

MATLAB ODE Suite について

Ryuichi ASHINO*, Michihiro NAGASE[†],
Rémi VAILLANCOURT[‡]

概要

本論文は、MATLAB ODE ソルバーで使われている数値計算法の背後には広大な理論があることを MATLAB ユーザーに知らせるために書いた解説的論文 R. Ashino, M. Nagase and R. Vaillancourt, *Behind and beyond the Matlab ODE suite*, Computers Math. Applic. **40** (2000) 491 – 512 をもとに、MATLAB の簡単な解説を含め、日本語でまとめたものである。

Keywords : stiff and nonstiff differential equations, implicit and explicit ODE solvers, MATLAB odedemo

1 Matlab について

MATLAB は Fortran を発展させた高度な数値計算、ヴィジュアライゼーション（視覚化）、そしてプログラミングが扱える科学技術計算のための言語である。MATLAB について簡単にまとめてみよう。詳しくは [1] あるいは次のウェブページを参照していただきたい。

▷ 芦野のホームページ

<http://www.osaka-kyoiku.ac.jp/~ashino/>

▷ *A survey of the MATLAB ODE suite*, CRM Technical Report 2651

<http://www.osaka-kyoiku.ac.jp/~ashino/pdf/2651.pdf>

Matlab の歴史

MATLAB の名前は *matrix laboratory* に由来する。MATLAB は数値解析学者 Moler により、Fortran のサブルーチン集である連立 1 次方程式を解くための LINPACK と固有値を計算するための EISPACK（現在ではまとめられて LAPACK になっている）をもとに、これらのアルゴリズムが対話的に使えるように作られた。当初の MATLAB は Fortran 言語で書かれていたが、現在の MATLAB は対話型の操作環境とヴィジュアライゼーション機能を統合して C 言語で書かれている。

Matlab と *Mathematica* や Maple との比較

MATLAB は *Mathematica* や Maple と並び称せられる新しい世代の高級言語であるが、数値計算に重きを置いている点で、数式処理をサポートしている *Mathematica* や Maple とは一線を画している。具体的には、MATLAB は IEEE 規格に合わせたコンピュータのハードウェアの精度を使うため、数値計算をすべて倍精度の浮動小数で行い、計算が速い。これに反し、*Mathematica* と Maple は可変精度で数値計算をするため、必要な精度を確保できるが、この精度はソフトウェアによるため計算は遅い。MATLAB と Maple はオープンアーキテクチャーであるので、ほとんどの関数をソースコードとして参照することができるが、*Mathematica* はどちらかといえばクローズドアーキテクチャーである。このため、*Mathematica* は一部のエキスパートからは敬遠されることもある。MATLAB は Symbolic Math Toolbox という拡張パッケージにより Maple のカーネルにアクセスすることができて、Maple の持つ数式処理機能までも使うことができる。

*Division of Mathematical Sciences, Osaka Kyoiku University, Kashiwara, Osaka 582-8582, Japan, E-mail: ashino@cc.osaka-kyoiku.ac.jp

[†]Department of Mathematics, Graduate School of Science, Osaka University, Toyonaka, Osaka 560-0043, Japan, E-mail: nagase@math.wani.osaka-u.ac.jp

[‡]Department of Mathematics and Statistics, University of Ottawa, Ottawa, Ontario, Canada K1N 6N5, E-mail: remi@uottawa.ca

北米での Matlab の使われ方

MATLAB は北米では応用数学の教育にも積極的に取り入れられている。たとえば、オタワ大学の場合、数学および統計学教室のコンピュータ室には 50 台程度の Windows 機があって、MATLAB と Maple がインストールされており、応用線型代数や微分方程式の講義で MATLAB が使われていた。また、工学部では MATLAB などの多くの言語がインストールされたコンピュータを持つコンピュータ室があり、授業によっては、学生の好む言語を使ってプログラミングできる環境にあった。

就職の条件などでは、Fortran, C と並んで MATLAB があり、このうちのどれかが使えることが必要であったりして、MATLAB が基本的な科学技術計算言語として認められていることがわかる。

Matlab と教育

教育に積極的に取り入れられている例を 2 つ挙げよう。

- 北米における行列計算の標準的教科書 [5] ではアルゴリズムの記述に MATLAB 言語の記法を使っている。
- 最近の微分方程式の数値解析の教科書 [6] では計算に MATLAB を使っている。

2 Matlab ODE suite

MATLAB ODE suite は硬い系 (stiff ODEs) に適したアルゴリズムを導入している。硬い系について簡単に説明した後、MATLAB ODE ソルバーとそのオプションについて述べる。硬い系については [2] あるいは [7], p. 1 や [8], p. 217–221 を見よ。また、MATLAB ODE suite を使って常微分方程式を学ぶ教科書として [3] がある。

Nonstiff and Stiff ODEs

微分方程式系は、nonstiff (硬くない) と stiff (硬い) に分類できる。硬くない系の解法は比較的良好に知られている。硬い系は応用上重要であり、難しい問題である。微分方程式の数値解法に要請される条件をまとめておこう。

