEIVES BY PLAYING STRATEGY j
* (j=0,1; 0=LEFT; 1=RIGHT).

gen int x11= 2*(y2==1)+0*(y2==0)
gen int x10= 0*(y2==1)+1*(y2==0)
gen int x21= (-2)*(y1==1)+0*(y1==0)
gen int x20= 0*(y1==1)+(-1)*(y1==0)

* INITIALISE OTHER VARIABLES

gen double A10=.
gen double A11=.
gen double A20=.
gen double A21=.
```

Read simulated data

Compute payoffs of possible action given opponent strategy

Belief Learning (BL): Weighted Fictitious Play

- In ge… be zero
  - No…
  - An…
- Esti… as well as R… …ter λ
  - In S… …package)
- Stan…
- Cou…

```
gen double N=.

gen double wx11=.
gen double wx10=.
gen double wx21=.
gen double wx20=.

gen double p11=.
gen double p21=.

* STARTING VALUES:

mat start=( 0.95,0.20,0.0,0.0,1.0)

*RUNNING ML

ml model lf belief /phi /lambda /A10_start /A20_start /N_start
ml init start, copy
ml max, trace search(norescale)
```

Set initial values

Maximum Likelihood Estimation

Results of Belief Learning (Weighted Fictitious Play)

Log likelihood = -6808.3011 Prob > chi2 = .

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
phi						
_cons	.9531451	.0188062	50.68	0.000	.9162856	.9900046
lambda						
_cons	.424634	.0355367	11.95	0.000	.3549833	.4942846
A10_start						
_cons	1.684257	.4686631	3.59	0.000	.7656939	2.602819
A20_start						
_cons	-4.255275	.7702032	-5.52	0.000	-5.764845	-2.745704
N_start						
_cons	.4723948	.1498293	3.15	0.002	.1787348	.7660549

Recency ϕ away from 0 and 1: Neither Cournot nor standard fictitious play

Sensitivity $\lambda > 0$: Attractions matter, significant

$A_1^0(0) > 0$: Pursuers start at **L**, significant

$A_2^0(0) < 0$: Evaders start at **R**, significant

Start with 0.5 observation equivalents of experience

Experience-Weighted Attraction (EWA) Learning Model

- EWA = Experience-Weighted + Attraction
- Has both Experience $N(t)$ and Attractions $A_i^j(t)$

- Experience $N(t)$ accumulated as Observation Equivalents
- Initial Experience estimated: $N(0)$
- Iteratively define $N(t) = \rho N(t-1) + 1,\ t = 1, \cdots, T$
  - Past Experience Depreciation Rate is $\rho < 1$

Experience-Weighted Attraction (EWA) Learning Model

- Initial Attractions estimated: $A_i^j(0)$, $j = 0, 1$
- Attractions to different actions iteratively define

$$A_i^j(t) = \frac{\phi N(t-1)A_i^j(t-1) + \left[\delta + (1-\delta)I_{s_i(t)=s_i^j}\right]\pi_i(s_i^j, s_{-i}(t))}{N(t)}$$

$$= \frac{\phi N(t-1)A_i^j(t-1) + 1 \cdot \pi_i(s_i^j, s_{-i}(t))}{N(t)} \quad \text{if } s_i^j \text{ chosen}$$

$$= \frac{\phi N(t-1)A_i^j(t-1) + \delta \cdot \pi_i(s_i^j, s_{-i}(t))}{N(t)} \quad \text{if not}$$

- RL ($\delta = 0$) vs. BL ($\delta = 1$) (for $j = 0, 1$; $t = 1, \ldots, T$)

Experience-Weighted Attraction (EWA) Learning Model

- Choice Probability obtained by logistic transformation

$$P_i^j(t) = \frac{\exp\left[\lambda A_i^j(t-1)\right]}{\exp\left[\lambda A_1^j(t-1)\right] + \exp\left[\lambda A_0^j(t-1)\right]}$$

  - $i = 1,\ 2;\ j = 0,\ 1;\ t = 1,\ 2, \ldots, \mathrm{T}$
- $\lambda =$ Sensitivity to attractions
  - Irrelevant ($\lambda = 0$)
  - Important (λ large)

Experience-Weighted Attraction (EWA) Learning Model

- Experience $N(t)$-weighted Attractions $A_i^j(t)$ model generates choice probabilities $P_i^j(t)$
- Estimate 7 parameters: $\rho, \delta, \phi, \lambda, A_1^1(0), A_2^1(0), N(0)$
- δ distinguishes RL: ($\delta = 0$) from BL ($\delta = 1$):

1. BL: $\delta = 1; \rho = \phi$
2. RL: $\delta = 0; \boxed{N_0 = 1}; \boxed{\rho = 0}$
   - Note that $A_1^1(0), A_2^1(0)$ not identified if $N_0 = 0$
   - Also, RL does not have depreciation ρ

Simulating EWA for 100 Subject Pairs, 50 Rounds Each

- Simulate EWA model with: $\rho = 0.97, \delta = 0.60, \phi = 0.94,$
  $$\lambda = 0.80, A_1^0(0) = 1.0, A_2^0(0) = -2.0, N(0) = 1.0$$
- Simulate round 1 choices and resulting attractions and loop over round 2-50 with `forvalues` to compute:
  1. Choice probability `p11`, `p21` from previous attractions
  2. Actual choices from probabilities
  3. Payoffs for each possible action
  4. Payoffs weighted by actual realizations
  5. Attractions (for next round's choice probability)

Simul... nds Each

- S
- Sir... ns and loc...

1.
2.
3.
4.
5.

