電気 303/電情 303 数値解析 (12)

微分方程式の数値解法 (2)
Runge-Kutta 法, 多段法

記号や用語および式の復習

- 解くべき微分方程式は

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, t)$$

  である.

- 微分方程式の解の存在と一意性を仮定する.

- この講義で取り扱う解法は離散変数法と呼ばれる解法である. の解法は, 離散的な時刻

$$t_0, t_1, \ldots, t_N$$

で微分方程式の数値解を計算する方法である.

- $h_k = t_{k+1} - t_k$ をステップ幅という. h_k が k によらず一定のときには h と書く.
- $\boldsymbol{x}(t_k)$ のことを $\boldsymbol{x}_k$ と書く.

- 常微分方程式の数値解法は
  - ▷ 1 段法
  - ▷ 多段法

  の 2 種類に大別される. 段は英単語 step の訳.

- 1 段法の一般形は以下の通り.
  - ▷ **Explicit(陽的) な公式:**

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + h_k \boldsymbol{\Psi}(t_k, \boldsymbol{\xi}_k, h_k)$$

  - ▷ **Implicit(陰的) な公式:**

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + h_k \boldsymbol{\Psi}(t_k, t_{k+1}, \boldsymbol{\xi}_k, \boldsymbol{\xi}_{k+1}, h_k)$$

- 前ページの式において, $\boldsymbol{\Psi}(\cdot)$ は, $\boldsymbol{x}_{k+1}$ と $\boldsymbol{x}_k$ との関係を与える写像 $\boldsymbol{\Phi}(\cdot)$ の近似である.
- $\boldsymbol{\Phi}(\cdot)$ は未知である一方で, $\boldsymbol{\Psi}(\cdot)$ は構成可能である.

- $\boldsymbol{\Psi}(\cdot)$ の構成法に関し, 前回の講義では, もっとも簡単な Euler 法 のみを取り扱った.
- 今回の講義では, まず Euler 法より精密な $\boldsymbol{\Psi}$ の構成法である Runge-Kutta 型公式について述べる (おもに explicit な公式).
- 続いて多段法について述べる.
- 参考文献は前回と同じなので再掲しない.

- 以下では, 微分方程式

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, t)$$

の解を, 初期値や初期時刻を省略して, $\boldsymbol{x}(t)$ と書く.

Runge-Kutta 型公式

- Runge-Kutta 型公式は代表的な 1 段法である.

- 平均値の定理から,

  $$\exists \boldsymbol{\Phi}(t_k, \boldsymbol{x}_k, h_k), ; \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + h_k \boldsymbol{\Phi}(t_k, \boldsymbol{x}_k, h_k)$$

  となるが, $\boldsymbol{\Phi}(\cdot)$ は未知である.

- Runge-Kutta 型の解法は, 未知の $\boldsymbol{\Phi}(\cdot)$ を, 一定の方法 (後述) を用い, 既知の $\boldsymbol{\Psi}(\cdot)$ で近似する.

- $\boldsymbol{\Phi}(\cdot)$ とその近似関数 $\boldsymbol{\Psi}(\cdot)$ の偏差

$$\boldsymbol{\Psi}(t, \boldsymbol{x}(t), \tau) - \boldsymbol{\Phi}(t, \boldsymbol{x}(t), \tau)$$

(ただし, 変数 h_k を τ に変更している) を局所離散化誤差といい,

$$\boldsymbol{\tau}(t, \tau)$$

と書く.

- ある公式が

$$\exists C > 0, \exists \rho > 0, \forall r : 0 < r < \rho, \forall t \in [a, b],$$
$$\|\boldsymbol{\tau}(t, r)\|_\infty \leq C r^p$$

を満たすとき, その公式を p 次精度の公式, あるいは p 次の公式という.

- ただし, 区間の右端では, たとえば

$$\boldsymbol{x}(t+r) - \boldsymbol{x}(t)$$

を

$$\boldsymbol{x}(t) - \boldsymbol{x}(t-r)$$

で読み換えるなどといった, 若干の読み換えが必要である.

- ステップ幅が一定であるとき (h とする), p 次精度の公式に対し, ある $L > 0$ が存在して,

$$\forall k, \|\boldsymbol{x}_k - \boldsymbol{\xi}_k\| \leq \frac{e^{L(b-a)}}{L} C h^p$$

となることが示せる ([齊藤], pp. 225–226).

- Runge-Kutta 型の公式は, いくつかの点における $\boldsymbol{f}(\boldsymbol{x}, t)$ の値を組み合わせて Ψ を構成することで p 次精度 $(p \geq 1)$ の公式を得る方法の総称である.
- Runge-Kutta 型の公式には, 色々なバリエーションがある.

- 前回の講義で述べた Heun 法は, 実は Runge-Kutta 法の一種である.

- Ψ を構成するために関数 $\boldsymbol{f}$ の値を s 個の点において評価するとき, これを **s 段数** (**s-stage**) の Runge-Kutta 法と呼ぶ ([齊藤]).

- s 段数の Runge-Kutta 法を構成する目的は, その公式が p 次精度となるようにすることである.
- $s \geq p$ である必要があることが示せるが ([齊藤]), $s = p$ とは限らない.

- 一般論の前に, 代表的な公式を紹介する (見方がわかればよく, 覚える必要はない).
- 以下の公式では, ある p に対し, 差分方程式

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + h_k \boldsymbol{\Psi}(t_k, \boldsymbol{\xi}_k, h_k)$$

における関数 Ψ が, p 次精度となるように構成されている.

§2 段数の Runge-Kutta 法

- 2 段数の Runge-Kutta 法は,

$$\boldsymbol{k}_1 = \boldsymbol{f}(\boldsymbol{\xi}_k, t_k),$$
$$\boldsymbol{k}_2 = \boldsymbol{f}(\boldsymbol{\xi}_k + h_k\beta\boldsymbol{k}_1, t_k + \alpha h_k)$$

とし,

$$\Psi(t_k, \boldsymbol{\xi}_k) = c_1\boldsymbol{k}_1 + c_2\boldsymbol{k}_2$$

とする公式である.

- α, β,c_1, c_2 は任意ではなく, これをうまく選ぶことにより, 2 次精度の公式が得られる.

- 具体的には, $\alpha > 0$ とし,

$$\beta = \alpha,$$

$$c_1 = 1 - \frac{1}{2\alpha}$$

$$c_2 = \frac{1}{2\alpha}$$

とすると, 2 次精度が達成できることが示せる.

- α の値は正であれば任意なので, 2 段数 2 次精度の explicit な Runge-Kutta 法には, 無限のバリエーションがある ([齊藤], pp. 227–229).

- よく用いられるパラメータ値は以下の通り.

2 段数 Runge-Kutta 法

	α	β	c_1	c_2
Heun 法	1	1	$\frac{1}{2}$	$\frac{1}{2}$
改良 Euler 法	$\frac{1}{2}$	$\frac{1}{2}$	0	1

§ パラメータの決定法

- Runge-Kutta 法のパラメータは, 関数 f を形式的に Taylor 展開し, その低次の項が零になるように係数合わせをすることによって得られる. 計算は繁雑ではあるが, 機械的にできる.
- この計算の過程はこの講義では興味の対象ではないので, 説明を省略する.

- 「低次の項が零になるような係数合わせ」の解は一意的ではなく, 様々な自由度がある. 少ない段数の Runge-Kutta 法では, すべての解を少数のパラメータで特徴付ける方法が知られている.

- とはいっても, すべての Runge-Kutta 法を特徴付けることにはあまり意味がないので, 以下では, まずよく知られたパラメータの選び方のみをを紹介する.
- 以下の議論では, 「なぜそんなパラメータを選ぶか」が一見しも明らかでないと思われるが, それらは「辻褄が合うように係数合わせをして, うまくいったものの例」に過ぎない.

