Chisiki氏による二数列分解管理モンテカルロ法の分析

1. 序論

1.1 背景

ギャンブルや投資の領域において、資金管理システムはリスクを制御し、成果を最適化するための重要なツールとして認識されています。特に、モンテカルロ法に代表されるような数列ベースのシステムは、損失回復と利益確保を目指すための体系的なアプローチとして広く知られています。これらのシステムは通常、一連のベットを通じて事前に設定された目標を達成しようと試みます。

1.2 本手法の紹介

本稿では、Chisiki氏によって執筆され、2023年9月24日に最終更新されたnote.comの記事で提示されている「二数列分解管理モンテカルロ法」と呼ばれる新しい資金管理手法について解説します 。この手法は、従来のモンテカルロ法とは一線を画す、独自のメカニズムを取り入れたアプローチとして注目されます。

1.3 本稿の目的

本稿の目的は、提供された情報源の要約に基づき、この「二数列分解管理モンテカルロ法」の原理、運用手順、独自のルール、そしてその根底にあるロジックを明確かつ分析的に説明することにあります 。手法の有効性や収益性の評価ではなく、その仕組みの理解に焦点を当てます。

1.4 著者について

この手法の考案者であるChisiki氏は、中学時代から投資に関与してきた経験を持つ音大生であると紹介されています。この背景は、本手法が純粋な数理的最適化の追求だけでなく、実践的な経験や直感、あるいは心理的な側面への配慮から生まれた可能性を示唆しています。

2. 中核原理と独自の特徴

2.1 基本構造：二数列の同時管理

本手法の最も基本的な特徴は、単一の数列を管理する従来のモンテカルロ法とは異なり、二つの数値列を同時に管理・運用する点にあります 。この二元的な構造が、後述する様々な独自機能の基盤となっています。

2.2 主要な差別化要因 1：損失時のストック獲得

本手法における最も顕著な独自性は、ベットに敗北（LOSE）した場合でも「ストック」を獲得できる仕組みを導入している点です 。従来の多くの資金管理法では、損失は単に回収すべき対象、あるいは

次のベット額を増加させる要因として扱われます。しかし、本手法では損失が「ストック」という新たなリソースを生み出す契機となり得ます。
このルールは、損失に対する力学を根本的に変える可能性があります。単に負債が増加するのではなく、損失によって将来の運用に利用可能なリソース（ストック）が得られるという考え方は、いくつかの側面から解釈できます。第一に、連敗中の心理的負担を軽減する効果が期待されます。損失にもかかわらず何らかの「獲得」があるという感覚は、精神的な安定に寄与するかもしれません。第二に、獲得したストックは、後述する平均化プロセスや「0」生成といった他のシステム管理機能の原資となる可能性があります。第三に、もしストック獲得が数列の値の増加を部分的に相殺するような仕組みであれば、損失が連続した場合のベットサイズの急激な増大を抑制する効果を持つかもしれません。これは、より低いボラティリティやリスクと引き換えに、回復速度が緩やかになる可能性を示唆します。ただし、この「ストック」自体の管理（獲得、消費、価値）が必要となり、システム全体の複雑性は増します。

2.3 主要な差別化要因 2：数列間でのストック共有

獲得または保持されている「ストック」は、二つの数列間で共有されると説明されています 。これは、各数列が独立したストックを持つのではなく、プールされたリソースとして管理されることを意味します。
ストックを共有する設計には、いくつかの意図が考えられます。もし各数列が独立したストックを持てば、一方の数列が他方よりも多くの損失（ストック獲得）を経験した場合、リソースの偏りが生じる可能性があります。共有メカニズムは、このような不均衡を防ぎ、システム全体としての安定性を優先するアプローチと言えます。必要に応じて、より逼迫している数列の管理に共有ストックを柔軟に割り当てることが可能になるかもしれません。また、一方の数列が極端に膨張したり、逆にリソース不足に陥ったりするリスクを低減する効果も期待できます。これは、二つの数列を単なる並列処理ではなく、相互に連携する一つの統合システムとして運用しようとする思想の表れと考えられます。

