卒業論文

高頻度データに対する
並列分散処理を用いた
ストラテジーの自動チューニング

Automatic Tuning of Strategies
for Multi-Objective Utility Maximization
in High Frequency Data

富山県立大学 工学部 情報システム工学科
2020042 八十住 捺輝
指導教員 António Oliveira Nzinga René 講師

提出年月: 令和 6 年 (2024 年)2 月

目 次

図一覧

表一覧

記号一覧

以下に本論文において用いられる用語と記号の対応表を示す.

用語	記号
インジケーターの計算に使用する期間	n, N, m,
時刻 t の終値	p_t
標準偏差	σ
ボリンジャーバンドのレート	p
ボリンジャーバンドの期間中のレート	$\overline{p}$
過去 m 日間の最高値	p_{max}^{m}
過去 m 日間の最安値	p_{min}^{m}
2^n 直交表に割付けられた主効果と交互作用の効果	θ'
s 番目の効果	θ_s
実数	$\mathbf{R}$
線形模型	$\mathbf{Y}$
計画行列	$\mathbf{A}$
零ベクトル	$\mathbf{0}$
$\mathbf{A}$ の i 番目の列ベクトル	$\mathbf{A}_i$
誤差ベクトル	ϵ
単位行列	$\mathbf{I}$
帰無仮説	$H_0^{(i)}$
未知母数の推定値	$\hat{\theta}$
k 番目の支店の効率性	η_k
k 番目の支店に対する変数	λ_k
k 番目の支店の m 番目の入力データ	x_{km}
k 番目の支店の n 番目の出力データ	$y_{k\,ん}$

第1章

はじめに

§ 1.1　本研究の背景

1996 年の外国為替証拠金取引（Foreign Exchanger: FX）の完全自由化により FX 取引が誕生した．現在では，50 万人を超えるトレーダーがいるとされている．FX は日本で人気のある投資先となっており，24 時間取引可能で手数料が低いことがその理由である．取引者は証拠金（保証金）を業者に預け，為替変動によって利益を得る．差金決済が可能なため，多額の利益を得ることができる一方で，過大な損失を被る可能性もある．そのため, トレーダーは取引ルールを設定し，それに従って取引することでリスクを回避しようとしている．ただし，これらの取引ルールは経験など主観的な要素が多く，トレーダーの実力に左右される．そこで，コンピュータを活用し，自動的にルールに従い取引を行うシステムトレードが導入され，機械的に取引を行う投資家も増加している．さらに，最近の AI ブームからは人工知能を活用し，価格の予測や戦略の獲得に関する研究も進んでいる [1].

§ 1.2　本研究の目的

従来の投資の判断基準として用いられているのが金融市場の要因によって得られた分析結果である．そこで用いられる分析は，過去の市場の動きから指標を算出して未来の市場の動向を予測するようなものである．しかし, 過去の研究の多くは複数ルールは適用していたとしても，効率的にルールを選ぶ方法は組み込まれていない.

また，売買ルールの評価方法もある一つの評価指標に対して比較を行っていることが多かった．それでは偶然そのタイミングだけ評価が高かった場合に適していないルールが選ばれてしまう可能性がある.

そのため本研究では，複数ルールの中から効率的に最適なルールを選ぶこと，かつその際には複数の評価指標を用いて最適なルールの評価を行うことを目的とした分析手法を考える．この目的を達成するために，複数ルールから効率的に最適なものを選ぶために必要な仕組みを考える．本研究では，直交表とデータ包絡分析 (Data Envelopment Analysis: DEA）を用いた為替の自動売買のための分析手法の提案を行う.

まず，為替の取引プラットフォームからリアルタイムの高頻度データを取得し，そのデータを蓄積することで分析に使うヒストリカルデータを作成する．その後，そのヒストリカルデータにおいて直交表をもとにしたルールの組み合わせでバックテストを行い，その結果から複数の評価指標を導出する.

ルール作成に使用した直交表と，得られた評価指標から主効果を導出する．その主効果から全てのルールの組み合わせにおける評価指標の予測値を算出し，DEA をもちいてそれぞれの組み合わせの効率値を導出する．最終的に得られた効率値が最大のものをその後の取引に使用する．以上のような流れで，複数ルールの多目的効用最大化を考慮した市場予測システムを作成する．

提案手法によって構築されたシステムの有効性を検証する必要がある．そのために，実際に以上のことを考慮した分析手法を基に自動売買システムを構築する．そして，構築した自動売買システムによって為替の売買を行い，収益を出すことで本研究の提案手法の有意性を示す．また，本研究では作成した自動売買システムをもちいて，実際にデモ口座を使ったリアルタイム取引を行う．

§ 1.3　本論文の概要

本論文は次のように構成される．

第 1 章 本研究の背景と目的について説明する．背景では金融市場の予測の困難さと金融市場に関する研究の重要性について述べる．目的は多目的効用最大化を考慮した最適なルール選択システムについて提案することを述べる．

第 2 章 為替取引に使用される取引プラットフォームや用語の説明についてまとめる．また，市場の予測に使われる分析手法の例とその最適化に使われるシステムについて述べる．

第 3 章 外界の要因に影響されにくい設計を目的としたロバストパラメータ設計について述べる．また，直交表とそれを用いた実験計画法の例についてまとめる．

第 4 章 提案手法中の Tick データからヒストリカルデータを作成する部分と，DEA を用いた多目的効用最大化を考慮した最適ルール選択の部分について説明する．その後，本研究の提案手法の流れについて述べる

第 5 章 提案手法に基づいて自動売買システムを構築して，運用テストを行う．そして，本研究の提案手法によって得られた結果が有意であることを示す．

第 6 章 本研究で述べている提案手法をまとめて説明する．また，今後の課題について述べる．

第2章

高頻度データ収集

§ 2.1　取引プラットフォーム

取引プラットフォームとは，オンライン投資家が異なる市場でポジションを取引し，注文を管理し，決済するためのソフトウェアである．これにはさまざまな機能や取引ツールが組み込まれており，一部には特定の市場分析ツールも含まれている．オンライン取引プラットフォームを利用することで，世界中の市場にいつでもどこからでもアクセスし，差金取引決済を通じて市場の変動を有効活用できる [2]．取引プラットフォームの例の一つに MetaTrader 5(MT5) がある．

MT5 は，外国為替及び為替市場でのテクニカル分析とトレーディング業務を行うトレーダー向けの無料のアプリケーションである．MT5 では，デモ口座を開設してバーチャルトレードを行うことができる．高機能かつ多様なチャットツールや分析ツール，及び自動売買ツールが搭載されており，リアルな取引に近い体験が可能である．

また，Python を使用して MT5 から Tick データを取得したり，取引の指示を送信したりすることができる．これにより取得した Tick データを活用して自動売買を行うことができる．

以下に FX で取引を行う際に関係する用語の説明を示す [3].

価格

外国為替レート上には 2 つの価格が表示されている．BID（SELL）はトレーダーが売却するときの価格（売値）を示し，ASK（BUY）は購入するときの価格（買値）を表す．また，売値と買値の差額をスプレッドと呼ばれ，このスプレッドが FX 取引における実質的なコストとなる．注文画面での価格表示の例を図 2.1 に示す.

レバレッジ

証拠金を担保にして，その何倍もの金額の取引が可能な仕組みを指すのがレバレッジである（最大レバレッジ 25 倍）．具体的には，預け入れた証拠金に対して最大 25 倍の取引が可能で，これにより大きな取引が可能である．

例えば，米ドル/円が 100 円の場合，通常であれば 1 万米ドルを購入するには 100 万円が必要である．しかし，レバレッジ 25 倍の場合，最低 4 万円の証拠金を預け入れることで，同額の 1 万米ドルを購入することができる．

レバレッジを上げることで，投資資金に比べて大きな金額の取引が可能となり，それにより大きな収益を得る可能性がある．ただし，同時に大きな損失を被るリスクもある．レバレッジの例を図 2.2 に示す.

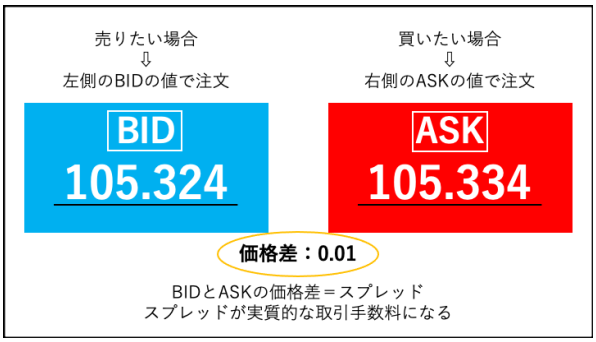

図 2.1: 注文画面の価格表示の例

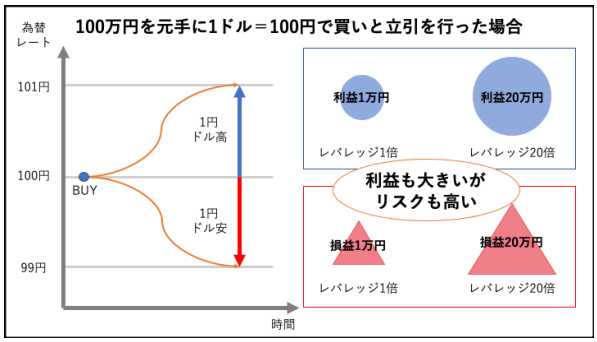

図 2.2: レバレッジの例

強制ロスカット

一定水準以上の含み損が発生した場合，FX 会社が保有ポジションを強制的に決済する制度．FX 取引では預け入れた資金を上回る取引が可能であるため，予測とは逆方向への急激な相場変動により元本を超えた損失が発生する可能性がある．しかし，強制ロスカットというメカニズムにより，通常は最低限の資金が守られる仕組みである．

成行注文

現在の市場価値で即座に取引を行う注文方法．急いで買いたい，または売りたい場合に利用され，初心者にとっても使いやすい注文方法である．ただし，指定した注文が取引サーバに到達した時点のレートで確定されるため，注文価格と確定価格の差が生じる可能性がある．

指値注文

現在の価格よりも有利な価格を指定して取引を行う注文方法．現在の価格を基準により安く買いたいか，高く売りたい場合に使用される．例えば，新規の買い注文では，米ドル/円が 105 円のときに「100 円まで下落したら買いポジションを取得したい」といった場合に利用される．逆に，新規の売り注文では，米ドル/円が 105 円のときに「110 円まで上昇したら売りポジションを取得したい」といった場合に利用される．すでにポジションを持っている場合は，100 円で買ったポジションを 105 円で利益確定するか，105 円で売ったポジションを 100 円で利益確定するなど，利益を確定する際にも利用される．

逆指値注文

現在の価格よりも不利な価格を指定して取引を行う注文方法．現在の価格を基準により高く買いたいか，安く売りたい場合に使用される．例えば，米ドル/円が 105 円のときに買いポジションを保有している状態で，「100 円まで下落したら損失を限定するために売り決済したい」といった場合に利用される．逆に，米ドル/円が 105 円のときに売りポジションを保有している状態で，「110 円まで上昇したら損失を限定するために買い決済したい」といった場合に利用される．

数値注文と逆数値注文の両方について注意すべき点として，現在のレートから離れすぎた価格で注文を設定すると，相場が予想通りに動かなかった場合に注文が確定しない可能性があるため，注文を設定する際には注意する必要がある．

IFD 注文

「If done」の略で，新規注文と利益確定または損切りの決済注文を同時に出す注文方法．新規注文でのみ利用可能であり，例えば，現在の米ドル/円が 105 円のときに，「104 円まで下落したら新規で買い注文を入れて，その後 105 円まで上昇したら決済して利益を確定したい」という場合に利用される．新規注文が成立すると，初めて決済注文が有効になる．

OCO 注文

「One Cancels the Other」の略で，2 つの注文（数値注文と逆数値注文）を同時に予約し，一方の注文が自動的にキャンセルされる注文方法．新規で注文する際だけでなく，既存のポジションに対する決済注文としても利用できる．例えば，米ドル/円が 105 円のときに買いポジションを保有している状態で,「106 円まで上昇したら利益確定のために売り決済したい（数値注文)」「104 円まで下落したら損失を限定するために売り決済したい（逆数値注文)」という場合に利用する．

予想通り値上がりして 106 円の売り数値注文が成立した場合は，104 円の売り逆数値注文が自動的にキャンセルされる．逆に，予想に反して値下がりして 104 円の売り逆数値注文が成立した場合は，106 円の売り数値注文が自動的にキャンセルされる仕組みである．

IFO 注文

IFD 注文と OCO 注文を組み合わせた注文方法で，新規注文と同時に利益確定の数値注文と損失限定の逆数値注文が一括で予約できる注文方法である．これは IFD 注文と同じく，新規注文にのみ利用可能である．例えば，現在の米ドル/円が 105 円の場合,「104 円まで下落したら新規で買い注文（数値注文）し，その後 106 円まで上昇したら利益確定のために売り注文（数値注文）と同時に，103 円まで下落したら損失を限定するための売り注文（逆数値注文）も同時に行う」といった戦略に利用される．

IFO 注文では，新規注文が実現された場合に初めて，利益確定と損失限定の決済注文が有効になる．そして，これらの決済注文のうちどちらか一方が成立すると，もう一方の決済注文はキャンセルされる．これにより，取引者は一連の注文を一括で管理し，市場変動に対する柔軟性を持つことができる．

