PAMS: Platform for Artificial Market Simulations
～Python ベースの人工市場シミュレーションプラットフォームと深層学習との融合～

平野　正徳[†,††] 高田　亮介[†††] 和泉　　潔[††]

†† 東京大学大学院工学系研究科　〒113–8656 東京都文京区本郷 7–3–1
††† 東京大学大学院総合文化研究科　〒153–8902 東京都目黒区駒場 3–8–1
E-mail: research@mhirano.jp, takata@sacral.c.u-tokyo.ac.jp, izumi@sys.t.u-tokyo.ac.jp

あらまし　本稿では，新しい人工市場シミュレーションプラットフォームの PAMS: Platform for Artificial Market Simulations を示す．PAMS は，深層学習技術などとのシームレスな融合を前提におき，Python ベースのアーキテクチャを採用しつつ，様々なシミュレーションが可能になるように，ユーザーが簡便にエージェントや環境をカスタマイズ可能になっている．実際に，使用例として，本稿では，深層学習による価格予測を行うエージェントを用いた研究を行い，PAMS の有効性について示す．
キーワード　人工市場，シミュレーション，PAMS，深層学習

PAMS: Platform for Artificial Market Simulations
– Python-based Platform for Artificial Market Simulations and Challenges on its Integration with Deep Learning –

Masanori HIRANO[†,††], Ryosuke TAKATA[†††], and Kiyoshi IZUMI[††]

†† School of Engineering, The University of Tokyo, 7–3–1, Hongo, Bunkyo-ku, Tokyo, 113-8656 Japan
††† Graduate School of Arts and Sciences, The University of Tokyo, 3–8–1 Komaba, Meguro-ku, Tokyo, 153–8902 Japan
E-mail: research@mhirano.jp, takata@sacral.c.u-tokyo.ac.jp, izumi@sys.t.u-tokyo.ac.jp

Abstract　This paper presents a new artificial market simulation platform, PAMS: Platform for Artificial Market Simulations. PAMS is developed as a Python-based simulator that is easily integrated with deep learning and enabling various simulation that requires easy users' modification. In this paper, we demonstrate PAMS effectiveness through a study using agents predicting future prices by deep learning.
Key words　Artificial Market, Simulation, PAMS, Deep Learning

1. はじめに

金融市場におけるマルチエージェントシミュレーションである人工市場シミュレーションはとても有用な技術である．金融市場は，非常に複雑な構成要素が相互に作用し合って成り立っており，特に，そのトレーダー間の相互作用の影響は非常に大きい．加えて，金融市場の現象に対する支配方程式も存在しない．そのため，それらの現象を再現するためには，トレーダー間およびトレーダーと環境 (市場) 間の相互作用をモデリング可能なマルチエージェントシミュレーションの必要性は極めて高い．例えば，Lux ら [1] は金融市場シミュレーションにおけるエージェント間の相互作用がない場合には，金融市場に広く見える現象が再現できないことを示しており，マルチエージェントシミュレーションの必要性が明らかになっている．

加えて，金融市場におけるマルチエージェントシミュレーションは，実市場で検証できないシナリオの検証などでも有用である．金融市場の制度設計は，市場の成長と安定性のためには非常に重要であり，特に，規制制度の設計においては，過度

(注†)：責任著者

な規制は副作用を引き起こす可能性などがある．しかしながら，新たな規制や制度の議論に当たっては，これまで存在しない局面を議論することになり，実際にその制度を導入して検証することは困難である．そのため，制度設計の議論においては，マルチエージェントシミュレーションが有効な手段であると言える．

これらの背景から，金融市場におけるマルチエージェントシミュレーションは，金融市場において，重要な研究分野であるとともに，さらなる進化を遂げる必要があると考えられる．

一方で，近年の多くのデータ分析の手法は，深層学習や機械学習を用いたものとなっている．例えば，株価予測のタスクを取ってみても，深層学習や機械学習は非常に多く活用されている [2]〜[7]．実際の市場においても，トレーダーの意思決定にこうした技術が使われていることは想像に難くない．

また，人工市場シミュレーションと深層学習技術を融合した手法も提案され始めている．トレーダーの深層強化学習戦略を人工市場シミュレーションで学習させる手法 [8] やデータマイニングで構築したエージェントモデルを人工市場で活用する手法 [9], [10]，データマイニング手法を用いて人工市場シミュレーションの妥当性を評価する手法 [11] などが提案されている．

これらの近年の深層学習技術の応用の流れを踏まえると，人工市場シミュレーションのプラットフォーム自体がそれらとスムーズに連携できるような基盤が重要であるが，実現されていない．人工市場シミュレーションプラットフォームとしては，“Platform for large-scale and high-frequency artificial market” (Plham) [12], [13]，後続のプラットフォームとして，Java 版の PlhamJ [14] があげられる．また他にも，U-MART [15]，Santa Fe Artificial Stock Market [16]， Agent-based Interactive Discrete Event Simulation (ABIDES) [17] なども提案されている．人工市場シミュレーションの実装に当たっては，計算速度の観点から，C++ や Java などのコンパイラ言語が用いられることが多い．しかしながら，深層学習や機械学習のライブラリは，Python などのスクリプト言語で実装されることが多く，これらを組み合わせる場合には，人工市場シミュレーション側からサブプロセスとして，深層学習や機械学習のスクリプトを呼び出すか，その逆をする必要があり，容易ではない．加えて，人工市場シミュレーションをうまく活用するためには，ユーザーがトレーダーエージェントを実装する必要があるが，トレーダーエージェントの実装を容易にするための機能が十分に備わっているのは，たかだか Plham および PlhamJ のみである．これらの要件を総合すると，深層学習との融合に便利である Python で記述されており，ユーザーが容易にエージェントのコードを変更できるようなクラス設計がされており，かつ，高速に動作する人工市場シミュレーションプラットフォームが必要であると言える．この要件を満たす人工市場シミュレーションはまだ存在しない．

