Федеральное государственное автономное образовательное учреждение
высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»
Физтех-школа Прикладной Математики и Информатики
Кафедра математического моделирования сложных систем и оптимизации

Направление подготовки / специальность: 03.03.01 Прикладные математика и физика (бакалавриат)
Направленность (профиль) подготовки: Методы системного анализа в экономике и управлении

ИССЛЕДОВАНИЕ ДИНАМИЧЕСКИХ ЭКОНОМИЧЕСКИХ МОДЕЛЕЙ МЕТОДАМИ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ

(бакалаврская работа)

Студент:
Лысяков Аркадий Олегович

_____
(подпись студента)

Научный руководитель:
Жукова Александра Александровна,
канд. физ.-мат. наук

_____
(подпись научного руководителя)

Консультант *(при наличии)*:
_____
_____
(подпись консультанта)

Москва 2020

Федеральное государственное автономное образовательное учреждение высшего образования «Московский физико-технический институт (национальный исследовательский университет)»
Физтех-школа прикладной математики и информатики
Кафедра математического моделирования сложных систем и оптимизации

Направление подготовки / специальность: 03.03.01 Прикладные математика и физика (бакалавриат)

ВЫПУСКНАЯ КВАЛИФИЦИРОВАННАЯ РАБОТА НА ТЕМУ:

Исследование динамических экономических моделей методами обучения с подкреплением

(бакалаврская работа)

Выполнил:
Студент 678 группы
Лысяков Аркадий Олегович

_____
подпись студента

Научный руководитель:
Жукова Александра Александровна, к.ф.-м.н.

_____
подпись научного руководителя

Москва 2020

Оглавление

Аннотация 3
Введение 4
1. Теоретическое аспекты задачи 5
1.1 Исследование задачи в терминах оптимального управления 5
1.1 1 Динамические экономические модели 5
1.1.2 Задача оптимального потребления при возможности сбережений 6
1.1.3 Аналитические методы решения задачи 7
1.2 Основные понятия обучения с подкреплением 10
1.3 Обзор методов обучения с подкреплением 13
1.3.1 Постановка задачи в терминах обучения с подкреплением 13
1.3.2 Глубокая q-сеть 13
1.3.3Глубокая детерминированная градиентная политика 15
2. Применение алгоритма глубокой детерминированной градиентной политики 18
2.1 Псевдокод алгоритма 18
2.2 Результаты экспериментов 20
3. Анализ результатов работы алгоритма 22
Заключение 23

Аннотация

Динамические экономические модели, в частности задачи оптимального потребления до сих пор остаются сложными задачами. Одним из подходов к их решению выступает подход обучения с подкреплением. Обучение с подкреплением представляет из себя набор методов, каждый из которых имеет свои допущения, которые влияют на итоговый результат.

В данной работе рассмотрена модификация задачи поедания пирога. Также предложен метод оценки оптимального управления на основе алгоритма глубокой детерминированной градиентной политики.

Введение

Представленная работа посвящена решению проблем экономических динамических моделей.

Данная работа является частью реализуемого в данное время в научном сообществе направлении по применению обучения с подкреплением к прикладным задачам и направлена на разработку технологии определения оптимального управления с помощью методов обучения нейронных сетей.

Предметом исследования работы является применение обучения с подкреплением к решению динамических экономических моделей

Задачей исследования является оценка применимости обучения с подкреплением к нахождению оптимального управления, при котором достигает максимальное значение функционала полезности в рамках задачи оптимального потребления при возможности сбережений

В качестве гипотезы выступает предположение о целесообразности использования обучения с подкреплением применительно к поставленной задаче оптимального потребления при возможности сбережений

В главе 1 сформулирована исследуемая модель среды, описывающая задачу оптимального потребления при возможности сбережений с дискретным временем и конечным горизонтам планирования. После осуществлен обзор методов нахождения оптимального управления в терминах данной задачи. Исследованы особенности модели среды, определяющие выбор методов для применения обучения с подкреплением в рамках поставленной задачи.

В главе 2 подробно изложен выбранный метод решения поставленной задачи, приведена программная реализация данного решения.

В главе 3 произведен обзор результатов работы алгоритма, оценены границы применимости метода

В заключении перечислены выводы данной работы

1. Теоретическое аспекты задачи

1.1 Исследование задачи в терминах оптимального управления

1.1 1 Динамические экономические модели

Динамические экономические модели – модели, описывающие экономику в развитии, в отличии от статических, характеризующих состояние экономики в определенный момент времени, либо долгосрочное состояние. Существуют два подхода к построению таких моделей. Первый – оптимизационный. Он заключается в выборе из различных траекторий экономического развития оптимальной траектории. Второй – исследования равновесия в экономических системах и используются равновесные траектории. В общем виде динамические модели представляют из себя описания начального состояния экономики. Кроме начального состояния, есть некоторые ограничения, эволюция состояния в зависимости от поведения экономических агентов. Так как состояниями можно до некоторой степени управлять, можно ввести критерий и поставить задачу оптимального выбора поведения с точки зрения критерия оптимальности.
Существуют несколько основных задач в случае оптимизационного подхода[1].Приведем примеры некоторых из них. Первая – подход вариационного исчисления.
Задача вариационного исчисления в общем виде формулируется следующим образом:

$$\begin{cases} V[y] = \int_0^T F(t, y(0), y'(t)\, dt) \\ y(0) = A \\ y(T) = B \end{cases}$$

Где A, B, T – даны. Такая задача с единственной переменной состояния y, фиксированными начальными и конечными значениями известна как основная задача вариационного исчисления. Предполагается, что все функции, участвующие в формуле непрерывные и непрерывно дифференцируемые. Это необходимо, поскольку методы вариационного исчисления полностью основаны на классическом дифференциальном исчислении
Второй задачей является подход теории оптимального управления. В теории оптимального управления, кроме переменной времени t и переменной

состояния $y(t)$, вводится еще переменная управления $u(t)$. Управляющая переменная задается на основе уравнения:

$$y'(t) = f(t, y(t), u(t))$$

Если удается найти оптимальное управление $u^*(t)$, то оптимальная кривая $y^*(t)$ находится путем интегрирования уравнения движения. Таким образом, постановка задачи выглядит следующим образом:

$$\begin{cases} V[y] = \int_0^T F(t, y(0), y'(t)\, dt) \\ y(0) = A, \qquad y(T) = B \\ y'(t) = f(t, y(t), u(t)) \end{cases}$$

Где A, B, T – также даны
Данные задачи можно решать различными способами. В качестве одного из них может выступать принцип максимумам [1]. Альтернативным подходом является подход динамического программирования. Основы динамического программирования изложены в [2] Беллманом. Основная идея принципа оптимальности Беллмана заключается в том, что если решать задачу нахождения оптимального пути, начиная с некоторого промежуточного момента, то это решение является частью общего оптимального пути. По сути Беллман предложил алгоритм последовательного решения оптимизационных задач.