トレール注文

価格の変動に応じて逆指値の価格を変更していく，逆指値注文の一種である (FX 会社によっては使えない場合がある)．すでに保有しているポジションに対する決済注文として使用した場合，損失を限定しながら利益の最大化を狙うことができる．

例えば，米ドル/円が 105 円のときに買いポジションを保有し，損失限定のため 104 円に売りの逆指値注文を入れた場合を考える．トレール注文では，この 105 円と 104 円の値幅である 1 円がトレール幅となり，相場が上昇し続ける限り，その 1 円幅を保ちつつ逆指値の価格も切り上げる．逆に価格が値下がりした場合は，逆指値注文はそのまま動かない．

価格が上がった分だけ自動的に逆指値の価格も上がるため，相場が一方向に動く局面では相場が反転するまで利益を追及できる．

§ 2.2　インジケーターを用いたテクニカル分析

テクニカル分析の分析対象は，市場内要因や銘柄別要因であることから過去のデータを用いて分析することが多い．過去のデータからテクニカル指標を算出することによって市場の傾向を把握し，この先の未来の値動きについて予測を行う．

テクニカル分析で用いられる指標には様々なものが存在しており，有効な指標の選択はデータの性質や分析対象によって異なる．そして，より精度の高い予測を行うためには複数のテクニカル指標の組み合わせも考慮する必要があると考えられる．

またテクニカル分析では，テクニカル指標によってはワンテンポ遅れる指標があったり，様々な指標の組み合わせを考慮する必要があったりするなど課題は多い．その中でも，テクニカル分析の最大の問題点として，急なニュースや決算の発表や業績の発表によって起こる突然の値動きの大きな変化に対応できないことが挙げられる．

これは，テクニカル分析では過去のレートの情報で指標を算出し，それによって得られた指標は過去の情報量しか持っていないことがこのような急激な変化に対して柔軟に機能することができない原因として挙げられる．

以下に代表的なテクニカル指標と使い方の例を，また，図 2.3 から図 2.6 に EMA，ボリンジャーバンド，MACD，ストキャスティクスそれぞれのグラフを示す．

指数移動平滑平均 (Exponential Moving Average: EMA)

一般的な移動平均線は，単純にある期間における平均から算出したものであるが，EMA では直前の為替のレートを重視するように工夫した指標である．期間 n の単純移動平均 (Simple Moving Average: SMA) は p_t を時刻 t の終値とすると，

$$SMA = \frac{p_t + p_{t-1} + p_{t-2} + \cdots + p_{t-(n-1)}}{n} \tag{2.1}$$

となる．一方，期間 n の EMA は 1 日目は SMA と計算方法は同じだが 2 日以降は，

$$EMA_t = EMA_{t-1} + \alpha(p_t - EMA_{t-1}) \tag{2.2}$$

ここで，

$$\alpha = \frac{2}{n+1} \tag{2.3}$$

そして，EMA で取引を行う際には短期移動線と長期移動線の 2 つを用いる．基本的には，以下のゴールデンクロスとデッドクロスを指標としてルールを作成する．

ゴールデンクロス

短期移動線が長期移動線を下から上へ突き抜けたときに買いサインとなる．

デッドクロス

短期移動線が長期移動線を上から下へ突き抜けたときに売りサインとなる．

ボリンジャーバンド

ボリンジャーバンドは,「標準偏差」と「正規分布」に基づいた考え方であり，68-95-99 ルールを使って取引のルールを決める．これは，正規分布は ±1 標準偏差の中で 68 %

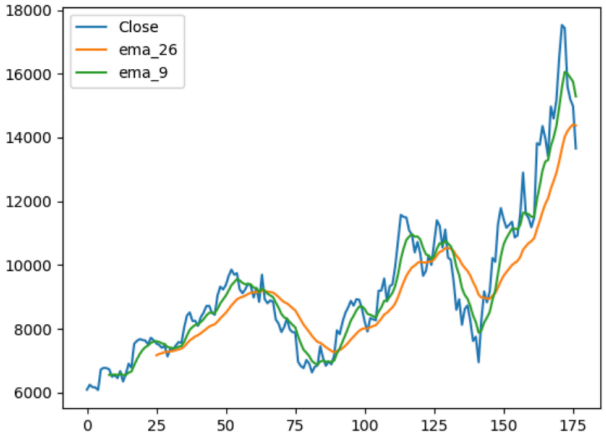

図 2.3: EMA のグラフ

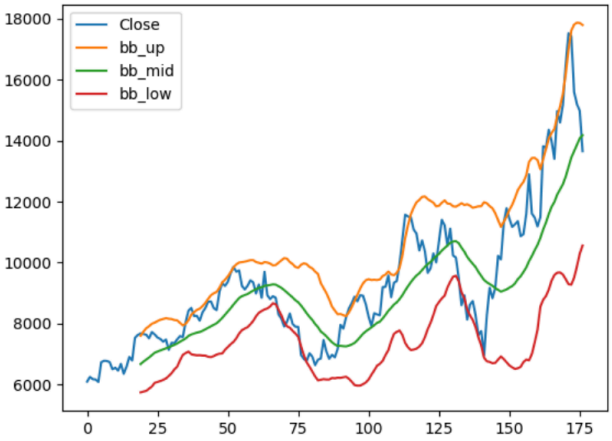

図 2.4: ボリンジャーバンドのグラフ

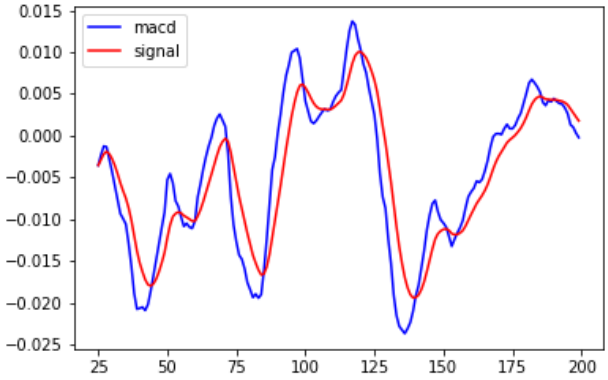

図 2.5: MACD のグラフ

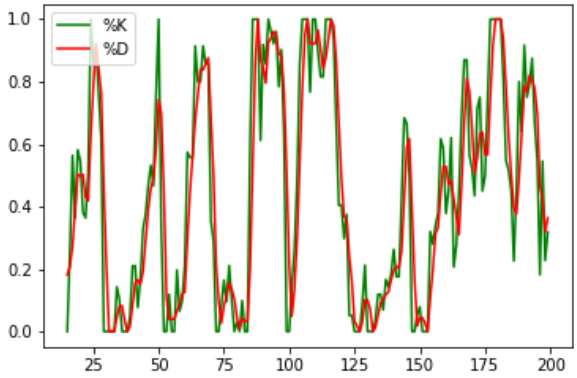

図 2.6: ストキャスティクスのグラフ

の確率で収まり，±2 標準偏差の中で 95 %の確率で収まり，±3 標準偏差の中で 99 %の確率で収まることを前提としている．標準偏差 σ は，N は期間，p はレート，$\overline{p}$ は期間中の平均のレートとすると，

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(p_i - \overline{p})^2}{N-1}} \tag{2.4}$$

この指標を使った取引ルールの例は，レートが -2σ を超えたときに統計上ではこのあと平均付近に戻る可能性が高いことから買いサインとなる．同じように，レートが 2σ を超えたときに統計上ではこのあと平均付近に戻る可能性が高いことから売りサインとなる．

MACD

この指標では,「MACD」,「シグナル」の 2 つの指標を用いて取引ルールを作るが,「MACD」は「ShortEMA」,「LongEMA」の 2 つの指標から算出される．MACD は以下のように求めることができる．

$$MACD = ShortEMA - LongEMA \tag{2.5}$$

また，シグナルは MACD の指数平滑移動平均により求めることができる．

$$signal = MACD_{t-1} + \alpha(p_t - MACD_{t-1}) \tag{2.6}$$

ここで，α は式 2.3 と同じである．取引ルールとしては，MACD がシグナルを上から下へ抜ける時が売りサインとなり，MACD がシグナルを下から上へ抜けている時が買いサインとなる．また，MACD の値がプラスの値からマイナスの値に変換したときに下降トレンドとなり売りサインとなり，MACD の値がマイナスの値からプラスの値になったときは上昇トレンドとなり買いサインとなる．

ストキャスティクス

現在の市場に対して一定期間の変動幅に基づいて，売られすぎているか買われすぎているかを判断するための指標である．この指標では,「%K」,「%D」,「%SD」の 3 つの指標を算出する．

m 日間の%K は，n 日間の%K の単純移動平均，%SD は n 日間の%D の単純移動平均とすると，

$$\%K = \frac{p_t - p_{min}^m}{p_{max}^m - p_{min}^m} \times 100 \tag{2.7}$$

また，%D は，n 日間の%K の単純移動平均，%SD は n 日間の%D の単純移動平均となる．

%K と%D を用いた取引ルールの例は，%K・%D が共に 20%以下の時に、%K が%D を下から上抜いた時が買いサイン，%K・%D が共に 80%以上の時に、%K が%D を上から下抜いた時が売りサインとなる．

表 2.1: 銘柄ごとの的中率

銘柄	的中率（%）：最良	的中率（%）：平均
APC	57	55
BBY	59	57
BP	65	63
CA	63	62
F	69	68
GM	79	77
IBM	82	81
SMRT	67	64

同様に，%D と%SD を用いた取引ルールの例は，%D・%SD が共に 20%以下の時に、%D がスロー%D を下から上抜いた時が買いサイン，%D・%SD が共に 80%以上の時に、%D がスロー%D を上から下抜いた時が売りサインとなる．

テクニカル分析の従来研究

テクニカル指標を用いたテクニカル分析の従来研究で，テクニカル指標の最適な組みわせを学習することによって 10tick 先の価格変動の方向を自動予測する研究がある [?].

この研究では，遺伝的アルゴリズムを用いて指標の組合せを最適化し，得られた指標の組合せを用いて未来の価格の上下を予測する手法を提案している．様々な業種の銘柄の株価の tick データからテクニカル指標をそれぞれ算出し，それらを分析対象とする．また，使用したテクニカル指標を以下に示す

1. 単純移動平均 (MA)
2. 短・長期移動平均 (SLMA)
3. 短・長期指数移動平均 (SLEMA)
4. 移動平均乖離率 (MAD)
5. 移動平均収束拡散指標 (MACD)
6. 順位相関係数 (RCI)
7. 相対強度指数 (RSI)
8. モメンタム (MO1, MO2)
9. サイコロジカルライン (PHL)

これらのテクニカル指標を用いて，予測を行う．予測手法の流れは，テクニカル指標の組み合わせを決め，得られた指標ごとに予測戦略を作成する．そして，テストデータから的中率を算出する．

指標の組み合わせは，遺伝的アルゴリズムにより最適化し，決定木によって作成した予測戦略から結果を求める．結果としては，銘柄によってばらつきがあり，最も良いもので 81 %であった (表 2.1 参照)．銘柄によって得られた tick のデータ数が違い，データ数の多い銘柄ほど精度がよくなるといった結果となった．

他にも，時系列データの予測が得意であるニューラルネットワークを用いて分析を行う研究も行われている [?].

§ 2.3　ストラテジーテスターにおけるバックテストと最適化

バックテストとは，自分が使っている売買ルールが有効であるかを確認するために，ツールを使って過去の相場でシミュレートすることである [?]．主にシステムトレードの分野で使われることの多い言葉だが，裁量トレードの分野でも，売買ルールの有効性を確認するためにバックテストが行われることがある．

一般的に検証というと，自分が使っている売買ルールを現行チャートもしくは過去チャートに照らし合わせ,「実際にその売買ルールで取引してみたらどうなるか」ということを手動で確認することを言う．手動の検証は，ツールなどが必要ない分誰でも簡単に行うことができるが，やはり手動であるために完全ではない．手動の検証には以下のような弱点がある．

1. 検証の精度にブレが出やすい（感情や疲労が精度を下げる）
2. 長期間の検証には向いていない（不可能ではないが膨大な手間と時間がかかる）

裁量トレードがメインであれば，手動で検証を行った方が相場感も養われやすく，メリットであると言える場合もある．しかし，純粋に売買ルールの有効性が知りたい場合は，上にあげた 2 つの弱点は大きなデメリットである．

その点，バックテストは一度ツールを設定してしまえば残りの行程は自動的に行われてしまうので，感情や疲労といった精度を下げる要因を一切排除している．また，機械を使って処理を行なわれるので長い期間の検証でも短時間で終了する．ここが手動の検証との大きな差である．

ある売買ルールを手動で 1ヶ月分検証した場合には利益が出ていたとしても、バックテストで数年分検証してみるとトータルではマイナスになっていたという場合もある．

現在使っている売買ルールが短期間 ”たまたま ”勝てていただけなのか，長期間使ってもきちんと勝てるのかを検証できるという意味でも，バックテストを行うことには大きな意義がある．

売買ルールに用いているインジケーターの中には，計算する際に使用する期間等のパラメータを設定する必要がある．通常はあらかじめ設定したパラメータでインジケーターの計算を行って売買ルールに使用するが，本当にその値が適しているのかどうかはわからない．

そこでパラメータの値を変動させてそれぞれの値でバックテストを行い，より相場の値動きにあった設定に調整する事を最適化と言う．最適化を行うことで複数のパラメータの値の中から一番利益が出る値を探し出すことができるが，全ての値でバックテストを行うことになるため，値の幅の設定が広すぎる場合最適化に時間がかかってしまう．

MT5 ストラテジーテスター

MT5 ではエキスパートアドバイザ（EA）やインジケーターなどのバックテストを行うストラテジーテスターが標準装備されている [?].