そこで，本研究では，Python ベースで Plham の設計思想を継承した人工市場シミュレーションプラットフォームである，PAMS: Platform for Artificial Market Simulations を提案する．このプラットフォームは，Python のパッケージレポジトリである PyPI(注1) より，pip コマンド等を通じてインストール可能であり，かつ，容易にそのシミュレーションのコードを変更可能である．深層学習との融合も容易に実施可能であり，動作速度も従来の Plham 等と比較して遜色がない．コードは `https://github.com/masanorihirano/pams` で公開しており，ドキュメントも `https://pams.hirano.dev/` で公開している．加えて，コードとともに，examples として(注2)，簡易に動かすことができる Python の Notebook 実装も公開しており，これを用いて，PAMS の例を試すことが可能となっている．

本研究では，さらに，深層学習を用いた PAMS 実装の人工市場シミュレーションの実験と分析を紹介する．

2. 関 連 研 究

マルチエージェントシミュレーションは 1970 年前後から多くの社会現象の解析・理解に活用されてきている．Schelling [18] は，マルチエージェントシミュレーションを用いて，分居のシミュレーションを行い，人種ごとの分居が起こるメカニズムを明らかにした．Axelrod [19], [20] は，マルチエージェント環境において，囚人のジレンマコンテストを実施し，様々な種のエージェントを分析した．Epstein ら [21] は，アリと食料を模した人工的な世界をシミュレーションすることで，人工社会の構築可能性を示した．Lux ら [1] は， 金融市場において観測される特有の現象は，エージェント間のインタラクションがなければ再現することができない，ということを示し，マルチエージェントシミュレーションの必要性を解いた．他にも様々なシミュレーションが社会的に活用されている．Sajjad ら [22] は，人口動態のシミュレーションを実データに基づいて構築した．Nonaka ら [23] は，人流シミュレーションを構築し，避難シミュレーションを実施した．Braun-Munzinger ら [24] は，債券市場向けのマルチエージェントシミュレーションを構築した．Kurahashi ら [25] は，マルチエージェントシミュレーションを用いて，COVID-19 の感染拡大防止策の検討を行った．

社会現象においては，その支配方程式が存在していない，あるいは解明されていないために，その解析が難しい．そこで，シミュレーションは，社会科学分野での有用であると主張されている [26]．また，特にマルチエージェントシミュレーションは重要であると主張する研究も存在する [27], [28].

このように多くの活用が行われている一方で，マルチエージェントシミュレーションの構築論に関しても議論が行われている．Axelrod [29] は，“Keep It Simple Stupid”(KISS) 原理をマルチエージェントシミュレーションにおいて提唱しており，より簡単なモデルで複雑な現象を再現することにより，その本質にある現象の理解に寄与することができるとした．さらに Edmonds ら [30] は “Keep it Descriptive Stupid”(KIDS) 原理を提唱し，その目的のためであれば，KISS でなくとも，説明可能性があればよいと提唱した．寺野 [31] も同様に，KISS 原理を超えるようなエージェントシミュレーションの手法の可能性を議

(注1)：`https://pypi.org/project/pams/`
(注2)：`https://github.com/masanorihirano/pams/tree/main/examples`

論している．

本稿では，マルチエージェントシミュレーションの中でも，特に，金融市場にフォーカスした，人工市場シミュレーションをターゲットとしている．社会シミュレーションの重要性と同様に，金融市場におけるシミュレーションの重要性は多く議論されてきている [27], [28]．前述の通り，Lux ら [1] は金融市場シミュレーションにおけるエージェントインタラクションの必要性を示しており，特にマルチエージェントシミュレーションの必要性が明らかになっている．さらに，Mizuta [32] は，金融におけるマルチエージェントシミュレーションが金融の規制や制度設計に貢献できる可能性を論じている．

さらに，既存の金融理論の限界が主要な実務者から主張されていることも，人工市場シミュレーションの研究に拍車をかけている．2007 年から 2008 年にかけて発生した金融危機では，米国住宅市場の悪化による住宅ローンのデフォルト問題を契機として，投資銀行の破綻を引き起こし，さらにそれが世界的な金融市場全体の混乱を引き起こす結果となった．当時，欧州中央銀行 (ECB) の総裁であった Trichet は，従来の金融理論では金融危機中の政策決定の役に立たなかったと述べ，行動経済学やマルチエージェントシミュレーションの必要性を述べた [33]．また，投資銀行やヘッジファンドでリスク管理をとりあつかい，米国財務省でも勤務経験のある Bookstaber は，著書 [34] 内で金融危機を振り返って，従来の経済学では，危機の時のような歪みが増幅した状態を取り扱うことが難しく，エージェントシミュレーションのような複雑さを取り込むことのできる手法へのパラダイムシフトを推奨している．