1.1.2 Задача оптимального потребления при возможности сбережений

Одной из задач оптимального управления является задача оптимального потребления при возможности сбережений, которая является вариацией задачи поедания пирога[3].

$$\sum_{t=0}^{k-1} \beta^t \ln(u_t) \rightarrow max,$$
$$x_{t+1} = \alpha, x_t - u_t, x_0 = a, 0 \leq x_k$$
(3.1)

Здесь u_t – потребление в момент времени t,

x_t – остаток денег на счету

$\alpha > 1$ – коэффициент прироста денежных средств на счету

$\alpha - 1$ это процент по вкладу

$\beta \in (0, 1)$ – коэффициент дисконтирования.

Чем он меньше, тем менее важно потребление в дальнейшем будущем.

Начальный запас денег $a > 0$

В данной работе рассматривается случай $k = 3$, задача состоит в том, чтобы найти оптимальное управление u_t, которое максимизирует функционал полезности.

1.1.3 Аналитические методы решения задачи

Обычно такие задачи решают аналитическими методами. Приведем пример некоторых из них.

Известно решение задачи, полученное принципом оптимальности Лагранжа[1]:

$$x_0 = a, x_1 = \frac{\alpha a \beta (1 + \beta)}{1 + \beta + \beta^2}, x_2 = \frac{\beta^2 \alpha^2 a}{1 + \beta + \beta^2},$$

$$u_0 = \frac{\alpha a}{1 + \beta + \beta^2}, u_1 = \frac{a \alpha^2 \beta}{1 + \beta + \beta^2}, u_2 = \frac{\beta^2 \alpha^3 a}{1 + \beta + \beta^2}$$

Ответ к задаче является последовательностью состояний и управлений.

Управление в этом случае называют программным управлением

Можно решать задачу по-другому и определять управление в зависимости от текущего состояния системы $u_t(x_t)$. Такое управление называют синтезом управлений

Одним из таких подходов к решению задачи (3.1) является подход динамического программирования[4]

Вводится функция Беллмана $S_j(x)$ – оптимальное значение функционала для вспомогательной задачи:

$$\sum_{t=j}^{k-1} \beta^t \ln(u_t) \rightarrow max,$$

$$x_{t+1} = \alpha x_t - u_t, x_j = x, 0 \leq x_k, t = j, \dots, k-1$$

(3.2)

Запишем в качестве необходимого условия оптимальности уравнение Гамильтона-Якоби-Беллмана[2]:

$$S_j(x) = \max_v(\beta^j \ln(v) + S_{j+1}(\alpha x - v)) \quad (3.3)$$

$$S_k(x) = \begin{cases} 0, x \geq 0 \\ -\infty, x < 0 \end{cases} \quad (3.4)$$

Из данных условий можно последовательно, начиная с $j = k - 1$ получить все управления, как аргументы, доставляющие максимум в уравнении Гамильтона-Якоби-Беллмана:

$$\max_v \left(\beta^j \ln(v) + S_{j+1}(\alpha x - v)\right) = \beta^j \ln(u_t(x)) + S_{j+1}(\alpha x - u_t(x))$$

При $k = 3$ имеем:

$$S_3(x) = \begin{cases} 0, x \geq 0 \\ -\infty, x < 0 \end{cases} \quad (3.5)$$

$$S_2(x) = \beta^2 \ln(\alpha x) \quad (3.6)$$

Уравнение Гамильтона-Якоби-Беллмана для $S_1(x)$:

$$\max_v(\beta \ln(v) + \beta^2 \ln(\alpha(\alpha x - v))) = S_1(x)$$

Приводится к уравнению на управление u_1:

$$\frac{\partial}{\partial v}\left(\beta \ln(v) + \beta^2 \ln\left(\alpha(\alpha x - v)\right)\right) = \frac{\beta}{v} - \frac{\beta^2}{\alpha x - v} = 0$$

Откуда:

$$u_1(x) = \frac{\alpha x}{1 + \beta}$$

$$S_1(x) = \beta \ln\left(\frac{\alpha x}{1 + \beta}\right) + \beta^2 \ln\left(\alpha(\alpha x - \frac{\alpha x}{1 + \beta})\right)$$

Далее необходимо решить уравнение Гамильтона-Якоби-Беллмана для $S_0(x)$. чтобы найти управление u_0:

$$S_0(x) = \max_v \left(\ln(v) + \beta \ln\left(\frac{\alpha(\alpha x - v)}{1+\beta}\right) + \beta^2 \ln\left(\alpha\left(\alpha(\alpha x - v) - \frac{\alpha(\alpha x - v)}{1+\beta}\right)\right)\right)$$

Откуда:

$$u_0(x) = \frac{\alpha x}{1+\beta+\beta^2}$$

И функция Беллмана:

$$S_0(x) = \ln\left(\frac{\alpha x}{1+\beta+\beta^2}\right) + \beta \ln\left(\frac{x\alpha^2\beta}{1+\beta+\beta^2}\right) + \beta^2 \ln\left(\alpha^3 x - \frac{\alpha^3 x}{1+\beta+\beta^2} - \frac{\alpha^4 x}{1+\beta}\left(1 - \frac{1}{1+\beta+\beta^2}\right)\right)$$

Оптимальную последовательность состояний системы получаем применяя последовательно оптимальные управления:

$$x_0 = a, x_1 = \alpha a - u_0(a) = \alpha a - \frac{\alpha a}{1+\beta+\beta^2} = \frac{\alpha a \beta(1+\beta)}{1+\beta+\beta^2},$$
$$x_2 = \alpha x_1 - u_1(x_1) = \frac{\beta^2 \alpha^2 a}{1+\beta+\beta^2}$$

Хотя аналитические решения дают точный ответ на поставленную задачу, остается открытым вопрос обобщения задачи и перехода от детерминированной процентной ставки к стохастической или добавления других видов активов. Точных аналитических решений в такой видоизмененной практически задаче нет, есть только в очень упрощенном варианте. Встает вопрос о разработке численных методов, не привязанных строго к формализму методов оптимального управления и численно решающих обобщенные задачи, без изменения структуры этих самых методов и существенного их усложнения. В качестве одного из таких подходов может выступать обучение с подкреплением[5]

