2024 年度版

システムプログラミング実験

確率プログラミング　第1回

担当：中嶋 一貴

1　確率プログラミング実験の目標

これまでのプログラミング基礎演習では，指定された仕様を満たすプログラミングを書く練習がメインだった．つまり，単に「言われたとおりのコードがかける」というだけのスキルと言える．習得したプログラム技術を活かし，より複雑なタスクをこなせるようになるためには，やるべきことを正しく把握し，自分の考えたアプローチが適切であるかを判断するスキルが必要となる．そこで本実験では，求められている内容からどの様なアルゴリズムでプログラムを作るのかを自ら考え，さらに実行した結果からどの様なことがわかるのかを考察する，という技術についての向上を目指す．

2　実験環境とプログラミング言語

本実験を行うにあたって推奨する言語はC++とする．課題のプログラミング方針は C++ を前提として記載されている．しかし，Python などの他言語の使用を禁止するわけではなく，環境構築やコーディングのサポートが得られないことを認識した上であれば，好きな言語を使用してよい．

3　実験の進め方

わからないことやプログラムのエラーを逐一調べながら，求められる仕様に則ったプログラミングを行い，シミュレーション実験を行う．学生間での議論を推奨する．自力で調べ考えた上で解決しないエラーなどに遭遇した場合，教員やティーチング・アシスタントに遠慮なく質問するとよい．下記の C++ のリファレンスは，本実験課題を進める上で参考になるはずである．

- cpprefjp - C++日本語リファレンス
  - `https://cpprefjp.github.io/`
- さくさく理解する C 言語/C++ プログラミング 入門
  - `http://vivi.dyndns.org/tech/cpp/cpp.html`

第 1 回：一様乱数の生成とシミュレーション

確率的現象をシミュレーションする際に乱数を用いることは基本である．例えば，人間社会における感染症の伝搬について考えてみる．感染症の伝搬が 2 つの確率的事象から成るとする：（1）病気に感染している人は，近接する人（例：友人，家族）に確率 p_A で病気を伝搬する.（2）病気に感染している人は，確率 p_B で回復する．コンピュータ上で，一定の人々（例えば 100 人）の模擬集団に対して（1）と（2）の確率的事象を繰り返すことにより，病気の感染伝搬を簡易的にシミュレーションすることができる．当然，上記の（1）と（2）の 2 つの確率的事象だけでモデリングできるほど感染症の伝搬は単純ではない．しかし，確率的現象の数値解析には，優れたモデリングだけでなく，乱数の使用方法を理解した優れたシミュレーション技術も必須である．

乱数をコンピュータ上でどのように生成するか，が問題である．確定的な動作しかしないコンピュータでは，いわゆる純粋な乱数を生成できない．そこで，確定的な計算手順を与えてコンピュータに “乱数のようなもの” を生成させる．そのような乱数を**擬似乱数**と呼ぶ．また，そのような計算手順を**擬似乱数列生成法**と呼ぶ．乱数生成法は注意深く選ぶ必要がある．なぜなら，乱数の生成方法を誤れば，どんなに優れたモデリングであっても，誤ったシミュレーション結果を招き，現象の正確な理解が困難になるからである．

乱数を用いたサンプリングの繰り返しを行うことでシミュレーションや数値計算を行う手法をモンテカルロ法という．様々な物事の事象や現象を数式で表現し，今後を予想することは難しい．その様な場合に，乱数を使ってコンピュータで数値計算することにより答えを推定できる実践的な方法であり，究極の力技とも言える．モンテカルロ法によるシミュレーションは，難しい数学を駆使しなくても予測を行うことが可能であり，プログラミングについては一般的な技術があれば記述可能であるという利点がある．また，前提とする定式化などを必要としないため様々な分野での応用が可能となるため非常に有用なシミュレーション法であると言える．

確率的現象をシミュレーションする技術を体得する 1 つの方法は，多くのシミュレーションを実行することである．エラーと格闘しながらコードを組み，数値結果を出力し，出力された結果を図にプロットし，結果の検証および考察を行うまでが “シミュレーションの実行” である．本実験では，一様乱数の生成とシミュレーションを実行する．

課題 1-1：一様乱数の生成

区間 $[0, 1)$ の一様乱数を独立に n 個生成する関数を作成せよ．関数名は rnd_exp とし，n を引数として受け取るとする．作成した関数を用いて n =1,000 個の乱数を生成し，その平均値と分散を計算せよ．

なお，各言語の乱数ライブラリの関数を利用してよい．例えば C++ では乱数ライブラリ ‘std::random’ 内の関数 ‘std::random_device’, ‘std::mt19937’ および ‘std::uniform_real_distribution’ を利用してよい．