```
/* SIMULATION OF EWA MODEL
n=100 subject pairs, T=50 rounds.
PLAYER 1 = PURSUER; PLAYER 2= EVADER
0 = LEFT; 1 = RIGHT
*/

clear
drop _all
set obs 5000
set seed 56734512
set more off

* SET TRUE PARAMETER VALUES:

scalar rho=0.97
scalar delta=0.60
scalar phi=0.94
scalar lam=0.80
scalar A10_start=1.0
scalar A20_start=-2.0
scalar N_start=1.0

* GENERATE PAIR NUMBER (i), PERIOD NUMBER (t), AND DECLARE PANEL:
```

```
* GENERATE PAIR NUMBER (i), PERIOD NUMBER (t), AND DECLARE PANEL:

egen int i=seq(), f(1) b(50)
egen int t=seq(), f(1) t(50)
tsset i t
```

50 pairs
50 periods

```
* GENERATE TWO RANDOM UNIFORMS FOR LATER USE:

gen double u1=runiform()
gen double u2=runiform()
```

```
* GENERATE PERIOD 1 PROBABILITIES:

by i: generate double p11=exp(lam*0)/(exp(lam*0)+exp(lam*A10_start)) if _n==1
by i: generate double p21=exp(lam*0)/(exp(lam*0)+exp(lam*A20_start)) if _n==1

* GENERATE PERIOD-1 CHOICES OF PLAYERS 1 AND 2 (USING RANDOM UNIFORMS):

by i: gen int y1=u1<p11 if _n==1
by i: gen int y2=u2<p21 if _n==1

* GENERATE PERIOD-1 PAY-OFFS:

by i: generate double x11= 2*(y2==1)+0*(y2==0) if _n==1
by i: generate double x10= 0*(y2==1)+1*(y2==0) if _n==1
by i: generate double x21= (-2)*(y1==1)+0*(y1==0) if _n==1
by i: generate double x20= 0*(y1==1)+(-1)*(y1==0) if _n==1

* GENERATE PERIOD-1 WEIGHTED PAY-OFF

by i: generate double wx11= (delta+(1-delta)*(y1==1))*x11 if _n==1
by i: generate double wx10= (delta+(1-delta)*(y1==0))*x10 if _n==1
by i: generate double wx21= (delta+(1-delta)*(y2==1))*x21 if _n==1
by i: generate double wx20= (delta+(1-delta)*(y2==0))*x20 if _n==1
```

Period 1 probabilities

Period 1 choices

Period 1 payoffs

Period 1 weighted payoffs

Simul… …ach

```
* GENERATE PERIOD-1 WEIGHTED PAY-OFFS:

by i: generate double wx11= (delta+(1-delta)*(y1==1))*x11 if _n==1
by i: generate double wx10= (delta+(1-delta)*(y1==0))*x10 if _n==1
by i: generate double wx21= (delta+(1-delta)*(y2==1))*x21 if _n==1
by i: generate double wx20= (delta+(1-delta)*(y2==0))*x20 if _n==1

* GENERATE EXPERIENCE VARIABLE, N(t), STARTING WITH PERIOD 1:

by i: generate double N=rho*N_start+1 if _n==1
by i: replace N=rho*N[_n-1]+1 if N==.