1.2 Основные понятия обучения с подкреплением

Прежде, чем описывать алгоритмы обучения с подкреплением, введем необходимые для этого понятия[6].
Обучение с подкреплением применяется для решения задач, моделирующих в виде взаимодействия агента со средой.
На каждом шаге t агент находится в некотором состоянии среды s_t и может совершить одно действие a_t. Под состояние s_t понимается полное описание среды, оно всегда описывается вектором, матрицей или тензором действительных чисел. Множеством всех допустимых действий в данной среде называют пространством действий. Оно может быть непрерывным или дискретным. При решении задачи важно знать непрерывно или дискретно пространство действий, поскольку некоторые методы предназначены только для дискретного случая, некоторые только для непрерывного
В ответ среда возвращает некоторое вознаграждение r_t и переводит агента в новое состояние s_{t+1}. Этот процесс продолжается до тех пор, пока агент не перейдет в терминальное состояние s_T.
Траекторией τ называется упорядоченный набор состояний-действий $(s_0, a_0, s_1, a_1, \dots, a_{T-1}, s_T)$.
Определим задачу обучения с подкреплением. Естественной постановкой является максимизация суммарной награды $R = \sum_{i=t+1}^{T} \gamma^{i-(t+1)} r_i$ в любой момент времени t. Здесь $\gamma \in [0,1]$ – коэффициент дисконтирования, определяющий долгосрочность планирования агента. Например, значение $\gamma = 0$ соответствует агенту, для которого ценность имеет только вознаграждение после действия в данный момент времени t. Так как награды r, как правило, неизвестны агенту, и, вообще говоря, могут не быть однозначно определенными для каждого действия, постановка задачи в таком виде имеет непрактичный характер. Рассмотрим суммарную награду как функцию от траектории $\tau - R(\tau)$. С такой точки зрения агент имеет задачу нахождения $\max_{\tau} R(\tau)$. Теперь определим отображение множества состояний в множество действий, наличие которого и будет определять возможное множество траекторий.
Политика $\pi(a_t|s_t)$— вероятность (1 или 0 в детерминированном случае) принять в состоянии s_t действие a_t. Тогда о траектории можно говорить, как о случайной величине, имеющей распределение, генерируемое политикой, то есть $\tau \sim \pi$. Это позволяет ввести функцию оценки политики $J(\pi) = E_{\tau \sim \pi} R(\tau)$. Во многих алгоритмах, в том числе и глубокой детерминированной градиентной политики работают с параметризованной политикой, то есть с политикой, выходными данными которых является

функции, зависящие от набора параметров, таких как веса нейронных сетей. Окончательно, задача обучения с подкреплением формулируется как нахождение политики, имеющую максимальную оценку, $\pi^* = argmax_\pi J(\pi)$ где π^* - оптимальная политика

Введем величину $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots$ Данная величина показывает величину суммарного дисконтированная награда, начиная с момента времени t, данная величина обладает следующими свойствами

$$G_t = r_t + \gamma G_{t+1}$$
$$G_0 = R(\tau)$$

Иными словами, величина G_0 показывает суммарную награду за всю траекторию.

Поскольку политика может быть вероятностной, то есть в произвольном состоянии s агент выбирает действие не детерминированным образом, а случайным, оптимальная политика должна максимизировать следующее математическое ожидание:

$$E[G_0] = E[r_0 + \gamma r_1 + .. + \gamma^T r_T] = E_{\pi_\theta}[G_0]$$
$$= E[G_0|\pi_\theta] = E_{s_0}[E_{a_0|s_0}\left[r_0 + E_{a_1|s_1}[\gamma r_1 + ..]\right]$$
$$= \sum_{t=0}^{T} E_{(s_t|a_t)\sim p_\theta}[\gamma^t r_t] = E_{\tau\sim p_\theta(\tau)}[G(\tau)],$$
где $\tau = (s_0, a_0, s_1, a_1, \ldots, a_{T-1}, s_T)$,
$$p_\theta(\tau) = p(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$$
вероятность T – шаговой траектории

$v(s)$ – функция полезности состояния, определяемая как математическое ожидание награды, если агент, начиная с данного состояния s далее действует по политике π:

$$v_\pi(s) = E_\pi[G_t|S_t = s] = E_\pi[r_t + \gamma G_{t+1}|S_t = s]$$
$$= \sum_a \pi(a|s)\sum_{r,s''} p(r, s')|s, a)[r + \gamma E_\pi[G_{t+1}|S_{t+1} = s']$$

$$= \sum_a \pi(a|s)\sum_{r,s\prime} p(r, s'|s, a)[r + \gamma v_\pi(s')] \quad (3.7)$$

Таким образом, $v(s)$ является мерой полезности состояния s

$q(s,a)$ – функция полезности состояния-действия, определяемая как математическое ожидание награды, если агент, находясь в состоянии s, совершая действие a действует далее по политике π:

$$\begin{aligned} q_\pi(s,a) &= E_\pi[G_t|S_t = s, A_t = a] \\ &= E_\pi[R_t + \gamma G_{t+1}|S_t = s, A_t = a] \\ &= \sum_{r,s'} p(r,s'|s,a)[r + \gamma E_\pi[G_t|S_t = s']] \\ &= \sum_{r,s'} p(r,a'|s,a)[r + \gamma v_\pi(s')] \quad (3.8) \end{aligned}$$

Таким образом, $q(s,a)$ является мерой полезности действия a в состоянии s
Теперь выразим связь между $q(s,a)$ и $v(s)$
$q(s,a)$ выражается через $v(s)$:

$$q_\pi(s,a) = \sum_{r,s'} p(r,a'|s,a)[r + \gamma v_\pi(s')]$$

$v(s)$ выражается через $q(s,a)$ аналогично:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{r,s'} p(r,s'|s,a)[r + \gamma v_\pi(s')] = \sum_a \pi(a|s) q_\pi(s|a)$$

Откуда можно теперь выразить $q(s,a)$ через $q(s',a')$:

$$q_\pi(s,a) = \sum_{r,s'} p(r,s'|s,a)[r + \sum_{a'} \pi(a'|s') q_\pi(s',a')$$

Для оптимальной политики π^* в состоянии s имеем оптимальные $v^*(s)$ и $q^*(s,a)$:

$$\begin{aligned} v^* &= max_\pi v_\pi(s) \\ q^*(s,a) &= max_\pi(s,a) \end{aligned}$$

