مجله کنترل
ISSN (print) 2008-8345
ISSN (online) 2538-3752
مقاله پژوهشی
جلد۱۷، شماره ۱، بهار ۱۴۰۲، صفحه ۷۶- ۶۱

یادگیری تکرار سیاست حداقل مربعات عصبی با معماری نقاد- تنها

امید محرابی[۱]، احمد فخاریان[۲]، مهدی سیاهی[۳]، امین رمضانی[۴]

[۱]دانشجوی دکتری مهندسی برق- کنترل، واحد علوم و تحقیقات، دانشگاه آزاد اسلامی، تهران، ایران o.mehrabi62@gmail.com

[۲]دانشیار، گروه مهندسی برق، واحد قزوین، دانشگاه آزاد اسلامی، قزوین، ایران ahmad.fakharian@qiau.ac.ir

[۳]دانشیار، دانشکده مهندسی برق و کامپیوتر، واحد علوم و تحقیقات، دانشگاه آزاد اسلامی، تهران، ایران mehdi_siahi@yahoo.com

[۴]دانشیار، دانشکده مهندسی برق و کامپیوتر، دانشگاه تربیت مدرس، تهران، ایران ramezani@modares.ac.ir

دریافت: ۱۴۰۱/۱۰/۰۲ ویرایش: ۱۴۰۲/۰۱/۲۵ پذیرش: ۱۴۰۲/۰۳/۱۶

چکیده: کنترل هوشمند مسائل کنترلی واقعی بر پایه یادگیری تقویتی اغلب نیاز به تصمیم‌گیری در فضای حالت- عمل بزرگ و یا پیوسته دارد. از آنجا که تعداد پارامترهای قابل تنظیم در یادگیری تقویتی گسسته، رابطه مستقیمی با عدد اصلی فضای متغیرهای حالت- عمل مسأله دارد، لذا در چنین مسائلی مشکل تنگنای ابعاد، سرعت کم یادگیری و راندمان پایین وجود دارد. استفاده از روش‌های آموزش تقویتی پیوسته برای حل این مشکلات مورد توجه محققان است. در همین راستا، در این مقاله یک الگوریتم جدید یادگیری تقویتی عصبی (NRL) بر مبنای معماری نقاد- تنها برای حل مسائل کنترلی معرفی می‌گردد. روش ارائه شده یک روش مستقل از مدل و نرخ یادگیری است و از ترکیب روش "تکرار سیاست کمترین مربعات" (LSPI) با شبکه توابع پایه شعاعی (RBF) به عنوان یک تقریب زننده‌ی تابعی حاصل شده است. الگوریتم پیشنهادی "تکرار سیاست کمترین مربعات عصبی" (NLSPI) نامیده می‌شود. در این روش، با استفاده از توابع پایه تعریف شده در ساختار شبکه عصبی RBF، راهکاری برای رفع چالشِ تعریف توابع پایه حالت- عمل در LSPI ارائه شده است. ورودی‌های شبکه جفت حالت و عمل‌های مسأله و خروجی آن تابع ارزش عمل تقریب زده شده می‌باشد. هدف، به روز رسانی برخط وزن‌های شبکه عصبی با استفاده از روش ارائه شده به صورتی است که بهترین تقریب از تابع ارزش عمل صورت گیرد. به منظور اعتبارسنجی روش ارائه شده، عملکرد الگوریتم پیشنهادی در مورد حل دو مسأله کنترلی با روش‌های دیگر مقایسه شده است. نتایج بدست آمده، برتری روش در یادگیری سیاست شبه بهینه را بخوبی نشان می‌دهد.

کلمات کلیدی: یادگیری تقویتی عصبی، معماری نقاد- تنها، تکرار سیاست کمترین مربعات، شبکه توابع پایه شعاعی

Neural Least Square Policy Iteration learning with Critic-only architecture

Omid Mehrabi, Ahmad Fakharian, Mehdi Siahi, Amin Ramezani

Abstract: Intelligent control of real control problems based on reinforcement learning often requires decision-making in a large or continuous state-action space. Since the number of adjustable parameters in discrete reinforcement learning has a direct relationship with cardinality of the state-action space of the problem, so in such problems, we are faced with the curse of dimensiality, low learning speed and low efficiency. The use of continuous reinforcement learning methods to overcome these problems have attracted many research interests. In this paper a novel Neural Reinforcement Learning (NRL) scheme is proposed. The presented method is model free and learning rate independent, and is obtained by combining Least Squares Policy Iteration (LSPI) with Radial Basis Functions (RBF) as a function approximator, and we call it "Neural Least Squares Policy Iteration" (NLSPI). In this method, by using the basis functions defined in the RBF neural network structure, we have provided a solution to solve the challenge of defining the state-action basis functions in LSPI. In order to validate the presented method, the performance of the proposed

[DOI: 10.61186/joc.17.1.61]

algorithm in solving two control problems has been compared with other methods. The overall results show the superiority of our method in learning the pseudo-optimal policy.

Keywords: Neural reinforcement learning, Critic-only architecture, Least Square Policy Iteration, RBF network.

۱- مقدمه

در تئوری کنترل کلاسیک، کنترل یک فرآیند مستلزم داشتن دانشی کامل از سیستم مورد نظر است و به طور کلی در آن کنترل‌کننده برای یک مدل ریاضی شناخته شده طراحی می‌گردد. با پیشرفت‌های انجام شده در تئوری کنترل، عدم قطعیت[1] و نایقینی[2] سیستم‌های واقعی نیز که مانع از تحقق مطلوب اهداف کنترلی بود در نظر گرفته شدند. هر چند در تئوری-های جدید فرض می‌شود که بعضی از ویژگی‌های غیرقطعی شناخته شده هستند با این وجود در بعضی موارد ممکن است این مفروضات برای کنترل موفقیت آمیز سیستمی که با زمان تغییر می‌کند، کافی نباشند بنابراین لازم است که سیستم را حین عمل تحت نظر قرار داد و دانش بیشتری را بدین طریق بدست آورد. بعبارت دیگر از آنجایی که مفروضات اولیه کافی نمی‌باشند، اطلاعات اضافی را باید به صورت همزمان کسب نمود. یک رویکرد بررسی این موارد در نظر گرفتن آن‌ها بعنوان مسائلی در یادگیری است. ایده طراحی یک سیستم یادگیر، تضمین ارائه رفتاری مقاوم، بدون داشتن دانشی کامل در مورد سیستم یا محیط مورد نظر می‌باشد [۱، ۲]. یادگیری را بدین صورت می‌توان تعریف کرد: "بدست آوردن دانش جدید، مهارت کنترل خود و درک و استفاده از آنها در فعالیت‌های بعدی در صورتی که این بدست آوردن و استفاده کردن توسط خود سیستم انجام گرفته و باعث بهبود رفتار گردد. یادگیری تقویتی[3] (RL)، یک روش قوی مدرن برای آموزش رویِ خط استراتژی‌های کنترل از طریق تعامل با محیط است. در این رویکرد، کنترل‌کننده مبتنی بر یادگیری تقویتی تلاش می‌کند تنها از طریق یک معیار عددی راندمان که سیگنال تقویت یا پاداش نامیده می‌شود و بدون نیاز به سرپرست از طریق تعامل هوشمند با محیطی که در حالت کلی دارای روندهای غیرخطی و اتفاقی است تجربه‌اندوزی کرده و به سیاست و استراتژی بهینه برای رسیدن به اهدافش دست یابد. قابلیت‌های مذکور باضافه قدرت کاوش بالا در جهت یافتن پاسخ بهینه، عدم نیاز به داده‌های آموزشی، توانایی برخط[4] در جهت بهبود تدریجی عملکرد، توانایی تطبیق‌پذیری با نوسانات دینامیکی فرآیند و تواناییش در کنترل بدون دانستن مدل محیط[5] آن را تبدیل به یک روش کارآمد جهت تنظیم پارامترهای یادگیرنده (کنترل‌کننده هوشمند) نموده است [۲، ۳].

سرعت کم یادگیری و نفرین ابعاد[6] از جمله مشکلاتی هستند که کاربرد روش‌های رایج یادگیری تقویتی (حالت گسسته) را برای حل مسائل کنترل واقعی که دارای فضای حالت و عمل بزرگ و یا پیوسته هستند محدود نموده است [۳، ۴]. راه‌حل پیشنهادی برای این چالش، استفاده از یادگیری تقویتی پیوسته است که از ترکیب الگوریتم‌های یادگیری تقویتی گسسته با تقریب زننده‌های تابعی[7]، همچون شبکه‌های عصبی و سیستم‌های فازی حاصل می‌شود [۵، ۶]. از طرفی، به کارگیری الگوریتم‌های یادگیری تقویتی پیوسته در مسائل کنترل با دو چالش عمده روبروست: ۱) قالب کلی اکثر پژوهش‌های انجام شده در زمینه یادگیری تقویتی پیوسته دارای معماری عملگر- نقاد[8] است. این معماری دارای دو بخش عملگر و نقاد بوده و در آن بخش نقاد برای تقریب تابع ارزش و بخش عملگر برای تولید عمل استفاده می‌گردد [۱۲-۷]. با وجود کاربرد گسترده‌ی الگوریتم‌های عملگر- نقاد در بسیاری از مسائل توسط محققین، در مطالعات قبلی نشان داده شده است که الگوریتم‌های مذکور دارای فقدان کاوش مناسب می‌باشند [۱۳]. درمقابلِ معماری عملگر- نقاد، روش نقاد- تنها[9] فقط دارای یک بخش نقاد است که برای تقریب تابع ارزش عمل[10] استفاده می‌شود و عمل نهایی با توجه به مقادیر ارزش تقریب‌زده شده با استفاده از یکی از روش‌های انتخاب عمل همچون شبه‌حریصانه[11] و یا بیشینه هموار[12] تولید می‌گردد. این نحوه‌ی انتخاب عمل درجه کاوش بالا را باعث می‌شود [۳، ۵]. ۲) وابستگی به نرخ آموزش: تنظیم پارامتر نرخ آموزش برای هر مسأله به صورت مجزا، در الگوریتم‌های یادگیری تقویتی پیوسته، یک چالش است. با توجه به اینکه اکثر روش‌های ارائه شده به نوعی توسیعی از الگوریتم‌های گسسته وابسته به نرخ آموزش می‌باشند لذا ریشه مشکل را به نوعی می‌توان در الگوریتم پایه استفاده شده دانست.

یک کلاس از الگوریتم‌های مبتنی بر تابع ارزش، تکرار سیاست کمترین مربعات[13] (LSPI) است که دارای ویژگی‌های مطلوبی همچون تحلیل ریاضی مثبت، عدم وابستگی به نرخ آموزش و کارایی مناسب می-باشد این خواص وام گرفته شده از الگوریتم پایه روش یعنی تکرار سیاست (PI) است و اولین بار توسط Lagoudakis و Parr بر روی فضای حالت- عمل متناهی معرفی شد [۱۲]. بعد از آن توصیف کامل‌تری از روش توسط خود آن‌ها در [۱۴] ارائه گردید. تحلیل ریاضی LSPI با استناد به تحلیل

[1] Nondeterministic
[2] Uncertainty
[3] Reinforcement Learning
[4] Online
[5] Model Free
[6] Curse of dimentiality
[7] Function approximators
[8] Actor-Critic
[9] Critic-Only
[10] Action Value Function
[11] ε-Greedy
[12] Softmax
[13] Least Square Policy Iteration (LSPI)

[DOI: 10.61186/joc.17.1.61]

ریاضی روش تکرار سیاست[1] (PI) ارایه شده است. یک نسخه برخط روش برای فضای حالت- عمل نامحدود توسط Bus¸oniu و همکارانش در [۱۴] تشریح شده اما هیچ تحلیل ریاضی مثبتی در جهت همگرایی برای فضای حالت- عمل نامتناهی ارائه نشده است. ایده بنیادین تمام روش‌های مذکور استفاده از ترکیب خطی وزن‌دار توابع پایه جهت تقریب تابع ارزش حالت- عمل به جای محاسبه دقیق آن می‌باشد اما در هیچکدام نحوه تعریف این توابع پایه که خود یک چالش است بیان نشده است. جدول

جدول ۱: مقایسه الگوریتم‌های یادگیری تقویتی مبتنی بر یادگیری تکرار سیاست حداقل مربعات

