Learning to predict Nash equilibria from data using fixed point networks

SIAM OP21

Howard Heaton[1], Daniel McKenzie[1], Qiuwei Li[1], Samy Wu Fung[1], Stanley Osher[1] and Wotao Yin[2]

[1]University of California, Los Angeles

[2]Alibaba DAMO Academy

July 20, 2021

Overview

- Study games depending on external parameter d.
- Wish to predict outcome of game knowing only d.
- Reduce game to variational inequality then to fixed point problem.
- Train neural network to solve FP problem.

Papers and Code

- Wu Fung, Heaton, Li, McKenzie, Osher & Yin:
  FPN's: Implicit Depth Models with Jacobian-Free Backprop.
- `https://github.com/howardheaton/nash_fixed_point_networks`
- Heaton, McKenzie, Li, Wu Fung, Osher & Yin:
  Learn to Predict Equilibria via Fixed Point Networks.
- `https://github.com/howardheaton/fixed_point_networks`

Outline

Contextual Games

Naive N-FPN

Scaling to large games

Contextual Traffic Routing

Contextual Games

- Consider game with K interacting agents.
- Contextual[1]: d represents factors beyond agents' control.
- Agent k chooses x_k. Incurs cost $u_k(x_k, x_{-k}; d)$.
- All agents self-interested; seek to minimize u_k.

Nash Equilibrium (NE)

$x_d^\star = \left[x_{d,1}^\star \cdots x_{d,K}^\star\right]$ is a NE if no agent can decrease their cost by unilaterally deviating.

[1] *Contextual Games: Multi-Agent Learning with Side Information* Sessa *et al* (2020)

Example: Contextual Rock, Paper, Scissors

- $K = 2$. Available actions $= \{R, P, S\}$ for $k = 1, 2$.
- Payoff *parametrized* by d:

$$u_1(x_1) = x_1^\top B(d) x_2 \quad \text{and} \quad u_2(x_2) = -x_1^\top B(d) x_2 \tag{1}$$

- **Mixed strategies:** $x_k \in \Delta^3 = \{x \in \mathbb{R}^3 : x \geq 0 \,, \sum x[i] = 1\}$

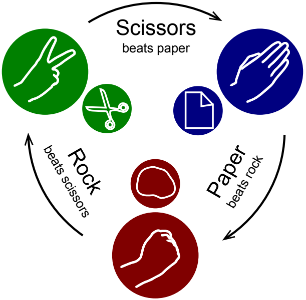

Figure: By Enzoklop - Own work, CC BY-SA 3.0,

https://commons.wikimedia.org/w/index.php?curid=27958688

	R	P	S
R	0	-1	1
P	1	0	-1
S	-1	1	0

	R	P	S
R	0	$-\langle w^1, d\rangle$	$\langle w^2, d\rangle$
P	$\langle w^1, d\rangle$	0	$-\langle w^3, d\rangle$
S	$-\langle w^2, d\rangle$	$\langle w^3, d\rangle$	0

Figure: Payoff/cost matrices for classical RPS (top) and contextual RPS (bottom)

Technical Slide: Assumptions and Notation

- Assume $u_k(\cdot, x_{-k}; d)$ convex and smooth for all x_{-k}, d.
- **Action set:** $x_k \in \mathcal{V}_k \subset \mathcal{X}_k$ where:
  - $\mathcal{X}_k$ Hilbert space.
  - $\mathcal{V}_k$ compact and convex.
- $\mathcal{C} \triangleq \mathcal{V}_1 \times \cdots \times \mathcal{V}_k$,
- Reserve F for **game gradient**:

$$F(x; d) \triangleq \left[\nabla_{x_1} u_1(x; d)^\top, \ldots, \nabla_{x_K} u_K(x; d)^\top\right]^\top. \qquad (2)$$

- Assume $F(\cdot\ ; d)$ is α-cocoercive:

$$\langle F(x; d) - F(y; d), x - y\rangle \geq \alpha \|F(x; d) - F(y; d)\|^2, \quad \text{for all} \quad x, y$$

- **Data space:** $d \in \mathcal{D}$.