§4 段数の Runge-Kutta 法

- 4 段数の Runge-Kutta 法では,

$$\boldsymbol{k}_j = \boldsymbol{f}(\boldsymbol{\xi}_k + h_k \sum_{l<j} \beta_{jl} \boldsymbol{k}_l, t_k + \alpha_j h_k),$$

$$j = 1, \ldots, 4, \sum_{l<j} \beta_{jl} = \alpha_j$$

のように $(\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{k}_3, \boldsymbol{k}_4)$ を取る.

- 4 段数の Runge-Kutta 法は, 上記の

$$(\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{k}_3, \boldsymbol{k}_4)$$

を用い,

$$\Psi(t_k, \boldsymbol{\xi}_k) = \sum_{j=1}^{4} c_j \boldsymbol{k}_j$$

とする公式である.

- 係数 c_j は,

$$\sum_{j=1}^{4} c_j = 1$$

を満たすように取るのであるが, この取り方には自由度がある.

- なお, $j = 1$ のときは $\sum_{l<j} \cdots$ の項は $\cdots$ の部分にかかわらず零と定義する.

- 4 段数の Runge-Kutta 法にも色々なバリエーションがあり得るが, もっともよく用いられるのは, Runge の 1/6 公式と呼ばれる公式 (次ページ) である. 4 次精度である.

Runge の 1/6 公式

α_1	α_2	α_3	α_4	c_1	c_2	c_3	c_4
0	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$
$\beta_{21} = \frac{1}{2}, \beta_{32} = \frac{1}{2}, \beta_{43} = 1$							
記載のないパラメータは零							

- Runge の 1/6 公式は，RK4，RK41，古典的 Runge-Kutta 法など, 色々の名前で呼ばれる.

§Runge-Kutta 法の一般形とその表記

- Runge-Kutta 法の一般形とその表記法について述べる.

- s 段数の Runge-Kutta 法の一般形は,

$$\Psi(t_k, \boldsymbol{\xi}_k) = \sum_{j=1}^{s} c_j \boldsymbol{k}_j,$$

という形である. ただし,

$$\sum_{j=1}^{s} c_j = 1$$

である.

- Explicit(陽的) な Runge-Kutta 法では,

$$\boldsymbol{k}_j = \boldsymbol{f}(\boldsymbol{\xi}_k + h_k \sum_{l<j} \beta_{jl} \boldsymbol{k}_l, t_k + \alpha_j h_k)$$

とする. ただし, $\alpha_j = \sum_{l<j} \beta_{jl}$ である.

- Implicit(陰的) な Runge-Kutta 法では,

$$\boldsymbol{k}_j = \boldsymbol{f}(\boldsymbol{\xi}_k + h_k \sum_{l=1}^{s} \beta_{jl} \boldsymbol{k}_l, t_k + \alpha_j h_k)$$

とする. ただし, $\alpha_j = \sum_{l<j} \beta_{jl}$ である.

- Explicit(陽的) な Runge-Kutta 法と Implicit(陰的) な Runge-Kutta 法の相異は一見ではわかりにくいが,

$$\sum_{l<j}$$

と

$$\sum_{l=1}^{s}$$

が違う点である.

- Euler 法は 1 段数の Runge-Kutta 法であると解釈できる.

- s 段数の Runge-Kutta 型公式は,
  - ▷ s 個のパラメータ $\alpha_1, \ldots, \alpha_s$,
  - ▷ s 個のパラメータ $c_1, \ldots, c_s$
  - ▷ s^2 個のパラメータ (β_{il})

  によって特徴付けられる. ただし, $\alpha_j = \sum_{l=1}^{s} \beta_{jl}$ という制約条件が課されている.

- これらのパラメータは, 希望する精度を実現するために調整されるのであるが, 希望する精度が達成できる場合と, そうでない場合がある.
- 文献に記載されたり, 数値計算ソフトウェアに採用されていたりするものは, 希望する精度を達成したパラメータの組み合わせである.

- Runge-Kutta 型公式のパラメータを簡潔に表示するために, Butcher 配列という形式が用いられる. これは, 次のような表である.

$$
\begin{array}{c|cccc}
\alpha_1 & \beta_{11} & \beta_{12} & \cdots & \beta_{1s} \\
\alpha_2 & \beta_{21} & \beta_{22} & \cdots & \beta_{2s} \\
\vdots & \vdots & \vdots & & \vdots \\
\alpha_s & \beta_{s1} & \beta_{s2} & \cdots & \beta_{ss} \\
\hline
 & c_1 & c_2 & \cdots & c_s
\end{array}
$$

- Explicit な公式の Butcher 配列では, 右下がりの対角線とその上の要素は零である.

- $s = p = 2$, $s = p = 3$, $s = p = 4$ の場合は, パラメータを含んだ Runge-Kutta 型公式の一般形 (の Butcher 配列) を書き下すことができる ([Butcher]).

- $s = p = 2$ とした場合の explicit な公式の一般形についてはすでに述べた.

- $s = p = 3$, $s = p = 4$ については, パラメータ値に関する場合分けが必要で繁雑なので, この講義では述べない ([Butcher] 参照).

- 以下, Butcher 配列の例を示す ([齊藤]).

- 1 段数, 1 次精度の公式 ([齊藤])
  - ▷ 前進 Euler 法:

0	0
	1

  - ▷ 後退 Euler 法:

1	1
	1

- 2 段数, 2 次精度の公式 ([齊藤])
  - ▷ Heun 法:

0	0	0
1	1	0
	$\frac{1}{2}$	$\frac{1}{2}$

▷ 改良 Euler 法:

0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0
	0	1

▷ Crank–Nicolson 法:

0	0	0
1	$\frac{1}{2}$	$\frac{1}{2}$
	$\frac{1}{2}$	$\frac{1}{2}$

- 3 段数, 3 次精度の公式 ([齊藤])

  - ▷ Heun 法:

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & 0 & 0 \\ \frac{2}{3} & -\frac{2}{9} & \frac{8}{9} & 0 \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array}$$

▷ Kutta 法:

0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	0
1	-1	2	0
	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$

- 4 段数, 3 次精度の公式 ([斎藤])

  - ▷ Lambert 法

0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	0	0
-1	$\frac{1}{2}$	$-\frac{3}{2}$	0	0
1	0	$\frac{4}{3}$	$-\frac{1}{3}$	0
	$\frac{1}{6}$	$\frac{2}{3}$	0	$\frac{1}{6}$

- 4 段数, 4 次精度の公式 ([斎藤])

  - ▷ Runge の 1/6 公式:

0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	0	0
$\frac{1}{2}$	0	$\frac{1}{2}$	0	0
1	0	0	1	0
	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$

- Explicit な Runge-Kutta 法に関し，次ページで述べるようなが知られている ([Hairer et al], Vol. 1).

- $p \geq 5$ のとき, p 段数では p 次精度は 実現できない.
- $p \geq 7$ のとき, $p+1$ 段数では p 次精度は実現できない.
- $p \geq 8$ のとき, $p+2$ 段数では, p 次精度は実現できない.

- 上記は不可能性に関する事実であって, 指定された下限より大きい s に対して公式の存在を保証するわけではない.

- 公式としては, 以下のようなものが知られている ([Hairer et al], Vol. 1).
  - ▷ 6 段数 5 次精度の公式
  - ▷ 7 段数 6 次精度の公式
  - ▷ 11 段数 8 次精度の公式,
  - ▷ 17 段数 10 次精度の公式

  具体的なパラメータの値は省略する.

- 浮動小数点演算の誤差のため, 高次精度の公式がつねに実用的とは限らないが, 後で述べる数値解の誤差の制御のために高次精度の公式が用いられることがある.