Approach		On-Line	Continues		Admiss. System dimensionality	Kernel trick	Initial Policy required
Ref.	Algorithm		State	Action			
Lagoudakis and Parr (2003)	LSPI	×	✓	×	Low	×	×
Bu˘soniu et al. (2010)	OLSPI	✓	✓	×	Low	×	×
Bu˘soniu et al. (2010)	OLSPI with cont. action	✓	✓	✓	Low	×	×
Xu et al. (2007)	KLSPI	×	✓	×	Mid	✓	✓
Yahyaa and Manderick (2014)	Online KBLSPI	✓	✓	×	Mid	✓	×
Jakab and Csato (2015)	KRLSTD	×	✓	×	Low	✓	✓
Cui et al. (2017)	KDPP	×	✓	×	High	✓	×
Ghorbani, et al. (2017)	FLSPI	✓	✓	✓	Mid	✓	✓
Our Approach	NLSPI	✓	✓	×	Mid	✓	✓

(۱) چند الگوریتم مختلف به کار گرفته شده در این زمینه را بر اساس چند معیار مختلف مورد مقایسه قرار داده است [۱۹-۱۶]. درمقاله حاضر با استفاده از یک ساختار تقریب زننده عصبی و ترکیب آن با الگوریتم LSPI، یک الگوریتم ترکیبی یادگیری تقویتی پیوسته جدید با معماری نقاد- تنها که ما آن را یادگیری تکرار سیاست حداقل مربعات عصبی (NLSPI) می‌نامیم مورد مطالعه و بررسی قرار می‌دهیم. الگوریتم مذکور از ترکیب الگوریتم LSPI با یک شبکه عصبی بر پایه توابع شعاعی حاصل شده است به نحوی که شرایط LSPI برآورده شود. از یک جهت از شبکه عصبی برای حل چالش تعریف توابع پایه حالت- عمل در الگوریتم LSPI و از جهت دیگر از LSPI برای تنظیم وزن‌های شبکه عصبی استفاده شده است.

ساختار مقاله در ۵ بخش به شرح ذیل تنظیم شده است. در بخش بعد مفاهیم پایه‌ای یادگیری تقویتی و یادگیری تکرار سیاست کمترین مربعات به طور مختصر توضیح داده شده است. یادگیری تکرار سیاست کمترین مربعات عصبی در بخش سوم توضیح داده می‌شود. در بخش چهارم مسأله‌ای که برای ارزیابی عملی ایده‌ها مورد شبیه‌سازی قرار گرفته، تشریح گشته و مختصری از نتایج شبیه‌سازی ارائه می‌گردد. در نهایت بخش پنجم به طرح نتایج مهم، نتیجه‌گیری مطالب و پیشنهادها اختصاص دارد.

۲- مقدمات و تعاریف اولیه

یافتن یک سیاست بهینه در مسأله‌ای که به صورت فرآیند تصمیم‌گیری مارکوف (MDP)[2] مدل شده است یکی از مباحثی است که در نظریه بهینه‌سازی و مهندسی کنترل بسیار مورد توجه بوده است [۲۰]. دریک سیستم مارکوف حالت بعدیِ محیط و پاداش دریافتی تنها به عمل و حالت قبلی عامل در محیط بستگی دارد. یک MDP یک چندتایی $\langle S, A, P, \mathcal{R}, \psi, \gamma \rangle$ است. جایی که $S = \{s_1, s_2, \dots, s_{|S|}\}$ یک مجموعه محدود حالت‌های گسسته محیط، $A = \{a_1, a_2, \dots, a_{|A|}\}$ مجموعه محدود عمل‌های گسسته عامل، $\psi \subseteq S \times A$ مجموعه زوج‌های حالت- عمل قابل قبول، $P: \psi \times S \to [0,1]$ ماتریس احتمال گذر از حالت $s_t = s$ به حالت بعدی $s_{t+1} = s'$ با انجام عمل $a_t = a$ است به طوری که $P(s,a,s') \geq 0$ و $\forall (s,a) \in \psi \sum_{s' \in S} P(s,a,s') = 1$ و $\mathcal{R}: \psi \to \mathbb{R}$ ماتریس پاداش دریافتی توسط عامل می‌باشد به گونه‌ای که $(r_{t+1} = r(s_t, a_t)) \in \mathbb{R}_{s,s' \in S, a \in A}$ متغیر تصادفی با میانگین $r(s,a,s')$، پاداش یا جریمه‌ای آنی است که محیط به عامل بر حسب کیفیت عمل انتخابی او در حالت $s_t = s$، می‌دهد. $\gamma \in [0,1]$ نرخ تنزیل می‌باشد [۳-۱].

در یک مسأله یادگیری تقویتی دوره‌ای[3] استاندارد، عامل[4] (کنترل‌کننده) تلاش می‌کند رفتار بهینه را در تعامل با یک محیط پویا از طریق پروسه سعی و خطا، بدون آنکه مدل محیط و یا حتی نحوه‌ی انجام عمل برای آن مشخص باشد، تنها از طریق یک معیار عددی راندمان بیاموزد [۳]. هر اپیزود شامل T گام زمانی $s_0, a_0, r_1, s_1, a_1, \dots, r_T, a_T$ است. در هر وضعیت $s_t = s$، $0 < t < T$، عمل $a_t = a$، با استفاده از یک تابع احتمال $\pi: \psi \to [0,1]$ که سیاست یا استراتژی[5] عامل نامیده می‌-

[1] Policy Iteration
[2] Markov Decision Process (MDP)
[3] Episodic
[4] Agent
[5] Policy

[DOI: 10.61186/joc.17.1.61]

شود انتخاب می‌گردد. هدف عامل یادگیری سیاستی است که به بیشینه کردن مجموع پاداش‌ها یا حداقل نمودن یک تابع هزینه $g(s_t,a_t,s_{t+1}) \in [-g_{max}, g_{max}]$ در طول زمان منجر شود:

$$\lim_{T\to\infty} \sup\ E\left\{\frac{1}{T}\sum_{t=0}^{T} g(s_t,a_t,s_{t+1})\right\} \qquad (۱)$$

نحوه تعامل عامل (کنترل‌کننده) و محیط (فرآیند) در یک مسأله نوعی کنترلی با چارچوب یادگیری تقویتی در شکل (۱) نشان داده شده است. عامل یادگیرنده از طریق حس‌گرها، توصیفی از حالت محیط دریافت می‌کند $s_t \in S$ و بر مبنای این حالت، عمل خود $a_t \in A(s_t)$ را بر اساس سیاست اتخاذ شده $\pi(s_t,a_t)$ انجام می‌دهد. یک گام بعد یعنی در $t+1$، بسته به میزان مطلوبیت عمل عامل، محیط یک پاداش عددی $r_{t+1}=r(s_t,a_t)$ به وی می‌دهد و با احتمال $P(s_t,a_t,s')$ به حالت جدید $s_{t+1}=s'$ می‌رود. در حالت کلی، منظور از حل یک مسأله‌ی یادگیری تقویتی، پیدا کردن سیاست بهینه π^* است به نحوی که مقدار خروجی سیاست و یا ارزش هر کدام از حالت‌ها، بیشینه شوند [۱، ۲]. روش‌های متفاوتی برای تعریف خروجی وجود دارند. روشی که در اکثر کاربردها معمول است تعریف خروجی به صورت تنزیلی می‌باشد:

$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k\, r_{t+k+1} \qquad (۲)$$

در این صورت ارزش حالت- عمل به صورت امید ریاضی کل پاداشی که عامل با شروع از حالت $s_t=s$، انجام عمل $a_t=a$ و سپس در پیش گرفتن سیاست π بدست می‌آورد $Q^\pi: S\times A \to \mathbb{R}$ تعریف می‌شود [۲]:

$$Q^\pi(s,a) = E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}\,|s_t=s,\ a_t=a\right\} \qquad (۳)$$

که در آن $E\pi\{\}$ نشان دهنده امید ریاضی است در صورتی که از سیاست π پیروی شود.

رابطه فوق را می‌توان به فرم ماتریسی زیر نوشت:

$$Q^\pi = \mathcal{R} + \gamma P B_\pi Q^\pi \qquad (۴)$$

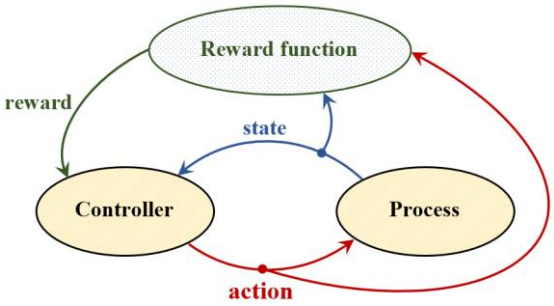
که در آن Q^π و $\mathcal{R}$ بردارهایی با اندازه $|S|.|A|$ و $P \in \mathbb{R}^{|S||A|\times|S|}$ یک ماتریس تصادفی است که شامل مدل انتقال فرآیند می‌باشد:

$$P\big((s,a),s'\big) = p(s,a,s') \qquad (۵)$$

$B_\pi \in \mathbb{R}^{|S||A|\times|S|}$ یک ماتریس تصادفی است که سیاست π را شرح می‌دهد:

$$B_\pi\big(s',(s',a')\big) = \pi(a';s') \qquad (۶)$$

سیستم خطی حاصل، $(I-\gamma P\, B_\pi)=\mathcal{R}$ می‌تواند از طریق تجزیه کردن یا به صورت تکراری برای بدست آوردن مقادیر Q^π حل شود. تابع ارزش حالت- عمل Q^π، نقطه ثابت[۱] عملگر بلمن است: $T_\pi\big(Q(s,a)\big)=\mathcal{R}+PB_\pi Q^\pi$ همچنین برای هر سیاست π و



شکل ۱: چارچوب یادگیری تقویتی در یک مسأله کنترلی

برداراولیه Q داریم: $\lim_{k\to\infty}(T_\pi)^k Q = Q^\pi$. T^π یک عملگر یکنواخت[۲] و شبه خطی[۳] است و یک نگاشت انقباض در نرم L_∞ با نرخ انقباض γ می‌باشد. اعمال متوالی T^π بر روی هر بردار اولیه Q به تابع ارزش عمل Q^π از سیاست π همگرا می‌شود.

۲-۱- مدلسازی الگوریتم تکرار سیاست کمترین مربعات

تکرار سیاست[۴] (PI)، یک روش برای یافتن سیاست بهینه برای هر MDP دلخواه است. در این روش یک حلقه بین ارزیابی سیاست[۵] و بهبود[۶] آن شکل می‌گیرد. در ارزیابی سیاست، مقدار تابع ارزش را بر حسب سیاست حریصانه‌ای که درنتیجه‌ی بهبود سیاست قبلی به دست آمده، تخمین می‌زند. از سوی دیگر، در بهبود سیاست، سیاست فعلی را برحسب اقدامی که مقدار تابع ارزش را در هر وضعیت حداکثر شود، به روز رسانی می‌کند. به روز رسانی معادلات بر پایه معادله بلمن انجام می‌گیرد و تکرار حلقه تا زمانی که این معادلات همگرا شوند، ادامه می‌یابد [۶، ۲۱].

$$Q^{\pi_m}(s,a) = \mathcal{R}(s,a) + \gamma\sum_{s'\in S} P(s,a,s')\sum_{a'\in A}\pi(s',a')Q^{\pi_m}(s',a') \qquad (۷)$$

$$\pi_{m+1}(s) = arg\max_a Q^{\pi_m}(s,a) \qquad (۸)$$

[۱] در صورتی که رابطه‌ی $F(x)=x$ برای تابع $F(x)$ به ازای حداقل یک x برقرار باشد، آنگاه x یک نقطه ثابت یا Fixed Point نامیده می‌شود [۲].

[۲] Monotonic
[۳] Quasi-Linear
[۴] Policy Iteration
[۵] Policy Evaluation
[۶] Policy Improvement

سیاست π_{m+1} یک سیاست قطعی است که اگر بهتر از π_m نباشد حداقل به خوبی آن است. این دو مرحله (ارزیابی سیاست و بهبود سیاست) تا زمانی که دیگر تغییری در سیاست تولید شده وجود نداشته باشد، تکرار می‌شوند. در چنین حالتی دنباله تولید شده سیاست‌ها به سیاست بهینه همگرا شده است.