ストラテジーテスターでは以下の設定を行いバックテストを実行する．

1. バックテストを行う EA の選択
2. バックテストする銘柄、時間足の選択
3. バックテストを行う期間の選択
4. 指定したバックテストの期間の一部をバックテストに使用し、結果を元にフォワードテスト（検証）を行うかどうかの選択
5. バックテストを行う際に発生すると想定される遅れの選択
6. バックテストを行うデータの精度の種類を選択
7. バックテストを開始する時点での金額，口座のレバレッジを指定
8. 最適化を行うかどうかの選択

設定後ストラテジーテスターを実行するとバックテストが行われる．バックテスト終了後には結果が表示され，損益やドローダウン，プロフィットファクターなどの様々なデータや，時間毎，曜日別，月ごとの取引件数のほか，それぞれの損益の推移等を確認することもできる．また，口座残高の推移をグラフで確認することもできる．

ストラテジーテスターの設定で最適化を行う設定をした場合,「完全アルゴリズム」と「遺伝的アルゴリズム」から選択することができる.「完全アルゴリズム」とは全てのパラメータでテストをする方法で，最も精度が高いですが，テストに費やす時間が長くなる.「遺伝的アルゴリズム」は総当たりのテストではなく，劣勢なものを省いてテストを行うため，完全アルゴリズムと比較すると，ある程度の精度は保ちつつテストの時間を大幅に短縮できる．また，最適化を行う場合に結果のどの部分の最適化を行うか選択する．

次に，パラメータの設定が必要なものは，パラメータの設定を行う．最適化を行わない場合は値の欄にパラメータの数値を入力し，最適化を行う場合は，最小値となる「スタート」，最大値となる「ストップ」，テストを行う間隔「ステップ」を設定する．

パラメータの設定後ストラテジーテスターを実行すると最適化が行われる．最適化終了後，最適なものから順に損益，そのときのパラメータが表示される．また，バックテストの結果のチャート図を確認することもできる．ストラテジーテスターでの最適化結果を図 2.7 に示す．

Python によるバックテスト

Backtesting.py は Python で為替のバックテストを行うことができるライブラリである．ヒストリカルデータを設定して，バックテストを実行し，分析結果を見るというバックテストの基本的な部分がシンプルに作成されているうえに，結果が簡単に一覧表示できるため初心者でも扱いやすいライブラリになっている．

ヒストリカルデータが格納されたデータフレーム，売買ルールが記述されたストラテジークラスを用意するだけで，ヒストリカルデータの期間におけるバックテストを行うことができる．また，ストラテジークラス内に存在する変数の範囲を指定することで，その範囲内で 1 番利益が出る変数の値を最適化して求めることができる．

図 2.8 は Backtesting.py で最適化を行う部分のコードである．変数の最低値，最高値，飛び幅を入力することで，その範囲で一番最適な結果をもたらす変数の組み合わせを求めることができる．

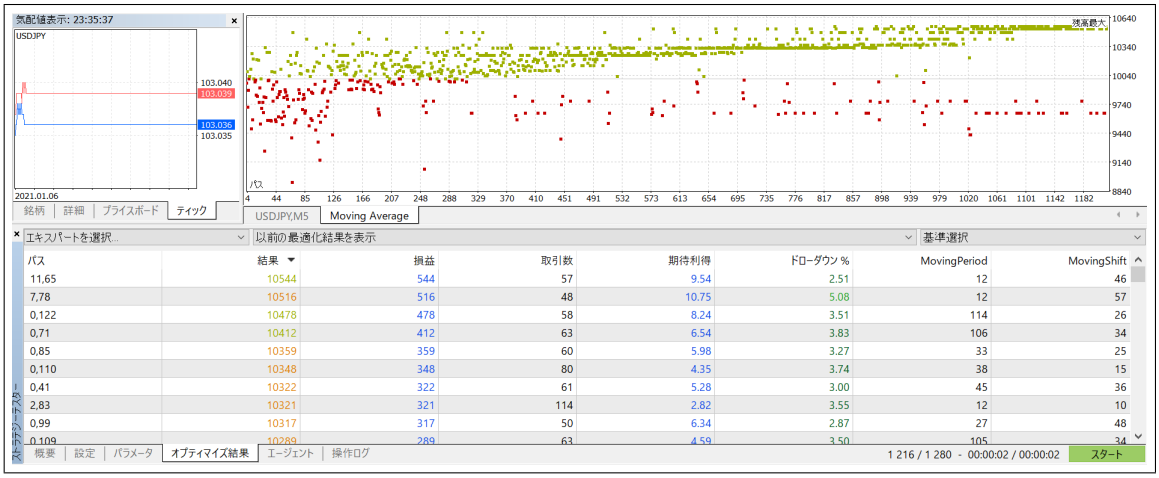

図 2.7: ストラテジーテスターでの最適化結果

```
#EMA最適化
print("----------------EMA----------------")
bt = Backtest(df_, EMAt, cash=10000, commission=.0012, trade_on_close=True)
m=list(range(10, 50, 5))
n=list(range(15, 100, 5))
r=list(range(10, 30, 5))
stats = bt.optimize(m=m,n=n,r=r,maximize='Equity Final [$]',constraint=lambda p: p.m < p.n)
```

図 2.8: Backtesting.py の最適化部分のコード

図 2.9 は Backtesting.py で最適化を行った結果と指標の説明である．Backtesting.py では最初から複数の評価指標について計算を行うことができるため，結果からそのままストラテジーの評価を行うことができる．

パラメータ最適化の従来研究

パラメータの最適化に関する従来研究で，いくつかのテクニカル指標の組合せを用い，過去の学習データから最も利益を上げやすい売買ルールを探索する研究がある [?]．この研究では，遺伝的アルゴリズムを用いて指標のパラメータを最適化し，そのパラメータを用いた売買ルールを使用して未来の価格の上下を予測する手法を提案している．

テスト期間は，2006 年（1/2 12/29）および 2007 年（1/2 12/31）の 2 年間（1 時間足）の USD/JPY レートと EUR/JPY レートとする．また学習期間は，各テスト期間の直前に あたる 2 年間のデータを用いる．また，最適化を行ったパラメータを以下に示す

1. 学習データの RSI の上限・下限
2. 学習データの移動平均乖離率の上限・下限
3. 学習データの直前 1 時間からの上昇 (下落) 率の上限・下限
4. 学習データの指数加重移動平均の RSI の上限・下限
5. RSI で参照する過去時間数
6. 移動平均で参照する過去時間数

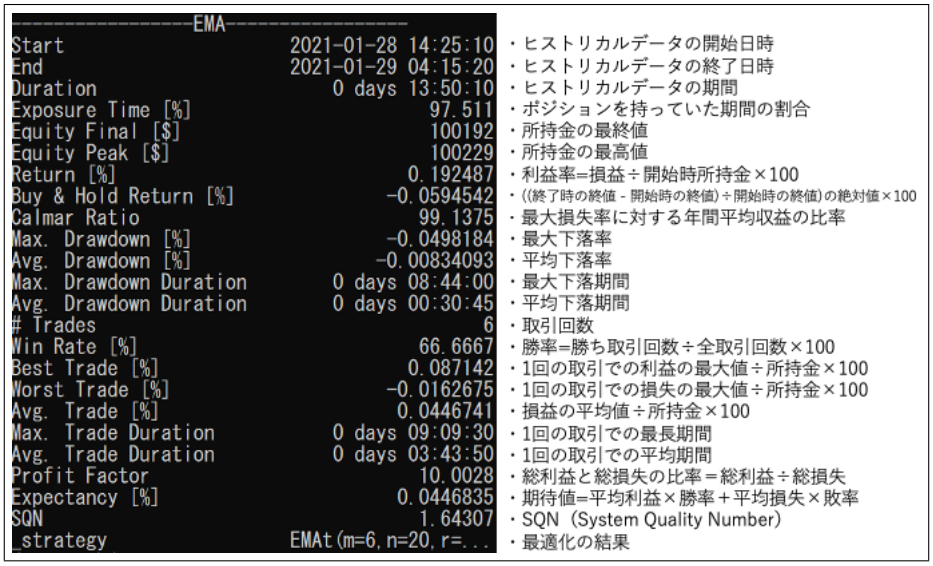

図 2.9: Backtesting.py での最適化結果

表 2.2: 適用レートごとの実験結果

通貨	利益（￥）	初期保有額（￥）	利益率（%）
ドル(2006)	58100	1250000	4.65
ユーロ(2006)	133600	1500000	8.91
ドル(2007)	-43300	1250000	-3.46
ユーロ(2007)	124500	1700000	7.32

7. 加重移動平均で参照する過去時間数
8. 利食い金額
9. 損切り金額

これらのパラメータを最適化し，予測を行う．予測手法の流れは，学習データから各テクニカル指標を計算，遺伝的アルゴリズムによるルール探索，最適な個体をテストデータに適用して予測を行うといったものである．結果としては，2007 年の USD/JPY のみ損失を出していて，それ以外の 3 つは利益率がプラスであった（表 2.2 参照）．

第3章

直交表に基づくロバスト設計

§ 3.1　ロバストパラメータ設計

制御システムを設計する際には，定常特性と速応性，減衰性など過渡特性に着目し，これらの三大機能が設計仕様を満足するようにシステムを構成する必要がある [?]．しかし，実システムでは動特性が正確にわからなかったり，環境や動作条件によって変化するので，時々刻々の動特性を正確に測定し，適切なシステムを設計しなければならない．

制御対象の実際の特性が，制御系設計の際想定したモデルと多少くい違っても制御性をあまりそこなわない制御を，ロバスト制御と呼ぶ [?]．一口でいえばロバスト制御とは「モデルの不確かさを許容する制御」である．有界な外乱を受けてもシステムの特性 (性能) が有界領域に止まるとき，そのシステムはロバストであるという．

実測データとよく合い，しかも設計に使える制御対象のモデルを作ることは非常に困難である．これより，ロバスト制御は制御系設計に課せられるごく自然な要求であり，またモデリングの負担を軽くする道でもある．

パラメータ設計の基本的な考え方は，ばらつきの原因となる誤差因子をコントロールするのではなく，設計に有効な制御因子と誤差因子の交互作用 (誤差因子の影響がなるべく小さくなるような制御因子の水準条件) を見つけることにより誤差因子の影響を減衰させようとするものである．

図 3.1 にロバストパラメータ設計の考え方のイメージ図を示す．2 つの誤差因子に対して，まずパラメータ X の値を変動させることで誤差因子ごとの特性値のばらつきを抑える．その後．パラメータ Y の値を変動させることで目的の特性値に近づけていく．パラメータ設計は，制御因子の水準変更のみでばらつきの低減を図れるという経済的かつ効果的な方法であるため，ものづくりの設計開発の現場を中心に利用されている．

ロバストパラメータ設計の従来研究に，タグチメソッドがある．タグチメソッドとは，品質工学と呼ばれ，田口玄一博士が独自に開発した工学手法である [?]．田口氏は実験計画法を電気通信研究所や企業の実験に適用する中で，かなり初期の時代から顧客の使用条件，使用環境や使用期間を考慮した最適化を行うべきであることを痛感するようになり，現在ではパラメータ設計 (ロバスト設計) と呼ばれるようになった実験方法を試みている．

田口氏は使用条件のばらつき，環境条件のばらつき，時間経過などに影響されない設計条件を得るための実験方法 (パラメータ設計) を試行錯誤的に研究していった．実験計画法的に表現すれば，技術者が構造や種類や値を指定できる制御因子を直交表に割り付け，その外側に誤差因子 (使用条件，環境条件，劣化条件など) と信号因子 (動特性と呼ばれるシステムの入力) を割り付ける，いわゆる直積実験をパラメータ設計のための実験配置として

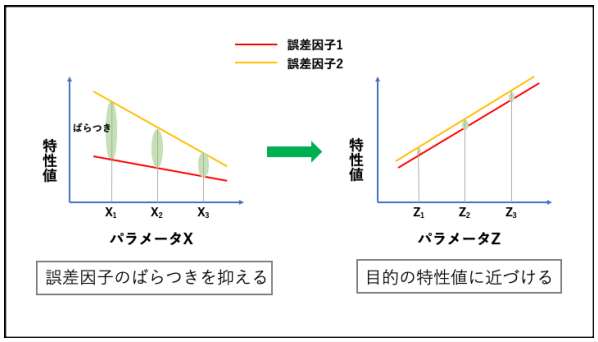

図 3.1: ロバストパラメータ設計の考え方

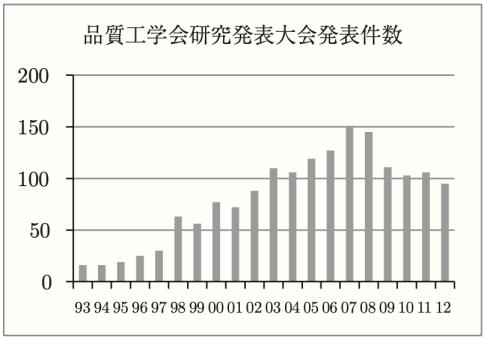

図 3.2: 年度別発表件数

提唱した.

