Androidに対応した高品質数値計算が可能な数値計算言語
Numerical Computation Language for High-quality Computation in Android

九州工業大学　○川端　悠一郎，古賀 雅伸，元山 忠，野口 剛史，松竹 弘海
福岡工業大学短期大学部　矢野 健太郎
Y. Kawabata, M. Koga, A. Motoyama, G. Noguchi, and H. Matsutake
Kyushu Institute of Technology
K. Yano
Fukuoka Institute of Technology Junior College

Abstract In this research, we developed a numerical computation language which suppports verified numerical computation. It is able to write numerical computation programs like mathematical representation by using the language. And we ported the language to Android in order to allow us to perform high-quality computation on portable information devices.

1 はじめに

近年，タブレットなどの携帯情報端末が普及しており，携帯情報端末で業務をこなすことができる実用的なアプリケーションの開発が盛んに行われている．制御系設計や数値計算プログラムにおいても携帯情報端末で開発が行えるような環境が徐々に整ってきている [1]．しかし携帯情報端末で数値計算言語を扱う際はPCで扱うのとは異なりキーボードやマウスがないので計算の入力効率が下がってしまう．

本研究では，数学的表現に近い形でプログラムを記述できる数値計算言語 MATX[2] を精度保証付き数値計算や多倍長演算に対応させ，ユーザーの記述したプログラムコードを変更することなく高品質な数値計算を可能にした. そして，携帯情報端末で信頼性や品質の高いシミュレーション計算や制御系設計が行えるよう高品質数値計算が扱える数値計算言語 MATX を Android に対応させた. その際に計算の入力効率を改善する UI を考案・実装し，制御系設計や計算シミュレーション効率的に行えるようにした．

2 数値計算言語

2.1 JMaTX

JMATX[3] は，数値計算言語 MATX の処理系であり，Java 言語で実装されている．本研究では，高品質数値計算を実現するため JMATX と基盤数値計算パッケージ NFC[4]，精度保証付き数値計算パッケージ CGA[5]，多倍長演算パッケージ MPFloat[6] を連携させた．その際，精度保証付き数値計算で使用する区間や，多倍長精度浮動小数点数という新しいデータの種類をプラグイン方式で追加できるようにした．こうすることで，JMATX 単体では倍精度が利用可能であり，区間や多倍長を組み合わせた高品質な数値計算も可能となった．JMATX のアーキテクチャを図 1 に示す．倍精度を表す DoubleConstant クラス，区間を表す IntervalConstant クラス，多倍長精度を表す MPFloatConstant クラスは．実数を表す RealConstant インターフェースを実装しており，全て統一的に扱うことができる．

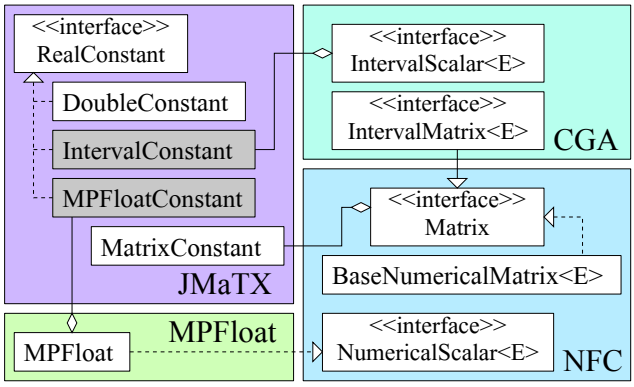

図 1: JMATX のアーキテクチャ

2.2 多倍長演算

2.2.1 多倍長浮動小数点数

倍精度以上の精度を持つ浮動小数点数を実現する方法として，浮動小数点数フォーマットの指数部と仮数部のビット数を増やした構造を利用する方法がある．IEEE 754 規格における，単精度，倍精度と多倍長精度の浮動小数点型のフォーマットを図 2 に示す.