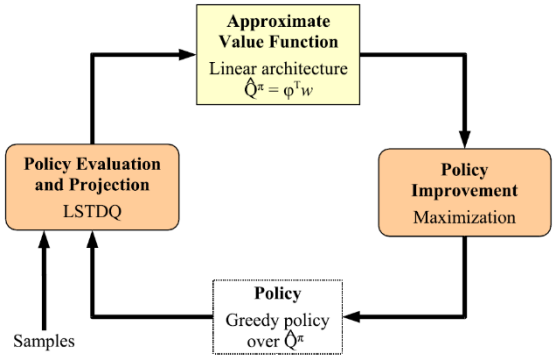
$$\pi_0 \xrightarrow{E} Q^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} Q^{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \dots \xrightarrow{I} \pi^* \xrightarrow{E} Q^*$$

اغلب با تعداد کمی تکرار، این همگرایی اتفاق می‌افتد. بهبود سیاست، به عنوان عملگر و ارزیابی سیاست، به عنوان نقاد شناخته می‌شوند. زیرا عملگر، مسئول روش عمل سیستم و نقاد، مسئول نقد نحوه عمل عامل است. بنابراین الگوریتم‌های تکرار سیاست، با معماری عملگر- نقاد در نظر گرفته می‌شوند. همگرایی تضمین شده تکرار سیاست به سیاست بهینه تحت نمایش جدولی تابع ارزش حالت- عمل، حل دقیق تساوی بلمن و نمایش جدولی هر سیاست، بسیار پر زحمت است. چنین نمایش‌ها و روش‌های دقیق، برای فضاهای حالت- عمل بزرگ، در عمل غیرممکن است. در این موارد، روش‌های تقریب مورد استفاده قرار می‌گیرند. تقریب تکرار سیاست[1] توسیع روش تکرار سیاست در فضای بزرگ یا پیوسته است [۲۲]. هر دو الگوریتم تکرار سیاست و تقریب تکرار سیاست به دلیل نیاز داشتن به یک مدل کامل[2] از محیط در یادگیری تقویتی کاربرد محدودی دارند زیرا فرض اساسی ما بر این است که عامل هیچ شناختی از مدل محیط ندارد لذا در چنین مواردی باید بر اطلاعاتی تکیه شود که از تعامل بین تصمیم‌گیرنده و فرآیند به دست می‌آیند.

الگوریتم تکرار سیاست کمترین مربعات (LSPI) یک روش تکرار سیاست در فضای بزرگ یا پیوسته است که سیاست را از روی نمونه‌ها یاد می‌گیرد [۱۲]. لذا در تبعیت از PI دارای دو مرحله ارزیابی و بهبود سیاست می‌باشد. در شکل (۲) بلوک دیاگرام الگوریتم LSPI نشان داده شده است. همچنان که از شکل مشخص است این روش از تخمین تابع ارزش حالت- عمل به جای محاسبه دقیق آن استفاده می‌کند. در حالت کلی، از ترکیب خطی وزن‌دار برای تخمین این تابع استفاده می‌شود. لذا در این حالت تابع ارزش- عمل $Q(s,a)$ با استفاده از تقریب زننده‌های خطی که دارای مجموعه ویژگی‌های $\mathrm{F} = \{\varphi(s,a): S \times A \rightarrow \mathbb{R}, i = 1, \dots, k\}$ متشکل از k توابع پایه[3] مستقل حالت- عمل دلخواه $\varphi(.)$ هستند برای تقریب تابع ارزش عمل $\hat{Q}(s,a)$ با استفاده از جمع وزندار توابع پایه پیشنهاد شده است [۱۲، ۱۶]:

$$\begin{aligned} Q^{\pi}(s,a) \approx \hat{Q}^{\pi}(s,a;\theta) &= \sum_{i=1}^{k} \varphi_i(s,a)\theta_i \\ &= \phi^T(s,a)\theta \\ \phi(s,a) &= [\varphi_1(s,a), \dots, \varphi_k(s,a)]^T \end{aligned} \quad (۹)$$

که $\theta \in \mathbb{R}^k$ بردار پارامترهای قابل تنظیم است. توابع $\varphi_i(s,a)$ در حالت کلی توابعی ثابت و دلخواه از s و a هستند که برای اطمینان از عدم افزونگی پارامترها لازم است که مستقل خطی باشند و ماتریس‌های



شکل ۲: بلوک دیاگرام الگوریتم تقریب سیاست کمترین مربعات *[۱۲]*

درگیر در محاسبات، دارای رتبه‌ی کامل[4] باشند. در حالت کلی $k \ll |S||A|$ در نتیجه، فضای مورد نیاز برای یک معماری خطی، بسیار کمتر از نمایش جدولی آن است. سیاست حریصانه π روی این تقریبِ تابع ارزش، برای هر حالت s، از بیشینه کردن تقریبِ ارزش روی مجموعه همه عمل‌های A بدست می‌آید:

$$\pi(s) = arg\max_{a \in A} \hat{Q}^{\pi}(s,a) = arg\max_{a \in A} \phi^T(s,a)\theta \quad (۱۰)$$

قطعه گم شده برای بستن این چرخه، روالی است که سیاست را با استفاده از نمونه‌ها ارزیابی می‌کند و تقریبِ تابع ارزش حالت- عمل را تولید می‌کند. در LSPI این مرحله توسط الگوریتم LSTDQ انجام می‌شود [۱۲]. روشی که بسیار شبیه الگوریتم LSTD است و تقریبِ تابع ارزشِ حالت- عمل $\hat{Q}^{\pi}$ را از سیاست π زمانی که معماری تقریب، خطی است به طور مناسبی یاد می‌گیرد. ایده اصلی از تقریبِ تابع ارزش حالت- عمل، تنظیم مناسب بردار پارامترهای قابل تنظیم $\theta \in \mathbb{R}^k$ به نحوی است که مقادیر حاصل از تقریب، به اندازه کافی به مقادیر واقعی نزدیک باشند. در این صورت $\hat{Q}^{\pi}$ می‌تواند به جای مقدار دقیق تابع Q^{π} مورد استفاده قرار گیرد. یک راه برای یافتن یک تقریبِ مناسب این است که تقریبِ تابع ارزش را وادار کرد که نقطه ثابت عملگر بلمن باشد: $T^{\pi}\hat{Q}^{\pi} \approx \hat{Q}^{\pi}$ برای تحقق این موضوع، نقطه ثابت باید در فضای تقریبِ توابع ارزش قرار بگیرد، به طوری که این فضا از اسپن توابع پایه به وجود آمده باشد. در حالت کلی ممکن است که $T^{\pi}\hat{Q}^{\pi}$ داخل این فضا نباشد، لذا ناچار به تصویرسازی هستیم. تصویر متعامد $(\phi(\phi^T\phi)^{-1}\phi^T)$ که نرم L_2 $(\|Q - \Phi w\|_2)$ را کمینه می‌کند در نظر بگیرید. به دنبال تقریبِ تابع ارزش $\hat{Q}^{\pi}$ هستیم که تحت اعمال عملگر بلمن T^{π} متعاقب تصویرسازی متعامد، بدون تغییر باشد [۱۲]:

$$\begin{aligned} \hat{Q}^{\pi} &= \phi(\phi^T\phi)^{-1}\phi^T\left(T^{\pi}\hat{Q}^{\pi}\right) \\ &= \phi(\phi^T\phi)^{-1}\phi^T\left(\mathcal{R} + PB_{\pi}\hat{Q}^{\pi}\right) \end{aligned} \quad (۱۱)$$

[1] Approximate Policy Iteration
[2] Model based
[3] Basis Functions (Kernels)
[4] Full Rank

[DOI: 10.61186/joc.17.1.61]

باید توجه داشت که تصویرسازی متعامد در فضای ستونی ϕ خوش تعریف است، زیرا ستون‌های ϕ (توابع پایه) طبق تعریف، مستقل خطی هستند. با دستکاری تساوی بالا می‌توان عبارتی به دست آورد که می‌تواند سیستم خطی $k \times k$ که k تعداد توابع پایه است، به دست آورد:

$$\phi(\phi^T\phi)^{-1}\phi^T\left(\mathcal{R} + PB_\pi\hat{Q}^\pi\right) = \hat{Q}^\pi \xrightarrow{\hat{Q}^\pi=\phi\theta^\pi}$$
$$\underbrace{\phi^T(\mathcal{R} + PB_\pi\phi)}_{k\times k}\underbrace{\theta^\pi}_{k\times 1} = \underbrace{\phi^T\mathcal{R}}_{k\times 1} \quad (۱۲)$$

برای هر B_π ، حل این سیستم:

$$\theta^\pi = \left(\phi^T(\mathcal{R} + PB_\pi\phi)\right)^{-1}\phi^T\mathcal{R} \quad (۱۳)$$

به صورت تضمین شده برای همه مقادیر γ، به جز موارد محدودی، وجود دارد [۲۳]. چون تصویرسازی متعامد نرم L_2 را کمینه می‌کند، راه حل θ^π باعث تولید تابع ارزش $\hat{Q}^\pi$ می‌شود که آن را تقریب نقطه ثابت کمترین مربعات[1]، برای مقدار واقعی تابع ارزش می‌نامند. با تعریف $A = \left(\phi^T(\mathcal{R} + PB_\pi\phi)\right)^{-1}$ و $b = \phi^T\mathcal{R}$ معادله فوق می‌تواند بصورت $\theta^\pi = A^{-1}b$ بازنویسی شود.

در LSPI، برای مسأله یادگیری ماتریس A و بردار b را نمی‌توان از ابتدا تعیین کرد زیرا یا ماتریس P و بردار $\mathcal{R}$ ناشناخته هستند و یا آنقدر بزرگ هستند که در عمل نمی‌توان آن‌ها را در محاسبات به کار برد. لذا A و b را باید از روی نمونه‌ها یاد گرفت. برای مجموعه‌ای متناهی از نمونه ها، $D = \{(s_i, a_i, r_i, s_i', a_i')i = 1,2 \dots, L\}$ که در آن $a_i = \pi(s_i)$ و $s_i'\sim P(s_i, a_i, .)$ و $a_i' = \pi(s_i')$ و $s_{i+1} = s_i'$ و $a_{i+1} = a_i'$ ماتریس A و بردار b می‌توانند به صورت زیر تقریب زده شوند [۱۲]:

$$\hat{A}' = \hat{A} + \phi(s_i, a_i)\left(\phi(s_i, a_i) - \gamma\phi(s_i', a_i')\right)^T \quad (۱۴)$$

$$\hat{b}' = \hat{b} + \phi(s_i, a_i)r_i \quad (۱۵)$$

همچنان که شرح داده شد پیاده‌سازی و استفاده عملی از LSPI مشروط به تعریف توابع پایه مناسب جهت تقریب تابع ارزش- عمل می‌باشد. اما در روش مذکور به این سوال که چگونه یک مجموعه مناسب از توابع پایه باید انتخاب شوند تا تقریب مناسبی از تابع ارزش عمل صورت پذیرد پرداخته نشده است (هیچ ایده‌ای داده نشده است). همچنین در اکثر مقاله‌های مرتبط در این زمینه، چگونگی تعریف توابع پایه مشخص نشده است. در بخش بعد یک روش ترکیبی که آن را تکرار سیاست کمترین مربعات عصبی نامیده می‌شود معرفی شده است. ایده بنیادی روش، بهره‌گیری از توابع پایه شعاعی در شبکه RBF جهت رفع چالش تعریف توابع پایه مناسب مورد نیاز در LSPI است.