- どの方法も適合 (consistent) しなければならない、つまり、次数が 1 以上でなければならない。
- 次数はできるだけ高いほうがよい。
- 絶対安定領域 R (region of absolute stability) はできるだけ広いほうがよい。

微分方程式系

$$y' = f(t, y)$$

を考える。Jacobi 行列 $f_y(t, y)$ の固有値を λ_j とする。微分方程式の数値解法が数値的に安定であるためには、ステップ幅 $h > 0$ に対して、複素数 $h\lambda_j$ は複素平面上の絶対安定領域 R に含まなければならない。微分方程式系が硬い場合は、原点から遠く離れている負の実部を持つ固有値と原点の近くにある負の実部を持つ固有値があるから、 R が負の実軸を含まないならば、 $h > 0$ を非常に小さくとらなければならない。したがって、精度のためにではなく安定性のためにステップ幅 $h > 0$ を非常に小さくとらなければならないことになる。硬い微分方程式系に対して、陰的解法は負の実軸を含む絶対安定領域を持つから、ある程度大きい $h > 0$ がとれて安定になるが、精度を確保するためには $h > 0$ はそう大きくはとれない。

Nonstiff ソルバー (陽的方法)

Nonstiff ソルバーには次の 3 つがある .

- ode23: 2 次と 3 次の陽的 Runge–Kutta.
- ode45: Dormand–Prince による 4 次と 5 次の陽的 Runge–Kutta.
- ode113: Adams–Bashforth–Moulton による 1 次から 13 次の予測子・修正子法 .

ここでは , 最もよく使われる ode45 ソルバーについて詳しく述べよう . このコードは Mark W. Reichelt と Lawrence F. Shampine によって書かれている . 使われているアルゴリズムは Dormand, Prince による陽的 Runge-Kutta (4,5) 法であり , これは RK5(4)7FM, DOPRI5, DP(4,5), DP54 などとも呼ばれる [9] . また , Dormand, Prince から私的に得た 4 次の “free” interpolant を使っており , 局所的に補外している . 詳しくは [10] をみよ .

Stiff ソルバー (陰的方法)

Nonstiff ソルバーには次の 4 つがある .

- ode23s: 2 次と 3 次の陰的 Runge–Kutta.
- ode15s: 1 次と 5 次の陰的微分公式 .
- ode23t: 台形則によって中程度の硬い微分方程式や微分・代数方程式を解く .
- ode23tb: 硬い微分方程式を解く低次の方法 .

ode15s では Mass オプションを on にすると ,

$$M(t)y' = f(t, y)$$

の形の微分方程式も解ける .

ここでは ode23tb ソルバーについて詳しく述べよう . このコードは Yanyuan Ma, Mark W. Reichelt, Lawrence F. Shampine によって書かれている . 使われているアルゴリズムは TR-BDF2 法である . これは陰的 Runge-Kutta 法で , 最初の段階で陰的 Runge–Kutta を使い , 第 2 段階で 2 次の後退微分公式を使っている . 構造上 , どちらの段階でも同じ繰り返し行列を使うが , この方法は Bank, Rose の提案による . また , M. Hosea と L.F. Shampine による改良型誤差評価とさらに効果的な計算を行っており , “free” interpolant を使っている .

Matlab ODE Suite の使用法

MATLAB ODE Suite は次のように実験的に使うのがよい .

- (i) まず , ode45 を使ってみる . うまく働けばそれでよし . 問題は硬くない系であった .
- (ii) 次に ode23 や ode113 を使ってみる .
- (iii) うまいかなければ , 問題は硬い系かもしれないので , ode15s を使ってみる . うまく働けばそれでよし . さもなくば , ode23s を使う .
- (iv) 最後に ode23t や ode23tb を試す .

MATLAB ODE Suite は下のように多くのオプションを持っており、コマンド `odeset` でオプションを変更できる。デフォルトを中括弧で表す。

```
odeset
  AbsTol: [ positive scalar or vector {1e-6} ]
  BDF: [ on | {off} ]
  Events: [ on | {off} ]
InitialStep: [ positive scalar ]
  Jacobian: [ on | {off} ]
JConstant: [ on | {off} ]
JPattern: [ on | {off} ]
  Mass: [ on | {off} ]
MassConstant: [ on | {off} ]
  MaxOrder: [ 1 | 2 | 3 | 4 | {5} ]
  MaxStep: [ positive scalar ]
NormControl: [ on | {off} ]
OutputFcn: [ string ]
OutputSel: [ vector of integers ]
  Refine: [ positive integer ]
  RelTol: [ positive scalar {1e-3} ]
  Stats: [ on | {off} ]
```

3 M-ファイル

ここでは MATLAB を使う上で必要となる M-ファイルについて簡単に説明する。MATLAB の実行文を含んだ拡張子 `.m` を持つ単なるテキストディスクファイルを M-ファイルと呼ぶ。M-ファイルはテキストエディタで作成し編集することができる。M-ファイルは関数 M-ファイル (function M-file) とスクリプト M-ファイル (script M-file) の 2 種類に分類される。スクリプト M-ファイルとは MATLAB の実行文を含んだ M-ファイルであり、単にスクリプトあるいは M-ファイルと呼ばれる。関数 M-ファイルは別の種類の M-ファイルであり、単に関数と呼ばれる。この 2 種類の M-ファイルの重要な違いを述べると