Reducing to Fixed Points

Variational Inequality (VI) Problem

Find $x_d^\circ \in \mathcal{C}$ such that:

$$\langle F(x_d^\circ; d), x - x_d^\circ \rangle \geq 0, \quad \text{for all} \quad x \in \mathcal{C}. \tag{3}$$

$$\mathrm{VI}(F(\cdot\,; d), \mathcal{C}) = \{\text{all such } x_d^\circ\}.$$

We have the following equivalence[2]:

$$x_d^\circ \text{ is a Nash Equilibrium} \iff x_d^\circ \in \mathrm{VI}(F(\cdot\,; d), \mathcal{C}). \tag{4}$$

[2] Prop. 1.4.2 in *Finite-dimensional variational inequalities* Facchinei & Pang

Reducing to Fixed Points

- Consider PGD-type operator:

$$R(x;d) \triangleq P_{\mathcal{C}}(x - \alpha F(x;d)), \tag{5}$$

- The indicator function $\delta_{\mathcal{C}}(x) \triangleq \begin{cases} 0 & \text{if } x \in \mathcal{C} \\ +\infty & \text{otherwise} \end{cases}$ is convex.
- Subgradient $\partial \delta_C$ also known as **normal cone operator**.
- We have the following equivalence[3]:

$$x_d^\circ \in \mathrm{VI}(F(\cdot\,;d), \mathcal{C}) \iff 0 \in F(x_d^\circ; d) + \partial \delta_{\mathcal{C}}(x^\circ) \iff x_d^\circ = R(x_d^\circ; d).$$

- Find NE by finding FP of $R(\cdot\,;d)$.

[3] Chpt. 12 in *Finite-dimensional variational inequalities* Facchinei & Pang

Outline

Problem Formulation I

Recall the following:

- $F \triangleq \left[\nabla_{x_1} u_1^\top \cdots \nabla_{x_K} u_K^\top\right]^\top$ is game gradient.
- PGD-type operator: $R(x;d) \triangleq P_{\mathcal{C}}(x - \alpha F(x;d))$.
- NEs are FPs of $R(\cdot\,;d)$.

Henceforth assume u_k are *unknown*.

Proposal I

Use historical data $\{(d, x_d^\star)\}$ to learn operator F_Θ such that if

$$x_d^\circ = P_{\mathcal{C}}\left(x_d^\circ - \alpha F_\Theta(x_d^\circ; d)\right) \qquad (6)$$

then $x_d^\circ \approx x_d^\star$.

Similar ideas proposed in *What game are we playing?* Ling *et al* (2018); *End-to-End Learning and Intervention in Games* Li *et al* (2020).

Naive N-FPN

- Formalize this as an N-FPN $\mathcal{N}_\Theta : \mathcal{D} \to \mathcal{C}$.
- F_Θ is a tunable operator (*e.g.* a neural network).
- Define $T_\Theta(x; d) = P_\mathcal{C}(x - \alpha F_\Theta(x; d))$. Then:

$$\mathcal{N}_\Theta(d) = x_d^\circ \text{ where } x_d^\circ = T_\Theta(x^\circ; d) \tag{7}$$

- $\mathcal{N}$ is an **implicit depth neural network**[4].

Algorithm 1 Naive N-FPN

$\mathcal{N}_\Theta(d)$: ◁ Input data is d
  $x^0, x^1 \leftarrow \tilde{x}$, $n \leftarrow 1$, ◁ Initializations
  while $\|x^n - x^{n-1}\| > \varepsilon$ or $n = 1$ ◁ Loop to fixed point
    $x^{n+1} \leftarrow T_\Theta(x^n; d)$ ◁ Apply T update
    $n \leftarrow n + 1$ ◁ Increment counter
  return x^n ◁ Output inference

[4] See also *Implicit deep learning* El Ghaoui *et al* (2019), *Deep equilibrium models* Bai *et al* (2019) and many others.

Backprop for N-FPN

- Recall: $\mathcal{N}_\Theta(d) = x_d^\circ$ where $x_d^\circ = T_\Theta(x^\circ; d)$.
- Given data $\{(d^i, x_{d^i}^\star)\}$ the training problem is:

$$\min_\Theta \mathbb{E}_{d\sim\mathcal{D}}\left[\ell(\mathcal{N}_\Theta(d), x_d^\star)\right] \approx \sum_i \ell\left(\mathcal{N}_\Theta(d^i), x_{d^i}^\star\right) \tag{8}$$

- For gradient based training need:

$$\begin{aligned}
\frac{\mathrm{d}\ell}{\mathrm{d}\Theta} &= \frac{\mathrm{d}\ell}{\mathrm{d}x}\frac{\mathrm{d}\mathcal{N}_\Theta}{\mathrm{d}\Theta} \quad \textit{(chain rule.)} \\
&= \frac{\mathrm{d}\ell}{\mathrm{d}x}\frac{\mathrm{d}x_d^\circ}{\mathrm{d}\Theta} \\
&= \frac{\mathrm{d}\ell}{\mathrm{d}x}\left(\mathrm{Id} - \frac{dT_\Theta}{dx}\right)^{-1}\frac{\partial T_\Theta}{\partial\Theta} \quad \textit{(implicit function theorem.)}
\end{aligned}$$

- Computing/inverting Jacobian dT_Θ/dx is computationally taxing.