- 後の議論の準備のために, Butcher 配列に対して,

$$\boldsymbol{B} = (\beta_{jl})_{1\leq j\leq s, 1\leq l\leq s}$$
$$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_s)^T$$
$$\boldsymbol{c} = (c_1, \ldots, c_s)$$

と定義する.

- 上記のように $\boldsymbol{B}$, $\boldsymbol{\alpha}$ および $\boldsymbol{c}$) を定めたとき, 今後は, これに対応する Butcher 配列を

$$(\boldsymbol{B}, \boldsymbol{\alpha}, \boldsymbol{c})$$

と書くことにする.

ステップ幅と誤差の制御

- 今までの議論では, ステップ幅の決め方は議論されていなかった.

- 後に述べるように, Butcher 配列

$$(\boldsymbol{B}, \boldsymbol{\alpha}, \boldsymbol{c})$$

持つ $p+1$ 次精度の公式と, Butcher 配列

$$(\boldsymbol{B}, \boldsymbol{\alpha}, \overline{\boldsymbol{c}})$$

を持つ p 次精度の公式が存在すれば, これを使って誤差を見積ることができる.

- 前ページの 2 個の公式において, $\boldsymbol{B}$ と $\boldsymbol{\alpha}$ は共通であったことに注意する.

- いくつかの $(p, p+1)$ の組み合わせについて, 上述の条件を満たす Butcher 配列が存在することが知られている.

- このようになっているとき, s 段数 $p+1$ 次精度の公式に p 次精度の公式が埋め込まれていると表現する ([齊藤]).

- 上記のような Runge-Kutta 法の組み合わせを**埋め込み型 Runge-Kutta 法**と呼ぶ ([齊藤]). その用途は, 数値解に含まれる誤差の見積りである.

- $(\boldsymbol{B}, \boldsymbol{\alpha}, \boldsymbol{c})$ に対応する Runge-Kutta 法が生成する数値解を $(\boldsymbol{\xi}_k)$, $(\boldsymbol{B}, \boldsymbol{\alpha}, \overline{\boldsymbol{c}})$ に対応する Runge-Kutta 法が生成する数値解を $(\overline{\boldsymbol{\xi}}_k)$ とする.

- 詳細は省くが, $\|\boldsymbol{\xi}_k - \overline{\boldsymbol{\xi}}_k\|$ の値から, 数値解に含まれる誤差を見積ることができる (ただし, 近似を含むので, 精密な評価ではない).

- この見積りを用いると, 見込まれる誤差が一定以下になるようにステップ幅を調整することができる. これをステップ幅の自動調整という.

- ステップ幅の自動調整により, 数値解の正しさを, ある程度保証することができる (近似が含まれるので厳密ではない). 詳細については [齊藤] を参照.

- Scilab では RKF45 という公式が実装されているが, これは埋め込み型 Runge-Kutta 法で, 6 段数 5 次精度の公式に 4 次精度の公式が埋め込まれている.

Implicit な Runge-Kutta 型公式

- この講義では今までほとんど述べなかったが, implicit な Runge-Kutta 法は硬い問題 (後述) や differential-algebraic equation を解くのに用いられる. ただし, 内部反復を必要とするため, 必ずしも使いやすいわけではない.

- implicit な Runge-Kutta 法に興味がある者は [Hairer et el.] や [Butcher] などを参照せよ.

硬い問題

- 線形微分方程式

$$\frac{d}{dt}\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x}$$

において, 行列 $\boldsymbol{A}$ の固有値がすべて相異なり, それら実部がすべて負であるものとする.

- $\boldsymbol{A}$ の固有値のうち実部が最小のものを λ_m, 最大のものを λ_M とする.
- $|\mathrm{Re}\,\lambda_M|/|\mathrm{Re}\,\lambda_m|$ を硬度比という.
- 硬度比が大きい問題を硬い問題という.
- 硬い問題は数値的に解きにくいことが知られている.

- Explicit(陽的) な Runge-Kutta 法は, 実装も特性解析も簡単なので, 常微分方程式の数値解法としてもっとも普通に用いられるが, 硬い問題を解くには必ずしも適していない.
- このため, 硬い問題を解くときには, implicit(陰的) な公式が選択されることがある.

多段法

- 多段法は, $\boldsymbol{\xi}_{k+1}$ を構成するために, $\boldsymbol{\xi}_k$ だけでなく, 過去の数値解の系列

$$(\boldsymbol{\xi}_{k-L}, \ldots, \boldsymbol{\xi}_k)$$

に関連した情報を使う方法の総称であった.

- Runge-Kutta 法では, 複数の点での関数値の線形結合により高精度の公式を得ていたが, 多段法は,

$$(\boldsymbol{\xi}_{k-L}, \ldots, \boldsymbol{\xi}_k)$$

を使うことで類似した効果を得ることを意図している.

- 記法の便宜上, $\boldsymbol{\xi}_{k+L}$ を $(\boldsymbol{\xi}_k, \ldots, \boldsymbol{\xi}_{k+L-1})$ に関連した情報を使って構成するという形で問題を考え直す.

- 簡単のために, この講義では, 多段法ではステップ幅は h に固定されているものとする (ステップ幅可変の場合については [Hairer et al.], Vol. 1 を参照).

- Runge-Kutta 型公式のアイデアと比較すると,

$$\boldsymbol{f}(\boldsymbol{\xi}_k, t_k), \ldots, \boldsymbol{f}(\boldsymbol{\xi}_{k+L}, t_{k+L})$$

の線形結合で近似関数を構成するという方法が考えられる. すなわち,

$$\boldsymbol{\xi}_{k+L} = \boldsymbol{\xi}_k + h\left(\sum_{l=0}^{L} \beta_l^{(0)} \boldsymbol{f}(\boldsymbol{\xi}_{k+l}, t_{k+l})\right)$$

という形の公式を考える.

- $\beta_l^{(0)}$ $(0 \leq l \leq L)$ はパラメータとして残しておく.
- $\boldsymbol{\xi}_k$ を左辺に移項すると, 次式が得られる.

$$\boldsymbol{\xi}_{k+L} - \boldsymbol{\xi}_k = h \left(\sum_{l=0}^{L} \beta_l^{(0)} \boldsymbol{f}(\boldsymbol{\xi}_{k+l}, t_{k+l}) \right)$$

- 前ページの式の左辺をを

$$\{\boldsymbol{\xi}_{k+l}\}_{0\leq l\leq L}$$

をすべて使う形に拡張すると, 以下に示すような線形多段法の一般形が得られる.

$$\sum_{l=0}^{L}\alpha_l\boldsymbol{\xi}_{k+l} = h\left(\sum_{l=0}^{L}\beta_l\boldsymbol{f}(\boldsymbol{\xi}_{k+l}, t_{k+l})\right)$$

- 先の式の $\{\alpha_l\}_{0\leq l\leq L}$ と $\{\beta_l\}_{0\leq l\leq L}$ はパラメータで, これらの取り方によって色々な公式が得られる.

- 先に述べた一般形は, 以下の 2 種類に分類される.
  - ▷ **Explicit(陽的)** な公式: 右辺に $\boldsymbol{\xi}_{k+L}$ が含まれないものを
  - ▷ **Imxplicit(陰的)** な公式: 等号の右辺に $\boldsymbol{\xi}_{k+L}$ が含まれるもの

- Explicit (陽的) な公式と implicit (陰的) な公式の特徴は, 1 段法と同様である.
- 一般には explicit (陽的) な公式の方が使いやすいが, 硬い問題などでは implicit (陰的) な公式が必要になることがある.

- 多段法を非線形にまで (形式的に) 拡張することも可能, この形で一般形が述べられている本もあるので ([山本]), 念のため書いておく.

$$\sum_{l=0}^{L} \alpha_l \boldsymbol{\xi}_{k+l} = h\boldsymbol{\Psi}\left(t_k, \ldots, t_{k+L}, \boldsymbol{\xi}_k, \ldots, \boldsymbol{\xi}_{k+L}; h\right)$$

- 以下で, 代表的な多段法をいくつか紹介するが, その前に注意点を述べる.