* GENERATE PERIOD-1 ATTRACTIONS:

by i: generate A11=(phi*N_start*0+wx11)/N if _n==1
by i: generate A10=(phi*N_start*A10_start+wx10)/N if _n==1
by i: generate A21=(phi*N_start*0+wx21)/N if _n==1
by i: generate A20=(phi*N_start*A20_start+wx20)/N if _n==1

quietly{
```

Period 1 weighted payoffs

Period 1 experience

Period 1 attractions

Simul

- S
- Si loc

1.
2.
3.
4.
5.

```
quietly{

* LOOP OVER PERIODS STARTS HERE

forvalues t = 2(1)50 {

* GENERATE p11 AND p21 (PROBABILITIES OF PLAYERS 1 and 2 CHOOSING STRATEGY 1):

by i: replace p11=exp(lam*A11[_n-1])/(exp(lam*A11[_n-1])+exp(lam*A10[_n-1])) ///
if (_n==`t')
by i: replace p21=exp(lam*A21[_n-1])/(exp(lam*A21[_n-1])+exp(lam*A20[_n-1])) ///
if (_n==`t')

* GENERATE y1 AND y2 (CHOICES OF PLAYERS 1 AND 2) USING RANDOM UNIFORMS:

by i: replace y1=0 if (_n==`t')
by i: replace y1= (u1<p11) if (_n==`t')

by i: replace y2=0 if (_n==`t')
by i: replace y2= (u2<p21) if (_n==`t')
```

Loop over t = 2-50

Period t probabilities

Period t choices

```
* GENERATE xij (PAY-OFF PLAYER i WOULD HAVE RECEIVED WITH STRATEGY j):

by i: replace x11= 2*(y2==1)+0*(y2==0) if (_n==‘t’)
by i: replace x10= 0*(y2==1)+1*(y2==0) if (_n==‘t’)
by i: replace x21= (-2)*(y1==1)+0*(y1==0) if (_n==‘t’)
by i: replace x20= 0*(y1==1)+(-1)*(y1==0) if (_n==‘t’)

* GENERATE wxij (PAY-OFFS WEIGHTED BY DELTA PARAMETER):

by i: replace wx11= (delta+(1-delta)*(y1==1))*x11 if (_n==‘t’)
by i: replace wx10= (delta+(1-delta)*(y1==0))*x10 if (_n==‘t’)
by i: replace wx21= (delta+(1-delta)*(y2==1))*x21 if (_n==‘t’)
by i: replace wx20= (delta+(1-delta)*(y2==0))*x20 if (_n==‘t’)

* GENERATE Aij (ATTRACTION, UPDATED BY PAY-OFFS IN t, TO BE USED
* TO DETERMINE CHOICE PROBABILITIES in t+1)

by i: replace A11=(phi*N[_n-1]*A11[_n-1]+wx11)/N if (_n==‘t’)
by i: replace A10=(phi*N[_n-1]*A10[_n-1]+wx10)/N if (_n==‘t’)
by i: replace A21=(phi*N[_n-1]*A21[_n-1]+wx21)/N if (_n==‘t’)
by i: replace A20=(phi*N[_n-1]*A20[_n-1]+wx20)/N if (_n==‘t’)
}
* END OF LOOP
}
```

Period t payoffs

Period t weighted payoffs

Period t attractions

```
* DISCARD SUPERFLUOUS VARIABLES:

keep i t y1 y2
```

Estimating the Full EWA Model to Recover Parameters

- Estimate the Full EWA Model, the Restricted Models of RL and BL, and:
  - Conduct LR Tests to see if EWA performs better
  - Compare uncovered parameters with: $\rho = 0.97, \delta = 0.60, \phi = 0.94, \lambda = 0.80, A_1^0(0) = 1.0, A_2^0(0) = -2.0, N(0) = 1.0$
  - STATA Code

Estim… Parameters

- Es… Models of RL…
  - C…
  - C… $= 0.60$,

    ϕ … $(0) = 1.0$
  - S…

```
* LIKELIHOOD EVALUATION PROGRAM STARTS HERE

program drop _all

program define ewa
args lnf rho delta phi lambda A10_start A20_start N_start

tempvar
tempname

quietly{

replace A10=.
replace A11=.
replace A20=.
replace A21=.