Jacobian-Free Backprop (JFB)

- Gradient: $\dfrac{\mathrm{d}\ell}{\mathrm{d}\Theta} = \dfrac{\mathrm{d}\ell}{\mathrm{d}x}\left(\mathrm{Id} - \dfrac{dT_\Theta}{dx}\right)^{-1}\dfrac{\partial T_\Theta}{\partial \Theta}.$
- In prior work[5] we show $p \triangleq \dfrac{\mathrm{d}\ell}{\mathrm{d}x}\dfrac{\partial T_\Theta}{\partial \Theta}$ is a descent direction for $\ell(\Theta)$.
- JFB: Use p instead of $d\ell/d\Theta$.

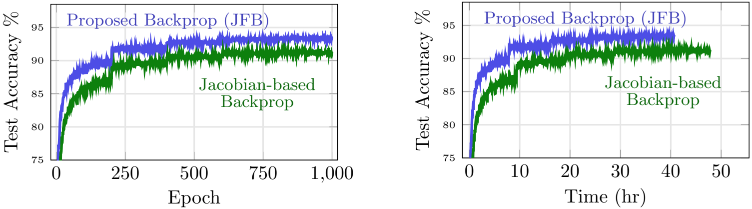

Figure: Training an implicit neural network on CIFAR10. JFB is faster and yields higher test accuracy than Jacobian-based backprop.

[5] *Fixed Point Networks* Wu Fung *et al* (2021)

Experimental Results I: Contextual RPS

- Recall game setup:

$$u_1(x_1) = x_1^\top B(d)x_2 \ , \ u_2(x_2) = -x_1^\top B(d)x_2 \text{ and } \mathcal{C} = \Delta^3 \times \Delta^3 \quad (9)$$

- Randomly generate $d^i \in [0,1]^3$. Solve exactly to obtain $x^\star_{d^i}$.
- F_Θ is residual, two-layer, fully connected network.
- N-FPN trained using JFB.

	R	P	S
R	0	$-\langle w^1, d\rangle$	$\langle w^2, d\rangle$
P	$\langle w^1, d\rangle$	0	$-\langle w^3, d\rangle$
S	$-\langle w^2, d\rangle$	$\langle w^3, d\rangle$	0

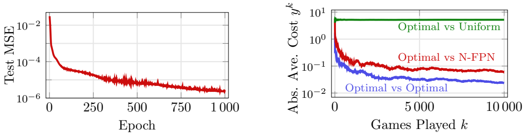

Figure: **Left:** Cost matrix. w^i are fixed. **Middle:** Test loss decreases rapidly when using JFB. **Right:** Average cost of uniform random, N-FPN and optimal player against an optimal player. Lower is better.

Outline

The Primary Bottleneck is Projection

- Recall the following:

$$T_\Theta(x; d) = P_\mathcal{C}(x - \alpha F_\Theta(x; d))$$
$$\mathcal{N}_\Theta(d) = x_d^\circ \text{ where } x_d^\circ = T_\Theta(x^\circ; d)$$
$$p = \frac{\mathrm{d}\ell}{\mathrm{d}x}\frac{\partial T_\Theta}{\partial \Theta} = \frac{\mathrm{d}\ell}{\mathrm{d}x}\frac{\mathrm{d}P_\mathcal{C}}{\mathrm{d}x}\frac{\partial}{\partial \Theta}\left[x - \alpha F_\Theta(x; d)\right]$$

- Forward pass requires $P_\mathcal{C}$. Backprop requires $dP_\mathcal{C}/dx$.
- For nice $\mathcal{C}$ (*e.g.* simplex) compute $P_\mathcal{C}, dP_\mathcal{C}/dx$ in[6] $\mathcal{O}(m)$ FLOPS[7].
- General case[8] requires $\mathcal{O}(m^3)$ FLOPs to approx. to machine ε.

---

[6] Here $m = \dim(\mathcal{C})$.