具体的な人工市場シミュレーションを用いた研究も様々ある．Cui ら [35] はエージェントの行動において知能がない場合 (Zero-Intelligence)，一部の金融市場で見られる現象を再現できないことを示した．Torii ら [36] は価格ショックが他の株式にも伝搬するシミュレーションを実施し，そのメカニズムを分析した．Mizuta ら [37] は，株式市場における価格の呼値の影響を人工市場で分析し，呼値の引き下げが市場シェアの維持には必要であると主張し，東京証券取引所における呼値の切り下げの議論に貢献した．Hirano ら [38] は，自己資本比率規制の影響を人工市場を用いて分析を行い，市場の安定性を確保するために導入されたはずの自己資本比率規制が逆に価格ショックの増幅や値上がりを抑えてしまう可能性があることを示した．他にもフラッシュクラッシュを人工市場で再現した研究なども存在する [39], [40]．

これらの人工市場シミュレーションを実現するためのプラットフォームも複数提案されている．Torii ら [12] は “Platform for large-scale and high-frequency artificial market”(Plham) [13] を提案・公開している．さらに，後続のプラットフォームとして，Java 版の PlhamJ [14] も提案・公開されている．また他にも，U-MART [15]，Santa Fe artificial stock market [16]， agent-based interactive discrete event simulation (ABIDES) [17] なども提案されている．本研究では新たに，PAMS というプラットフォームを示す．

3. PAMS: Platform for Artificial Market Simulations

3.1 基本コンセプト

PAMS の基本コンセプトは，深層学習との融合を前提とした，ユーザーが容易にカスタマイズ可能な tick-time スケールの人工市場シミュレーションである．深層学習の融合という観点では，Python ベースのパッケージとしてリリースを行っており，ユーザーが容易にカスタマイズ可能であるという観点は，オブジェクト指向の Class 型のアーキテクチャを採用することで，オーバーライドしてさまざまなカスタマイズを行えるということに加え，エージェントやマーケットの設定を json または Python の Dict で設定可能であることで実現している．

人工市場シミュレーションには，time-driven なシミュレーションと event-driven なシミュレーションを考えることができる．time-driven なシミュレーションとは，一定の時間軸が動いている上で，様々なエージェントが同時的に行動を行う一方で，event-driven なシミュレーションでは，エージェントが行動を起こす時にのみ，時刻が進むという仕組みである．人工市場シミュレーションにおいては，前者の time-driven なシミュレーションは難しさを持っている．実際の金融市場はマイクロ秒単位での注文が入っており，これを実際のシミュレーションで time-driven なシミュレーションを行おうとすると，高速で複雑な計算を要することになる．そのため，注文が発生した回数ベースの tick-time 時間スケールでの event-driven なシミュレーションが向いており，PAMS で tick-time スケールでのシミュレーションを採用している．tick-time スケールのシミュレーションを採用することで，短時間に膨大な意思決定と注文が行われるような，フラッシュクラッシュのような現象も再現可能であり，一方で，近似的に注文間隔を間引けば，複数年レベルでのシミュレーションも可能になる．

さらに，PAMS では，ユーザーが簡単に利用できるように様々な工夫を行っている．従来の人工市場シミュレーションのプラットフォームである Plham [12], [13] や PlhamJ [14] においては，Large-scale であることが重要視されており，開発当時の計算リソースにおいては，スーパーコンピューターを使うことを前提と考えられている機能が多々取り込まれていた．例えば，Plham は X10 というスーパーコンピュータ向けの言語で書かれている．しかしながら，近年のコンシューマー向け計算機の性能の向上等を鑑みると，必ずしもそういった大規模計算リソースがなくともマルチエージェントシミュレーションは実現可能である．そこで，PAMS においては，Large-scale という概念を取り払い，Jupyter Notebook などでも簡単に実行可能であることを開発目標に据え，開発を行った．

3.2 利 用 方 法

詳細な利用方法は，Github のページ(注3)や公式ドキュメント(注4)を確認していただきたいが，ここでは，簡単に利用方法を

(注3)：https://github.com/masanorihirano/pams
(注4)：https://pams.hirano.dev/en/latest/

紹介する．

まず，PAMS は図 1 の通り，Python のパッケージ管理ソフトである，pip 等を通じ，Python パッケージレポジトリの PyPi からダウンロード可能である．もちろん，pipenv や poetry 環境でも同様に使用できる．

```
$ pip install pams
```

図 1　PAMS は pip インストール可能である

図 2 は簡易なプログラムの実行例である．Market と FCNAgents というのは，PAMS に built-in のマーケットとエージェントであり，これらを用いると，複雑なクラスのオーバーライドなしでもシミュレーションを簡単に回すことができる．図の中の config は，それぞれのマーケットやエージェントの内部パラメータの設定であり，このようにパラメータを Python の辞書型または外部から json で設定することが可能である．また，Plham と同様に，パラメータに乱数を設定する機能もついており，config から設定可能である．saver は，シミュレーションの特定の結果 (約定や各ステップの価格など) のみを透過的に保存することができる機能を持っており，これを runner に設定することにより，必要な統計値を計算することなども可能である．saver を用いて，各ステップの価格をプロットした図が図 3 となる．