- 段数 L の多段法を使うときには, 最初の L ステップ分, すなわち

$$\boldsymbol{\xi}_0, \cdots, \boldsymbol{\xi}_{L-1}$$

を計算するのに必要な「過去の系列」は存在しないので, これらを 1 段法等によって事前に計算しておく必要がある.

- (線形) 多段法のメリットは, パラメータ

$$\{\alpha_l\}_{0\leq l\leq L},\ \{\beta_l\}_{0\leq l\leq L}$$

をうまく選ぶと, 関数を評価する “標本点” を増やすことなく, 高次の精度の公式を得ることができることであるが, 一方で, 安定性などに関して注意すべき点も多い.

- 多段法が安定で収束するための条件として, 以下のものが知られている ([三井]). 証明はこの講義では述べない.

(1) 多項式

$$\alpha_0 + \alpha_1 z + \cdots + \alpha_L z^L$$

の根の絶対値がすべて 1 以下で絶対値 1 の根は単根,

(2) $\displaystyle\sum_{l=0}^{L} \alpha_l = 0,$

(3) $\displaystyle\sum_{l=0}^{l} l\alpha_l - \sum_{l=0}^{L} \beta_l = 0.$ ■

Adams 型公式

- 以下では,

$$\boldsymbol{x}(t_{k+L-1}) =: \boldsymbol{x}_{k+L-1}$$

と書く.

- Adams 型公式の説明に先立って, まず, 微分方程式

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, t)$$

は積分方程式

$$\boldsymbol{x}_{k+L} - \boldsymbol{x}_{k+L-1} = \int_{t_{k+L-1}}^{t_{k+L}} \boldsymbol{f}(\boldsymbol{\varphi}(\tau, t_k, \boldsymbol{x}_k), \tau) d\tau$$

と等価であることに注意する.

- 前ページの積分は

$$\boldsymbol{\varphi}(\tau, t_k, \boldsymbol{x}_k)$$

が未知だから計算できない.

- そこで, $\boldsymbol{\varphi}(\tau, t_k, \boldsymbol{x}_k)$ を時間 τ に関する 1 変数関数と見做し, これを Lagrange 補間多項式で近似することを考える.

- 近似に

$$(t_k, \boldsymbol{x}_k), \ldots, (t_{k+L-1}, \boldsymbol{x}_{k+L-1})$$

を使うと,

$$\boldsymbol{x}_{k+L} - \boldsymbol{x}_{k+L-1} \simeq \sum_{l=0}^{L-1} \beta_l \boldsymbol{f}(\boldsymbol{x}_{k+l}, t_l)$$

という形になる.

- 前ページの式は右辺に $(\boldsymbol{\xi}_{k+L}, t_{k+L})$ を含まないので explicit (陽的) である. この形の公式を **Adams-Bashforth 公式**と呼ぶ.

- 一方, 近似に

$$(t_k, \boldsymbol{x}_k), \ldots, (t_{k+L}, \boldsymbol{x}_{k+L})$$

を使うと,

$$\boldsymbol{x}_{k+L} - \boldsymbol{x}_{k+L-1} \simeq \sum_{l=0}^{L} \beta_l \boldsymbol{f}(\boldsymbol{x}_{k+l}, t_l)$$

という形になる.

- 前ページの式は右辺に $(\boldsymbol{\xi}_{k+L}, t_{k+L})$ を含むので implicit (陰的) である. この形の公式を **Adams-Moulton 公式**という.

- 右辺の係数 $\{\beta_l\}_{0\leq l\leq L}$ は, Lagrange 補間多項式を積分することで得られる. この係数は L の値だけで決まり, 問題によって変わるわけではないので, 事前に計算するか, 文献等の値をそのまま用いればよい.

- **Adams-Bashforth 公式の導出例.** $L = 2$ の場合の Adams-Bashforth 公式を導出する.

  ▷ t_k において $\boldsymbol{f}(\boldsymbol{\xi}_k, t_k)$,
  t_{k+1} において $\boldsymbol{f}(\boldsymbol{\xi}_{k+1}, t_k + 1)$
  という値を取る Lagrange 補間多項式は,

$$\frac{t - t_{k+1}}{t_k - t_{k+1}} \boldsymbol{f}(\boldsymbol{\xi}_k, t_k) + \frac{t - t_k}{t_{k+1} - t_k} \boldsymbol{f}(\boldsymbol{\xi}_{k+1}, t_{k+1})$$

  である.

▷ $t_{k+1} - t_k = h$ であることに注意すると,

$$\int_{t_{k+1}}^{t_{k+2}} \frac{t - t_{k+1}}{t_k - t_{k+1}} = -\frac{h}{2}$$

である.

▷ 一方,

$$\int_{t_{k+1}}^{t_{k+2}} \frac{t - t_k}{t_{k+1} - t_k} = \frac{3h}{2}$$

▷ よって, L=2 の場合の Adams-Bashforth 公式は,

$$\begin{aligned}&\boldsymbol{\xi}_{k+2} - \boldsymbol{\xi}_{k+1} \\ &= h\left(-\frac{1}{2}\boldsymbol{f}(\boldsymbol{\xi}_k, t_k) + \frac{3}{2}\boldsymbol{f}(\boldsymbol{\xi}_{k+1}, t_{k+1})\right).\end{aligned}$$

となる.

BDF 法

- BDF 法は Backward Difference Formula 法の略であり, は implicit な解法の一種である.
- BDF 法は硬い系向けの解法として広く使われている ([三井]).

- $\boldsymbol{\xi}_{k+L}$ を $(\boldsymbol{\xi}_k, \ldots, \boldsymbol{\xi}_{k+L-1})$ に関連した情報から構成するという多段法の問題設定に戻る.

- $\boldsymbol{\xi}_{k+L}$ を未定のままにしたまま，
  $(t_k, \boldsymbol{\xi}_k), (t_{k+1}, \boldsymbol{\xi}_{k+1}), \ldots, (t_{k+L}, \boldsymbol{\xi}_{k+L})$ を通る L 次の Lagrange 補間多項式を構成する．これを $\boldsymbol{p}(t)$ とする．

- ステップ幅が等間隔だったから, $t_{k+L-1} = t_{k+l} - h$, $t_{k+L-2} = t_{k+l} - 2h$, ... 等に注意する.

- $\boldsymbol{p}(t)$ が微分方程式の解の近似関数となるように $\boldsymbol{\xi}_{k+L}$ を決めたい.

- 決め方はひと通りというわけではないが, $\boldsymbol{p}(t)$ の導関数の $t = t_{k+L}$ における値が微分方程式

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, t)$$

の右辺と一致するように決めるというのはひとつの考え方である.

- $\left.\dfrac{d}{dt}\boldsymbol{p}(t)\right|_{t=t_{k+L}} = \boldsymbol{f}(\boldsymbol{\xi}_{k+L}, t)$ となるように $\boldsymbol{\xi}_{k+L}$ を定める解法を **BDF 法**という.

- Lagrange 補間多項式を微分し, $t = t_{k+L}$ における値を評価することは, 代数的にできる (ステップ幅が等間隔であることに注意).

- いくつかの L に対する BDF 法を列挙する ([三井]).