[7] *Fast projection onto the simplex* Condat (2016)

[8] *Optnet: Differentiable optimization as a layer* Amos & Kolter (2018)

Revisiting N-FPN formulation

- In many cases $\mathcal{C} = \mathcal{C}^1 \cap \mathcal{C}^2$ where $P_{\mathcal{C}^1}$ and $P_{\mathcal{C}^2}$ are simple.
- Recall: $x_d^\star$ is a Nash Equilibrium $\iff x_d^\star \in \mathrm{VI}(F(\cdot\,; d), \mathcal{C})$.
- We show following equivalence:

$$x_d^\star \in \mathrm{VI}(F(\cdot\,; d), \mathcal{C}) \iff 0 \in F(x_d^\star; d) + \partial\delta_{\mathcal{C}^1}(x_d^\star) + \partial\delta_{\mathcal{C}^2}(x_d^\star)$$

- Key idea[9]: $\partial\delta_{\mathcal{C}} = \partial\delta_{\mathcal{C}^1} + \partial\delta_{\mathcal{C}^2}$ for polyhedral $\mathcal{C}^i$.

[9] Theorem 23.8.1 in *Convex analysis* Rockafeller

Revisiting N-FPN formulation

- Define the operator:

$$T(x; d) \triangleq x - P_{\mathcal{C}^1}(x) + P_{\mathcal{C}^2}\left(2P_{\mathcal{C}^1}(x) - x - F(P_{\mathcal{C}^1}(x); d)\right)$$

- Apply Davis-Yin splitting[10]:

$$0 \in F(x_d^\star; d) + \partial\delta_{\mathcal{C}^1}(x_d^\star) + \partial\delta_{\mathcal{C}^2}(x_d^\star)$$
$$\iff x_d^\star = P_{\mathcal{C}^1}(z_d^\star) \text{ where } z_d^\star = T(z_d^\star; d)$$

Proposal II

Use historical data $\{(d, x_d^\star)\}$ to learn operator F_Θ such that if

$x_d^\circ = P_{\mathcal{C}^1}(z_d^\circ)$ where $z_d^\circ = T_\Theta(z_d^\circ; d)$ and

$T_\Theta(x; d) \triangleq x - P_{\mathcal{C}^1}(x) + P_{\mathcal{C}^2}\left(2P_{\mathcal{C}^1}(x) - x - F_\Theta(P_{\mathcal{C}^1}(x); d)\right)$

then $x_d^\circ \approx x_d^\star$.

[10] *A three-operator splitting scheme ...* Davis & Yin (2017)

Full N-FPN

- Again, formalize this as N-FPN $\mathcal{N}_\Theta : \mathcal{D} \to \mathcal{C}$.
- F_Θ is a tunable operator (*e.g.* a neural network).

$$\mathcal{N}_\Theta(d) \triangleq P_{\mathcal{C}^1}(z_d^\circ) \text{ where } z_d^\circ = T_\Theta(z_d^\circ; d) \text{ and}$$

$$T_\Theta(x; d) \triangleq x - P_{\mathcal{C}^1}(x) + P_{\mathcal{C}^2}\left(2P_{\mathcal{C}^1}(x) - x - F_\Theta(P_{\mathcal{C}^1}(x); d)\right)$$

Algorithm 2 Nash Fixed Point Network (N-FPN)

$\mathcal{N}_\Theta(d)$: ◁ Input data d
    $z^1 \leftarrow \tilde{z}, \;\; z^0 \leftarrow \tilde{z}, \;\; n \leftarrow 1$ ◁ Initialize
    while $\|z^n - z^{n-1}\| > \varepsilon$ **or** $n = 1$ ◁ Loop till fixed point
        $x^{n+1} \leftarrow P_{\mathcal{C}^1}(z^n)$ ◁ Project
        $y^{n+1} \leftarrow P_{\mathcal{C}^2}(2x^{n+1} - z^n - F_\Theta(x^{n+1}; d))$ ◁ Project
        $z^{n+1} \leftarrow z^n - x^{n+1} + y^{n+1}$ ◁ Combine sequences
        $n \leftarrow n + 1$ ◁ Increment counter
    return $P_{\mathcal{C}^1}(z^n)$ ◁ Output inference

Decoupling constraints is much cheaper

- A typical case: $\mathcal{C} = \underbrace{\{x : Nx = b_k\}}_{\mathcal{C}^1} \bigcap \underbrace{\{x : x \geq 0\}}_{\mathcal{C}^2}$.
- Computing $P_{\mathcal{C}^1}$ *exactly* is $\mathcal{O}(m)$.
- Computing $P_{\mathcal{C}^2}$ *exactly* is[11] $\mathcal{O}(m^2)$.
- Computing $dP_{\mathcal{C}^1}/dx$ and $dP_{\mathcal{C}^2}/dx$ handled by autodiff.
- Extend decoupling to multi-intersection: $\mathcal{C} = \mathcal{C}^1 \cap \cdots \cap \mathcal{C}^K$.
- Also extend to Minkowski sum:

$$\mathcal{C} = \mathcal{C}_1^1 \cap \mathcal{C}_1^2 + \cdots + \mathcal{C}_K^1 \cap \mathcal{C}_K^2 \tag{10}$$