2.4 主要な差別化要因 3：数列値の平均化

本手法では、二つの数列に含まれる値を定期的に平均化するプロセスが組み込まれています 。平均化が実行される具体的なタイミングやトリガー条件は要約からは不明ですが、この操作が存在することが明記されています。
平均化は、二つの数列の状態を互いに近づける収束メカニズムとして機能します。この操作の目的としては、リスクの低減が挙げられます。一方の数列が特異な連勝や連敗によって他方から大きく乖離することを防ぎ、ベットサイズやリスクエクスポージャーの偏りを抑制しようとする意図がうかがえます。また、システム全体の進行をより滑らかで安定したものにすることを目指しているとも考えられます。さらに、各数列の特定の勝敗履歴（パス依存性）がシステム全体に与える影響を緩和する効果も期待できるでしょう。平均化は、システムの均衡を維持するための意図的な介入であり、ボラティリティを抑制する可能性がある一方で、一方の数列における好調な流れ（連勝など）の影響を希薄化させる可能性もあります。

2.5 主要な差別化要因 4：「0」の生成と再配布

特定の条件下（要約では詳細は不明）で、数列内に「0」を生成し、その「0」が持つ（あるいは表す）価値を残りの数列要素に再配布するという、非常にユニークなルールが存在します 。
これは、標準的な数列管理システムには見られない操作であり、数列のリセット、整理、あるいは統合ツールとして機能する可能性があります。「0」を生成する行為は、おそらくベット単位や負債の一部を数列から取り除くことを意味します。そして、その価値を再配布する（詳細は後述）ことは、数列の根本的な目標額や構造を維持しつつ、より管理しやすい形に修正しようとする試みと考えられま

す。このような操作が導入された理由としては、数列が長くなりすぎて扱いにくくなることの防止、通常の勝利では解消が困難になった数列への対処、あるいは定期的な数列構造の簡素化などが考えられます。この「0」生成・再配布ルールは、システムの長期的な挙動に大きな影響を与える要素であり、そのトリガー条件と具体的な再配布ロジックの理解が、本手法を把握する上で極めて重要となります。これは、標準的なシステムと比較して、非常に積極的な介入を伴うルールと言えます。

3. 運用手順

3.1 概要

元の記事では、各ベットの結果（WINまたはLOSE）に応じて数列を管理するための具体的なステップバイステップの手順が詳述されていると報告されています。これらの手順には、前述した独自の機能（ストック獲得、平均化、「0」生成など）が組み込まれていると考えられます。

3.2 WIN（勝利）シナリオの手順

ベットに勝利した場合に実行される一連のアクションが記述されています。これには通常、以下のようなステップが含まれると推測されます。

- 勝利処理: 数列の両端の数字を消去するなど、モンテカルロ法の基本に則った処理、および利益の計算。
- 平均化ステップ: 勝利が平均化のトリガー条件を満たす場合に実行される可能性。
- ストック消費/調整: 勝利後にストックを使用または調整するルールが存在する場合の処理。
- 「**0**」生成条件の確認: 勝利時に「0」生成の条件が満たされる可能性は低いかもしれませんが、ルールによってはチェックが必要となる場合も考えられます。

3.3 LOSE（敗北）シナリオの手順

ベットに敗北した場合に実行される一連のアクションが記述されています。このシナリオでは、本手法の独自性がより顕著に現れると考えられます。

- 敗北処理: ベット額またはそれから派生した値を数列の末尾に追加するなど、損失を数列に反映させる処理。
- ストック獲得: このステップにおいて、敗北にもかかわらずストックが獲得される具体的なプロセスが実行されます 。
- 平均化ステップ: 敗北が平均化のトリガー条件を満たす場合に実行される可能性。
- ストック消費/調整: 獲得したストック、あるいは既存のストックに関する処理。
- 「**0**」生成条件の確認と再配布: 敗北が「0」生成のトリガーとなり得る場合、その生成と再配布のプロセスが実行される可能性。