1) 単精度浮動小数点型(10進約7桁) float型

1bit	8bit	23bit

2) 倍精度浮動小数点型(10進約16桁) double型

1bit	11bit	52bit

3) 多倍長精度浮動小数点型(任意桁)

1bit	α bit	β bit

図 2: 浮動小数点数のフォーマット

2.2.2 多倍長演算演算ライブラリ

JND

JND(JND is Not a Double Precision Arithmetic Library) [7] は，2011 年から開発されている Java 言語で記述された任意倍長演算ライブラリである．Java の double を複数使用し，Knuth[8] や Dekker[9] のアルゴリズムを利用し任意倍長の精度を実現する．Android 端末で多倍長を扱う際はこの Java 言語で記述されたライブラリを使用する．

MPFloat

MPFloat[6] は，Exflib[10] や GMP(MPFR)[11] や JND などの多倍長・任意倍長の高精度環境を切り替えて利用できる多倍長演算パッケージである． Exflib では，同時に複数の精度桁での計算を行うことができないので，桁数の異なる複数の Exflib のライブラリを準備し，MPFloat から適当な桁数のライブラリを利用し，桁数変換を行うことで，精度桁の動的な変更を実現した．MPFloat のクラス図を図 3 に示す．

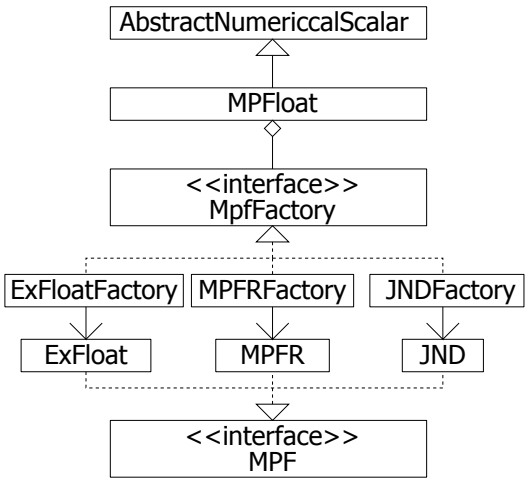

図 3: MPFloat のクラス図

2.3 精度保証付き数値計算

精度保証付き数値計算パッケージ CGA[5] では，区間演算，区間行列演算，関数の微分，線形方程式，非線形方程式，固有値問題，多項式の評価の精度保証付き数値計算を行うことができる．

CGA は NFC の汎用的な数値データ型に対応しているので，MPFloat と組み合わせて用いることが出来る．図 4 に CGA のアーキテクチャ(パッケージ図) を示す．

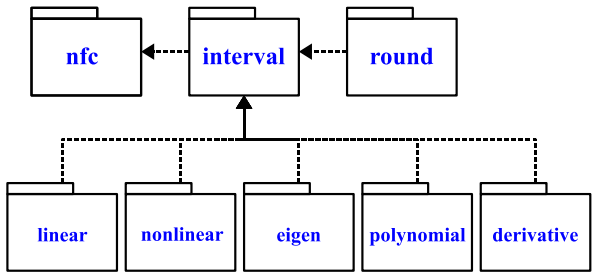

図 4: CGA のパッケージ構造

2.4 Android への対応

本研究では数値計算言語 MATX[2] を Java 上で動作するようにした処理系 JMATX[3] を携帯情報端末からでも扱えるように Android に対応させた．Android 上で数値計算言語を扱えるよう Android 用の UI を作成し，既存の JMATX プロジェクトで計算を行なっている core に依存して Android で処理を行うことを可能とした．また，

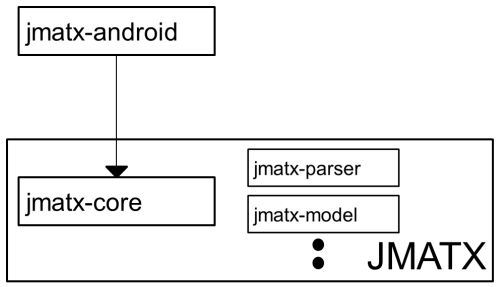

図 5: 依存関係

Android での開発は Java 言語で行う必要があるので高品質な数値計算を行うためのライブラリとして JND を使用する．

3 入力効率の高い UI の考案と実装

Android や Windows8 搭載の携帯情報端末で文字を入力する際にキーボードはソフトウェアキーボードを用いることが多い．しかし，ソフトウェアキーボードは PC で使用するような物理キーボードと違い操作性が悪く入力効率が落ちてしまうことがある．そこで，入力を支援する UI を考案し実装した．

3.1 入力補完機能

MATX には多くの内部関数が用意されている．しかし，ソフトウェアキーボードでは長い名前の関数や，一度使

用した名前の長い変数などをタイプするのに PC で扱う物理キーボードでの入力に比べて効率が遅くなってしまう．その対策として入力途中の関数名や変数名を補完できる入力補完機能を実装した．図 6 に使用例を示す．