[11] There is a once-off cost of $\mathcal{O}(m^3)$ for computing and storing SVD of N

Outline

Modelling large-scale traffic routing[12]

- Road network = directed graph. V = vertices, E = edges.
- $N \in \mathbb{R}^{|V|\times|E|}$ is **incidence matrix**.
- Aggregative game: agents are infinitesimal.
- **OD-pair**: (v_1, v_2, q). q units of traffic to be routed from v_1 to v_2.
- Vectorize to $b \in \mathbb{R}^{|V|}$ where $b_{v_1} = q$, $b_{v_2} = -q$,$b_i = 0$ if $i \neq v_1, v_2$.
- **Traffic flow** is $x \in \mathcal{C}$ where $x[e] =$ (traffic density on road e) and

$$\mathcal{C} = \underbrace{\{x : Nx = b\}}_{\mathcal{C}^1} \cap \underbrace{\{x : x \geq 0\}}_{\mathcal{C}^2} \tag{11}$$

- Travel time operator: $F : \mathbb{R}^{|E|} \times \mathcal{D} \to \mathbb{R}^{|E|}$ with $F_e(x) \triangleq t(x[e]; d)$.

Wardrop's First Principle

If players are self-interested then resulting flow satisfies $x_d^\star \in \mathrm{VI}(F(\cdot\,; d), \mathcal{C})$.

[12] *The traffic assignment problem: models and methods* Patriksson

Contextual traffic routing

- d captures global factors, *e.g.* weather.
- Typically $t(x[e]; d)$ unknown.
- Extend to multiple OD pairs:

$$\mathcal{C} = \mathcal{C}_1 + \cdots + \mathcal{C}_K$$

$$\mathcal{C}_k = \underbrace{\{x : Nx = b^k\}}_{\mathcal{C}_k^1} \cap \underbrace{\{x : x \geq 0\}}_{\mathcal{C}_k^2}$$

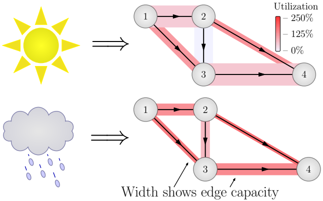

Figure: Predicted traffic flow on a sunny and rainy day.

Proposal III

Use historical data $\{(d, x_d^\star)\}$ to train N-FPN $\mathcal{N}_\Theta$ such that $\mathcal{N}_\Theta(d) \approx x_d^\star$.

(Somewhat) similar ideas proposed in *Data-driven estimation in equilibrium using inverse optimization* Bertsimas *et al* (2015) and others.

New contextual traffic routing dataset

- Used road networks for real cities[13] (Anaheim, Berlin ...).
- Fixed travel-time function $t(x[e]; d)$.
- Generated random d^i for $i = 1, \ldots 5.5K$.
- Solved $\mathrm{VI}(F(\cdot\,; d), \mathcal{C})$to obtain $x^\star_{d^i}$.
- Data available at Git repo.

[13]From *Transportation Networks for Research*

Example II: Contextual Traffic Routing

- F_Θ is 2–3 layer fully connected N-FPN (~ 100K trainable params.)
- N-FPN trained using JFB.
- Given $x_d^\circ = \mathcal{N}(d)$ quantify accuracy as

$$\text{TRAFIX}(x_d^\circ, x_d^\star) \triangleq \frac{\#\left\{e \in E : |x_d^\circ[e] - x_d^\star[e]| < \varepsilon |x_d^\star[e]|\right\}}{|E|}$$

dataset	edges/nodes	OD-pairs	TRAFIX score
Sioux Falls	76/24	528	0.94
Eastern Mass.	258/74	1113	0.97
Berlin-Friedrichshain	523/224	506	0.97
Berlin-Tiergarten	766/361	644	0.95
Anaheim	914/416	1406	0.95

Table: Results of using N-FPN to predict traffic flows. For TRAFIX score, $\varepsilon = 5 \times 10^{-3}$.

Thank you!

Papers:

- Wu Fung, Heaton, Li, McKenzie, Osher & Yin: *Fixed Point Networks: Implicit Depth Models with Jacobian-Free Backprop*.
- Heaton, McKenzie, Li, Wu Fung, Osher & Yin: *Learn to Predict Equilibria via Fixed Point Networks*.

Code:

- `https://github.com/howardheaton/nash_fixed_point_networks`
- `https://github.com/howardheaton/fixed_point_networks`