Далее выпишем уравнение Беллмана для оптимальности $v(s)$:

$$v^*(s) = max_a \sum_{r,s'} p(r,s'|s,a)[r + \gamma v^*(s')]$$
$$= max_a E[r_t + \gamma v^*(s_{t+1})|S_t = s, A_t = a]$$

И уравнение Беллмана для оптимальности $q(s,a)$:

$$q^*(s,a) = E[r_t + \gamma \cdot max_{a'} q(s_{t+1}, a')|S_t = s, A_t = a]$$
$$= \sum_{r,s'} p(r,s'|s,a)[r + \gamma \cdot max_{a'} q^*(s',a')]$$

1.3 Обзор методов обучения с подкреплением

1.3.1 Постановка задачи в терминах обучения с подкреплением

В терминах обучения с подкреплением задача (3.1) может быть переформулирована следующим образом:

В качестве самой среды выступает модель банка с фиксированной процентной ставкой $\alpha - 1$,
состояние s_t характеризуется вектором:
$(x_t, t, k - t)$, где x_t - остаток денег на счету в момент времени t,
t – время, прошедшее с начала, $k - t$ – оставшееся время
Действием a_t. в момент времени tявляется управление u_t,
Наградой $r_t = \beta^t \ln(u_t)$
Коэффициент дисконтирование $\gamma = 1$, поскольку дисконтирование β присутствует в самой среде,

$$G_0 = \sum_{t=0}^{k-1} \beta^t \ln(u_t),$$

(3.9)

Таким образом задача нахождения оптимальной политики (оптимальных действий) и задача поиска оптимального управления эквиваленты, поскольку они обе максимизирует суммарную награду.

1.3.2 Глубокая q-сеть

В случае дискретизации пространства состояний и действий в контексте поставленной задачи

Алгоритм глубокой q-сети[7] сочетает обучение с подкрепление и глубокий нейронные сети. На каждом шаге алгоритм выбирает действие из набора допустимых действий $A = \{1, \dots, K\}$. Предполагается, что последовательность состояний s_tявляется Марковским процессом, то есть каждое следующее состояние не зависит от предыдущего.
Для оптимальной $q^*(s, a)$ верно уравнение оптимальности Беллмана:

$$q^*(s, a) = E_{s'}[r + \gamma max_{a'} \ q^*(s', a')|s, a]$$

Основная идея для многих алгоритмов обучения с подкреплением заключается в оценке функции $q(s, a)$ состояния-действия с использованием уравнения Беллмана в качестве итеративного обновления:

$$q^{i+1}(s, a) = E_{s'}[r + \gamma max_{a'} \ q^i(s', a')|s, a]$$

Такие итерационные алгоритмы сходятся к оптимальной функции состояния-действия $q^i \rightarrow q^*$ при i $\rightarrow \infty$. На практике, такой подход может быть неудобным, поскольку функция состояния-действия оценивается отдельно для каждой траектории $(s_0, a_0, s_1, a_1, \dots, a_{T-1}, s_T)$. Предлагается вместо этого использовать аппроксимацию функции для оценки функции состояния-действия $q(s, a; \theta) \approx q^*(s, a)$. Данный алгоритм использует в качестве аппроксимации нейронную сеть, называемую q-сеть. Ее можно обучить, отрегулировав параметры θ_i на i-ой итерации, чтобы уменьшить среднеквадратичную ошибку в уравнении Беллмана, где оптимальное целевое значение $r + \gamma max_{a'} \ q^i(s', a')$ заменяется приблизительным целевым значением $y = r + \gamma max_{a'} \ q(s', a'; \theta_i^-)$, используя параметры θ_i^- из какой-то предыдущей итерации:

$$L_i(\theta_i) = E_{s,a,r}\left[\left(E_{s'}[y|s, a] \ - \ q(s, a; \theta_i)\right)^2\right]$$
$$= E_{s,a,r,s'}\left[\left(y \ - \ q(s, a; \theta_i)\right)^2\right] + E_{s,a,r}\left[V_{s'}[y]\right]$$

На каждом этапе оптимизации хранятся параметры из предыдущей итерации θ_i^-. Функция потерь $L_i(\theta_i)$ – последовательность определенных оптимизационных задач. Последний параметр $V_{s'}[y]$ – дисперсия таргетов, не зависит от параметров θ_i, поэтому может просто игнорироваться в оптимизации. Продифференцируем функцию потерь:

$$\nabla_{\theta_i} L(\theta_i) = E_{s,a,r,s'}\left[\left(r + \gamma max_{a'} q(s', a'; \theta_i^-) - q(s, a; \theta_i)\right)\nabla_{\theta_i} q(s, a; \theta_i)\right]$$

Вместо вычисления математических ожиданий в вышеупомянутом градиенте, будем оптимизировать функцию потерь стохастическим градиентным спуском
Из этого алгоритма может быть восстановлен алгоритм q-обучения[8], путем обновления весов нейронной сети после каждого шага, заменяя отдельные веса и полагая $\theta_i^- = \theta_{i-1}$
Алгоритм q-сети решает задачу обучения с подкреплением напрямую, без явной оценки оптимальной политики π^*. Он учит жадную политику $a = argmax_{a'} q(s, a'; \theta)$. Поскольку алгоритм предполагает конечное количество состояний, он может быть применим в поставленной задаче (3.9) только в случае дискретизации пространства состояний и действий.