* GENERATE EXPERIENCE VARIABLE, N(t), STARTING WITH PERIOD 1:

replace N=.
by i: replace N=‘rho’*‘N_start’+1 if _n==1
by i: replace N=‘rho’*N[_n-1]+1 if N==.

* GENERATE wxij (PAY-OFFS WEIGHTED BY DELTA PARAMETER):
```

Generate Experience $N(\mathrm{t})$

```
* GENERATE wxij (PAY-OFFS WEIGHTED BY DELTA PARAMETER):

replace wx11= (`delta'+(1-`delta')*(y1))*x11
replace wx10= (`delta'+(1-`delta')*(1-y1))*x10
replace wx21= (`delta'+(1-`delta')*(y2))*x21
replace wx20= (`delta'+(1-`delta')*(1-y2))*x20

* GENERATE PERIOD-1 ATTRACTIONS:

by i: replace A10=(`phi'*`N_start'*`A10_start'+wx10)/N if _n==1
by i: replace A11=(`phi'*`N_start'*0+wx11)/N if _n==1
by i: replace A20=(`phi'*`N_start'*`A20_start'+wx20)/N if _n==1
by i: replace A21=(`phi'*`N_start'*0+wx21)/N if _n==1

* GENERATE ATTRACTIONS FOR t>1:

by i: replace A11=(`phi'*N[_n-1]*A11[_n-1]+wx11)/N if A11==.
by i: replace A10=(`phi'*N[_n-1]*A10[_n-1]+wx10)/N if A10==.
by i: replace A21=(`phi'*N[_n-1]*A21[_n-1]+wx21)/N if A21==.
by i: replace A20=(`phi'*N[_n-1]*A20[_n-1]+wx20)/N if A20==.

* GENERATE p11 AND p21 (PROBABILITIES OF PLAYERS 1 and 2 CHOOSING STRATEGY 1):

replace p11=.
```

Generate Weighted Payoffs

Generate Attractions

Generate Choice Probabilities

Estim

Es

RL

ϕ

```
replace p11=.
replace p21=.

by i: replace p11=exp(`lambda'*0)/(exp(`lambda'*0) ///
+exp(`lambda'*`A10_start')) if _n==1
by i: replace p21=exp(`lambda'*0)/(exp(`lambda'*0) ///
+exp(`lambda'*`A20_start')) if _n==1

by i: replace p11=exp(`lambda'*A11[_n-1])/(exp(`lambda'*A11[_n-1]) ///
+exp(`lambda'*A10[_n-1])) if p11==.
by i: replace p21=exp(`lambda'*A21[_n-1])/(exp(`lambda'*A21[_n-1]) ///
+exp(`lambda'*A20[_n-1])) if p21==.

* GENERATE LOG-LIKELIHOOD

replace `lnf'=ln((p11*y1+(1-p11)*(1-y1))*(p21*y2+(1-p21)*(1-y2)))
}
end

* LIKELIHOOD EVALUATION PROGRAM ENDS HERE

* READ DATA

use "pursue_evade_sim.dta", clear
```

Generate Choice Probabilities

Generate log-Likelihood

```
* GENERATE PAY-OFFS (xij) THAT PLAYER i WOULD RECIEVE FROM CHOOSING STRATEGY j.

gen int x11= 2*(y2==1)+0*(y2==0)
gen int x10= 0*(y2==1)+1*(y2==0)

gen int x21= (-2)*(y1==1)+0*(y1==0)
gen int x20= 0*(y1==1)+(-1)*(y1==0)
```

Period 1 payoffs