۳- تکرار سیاست کمترین مربعات عصبی[2]

شبکه‌های عصبی بر پایه توابع شعاعی (RBF) به طور گسترده برای تخمین غیر پارامتریک توابع چند بعدی ازطریق مجموعه‌ای محدود از اطلاعات آموزشی به کار می‌روند. این نوع شبکه‌ها با آموزش سریع و فراگیر، ساختار ساده و کارآمدی بالا در مسائل تخمین تابع بسیار مورد توجه هستند [۳]. هارتمن و کیلر در سال ۱۹۹۰ میلادی ثابت کردند که شبکه‌های RBF، تقریب زننده‌های بسیار قدرتمندی هستند؛ به‌طوری‌که با داشتن تعداد نرون‌های کافی در لایهٔ مخفی، قادر به تقریب‌زدن هر تابع پیوسته و با هر درجه دقت می‌باشند [۲۴]. در این شبکه لایه پنهان، یک انطباق غیرخطی مابین فضای ورودی و یک فضا معمولاً با بعد بزرگتر برقرار می‌کند که در آن الگوها به صورت تفکیک‌پذیر خطی در می‌آیند. خصوصیت منحصر به فرد RBF پردازشی است که در لایه پنهان انجام می‌گیرد. ایده اصلی آن است که الگوهای فضای ورودی تشکیل خوشه دهند. به این صورت تابع غیرخطی به صورت تابع شناخته شده شعاع مدار در می‌آید. شبکه عصبی توابع پایه شعاعی نرمالیزه شده[3] (NRBF) به وسیله نرمالیزه کردن توابع پایه با جمع جبری خروجی کل توابع پایه حاصل می‌شود. استفاده از شبکه NRBF در یادگیری تقویتی به واسطه‌ی حذف تأثیرات محدوده‌های توابع پایه و عملکرد بهتر در تقریب توابع، مناسب‌تر است [۲۵]. در این مقاله ساختار تقریب زننده عصبی مطابق با شکل (۳) که در آن از یک شبکه عصبی NRBF چهار لایه با یک نرون در لایه‌ی خروجی برای تقریب تابع ارزش عمل، استفاده شده است. لایه‌ی ورودی شبکه شامل $n+1$ نرون می‌باشد. $x(t) \in \mathbb{R}^{n+1}$ که متشکل از جفت حالت و عمل‌های مسأله است. که در آن $s_t = (s_1, s_2, \dots, s_n)^T \in \mathbb{R}^n$ بردار حالت‌های مسأله می‌باشد و همچنین $u \in A$ که $A = [a_1, a_2, \dots, a_m]$ بردار عمل‌های مسأله است. لایه‌ی ورودی، بردار ورودی $x(t) \in \mathbb{R}^{n+1}$ را به هر یک از گره‌های لایه‌ی پنهان ارسال می‌کند. با توجه به اینکه شبکه‌های توابع پایه شعاعی بر تعریف تابعی وابسته به فاصله از بردار مراکز استوار هستند لذا برای یک الگوی ورودی x خروجی j-امین گره $\varphi_j: \mathbb{R}^{n+1} \to \mathbb{R}$ در لایه‌ی پنهان با توابع پایه شعاعی ρ برابر است با:

$$\varphi_j(x) = \rho\left(\left\|x - c_j\right\|_2\right), j = 1,2, \dots, h \quad (۱۶)$$

که در آن c_j را بردار مراکز[4] توابع فعال‌ساز نرون‌های لایه پنهان می‌نامند:

$$c_j = \left[c_{1j}, c_{2j}, \dots, c_{(n+1)j}\right]^T_{1\times(n+1)}, j = 1,2, \dots, h$$
$$= \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1h} \\ c_{21} & c_{22} & \cdots & c_{2h} \\ \vdots & \vdots & \ddots & \vdots \\ c_{(n+1)1} & c_{(n+1)2} & \cdots & c_{(n+1)h} \end{bmatrix}_{(n+1)\times h} \quad (۱۷)$$

[1] Least Square fixed-point approximation
[2] Neural Least Square Policy Iteration (NLSPI)
[3] Normalized RBF (NRBF)
[4] Center vector

[DOI: 10.61186/joc.17.1.61]

تعداد کل نرون‌های لایه‌ی پنهان نیز h است. همچنین $\|.\|_2$ نشان دهنده نرم اقلیدسی[1] (L_2) است: $\|\ell\|_2 := \left(\sum_i \ell_i^2\right)^{\frac{1}{2}} = \sqrt{\sum_i \ell_i^2}$.

در حالت خاص (مورد استفاده در این مقاله) برای یک شبکه عصبی RBF با توابع پایه گوسی خواهیم داشت:

$$\rho\left(\left\|x - c_j\right\|_2\right) = exp\left(-\frac{1}{2}(x-c_j)^T \sigma_j^{-1}(x-c_j)\right) = exp\left(-\frac{\left\|x-c_j\right\|_2^2}{2{\sigma_j}^2}\right),\ j = 1,2,\dots,h \quad (۱۸)$$

به طوری که:

$$\lim_{\|x\|\to\infty} \rho\left(\left\|x - c_j\right\|_2\right) = 0\ ,\ j = 1,2,\dots,h \quad (۱۹)$$

σ_j مشخص کننده مقادیر انحراف استاندارد[2] (سطح گسترش) توابع فعال‌ساز گوسی نرون jام لایه پنهان شبکه می‌باشد:

$$\sigma_j = \left[\sigma_{1j}, \sigma_{2j}, \dots, \sigma_{(n+1)j}\right]^T_{1\times(n+1)},\ j = 1,2,\dots,h = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1h} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2h} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{(n+1)1} & \sigma_{(n+1)2} & \cdots & \sigma_{(n+1)h} \end{bmatrix}_{(n+1)\times h} \quad (۲۰)$$

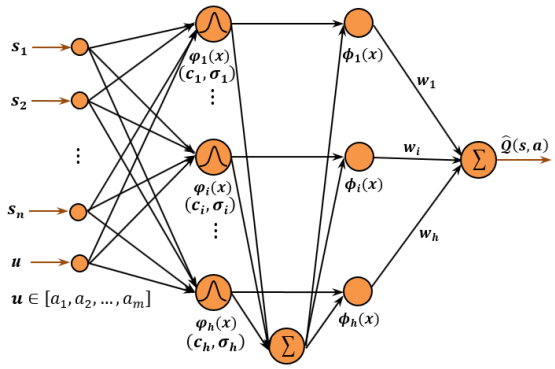
نرمالیزه سازی در لایه‌ی سوم به صورت زیر انجام می‌گیرد:

$$\phi_j(x) \stackrel{\text{def}}{=} \frac{\rho\left(\left\|x - c_j\right\|_2\right)}{\sum_{k=1}^{h} \rho_k\left(\left\|x - c_j\right\|_2\right)},\ j = 1,2,\dots,h \quad (۲۱)$$

خروجی شبکه $\hat{Q}: \mathbb{R}^{n+1} \to \mathbb{R}$ با استفاده از حاصل‌جمع وزندار خطی پاسخ‌های لایه‌ی پنهان در گره‌ی خروجی بدست می‌آید:

$$\hat{Q}_t(s_t, a_t; w) = \sum_{j=1}^{h} \phi_j\left(\left\|x - c_j\right\|_2\right) w_j(t) \quad (۲۲)$$

$$= \sum_{j=1}^{h} \phi_j(x) w_j(t)\ , j = 1,2,\dots,h \quad (۲۳)$$



شکل ۳: شبکه عصبی NRBF به عنوان تقریب زننده تابع ارزش عمل

که $w_1, w_2, \dots, w_h$ پارامترهای وزن شبکه عصبی و $\phi_1, \phi_2, \dots, \phi_h$ توابع پایه ارزش حالت- عمل نرمالیزه شده روی فضای حالت- عمل می‌باشند [۳].

رابطه‌ی (۲۳) را می‌توان به فرم زیر به صورت برداری نوشت:

$$\hat{Q}_t(s_t, a_t; w) = \phi^T(s,a) w_t \quad (۲۴)$$

که $\phi^T(s,a) = [\phi_1(s,a), \dots, \phi_h(s,a)]$ ترانهاده ماتریس $\phi(s,a)$ می‌باشد. ماتریس فضای حالت- عمل Φ با ابعاد $|S| \times |A| \times k$ را به صورت زیر تعریف می‌کنیم:

$$\Phi = \begin{bmatrix} \phi_1(s_1,a_1) & \cdots & \phi_h(s_1,a_1) \\ \phi_1(s_1,a_2) & \cdots & \phi_h(s_1,a_2) \\ \vdots & & \\ \phi_1(s_1,a_{|A|}) & \cdots & \phi_h(s_1,a_{|A|}) \\ \vdots & & \\ \phi_1(s_{|S|},a_1) & \cdots & \phi_h(s_{|S|},a_1) \\ \phi_1(s_{|S|},a_2) & \cdots & \phi_h(s_{|S|},a_2) \\ \vdots & & \\ \phi_1(s_{|S|},a_{|A|}) & \cdots & \phi_h(s_{|S|},a_{|A|}) \end{bmatrix} \quad (۲۵)$$

$$= \begin{bmatrix} | & & | \\ \phi_1(s,a) & \dots & \phi_h(s,a) \\ | & & | \end{bmatrix} \quad (۲۶)$$

که S مجموعه حالت‌ها A مجموعه عمل‌ها است. همچنین $\phi_i(s_j, a_k)$ خروجی نرمالیزه شده i- امین گره RBF است.

بر اساس تئوری michelli و همچنین از آنجا که بردار مراکز و پارامترهای گستردگی توابع RBF متمایز هستند، به سادگی می‌توان استدلال نمود که ماتریس Φ دارای مرتبه کامل بوده و یا به عبارت دیگر توابع پایه ارزش عمل در الگوریتم NLSPI مستقل خطی هستند.

لذا داریم:

$$\hat{Q}_t = \Phi w_t \quad (۲۷)$$

بردار مراکز c_j برای k_i تابع پایه گوسی تعریف شده برای ورودی x_i به صورت زیر تنظیم می‌گردند:

[1] Euclidean
[2] Spread

[DOI: 10.61186/joc.17.1.61]

$$c_{i1} = x_{i_{min}}, i = 1,2,\dots,n+1 \quad (۲۸)$$

$$c_{ir} = c_{i(r-1)} + \Delta c_i \,, for\ r = 2,3,\dots,k_i \quad (۲۹)$$

که Δc_i فاصله بین دو مرکز مجاور می‌باشد و بر طبق رابطه زیرتعیین می‌گردد:

$$\Delta c_i = \frac{x_{i_{max}} - x_{i_{min}}}{k_i - 1}, i = 1,2,\dots,n+1 \quad (۳۰)$$

با این توصیف، در هر مسأله از $h = k_1 \times k_2 \times \dots \times k_{n+1}$ تابع پایه حالت- عمل بر روی فضای $n+1$ بعدی فضای حالت- عمل برای تقریب تابع ارزش عمل استفاده می‌گردد.

در این مقاله پارامترهای انحراف استاندارد σ_j نیز با استفاده از رابطه زیر تنظیم می‌گردند:

$$\sigma_i = \frac{x_{i_{max}} - x_{i_{min}}}{\sqrt{2k_i}}, i = 1,2,\dots,n+1 \quad (۳۱)$$

لذا با این تعریف σ به فرم زیر تبدیل خواهد شد:

$$\sigma = \left[\sigma_1, \sigma_2, \dots, \sigma_{(n+1)}\right]^T_{1\times(n+1)} = \left[\frac{d_{1_{max}}}{\sqrt{2k_1}}, \frac{d_{2_{max}}}{\sqrt{2k_2}}, \dots, \frac{d_{n+1_{max}}}{\sqrt{2k_{n+1}}}\right]^T_{1\times(n+1)} \quad (۳۲)$$

در شکل (۴) نحوه تعیین بردار مراکز و انحراف معیار مربوط به ورودی x_i نشان داده شده است.

هدف آموزش، به روز رسانی روی خط مقادیر وزن شبکه عصبی w به گونه‌ای است که بهترین تقریب از تابع ارزش عمل صورت پذیرد.

