統計解析システムRの多倍長精度パッケージ

中野 純司　　モデリング研究系　教授

はじめに

Rはフリーの統計解析ソフトウェアであり、統計学研究者にとっては基礎教養の一つとなっているばかりではなく、多くの分野の統計利用者に広く利用されている。

Rは計算機上で浮動小数点表現のdouble型を主に用いている。通常の演算ではこれらの演算精度(53bitで15〜16桁)でも十分であるが、データやアルゴリズムによっては著しく精度が落ちることもある。もちろん、精度を落とさないための工夫は考えられている。例えばC99で追加されたhypot(平方和の平方根)を求める数値関数では通常 $\sqrt{x^2+y^2}$ で表されるこの関数は、式変形を行い、例えば $|x|\sqrt{1+(y/x)^2}$ として演算を行い、doubleの範囲内で無難な結果を返している。このような演算の工夫(誤差の回避)は一つの研究テーマであり、一般に簡単なものではない。浮動小数点演算の性質上、桁落ち（近い数字の差）や積残し（大きさの極端に異なる数字の和）のような誤差を生みやすい計算においても、多倍長演算を利用すればかなりの問題を回避できる。

既に多倍長精度のRパッケージはいくつか存在するが、満足のいく物はないとおもわれる。われわれは複素数および線形計算を視野に入れ、R上で多倍長精度を平易に扱うためのパッケージを開発している。ある程度の機能としてまとまらないとCRANには登録出来ない為、現在名称は仮にRmpenvとしている。なお2016/5/19現在CRANには8424のパッケージが登録されている。

Rと組み合わせ可能な多倍長精度計算の実装

Rは主としてCおよびFortranの倍精度演算で書かれており、精度の異なる演算をサポートするには、R本体を書き換える、もしくはRAW型等を用いて各種ライブラリ等とハンドリングを考える必要がある。われわれは幾つかの高速な多倍長精度ライブラリや既存のデータ型について調査を行った。

- long double
  128bitのデータサイズをとるが、x86系ではFPU(80bit)のサイズでしか演算を行わない。このため、異機種間(x86以外では概ね128bitで演算)では演算結果が異なる。
- __float128(gcc)
  gccに組み込まれたIEEE準拠の4倍精度ライブラリである。 最適化を施してある為に、アライメントが16byteでなければ動作せず、RからRAW型でハンドルする場合はR本体のアライメント(現状8byte)を変更する必要がある。Windows上でのスタックアライメントで動作するのか検証はしていない。なお、内部はmpfrを最適化したlibquadmathとなっている。
- QD (C++/C)
  基本的演算がdoubleに制限されており、悪条件下では補正する必要がある。また複素数がサポートされていない。
- ARPREC (C++), MPFUN (Fortran90)
  元々Fortran上からでも使えるようになっており、MPI等との親和性も他のライブラリから見れば高いように思われる。ただ、RはC言語ベースなので、採用を見送った。
- mpfr, mpc (C)
  最も利用者数が多く、比較的枯れていると言っても良いので、現在これらをベースにRmpenvの開発を行っている。

Rにおける既存の多倍長精度パッケージ

- CRAN:gmp
  有理数型や多倍長整数が扱える。
- CRAN:Rmpfr
  Rのコアメンバーである Martin Maechler によるS4クラスを用いたパッケージであり、仮数部のビット列はRのRAW型として保持し、精度、符号、指数については属性として仮数のRAW型に付与している。多倍長精度ライブラリの属性値すべてをRにバインドしてあるので、既存のparallelパッケージ等を用いてそのまま並列化が可能である。S4のオーバーヘッドによる遅延が大きく、また複素数が扱えない。
- R-forge:mpc
  S3で実装されておりmpcのデータを全て外部ポインターとして扱いRとのハンドリングを行っている。速度的にはほぼネイティブで動作するライブラリへのラッパー的なパッケージである。ただし、Rでは外部ポインタをシリアライズ(Rからはデーター構造がわからない)することは出来ないので、既存のparallel等のパッケージを用いて並列化を行う事は不可能である。演算結果の保存(save.image等)も不可能であり、処理時間がかかる多倍長精度のライブラリとしては扱いにくい。

実行例

当初は様々なライブラリの利用を実験的に試みた。現状でもまだ実験的ではあるが、以下のように動作している。

```
> library(Rmpenv)
> default_prec(512)
[1] 512
> options(Rmp.decprec=floor(log10((2^512)))) # 10進での有効桁数
> Const(pi)             # 引数型がシンボルだった場合、文字列として扱う
 [1] 3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706798214808651328230664709384460955058223172535940812848 11e+00
```

```
> x<-1:100000             # 普通の倍精度計算
> object.size(x)
400040 bytes
> system.time(sqrt(x))
   ユーザ    システム        経過
     0.000       0.001       0.001
> library(Rmpenv)          # 我々の(S3)での性能
> default_prec(256)
[1] 256
> x<-mpreal(1:100000)
> object.size(x)
56000248 bytes
> system.time(sqrt(x))
   ユーザ    システム        経過
     0.076       0.000       0.075
> library(Rmpfr)          # Rmpfr(S4)での性能
> x<-mpfr(1:100000,prec=256)
> object.size(x)
114400456 bytes
> system.time(sqrt(x))
   ユーザ    システム        経過
     0.664       0.012       0.677
```

おわりに

このパッケージ（仮称 Rmpenv）は現在も開発中なため仕様や概要は今後大きく変わる可能性があるが、線形計算、複素数計算などが容易に利用できる多倍長演算パッケージを目指している。本研究は中間栄治氏（株式会社COM-ONE）との共同研究である。