1.3.3 Глубокая детерминированная градиентная политика

Хотя алгоритм глубокой q-сети решает задачи с многомерными пространствами наблюдений, он может обрабатывать только дискретные и низко размерные пространства действий. Глубокая q-сеть не может быть непосредственно применена к непрерывным пространствам действий, поскольку она основана на обнаружении действия, которое максимизирует функцию состояния-действия, которая в случае непрерывного значения требует итеративного процесса оптимизации на каждом шагу
Для решения поставленной задачи предлагается воспользоваться алгоритмом Глубокой детерминированной градиентной политики[9].
В данном алгоритме рассматривает параметрическая политика π_θ. Алгоритм максимизирует математическое ожидание суммарной награды $J(\pi_\theta) = E_{\tau\sim\pi_\theta}[R(\tau)]$, максимум достигается при оптимальной политике π^*.
Параметризованную политику предлагается оптимизировать итеративно градиентным спусков:

$$\theta_{k+1} = \theta_k + \alpha \cdot \nabla_\theta J(\pi_\theta)|_{\theta_k}$$

Где $\nabla_\theta J(\pi_\theta)|_{\theta_k}$ называется градиентом политики. Вероятность траектории $\tau = (s_0, a_0, s_1, .., s_T)$ при действиях по политике π_θ выражается следующим образом:

$$P(\tau|\theta) = \rho_0(s_0) \cdot \prod_{t=0}^{T} P(s_{t+1}|s_t, a_t) \cdot \pi_\theta(a_t|s_t)$$

Далее выразим логарифм этой вероятности и выразим производную через логарифм:

$$\log P(\tau|\theta) = \log \rho_0(s_0) + \sum_{t=0}^{T} (\log P(s_{t+1}|s_t, a_t) + \log \pi_\theta(a_t|s_t))$$

$$\nabla_\theta P(\tau|\theta) = P(\tau|\theta) \cdot \nabla_\theta \log P(\tau|\theta)$$

Поскольку среда не зависит от политики θ, поэтому производная по θ от $\rho_0(s_0)$ и $P(s_{t+1}|s_t, a_t)$ равны 0. Итого имеем:

$$\nabla_\theta \log P(\tau|\theta) = \underbrace{\nabla_\theta \log \rho_0(s_0)}_{=0}$$

$$+ \sum_{t=0}^{T} \{ \underbrace{\nabla_\theta \log P(s_{t+1}|s_t, a_t)}_{=0} + \nabla_\theta \log \pi_\theta(a_t, s_t) \}$$

$$= \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t, s_t)$$

Теперь выразим производную математического ожидания награды по параметрам политики θ:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta E_{\tau \sim \pi_\theta}[R(\tau)] =$$

$$\nabla_\theta \int P(\tau|\theta) R(\tau) =$$

$$\int \nabla_\theta P(\tau|\theta) R(\tau) =$$

$$\int P(\tau|\theta) \cdot \nabla_\theta \log P(\tau|\theta)\, R(\tau) =$$

$$E_{\tau \sim \pi_\theta}[\nabla_\theta \log P(\tau|\theta) \cdot R(\tau)] =$$

$$E_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t, s_t) \cdot R(\tau)]$$

При этом, если есть множество траекторий $D = \{\tau_i\}_{i=1,..,N}$, где каждая траектория получена одной игрой, когда агент действует по политике π_θ, то можно использовать аппроксимацию градиента:

$$g \approx \frac{1}{N} \sum_{\tau \in D} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t, s_t) R(\tau)$$

Алгоритма глубокой детерминированной градиентной политики является одним из алгоритмом семейства оптимизации политики. Он одновременно обновляет q-функцию и политику по ней. Аналогично алгоритму q-сети оптимальное действие выбирается исходя из $a^* = argmax_a\ q(s,a)$, зная оптимальную $q^*(s,a)$ для состояния s.
Алгоритм использует два типа нейронных сетей. Первый – актер-сеть (задают политик π), второй – критик-сеть (оценивают значение q-функции) Глубокая детерминированная градиентная политика использует уравнение Беллмана оптимальности:

$$q^*(s,a) = E_{s' \sim P}[r(s,a) + \gamma \cdot max_{a'} q^*(s',a')$$

Аналогично q-сети используется аппроксимация $q^*(s,a)$. Она задается нейронной сетью-критиком $q_\varphi(s,a)$. Где $\varphi -$ параметры данной сети. Для хранения данных по истории игр используется буфер памяти. В нем хранятся данные вида $< s, a, r, s', d >$. Где d флаг, является ли состояние s терминальным или нет. Для обучения данной сети используется среднеквадратичная функция потерь Беллмана следующего вида:

$$L(\varphi, D) = E_D \left\{ q_\varphi(s,a) - \left(r + \gamma \cdot (1-d) max_{a'} q_\varphi(s',a') \right) \right\}^2$$

Где $r + \gamma \cdot (1-d) max_{a'} q_\varphi(s',a')$ называется целевым значением. Итеративная минимизация подобного вида среднеквадратичной ошибки Беллмана приводит последовательно $q_\varphi(s,a)$ к целевому значению. Однако, целевое значение также зависит от параметров φ, которые оптимизируются, что делает процесс оптимизации неустойчивым. Именно для решения данной проблемы используют вторую критик-сеть, называемую целевой критик-сетью, обозначаемую φ_{targ}. И среднеквадратичная ошибка Беллмана преобразуется в следующий вид:

$$L(\varphi, D) = E_D \left\{ q_\varphi(s,a) - \left(r + \gamma \cdot (1-d) q_{\varphi_{targ}}(s', a') \right) \right\}^2$$

Параметры φ обновляются путем минимизации ошибки $L(\varphi, D)$, а веса целевой целевой критик-сети обновляются по правилу:

$$\varphi_{targ} \longleftarrow \tau \cdot \varphi + (1 - \tau) \cdot \varphi_{targ}$$

Отметим, что для медленного стабильного обучения используют $\lambda \ll 1$. Поскольку алгоритм детерминированный, в самом начале у агента нет возможности попробовать достаточного количества действий для оптимизации параметров нейронных сетей, так как он всегда будет совершать одни и те же действия исходя из инициализации этих нейронных сетей. Для решения этой проблемы к действию по политике добавляется шум Орнштейна-Ухленбека[10]. Алгоритм учит политику $\pi_\theta(s)$ (для это используются актер-сети), которая максимизирует $q_\varphi(s,a)$ путем решения следующей задачи оптимизации:

$$max_\theta E_{s\sim D}[q_\varphi(s, \pi_\theta(s)]$$

Параметры политики также обновляются постепенно, путем итеративного стохастического градиентного спуска:

$$\theta \longleftarrow \tau \cdot \theta + (1 - \tau) \cdot \theta'$$

Далее будет рассмотрен непосредственно сам алгоритм, применительно к поставленной задаче.