رابطه به روز رسانی پارمترهای وزن شبکه عصبی به صورت زیر است:

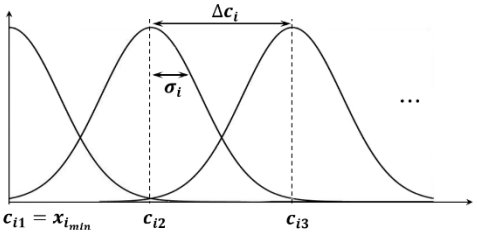
$$\frac{1}{l-1}W_l = \frac{1}{l-1}A_t^{-1}b_t \quad (۳۳)$$

که در آن ماتریس A و بردار b به صورت زیر به روز رسانی می‌شوند:

$$A_{t+1} = A_t + \vec{e}_{t+1}\left(\phi(s_t,a_t) - \gamma\phi\left(s_{t+1},\pi(s_{t+1})\right)\right)^T \quad (۳۴)$$

$$b_{t+1} = b_t + \vec{e}_{t+1} r_t \quad (۳۵)$$

$\vec{e}_t$ پارامتر "شایستگی مسیر" است که تاریخچه‌ای از وضعیت عمل‌های انتخاب شده و میزان تغییرات ارزش آنها را نسبت به هر ارزشی که عامل در آینده بدست می‌آورد، مشخص می‌کند و برای الگوریتم NLSPI به صورت زیر محاسبه می‌گردد:



شکل ۴: بردار مراکز و انحراف معیار مربوط به ورودی x_i

$$\begin{aligned}\vec{e}_t &= \sum_{k=1}^{t}(\lambda\gamma)^{t-k}\frac{\partial Q(s_t,a_t,w_t)}{\partial w(k)} \\ &= \sum_{k=1}^{t-1}(\lambda\gamma)^{t-k}\frac{\partial Q(s_t,a_t,w_t)}{\partial w(k)} + \frac{\partial Q(s_t,a_t,w_t)}{\partial w(t)} \\ &= \gamma\lambda\vec{e}_{t-1} + \frac{\partial Q(s_t,a_t,w_t)}{\partial w(t)} \\ &= \gamma\lambda\vec{e}_{t-1} + \phi_j(x_t)\end{aligned} \quad (۳۶)$$

پر واضح است که می‌توان از رابطه فوق فرمول زیر را مستقیماً نتیجه گرفت:

$$\vec{e}_{t+1} = \gamma\lambda\vec{e}_t + \phi_j(x_{t+1}) \quad (۳۷)$$

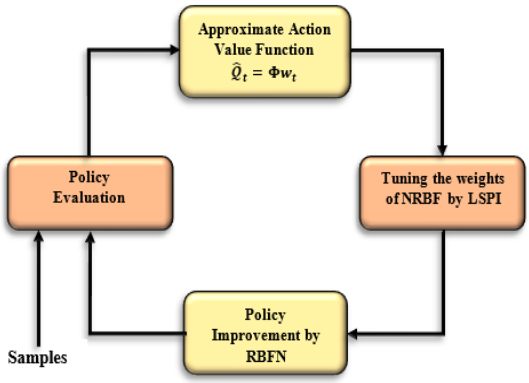
در این رابطه γ فاکتور نزول و λ ضریب تعقیب در بازه $[0,1]$ است. مقدار اولیه $\vec{e}_t$ بردار صفر در نظر گرفته می‌شود. روش مسیرهای شایستگی علاوه بر آنکه نگاه ترکیبی مونت کارلو و تفاوت گذرا را در بروزرسانی ارزش جفت حالت- عمل دارا می‌باشد، دارای نگاه به عقب نیز می‌باشد. نگاه دوسویه بدین معنا می‌باشد که برای بروزرسانی ارزش جفت حالت- عمل فعلی، نه تنها ارزش حالت- عمل در لحظات بعد تاثیرگذار است، بلکه ارزش کل زنجیره حالت و عمل های قبلی نیز در بروزرسانی آن تاثیرگذار خواهد بود.که این موضوع باعث بهبود سرعت و دقت همگرایی می‌شود.

به منظور کاهش حجم محاسبات در الگوریتم NLSPI، می‌توان با بهره‌گیری از تساوی شرمن- موریسون به جای به روز رسانی ماتریس A و سپس محاسبه معکوس ماتریس، به یکباره معکوس ماتریس Aرا به روز رسانی کرد. رابطه شرمن- موریسون برای محاسبه وارون حاصل جمع یک ماتریس وارون پذیر و حاصل ضرب خارجی دو بردار u و v به صورت زیر است:

$$(A+uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u} \quad (۳۸)$$

با تعریف:

[DOI: 10.61186/joc.17.1.61]



شکل۵: بلوک دیاگرام تقریب سیاست کمترین مربعات عصبی

$$u = \phi(s_t, a_t) \quad (۳۹)$$

$$v = \vec{e}_{t+1}\left(\phi^T(s_t, a_t) - \gamma\phi^T(s_{t+1}, a_{t+1})\right) \quad (۴۰)$$

رابطه به روز رسانی معکوس ماتریس A به صورت رابطه زیر در می‌آید:

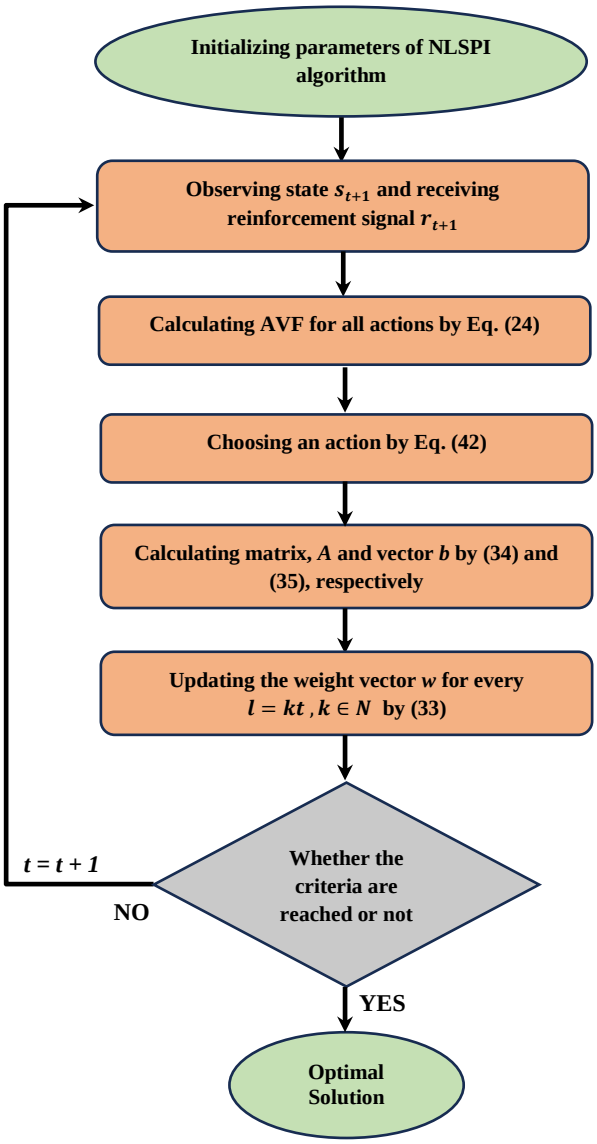
$$A_{t+1}^{-1} = A_t^{-1} - \frac{A_t^{-1}uv^TA_t^{-1}}{1 + v^TA_t^{-1}u} \quad (۴۱)$$

در این مقاله برای انتخاب عمل از رابطه‌ی بیشینه هموار به فرم زیر استفاده شده است:

$$\pi_w^\tau(s_t, a_t = a_k) = \frac{exp\,(\hat{Q}(s_t, a_k)/\tau)}{\sum_{b\epsilon A} exp\,(\hat{Q}(s_t, b)/\tau)} = \frac{1}{\sum_{b\epsilon A} exp\left(\frac{\hat{Q}(s_t, b) - \hat{Q}(s_t, a_k)}{\tau}\right)} \quad (۴۲)$$

که در آن $\tau > 0$ ضریب دما[1] نامیده می‌شود. معمولاً در ابتدای آموزش مقدار ضریب دما بزرگ و در حین آموزش هر چه به سمت جلو می رویم مقدار ضریب دما کاهش می‌یابد تا از تجربیات قبلی بیشتر استفاده گردد. از آنجا که در رابطه (۴۲) عمل‌ها بر طبق تابع ارزش عمل تقریب زده شده حاصل می‌گردند، احتمال انتخاب عمل به τ و w وابسته است که برای تاکید بر این موضوع در اندیس π آورده شده‌اند. تابع توزیع احتمال عمل در فرمول بیشینه هموار از توزیع بولتزمن[2] که یک توزیع پیوسته است، تبعیت می‌کند. این ویژگی یک خاصیت مهم از لحاظ همگرایی در یادگیری تقویتی پیوسته می‌باشد [۵]. بلوک دیاگرام الگوریتم NLSPI در شکل (۵) و همچنین رویه‌ی اجرای یادگیری بر اساس الگوریتم برخط NLSPI در شکل (۶) نشان داده شده است.

لازم به یادآوری است که به روز رسانی بردار w در واقع به روز رسانی $\hat{Q}$ می‌باشد و منجر به استخراج سیاست جدید می‌شود.



شکل ۶: بلوک دیاگرام رویه اجرای الگوریتم NLSPI

٤- شبیه‌سازی

برای ارزیابی کارایی استراتژی آموزشی ارائه شده، عملکرد الگوریتم پیشنهادی با روش‌های دیگر در حل مسائل حالت پیوسته‌ی معروف بالا رفتن ماشین از تپه و ربات آکروبات مورد مقایسه قرار می‌گیرد.

۴-۱- مسأله بالا رفتن ماشین از تپه

بالا رفتن ماشین از تپه[3] یکی از مسائل معروف در زمینه یادگیری تقویتی است، که در مقالات مختلف [۲۷، ۲۶] و همچنین مسابقات علمی مرتبط در این زمینه جهت ارزیابی الگوریتم‌های یادگیری تقویتی از آن استفاده شده است [۲۹]. اتومبیلی را در سراشیبی یک تپه مانند شکل (۷) در نظر بگیرید. هدف راندن ماشین از هر نقطه دره مشرف به تپه به بالای

[1] Temperature factor
[2] Boltzman Distribution
[3] Mountain car problem

[DOI: 10.61186/joc.17.1.61]

آن در کوتاه‌ترین زمان ممکن می‌باشد، اما از آنجا که نیروی محرک موتور ماشین توان لازم جهت عبور از سطح شیب‌دار جاده را ندارد تنها راه حل ممکن برای موفقیت در این شرایط انجام متوالی عقبگرد از شیب سمت چپ و دور شدن از هدف و حرکت رو به جلو تا زمانی است که اینرسی لازم جهت غلبه بر شتاب گرانش و عبور از شیب سمت راست جاده حاصل شود. این مسأله یک مثال از بکارگیری یک وظیفه کنترل پیوسته است که از لحاظی ممکن است قبل از اینکه بهتر شود، بدتر شود. چون باید قبل از رسیدن به هدف از آن فاصله بگیرد. بسیاری از روش‌شناسی‌های کنترلی، مشکلات بزرگی با این قبیل مسائل دارند مگر اینکه شهود طراح به آنها کمک کند. متغیرهای ورودی در این مسأله، موقعیت ماشین (x) و سرعت آن (v) می‌باشند. هدف از یادگیری، تنظیم روی خط وزن‌های یک کنترلگر عصبی به گونه‌ای است که بتواند نیروی کنترلی لازم (F) جهت راندن اتومبیل به بالای تپه از هر موقعیت و سرعت اولیه را در حداقل زمان فراهم نماید.

۱-۱-۴- دینامیک سیستم

معادلات حرکت ماشین به صورت زیر است [۲، ۲۷]:

$$v_{t+1} = \min\Big(0.07, \max\big(-0.07, v_t + 0.001 \times F_t + \mathrm{g} \times \cos(3x_t)\big)\Big)$$
$$x_{t+1} = \min\big(0.5, \max(-1.2, x_t + v_{t+1})\big) \quad (۴۳)$$

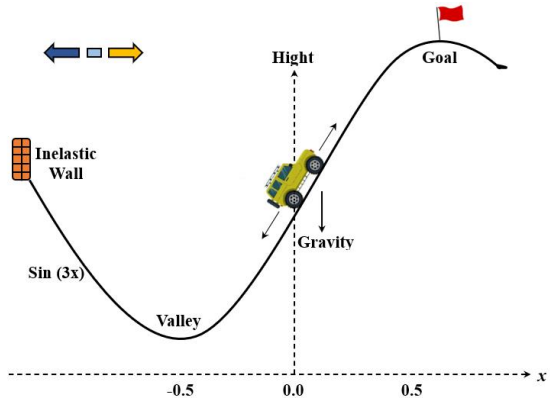
که در آن $g = 0.0025$ مقداری ثابت و $F \in \{-1,0,1\}$، کل عمل‌های ممکن قابل انجام در این مسأله می‌باشد. محدوده تغییرات موقعیت ماشین (x) و سرعت آن (v) به صورت زیردر نظر گرفته می‌شوند:

$$\{(x, v \in \mathbb{R}^2) | -1.2 \le x \le 0.5, -0.07 \le v \le 0.07\} \quad (۴۴)$$

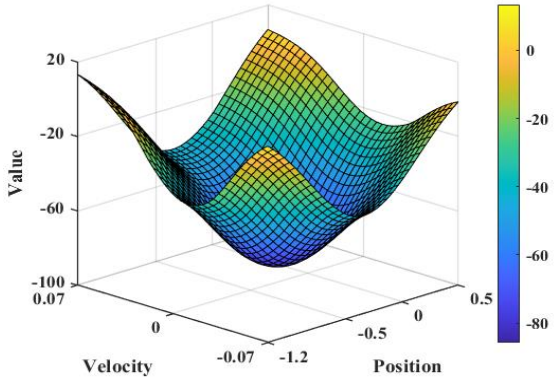
اگر ماشین به نقطه‌ی انتهای محدوده سمت چپ خود ($x_t = -1.2$) برسد، سرعت آن با صفر مقداردهی می‌شود و رسیدن به نقطه انتهایی محدوده سمت راست ($x_t = 0.5$) هدف عامل بوده و به عنوان موفقیت در نظر گرفته می‌شود. معادله دینامیکی حاکم بر مسیر حرکت ماشین نیز $\sin(3x)$ در نظر گرفته می‌شود.