- スクリプトは呼び出されるたびにディスクから読み込まれて各行ごとに実行される。関数は実行にあたって RAM に読み込まれる。
- 関数は入力と出力のパラメータを持つ。スクリプトはスクリプト自身に書き込まれた変数に対してのみ作用する。

関数は書式の異なるデータに対して作用する一般的な作業に柔軟に対応でき、適している。スクリプトは同じ作業を繰り返し行う場合に適している。

- スクリプトはキーボードマクロと考えられる。関数の変数は関数の中だけで有効な局所変数である。

スクリプト名をタイプするだけで、スクリプトに含まれるすべての実行文がコマンドウィンドウにタイプされたときと同じように順番に実行されていく。スクリプトに含まれる変数名がすでにワークスペースに記憶されている場合にはスクリプトはこの記憶されている変数に作用する。したがって、この変数の値はスクリプトによって変わり得る。この性質を利用することができる反面、思わぬ結果を招くことがある。関数の変数が関数の中だけで有効であるという性質は非常に安全であり柔軟性を持つ。関数との情報のやり取り

はパラメータリストにある変数を通してのみ可能である。唯一の例外はグローバル変数 (global variable) を使ったときであるが、これにはグローバル変数の使用を宣言する必要がある。

関数 M-ファイル

MATLAB の関数は C の関数や Fortran のサブルーチンと同様である。それぞれの関数 M-ファイルには 1 つの関数が定義される。関数 M-ファイルでは別の関数を呼び出して使うこともできるし、自分自身を呼び出すことにより再帰的に関数を定義することもできる。

MATLAB の関数は入力のためと出力のための 2 つのパラメータリスト (いくつかのパラメータからなる行ベクトル) を持つ。グローバル変数を global を使って宣言しない限り、MATLAB の関数の変数はその関数自身にだけ通用する局所的な変数である。つまり MATLAB は入力パラメータの値を変えることは決してなく、出力パラメータに値を返すのみである。

関数 M-ファイルの構文

関数に与える名前を function_name とすると、ファイル名 function_name.m でディスクに保存しなければならない。関数 M-ファイルの構文は

```
function [output_list] = function_name(input_list)
definition_of_function
```

である。関数 M-ファイルの第 1 行の書き出しは function で始めなければならない。角括弧 [] で囲まれた出力パラメータはオプションである。パラメータが 1 つならベクトルであることを表す角括弧は必要ないし、出力パラメータがない場合は等式の等号と左辺は省略する。文字列 function_name はこの関数を呼び出すときに使う。関数名に続く入力パラメータ input_list はオプションである。関数の定義は definition_of_function でなされる。1 つの関数 M-ファイルにはただ 1 つの関数が定義される。入力と出力のパラメータリストはコンマで区切られた変数のリスト、つまり変数を成分とする行ベクトルである。

これらの変数にはスカラー、ベクトル、行列、文字列が許される。MATLAB はこれらの変数を含んだ計算やこれらの変数に対する作用が実行されるまではこれらの変数のタイプを区別しない。たとえば実行文 $y = \sin(x)$ は組込み関数のサイン関数を呼び出すのだが、 x がスカラーなら y もスカラーになるし、 x が行列なら y も行列になるのである。この入力変数と出力変数に対応して演算できる能力は非常に強力である。

底辺の長さが b で高さが h の三角形の面積を求める関数 triarea.m を考えてみよう。入力パラメータは b と h 、出力パラメータを area として

```
function area = triarea(b,h)
area = 0.5*b*h;
```

通常、MATLAB の関数には return が必要でないことを注意しておく。出力パラメータ area はファイルの第 1 行で定義されている。出力パラメータ area に割り当てられた値は関数 triarea を呼び出した関数もしくはコマンドウィンドウに返される。この関数 M-ファイルを MATLAB 検索パス上のディレクトリにファイル名 triarea.m で保存した後、 triarea(3,4) を入力すると、出力 ans = 6 を得る。

スクリプト M-ファイル

MATLAB のコマンドリストを含んだ M-ファイルをスクリプト M-ファイルと呼ぶ。簡単にスクリプトあるいは M-ファイルと呼ばれることも多い。これらのスクリプトは関数 M-ファイルとは違い、function から始まらない。スクリプト script_name.m を実行するにはプロンプトで script_name とタイプするだけでよい。スクリプトに記述された一連のコマンドが逐次実行される。スクリプト M-ファイルは関数 M-ファイルと違って出力する値はなく、MATLAB のワークスペースと同じ変数が使われる。

スクリプト M-ファイルの構文

スクリプト M-ファイルは `help` コマンドのためのコメント文を必要としない。M-ファイルであるから拡張子 `.m` を持つファイル名で保存する。スクリプトファイルを作るにはテキストエディタで実行したいコマンドを順に書き込んだテキストファイルを作るだけでよい。