これらの手順の存在は、本手法が構造化されている一方で、運用上の複雑性が高いことを示唆しています。各ベットの結果が、数列の変更、ストックの調整、平均化の実行、「0」生成ルールのチェックといった、連鎖的なアクションを引き起こす可能性があるためです。これは、単純な数列操作だけでなく、常にルールに基づいた多面的な調整を必要とするシステムであることを物語っています。したがって、この手法を正確に実行するには、定められた操作順序 への細心の注意と厳格な遵守が不可欠であり、手順の誤りはシステムのパフォーマンスに重大な影響を与える可能性があります。

4. 特殊な条件とルールの処理

4.1 シナリオ：二数列同時敗北かつストックゼロ

本手法では、二つの数列に関連するベットが同時に敗北し、かつ共有されている利用可能なストックがゼロであるという、特定の厳しい状況に対処するための管理プロトコルが詳述されています。
これは、システムにとって重大なストレステストとなるシナリオです。両方の数列で同時に損失が発生することは、二数列構造の有効性を試す状況であり、さらにストックがゼロであることは、「損失時のストック獲得」という特徴によって提供されるバッファー機能が存在しないことを意味します。このような状況でシステムがどのように応答するのか（例えば、より厳しい条件で数列に値を追加するのか、強制的に「0」を生成するのか、あるいは一時停止するのか）は、その耐性を示す重要な指標となります。この特定の危機的状況に対する専用のルール が存在するという事実は、考案者がこの脆弱性を予見し、対応策を設計したことを示唆しており、管理された（ただし複雑な）システムであるという印象を強めます。このルールを理解することは、深刻な資金減少局面に対する本手法の堅牢性を評価する上で鍵となります。

4.2 「0」生成と再配布メカニズム

前述の「0」生成ルールについて、元の記事では生成された「0」がどのように残りの数列要素に再配布されるかについて、具体的なルールと例を挙げて詳しく解説されていると報告されています。
このルールは、本手法の長期的な数列管理と、潜在的な回復力学の中心的な要素です。再配布に関する詳細なルールと例が存在する という事実は、これが単純な操作ではなく、特定の計算ロジックに基づいていることを示唆しています。「0」の価値は比例配分されるのか、均等に割り振られるのか、あるいは特定の数値に加算されるのか？ それは数列の長さや値に依存するのか？ このメカニズムは、おそらく数列の根底にある財務目標（回収すべき損失額など）を維持しながら、数列を整理・統合することを目的としていると考えられます。このメカニズムは、要約情報だけからは最もユニークで不透明な部分であり、その正確なロジックが、システムがどのように自己リセットまたは再構築されるかを決定し、将来のベット計算と回復経路に大きな影響を与えます。
この複雑で非直感的な概念を理解するために、以下に「0」の再配布がどのように機能するかを示す概念的な表を示します。これは、元の情報源で例が提供されているという記述 に基づくものであり、具体的な数値は仮定です。
表**1**：「**0**」再配布の概念的例示

シナリオ記述	「0」生成前の数列 (例)	生成された「0」(とその価値)	再配布後の数列 (例)	関連ルール (出典: S48-S65)
特定のトリガー条件充足 (仮定)	数列A:	数列Aで「0」生成 (価値30)	数列A:	(元の記事参照)
	数列B:		数列B:	
説明	仮に、数列*A*の末尾*30*が「*0*」となり、その価値が残りの*10*と*20*に均等に再配布された場合。			

注意*:* 上記はあくまで概念を示すための架空の例です。実際のルールと計算方法は元の記事 を参照する必要があります。
この表は、再配布ルールの抽象的な説明を読むだけでなく、具体的な「前」と「後」の数列を見ることで、このユニークなルールの具体的な影響を視覚的に理解する助けとなります。