ここでは，簡単な例を示しているが，実際にはもっと複雑なカスタマイズも行うことができる．図 4 に抽象クラス間の関係を示す．PAMS においては，Main から Runner クラスが呼び出される．この Runner クラスは，シミュレーションの処理の順番やエージェントの呼び出しを司るものであり，並列化されたものと，そうではない逐次のものが存在する(注5)．この Runner が Simulator を操作することでシミュレーションは機能する．Simulator は，ある種の仮想世界のようなもので，Market と Agent と Event が存在する．Runner は，Simulator のインターフェスを通じて，これらの操作を行う．その過程で，Logger が呼び出され，シミュレーションの記録が行われる．この Logger は，Main で定義されており，Runner を通じて Simulator に提供されている．

これらの抽象クラスは，どれも拡張可能であり，特に，Simulator 配下の Market や Agent，Event は，ユーザーが自由にルールを記述し，Runner に登録することで使用可能になる．ユーザー定義のクラスの使用の例も作成している (GitHub レポジトリの samples/user_class) ので，そちらも参照されたい．

tick-tiem スケールのシミュレーションを実現するために，シミュレーションの各ステップでは，決められた数 (デフォルトは 1) のエージェントをランダムに呼び出し，その注文行動を行うチャンスを付与する．

このように，tick-time ベースの金融市場シミュレーションとしては汎用的に利用できるようなパッケージとなっている．

(注5)：2023/7 月時点で並列版は未実装

```python
import random
import matplotlib.pyplot as plt
from pams.runners import SequentialRunner
from pams.logs import MarketStepSaver
config = {
  "simulation": {
    "markets": ["Market"],
    "agents": ["FCNAgents"],
    "sessions": [
      { "sessionName": 0,
        "iterationSteps": 100,
        "withOrderPlacement": True,
        "withOrderExecution": False,
        "withPrint": True,
        "hiFrequencySubmitRate": 1.0
      },
      { "sessionName": 1,
        "iterationSteps": 500,
        "withOrderPlacement": True,
        "withOrderExecution": True,
        "withPrint": True
      }
    ]
  },
  "Market": {
    "class": "Market",
    "tickSize": 0.00001,
    "marketPrice": 300.0
  },
  "FCNAgents": {
    "class": "FCNAgent",
    "numAgents": 100,
    "markets": ["Market"],
    "assetVolume": 50,
    "cashAmount": 10000,
    "fundamentalWeight": {"expon": [1.0]},
    "chartWeight": {"expon": [0.0]},
    "noiseWeight": {"expon": [1.0]},
    "meanReversionTime":{"uniform":[50,100]},
    "noiseScale": 0.001,
    "timeWindowSize": [100, 200],
    "orderMargin": [0.0, 0.1]
  }
}

saver = MarketStepSaver()

runner = SequentialRunner(
    settings=config,
    prng=random.Random(42),
    logger=saver,
)
runner.main()
```

図 2　簡易なプログラムの実行例 (Github レポジトリの examples/CI2002.ipynb より抜粋)

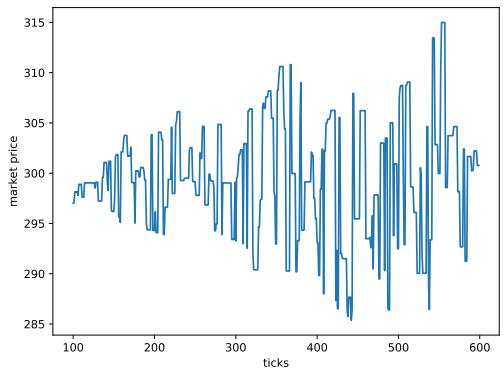

図 3　図 2 で示したシミュレーションの各ステップの市場価格のプロット

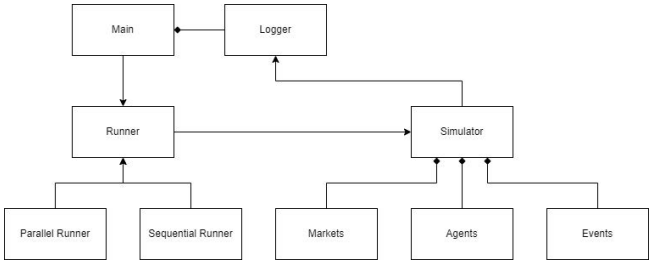

図 4　PAMS の抽象クラス間の関係

4. PAMS と深層学習の融合

次に，本章では，PAMS を実際に深層学習と融合した例を示す．

エージェントシミュレーションでの深層学習の融合としては，エージェントの行動決定プロセスに深層学習を用いるものと，シミュレーションの出力を深層学習に用いるものが考えられる．前者の場合，シミュレーションの内部で深層学習コードを呼び出す必要がある一方で，後者は深層学習のコードからデータ取得のためにシミュレーションを呼び出すという点で逆の系統の実装となる．Python ベースのアーキテクチャを採用していない従来のシミュレーションは後者の実装は比較的に容易に実装可能であったが，前者の実装は，HTTP 通信や RPC などで深層学習モデルを外部から呼び出す必要があり，実装は可能である者の，難しさが存在した．

今回は，特に従前のプラットフォームでは実装難易度の高かった，人工市場シミュレーション内のエージェントの行動決定に深層学習モデルが使用されるケースについて取り組む．