当時の活用は，田口氏が講師を務めたセミナーの受講者，田口氏がコンサルを行っていた企業， 日本規格協会と中品協の両品質工学研究グループのメンバーなど一部に限られ，広く活用されたわけではなかった．しかし，1993 年の品質工学フォーラム (現在の品質工学会) の設立以降，品質工学会での発表件数が急速に増し，日本全体の活用の広がりが推測できるようになった (図 3.2 参照).

他にも，直交表を用いた最適化手法に設計要因の変動を考慮できる機能を組み込み，目的関数の変動が少ないロバスト性に優れた最適解を探索できる手法についての研究がある [?]. 手法の特徴は, 最適解探索ステップ毎の設計要因の変化のさせ方に直交表を活用する点にある．最適解の探索は，探索ステップ毎に直交表の大きさ (探索範囲) をアルゴリズムにより拡大・縮小し，この直交表で示されるケースについて解析を行い，その中から最適解を求めることを繰り返すことにより行う.

また，ロバスト性を評価するために，探索ステップ毎に，各種の誤差要因を公差の幅で変化させたケースの組み合わせにより作成された直交表を用いることで，公差に伴う特性変化の平均値と標準偏差を計算している．この研究では，開発した手法を事業用蒸気タービンの段落計算法に適用している．図 3.3 に全ての公差を 0.5%とした時の各誤差要因の寄与率の計算例を示す．これより，N2 の寄与率が最も大きく，次に N3 の順になっていることがわかる.

図 3.4 に 7 個の誤差要因 (公差) の値を N1 から N7 とし，種々に変化させた場合の 22 ステップ目における効率の平均値と初期効率との差 $\Delta\eta$ と標準偏差値を示す．これより，各要因の公差が大きくなるほど $\Delta\eta$ は小さくなるとともに標準偏差は大きくなることが分かる．また，一般に公差を小さくするほど製品の加工コストは上昇するため，変動に対する寄与率が高い要因の公差だけを小さくし，寄与率が低い要因の公差は，むしろ大きくしてコストを低減しようという意図がある.

例えば，N2，N3 の公差を 0.25%，他の公差を 0.5%とした場合では，全ての公差を 0.5%とした場合よりも標準偏差が 1/10 以下に，また全ての公差を 0.25%としたと比較しても同レベルの $\Delta\eta$ と標準偏差であることが分かる．さらに，前述のように N2，N3 の公差を 0.25%，他の公差を 1.0%とした場合は全ての公差を 0.5%とした場合と比較すると $\Delta\eta$ と標準偏差を大幅に改善していると言える.

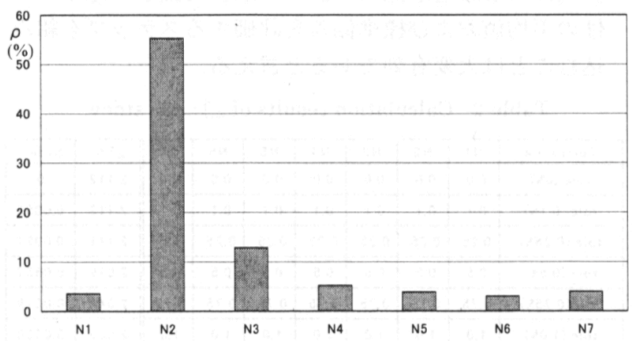

図 3.3: 各誤差要因の寄与率の計算例

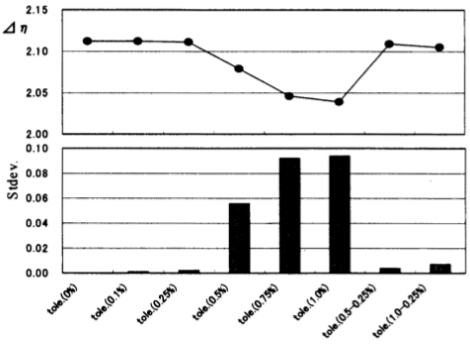

図 3.4: 誤差要因と標準偏差の関係

§ 3.2　実験計画法と直交表

実験計画法とは，文字通り「実験」を「計画」するための方法論ならびにその考え方を統合した学問体系のことである [?]．データ解析では，当然ながらデータの質が最も重要であり，データの質が悪ければ，いくらそのデータが大量にあったとしても，そこから意味のある結論を得ることは困難である．実験計画法は，実験対象の特徴を的確に捉え，その後の分析を円滑に進めるための観測値を得るための方法論の集大成と言える．

実験というと研究室で行われる物理実験，化学実験を思い浮かべる．しかし，それだけでなく，新薬開発の臨床実験や農作物の品種改良のための農事実験，心理学分野の研究もある種の条件を満たせば実験である．また，実験室での実験もその条件を満たさなければ実験ではない．

その条件とは，データを取るための様々な条件の設定を研究者自らができるかどうかという点である．化学実験での実験試料や触媒の種類の選択，化学反応のための温度や時間の設定等が実験の計画者の選択に委ねられているかどうかが問題である．実験の主たる目的は因果関係の確立である．因果関係の原因系は要因もしくは因子と呼ばれ，因子における複数の設定条件を水準という．結果系は特性値や反応，応答と呼ばれる．

実験では，実験に取り上げた因子のほかに特性値に影響を及ぼす別の要因があり，その影響が真に確立したい因果関係と分離できないことがある．このとき，交絡が生じたといい，そのときの要因を交絡要因という．実験では，交絡要因の排除は重要な鍵となる．交絡の例を図 3.5 に示す．

実験計画では，交絡の排除の工夫が必要になる．そして，実験計画では次のような点が問題とされる．

1. それぞれの因子の中で特性値に対して最も影響のある因子はどれか．あるいは逆に影響のないものはどれか．
2. 特性値と各因子との間の関係は具体的にどのように表現されるか．
3. 例えば作物の最大収穫のような，ある意味での特性値の最適解を与える因子の条件の組み合わせは何か．
4. 実験に取り上げた因子以外に特性値に影響を及ぼす要因はないか．あるとすれば，それらの要因の影響はどのようになるか．

そして，これらの諸問題を解決するために，

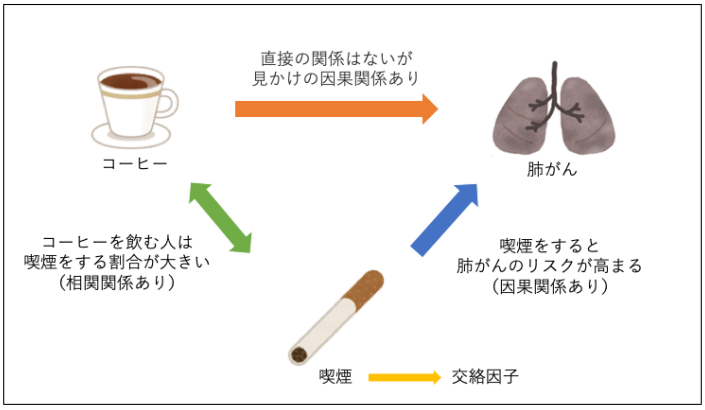

図 3.5: 交絡の例

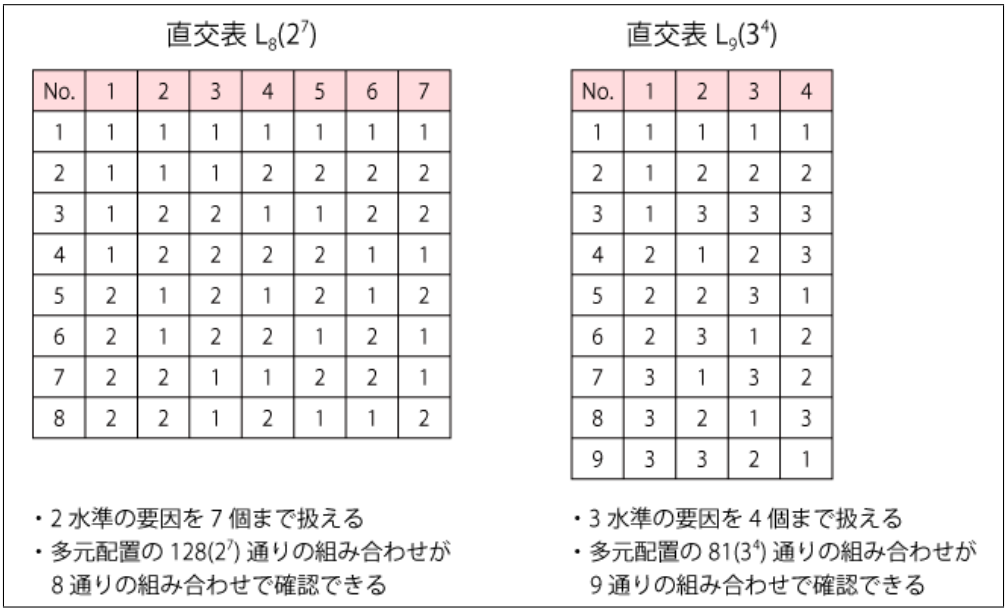

図 3.6: 直交表の例

1. どのような実験をするのが最適か
2. 得られたデータをどのように解析すればよいか

の 2 点についての正しい知識と適切な手法が必要になる．

直交表は図 3.6 のような表である [?]．例えば，図 3.6 の左側の L8 直交表の場合，列 1 から列 7 に要因を割り付け，それぞれの要因の水準を表の 1，2 の数字に対応させることで No.1 から No.8 までの実験条件を設定する．つまり，この表をそのまま計画行列として利用できる．直交表は以下のような性質を持っている．この性質ゆえに，直交表は要因計画に利用できると言える．

1．全ての列において数値の組み合わせが均衡

直交表では，ある 2 列を抜き出したときの数値の組み合わせが，どの列を抜き出しても同じ回数になる．例えば，L8 直交表の列 1 と列 2 を抜き出すと，その値の組み合

表 3.1: L8 直交表の相関

	列1	列2	列3	列4	列5	列6	列7
列1	1						
列2	0	1					
列3	0	0	1				
列4	0	0	0	1			
列5	0	0	0	0	1		
列6	0	0	0	0	0	1	
列7	0	0	0	0	0	0	1

表 3.2: L8 直交表（1 と-1 の 2 水準）

No.	列1	列2	列3	列4	列5	列6	列7
1	1	1	1	1	1	1	1
2	1	1	1	-1	-1	-1	-1
3	1	-1	-1	1	1	-1	-1
4	1	-1	-1	-1	-1	1	1
5	-1	1	-1	1	-1	1	-1
6	-1	1	-1	-1	1	-1	1
7	-1	-1	1	1	-1	-1	1
8	-1	-1	1	-1	1	1	-1

わせのうち (1, 1)，(1, 2)，(2, 1)，(2, 2) が全て 2 回ずつ現れる．他のどの 2 列を取ってもその組み合わせは全て 2 回ずつ現れ，数値の組み合わせの均衡が完全に取れている．この結果，列に要因を割り付け，列の値を水準として考えると，要因と水準の組み合わせを非常にバランスよく実現できる．

2．列同士の相関が全くない

L8 直交表の相関係数の一覧表を表 3.1 に示す．同じ列同士では当然ながら相関係数は 1 となるが，異なる列同士の相関係数が全て 0 となって，全く相関がないことがわかる．列同士の相関が全くないので，列に割り付けた要因の影響が，他の列の要因に影響されることなく解析できる．

3．列同士が直交している

L8 直交表の水準を 1 と-1 で表したものを表 3.2 に示す．ここで各列を 1 つのベクトルとして考える．例えば，列 1 をベクトル表示すると

$$(1, 1, 1, 1, -1, -1, -1, -1)$$

となり，列 2 は

$$(1, 1, -1, -1, 1, 1, -1, -1)$$

となる．この 2 つのベクトルの内積はそれぞれの成分を掛け合わせて合計して求めるため，

$$1 \times 1 + 1 \times 1 + 1 \times (-1) + 1 \times (-1) + (-1) \times 1 + (-1) \times 1 + (-1) \times (-1) + (-1) \times (-1)$$

となり，結果は 0 になる．このように，直交表のどの 2 列を取っても内積は 0 になる．

4．列同士の成分を掛け合わせた値を持つ列が存在する

L8 直交表の列 1 と列 2 の成分をそれぞれ掛け合わせてみる．その結果を見ると，列 1 と列 2 の積は列 3 と全く等しくなる．このように，直交表の全ての列は互いにこのような関係を持ち，ある 2 列の積がいずれかの列の成分になっている．直交表の下に成分表示を付け足したものを以下の表 3.3 に示す．

また，成分表示では同じ文字の 2 乗は 1 として扱う．例えば，列 1 の a と列 5 の ac の積 a^2c は，$a^2 = 1$ としてその積は c となって列 4 に現れる．この性質を利用すると，交互作用を要因と同様に扱うことができる．すなわち，直交表で要因を割り付けた列はその要因の主効果を表し，積として表される列がその交互作用を表す列となる．この結果，直交表はラテン方格と異なり交互作用も扱うことができる．