4 Matlab ODE ソルバーの使用例

例 1 :

次の van der Pol 方程式を考える。

$$x'' + \mu(x^2 - 1)x' + x = 0.$$

これを解くために同値な 1 階の連立常微分方程式で表現する。 $x_1 := x$, $x_2 := x_1'$ とおくと

$$\begin{aligned}x_1' &= x_2, \\x_2' &= \mu x_2(1 - x_1^2) - x_1\end{aligned}$$

である。 $\mu = 1$ のとき、van der Pol 方程式を表現する関数 M-ファイルは

```
function xprime = vdpol(t,x)
xprime = [x(2); x(2).*(1-x(1).^2)-x(1)];
```

`ode23` を使うスクリプトを下に示す。

```
t0 = 0; tf = 20; tspan = [t0,tf];
x0 = [0 0.25]';
[t,x] = ode23('vdpol',tspan,x0);
plot(t,x(:,1));
```

Built-in interpolant により解をプロットするのが簡単である。ここではコマンド `plot` を使って図 1 が得られる。さらに 2 次元空間に彗星のような動きを伴った軌跡でグラフを表示する `comet` などもある。 ■

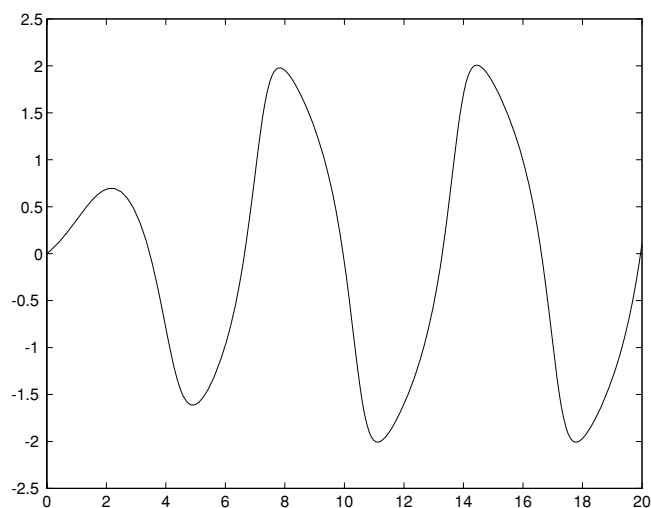


図 1: van der Pol 方程式の解

例 2 :

次の初期値問題 $y' = Ay$, $y(0) = y_0$ を考える .

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}' = - \begin{bmatrix} 1 & 0 \\ 0 & 10^q \end{bmatrix} \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}, \quad \begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} .$$

A の固有値は

$$\lambda_1 = -1, \quad \lambda_2 = -10^q$$

であるから , 硬度比は

$$r = 10^q$$

であり , 解は

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} e^{-t} \\ e^{-10^q t} \end{bmatrix}$$

である . 解の第 2 の部分は , q が大きいとき , 速く減少する因子 $\exp(-10^q t)$ を持つが , 大きな硬さのために , 予測子・修正子法を含めたいかなる陽的解法もステップ幅が制限される .

$q = 1, q = 5$ として , 5 つの MATLAB 常微分方程式ソルバーにより方程式を解き , 硬度比がステップ幅に与える影響を見てみよう . 関数 M-ファイル `exp_2.m` は

```
function uprime = exp_2(t,u); % Example 2
global q % global variable
A=[-1 0;0 -10^q]; % matrix A
uprime = A*u;
```

次のスクリプトは $q = 1$, つまり $r = e^{10}$ の場合の硬くない系の初期値問題を相対許容誤差 (relative tolerance) が 10^{-12} , 絶対許容誤差 (absolute tolerance) が 10^{-14} で解く . オプション `stats on` とすることにより , 関数を評価した回数を記録できる . 同様に , $q = 5$, つまり $r = \exp(10^5)$ の場合の疑似的硬い系の初期値問題を解くことができる .

```
clear;
global q; q = 1;
tspan = [0 1]; y0 = [1 1]';
options = odeset('RelTol',1e-12,'AbsTol',1e-14,'Stats','on');
[x23,y23] = ode23('exp_2',tspan,y0,options);
[x45,y45] = ode45('exp_2',tspan,y0,options);
[x113,y113] = ode113('exp_2',tspan,y0,options);
[x23s,y23s] = ode23s('exp_2',tspan,y0,options);
[x15s,y15s] = ode15s('exp_2',tspan,y0,options);
```

$q = 1, q = 5$ のときに MATLAB ODE Suite の 5 つのソルバーを使って解いた場合のステップ数を表 1 にまとめる . これより , `nonstiff` ソルバーが疑似的硬い系を解くとき , 非常に遅く , かつ高価であることがわかる . ■

5 Matlab ODE Demos

MATLAB `odedemo` には MATLAB ode suite のパフォーマンスを見ることができる微分方程式の `non-stiff` 問題を 4 問題と `stiff` 問題を 15 問題が関数 M-ファイルとして与えられている .