۲-۱-۴- جزئیات آموزش

در این مسأله ورودی‌های شبکه عصبی $x = [s,u]^T = [x,v,u]^T$ متشکل از موقعیت و سرعت عامل $s \in [x,v]$ باضافه تک تک عمل‌ها $u \in A = [-1,0,+1]$ در آن حالت می‌باشند. به منظور تقریب تابع ارزش عمل از $3 \times 3 \times 3$ تابع پایه حالت و عمل بر روی کل فضای حالت- عمل مسأله استفاده شده است.



شکل ۷: شماتیک مسأله بالا رفتن ماشین از تپه



شکل ۸: ارزشِ عمل با بالاترین ارزش در

NLSPI

همچنین $\lambda = 0.99$ و نرخ نمونه‌برداری 0.02 ثانیه در نظر گرفته شده است. در این مسأله در هر قدم زمانی عامل پاداش ۱- دریافت می‌کند و زمانی که به هدف برسد پاداش صفر می‌گیرد [۲۸]. نتایج هر آزمایش متوسط ارزیابی انجام شده در ۱۰ اجرا است. در ابتدای هر اجرا مقادیر وزن‌های شبکه عصبی با مقدار اولیه صفر مقداردهی می‌شوند. اگر تعداد رویدادها از ۳۰۰ بیشتر شود یا تعداد دفعات متوالی که عامل به هدف رسیده به ۴۰ برسد بخش آموزش پایان می‌یابد. ضریب دما مطابق با رابطه زیر در هر مرحله آموزش کاهش می‌یابد [۵]:

$$\tau_t = \begin{cases} \tau_{t-1} - (0.99)^t \times 0.4 \times \tau_t & if\ t = 4k, t < 25 \\ \tau_{t-1} - (0.99)^t \times 0.2 \times \tau_t & if\ t = 4k, t \ge 25 \\ \tau_{t-1} & otherwise \end{cases} \quad (۴۵)$$

که $t \ge 1$ شماره رویداد می‌باشد. همچنانکه از رابطه (۴۵) مشخص است مقدار ضریب دما در طول هر رویداد ثابت است و در شروع هر رویداد طبق رابطه مذکور مقدار آن تعیین می‌گردد.

[DOI: 10.61186/joc.17.1.61]

جدول ۲: مقایسه نتایج شبیه‌سازی در مسأله راندن ماشین به بالای تپه

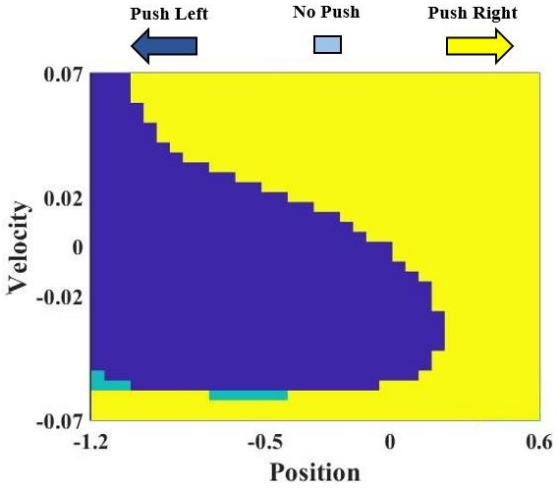
Method	NLSPI	LSPI	KLSPI	RPI	RBLSPI	REPS	CMA-ES	DDPG	DDQN
Mean Return	-83.6	-202.3	-179.1	-173.8	-110	-275.6	-85	-288.4	-135.7

۳-۱-۴- نتایج شبیه‌سازی

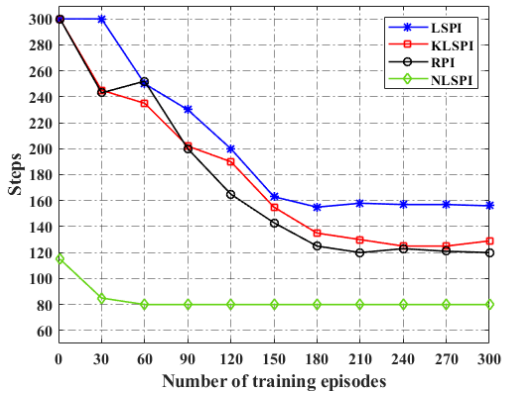
شکل (۸) منحنی ارزشِ عمل با بالاترین ارزش $\max_a \left(\hat{Q}(s,a)\right)$ تولید شده توسط الگوریتم NLSPI را نشان می‌دهد. همانطور که از شکل مشخص است تابع ارزش تقریب‌زده شده به اندازه کافی رضایت بخش است که بتوان سیاست مناسب جهت رسیدن به هدف را، از طریق آن استخراج کرد. شکل (۹) نیز سیاست یادگرفته شده توسط عامل بر روی کل فضای ورودی را نشان می‌دهد. در جدول (۲) نتیجه حاصل از بکارگیری الگوریتم NLSPI در حل مسأله راندن ماشین به بالای تپه، بر حسب معیار میانگین بازده کسب شده در تمام تکرارهای آموزشی با بهترین نتایج حاصل شده از روش‌های LSPI، KLSP، RPI، RBLSPI، REPS، CMA-ES، DDPG[1] و DDQN که در مقالات [۲۷-۲۵] گزارش شده‌اند مقایسه شده است. همچنانکه از نتایج شبیه‌سازی مشخص است NLSPI با اختلاف بسیار مناسب، عملکرد بهتری نسبت به بقیه داشته است. همچنین در شکل (۱۰) عملکرد سه الگوریتم از خانواده LSPI با الگوریتم NLSP مقایسه شده است. همانگونه که از شکل مشخص است NLSPI عملکرد بسیار بهتری نسبت به روش‌های هم خانواده خود از نظر سرعت همگرایی و همچنین سیاست نهایی که بدان همگرا شده، داشته است.

۲-۴- مسأله آکروبات

از جمله مسائل اساسی کنترل غیرخطی، مسأله تاب خوردن به بالا[2] است که آکروبات[3] در زمره چنین مسائلی جای می‌گیرد. آکروبات در واقع یک ربات صفحه‌ای دو لینکی ($l_i, i = 1,2$) با دو درجه آزادی است که دارای دو مفصل چرخشی و یک عملگر در محل تقاطع بازوهای ربات می‌باشد (شکل۱۱). با توجه اینکه حرکات ربات از نظر فیزیکی رفتار یک ژیمناستیک آکروبات باز را مدل می‌کند به ربات آکروبات معروف شده است [۳]. در این مسأله هدف رسیدن ربات به نقطه تعادل ناپایدار می‌باشد (موقعیت (۵) در شکل (۱۲)). برای این منظور ربات باید حالت تاب خوردن گرفته تا بتواند انرژی لازم را برای رسیدن به وضعیت عمودی نامتعادل در بالای میله در حالی که کماکان به همان نقطه اولیه متصل است به دست آورد.



شکل ۹: سیاست یادگرفته شده توسط NLSPI بر روی کل فضای ورودی

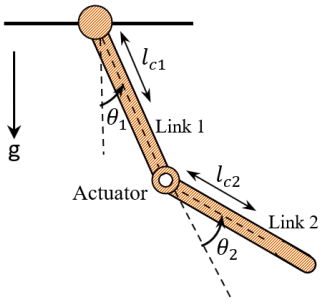


شکل ۱۰: مقایسه عملکرد الگوریتم‌های LSPI، KLSPI، RPI و NLSPI

آکروبات بدلیل نداشتن عملگر در مفصل اول، در زمره سیستم‌های مکانیکی زیرکارانداز[4] مرتبه یک قرار دارد زیرا تعداد عملگرهای آن از تعداد درجه آزادی‌اش باندازه‌ی یک واحد کمتر است. از جمله خصوصیات این مسأله می‌توان به دینامیک پیچیده داخلی، رفتار غیرهولونومیک[5] و غیرقابل خطی‌سازی بودن فیدبک اشاره کرد. به طور کلی، نظریه جامعی برای کنترل ربات‌های زیرکارانداز همچون آکروبات وجود ندارد. لذا با توجه به این خصوصیات، این مسأله از نقطه نظر کنترلی، موردی غنی و حائز اهمیت تلقی می‌شود که در دهه‌های اخیر

[1] Deep Deterministic Policy Gradient
[2] Swing-Up Control
[3] Acrobot
[4] Underactuated
[5] Non-Holonomic

[DOI: 10.61186/joc.17.1.61]



شکل ۱۱: مسأله آکروبات

حجم گسترده‌ای از پژوهش‌های علم کنترل را به خود جلب کرده است [۳۰-۳۲]. فرض بر این است که در ابتدا آکروبات در نقطه تعادل پایدار قرار دارد (موقعیت (۱) در شکل (۱۲))، عامل باید به استراتژی دست یابد که تنها با اعمال گشتاور به مفصل دوم ربات و بوسیله تاب دادن‌های متوالی، انرژی آکروبات را به گونه‌ای افزایش دهد که ربات به نیم صفحه بالایی رفته و به نقطه تعادل ناپایدار خود برسد. متغیرهای حالت مسأله، شامل زاویای دو لینک و مشتقات‌آنها می‌باشد. در این مسأله سرعت زاویه‌ای بازوی اول به بازه‌ی $\dot{\theta}_1 \in [-4\pi, 4\pi]$ و سرعت زاویه‌ای بازوی دوم به بازه‌ی $\dot{\theta}_2 \in [-9\pi, 9\pi]$ محدود شده‌اند [۶]. شکل (۱۲) نمایی از چند حرکت نوعی آکروبات برای رسیدن از نقطه تعادل پایدار به نقطه تعادل ناپایدار را نشان می‌دهد.

۴-۲-۱- دینامیک سیستم

معادلات دینامیکی حاکم بر سیستم ربات آکروبات، عبارتند از [۳، ۳۱]:

$$\ddot{\theta}_1 = -{d_1}^{-1}\left(d_2\ddot{\theta}_2 + \phi_1\right) \quad (۴۶)$$

$$\ddot{\theta}_2 = \left(m_2{l_{c2}}^2 + I_2 - \frac{{d_2}^2}{d_1}\right)^{-1}\left(\tau + \frac{d_2}{d_1}\dot{\phi}_1 - \dot{\phi}_2\right) \quad (۴۷)$$

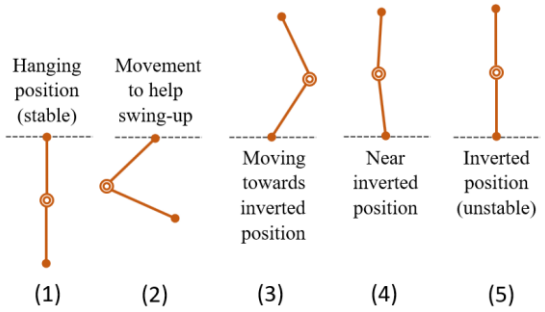
که در آن:

$$d_1 = m_1{l_{c1}}^2 + m_2\left({l_1}^2{l_{c2}}^2 + 2l_1l_2cos\theta_2\right) + I_1 + I_2 \quad (۴۸)$$

$$d_2 = m_2\left({l_{c2}}^2 + l_1l_{c2}cos\theta_2\right) + I_2 \quad (۴۹)$$

$$\dot{\phi}_1 = -m_2l_1l_{c2}{\dot{\theta}_2}^2 sin\theta_2 - 2m_2l_1l_{c2}\dot{\theta}_2\dot{\theta}_1 sin\theta_2 + (m_1l_{c1} + m_2l_1)gcos(\theta_1 - \pi/2) + \dot{\phi}_2 \quad (۵۰)$$

$$\dot{\phi}_2 = m_2l_{c2}gcos(\theta_1 + \theta_2 - \pi/2) \quad (۵۱)$$



شکل ۱۲: نمایی از حرکات ربات آکروبات برای رسیدن به هدف

جدول ۳: پارامترهای مسأله آکروبات شبیه‌سازی شده

Model Parameter	Symbol	Value	Unit
Link lengths	l_1, l_2	1.0	m
Joint to mass center	lc_1, lc_2	0.5	m
Link masses	m_1, m_2	1.0	kg
Link inertias	I_1, I_2	1.0	$kg.m^2$
Torque range	τ	$\{-1, 0, 1\}$	$N.m$
Gravitational force	g	9.8	m/s^2
Integration time step	t_i	0.05	s
Control time step	t_c	0.2	s

شایان ذکر است معادلات فوق تنها جهت شبیه‌سازی مورد استفاده قرار گرفته‌اند و از دید عامل (بخش کنترل) دینامیک سیستم ناشناخته می‌باشد.

