第 1 章

LaTeX の乱数生成アルゴリズムを改善する

1.1 乱数生成コマンド

LaTeX のパッケージに固定小数演算ができる FP パッケージがあります.

1.1.1 固定小数演算

TeX,LaTeX は共に整数の演算は可能ですが, 小数点を含む計算は行うことができません (寸法を除く). 固定小数点の演算をすることを目的としてあるのが FP パッケージです.

1.1.2 乱数を出力する FPrandom コマンド

fp パッケージの中には上記以外にも 0 ～ 1 の範囲の疑似乱数を出力する FPrandom というコマンドがあります. FPseed でシード値を決めてから FPrandom を使って変数に乱数を格納します.

```
\FPseed = 156
\FPrandom{\result}
\FPprint{\result}
```

1.2 FPrandom の乱数生成アルゴリズムを調べる

1.2.1 目的

今回は FPrandom に使われている疑似乱数がいわゆる多くの問題点がある昔のアルゴリズムかどうかを調べる為に行います．

1.2.2 疑似乱数アルゴリズムの問題点

現在，世に出回っている疑似乱数アルゴリズムは様々あります．これは今まで研究されてきたアルゴリズムに何かしらの問題があるからです．偏りが出たり，パターンがあったり様々です．また，疑似乱数には周期があります．一定の数の乱数を出力したらまた，最初から同じ疑似乱数列を出力し始めます．現在は計算機のスペックも高くなりより複雑なシミュレーションを行えるようになりました．その為，用いられる疑似乱数の数が従来のアルゴリズムでは周期を上回る危険性もありました．周期を伸ばすことも新しい疑似乱数アルゴリズムを開発する理由の一つです．

1.2.3 ソースを読む

それでは fp パッケージのソースを読んでいきます．しかし，実際には fp パッケージ本体が内部的に呼び出している fp-random パッケージのソースを読んでいきます．

コメントに正解が書いてあった

22 行目から FPrandom の定義が始まります．その直後，コメントが長く続いています．コメントには次のようなことが書かれていました．Algorithm reproduce from a very old Fortran program (unknown origin!) どうやら大昔の Fortran の疑似乱数アルゴリズムを TEX に起こしたものが FPrandom の正体みたいですね．これはいろいろ問題点がありそうですね．更にその下のコメントには Fortran で書かれた疑似乱数アルゴリズムらしきものがあります．これを読んでいけばどんなアルゴリズムが使われているか分かりますがそれではつまらないのでコメントを抜けた後の TEX で書かれた疑似乱数アルゴリズムの方を見ていきましょう．

TeX のマクロで実装された疑似乱数アルゴリズム

```
\ifnum\FPseed=0%
     \FPseed=123456789%
     \FP@debug{random: seed value undefined! We will used
     ↪  \the\FPseed.%
        Define it if you want to generate a different sequence of
        ↪  random numbers.}%
\else%
     \FP@debug{random: seed value used: \the\FPseed}%
\fi%
```

これはシードを指定してるかどうかを判定してしてない場合 123456789 をシードとするというマクロですね．次見ていきます．

```
\FP@xia=\FPseed%
\divide\FP@xia by 127773%
\FP@xib=\FP@xia%
\multiply\FP@xib by 127773%
\advance\FP@xib by -\FPseed%
\FP@xib=-\FP@xib%
\multiply\FP@xia by 2836%
\FPseed=\FP@xib%
\multiply\FPseed by 16807%
\advance\FPseed by -\FP@xia%
```

このアルゴリズムのコアとなる乱数の計算ですね．TeX で書いてる為ごちゃごちゃしていますが数式で表すと以下のようになります．

$$s_{n+1} = \frac{S_n}{127773} \times 2836 + 16807 S_n \bmod 127773$$

s_n が n 番目に出力した乱数で s_{n+1} は $n+1$ 番目に出力する乱数です．つまり漸化式になっています．線形合同法の漸化式と似ていますね．線形合同法は C 言語の rand 関数に採用されているアルゴリズムですが多くの問題点が発見されていて現在は非推奨となっ

ています．線形合同法では $16807S_n$ の部分が定数ですが，それが乱数を含む形になっています．線形合同法の式は以下のとおりです．

$$S_{n+1} = A \times S_n + B \bmod M \quad \text{(A,B,M は定数)}$$

... 構造が非常に似ていますね．

```
1 \ifnum\FPseed>0%
2  \else%
3      \advance\FPseed by 2147483647%
4  \fi%
5  \FPdiv\FP@tmpa{\the\FPseed}{2147483647}%
6  \global\let\FP@tmp\FP@tmpa%
7  \global\FPseed=\FPseed%
```

乱数の正規化を行っています．214793647 がこの乱数列における最大値 M でこの時点で $n+1$ 番目の乱数が負の値の場合は M を乱数に足します．その後 FPdiv と呼ばれる小数の割り算ができるマクロによって乱数を M で割り，乱数を $0 \sim 1$ の範囲の値に正規化しています．

タイトル

著者：俺
挿絵：俺
発行：2018 年 1 月 6 日
印刷：俺