表 3.3: L8 直交表の成分表示

No.	列1	列2	列3	列4	列5	列6	列7
1	1	1	1	1	1	1	1
2	1	1	1	-1	-1	-1	-1
3	1	-1	-1	1	1	-1	-1
4	1	-1	-1	-1	-1	1	1
5	-1	1	-1	1	-1	1	-1
6	-1	1	-1	-1	1	-1	1
7	-1	-1	1	1	-1	-1	1
8	-1	-1	1	-1	1	1	-1
成分	a	b	ab	c	ac	bc	abc

主効果

主効果とは，それぞれの独立変数がそれぞれ「独自」に従属変数へ与える単純効果のこと．

交互作用

交互作用とは，独立変数を組み合わせた場合の複合効果のこと．特定の結果において要因 A の主効果と要因 B の主効果だけでは説明できない組み合わせ特有の効果がみられること．

§ 3.3　主効果の導出

2^n 直交表に割付けられた因子の主効果と交互作用の効果を $\theta' = [\theta_0, \theta_1, \cdots, \theta_s]^{\mathrm{T}} \in \mathbf{R}^{s\times 1}$, $(0 < s < 2^n = N)$ とする．未知母数 θ' を推定するために，線形模型 $\mathbf{Y} = \mathbf{A}\theta + \epsilon$ を考える．ここで，$\mathbf{Y} \in \mathbf{R}^{N\times 1}$ は観測値ベクトルであり，母数ベクトル $\theta \in \mathbf{R}^{N\times 1}$ は θ' に零ベクトル $\mathbf{0} \in \mathbf{R}^{(N-s-1)\times 1}$ を付加したものであり，

$$\begin{aligned}\theta &= [\theta_0\ \theta_1\ \cdots\ \theta_s\ \theta_{s+1}\ \theta_{s+2}\ \cdots\ \theta_{N-1}]^{\mathrm{T}} \\ &= [\theta_0\ \theta_1\ \cdots\ \theta_s\ 0\ 0\ \cdots\ 0]^{\mathrm{T}}\end{aligned}$$

である．また，θ_0 は一般平均である．計画行列 $\mathbf{A}$ は ± 1 の要素からなる 2^n 型直交行列であり，

$$\mathbf{A} = \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ -1 & -1 & +1 & +1 & -1 & -1 & +1 \\ -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$$

これを

$$\mathbf{A} = [\mathbf{A}_0, \mathbf{A}_1, \cdots, \mathbf{A}_{N-1}]^{\mathrm{T}}$$

とおく．ここで，各ベクトルは $\mathbf{A}_i \in \mathbf{R}^{N\times 1}$ であり，$\mathbf{A}_0 = \mathbf{1}$ である．ベクトル $\mathbf{A}_i,\ (i = 1, 2, \cdots, s)$ は割付けを行った因子 $\theta_i, (1, 2, \cdots, s)$ に対応する直交表の列ベクトルに相当し，ベクトル $\mathbf{A}_i,\ (i = s+1, s+2, \cdots, s_{N-1})$ は任意の順で残りの割付けられていない直交表の列ベクトルに相当させる．誤差ベクトル ϵ は

$$\epsilon \sim N(\mathbf{0}, \sigma^2\mathbf{I})$$

とする．

再び，問題は未知母数 θ を推定することと，

$$\text{帰無仮説} \qquad H_0^{(i)}:\ \theta_i = 0, \qquad (i = 1, 2, \cdots, s)$$

を検定することとなる．

まず，未知母数 θ の推定値を求めることとする．未知母数の推定値は正規方程式

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}\hat{\theta} = \mathbf{A}^{\mathrm{T}}\mathbf{Y}$$

の解である．2^n 型直交行列 $\mathbf{A}$ の任意の 2 つの列ベクトルは互いに直交しており，

$$\mathbf{A}_i^{\mathrm{T}}\mathbf{A}_j = 0$$

それらの内積は

$$< \mathbf{A}_i, \mathbf{A}_j >= \begin{cases} \mathbf{0} & (i \neq j) \\ N & (i = j) \end{cases}$$

となる．そこで，行列 $\mathbf{S} = \mathbf{A}^{\mathrm{T}}\mathbf{A} \in \mathbf{R}^{N\times N}$，$\mathbf{B} = \mathbf{A}^{\mathrm{T}}\mathbf{Y} \in \mathbf{R}^{N\times 1}$ を求めてみると

$$\mathbf{S} = \left[< \mathbf{A}_i^{\mathrm{T}}\mathbf{A}_j >\right] = N\mathbf{I}$$

$$\mathbf{B} = [< \mathbf{A}_0, \mathbf{Y} >, < \mathbf{A}_1, \mathbf{Y} >, \cdots, < \mathbf{A}_{N-1}, \mathbf{Y} >]^{\mathrm{T}}$$

である．

未知母数 θ の最小二乗推定量は

$$\hat{\theta} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{Y} = \frac{1}{N}\mathbf{B}$$

となる．実際には，$\theta_i = 0,\ (i = s+1, s+2, \cdots, N-1)$ であるが，形式的にこれらの値も求めておく．

第4章

提案手法

§ 4.1　データ取得とパラメータの最適化

従来研究の多くは，自身が作成したシステムを過去のヒストリカルデータに適用して予測精度の向上について検証しているものが多い．しかし，過去のデータはあくまでも過去のデータであり，これから先の未来の市場で同じ値動きが発生する可能性は低いと言える．

本研究では，リアルタイムで動いている市場から最新の Tick データを取得・蓄積した後に指定秒足でリサンプリングを行うことで，より直近の値動きから作成されたヒストリカルデータを使用する．その後，最適化したヒストリカルデータを用いてインジケータのパラメータを最適化する．ヒストリカルデータの作成からインジケータのパラメータ最適化は以下のような流れで行う．

1. Tick データ更新ごとに MT5 から Tick データの取得
2. 1 秒間に一回だけ Tick データを保存，データフレームに格納
3. 指定秒足で OHLCV の形にリサンプリングしい，CSV ファイルに保存
4. 作成したヒストリカルデータを用いてインジケーターのパラメータ最適化を行う
5. 最適化したパラメータを CSV ファイルに書き出し保存

まず最初に MT5 から Tick データを取得してくる．取得は Python を使用して行う．MT5 には Python を使ってデータをやり取りするためのモジュールが用意されているため今回はそれを利用した [?]．while 文で永続的にプログラムは動き続け，該当部分のコードに到達するたびに，指定した通貨ペアの Tick の値を MT5 から取得する．取得した Tick のデータ例を図 4.1 に示す．

また，Tick データは同じ時間に複数のデータが送られてくる場合がある．このような事態を避けるために，Tick データは永続的に取得はするが，保存は一つ前のデータから秒数の部分が変更したタイミングだけ行うように設定している．また，取得した Tick データ内に含まれる時間のデータは UNIX 時間であるため，保存する前に datetime 型に変換してからデータフレームに保存している．

さらに，今回の手法では取得してきた Tick データに含まれている BID と ASK の値の平均を取り，その値を今後使う価格の値としている．両方の値を使用しようとした場合，ヒストリカルデータの作成やパラメータ最適化，この後説明するルール選択の処理がそれぞれ 2 倍になってしまうため，処理時間も増えてしまう．本研究ではルール選択にかかる時間はなるべく少なくしたいため，BID と ASK の値の平均値を使うことで 1 つのヒストリカ

```
2021-02-08 10:33:03.278254
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
Tick(time=1612755182, bid=105.526, ask=105.526,
```

図 4.1: 取得する Tick データ

	time	open	high	low	close	volume
1	2021-02-03 20:39:00	105.0580	105.0605	105.0580	105.0585	8.0
2	2021-02-03 20:39:30	105.0595	105.0595	105.0570	105.0575	13.0
3	2021-02-03 20:40:00	105.0570	105.0590	105.0570	105.0570	5.0
4	2021-02-03 20:40:30	105.0580	105.0580	105.0565	105.0565	13.0
5	2021-02-03 20:41:00	105.0565	105.0580	105.0565	105.0580	5.0
6	2021-02-03 20:41:30	105.0575	105.0580	105.0565	105.0580	5.0
7	2021-02-03 20:42:00	105.0570	105.0575	105.0570	105.0575	2.0
8	2021-02-03 20:42:30	105.0565	105.0575	105.0565	105.0570	11.0
9	2021-02-03 20:43:00	105.0565	105.0575	105.0565	105.0575	5.0
10	2021-02-03 20:43:30	105.0575	105.0575	105.0575	105.0575	2.0
11	2021-02-03 20:44:00	105.0565	105.0565	105.0550	105.0565	5.0

図 4.2: 保存後の CSV ファイルの例

ルデータとしている．

また，インジケーターの計算には価格のデータのほかに Volume の値も必要なものも存在する．そのため，保存する Tick データに含まれる Volume を一緒に保存する．つまり，保存先のデータフレームには，Time，Price，Volume の 3 つが保存されていく．

このままでは保存したデータフレームはプログラムが動いている間は溜まり続けるが，プログラムが再度動きなおした際には集めた Tick データは全てリセットされてしまう．それを避けるために Tick データが増えるたびに CSV ファイルに保存し，プログラムの 1 番最初でその CSV ファイルを読み取りに行くことで，今まで貯めたデータを継続して使用できるようにしている．

保存したデータフレームには価格の値は 1 つだけである．しかし，インジケータの計算に使用するデータセットは OHLC 型もしくは OHLCV 型である必要がある．そこで，データフレームの中身を指定時間ごとにリサンプリングを行い，OHLCV 型に変換したデータフレームを作成したのちに CSV ファイルに保存する．

ここで，リサンプリングに指定する時間は 10 秒，30 秒，1 分，3 分，5 分，10 分，15 分の 7 種類用意し，それぞれ別の CSV ファイルに保存する．これは，使用時に様々な時間足のデータセットの中から予測に使うものを選ぶことができるようにするためである．30 秒足にリサンプリング後の OHLCV 型のヒストリカルデータの一部を図 4.2 に示す．

また，保存したデータセットはリサンプリングが行われるたびに最新のものに更新するために書き込みが行われる．また，この後のインジケータの計算に使用するために頻繁に読み込みが行われる．今回複数のインジケータを同一のデータセットを使って計算するため，データセットを保存したファイルがが 1 つだけだと読み書きの際に衝突が起こってしまう恐れがある．これらを避けるために，今回複数の CSV ファイルをインジケーターの数だけ作成し，全て同じデータセットを保存しそれぞれ別の CSV ファイルから読み込みを行うことで衝突が起こることを避けている．

取得した Tick データや作成したヒストリカルデータはプログラムが動き続けている間は常に溜まり続ける上，プログラムを再起動してもリセットされることなく続きから蓄積されていく．このままではデータの数が増えすぎて読み書きの処理が重くなってしまう恐れがあるため，あらかじめ指定したデータ数を超えたら古いものから削除するようにしている．

ヒストリカルデータの作成後，そのデータを用いてインジケーターのパラメータの最適化を行う．今回は 2.3 章で説明した Python でバックテストが行える Backtesting.py というライブラリを使用する．また，インジケーターの計算には TA-Lib というライブラリを使用する．TA-Lib は，複数の言語で使用可能なテクニカル指標の分析ツールで，ヒストリカルデータとそれぞれのインジケーターに必要な期間等の数値を与えるだけでインジケーター

の計算をすることができる．

各インジケーターのバックテストの際にはそれぞれのインジケーターだけでなく，同時に Average True Range（ATR）というインジケータも使用する．ATR は価格変動の度合いを計測するためのインジケーターで，ATR の値が高ければ取れる利幅も大きくなる一方で大きな損失を被るリスクも増大し，ATR の値が低ければ損失を被るリスクは低減するが取れる利幅も少なくなるといった判断ができる．

バックテストの際にはそれぞれのインジケーターごとに売買ルールを設定し，売買タイミングが発生したらオーダーを送る．その際には利確と損切りの幅の値も同時に送り，それぞれの幅分の価格が動いた時点で決済を行っている．今回のシステムではこの利確と損切りの幅に ATR を利用する．

また，ATR の値をそのまま利確と損切りの幅に使用してしまうと幅が小さすぎて決済するまでの時間が早くなり，利確したとしてもスプレッドの方が大きくなってしまう．それを避けるために，ATR の値を数倍した値を利確と損切りの幅に指定し，その倍率も最適化を行う．つまり，今回のシステムで最適化を行うパラメータは以下の 3 点である．

1. 各インジケーターの計算に必要な期間（複数必要な場合もある）
2. ATR の計算に必要な期間
3. ATR の倍率

最適化が終了したらその結果得られた最適なパラメータを CSV ファイルに保存する．インジケーターの最適化を行うプログラムは各インジケーターごとに用意し，それぞれのインジケータごとに最適化が終わり次第新しいパラメータを更新していく．

§ 4.2　多目的効用最大化を考慮した最適なルール選択

Backtesting.py でバックテストを行うと，結果として複数の評価指標の値が表示される．その中から一つを選んで比較して最適なルールを選ぶこともできるが，今回は複数の評価指標を考慮したルール選択を行う．そのために，今回は DEA を用いる．

DEA は多入力多出力系のシステムの効率性を相対的に評価するための手法であり，1978 年に Charnes 等によって提唱された [?] [?]．DEA ではすべての評価対象に対して，評価対象ごとに効率が最大になるように異なる評価基準を用いるのが特徴であり、簡便で広く適用できる利点がある [?].