表 1: $q = 1$, 相対許容誤差 $RT = 10^{-3}$, 絶対許容誤差 $AT = 10^{-6}$ のときのステップ数と $q = 5$, 相対許容誤差 $RT = 10^{-12}$, 絶対許容誤差 $AT = 10^{-14}$ のときのステップ数の比較 .

(RT, AT)	$(10^{-3}, 10^{-6})$		$(10^{-12}, 10^{-14})$	
	1	5	1	5
ode23	29	39 823	24 450	65 944
ode45	13	30 143	601	30 856
ode113	28	62 371	132	64 317
ode23s	37	57	30 500	36 925
ode15s	43	89	773	1 128

Matlab odedemo の non-stiff 問題

1. orbitode 問題: ORBITODE は制限された 3 体問題である . Shampine and Gordon [12], p. 246 にある non-stiff ソルバーの標準的なテスト問題である .
2. orbt2ode 問題: ORBT2ODE は Hull *et al.* [13] の [14], p. 121 にある non-stiff problem D5 である .
3. rigidode 問題: RIGIDODE は外力がない場合の剛体のオイラー方程式を解く . これは Krogh による non-stiff ソルバーの標準的なテスト問題である . Shampine and Gordon [12], p. 243 に図示されている解の場合には積分区間 $[t_0, t_f]$ はおよそ 1.5 周期である .
4. vdpode 問題: VDPODE はパラメータを持つ van der Pol 方程式である . μ が小さい場合 (たとえば $\mu = 1$ のとき) は non-stiff である .

Matlab odedemo の stiff 問題

1. a2ode 問題と a3ode 問題: A2ODE と A3ODE は [15] の Problem A2 で与えられた実固有値を持つ stiff 線型問題である .
2. b5ode 問題: B5ODE は [15] の Problem B5 で与えられた複素固有値を持つ stiff 線型問題である . Shampine [14], p. 298, Ex. 5 を参照せよ .
3. buiode 問題: BUIODE は Bui による解析解を持つ stiff 問題である . ここでのパラメータは [16] で述べられた最も硬い場合に対応する .
4. brussode 問題: BRUSSODE は化学反応 (the Brusselator) にモデルを持つ stiff 問題である [7] .
5. chm6ode 問題: CHM6ODE は Enright and Hull [17] の stiff 問題 CHM6 である . 触媒流動層力学 (下方から流体を吹送して粒子を浮遊状態にした層) にモデルを持つ微分方程式系である .
6. chm7ode 問題: CHM7ODE は Enright and Hull [17] の stiff 問題 CHM7 である . オゾンの熱分解にモデルを持つ微分方程式系である .
7. chm9ode 問題: CHM9ODE は Enright and Hull [17] の stiff 問題 CHM9 である . 有名な Belousov の振動化学系の方程式である . たとえば Aiken [18], p. 49 を参照せよ .
8. d1ode 問題: D1ODE は [15] の Problem D1 で与えられた実固有値を持つ stiff 非線型問題である . 原子炉理論にモデルを持つ微分方程式系である . [15] では自励系に変換されていたが, ここではもともとの非自励系として扱う . van der Houwen は [19], p. 151 で値

$$t = 400, \quad y(1) = 22.24222011, \quad y(2) = 27.11071335$$

を使っている .

9. fem1ode 問題 : FEM1ODE は時間に依存する質量行列を持つ微分方程式系

$$M(t)y' = f(t, y)$$

からなる stiff 問題である .

10. fem2ode 問題 : FEM2ODE は時間に依存しない質量行列を持つ微分方程式系

$$My' = f(t, y)$$

からなる stiff 問題である .

11. gearode 問題 : GEARODE は Gear による簡単な stiff 問題である . van der Houwen は [19], p. 148 で値

$$t = 50, \quad y(1) = 0.5976546988, \quad y(2) = 1.40234334075$$

を使っている .

12. hb1ode 問題 : HB1ODE は Hindmarsh and Byrne [20] の stiff problem 1 である .
13. hb2ode 問題 : HB2ODE は Hindmarsh and Byrne [20] の stiff problem 2 である . ステップサイズ
の選択スキームを損なう非自励系の日周期的な動力学問題である . 絶対許容誤差を非常に小さい値とし
なければならない例である .
14. hb3ode 問題 : HB3ODE は Hindmarsh and Byrne [20] の stiff problem 3 である . 日周期的な変動問
題である .
15. vdpode 問題 : VDPODE VDPODE はパラメータを持つ van der Pol 方程式である . μ が大きい場
合 (たとえば $\mu = 1000$ のとき) は stiff である [22] .

non-stiff 問題と stiff 問題からそれぞれ 1 つ選び紹介しよう .