図 6: 入力補完使用例

3.2 ジェスチャー操作機能

通常コンソールや CLI(Command Line Interface) を PC で扱う場合は上の矢印キーを押すことで入力履歴が閲覧できる場合が多い．しかし，ソフトウェアキーボードには上下の矢印キーがない場合が多い．そこで，Android 上でジェスチャーを用いて入力履歴の閲覧等を行える UI を実装した．図 7 に使用例を示す．

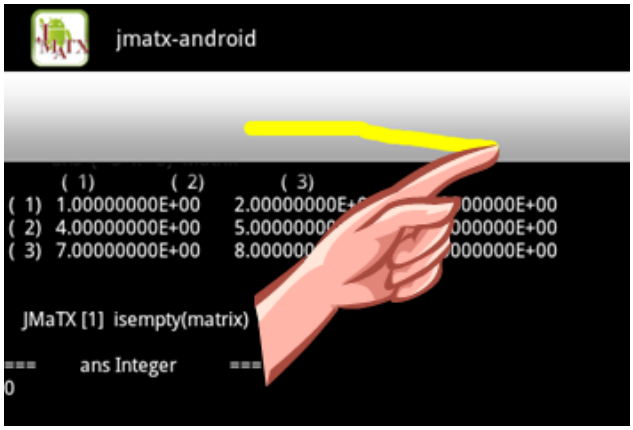

図 7: ジェスチャー操作使用例

3.3 行列入力ダイアログ

行列をソフトウェアキーボードで入力しようと思うと成分の数が増すにつれタイプミスする可能性が高くなる．その対策として，行列用の入力ダイアログを実装した．図 8 に使用例を示す．

4 例題

4.1 計算時間比較

2 個の正方行列の積の計算時間を PC と Android で比較した結果を図 9 に示す．実行環境を表 1 と表 2 に示す．Android 端末のほうが PC と比べて約 10 倍遅い結果となった．

図 8: 行列入力ダイアログ使用例

表 1: PC　実行環境

	環境
CPU	Intel(R) Core(TM) i5-2500K CPU @ 3.30GHz
メモリ	8.0GB
OS	Windows 7 Professional　64bit
Java VM	Java(TM) SE Runtime Environment (build 1.7.0_03-b05)

表 2: 携帯情報端末　実行環境

	環境
CPU	Tegra 2　1GHz
メモリ	1024 MB
OS	Android 3.2.1

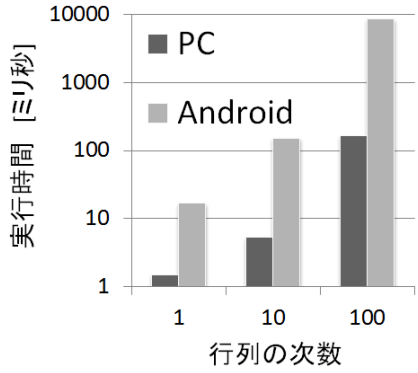

図 9: 行列の積の速度比較

4.2 精度保証付き多倍長演算

リカッチ方程式

$$A^TP + PA - PBR^{-1}B^TP + Q = 0$$

の真の解を含む区間 P と LQ 最適問題に対応する状態フィードバック行列の真の解を含む区間 F を求める関数を図 10 に示す.

```
Func List lqrVerifier(Matrix pA, Matrix pB,
                      Matrix pQ, Matrix pR) {
  Matrix A, B, Q, R, D, V, F, Dr;
  A = intval(pA);
  ...(略)
  {D,V} = eig([A, B/R*B#][Q, -A#]]);
  D = diag2vec(D);
  ...(略)
  {Dr,idx} = sort(Re(D));
  P = -Re(V(n+1:2*n,idx(1:n))/V(1:n,idx(1:n)));
  F = R \ (S# + B#*P);%

  return {F,P};
}
```

図 10: MATX 言語で記述した例

既存のコードを精度保証に対応させるには，A = intval(pA); のように変数を区間に変換する関数を追加し，区間演算特有の処理を追加すればよい．四則演算等は従来通り記述可能なため記述性が高い．また，既存のコードを多倍長演算に対応させるには，図 11 のように，実数型を多倍長精度に切り替え，精度桁数を（150 桁等に）変更，多倍長用の丸めモードを設定，というコードをプログラムの先頭に追加すればよい．上記の精度保証付き数値計算への対応と多倍長演算への対応は組み合わせることが可能であり，高品質な数値計算を高い記述性で実現できる.

```
setRealType("MPFloat");
setPrecision(150);
setMPFloatRoundModeSelector();
```