4.1 タスク設定

ここでは，高度な取引戦略の普及に伴う，各トレーダーの収益構造の変化について研究を行う．ここで，高度な取引戦略とは，深層学習を用いた価格予測に基づく取引を指すこととする．深層学習を用いると言っても，価格予測を行うところから始まり，強化学習で取引戦略まで最適化する方法など，様々であるが，今回は，簡便化のため，価格予測に基づいて取引を行う手法のみを対象とする．

本タスクにおいては，2 種のエージェントを想定し，一般的なルールベースの Stylized Agent と深層学習を用いる Deep Agent が存在すると仮定する．この時，深層学習が普及した場合，つまり，Deep Agent の比率が増えた時に，Deep Agent 同士が利益を潰し合い，結果として，収益が獲得できなくなるのではないか，という仮説を立て，これを検証する．

4.2 モデル

本研究では，先行研究 [36] で用いられている，一般的な人工市場モデルを採用した．シミュレーションには 1 つの連続二重オークション形式の市場と n_{sa} 体の Stylized Agent と n_{da} 体の Deep Agent が存在する．全体のエージェント数は一定で，Deep Agent の比率が与える影響を見るために，$n_{\mathrm{sa}} + n_{\mathrm{da}} = 100$ とする．以下では，それぞれのエージェントのアルゴリズムを説明する．

4.2.1 Stylized Agent

時刻 t において，Stylized Agent i は，ファンダメンタル，チャート (トレンド)，ノイズファクターに基づいて，注文行動の決定を行う．まず，エージェントは下記の通り 3 ファクターを計算する．

- ファンダメンタルファクター：

$$F_t^i = \frac{1}{\tau^{*i}} \ln \left\{ \frac{p_t^*}{p_t} \right\}. \tag{1}$$

ここで，τ^{*i} はエージェント i の平均回帰時間のパラメータであり，p_t^* は時刻 t におけるファンダメンタル価格，p_t は時刻 t における価格である．

- チャート (トレンド) ファクター：

$$C_t^i = \frac{1}{\tau^i} \sum_{j=1}^{\tau^i} r_{(t-j)} = \frac{1}{\tau^i} \sum_{j=1}^{\tau^i} \ln \frac{p_{(t-j)}}{p_{(t-j-1)}}. \tag{2}$$

ここで，τ^i はエージェント i の time window size であり，r_t は時刻 t におけるログリターンであり，価格時系列から計算可能である．

- ノイズファクター：

$$N_t^i \sim \mathcal{N}(0, \sigma). \tag{3}$$

ここで，N_t^i は，平均 0，分散 $(\sigma)^2$ の正規分布を意味する．

続いて，エージェントは，上記の 3 つのファクターの重み付き平均を下記のように計算する：

$$\widehat{r_t^i} = \frac{1}{w_F^i + w_C^i + w_N^i} \left(w_F^i F_t^i + w_C^i C_t^i + w_N^i N_t^i \right). \tag{4}$$

ここで，w_F^i, w_C^i, w_N^i は，エージェント i の 3 つのファクターに対する重み付けである．

続いて，エージェント i の期待価格が以下の通り計算される：

$$\widehat{p_t^i} = p_t \exp\left(\widehat{r_t^i} \tau^i \right). \tag{5}$$

そのうえで，固定注文マージン $k^i \in [k_{\min}, k_{\max}]$ を用いて，最終的な注文は以下のように計算される：

- もし，$\widehat{p_t^i} > p_t$ であれば，エージェント i は以下の価格で買い注文を立てる．

$$\min\left\{\widehat{p_t^i}(1-k^i), p_t^{\text{bid}}\right\} \tag{6}$$

- もし，$\widehat{p_t^i} < p_t$ であれば，エージェント i は以下の価格で売り注文を立てる．

$$\max\left\{\widehat{p_t^i}(1+k^i), p_t^{\text{ask}}\right\} \tag{7}$$

ここで，p_t^{bid} と p_t^{ask} は，売りと買いの裁量気配値を意味する．

パラメータとして，先行研究を参考に，$p_t^* = 300, w_F^i \sim Ex(1.0), w_C^i \sim Ex(0.0), w_N^i \sim Ex(1.0), \sigma = 0.001, \tau^* \in [50, 100], \tau \in [100, 200], k^i \in [0.0, 0.1]$ を採用した．

4.2.2 Deep Agent

深層学習ベースのエージェントは，自分の行動チャンスが訪れた際に，LSTM ベースのニューラルネットワークを用いて，過去 100 価格時系列 (現在価格と仲値の 2 系列を使用) に基づいて，将来 100 tick 後の価格の up/down を予測する．この予測モデルの学習には予測時点以前のシミュレーションデータを用い，time windows をスライドさせていくことでデータを作成する．この学習データのうち，最後の 100 個を評価データとして，モデルの評価に用いる．評価データにおける予測精度が 51% を超えた場合にのみ，良いモデルができたものとして，実際に取引に使用する．予測値が up である場合には，買い注文を行うか，すでにポジションが 1 の場合には，そのままとする．予測値が down である場合には，売り注文を行うか，すでにポジションが-1 の場合にはそのままとする．

ニューラルネットのアーキテクチャとしては Hidden state が 32 次元の LSTM で時系列データを処理したのちに，Layer Normalization と ReLU によるアクティベーション層と全結合層からなる 2 層のニューラルネットワークを通じ，予測のロジット値を獲得するアーキテクチャとなっている．