4.3 数列解消と手法の移行

二つある数列のうち、一つが成功裏に解消（クリア）された場合のプロトコルも定義されています。この状況が発生すると、システムは「オリジナルグッドマン法」（恐らくはよりシンプルで既知のベットシステム）に移行し、残った一つのアクティブな数列を管理するためにその方法を使用します 。
これは、二数列構造の運用が終了する時点を定義し、資金管理に対する階層的なアプローチを採用していることを示しています。なぜ移行するのかについては、いくつかの理由が考えられます。第一に、複雑性の低減です。平均化やストック共有といった機能を備えた二数列管理法は、数列が一つだけ残った場合には過剰、あるいは不要と見なされるのかもしれません。第二に、特定の目的のためです。二数列構造は、特に複数のアクティブな数列（特定のベットパターンや損失回復の必要性から生じる可能性がある）を管理する際の複雑さに対処するために特別に設計されたのかもしれません。状況が単一の数列に単純化されれば、標準的な方法で十分と判断される可能性があります。第三に、グッドマン法の適合性です。「オリジナルグッドマン法」が、最後の数列をクリアする最終段階に適していると考案者によって判断されたのかもしれません。
したがって、「二数列分解管理モンテカルロ法」は、永続的に使用されるシステムとしてではなく、ベット管理の特定のフェーズ（二つの数列がアクティブな状態）のための特定のツールとして意図されているようです。その価値は、このより複雑な状態を管理する能力にあると、考案者は認識していると考えられます。

5. 戦略的決定点：ストック消費 vs「0」生成

5.1 優先順位付けルール

元の記事では、どちらかの実行が可能となるような状況が発生した場合に、蓄積された「ストック」を消費することを優先すべきか、それとも「0」の生成/再配布メカニズムを発動させるべきかについての基準やガイドラインが提供されていると報告されています。
これは、手法の運用における内部的な制御パラメータ、あるいは戦略的な選択肢の存在を浮き彫りにします。ストックの使用と「0」の生成の間に選択/競合が存在するという事実は、それらが類似の目標（例えば、数列の負担軽減）を達成する可能性があること、あるいは「0」の生成/再配布を円滑に進めるためにリソース（ストック）が必要となる可能性があることを示唆しています。優先順位付けが必要となる理由としては、リソース管理（ストックは他の機能、例えば平均化のために必要な有限のリソースかもしれない）、影響制御（「0」生成はストック消費よりも抜本的な介入であり、特定の状況のために留保されるべきかもしれない）、あるいは効率性（特定の条件下では、一方の選択肢が他方よりも数列解消において効率的かもしれない）などが考えられます。
このルール は、戦略的な深み（と複雑性）をさらに一層加えています。これは、ユーザーがシステムの状況に基づいて判断を下す必要があるか、あるいは考案者によって定義された厳格な階層に従う必要があることを示唆しています。この決定点が、数列の進行と利用可能なリソースに直接影響を与えることになります。

6. 要約と結論

6.1 手法の要約

「二数列分解管理モンテカルロ法」は、Chisiki氏によって提案された独自の資金管理手法であり、その際立った特徴は以下の通りです：二つの数列の同時管理、損失時のストック獲得、数列間でのストック共有、定期的な数列値の平均化、特定の条件下での「0」生成とその価値の再配布、そして一方の数列解消後にはオリジナルグッドマン法へ移行するというロジック 。

6.2 考案者の意図（推察）

記述された特徴に基づくと、本手法は、従来のモンテカルロ法と比較して、より制御され、ボラティリティが管理されたアプローチを提供することを目指してChisiki氏によって設計されたと考えられます。特に、損失時のストック獲得や平均化といったユニークなルールは、システムの安定性や、連敗に対する心理的な耐性を重視している可能性を示唆しています。

6.3 免責事項

本稿は、提供された情報源の要約に基づいて、当該手法を説明することに焦点を当てています。その実際の有効性、収益性、またはリスクプロファイルについては、独立した分析、シミュレーション、および確立された他の資金管理手法との比較が必要となりますが、それは本稿の範囲外です。元の記事には、詳細を求めるユーザー向けに内容をダウンロードできる機能があると述べられています 。

引用文献

1. 二数列分解管理モンテカルロ法(メモ)｜Chisiki - note,
https://note.com/chisiki/n/n09e0fdee6d7b