```
* INITIALISE VARIOUS VARIABLES USED WITHIN PROGRAM:

gen double A10=.
gen double A11=.
gen double A20=.
gen double A21=.

gen double N=.

gen double wx11=.
gen double wx10=.
gen double wx21=.
gen double wx20=.

gen double p11=.
gen double p21=.
```

Set initial values

```
* SET STARTING VALUES:

mat start=( 0.993, 0.78,0.998,0.7531,0.657,-1.863,0.833)

*RUNNING ML: FULL EWA MODEL
```

Maximum Likelihood Estimation

```
ml model lf ewa /rho /delta /phi /lambda /A10_start /A20_start /N_start
ml init start, copy
ml max, trace search(norescale)
est store ewa
```

eters

s of

```
* DEFINE CONSTRAINTS REQUIRED FOR RL AND BL:

constraint 1 [delta]_b[_cons]=0.0
constraint 2 [rho]_b[_cons]=0.0
constraint 3 [delta]_b[_cons]=1
constraint 4 [rho]_b[_cons]=[phi]_b[_cons]
constraint 5 [N_start]_b[_cons]=1
```

Constraints of RL and BL

```
* ESTIMATE RL AS RESTRICTED VERSION OF EWA:

ml model lf ewa /rho /delta /phi /lambda /A10_start /A20_start /N_start, constraints(1 2 5)
ml init start, copy
ml max, trace search(norescale)
est store rl
```

Estimate RL

```
* ESTIMATE BL AS RESTRICTED VERSION OF EWA:

ml model lf ewa /rho /delta /phi /lambda /A10_start /A20_start /N_start, constraints(3 4)
ml init start, copy
ml max, trace search(norescale)
est store bl
```

Estimate BL

```
* LR TESTS FOR RL AND BL AS RESTRICTIONS OF EWA:

lrtest ewa rl
lrtest ewa bl
```

LR tests

Results of Experienced-Weighted Attractions (Full EWA)

```
                                          Number of obs = 5000
                                          Wald chi2(0) = .
Log likelihood = -6800.9162               Prob > chi2 = .