レポートには，(i) 作成した関数のソースファイル名，(ii) 生成した乱数の平均値と分散を記載すること．ファイル名が ‘xyz.cpp’ の場合，‘xyz.cpp を参照のこと．’ と書けばよい．ファイルが複数ある場合，全てのファイル名を言及すること．生成した乱数の平均値と分散については，有効数字 3 桁で報告すること．

課題 1-2: コイン投げシミュレーション

課題 1-1 で作成したプログラムをもとにコイン投げのシミュレーションプログラムを作成せよ．作成したプログラムを用いて 1,000 回のコイン投げのシミュレーションを行い，表・裏それぞれが出た確率を求めよ．ただし, 表が出る確率を p, 裏が出る確率を $1-p$ とすること．

レポートには，$p = 0.2, 0.5, 0.7$ の 3 通りそれぞれについて，関数 rnd_exp を用いて生成した n =1,000 個のコイン投げに対する検証結果と考察を記載せよ．シミュレーションで得られたコインの表・裏の確率は，有効数字 3 桁で報告すること．

次の手順に従って取り組むとよい．

1. rnd_exp を用いて，区間 $[0, 1)$ の一様乱数を独立に $n = 1,000$ 個生成する．
2. 生成した n 個の乱数 $r_1, \ldots, r_n$ を用いて，i 回目のコイン投げ試行の結果を以下のように定める：(i) $r_i \leq p$ であれば，i 回目のコイン投げ試行で表が出たとする．(ii) $r_i > p$ であれば，i 回目のコイン投げ試行で裏が出たとする．

課題 1-3: サイコロ投げ 1

サイコロ投げのシミュレーションプログラムを作成せよ．ただし，各目の出る確率は等確率とする．作成したプログラムを用いて 1,000 回の試行を行い，1,000 回の試行で出たサイコロの目の平均値を求めよ．さらに，その結果を用いて，横軸を試行回数 n，縦軸を n 回の試行の平均値としたグラフを作成せよ．実験結果のグラフには，理論値の線もプロットすること．

レポートには，以下を記載せよ．

1. 1,000 回の試行で出たサイコロの目の平均値．有効数字 3 桁で報告すること．
2. 横軸を試行回数 n，縦軸を n 回の試行の平均値としたグラフ．理論値の線もプロットすること．

課題 1-4: サイコロ投げ 2

課題 1-3 において各目 i $(i = 1, \ldots, 6)$ の出る確率 p_i が次のように偏っていたとする：$p_1 = p_3 = p_5 = 1/9$, $p_2 = p_4 = p_6 = 2/9$. このとき，課題 1-3 の結果がどう変わるかをグラフを作成して示せ．グラフの様式は課題 1-3 と同様とする．理論値についても記載することを忘れないこと．

レポートには，以下を記載せよ．

1. 1,000 回の試行で出たサイコロの目の平均値．有効数字 3 桁で報告すること．
2. 横軸を試行回数 n，縦軸を n 回の試行の平均値としたグラフ．理論値の線もプロットすること．

レポート提出方法

以下を kibaco の課題ページから提出すること．

- 課題の内容を記載したレポート．
  - 課題 1-1, 1-2, 1-3, 1-4 の内容は必須である．各課題について，回答が正しくない場合でも，自力でコードの実装・数値結果の出力・結果の検証と考察がされていれば，部分点が付与される．なお，白紙や無回答の場合，その課題の点数は 0 点となる．
  - ファイル名は “syspro_pp_xxx_yyy.pdf” とする．xxx には学修番号，yyy には氏名を入力する．例えば学修番号が 22012345 で氏名が Kazuki Nakajima の場合，“syspro_pp_21012345_kazuki_nakajima.pdf” とする．漢字やひらがなをファイル名に含めないこと．
- 各課題を解くために使用したソースコード．
  - ソースコードが複数に分かれている場合は，zip ファイルにまとめて提出すること．
  - 各ソースコードが，誰のもので，どの課題に対応しているかわかるようにファイル名を設定すること．例えば，氏名が Kazuki Nakajima で課題 1-1 に対する C++ コードのファイル名は “kadai_1_1_kazuki_nakajima.cpp” とする．

注意事項

- **本課題の提出締め切りは 1 週間後（2024/10/23）の 12:00 である．**
- レポート作成時には，テクニカルライティングの資料を確認し，全体をよく検証してから提出すること．
- 提出期限を過ぎると，kibaco から課題を提出できない．提出期限後に課題を提出する場合，下記に連絡をすること．連絡先：nakajima [at] tmu.ac.jp（[at] を@に変える）
- 提出ファイルを間違えていないか，提出前に慎重に確認すること．間違ったファイルが提出されても，教員から学生に逐一連絡することはしない．
- 本授業の成績が確定するまで，tmu メールをよく確認すること．成績に関わる重大な事項で教員から連絡する場合がある．