支店 k の効率性の線形計画は，支店の効率性を η_k，m 番目の入力データを x_{km}，n の出力データを y_{kn}，支店に対する変数 λ_k により双対問題から定式化できる．

$$
\begin{aligned}
&min\ \eta_k \\
&subject\ to\ \lambda_s \geq 0\ (s = 1, 2, 3, \cdots, K) \\
&\qquad \eta_k x_{km} - \sum_{s=1}^{K} \lambda_s x_{sm} \geq 0\ (m = 1, 2, 3, \cdots, M)
\end{aligned}
$$

$$\sum_{s=1}^{K} \lambda_s y_{sn} - y_{kn} \geq 0 \ (n = 1, 2, 3, \cdots, N)$$

これを各支店で定式化し解くことで支店ごとの η を求められる．その後得られた η を比較することで 1 番効率がいいものを選ぶことができる．今回は Backtesting.py で得られる評価指標のうち，値が小さいほど良い指標を入力，値が大きいほど良い指標を出力に指定することで η を導出する．今回提案手法で入力に使用する評価指標は以下の 5 個である．

1. Exposure Time
2. Max. Drawdown
3. Ave. Drawdown
4. Trade
5. Worst Trade

また，出力に使用する評価指標は以下の 10 個である．

1. Equity Final
2. Equity Peak
3. Return
4. Calmar Ratio
5. Win Rate
6. Best Trade
7. Avg. Trade
8. Profit Factor
9. Expectancy
10. SQN

§ 4.3　提案手法のアルゴリズム

最後に，本研究で提案した複数ルールの多目的効用最大化を考慮した自動売買システムのアルゴリズムについてまとめる（図 4.3 参照）．

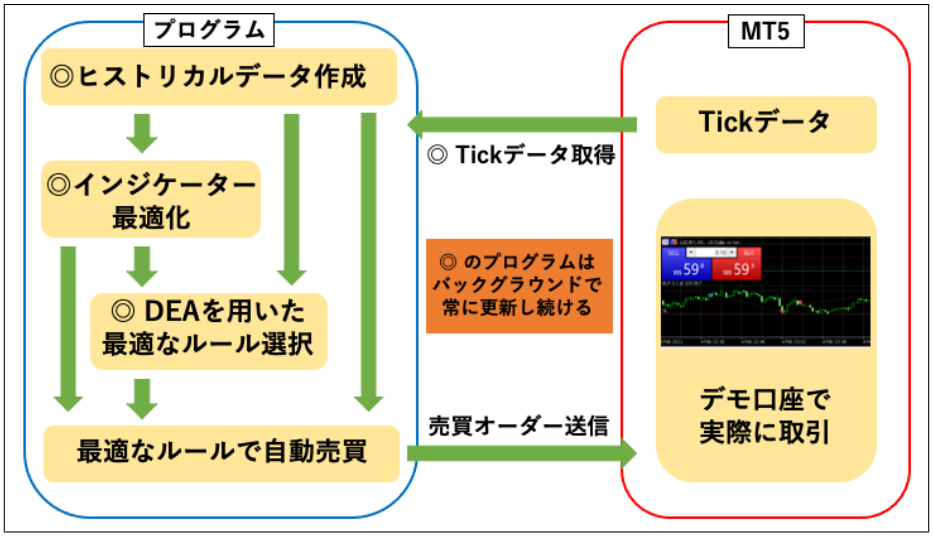

図 4.3: 提案手法の流れ

[Step1] データ取得とヒストリカルデータ作成

Python を用いて MT5 から Tick データを取得してくる．Tick データは永続的に取得はするが，保存は一つ前のデータから秒数の部分が変更したタイミングだけ行う．保存したデータフレームには価格の値は 1 つだけであるが，インジケータの計算に使用するデータセットは OHLC 型もしくは OHLCV 型である必要があるため，データフレームの中身を指定時間ごとにリサンプリングすることで，OHLCV 型に変換したデータフレームを作成したのちに CSV ファイルに保存する．リサンプリングに指定する時間は 10 秒，30 秒，1 分，3 分，5 分，10 分，15 分の 7 種類用意し，それぞれ別の CSV ファイルに保存することで，使用時に様々な時間足のデータセットの中から予測に使うものを選ぶことができるようにする．

[Step2] インジケーターのパラメータ最適化

作成したヒストリカルデータを用いてインジケーターのパラメータの最適化を行う．最適化には Backtesting.py というライブラリを使用し，各インジケーターの計算に必要な期間，利確と損切りの幅に利用する ATR の計算に必要な期間，ATR の倍率のパラメータを最適化する．最適化が終了したらその結果得られた最適なパラメータを CSV ファイルに保存する．インジケーターの最適化を行うプログラムは各インジケーターごとに用意し，それぞれのインジケータごとに最適化が終わり次第新しいパラメータを更新していく．

[Step3] 直交表に基づいたバックテスト

直交表に基づいてインジケーターを組み合わせたルールのバックテストを行う．各インジケーターは最適化を行ったパラメータの値を使用し，それぞれの売買ルールでの利確と損切りの幅はそれぞれのインジケーター用の ATR の値と倍率を使用する．また，売買ルールに則ってエントリーする際には，ポジションは常に一つしか持たず，既にポジションを持っている場合は手放すまでは次のエントリーが入らないようにしている．バックテスト後に複数の評価指標の値が得られるため，それらを指標ごとに保存する．直交表内の全てのルールでバックテストが終了したら，回帰分析を用いて評価指標ごとに各インジケーターの主効果を算出する．

[Step4] DEA を用いた最適なルール選択

インジケーター全ての組み合わせの直交表と評価指標ごとの主効果をまとめた表を用いて全ての組み合わせでの評価指標の予測値を算出する．その後，評価指標の中で値が小さいほど良いものを入力，値が大きいほど良いものを出力として利用して DEA を行い，各組み合わせについて効率値を算出する．全てのルールで効率値を算出した後，効率値が最大のものを抜き出して最適なルールとする．効率値が最大なものが複数あった場合はその中で WinRate が一番高いものを最適なルールとする．

[Step5] 最適ルールでの自動売買

最適なルールが得られたらそのルールに則って自動売買を行う．取引は MT5 のデモ口座を使用して行う．ヒストリカルデータと最適なパラメータから計算したインジケーターの値を利用して，売買タイミングになったら Python を用いて売買オーダーを送る．売買オーダーを送る際には，バックテストを行った際と同じ条件になるように設定し，エントリー後に利確と損切りの幅に達したら決済を行う．

第5章

実験結果ならびに考察

§ 5.1　実験概要および結果

本研究では，4.3 章で説明したように，データ取得とヒストリカルデータ作成，インジケーターのパラメータ最適化，直交表に基づいたバックテスト，DEA を用いた最適なルール選択，最適ルールでの自動売買の 5 つの工程がある．

まず，Pyhton を用いて MT5 から Tick データ収集する．Tick データは取得した Tick データの秒の部分が変わっていたらデータフレーム保存していく．収集した Tick データの一部を図 5.1 に示す．今回は収集した Tick データの中から時間，価格，ボリュームを抜き出して保存している．

価格に関しては BID と ASK の 2 種類が存在しているが，今回使用した Backtesting.py で 2 種類の価格でバックテストするには 2 種類それぞれでヒストリカルデータを作成し，それぞれバックテストを行わなくてはいけない．本研究ではルール選択にかかる時間はなるべく少なくしたいため，BID と ASK の値の平均値を使うことで 1 つのヒストリカルデータとしている．

その後，保存した Tick データを 10 秒，30 秒，1 分，3 分，5 分，10 分，15 分のそれぞれの時間足でリサンプリングし，それぞれ別の CSV ファイルに保存した．また，ファイルの読み書きの際に衝突が起こらないようにするため，それぞれの CSV ファイルを各インジケーター用，最適化ルールのバックテスト用，自動売買用，グラフ描画用で同じファイルを名前を変えて保存している．

また，今回インジケーターの最適化，直交表に基づいたバックテスト，自動売買，グラフ描画にヒストリカルデータを使用する際は，作成したヒストリカルデータを全て使うのではなく，最新のデータから使用する個数を変数で指定できるようにした．これにより作成したデータの全てを読み込むより処理時間を減少させることができる．10 秒足でリサンプルした後の OHLCV のヒストリカルデータを図 5.2 に示す．

ヒストリカルデータを作成後，各インジケーターのパラメータ最適化を行う．インジケーターの最適化はそれぞれインジケータごとに別々のプログラムで実行し，それぞれ最適化が完了するごとに最適なパラメータを更新していく．最適化には Backtesting.py を，それぞれのインジケーターの計算には TA-Lib を使用している．今回使用したインジケーターは以下の 7 つである．

1. EMA
2. ボリンジャーバンド

	time	price	volume
39608	2021-02-05 03:16:13	105.4440	1
39609	2021-02-05 03:16:21	105.4435	1
39610	2021-02-05 03:16:23	105.4445	1
39611	2021-02-05 03:16:27	105.4450	1
39612	2021-02-05 03:16:28	105.4455	1
39613	2021-02-05 03:16:31	105.4450	1
39614	2021-02-05 03:16:43	105.4470	1
39615	2021-02-05 03:16:45	105.4465	1
39616	2021-02-05 03:16:50	105.4480	1
39617	2021-02-05 03:16:51	105.4475	1
39618	2021-02-05 03:16:55	105.4470	1
39619	2021-02-05 03:16:56	105.4465	1
39620	2021-02-05 03:17:01	105.4460	1
39621	2021-02-05 03:17:07	105.4395	1
39622	2021-02-05 03:17:09	105.4390	1
39623	2021-02-05 03:17:12	105.4400	1
39624	2021-02-05 03:17:16	105.4415	1
39625	2021-02-05 03:17:17	105.4420	1

図 5.1: Tick データの一部

	time	open	high	low	close	volume
1	2021-02-05 03:09:40	105.4450	105.4460	105.4450	105.4460	2.0
2	2021-02-05 03:09:50	105.4490	105.4490	105.4490	105.4490	1.0
3	2021-02-05 03:10:00	105.4495	105.4495	105.4495	105.4495	1.0
4	2021-02-05 03:10:10	105.4495	105.4500	105.4490	105.4490	3.0
5	2021-02-05 03:10:20	105.4510	105.4510	105.4510	105.4510	1.0
6	2021-02-05 03:10:30	105.4515	105.4515	105.4515	105.4515	1.0
7	2021-02-05 03:10:40	105.4520	105.4520	105.4515	105.4515	2.0
8	2021-02-05 03:10:50	105.4530	105.4530	105.4530	105.4530	1.0
9	2021-02-05 03:11:00	105.4515	105.4515	105.4515	105.4515	1.0
10	2021-02-05 03:11:10	105.4520	105.4520	105.4485	105.4485	3.0
11	2021-02-05 03:11:20	105.4495	105.4495	105.4490	105.4490	2.0
12	2021-02-05 03:11:30	105.4480	105.4480	105.4480	105.4480	1.0
13	2021-02-05 03:11:40	105.4475	105.4475	105.4475	105.4475	1.0
14	2021-02-05 03:11:50	105.4470	105.4475	105.4470	105.4470	4.0
15	2021-02-05 03:12:00	105.4470	105.4470	105.4460	105.4460	2.0
16	2021-02-05 03:12:10	105.4465	105.4465	105.4465	105.4465	1.0
17	2021-02-05 03:12:20	105.4440	105.4440	105.4430	105.4430	2.0
18	2021-02-05 03:12:30	105.4440	105.4440	105.4440	105.4440	1.0

図 5.2: ヒストリカルデータ (10S)

表 5.1: インジケーターの最適化の際の値の幅

最適化するindi	EMA(短期)	EMA(長期)	BBAND	MACD(短期)	MACD(長期)	MACD(シグナル)	RSI	STOCH(%K)
最小値	6	20	5	8	22	5	8	5
最大値	18	32	17	16	30	13	20	17
飛び幅	2	2	2	2	2	2	2	2

最適化するindi	STOCH(%D)	DMI	ULTOSC(短期)	ULTOSC(中期)	ULTOSC(長期)	ATR	ATR倍率
最小値	3	8	5	10	24	6	9
最大値	15	20	13	18	32	24	21
飛び幅	2	2	2	2	2	3	2

3. MACD
4. 相対力指数（Relative Strength Index: RSI）
5. ストキャスティクス
6. 方向性指数（Directional Movement Index: DMI）
7. アルティメットオシレーター（Ultimate Oscillator: ULTOSC）

また，全てのインジケーターの売買ルールで，オーダー時の利確と損切り幅には ATR というインジケーターを数倍したものを用いている．それぞれのインジケーターの最適化を行なった際の数値の幅を表 5.1 に示す．最適化する際の幅は，一般的に使われている期間から飛び幅 2 ずつで，前後 3 つずつ入るように設定している．ただし，MACD と ULTOSC は変数の数が他より多く，最適化に時間がかかるため，前後 2 つずつ入るような範囲の設定している．

今回は初期所持金 100,000 円，手数料 0.2pips に設定してバックテストを行い，最適なパラメータを決定する．また，評価指標のうち Equity Final が最大になるように最適化を行う．最適化を行った際のプログラムの一部を図 5.3 に示す．最適化が完了したら，得られたパラメータをインジケータごとに別々の CSV ファイルに保存する．

パラメータ最適化後に，直交表に基づいたルールのバックテストを行う．今回は L8 直交表の 7 要因にインジケーターを割り当て，1 と 0 の 2 水準で使用するか使用しないかを判定する．L8 直交表では 8 通りの組み合わせができるため，その 8 通りのルールでバックテストを行う．バックテストの際は最適化したパラメータを各インジケーターのパラメータに設定する．各ルールでバックテスト終了後，評価指標ごとに値を保存する．