  ▷ $L = 1$:

$$-\boldsymbol{\xi}_k + \boldsymbol{\xi}_{k+1} = h\boldsymbol{f}(\boldsymbol{\xi}_{k+1}, t_{k+1})$$

▷ $L = 2$:

$$\frac{1}{2}\boldsymbol{\xi}_k - 2\boldsymbol{\xi}_{k+1} + \frac{3}{2}\boldsymbol{\xi}_{k+2} = h\boldsymbol{f}(\boldsymbol{\xi}_{k+2}, t_{k+2})$$

▷ $L = 3$:

$$-\frac{1}{3}\boldsymbol{\xi}_k + \frac{3}{2}\boldsymbol{\xi}_{k+1} - 3\boldsymbol{\xi}_{k+2} + \frac{11}{6}\boldsymbol{\xi}_{k+3} = h\boldsymbol{f}(\boldsymbol{\xi}_{k+3}, t_{k+3})$$

- BDF 法は $L > 6$ に対して不安定となることが知られている.

予測子修正子法

- 多段法において, explicit な公式で仮の解を求め, それを implicit な公式に代入してから修正する方法を, 予測子修正子法という. また, この方法で使われる explicit な公式を予測子, implicit な公式を修正子という.

- 色々な explicit な解法と implicit な解法 を組み合わせることで, 様々な 予測子修正子法が得られるが, 詳細は略す.

Scilab による実行例

硬くない系

- この例では, 微分方程式

$$\frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

を Scilab によって 数値的に解く. 初期値は $x_1(0) = 1, x_2(0) = 0$ とする.

- この微分方程式の厳密解は

$$x_1(t) = \cos t$$
$$x_2(t) = \sin t$$

である.

- この系は硬い系ではないので, BDF 法などを使う利点はないが, Scilab の ode で実装されている解法をいくつか使ってみる.

- Scilab の関数 `ode` は, 第一オプションの指定しだいで, ソルバを以下のように選択する.
  - ▷ 指定なし: 予測子-修正子法と BDF 法の自動切り換え
  - ▷ ”adams”: Adams 法
  - ▷ ”stiff”: BDF 法
  - ▷ ”rk”: 4 次の Runge-Kutta 法
  - ▷ ”rkf”: RKF45

- 関数定義のコードは以下の通り.

```
//関数定義（全解法共通）
function dx=f(t,x)
  dx=[0 -1;1 0]*x;
endfunction
t0=0;
//時間の刻みは問題に合わせて要調整.
t=0:.1:100;
x0=[1;0];
```

- ソルバを指定して解く部分は以下の通り．ただし, 注釈では 4 次の Runge-Kutta 法を RK 法と略している.

```
y=ode(x0,t0,t,f); //デフォルト
y=ode("adams",x0,t0,t,f); //Adams 法
y=ode("stiff",x0,t0,t,f); //BDF 法
y=ode("rk",x0,t0,t,f);    //RK 法
y=ode("rkf",x0,t0,t,f);   //RKF45 法
```

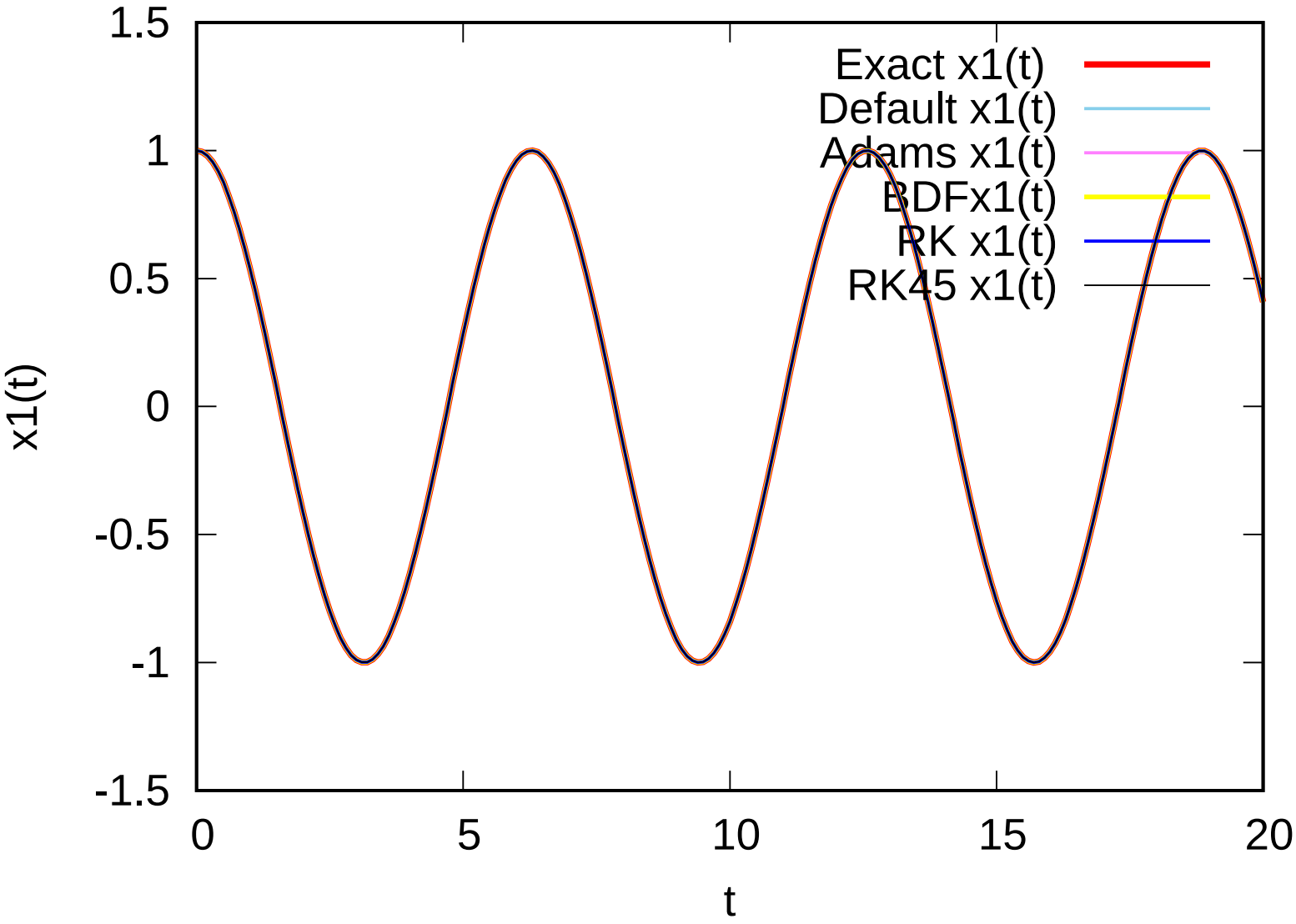
- この例では 5 通りのソルバを試しているが, 通常は, 自分が使いたいソルバの部分のみを実行する.

- 各解法の誤差 (数値解 $\boldsymbol{y}(t)$ (2 行 1001 列の行列) と対応する真の解の差 の Frobenius ノルム) を比較した結果は以下の通り.