Demo 1 : orbitode 制限された 3 体問題 (non-stiff)

$\mu := 1/82.45$, $\mu^* := 1 - \mu$ とおく . 平面上に質量の比が $\mu : \mu^*$ である 2 つの質点 A, B を考える . 質点 A, B の座標をそれぞれ $A(\mu^*, 0)$, $B(-\mu, 0)$ とすると , 2 つの質点 A, B の重心は原点 O にある . このとき , 無限小の質量を持つ質点 C が質点 A, B の周りを運動する運動方程式を考えよう . 質点 C の位置の座標を $(y_1(t), y_2(t))$ とおき , 速度ベクトルを $(y_3(t), y_4(t))$ とおく .

$$r_{13}(t) := \{(y_1(t) + \mu)^2 + y_2(t)^2\}^{3/2},$$
$$r_{23}(t) := \{(y_1(t) - \mu^*)^2 + y_2(t)^2\}^{3/2}$$

とすると , 質点 C は次の微分方程式系

$$\begin{aligned} y_1'(t) &= y_3(t), \\ y_2'(t) &= y_4(t), \\ y_3'(t) &= 2y_4(t) + y_1(t) - \frac{\mu^*}{r_{13}(t)}(y_1(t) + \mu) - \frac{\mu}{r_{23}(t)}(y_1(t) - \mu^*), \\ y_4'(t) &= -2y_3(t) + y_2(t) - \frac{\mu^*}{r_{13}(t)}y_2(t) - \frac{\mu}{r_{23}(t)}y_2(t) \end{aligned} \tag{1}$$

を満たす . この微分方程式系を

$$y' = f_{\text{orbitode}}(t, y)$$

と表わそう．この微分方程式系は non-stiff ソルバーをテストするための標準的な問題である．この微分方程式系は [12] による．

解 $y(t) = [y_1(t), y_2(t), y_3(t), y_4(t)]^T$ の初めの 2 つの成分は質点 C の座標を表す．したがって，座標平面に点 $(y_1(t), y_2(t))$ をプロットすると，2 つの質点 A, B の周りを回る質点 C の軌道を与える．初期値は軌道が周期的になるように選ぶ．軌道の定性的挙動を再現するにはかなり厳重な許容誤差が必要となる．具体的には，相対許容誤差として 10^{-5} 程度，絶対許容誤差として 10^{-4} 程度である．

この微分方程式系を関数 M-ファイルで表現すると，

```
function dydt = f_orbitode(t,y)
mu = 1 / 82.45;
mustar = 1 - mu;
r13 = ((y(1) + mu)^2 + y(2)^2) ^ 1.5;
r23 = ((y(1) - mustar)^2 + y(2)^2) ^ 1.5;
dydt = [ y(3)
         y(4)
         (2*y(4) + y(1) - mustar*((y(1)+mu)/r13) - mu*((y(1)-mustar)/r23))
         (-2*y(3) + y(2) - mustar*(y(2)/r13) - mu*(y(2)/r23)) ];
```

である．

区間 $[0, 6.19216933131963970674]$ において微分方程式系 (1) を初期値

$$y(0) = [y_1(0), y_2(0), y_3(0), y_4(0)]^T = [1.2, 0, 0, -1.04935750983031990726]^T$$

で解こう．ODE ソルバーには non-stiff ソルバー ode45 を使う．スクリプトは

```
clear; close all;
tspan = [0; 6.19216933131963970674];
y0 = [1.2; 0; 0; -1.04935750983031990726];
options = odeset('RelTol',1e-5,'AbsTol',1e-4);
[t,y] = ode45('f_orbitode',tspan,y0,options);
plot(y(:,1),y(:,2),'o'); title('Plot of orbit of the particle');
```

である．このようにして図 2 が得られる．

さらに，速度ベクトルの第 1 成分を加えて，いわゆる 3 次元相空間表示を行うスクリプトは

```
plot3(y(:,1),y(:,2),y(:,3),'o'); grid on;
title('Plot of orbit in 3D phase space');
```

である．このようにして図 3 が得られる．

Demo 2 : a2ode 回路理論に現れる実固有値を持つ硬い線型方程式系

次の微分方程式系

$$\begin{aligned} y_1'(t) &= -1800y_1(t) + 900y_2(t), \\ y_i'(t) &= y_{i-1}(t) - 2y_i(t) + y_{i+1}(t), \quad i = 2, \dots, 8, \\ y_1'(t) &= 1000y_8(t) - 2000y_9(t) + 1000 \end{aligned} \quad (2)$$