2. Применение алгоритма глубокой детерминированной градиентной политики

2.1 Псевдокод алгоритма

Программная реализация модели среды и алгоритм написаны при на языке программирования $Python$ на основе библиотеки $PyTorch$, вычисления производились в среде $Google\ Colaboratory$
Псевдокод алгоритма детерминированной градиентной политики применительно к задаче (3.9):

- Инициализация главную актер-сеть для политики $\pi(\theta)$ параметрами θ и главную критик сеть $q(s, a, \varphi)$ параметрами φ
- Инициализация целевых сетей параметрами $\varphi' \leftarrow \varphi, \theta' \leftarrow \theta$
- Инициализация буфера памяти R
- Цикл по j от 1 до M:

    Инициализация случайного шума $\mathcal{N}$

    Инициализация начального состояния s_0

    Цикл по t от 0 до T:

        Выбор $a_t = \pi(s_t|\theta) + \mathcal{N}$ согласно текущей политике

        Выполнение действия a_t, получение награды r_t, переход в состояние s_{t+1}

        Добавление (s_t, a_t, r_t, s_{t+1}) в R

        Случайный выбор подмножество вида (s_i, a_i, r_i, s_{i+1}) размера N из R

        $y_i = r_i + \gamma \cdot q(s_{i+1}, \pi(s_{i+1}|\theta')|\varphi')$

        Обновить главную критик-сеть путем минимизации ошибки:

$$L = \frac{1}{N}\sum_i (y_i - q(s, a|\varphi)^2$$

        Обновить главную актер-сеть используя вычисления приближения градиента:

$$\nabla_\theta J \approx \frac{1}{N}\sum_i \nabla_a q(s_i, a = \pi(s_i|\varphi) \cdot \nabla_\theta \pi(s|\theta)$$

        Обновить веса целевых нейронных сетей:

$$\theta' \leftarrow \tau \cdot \theta + (1 - \tau) \cdot \theta'$$
$$\varphi' \leftarrow \tau \cdot \varphi + (1 - \tau) \cdot \varphi$$

    Завершить цикл

Завершить цикл

В качестве архитектуры для нейронных сетей были выбраны два полно связных слоя, связанных между собой функция активациями $Relu$:

$$\text{f(x)} = \begin{cases} 0, x \leq 0 \\ x, x \leq x \end{cases}$$

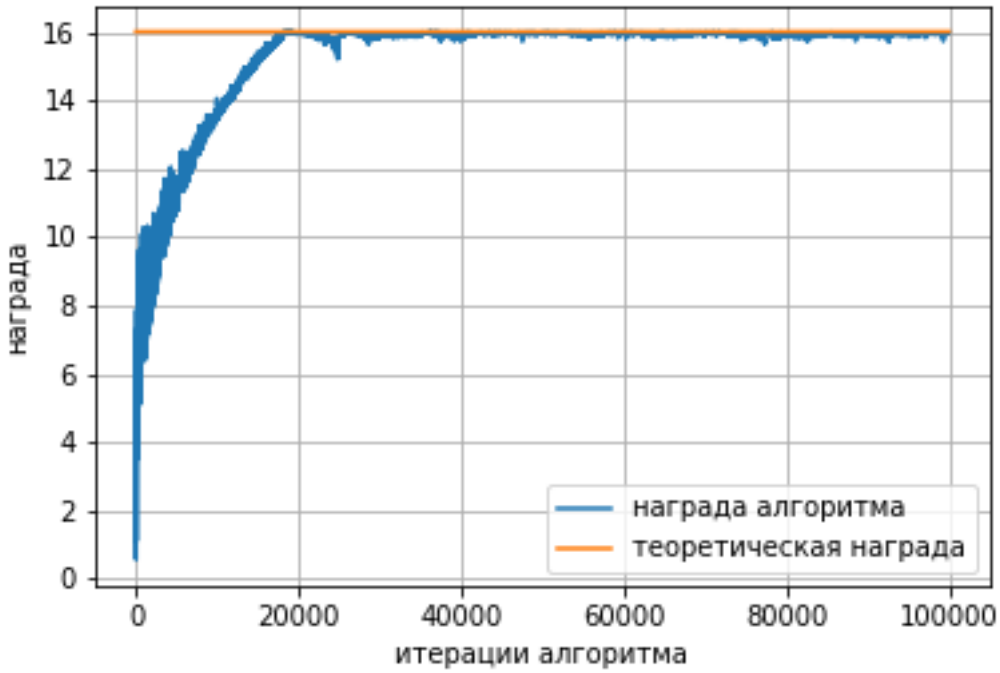
На параметры нейронных сетей была наложена слоевая нормализация[11] для борьбы с переобучением и устойчивой сходимости метода. Размер первого полно связного слоя – 400, размер второго – 300. Начальная инициализация весов соответствует рекомендациям работы[9](веса

выбираются случайно исходя из равномерного распределения в рекомендованных пределах). Размер буфера памяти выбран – 10000, размер $\mathrm{N} - 256$, параметр обучения $\tau - \ 5 * 10^{-4}$, $T = 2$. Программную реализацию можно посмотреть тут: https://drive.google.com/drive/folders/1xFvuj8MissGGKErcbQaGaR5iyyfcFUrX?usp=sharing. Доступ открыт для всех пользователей группы доступа MIPT с доменом phystech. edu

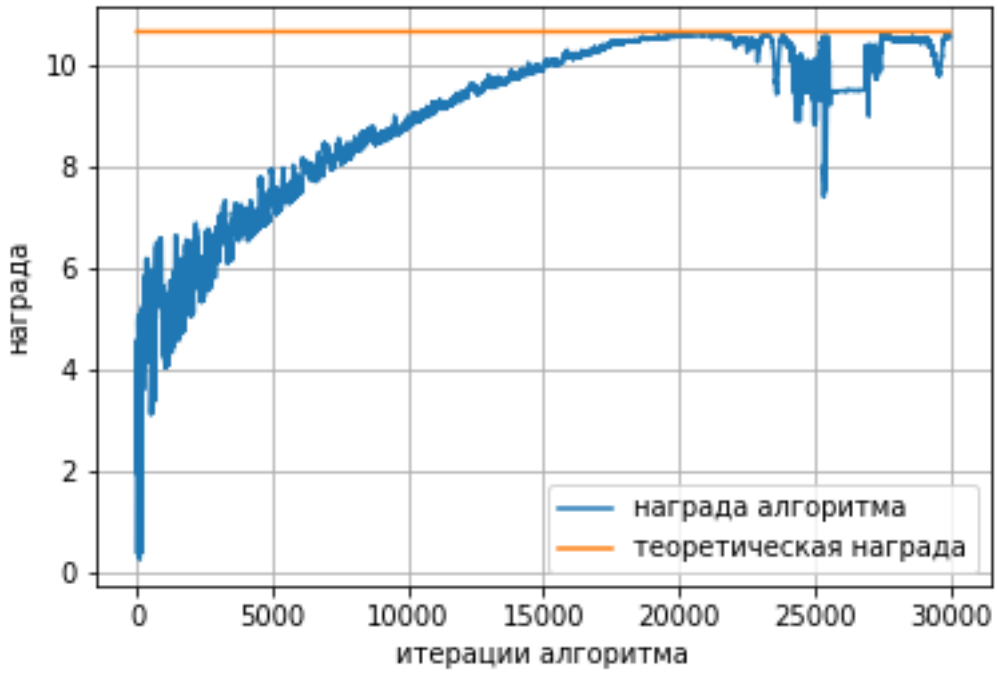
2.2 Результаты экспериментов

Ниже представлены графики сходимости алгоритма для различных вариаций среды.