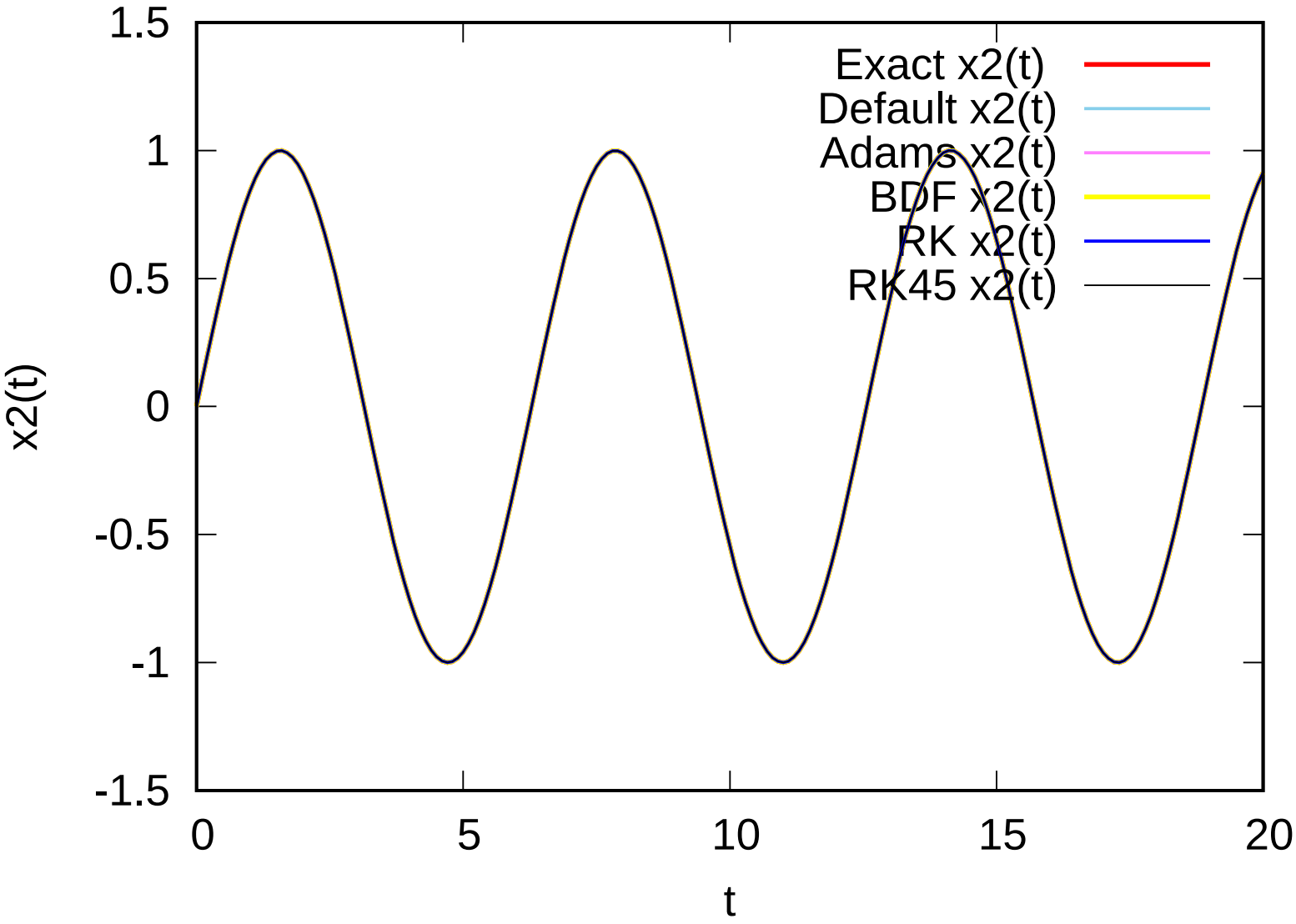
解法	計算時間	誤差
デフォルト	1.562500×10^{-2}	5.326591×10^{-5}
Adams	1.406250×10^{-1}	5.776883×10^{-5}
BDF 法	2.187500×10^{-1}	7.609895×10^{-4}
Runge-Kutta 法	5.625000×10^{-1}	4.228912×10^{-6}
RKF45	1.718750×10^{-1}	4.228912×10^{-6}

- 計算時間はどのソルバも同じオーダーであるが, Runge-Kutta 法のみ, 他の解法の数倍の時間がかかっている.
- 誤差については, 4 次の Runge-Kutta 法と RK45 が他のソルバより一桁小さくなっている.

- $x_1(t)$ の厳密解と数値解を比較したグラフを次ページに示す．どのソルバも誤差が非常に小さく，グラフがほぼ重なっている．


1.5
1
0.5
0
-0.5
-1
-1.5
x1(t)
0
5
10
15
20
t
Exact x1(t)
Default x1(t)
Adams x1(t)
BDFx1(t)
RK x1(t)
RK45 x1(t)

- $x_2(t)$ の厳密解と数値解を比較したグラフを次ページに示す．どのソルバも誤差が非常に小さく，グラフがほぼ重なっている．


1.5
1
0.5
0
-0.5
-1
-1.5
x2(t)
0
5
10
15
20
t
Exact x2(t)
Default x2(t)
Adams x2(t)
BDF x2(t)
RK x2(t)
RK45 x2(t)

硬い系

- 次ページの微分方程式は, 文献 [三井] に掲載されている硬い系である.

$$\frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -2 & 1 \\ 1998 & -1999 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} -\cos t \\ 1999\cos t - \sin t \end{pmatrix},$$

- 初期値を $x_1(0) = 1,\ x_2(0) = 2$ とする.

- この微分方程式の厳密解は,

$$x_1(t) = e^{-t}$$
$$x_2(t) = e^{-t} + \cos t$$

である.

- 関数定義のコードは以下の通り.

```
//関数定義（全解法共通）
function dx=f(t,x)
  dx=[-2 1;1998 -1999]*x+..
    [-cos(t);1999*cos(t)-sin(t)];
endfunction
t0=0;
t=0:.5:100;
x0=[1;2];
```

- ソルバを指定して解く部分は先と同様.

```
y=ode(x0,t0,t,f); //デフォルト
y=ode("adams",x0,t0,t,f); //Adams 法
y=ode("stiff",x0,t0,t,f); //BDF 法
y=ode("rk",x0,t0,t,f);    //RK 法
```

- RK45が省略されているのは, この例ではRK45は収束しないからである.

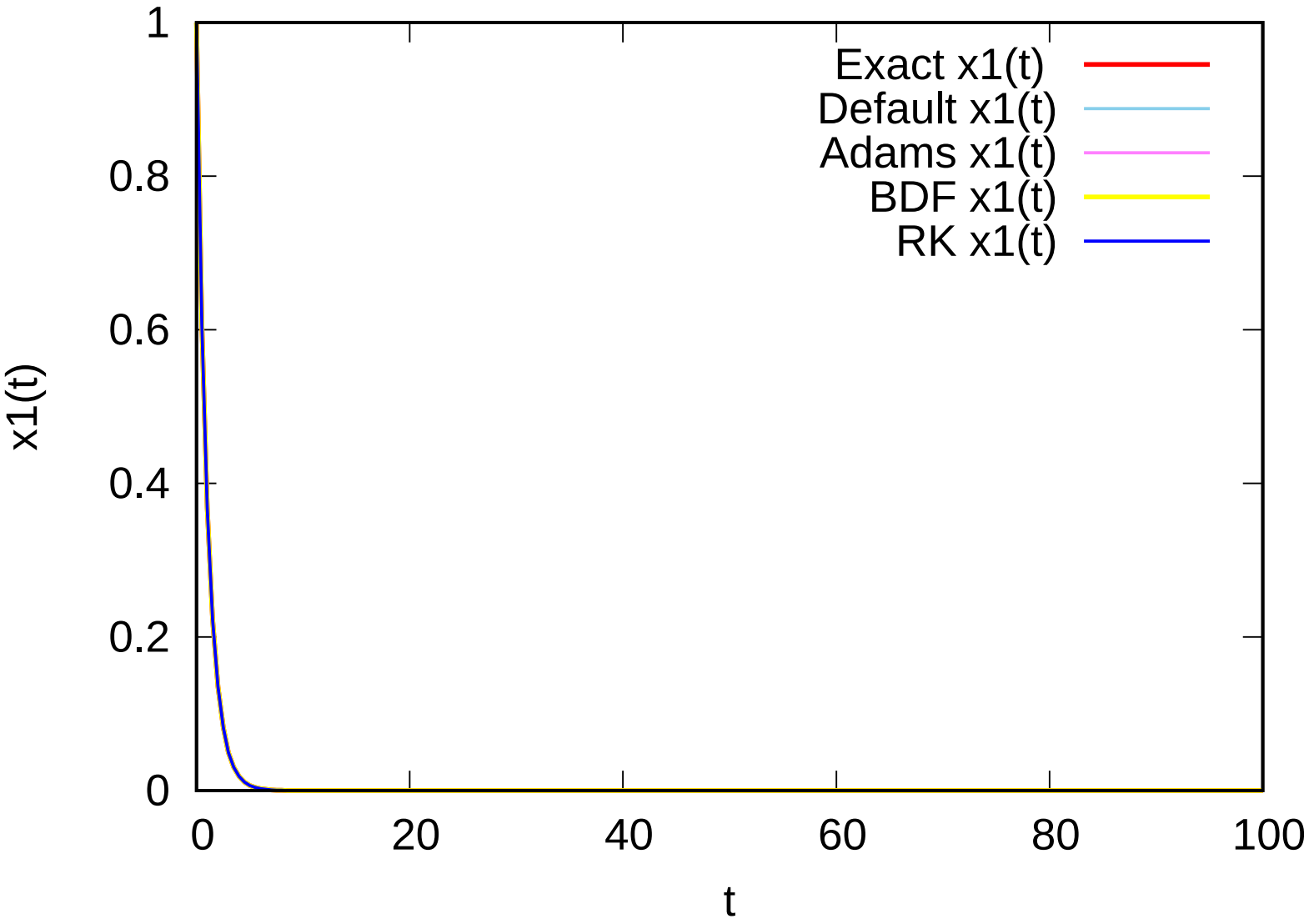
- 各解法の誤差 (数値解 $\boldsymbol{y}(t)$ (2 行 201 列の行列) と対応する真の解の差 の Frobenius ノルム) を比較した結果は以下の通り.