を考える．この微分方程式系を

$$y' = f_{a2ode}(t, y)$$

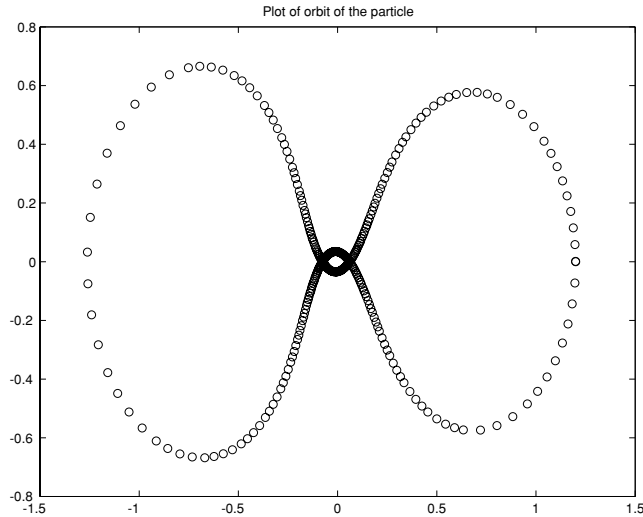


図 2: 微分方程式系 orbitode の解

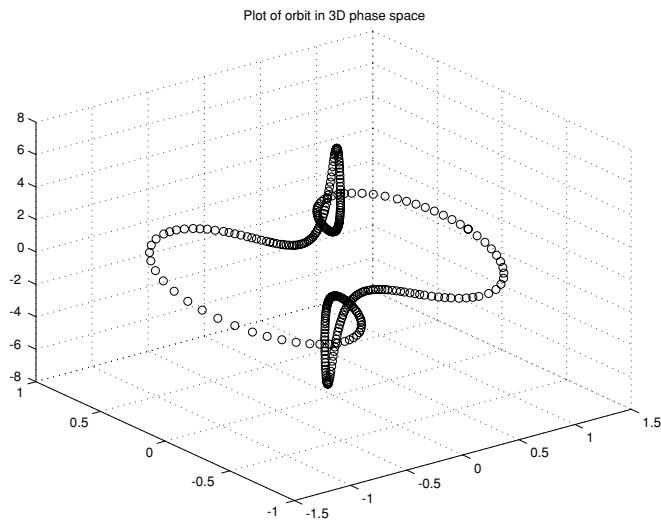


図 3: 微分方程式系 orbitode の解の 3 次元相空間表示

と表わそう。この微分方程式系は実固有値を持つ線型の硬い系である。この微分方程式系は [15] の Problem A2 による。

微分方程式系が自励系であるため、 $f_{a2ode}(t, y)$ の t に関する偏導関数は定数である。また、ヤコビ行列式は三重対角の定数行列である。stiff ソルバー ode15s と ode23s にはヤコビ行列式に関する 4 つのオプション JConstant, Jacobian, JPattern, Vectorized がある。stiff ソルバーのオプション JConstant を 'on' として、stiff ソルバーにヤコビ行列式の情報を与えると、ヤコビ行列式を計算することが不用となるため、より信頼のできる計算結果をより短い計算時間で得ることができる。

この微分方程式系を関数 M-ファイルで表現すると、

```
function dydt = f_a2ode(t,y)
dydt = zeros(9,size(y,2));           % preallocate dy/dt
dydt(1,:) = -1800*y(1,:) + 900*y(2,:); % Vectorized
i = (2:8);
dydt(i,:) = y(i-1,:) - 2*y(i,:) + y(i+1,:);
```

```
dydt(9,:) = 1000*y(8,:) - 2000*y(9,:) + 1000;
```

である . また , ヤコビ行列式を関数 M-ファイルで表現すると ,

```
function dfdy = jacobian(t,y)
dfdy = diag(ones(8,1),-1) - diag(2*ones(9,1)) + diag(ones(8,1),1);
dfdy(1,1) = -1800;
dfdy(1,2) = 900;
dfdy(9,8) = 1000;
dfdy(9,9) = -2000;
```

である .

区間 $[0, 5]$ において微分方程式系 (2) を初期値

$$y(0) = (0, \dots, 0)$$

で解こう . ODE ソルバーには non-stiff ソルバー ode45 と stiff ソルバー ode23s を使い , パフォーマンスを比較しよう . スクリプトは

```
clear; close all;
tspan = [0; 5];
y0 = zeros(9,1);
options45 = odeset('Stats','on');
options23s = odeset('JConstant','on','Vectorized','on','Stats','on');
tic; Time = [toc];
[t45,y45] = ode45('f_a2ode',tspan,y0,options45);
Time = [Time, toc]; RunTime_45 = diff(Time)
subplot(2,2,1); plot(t45,y45,'o'); title('Plot of solution by ode45');
Time = [toc];
[t23s,y23s] = ode23s('f_a2ode',tspan,y0,options23s);
Time = [Time, toc]; RunTime_23s = diff(Time)
subplot(2,2,2); plot(t23s,y23s,'o'); title('Plot of solution by ode23s');
```

とする . ode45 の 'Stats' と実行時間は

```
3023 successful steps
203 failed attempts
19357 function evaluations
0 partial derivatives
0 LU decompositions
0 solutions of linear systems
```

```
RunTime_45 = 97.2257
```

である . ode23s の 'Stats' と実行時間は

```
53 successful steps
0 failed attempts
171 function evaluations
1 partial derivatives
```