4.3 実　　験

前述の通り，$n_{\text{sa}} + n_{\text{da}} = 100$ という，合計エージェント数は変化させることなく，Deep Agent の数 (n_{da}) を変化させていくことで，どのように収益構造が変化するのかについて検証した．n_{da} は 1 から 20 で変化させる．

それぞれの状況に対して，10 試行行い，各エージェント種別の損益の平均と分散を調べる．

4.4 結　　果

図 5 は，結果のグラフである．この図を見ると，Stylized Agent だけが存在する場合には，ほぼ，ゼロサムゲームになっているものの，Deep Agent が増えてくると，その平均損失は大きくなっていく．一方で，Deep Agent は，Stylized Agent と異なり，平均損益は正であるが，Deep Agent の数が増えるほど，Deep Agent の利益は低減しているように見える．

これらの傾向性についてより正確に分析するため，回帰分析を行った結果が表 1 である．この結果を見ると，Deep Agent が 0 体の時の Stylized Agent の損益平均が 0 であるということが棄却されない一方で，Deep Agent の平均損益が正であることと，Deep Agent の数が増えると，Deep Agent，Stylized Agent ともに平均損失が悪くなっていくことが有意に認められる．

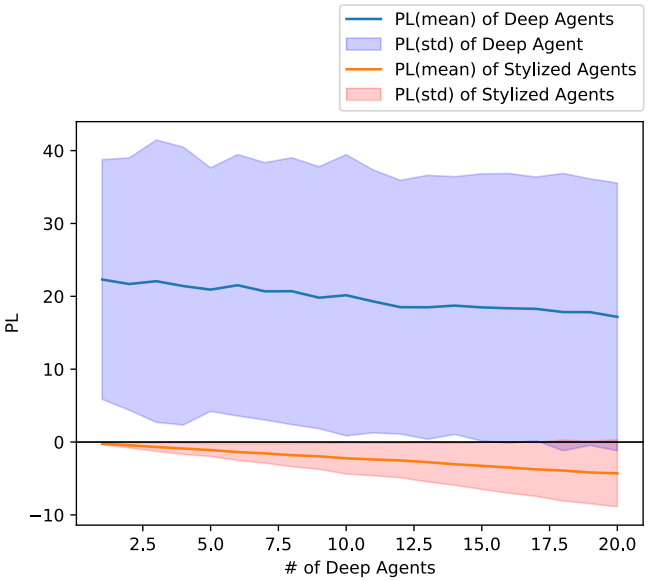

図 5　深層学習エージェントの数を変化させた場合の，深層学習エージェントと一般エージェントの平均損益．

表 1　回帰分析の結果．***は 99.9% の有意水準で 0 でないことを示す．

Metrics \ Agents	Deep Agents	Stylized Agents
Intercept	22.4953***	-0.0294
Coefficient	-0.2649***	-0.2157***
R^2	0.954	0.999
Adjusted R^2	0.952	0.999

4.5 考　　察

まず，実験結果から認められることとして，Deep Agent は，Stylized Agent と異なり，利益を挙げられていると言える．これは，シミュレーション環境であっても，深層学習が適切に特徴をとらえ，上手く取引ができていることを示している．そのため，有効な取引戦略として今回組み込んだ深層学習は，適切に機能していると言え，タスク設定に対して適切な実験になっていると言えるであろう．

また，実験結果から，Deep Agent が増えると，Stylized Agent と Deep Agent ともに平均収益が下がることがわかった．そもそも今回のシミュレーションは，ゼロサムゲームであり，収益性の高い Deep Agent が増えれば増えるほど，Stylized Agent の収益が下がるのは自然なことであると考えられる．一方で，Deep Agent が増えれば増えるほど，Deep Agent の収益が低減することは，必ずしも自明な結果ではない．例えば，同じ取引戦略をとるエージェントが増加した場合に，同方向の注文が増加し，利益を増幅する可能性もありうる．しかしながら，今回は，深層学習を用いているため，ルールベースの取引よりは，多様の注文が発生し，一定の方向性を持った価格変動が起きにくかった可能性がある．

これらの結果と考察を総合すると，Deep Agent のような高度な戦略を持つエージェントの登場は，既存の取引戦略の利益を減少させるだけでなく，高度な戦略を持つエージェント同士でも利益を潰しあってしまうということがいえる．

5. 全体を通した考察

本稿では，新しい人工市場シミュレーションプラットフォームとして，PAMS を示した．PAMS は，深層学習等との融合をスムーズにするために，Python ベースのアーキテクチャを採用しており，実例として示した通り，エージェント内の構造等でも，容易に深層学習を用いることができる．また，ユーザーが簡単にエージェントや環境をカスタマイズ可能になっており，非常に簡単に使用できるようになっている．このような人工市場シミュレーションプラットフォームはこれまでに存在せず，新たな取り組みとなる．

PAMS の可能性はとても幅広い．データマイニングとの融合が容易になったことから，実データを用いて，実市場のデジタルツインのように，より現実的なシミュレーションを実現するための取り組みも可能になるであろう．また，今回のような簡易な深層学習に基づいた取引戦略ではなく，深層強化学習を用いたエージェントを作成することも簡単になるであろう．また，データの可用性も高いため，シミュレーションを Data Augumentation 手法として活用することもできるのではないかと考えている．