図 11: 多倍長演算に対応させるための記述

4.3 Rump の例題

本節では文献 [12] で提案されている Rump の問題の解を導出し，携帯情報端末でも多倍長演算を PC と同様に扱えることを示す．実行環境を表 1 と表 2 に示す．Rump の例題は

$$a = 77617, b = 33096$$
$$f = b^6 \times 333.75 + (a^2 \times (a^2 \times b^2 \times 11 - b^6(b^4 \times 121) - 2)) + (b^8 \times 5.5) + (a \div (b \times 2))$$

の値を求める問題である．解は

$$f = -0.827396059946821368141165095\ldots \quad (1)$$

であるが，この問題を倍精度で解くと PC でも携帯情報端末でも結果は

```
===          ans Double             ===
7.2769772261375E11
```

のように出力され，大きな誤差が発生する．次に，この問題を多倍長演算パッケージを用いて多倍長精度の保証付きで解く．ただし，多倍長精度の精度桁は 10 数で 100 桁である.

この問題を多倍長精度で解くと PC でも携帯情報端末でも結果は

```
===          ans MPFloat            ===
-0.827396059946821368141165095479816
```

のように出力され，この値は (1) 式と比べて非常に高精度な値になる.

4.4 入力時間比較

実装した UI を使用する場合と UI を使用しない場合で以下のコードをユーザーに入力してもらい，入力に要した時間の平均を比較した.

```
matrix=[[1 2 3][4 5 6][7 8 9]]
isempty(matrix)
```

ユーザー 4 人による平均入力時間を表 3 に示す．UI を使用することにより入力速度が向上することを確認できた.

表 3: 入力時間

UI 使用	41.7s
UI 不使用	69.9s

5 おわりに

本研究では数値計算言語の処理系 JMATX を携帯情報端末からでも扱えるように Android に対応させた．JMATX を携帯情報端末から使用できるようにすること

で，どこでも数値計算言語を扱うことの出来る環境を整えることができた．また，精度保証付き数値計算や多倍長演算に対応させ，ユーザーの記述したプログラムコードを変更することなく高品質な数値計算を可能にした．今後は，入力効率の向上や使用出来る関数や機能を増加を行い現場で制御系設計を行える可能性を示していきたい．

参考文献

[1] 石倉雄飛, 古賀雅伸. Android 端末に対応した制御系 CAD システムの開発. 第 56 回システム制御情報学会, pp. 359–360, 2012.

[2] 古賀雅伸. 制御・数値解析のための MATX. 東京電気大学出版局, 2000.

[3] 杉永良太. 精度保証付き数値計算に対応した数値計算言語の開発と制御系設計への応用. 九州工業大学修士論文, 2011.

[4] 古賀雅伸, 松木毅. OS 中立な数値計算基盤ライブラリの開発と制御系設計への適用. 第 3 回制御部門大会, pp. 725–728, 2003.

[5] 矢野健太郎, 古賀雅伸. 精度保証付き数値計算に基づく制御系解析. 第 50 回自動制御連合講演会, pp. 894–895, 2007.

[6] 山村英介, 古賀雅伸, 矢野健太郎, 山田賢治. 多倍長計算を用いた制御系設計パッケージ. 第 52 回システム制御情報学会研究発表講演会, pp. 139–140, 2008.

[7] 野口剛史. *JND*, 2012. `http://ci.mk.ces.kyutech.ac.jp/job/JND-nightly/`.

[8] D.E.Knuth. The Art of Computer Programming（日本語版）. アスキー, 2004.

[9] T.J.Dekker. *A Floating-point Technique for Extending the Available Precision.* Numer. Math, 1971.

[10] 藤原宏志. Multiple-Precision Arithmetic Library exflib, 2006. `http://www-an.acs.i.kyoto-u.ac.jp/~fujiwara/exflib/`.

[11] GNU. *The GNU MP Bignum Library.* `http://gmplib.org/`.

[12] P. Benner, A. Laub, and V. Mehrmann. A Collection of Benchmark Examples for the Numerical Solution of Algebraic Riccati Equations I: Continuous-Time Case. *Tech. Report SPC 95 22, Fak. f. Mathematik, TU Chemnitz-Zwichau, 09107 Chemnitz, FRG.*, 1995.