解法	計算時間	誤差
デフォルト	6.250000×10^{-2}	2.777452×10^{-7}
Adams	1.390625	2.089365×10^{-6}
BDF 法	3.750000×10^{-1}	3.907983×10^{-7}
Runge-Kutta 法	1.408438×10^{2}	1.495169×10^{1}

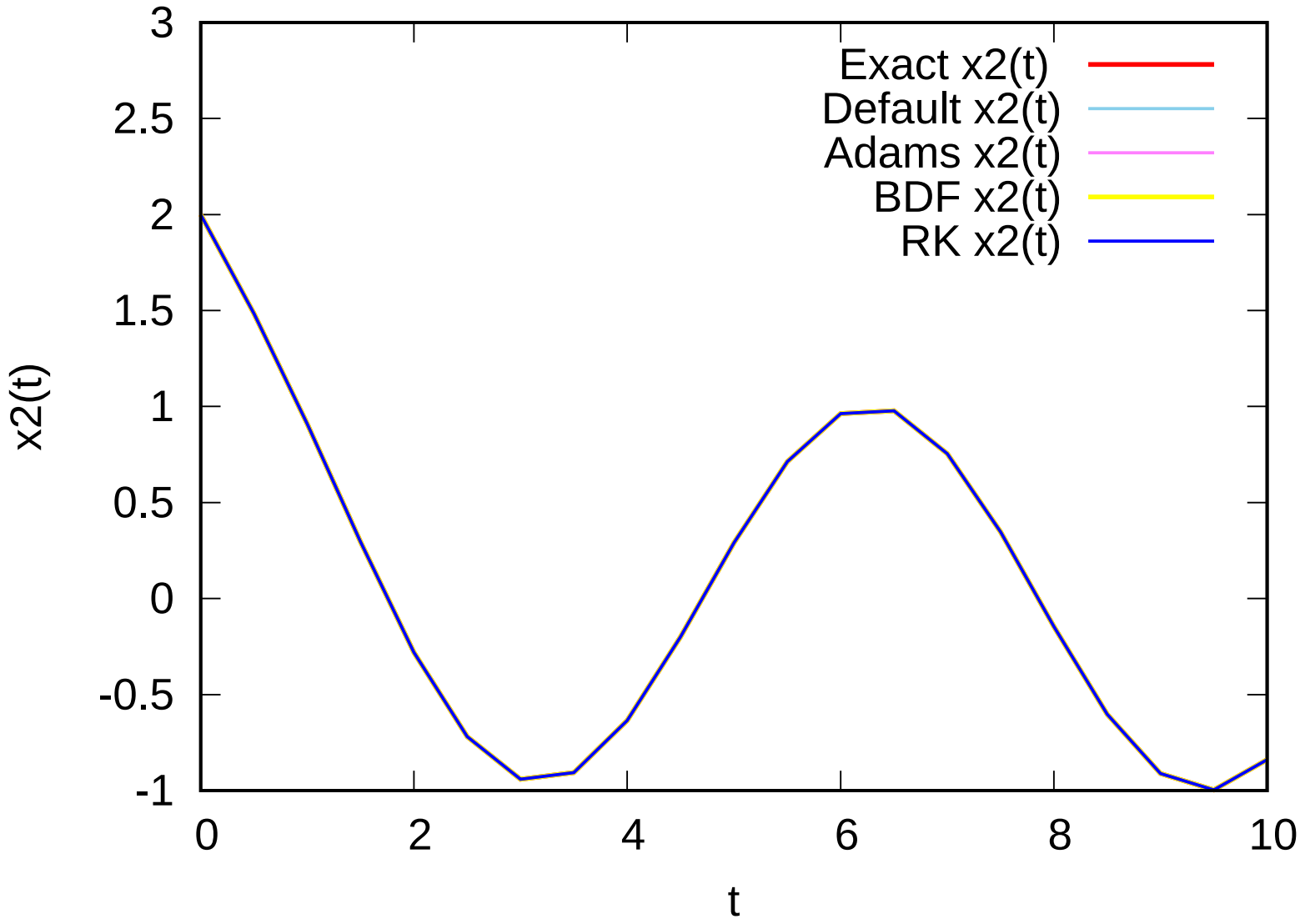
- 硬い系に対応しているのがデフォルトとBDF法であるので, これらはいずれも高速で, 精度もおおむね同等である.
- Adams法はデフォルトやBDF法と比較して, 誤差が1桁大きくなっている. 計算時間は数倍程度.