今後の発展として，現状の PAMS では，実時間スケールを取り込むことができていない．これは，現状，tick-time ベースと実時間スケールベースの両方を同時に取り込む取り組みがないことに起因しているが，tick-time ベースでも，ひとつ前からの注文間隔さえ注文の特徴量に含むことができれば，実時間スケールをも同時に持たせることができる．しかしながら，現状では，この注文間隔をうまくコントロールすることは難しい．つまり，この注文間隔は流動性の高い時間帯に関しては，トレーダー自身がコントロール可能なものではない．そのため，どのように注文間隔をモデルに取り込んでいくかということについては慎重な検討が必要であると考えている．

最後に，この PAMS が，大きの研究で使われていくことで，更なるフィードバックを獲得し，良いものにしていければ良いと考えており，多くのユーザーによる試行錯誤を期待している．

6. ま と め

本稿では，新しい人工市場シミュレーションプラットフォームの PAMS を示した．この PAMS は，深層学習などとの融合の簡便性の観点から Python ベースのアーキテクチャを採用しており，さらに，オブジェクト指向由来のユーザーのカスタマイズ性も確保してある．この PAMS の有効性を示すために，本稿では，さらに，深層学習をトレーダーエージェント内に実装した研究例を示した．

PAMS はデータマイニング技術や深層学習技術が発展した現在においては，プラットフォームとしての有効性は大きいと考えられ，更なる利用が増えることを期待している．

謝　　辞

本研究は，科研費 21J20074 の助成を一部受けたものである．

文　　献

[1] T. Lux and M. Marchesi, “Scaling and Criticality in a Stochastic Multi-agent Model of a Financial Market,” Nature, vol.397, no.6719, pp.498–500, 1999.

[2] K. Kim, “Financial Time Series Forecasting using Support Vector Machines,” Neurocomputing, vol.55, pp.307–319, 2003.

[3] A.N. Kercheval and Y. Zhang, “Modelling High-frequency Limit Order Book Dynamics with Support Vector Machines,” Quantitative Finance, vol.15, no.8, pp.1315–1329, 2015.

[4] J.A. Sirignano, “Deep Learning for Limit Order Books,” Quantitative Finance, vol.19, no.4, pp.549–570, 2019.

[5] A. Tsantekidis, N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, “Using Deep Learning to Detect Price Change Indications in Financial Markets,” Proceedings of the 25th European Signal Processing Conference, pp.2580–2584, 2017.

[6] M.F. Dixon, N.G. Polson, and V.O. Sokolov, “Deep Learning for Spatio‐temporal Modeling: Dynamic Traffic Flows and High Frequency Trading,” Quantitative Finance, vol.19, no.4, pp.549–570, 2019.

[7] L. Zhang, C. Aggarwal, and G.-J. Qi, “Stock Price Prediction via Discovering Multi-Frequency Trading Patterns,” Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.2141–2149, 2017.

[8] I. Maeda, D. deGraw, M. Kitano, H. Matsushima, H. Sakaji, K. Izumi, and A. Kato, “Deep Reinforcement Learning in Agent Based Financial Market Simulation,” Journal of Risk and Financial Management, vol.13, no.4, p.71, 2020. https://www.mdpi.com/1911-8074/13/4/71

[9] M. Hirano, K. Izumi, and H. Sakaji, “Implementation of Actual Data for Artificial Market Simulation,” Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, pp.1624–1626, 2022. https://doi.org/10.1007/s40844-015-0024-z

[10] M. Hirano, K. Izumi, and H. Sakaji, “Data-driven Agent Design for Artificial Market Simulation,” Proceedings of the 36th Annual Conference of the Japanese Society for Artificial Intelligence, pp.2S4IS2b01–2S4IS2b01, 2022.

[11] M. Hirano and K. Izumi, “Quantitative Tuning of Artificial Market Simulation using Generative Adversarial Network,” The 6th IEEE International Conference on Agents, pp.12–17, 2022.

[12] T. Torii, T. Kamada, K. Izumi, and K. Yamada, “Platform Design for Large-scale Artificial Market Simulation and Preliminary Evaluation on the K Computer,” Artificial Life and Robotics, vol.22, no.3, pp.301–307, 2017.

[13] T. Torii, K. Izumi, T. Kamada, H. Yonenoh, D. Fujishima, I. Matsuura, M. Hirano, and T. Takahashi, “Plham: Platform for Large-scale and High-frequency Artificial Market,” 2016. https://github.com/plham/plham.

[14] T. Torii, K. Izumi, T. Kamada, H. Yonenoh, D. Fujishima, I. Matsuura, M. Hirano, T. Takahashi, and P. Finnerty, “PlhamJ,” 2019. https://github.com/plham/plhamJ.

[15] H. Sato, Y. Koyama, K. Kurumatani, Y. Shiozawa, and H. Deguchi, “U-mart: A test bed for interdisciplinary research into agent-based artificial markets,” Evolutionary Controversies in Economics, pp.179–190, Springer, 2001.

[16] W.B. Arthur, J.H. Holland, B. LeBaron, R. Palmer, and P. Tayler, “Asset Pricing under Endogenous Expectations in an Artificial Stock Market,” The Economy as an Evolving Complex System II, pp.15–44, 1997.