53 LU decompositions
159 solutions of linear systems

RunTime_23s = 0.6893

である．このようにして図 4 が得られる．

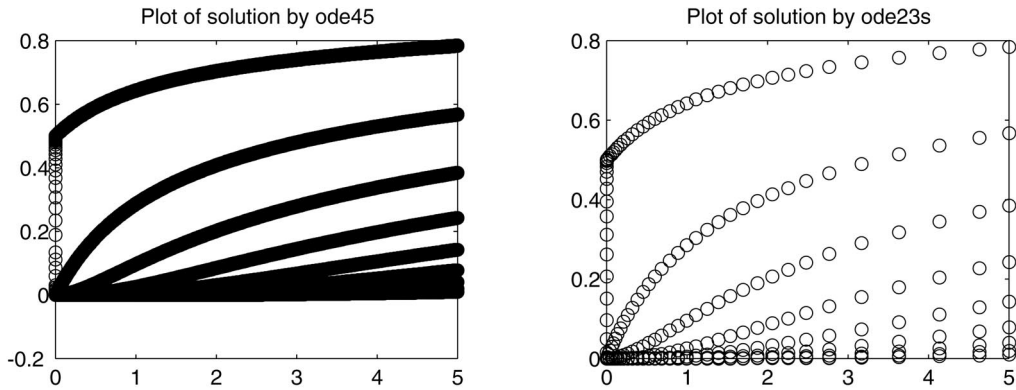


図 4: 微分方程式系 a2ode の解

参考文献

- [1] 芦野 隆一・Rémi Vaillancourt 『はやわかり MATLAB』 共立出版 1997.
- [2] 三井 斌友 『微分方程式の数値解法 I』 岩波講座 応用数学, 岩波書店 1993.
- [3] 芦野 隆一・Rémi Vaillancourt 『MATLAB による微分方程式とラプラス変換』 共立出版 2000.
- [4] R. Ashino, M. Nagase and R. Vaillancourt, *Behind and beyond the Matlab ODE suite*, Computers Math. Applic. **40** (2000) 491 – 512.
- [5] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., The John Hopkins University Press, 1996.
- [6] A. Iserles, *A First Course in the Numerical Analysis of Differential Equations*, Cambridge Texts in Applied Mathematics, Cambridge, 1996.
- [7] E. Hairer and G. Wanner, *Solving ordinary differential equations II, stiff and differential-algebraic problems*, Springer-Verlag, Berlin, 1991, pp. 5–8.
- [8] J. D. Lambert, *Numerical methods for ordinary differential equations. The initial value problem*, Wiley, Chichester, 1991.
- [9] J. R. Dormand and P. J. Prince, *A family of embedded Runge–Kutta formulae*, J. Computational and Applied Mathematics, **6**(2) (1980), 19–26.
- [10] L. F. Shampine and M. W. Reichelt, *The MATLAB ODE Suite*, SIAM Journal on Scientific Computing, **18**(1), (1997) 1–22.
- [11] *Using MATLAB*, Version, 5.1, The MathWorks, Chapter 8, Natick, MA, 1997.

- [12] L. F. Shampine and M. K. Gordon, *Computer solution of ordinary differential equations*, W.H. Freeman & Co., San Francisco, (1975) p. 246.
- [13] T. E. Hull, W. H. Enright, B. M. Fellen, and A. E. Sedgwick, *Comparing numerical methods for ordinary differential equations*, SIAM J. Numer. Anal., **9**(4) (1972) 603–637.
- [14] L. F. Shampine, *Numerical solution of ordinary differential equations*, Chapman & Hall, New York, 1994.
- [15] W. H. Enright, T. E. Hull, and B. Lindberg, *Comparing numerical methods for stiff systems of ODEs*, BIT **15**(1) (1975), 10–48.
- [16] L. F. Shampine, *Measuring stiffness*, Appl. Numer. Math., **1**(2) (1985), 107–119.
- [17] W. H. Enright and T. E. Hull, *Comparing numerical methods for the solution of stiff systems of ODEs arising in chemistry*, in Numerical Methods for Differential Systems, L. Lapidus and W. E. Schiesser eds., Academic Press, Orlando, FL, 1976, pp. 45–67.
- [18] R. C. Aiken, ed., *Stiff computation*, Oxford Univ. Press, Oxford, 1985.
- [19] P. J. van der Houwen, *Construction of integration formulas for initial value problems*, North-Holland Publishing Co., Amsterdam, 1977.
- [20] A. C. Hindmarsh and G. D. Byrne, *Applications of EPISODE: An experimental package for the integration of ordinary differential equations*, in Numerical Methods for Differential Systems, L. Lapidus and W. E. Schiesser eds., Academic Press, Orlando, FL, 1976, pp. 147–166.
- [21] D. Kahaner, C. Moler, and S. Nash, *Numerical methods and software*, Prentice-Hall, Englewood Cliffs, NJ, 1989.
- [22] L. F. Shampine, *Evaluation of a test set for stiff ODE solvers*, ACM Trans. Math. Soft., **7**(4) (1981) 409–420.
- [23] J. D. Lambert, *Computational methods in ordinary differential equations*, Wiley, London, 1973, Chapter 5.
- [24] J. C. Butcher, *The numerical analysis of ordinary differential equations. Runge–Kutta and general linear methods*, Wiley, Chichester, 1987, Chapter 4.
- [25] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving ordinary differential equations I, nonstiff problems*, Springer-Verlag, Berlin, 1987, Section III.8.