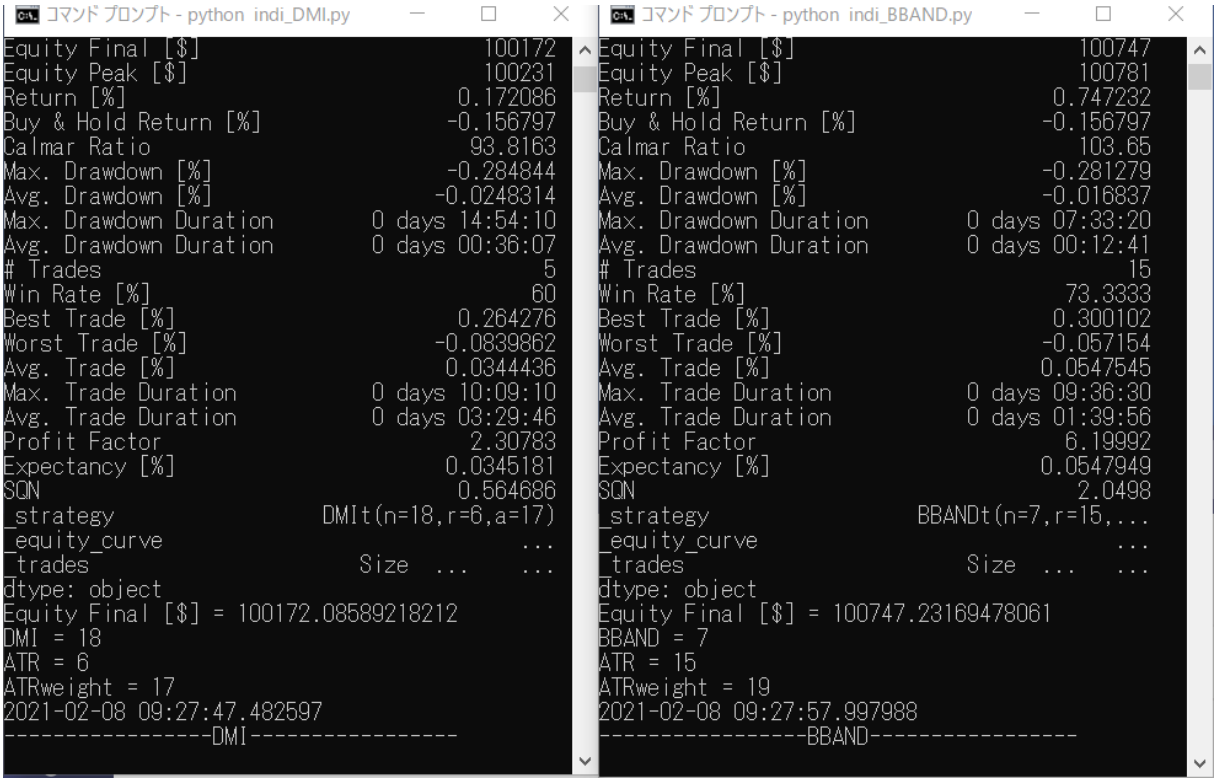

図 5.3: 最適化の際のプログラム例

8 個のルール全てでバックテストを終了したら，ルール作成に使用した L8 直交表とそれぞれの評価指標の列を回帰分析し，インジケーターごとの評価指標に対する主効果を算出する．回帰分析には，Python で機械学習をすることができるライブラリである scikit-learn を使用する．15 個の評価指標について主効果の算出が終了したら，結果を一つの 2 次元配列に保存する．

その後，2_7 通りのインジケーターの組み合わせ全てを表した直交表を作成する．作成の仕方は，0〜127 までの 10 進数を 7 桁の 2 進数に変換し，1 つずつ分割したものを 1 行とし，0 から 127 まで順番に作成していく．これにより，2 次元配列の index 番号と 7 種類のインジケーターの組み合わせを対応していると見ることができる．

この直交表と評価指標をまとめた 2 次元配列を転置したものを掛け合わせることで，全ての組み合わせにおける各評価指標の予測値を算出する．また，DEA を行う際に負の値は設定できないため，全ての値の絶対値を取る．掛け合わせた後の全てのインジケーターの組み合わせにおける評価指標の予測値の表の一部を表 5.2 に示す．

評価指標の予測値が算出できたら，DEA を用いて最適なルール選択を行う．DEA を解くために，Python で線形計画問題を解くことができるライブラリである PuLP を使用する．この評価指標の予測値一覧のうち，4.2 章で示したように評価指標を小さいほど良いものを入力に，大きいほど良いものを出力に指定して DEA を行う．組み合わせひとつずつで双対形の問題について解き，得られた効率値をデータフレームに保存していく．

すべての組み合わせについて効率値の算出が終了したら，先ほど作成した評価指標の一覧の最後の列に，得られた効率値の列を追加する．その後，全ての組み合わせの中から効率値が最大の 1 である行だけを抜き出した新しい 2 次元配列を作成する．今回は，作成し

表 5.2: 評価指標の予測値一覧

	Exposure	Equity Fin	Equity Pea	Return [%	Calmar Ra	Max. Drav	Avg. Draw	Trades	Win Rate [	Best Trad	Worst Tra	Avg. Trad	Profit Fact	Expectanc	SQN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1.34838	108.4638	51.11402	0.108464	19.59655	0.014235	0.035428	55	12.98892	0.101764	0.0505	0.07408	2.642346	0.074044	0.359145
2	0.017361	5.994642	55.0293	0.005995	7.066105	0.045814	0.026453	43	7.087805	0.013722	0.027888	0.034888	0.267101	0.03483	0.030601
3	1.331019	114.4585	3.915288	0.114458	26.66265	0.060049	0.008975	12	5.901116	0.088042	0.022612	0.039192	2.909447	0.039213	0.389746
4	1.09375	106.9412	44.29095	0.106941	15.64568	0.025549	0.027191	1.5	3.206313	0.051466	0.000154	0.020398	1.087876	0.020414	0.388632
5	0.25463	1.522623	95.40496	0.001523	3.950869	0.039784	0.06262	56.5	9.782609	0.050299	0.050345	0.053683	1.55447	0.053629	0.029487
6	1.111111	100.9466	10.73836	0.100947	8.579572	0.071363	0.000738	41.5	10.29412	0.065188	0.028043	0.055286	0.820776	0.055245	0.358031
7	0.237269	7.517265	40.37566	0.007517	11.01697	0.085598	0.036166	13.5	2.694804	0.036576	0.022457	0.018794	1.821571	0.018799	0.001114
8	0.237269	8.092729	40.37566	0.008093	11.01697	0.085598	0.036166	13.5	2.694804	0.036576	0.022457	0.018794	1.821571	0.018799	0.001114
9	1.111111	100.3711	10.73836	0.100371	8.579572	0.071363	0.000738	41.5	10.29412	0.065188	0.028043	0.055286	0.820776	0.055245	0.358031
10	0.25463	2.098086	95.40496	0.002098	3.950869	0.039784	0.06262	56.5	9.782609	0.050299	0.050345	0.053683	1.55447	0.053629	0.029487

```
目的関数算出開始
----------------------------------------------------------
処理にかかった時間
0:00:39.572650
    0
0  75
----------------------------------------------------------
データ読み込み開始
2021-02-05 19:56:02.005483
----------------------------------------------------------
ルール8通りのバックテスト開始
----------------------------------------------------------
評価指標ごとに回帰分析開始
2021-02-05 19:56:18.786905
----------------------------------------------------------
全ての組み合わせに対して係数から指標の値を計算開始
2021-02-05 19:56:18.847388
----------------------------------------------------------
最適なものを選択開始
2021-02-05 19:56:18.847388
----------------------------------------------------------
入出力データフレーム作成開始
----------------------------------------------------------
変数作成開始
----------------------------------------------------------
目的関数算出開始
----------------------------------------------------------
処理にかかった時間
0:00:39.646251
    0
0  53
----------------------------------------------------------
データ読み込み開始
2021-02-05 19:56:58.523807
```

図 5.4: ルール選択のプログラム例

```
----------------------------
2021-02-05 09:13:13.775203
[1 0 1 0 0 1 0]
インジケーター
EMA
MACD
DMI
----------------------------
2021-02-05 09:15:39.935751
[1 0 1 1 1 1 1]
インジケーター
EMA
MACD
RSI
STOCH
DMI
ULTOSC
MACD売ります
time = 2021-02-05 09:16:00
bid = 105.626
spread = 0.000999999999990564
tp & sl = 3.5 pips
----------------------------
2021-02-05 09:18:12.666676
[0 1 1 0 1 1 0]
インジケーター
BBAND
MACD
STOCH
DMI
```

図 5.5: 自動売買プログラム例

た配列の中で評価指標の内の一つである Win Rate が一番大きい組み合わせを最適なルールとして選択する．

一番最適なルールが選択できたら，その行の index 番号を取得することでその番号を最適なルール番号とみなす．取得した index 番号を CSV ファイルに保存し，自動売買の際に使用できるようにする．直交表に基づいたルールのバックテストから最適なルールを選択するまでのプログラムの実行結果の例を図 5.4 に示す．DEA を用いて最適ルールの選択をおこなった結果，最適なルール番号が変わっていることが図 5.4 からわかる．

最適なルール選択が終了したら，そのルールに基づいて実際に自動売買を行う．各インジケーターのパラメータはそれぞれ最適化した値を使用する。また，DEA によって選ばれた最適なルールの値は取得後に 7 桁の 2 進数に変換，それぞれの桁を 1 つずつ格納した配列を作成する．また，最適なルールが変更された際にはその際の 2 進数と使用しているインジケーターの名前を表示する．

タイプ	数量	価格	決済逆指値(S/L)	決済指値(T/P)	価格	損益
sell	0.1	105.626	105.661	105.589	105.620	60
金維持率: 936.04 %						60

図 5.6: 売買オーダー時の MT5

先ほど作った配列の中が 1 か 0 かによってそのインジケーターを使うか使わないかを判定する．売買ルールは各インジケーターのパラメータ最適化を行なった際と同じ売買ルールを使用し，ルールを満たしたタイミングで Python を用いて MT5 に売買オーダーを送る．売買オーダーを送る際にはあらかじめ MT5 のデモ口座を起動しておき，アルゴリズム取引を許可するように設定しておく必要がある．売買オーダーでは以下の 6 つの情報を設定して送る必要がある．

1. エントリーする銘柄
2. ロット数
3. エントリーする価格
4. 利確幅
5. 損切り幅
6. スリップページ

今回は銘柄は USDJPY，ロット数は 0.1，スリップページは 20 で設定している．価格はそのときの最新の Tick データを取得し，売りでエントリーする際には BID の価格で，買いでエントリーする際には ASK の値を使用する．利確と損切りの幅はそれぞれのインジケーターで最適化された ATR の値と倍率をかけたものを使用する．ただし，利確幅は一般に損切り幅よりもスプレッドの値分大きくする．今回はバックテストの際にスプレッドを 0.2pips に設定したので，自動売買の際も損切り幅から 0.2pips 分大きくした値を使用する．

売買オーダーを送った際のプログラムの例を図 5.5 に，その際の MT5 の画面を図 5.6 に示す．今の例では図 5.5 にあるように MACD の売りタイミングで，価格が 105.626，利確と損切り幅が 3.5pips でエントリーしている．このとき，図 5.6 を見るとオーダータイプが sell で，価格が 105.626，決済逆指値（損切り価格）が価格より 3.5pips 高い 105.661，決済指値（損切り価格）が価格より利確幅 3.6pips とスプレッド 0.2pips を足した 3.7pips 低い 105.589 でエントリーしていることがわかる．

このエントリーでは，指値か逆指値に価格が到達するまでポジションを持ち続け，どちらかに到達した段階でポジションを手放す．また，ポジションを持っている間に他のインジケーターでのエントリー条件を満たしたとしても，エントリーは MT5 側に送られないようにしている．

このときの値動きと各インジケーターの値を描画したグラフを図 5.7 に示す．図 5.7 のようにエントリーしたタイミングでメインチャートとエントリーを行ったインジケーターのチャートに縦線が入る．線の色は買いオーダーなら青線，売りオーダーなら赤線である．また，エントリーした際の価格と利確額，損切り額の 3 つについて波線の横線が引かれる．

実際にこのシステムを動かした結果を以下に示す．Tick データは 2021 年 2 月 2 日 13:00 から取得し始め，実際にその後のシステムを動かす際に必要なデータ数が収集できるよう