[17] D. Byrd, M. Hybinette, T. Hybinette Balch, and J. Morgan, “ABIDES: Towards High-Fidelity Multi-Agent Market Simulation,” Proceedings of the 2020 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, vol.12, pp.11–22, 2020.

[18] T.C. Schelling, “Models of segregation,” The American economic review, vol.59, no.2, pp.488–493, 1969.

[19] R. Axelrod, “Effective choice in the prisoner’s dilemma,” Journal of conflict resolution, vol.24, no.1, pp.3–25, 1980.

[20] R. Axelrod, “More effective choice in the prisoner’s dilemma,” Journal of conflict resolution, vol.24, no.3, pp.379–403, 1980.

[21] J.M. Epstein and R. Axtell, Growing artificial societies: social science from the bottom up, Brookings Institution Press, 1996.
[22] M. Sajjad, K. Singh, E. Paik, and C.W. Ahn, "A data-driven approach for agent-based modeling: Simulating the dynamics of family formation," Journal of Artificial Societies and Social Simulation, vol.19, no.1, p.9, 2016.
[23] Y. Nonaka, M. Onishi, T. Yamashita, T. Okada, A. Shimada, and R.I. Taniguchi, "Walking velocity model for accurate and massive pedestrian simulator," IEEJ Transactions on Electronics, Information and Systems, vol.133, no.9, pp.1779–1786, 2013.
[24] K. Braun-Munzinger, Z. Liu, and A.E. Turrell, "An agent-based model of corporate bond trading," Quantitative Finance, vol.18, no.4, pp.591–608, 2018.
[25] S. Kurahashi, "Estimating Effectiveness of Preventing Measures for 2019 Novel Coronavirus Diseases (COVID-19)," Proceeding of 2020 9th International Congress on Advanced Applied Informatics, IIAI-AAI 2020, pp.487–492, 2020.
[26] S.M. Edmonds and Bruce, "Towards Good Social Science," Journal of Artificial Societies and Social Simulation, vol.8, no.4, p.13, 2005. https://www.jasss.org/8/4/13.html.
[27] J.D. Farmer and D. Foley, "The economy needs agent-based modelling," Nature, vol.460, no.7256, pp.685–686, 2009.
[28] S. Battiston, J.D. Farmer, A. Flache, D. Garlaschelli, A.G. Haldane, H. Heesterbeek, C. Hommes, C. Jaeger, R. May, and M. Scheffer, "Complexity theory and financial regulation: Economic policy needs interdisciplinary network analysis and behavioral modeling," Science, vol.351, no.6275, pp.818–819, 2016.
[29] R. Axelrod, "The complexity of cooperation," The Complexity of Cooperation, pp.1–248, Princeton university press, 1997.
[30] B. Edmonds and S. Moss, "From kiss to kids–an ' anti-simplistic ' modelling approach," International workshop on multi-agent systems and agent-based simulationSpringer, pp.130–144 2004.
[31] 寺野隆雄, "エージェントベースモデリング: Kiss 原理を超えて (< 特集> 複雑系と集合知)," 人工知能, vol.18, no.6, pp.710–715, 2003.
[32] T. Mizuta, "An Agent-based Model for Designing a Financial Market that Works Well," 2020 IEEE Symposium Series on Computational Intelligence (SSCI), pp.400–406, 2019.
[33] J.-C. Trichet, "Reflections on the nature of monetary policy non-standard measures and financial study," Approaches to monetary policy revisited - lessons from the crisis, eds. by J. Marek, F. Smets, and C. Thimann, pp.12–22, European Central Bank, 2011.
[34] R.M. Bookstaber, The end of theory : financial crises, the failure of economics, and the sweep of human interaction, Princeton University Press, 2017.
[35] W. Cui and A. Brabazon, "An agent-based modeling approach to study price impact," Proceedings of 2012 IEEE Conference on Computational Intelligence for Financial Engineering and Economics, CIFEr 2012, pp.241–248, 2012.
[36] T. Torii, K. Izumi, and K. Yamada, "Shock transfer by arbitrage trading: analysis using multi-asset artificial market," Evolutionary and Institutional Economics Review, vol.12, no.2, pp.395–412, 2015.
[37] T. Mizuta, S. Kosugi, T. Kusumoto, W. Matsumoto, K. Izumi, I. Yagi, and S. Yoshimura, "Effects of Price Regulations and Dark Pools on Financial Market Stability: An Investigation by Multiagent Simulations," Intelligent Systems in Accounting, Finance and Management, vol.23, no.1-2, pp.97–120, 2016.
[38] M. Hirano, K. Izumi, T. Shimada, H. Matsushima, and H. Sakaji, "Impact Analysis of Financial Regulation on Multi-Asset Markets Using Artificial Market Simulations," Journal of Risk and Financial Management, vol.13, no.4, p.75, 2020.
[39] S.J. Leal and M. Napoletano, "Market stability vs. market resilience: Regulatory policies experiments in an agent-based model with low- and high-frequency trading," Journal of Economic Behavior and Organization, vol.157, pp.15–41, 2019.
[40] M. Paddrik, R. Hayes, A. Todd, S. Yang, P. Beling, and W. Scherer, "An agent based model of the E-Mini S&P 500 applied to flash crash analysis," Proceedings of 2012 IEEE Conference on Computational Intelligence for Financial Engineering and Economics, CIFEr 2012, pp.257–264, 2012.