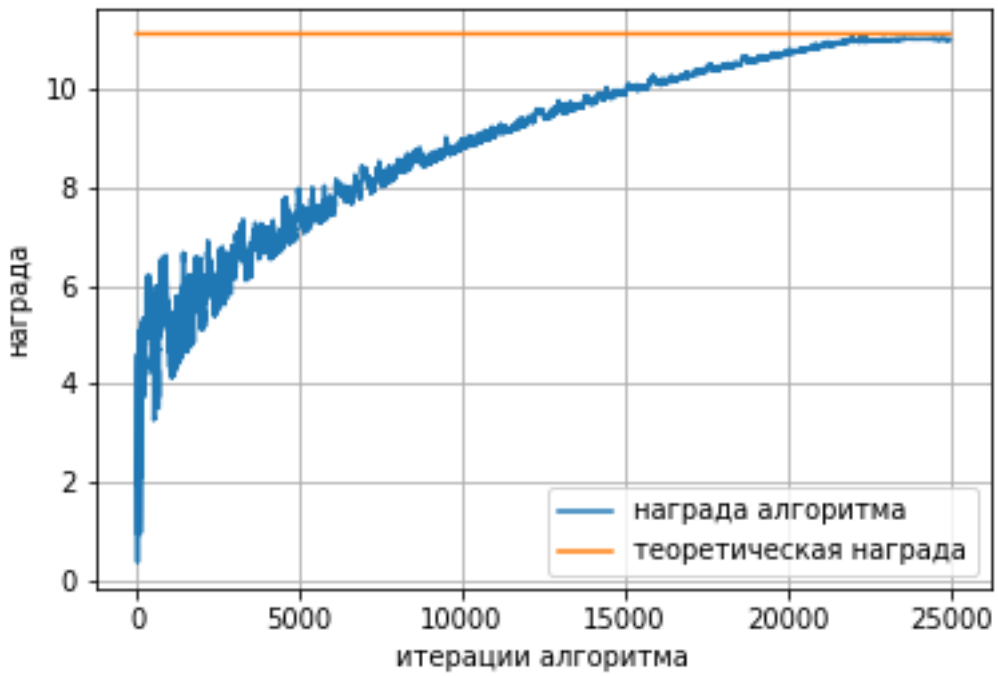
1) $a = 1000, \beta = 0.9, \alpha = 1.05$, теоретический максимум награды ≈ 16.01



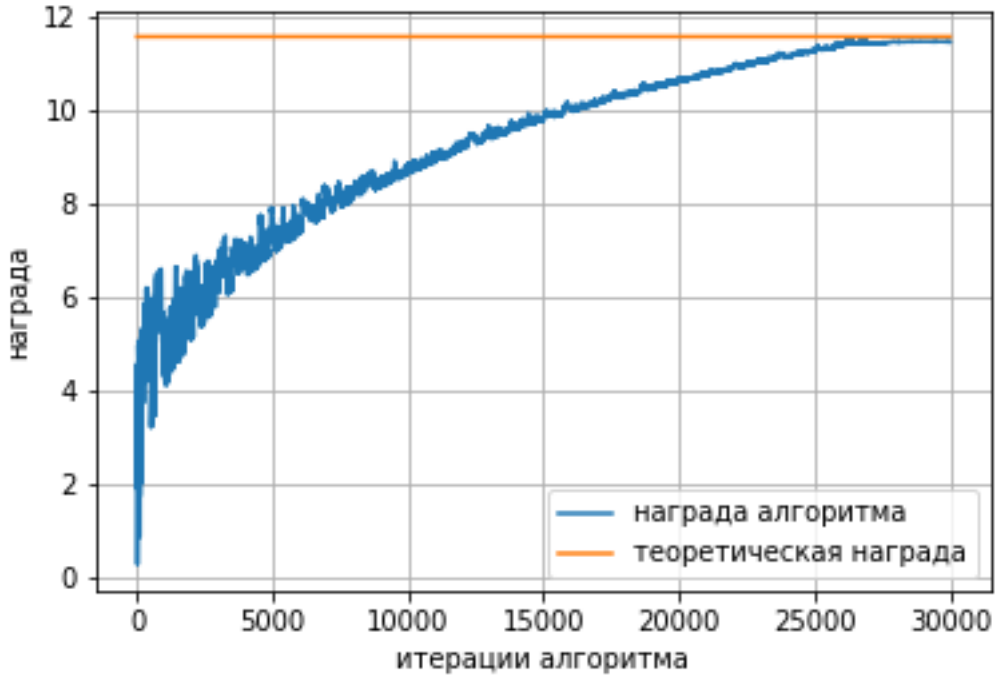
2) $a = 1000, \beta = 0.5, \alpha = 1.1$, теоретический максимум награды ≈ 10.68