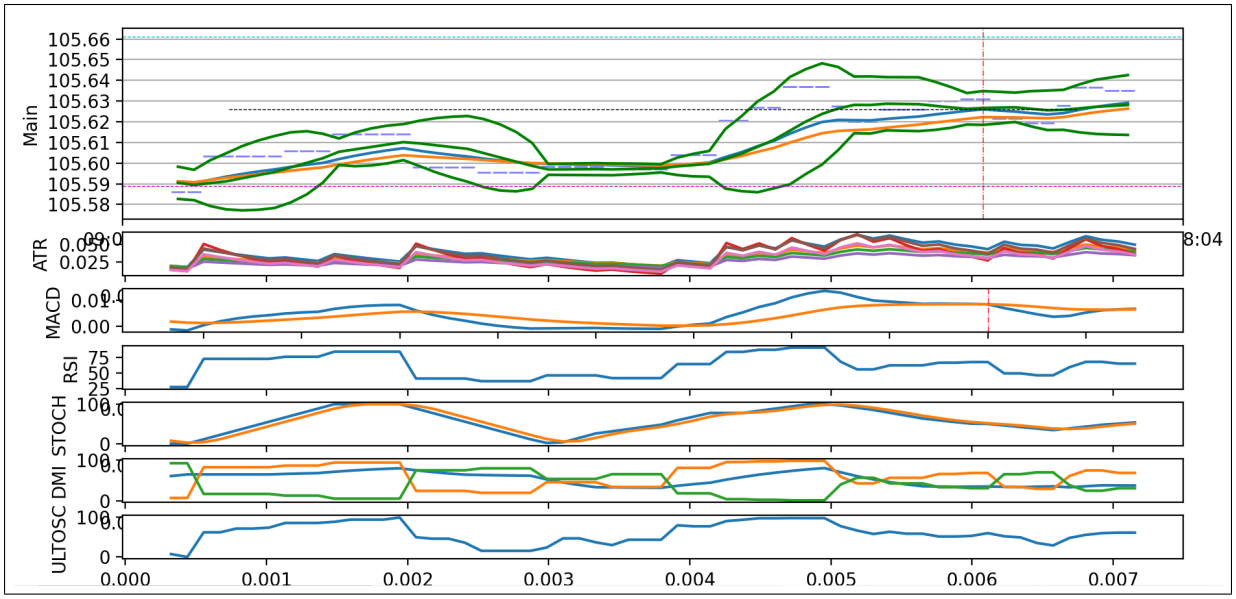

図 5.7: 値動きと各インジケーターのグラフ

にした．インジケーターのパラメータ最適化やルール選択時のバックテスト，自動売買の際に使用するヒストリカルデータには最新のデータから 8,640 個のデータを使用した．これは 10 秒足の際に 1 日分のデータ数となるように設定している．

また，インジケーターの最適化プログラムと直交表に基づく最適なルール選択のプログラムはインジケーターの計算に必要な期間以上のデータ数が蓄積した段階から動かし始め，その後は随時更新し続けて常に最適なパラメータと最適なルール番号を更新し続ける．

上記のプログラムを動かした後，自動売買のプログラムを起動することで自動売買が始まる．今回は自動売買のプログラムを 2/4 14:30 から 2/5 22:00 まで動かした．その際の結果を表 5.3 に示す．実験期間中の取引回数は 28 回，勝率は約 68%で収支は+5000 円であった．

§ 5.2　考察

今回の実験結果より，提案手法で利益を上げることができた．本研究のシステムでは，売買の判断は全てインジケーターの値からの情報のみで判断しているが，勝率も 7 割近くあることから各インジケーターの組み合わせの中で最適なルールを選択できていると考えられる．

また，収支がプラスの場合とマイナスの場合を比較すると，プラスの場合の方が決済時の価格が大きい割合が高く，マイナスの場合はあまり大きな損失につながっていないことがわかる．これは利確と損切り幅にスプレッド分の幅の違いを持たせていることがプラスに働いていると考えられる．

今回の実験の中で，初めの数回は勝率が 100%で取引を行うことができたが，一度価格上昇がマイナスが発生したタイミングから勝率が徐々に落ち始めてしまった．これは市場の騙しに引っかかってしまったり，突発的な値動きに対応できなかったのではないかと考えられる．

表 5.3: 実験結果

時刻	オーダー	収支 (円)
2/4 14:36:06	buy	+360
2/4 15:07:33	buy	+370
2/4 15:31:01	sell	+380
2/4 15:42:46	buy	+330
2/4 16:20:19	buy	+430
2/4 17:02:31	sell	+590
2/4 18:03:55	buy	+570
2/4 19:02:18	sell	+440
2/4 20:01:41	sell	-530
2/4 20:59:08	buy	+370
2/4 22:03:26	buy	+930
2/5 00:46:02	buy	+320
2/5 01:16:28	sell	-510
2/5 01:30:20	buy	-470
2/5 01:58:13	buy	+410
2/5 03:20:39	sell	-420
2/5 05:33:48	sell	-570
2/5 08:09:27	buy	-730
2/5 10:30:51	buy	+610
2/5 10:54:57	sell	+750
2/5 18:14:49	buy	-320
2/5 18:26:14	buy	-480
2/5 18:37:35	buy	+770
2/5 20:09:11	buy	+420
2/5 20:26:38	sell	-370
2/5 20:55:56	buy	+380
2/5 21:10:36	sell	+640
2/5 21:38:53	buy	+330
	総収支	+5000

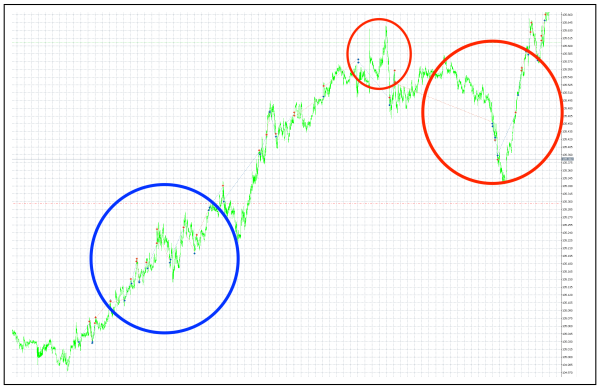

図 5.8: 実験時の 1 分足のグラフ

図 5.9: 実験時の 15 分足のグラフ

図 5.8 は実験時の 1M のチャートである．図中で青丸で囲んでいる部分のように短期の上昇トレンド中は非常に高い勝率で売買を行うことができた．しかし，赤丸で囲んだ 2 箇所のように価格が横ばいになりそうな中で上に抜けそうで抜けない動きや，急激な下落に対応することができず，マイナスの取引になってしまっている．今回使用した 10 秒足では 1 分足よりも値動きは敏感に変化するのでより引っかかりやすかったのではないかと考える．

また 10 秒足のヒストリカルデータを使用していたために，長期のトレンド予測がうまく行えていなかった可能性もある．図 5.8 の中で価格が横ばいになりかけた時点での価格はその日の最高額であり，このチャートだけを見ていれば通常であれば跳ね返されて下降トレンドに転じると予測することもできる．

しかし，このときの 15 分足のチャートを図 5.9 に示す．実際に売買を行なっていた期間は強い上昇トレンドの中にあったことがわかる．これは，アメリカの大統領が交代してからまだ日が経っておらず，バイデン新政権が進めようとしている大型景気対策のニュース等の影響が大きく出ているからだと考えられる．

このような騙しやトレンドの読み違いを避けるためには，複数の時間足で同じシステムを動かした際にどのような差が出るのかの検証が必要であるのではないかと考える．もし比較した結果，各時間足で成績が良い時間帯と悪い時間帯が存在することが確認できたとしたら，同一システム内で時間帯ごとに使用する時間足データを変更して取引を行うことで改善が見られるのではないかと考える．

第6章

おわりに

従来の投資の判断基準として用いられているのが金融市場の要因によって得られた分析結果である．そこで用いられる分析は，過去の市場の値動きからテクニカル指標を算出し，指標の動きから未来の市場の動向を予測するものが多い．しかし，従来研究の多くは複数ルールは適用していたとしても，複数のルールの中から効率的にルールを選ぶ方法は組み込まれていなかった．

また，売買ルールの評価方法もある一つの評価指標に対して比較を行なっていることが多かった．それでは偶然そのタイミングだけ評価が高かった場合と本当に適しているかの判定がしずらいことが問題としてあげられる．そのため，複数の売買ルールの中から最適なルールを選択し，かつその際には複数の評価指標を用いて最適なルールの評価を行うことを目的とした手法を提案する．

本研究では，多目的効用最大化を考慮した最適な売買ルール選択システムの開発を行った．まず，リアルタイムで Tick データを取得し，そのデータをもとにヒストリカルデータを作成，インジケーターのパラメータ最適化を行った．次に，得られたパラメータを使用し直交表をもとにしたインジケーターの組み合わせでバックテストを行い，その結果として複数の評価指標の値を取得，そこから評価指標ごとにインジケーターの主効果を導出した．

その後，主効果から全ての組み合わせにおける評価指標の予測値を算出し，得られた評価指標を入出力に分けて DEA を用いることで組み合わせごとに効率値を導出，最終的に得られた効率値が最大のものを最適なルールとして選択する．最後に，そのルールを用いて実際に自動売買を行なった．

提案手法の有効性を示すために，実際に提案手法のシステムに基づいて売買を行った結果，勝率は約 68%，収支は+5000 という結果を得ることができた．また，短期のトレンド中の予測精度は非常に高いことを示した．

今後の課題として，値動きにおける騙しや急な価格変動を回避することと，長期のトレンドで見た際のトレンドを考慮することが挙げられる．これは，予測に使用するヒストリカルデータの時間足を変更し，比較することで改善することができるのではないかと考える．

また，使用するインジケーターの数を増やすことと，その際に計算処理にかかる時間を短縮することが挙げられる．DEA は全ての組み合わせについて 1 つずつ計算を行う必要があるため，処理に多くの時間がかかる．この処理時間を減少することができたらより多くのインジケーターを用いて売買を行うことができると考える．

謝辞

本研究を遂行するにあたり，多大なご指導と終始懇切丁寧なご鞭撻を賜った富山県立大学電子・情報工学科情報基盤工学講座の奥原浩之教授，António Oliveira Nzinga René 講師に深甚な謝意を表します．最後になりましたが，多大な協力をしていただいた研究室の同輩諸氏に感謝致します．

2021 年 2 月

大谷 和樹

参考文献

[1] 温井 慧，徐 春暉，安藤 雅和，”強化学習を用いた金融市場取引の取引ルール獲得に関する研究”，経営情報学会 全国研究発表大会要旨集，pp. 137-139，2020.

[2] iFOREX，”取引プラットフォーム”，閲覧日 2023-12-20，
https://qr.paps.jp/bZIES.

[3] 木下大輔，”市場間分析を活用した高頻度データに対するパラメータ選択による最適なストラテジー構築”，富山県立大学学位論文，2022.

[4] 内閣府, ”内閣府における EBPM への取組”, 閲覧日 2022-02-08,
https://www.cao.go.jp/others/kichou/ebpm/ebpm.html.

[5] esri ジャパン,” GIS（地理情報システム）とは” , 閲覧日 2022-02-08,
https://www.esrij.com/getting-started/what-is-gis/.

[6] 国土交通省国土地理院, ”基盤地図情報の利活用事例集”, 閲覧日 2022-02-08,
https://www.gsi.go.jp/common/000062939.

[7] esri ジャパン, ”東日本大震災対応における政策形成支援に GIS を活用”,
閲覧日 2022-02-08, https://www.esrij.com/industries/case-studies/35859/.

[8] 田中貴宏, 佐土原聡, ”都市化ポテンシャルマップと二次草原潜在生育地マップの重ね合わせによる二次草原消失の危険性の評価：ー福島県旧原町市域を対象として”, 環境情報科学論文集, Vol. 23, pp. 191-196, 2009.

[9] 坪井利樹, 西田佳史, 持丸正明, 河内まき子, 山中龍宏, 溝口博, ”身体地図情報システム”, 日本知能情報ファジィ学会誌, Vol. 20, No. 2, pp. 155-163, 2008.

[10] 杉原豪, 塚井誠人, ”統計的因果探索による社会基盤整備のストック効果の検証”, 土木学会論文集 D3（土木計画学）, Vol. 75, no.6, pp. 583-589, 2020.

[11] Dentsu Digital Tech Blog, ”Google Colab で統計的因果探索手法 LiNGAM を動かしてみた”, 閲覧日 2022-02-08,
https://note.com/dd_techblog/n/nc8302f55c775.

[12] 藤井秀幸, 傅靖, 小林里佳子, ”データ包絡分析を用いたふるさと納税の戦略提案-K 市のふるさと納税への適用事例-”, 日本経営工学会論文誌, Vol. 71, No. 4, pp. 149-172, 2021.

[13] 刀根薫, ”包絡分析法 DEA”, 日本ファジィ学会誌, Vol. 8, No. 1, pp. 11-14, 1996.

[14] 金成賢作, 篠原正明, ”DEA における入力指向と出力指向の比較（その 1）”, 日本大学生産工学部第 42 回学術講演会, 2009.

[15] 日本オペレーション・リサーチ, ”第 4 章 包絡分析-入力と出力と”, 閲覧日 2022-02-08,
http://www2.econ.tohoku.ac.jp/ ksuzuki/teaching/2006/ch4.

[16] pork_steak, ”folium 事始め”, 閲覧日 2022-02-08,
https://qiita.com/pork_steak/items/f551fa09794831100faa.

[17] 保母敏行ほか, ”日本分析学会における標準物質の開発”, 日本分析化学会誌, vol. 57, No. 6, pp. 363-392, 2008.

[18] 射水市役所, ”総合戦略-射水市”, 閲覧日 2022-02-08,
https://www.city.imizu.toyama.jp/appupload/EDIT/054/054185.

[19] 射水市役所, ”共通課題-射水市”, 閲覧日 2022-02-08,
https://www.city.imizu.toyama.jp/appupload/EDIT/024/024383.