۴-۲-۲- جزئیات آموزش

در این مسأله ورودی‌های شبکه عصبی $x = [s, u]^T$ متشکل از زوایای دو لینک و مشتقات آنها $s = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]$ باضافه‌ی تک تک عمل‌ها $u \in A = [-1, 0, +1]$ در آن حالت می‌باشد. همچنین به منظور تقریب تابع ارزش عمل در الگوریتم NLSPI از ۳ تابع پایه حالت- عمل بر روی هریک از متغییرهای حالت- عمل ورودی استفاده شده است لذا بر روی کل فضای حالت- عمل مسأله $3 \times 3 \times 3 \times 3 \times 3$ تابع پایه حالت- عمل خواهیم داشت.

از آنجا که هدف کنترلی در این مسأله رسیدن به نقطه تعادل ناپایدار در حداقل زمان می‌باشد سیگنال تقویتی به صورت زیر تعریف می‌گردد:

$$r_t = \begin{cases} 0 & if\ \ s = s_T \\ -1 & otherwise \end{cases} \quad (۵۲)$$

که در آن s_T بیانگر حالت نهایی در نقطه تعادل ناپایدار است.

پارامترهای ضریب دما و λ مانند مثال قبل تنظیم می‌گردند. همچنین به منظور ارزیابی عادلانه تمام شرایط ذکر شده در تنظیم پارامترهای الگوریتم NSL نیز لحاظ می‌گردد باضافه اینکه نرخ یادگیری در روش NSL مطابق با رابطه زیر در هر مرحله از آموزش تنظیم می‌شود [۳]:

$$\eta_t = \begin{cases} \eta_{t-1}/1.001 & \text{if } t = 4k \\ \eta_{t-1} & \text{otherwise} \end{cases} \quad (۵۳)$$

برای مشاهده تأثیر کنترلِ اعمالی، کنترل‌کننده و مکانیسم آموزش در هر

[DOI: 10.61186/joc.17.1.61]

جدول ۳: مقایسه نتایج شبیه‌سازی

Initial Parameters		Method	Avg. LDI	Std. (LDI)	Failure Rate	LE	Avg. time (Sec)
$T_0 = 0.1$	$\eta_0 = 0.1$	NSL	39.26	12.8	1.7	75.42	149.2
$T_0 = 0.01$	---------	NLSPI	32.5	6.31	0.6	59.7	112.36

LDI, Learning duration index; LE, Length of Episode; NSL, Neural Sarsa Learning; NLSPI, Neural Least Square Policy Iteration

۰/۲ ثانیه تحریک می‌شوند. تعریف کلیه پارامترهای استفاده شده و همچنین مقادیری که در شبیه‌سازی برای آنها در نظر گرفته شده در جدول (۳) آمده است. نتایج هر آزمایش متوسط ارزیابی انجام شده در ۳۰ اجرا است. در ابتدای هر اجرا مقادیر وزن‌های شبکه عصبی با مقدار اولیه صفر مقداردهی می‌شوند.

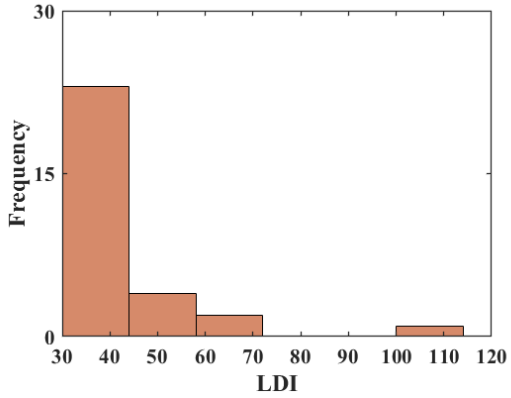
اگر تعداد رویدادها[1] از ۲۰۰ بیشتر شود یا تعداد دفعات متوالی که عامل به هدف رسیده به ۳۰ برسد بخش آموزش پایان می‌یابد. شماره رویداد در پایان بخش آموزش به عنوان "معیار زمان آموزش" (LDI)[2] در نظر گرفته می‌شود. هر رویداد از نقطه تعادل پایدار آغاز می‌شود و زمانی پایان می‌پذیرد، که یا عامل به هدف برسد و یا تعداد قدم‌های انجام شده توسط عامل در یک رویداد از ۱۰۰ بیشتر شود. انحراف معیار رویدادها به عنوان "معیار انحراف معیار" ((Std.(LDI)[3] در نظر گرفته می‌شود. تعداد قدم‌های انجام شده در رسیدن به هدف در یک رویداد به عنوان معیار "تعداد قدم زمانی رسیدن به هدف" (LE)[4] در نظر گرفته می‌شود. درصد شکست عامل در رسیدن به هدف به عنوان "معیار نرخ شکست[5]" لحاظ می‌گردد.

همچنین میانگین زمان اجرا ((Avg. (time) محاسبه می‌گردد. سیستم کامپیوتری مورد استفاده جهت شبیه‌سازی دارای پردازشگر Intel core i3 (2.20 GH) و ۴ گیگابایت حافظه می‌باشد.

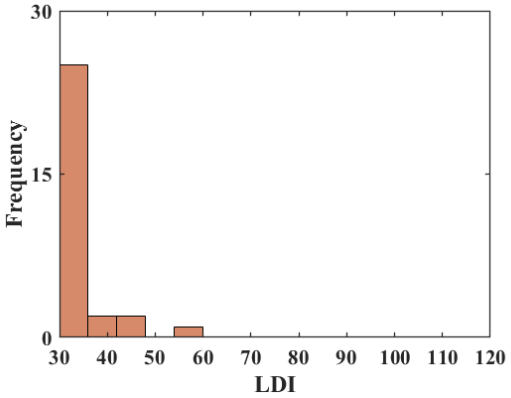
برای مقایسه دو الگوریتم تغییرات زوایای دو لینک در یک اجرا که شامل حداکثر ۲۰ رویداد با شروع از نقطه $(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) = (0,0,0,0)$ می‌باشد بررسی شده است. حداکثر تعداد قدم ها در یک رویداد در بخش آموزش ۱۰۰ در نظر گرفته شده است.

۳-۲-۴- نتایج شبیه‌سازی

جدول (۴) نتایج حاصل از ۳۰ اجرای متمایز روش را در مقایسه با بهترین نتیجه روش NSL به ازای نرخ یادگیری و ضریب دما نشان می‌دهد. همچنان که پیش از این نیز اشاره شد الگوریتم NLSPI مستقل از نرخ یادگیری است لذا در جدول مقداری برای آن لحاظ نشده است. همچنانکه نتایج شبیه‌سازی نشان می‌دهد، زمان آموزش و کیفیت عملکرد در روش NLSPI بسیار بهتر از روش NSL می‌باشد. بطوریکه از نظر



شکل ۱۳: هیستوگرام LDI ها در NSL



شکل ۱۴: هیستوگرام LDI ها در NLSPI

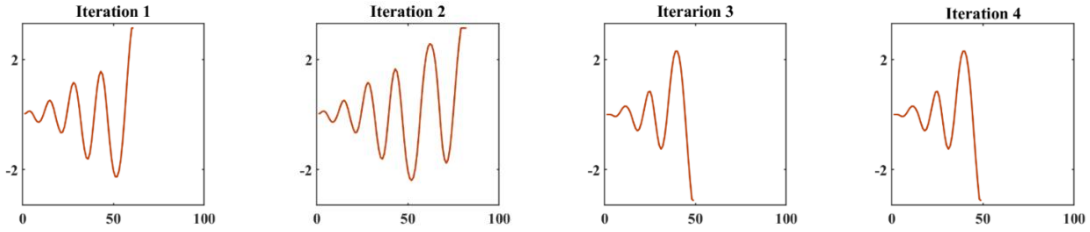
زمان آموزش (LDI)، در حدود ۲۰٪ و از لحاظ نرخ شکست، تقریباً ۳ برابر بهتر عمل کرده است. همچنین از لحاظ زمان اجرا با وجود بالاسری محاسباتی بیشتر NLSPI به دلیل درگیر شدن مقادیر ماتریس A و بردار b در محاسبات مشاهده می‌شود که NLSPI نسبت به NSL، ۳۰٪ سریع‌تر عمل کرده است. در حقیقت نحوه به روز رسانی وزن‌های شبکه عصبی در الگوریتم NLSPI به گونه‌ای است که تقریب مناسب‌تری از ارزش عمل صورت می‌پذیرد. هیستوگرام LDI ها برای هر دو الگوریتم در شکل‌های (۱۳) و (۱۴) نشان داده شده است. همان‌گونه که از شکل‌ها مشخص است به ازای ۳۰ اجرای متمایز در هیچ مورد واگرایی وجود نداشته و در همه اجراها، یادگیری با تعداد اپیزود کمتر از ۶۰ صورت گرفته است. همچنین پراکندگی کمتر LDIها در NLSPI مؤید عملکرد

[1] منظور از یک رویداد شروع از نقطه آغاز و رسیدن به هدف می باشد.
[2] Learning Duration Index (LDI)
[3] Standard Deviation Index (Std. (LDI))
[4] Length of Episode (LE)
[5] Failure Rate

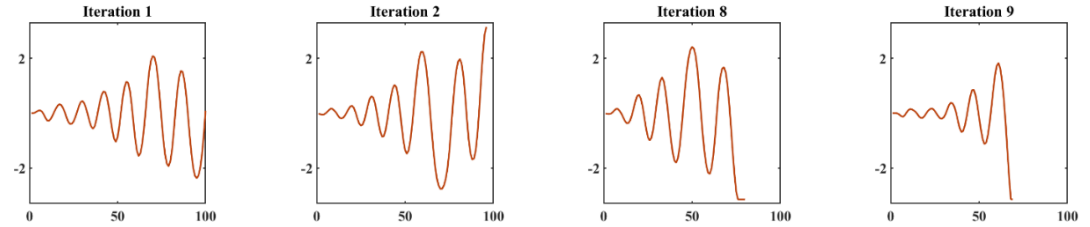
جدول ۴: مقایسه نتایج شبیه‌سازی در مسأله آکروبات

Initial Parameters		Method	Avg. LDI	Std. (LDI)	Failure Rate	LE	Avg. time (Sec)
$T_0 = 0.1$	$\eta_0 = 0.1$	NSL	39.26	12.8	1.7	75.42	149.2
$T_0 = 0.01$	---------	NLSPI	32.5	6.31	0.6	59.7	112.36

LDI, Learning duration index; LE, Length of Episode; NSL, Neural Sarsa Learning; NLSPI, Neural Least Square Policy Iteration



شکل ۱۵: تغییرات زاویه‌ای بازوی اول (θ_1) در روش NLSPI



شکل ۱۶: تغییرات زاویه‌ای بازوی اول (θ_1) در روش NSL

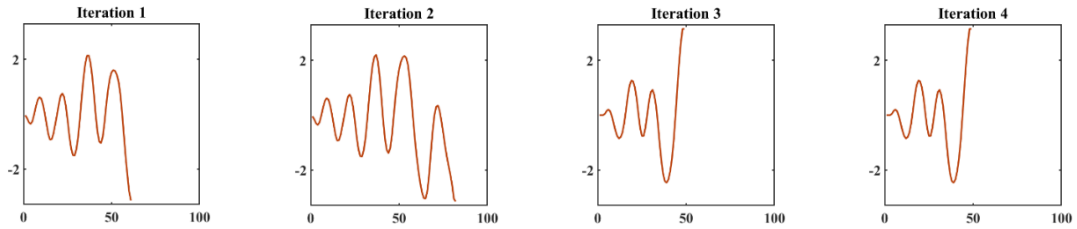
بهتر روش نسبت به NSL می باشد. شکل‌های (۱۵) و (۱۶) تغییرات زوایه‌ای لینک اول (θ_1) و شکل‌های (۱۷) و (۱۸) تغییرات زاویه‌ای لینک دوم (θ_2) را در هر دو روش برای حالتی که $T_0 = 0.1$ و $\lambda = 0.99$ نشان می‌دهند. همانگونه که از شکل‌ها مشخص است روش NLSPI هم از جهت سرعت همگرایی و هم از نظر سیاست نهایی که بدان همگرا شده است به مراتب بهتر از روش NSL عمل کرده است. در روش NLSPI به طور متوسط الگوریتم پس از ۳ رویداد به سیاست شبه بهینه خود همگرا شده است در حالی که این معیار در روشLSPI در حدود ۹ رویداد می‌باشد همچنین سیاست نهایی که در نهایت روش NLSPI بدان همگرا شده است به مراتب بهتر بوده و تعداد قدم‌های زمانی بسیار کمتری در رسیدن به هدف انجام شده است.