------------------------------------------------------------------------------
             |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
rho          |
       _cons |   .9834178   .0191123    51.45   0.000     .9459585    1.020877
-------------+----------------------------------------------------------------
delta        |
       _cons |   .5408416   .1124522     4.81   0.000     .3204393    .7612439
-------------+----------------------------------------------------------------
phi          |
       _cons |   .9427034   .0214316    43.99   0.000     .9006982    .9847085
-------------+----------------------------------------------------------------
lambda       |
       _cons |   .8208911   .1502486     5.46   0.000     .5264092    1.115373
-------------+----------------------------------------------------------------
A10_start    |
```

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
rho						
_cons	.9834178	.0191123	51.45	0.000	.9459585	1.020877
delta						
_cons	.5408416	.1124522	4.81	0.000	.3204393	.7612439
phi						
_cons	.9427034	.0214316	43.99	0.000	.9006982	.9847085
lambda						
_cons	.8208911	.1502486	5.46	0.000	.5264092	1.115373
A10_start						
_cons	.9261403	.2732887	3.39	0.001	.3905043	1.461776
A20_start						
_cons	-2.108425	.5609236	-3.76	0.000	-3.207815	-1.009035
N_start						
_cons	.8843406	.3065987	2.88	0.004	.2834182	1.485263

$\rho = 0.97,$

$\delta = 0.60,$

$\phi = 0.94,$

$\lambda = 0.80,$

$A_1^0(0) = 1.0,$

$A_2^0(0) = -2.0,$

$N(0) = 1.0$

All CIs contain true value!

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
rho						
_cons	.9834178	.0191123	51.45	0.000	.9459585	
delta						
_cons	.5408416	.1124522	4.81	0.000	.3204393	.7612439
phi						
_cons	.9427034					
lambda						
_cons	.8208911	.1502486	5.46	0.000	.5264092	1.115373
A10_start						
_cons	.9261403	.2732887	3.39	0.001	.3905043	1.461776
A20_start						
_cons	-2.108425	.5609236	-3.76	0.000	-3.207815	-1.009035
N_start						
_cons	.8843406	.3065987	2.88	0.004	.2834182	1.485263

$\rho =$

CI is reasonably precise given obs

Weight on Foregone Payoffs

$\delta = 0.60,$

Both actual (RL) and foregone (BL) payoffs important

δ=0.54: Foregone payoffs slightly more important

$\lambda = 0.80,$

$A_1^0(0) = 1.0,$

$A_2^0(0) = -2.0,$

$N(0) = 1.0$

All CIs contain true value!

Results of Reinforcement Learning (RL)

Log likelihood = -6863.0929

Number of obs = 5000
Wald chi2(0) = .
Prob > chi2 = .

	Coef.	Std. Err.	z	P>|z|	[95% Conf. Interval]	
phi						
_cons	$\boxed{.7676348}$	.0845556	9.08	0.000	.601909	.9333607
lam						
_cons	$\boxed{.1095563}$	.0156976	6.98	0.000	.0787895	.1403231
A10_start						
_cons	$\boxed{1.389702}$	1.457448	0.95	0.340	-1.466843	4.246247
A20_start						
_cons	$\boxed{-8.310659}$	2.286308	-3.63	0.000	-12.79174	-3.829578

Recency ϕ far from 1: Forget past quickly, significant

Sensitivity λ > 0: Attractions matter, significant

$A_1^0(0) > 0$: Pursuers start at **L**, not significant

$A_2^0(0) < 0$: Evaders start at **R**, significant

Results of Belief Learning (Weighted Fictitious Play)

Log likelihood = -6808.3011 Prob > chi2 = .

Recency ϕ away from 0 and 1: Neither Cournot nor standard fictitious play

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
phi						
_cons	.9531451	.0188062	50.68	0.000	.9162856	.9900046
lambda						
_cons	.424634	.0355367	11.95	0.000	.3549833	.4942846
A10_start						
_cons	1.684257	.4686631	3.59	0.000	.7656939	2.602819
A20_start						
_cons	-4.255275	.7702032	-5.52	0.000	-5.764845	-2.745704
N_start						
_cons	.4723948	.1498293	3.15	0.002	.1787348	.7660549

Sensitivity $\lambda > 0$: Attractions matter, significant

$A_1^0(0) > 0$: Pursuers start at **L**, significant

$A_2^0(0) < 0$: Evaders start at **R**, significant

Start with 0.5 observation equivalents of experience

LR Test Results

Model	Log-L	LR	df	p-value
EWA	-6800.92			
RL	-6863.09	124.34	3	0.0000
BL	-6808.30	14.76	2	0.0006

- Both RL and BL strongly rejected by LR test
  - Not surprising since we simulated EWA data with δ=0.6
- In between RL and BL (but slightly closer to BL)

Full EWA

	Coef.	Std. Err.		P>|z|	[95% Conf. Interval]
rho $\rho = 0.97,$	.9834178				
delta $\delta = 0.60,$	.5408416				
phi $\phi = 0.94,$	.9427034				
lambda $\lambda = 0.80,$	.8208911				
A10_start $A_1^0(0) = 1,$	.9261403				
A20_start $A_2^0(0) = -2,$	-2.108425				
N_start $N(0) = 1.0$	.8843406	.3065987	2.88	0.004	

RL

Log likelihood = -6863.0929

Initial attractions exaggerated

Underestimate recency

phi $\phi = 0.94,$	.7676348
lam $\lambda = 0.80,$	.1095563
A10_start $A_1^0(0) = 1,$	1.389702
A20_start $A_2^0(0) = -2,$	-8.310659

BL

Log likelihood = -6808.3011

Sensitivity to attractions underestimated

phi $\phi = 0.94,$	.9531451	.01880
lambda $\lambda = 0.80,$	.424634	.03553
A10_start $A_1^0(0) = 1,$	1.684257	.46866
A20_start $A_2^0(0) = -2,$	-4.255275	.77020
N_start $N(0) = 1.0$	.4723948	.14982

Acknowledgment

- This presentation is based on
  - Section 18.1-18.5 of the textbook on Experimetrics,
- An extension of the mini-course taught by Peter G. Moffatt (UEA) at National Taiwan University in Spring 2019