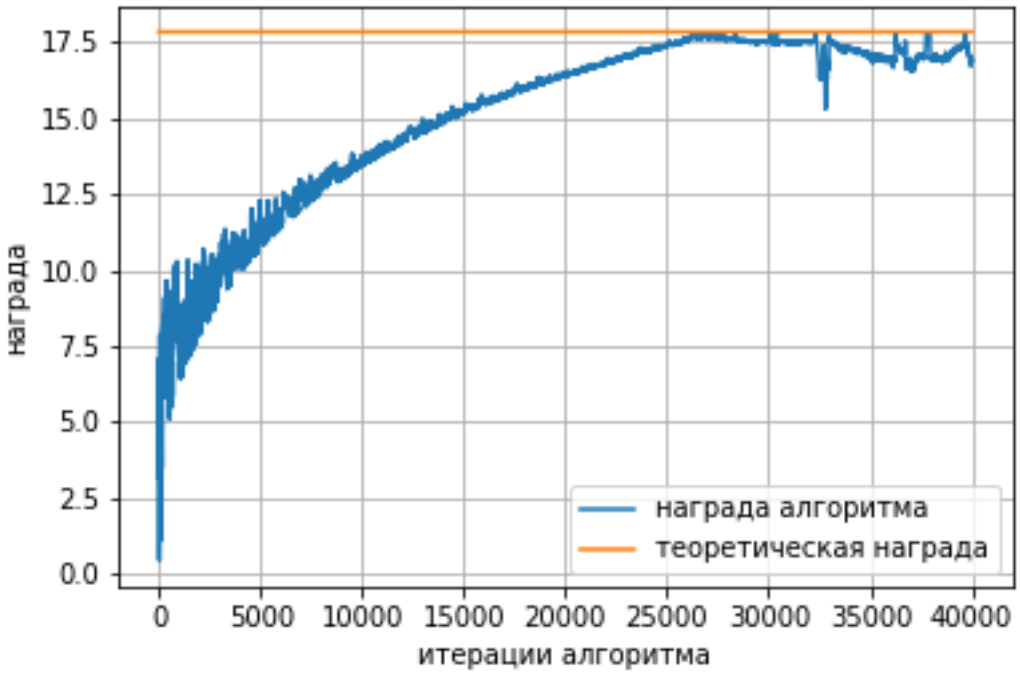
3) $a = 1000, \beta = 0.5, \alpha = 1.3$, теоретический максимум награды ≈ 11.14



4) $a = 1300, \beta = 0.5, \alpha = 1.3$, теоретический максимум награды ≈ 11.60



5) $a = 1300, \beta = 0.9, \alpha = 1.3$, теоретический максимум награды ≈ 17.84



3. Анализ результатов работы алгоритма

Ниже приведена сравнительная таблица теоретической максимальной награды с теми результатами, к которым сошелся алгоритм глубокой

детерминированной градиентной политики в зависимости от параметров задачи

a	β	α	$R_{\text{теория}}$	$R_{\text{практика}}$
1000	0.9	1.05	16.01	15.96
1000	0.5	1.1	10.68	10.44
1000	0.5	1.3	11.14	11.03
1300	0.5	1.3	11.60	11.50
1300	0.9	1.3	17.84	17.68

В таблице приведены суммарные теоретические и практические награды, другими словами, теоретические и практические функции полезности состояния $v(s_{t=0})_{\text{теория}}, v(s_{t=0})_{\text{практика}}$
Для вычисления остальных $v(s)$ можно воспользоваться детерминированной политикой, к которой сошелся алгоритм и формулой (3.7) для практических значений и аналитическими решениями задачи (3.1) и формулой (3.7) для теоретических значений
Хотя результаты могут показаться неплохими, существенным фактором в исследовании является тот факт, что в задаче (3.9) рассматривался только один горизонт планирования $T = 3$. Кроме этого, в ходе экспериментов было установлено, что настройка параметров алгоритма требует большого количества времени, не исключено, что увеличение горизонта планирования может существенно усложнить этот процесс и сделать применение данного метода не целесообразным. В качестве решения этой проблемы может выступать процесс распараллеливания программных вычислений и/или увеличение программных мощностей путем переноса вычислений на вычислительные кластеры. Еще одним существенным фактором является выбор архитектуры нейронных сетей. Ввиду отсутствия явной формальной теории для подбора архитектур, выбранные архитектуры могут быть далеки от оптимальных. Однако применение обучение с подкреплением может являться альтернативным инструментом для решения задач динамической оптимизации. Исследования на стыке экономических динамических моделей и обучения с подкреплением в перспективе могут способствовать разработке новой методологии, сочетающей в себе опыт накопленный в обеих областях.

Заключение

В заключении подведем итоги проделанной работы.

- Была изучена задача оптимального потребления при возможности сбережений.

- Для данной задачи были изучены аналитические подходы к решению и подходы, основанные на обучение с подкреплением.
- Были исследованы особенности модели среды для поставленной задачи и выбран метод на основе обучения с подкреплением для численного решения задачи.
- После исследования особенностей среды был реализован программный код для модели среды и алгоритма глубокой детерминированной градиентной политики на языке программирования $Python$.
- Произведены численные эксперименты для различных параметров модели
- Произведен анализ полученных результатов

План дальнейших действий включается в себя исследования поведения алгоритма для модели с $T > 3$, улучшения самого алгоритма путем подбора оптимальных архитектур нейронных сетей и подбора оптимальных методов оптимизации для обучения нейронных сетей для более быстрой и устойчивой сходимости алгоритма, поиск алгоритма для обобщенной задачи с несколькими видами активов[12].

Литература

1. Бекларян Л. А., Флёрова А. Ю., Жукова А.А. Методы оптимального управления : учебное пособие // Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего образования "Московский физико-технический институт (государственный университет)". - Москва: МФТИ. 2018
2. Bellman R. DYNAMIC PROGRAMMING AND A NEW FORMALISM IN THE CALCULUS OF VARIATIONS // Proc. Natl. Acad. Sci. Proceedings of the National Academy of Sciences, 1954. Vol. 40, № 4. P. 231–235.
3. Kamihigashi T., Le Van C. Necessary and Sufficient Conditions for a Solution of the Bellman Equation to be the Value Function: A General Principle.
4. Dean M. Dynamic Optimization and Optimal Control.
5. Bertsekas D.P. Neuro-Dynamic Programming // Encyclopedia of Optimization. Springer US, 2008. P. 2555–2560.
6. Kaelbling L.P., Littman M.L., Moore A.W. Reinforcement learning: A survey // J. Artif. Intell. Res. 1996. Vol. 4. P. 237–285.
7. Mnih V. et al. Human-level control through deep reinforcement learning // Nature. 2015. Vol. 518.
8. Watkins C.J.C.H., Dayan P. Q-learning // Mach. Learn. Springer Science and Business Media LLC, 1992. Vol. 8, № 3–4. P. 279–292.
9. Lillicrap T.P. et al. Continuous control with deep reinforcement learning // 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings. International Conference on Learning Representations, ICLR, 2016.
10. Uhlenbeck G.E., Ornstein L.S. On the theory of the Brownian motion // Phys. Rev. 1930. Vol. 36, № 5. P. 823–841.
11. Ba J.L., Kiros J.R., Hinton G.E. Layer Normalization.
12. Шананин А.А Элементы финансовой математики : учебное пособие // Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего образования "Московский физико-технический институт (государственный университет)". - Москва: МФТИ. 2018