- Runge-Kutta 法ではデフォルト解法の 2000 倍程度の時間がかかっている．誤差は 5×10^7 倍程度になってしまっている．

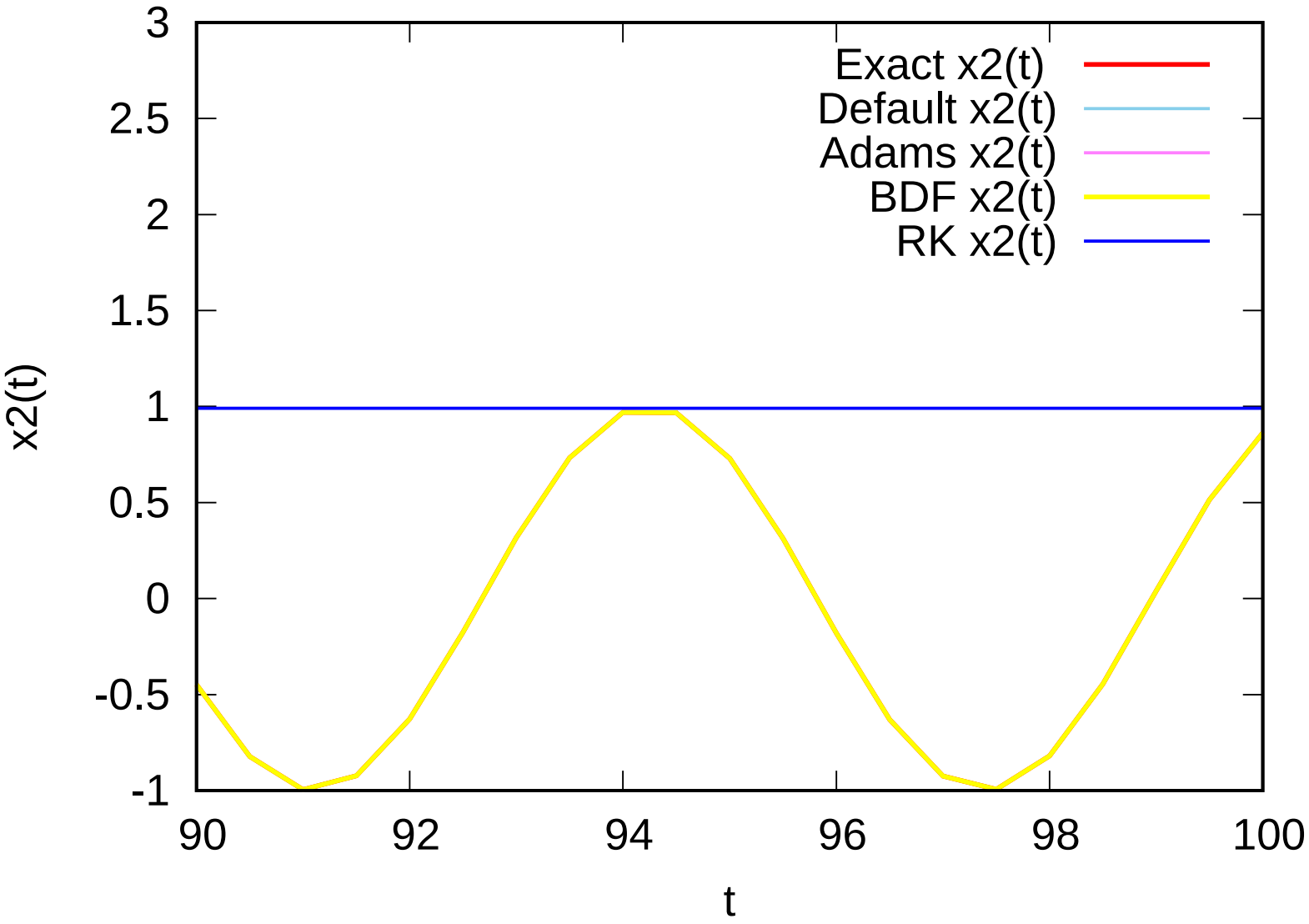
- $x_1(t)$ の厳密解と数値解を比較したグラフを次ページに示す. $x_1(t)$ については, どのソルバも大きな誤差を出しているわけではない.


Exact x1(t)
Default x1(t)
Adams x1(t)
BDF x1(t)
RK x1(t)
x1(t)
t
0
0.2
0.4
0.6
0.8
1
0
20
40
60
80
100

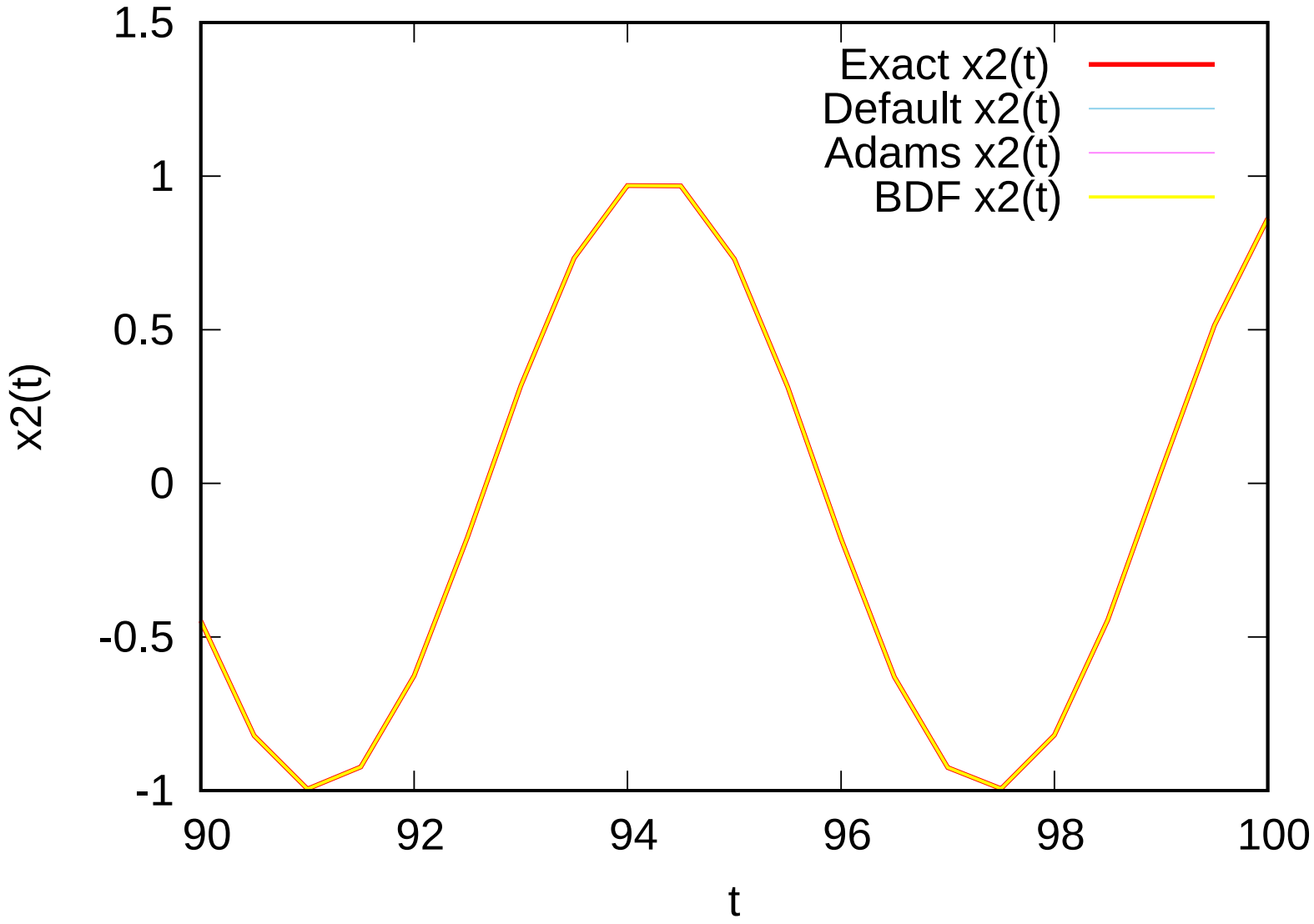
- $x_2(t)$ の厳密解と数値解を時刻 $[0, 10]$ で比較したグラフを次ページに示す．この時刻では，どの解法にも大きな誤差はない．


Exact x2(t)
Default x2(t)
Adams x2(t)
BDF x2(t)
RK x2(t)
x2(t)
t
3
2.5
2
1.5
1
0.5
0
-0.5
-1
0
2
4
6
8
10

- $x_2(t)$ の厳密解と数値解を時刻 $[9, 100]$ で比較したグラフを次ページに示す. この時刻には, 4 次の Runge-Kutta 法による数値解は, 厳密解とまったく異なる.


Exact x2(t)
Default x2(t)
Adams x2(t)
BDF x2(t)
RK x2(t)
x2(t)
t
3
2.5
2
1.5
1
0.5
0
-0.5
-1
90
92
94
96
98
100

- 4 次の Runge-Kutta 法を除いて $x_2(t)$ の厳密解と数値解を時刻 $[9, 100]$ で比較したグラフを次ページに示す. 厳密解とデフォルトソルバ, Adams 法, BDF 法の数値解はほぼ一致することがわかる.


Exact x2(t)
Default x2(t)
Adams x2(t)
BDF x2(t)
x2(t)
t
1.5
1
0.5
0
-0.5
-1
90
92
94
96
98
100