شایان ذکر است که در بسیاری از مقالات هدف کنترلی در مسأله آکروبات، رسیدن ربات به اندازه یک واحد بالای نقطه آویز آن در حداقل زمان تعریف شده است. در این حالت نیز NLSPI بسیار سریع و به طور متوسط با دو اپیزود همگرا می‌شود. همچنین معیار تعداد قدم زمانی برای آن به طور متوسط در حدود ۵۰ قدم زمانی است که بهتر از روش KLSPI با ۵۲ قدم زمانی [۱۶]، روش NSL با ۵۸ قدم زمانی [۳]، روش DDQN با ۹۸ قدم زمانی [۲۷]، روش LSPI با ۳۱۵ قدم زمانی [۳۰] و بهتر از دیگر روش‌های شبه بهینه قبلی با حدود ۷۰ قدم زمانی است [۳۱-۳۳].

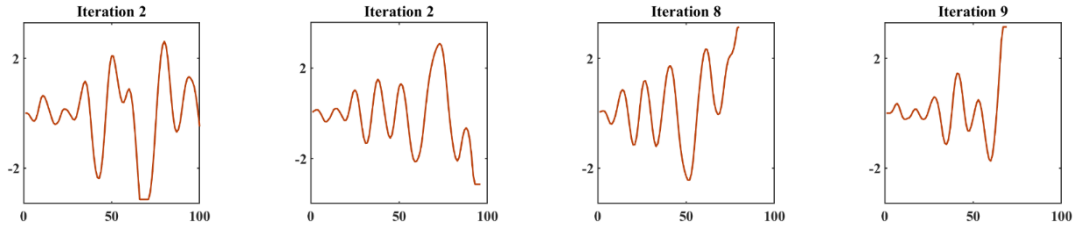
۵- نتیجه‌گیری

در این مقاله یک الگوریتم ترکیبی جدید یادگیری تقویتی پیوسته که از ترکیب روش LSPI با شبکه‌های عصبی حاصل شده معرفی گردید. الگوریتم مذکور که یادگیری تکرار سیاست کمترین مربعات عصبی (NLSPI) نامیده شد از لحاظ معماری دارای ساختار نقاد-تنها بوده و به منظور تنظیم وزن‌های یک شبکه عصبی به صورت بر خط به کار می‌رود. در این پیاده‌سازی با ارائه یک ساختار تقریب زننده عصبی جدید، راهکاری برای رفع چالش تعریف توابع پایه حالت- عمل در الگوریتم LSPI ارائه شد. شبکه عصبی مورد نظر یک شبکه RBF است که به عنوان تقریب زننده‌ی تابع ارزش عمل به کار گرفته شد. بر طبق نتایج به دست آمده، الگوریتم پیشنهادی هم از لحاظ سرعت همگرایی و هم از لحاظ کیفیت سیاستی که بدان همگرا می‌شود از کارایی بسیار مطلوبی برخوردار است. از مزایای روش ارائه شده می‌توان به موارد زیر اشاره کرد: ۱- روش ارائه شده می‌تواند بر چالش تنگنای ابعاد که در مسائل کنترلی با فضای حالت- عمل بزرگ و پیوسته رخ می‌دهد بخوبی غلبه کند. ۲- عدم نیاز به نرخ آموزش در الگوریتم که خود یک چالش در مسائل یادگیری تقویتی است امتیاز ویژه‌ایی برای روش محسوب می گردد. ۳- سرعت همگرایی بالا و عدم وجود موارد واگرایی ویژگی قابل توجه روش محسوب می‌شود.

[DOI: 10.61186/joc.17.1.61]



شکل ۱۷: تغییرات زاویه‌ای بازوی دوم (θ_2) در روش NLSPI



شکل : تغییرات زاویه‌ای بازوی دوم (θ_2) در روش NSL

مراجع:

[1] Sutton, R. S., and Barto, A. G., Reinforcement learning: An introduction, Second Edition, MIT Press, Massachusetts, 2017.

[2] Derhami, V., Alamiyan, F., Dowlatshahi, M.B., Reinforcement Learning, Yazd University Press, 2017.

[3] Derhami, V., Mehrabi, O., Action value function approximation based on radial basis function network for reinforcement learning, Journal of control, Vol.5, No. 1, pp. 50-63, 2011.

[4] Liu, Y. J., Tang, L., Tong, S., Chen, C. P., and Li, D. J., Reinforcement learning design-based adaptive tracking control with less learning parameters for nonlinear discrete-time MIMO systems, IEEE Transactions on Neural Networks and Learning Systems, vol. 26, no.1, pp. 165-176, 2015.

[5] Derhami, V., V.J. Majd, and M.N. Ahmadabadi, Fuzzy Sarsa Learning and The Proof of Existence of its Stationary Points, Asian Journal of Control, pp. 535-549, 2008.

[6] Ghorbani, F., Derhami, V., and Afsharchi, M., Fuzzy Least Square Policy Iteration and Its Mathematical Analysis, International Journal of fuzzy systems, pp.1-14, 2016.

[7] Barakat, A., Bianchi, P., and Lehmann, J. Analysis of a target-based actor-critic algorithm with linear function approximation. CoRR, abs/2106.07472, 2021.

[8] Zaki, M., Mohan, A., Goplan, A., and Manner, S., Actor-Critic based Improper Reinforcement Learning, arXiv, 2022.

[9] Allahverdy, D., Fakharian, A. & Menhaj, M.B. Back-Stepping Integral Sliding Mode Control with Iterative Learning Control Algorithm for Quadrotor UAVs. J. Electr. Eng. Technol. 14, 2539–2547, 2019.

[10] Sheikhlar, A., and Fakharian, A. "Online policy iteration-based tracking control of four wheeled omni-directional robots." Journal of Dynamic Systems, Measurement, and Control 140, no. 8, 2018.

[11] Jia, Y., and Zhou, X. Y., “Policy Gradient and Actor–Critic Learning in Continuous Time and Space: Theory and Algorithms”, arXiv, 2021.

[12] Lagoudakis, M. G., and Par, R., Least-squares policy iteration, Journal of Machine Learning Research, p. 1107-1249, 2003.

[13] Hwang, K.S., Tan, S.W., and Tsai, M. C., “Reinforcement Learning to Adaptive Control of Nonlinear Systems”, IEEE Transactions on Systems, Man, and Cybernetics-Part B, Vol.33, No.3, pp.514-521, 2003.

[14] Bus¸oniu, L., et al., Online least-squares policy iteration for reinforcement learning control, American Control Conference (ACC-10), 2010.

[15] Buşoniu, L., Lazaric, A., Ghavamzadeh, M., Munos, R., Babuška, R., De Schutter, B., Least-squares methods for policy iteration. In: Wiering, M., van Otterlo, M. (Eds.), Reinforcement Learning: State-of-the-Art. In: Adaptation, Learning, and Optimization, vol. 12, Springer, Heidelberg, Germany, pp. 75–109, 2012.

[16] Xu, X., Hu, D., Lu, X., Kernel-based least squares policy iteration for reinforcement learning. IEEE Trans. Neural Netw. 18 (4), 973–992, 2007.

[17] Yahyaa, S., Manderick, B., Knowledge gradient for online reinforcement learning. In: Duval, B., van den Herik, J., Loiseau, S., Filipe, J. (Eds.), Agents and Artificial Intelligence. In: ICAART 2014 LNCS, vol. 8946, Springer, Cham, pp. 103–118, 2014.

[DOI: 10.61186/joc.17.1.61]

using reinforcement learning with probabilistic policy search, Australian & New Zealand Control Conference, pp. 68-73, 2021.

[32] Lim, H. -K., Kim, J. -B., Ullah, I., Heo, J. -S., and Han, Y. -H., Federated Reinforcement Learning Acceleration Method for Precise Control of Multiple Devices, in IEEE Access, vol. 9, pp. 76296-76306, 2021.

[33] FRÄMLING, K., Light-weight reinforcement learning with function approximation for real-life control tasks, In: Proceedings of 5th International Conference on Informatics in Control, Automation and Robotics, Funchal, Madeira, Portugal, pp. 127-134, 2008.

[18] Jakab, H.S., Csató, L., Sparse approximations to value functions in reinforcement learning. In: Koprinkova-Hristova, 9999P., Mladenov, V., Kasabov, N.K. (Eds.), Artificial Neural Networks. Springer, Cham, pp. 295–314, 2015.

[19] Cui, Y., Matsubara, T., Sugimoto, K., Kernel dynamic policy programming: Applicable reinforcement learning to robot systems with high dimensional states, Neural Netw. 94, 13–23, 2017.

[20] Ruan, A., Shi, A., Qin, L., Xu, S., and Zhao, Y., "A Reinforcement Learning-Based Markov-Decision Process (MDP) Implementation for SRAM FPGAs," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 67, no. 10, pp. 2124-2128, 2020.

[21] Howard, R.A., Dynamic Programming and Markov Processes. MIT Press, Cambridge, Massachusetts, 1960.

[22] Perkins, T.J. and D. Precup, A convergent form of approximate policy iteration. Proc. Int. Conf. Neural Information Processing Systems, p. 1595-1602, 2002.

[23] Koller, D. and R. Parr, Policy iteration for factored MDPs. The Sixteenth Conference on Uncertainty in Artificial Intelligence, p. 326–334, 2000.

[24] Hartman, E., Keeler, J. D., Kowalski, J. M, "Layered neural networks with Gaussian hidden units as universal approximations", Neural Computation, Vol. 2, No. 2, pp. 210-215, 1990.

[25] R. M. Kretchmar and C. W. Anderson, "Comparison of CMACs and radial basis functions for local function approximators in reinforcement learning," Proceedings of International Conference on Neural Networks (ICNN'97), Houston, TX, USA, pp. 834-837 vol.2, 1997.

[26] Duan, Y., Chen, X., Houthooft, R., Schulman, J., Abbeel, P.: Benchmarking deep reinforcement learning for continuous control. arXiv preprint arXiv:1604.06778, 2016.

[27] Varga, B, Kulcsár, B, Chehreghani, MH. Deep Q-learning: A robust control approach Int J Robust Nonlinear Control. 33(1): 526– 54, 2023.

[28] Xin Xu, Lei Zuo, Zhenhua Huang, Reinforcement learning algorithms with function approximation: Recent advances and applications, Information Sciences, Volume 261, Pages 1-31, 2014.

[29] second Annual Reinforcement Learning Competition http://rl-competition.org.

[30] André da Motta Salles Barreto, Charles W. Anderson, Restricted gradient-descent algorithm for value-function approximation in reinforcement learning, Artificial Intelligence, Volume 172, Issues 4–5, Pages 454-482, 2008.

[31] Snehal, N., Pooja, Sonam, W., K., Wagh, S. R., and Singh, N. M., Control of an Acrobot system

[DOI: 10.61186/joc.17.1.61]