データサイエンス系科目群（理・博）：シミュレーション実習
第 3 回 講義資料

担当：川﨑猛史

名古屋大学大学院理学研究科理学専攻物理科学系・非平衡物理研究室 (R 研)

Last update: April 29, 2024

目次

目次

1. 講義のスケジュール

- 当実習は春 1 期にて実施する．
- 講義資料は各講義予定日当日朝 **11** 時迄にアップロードする．
- スケジュール
  1. 4/15：第 1 回
  2. 4/22：第 2 回
  3. **4/30**(火)：第 **3** 回
  4. 5/13：第 4 回 (中間レポート課題公開)
  5. 5/20：第 5 回
  6. 5/27：第 6 回 (中間レポート課題提出期限予定)
  7. 5/29：第 7 回
  8. **6/03**：休講
  9. 6/10：第 8 回 (期末レポート課題公開)
  10. 6/17：第 9 回　（補講）

1.1. シラバス

当実習では以下の内容を扱う予定である（進捗に合わせ変更する可能性がある）.

1. 導入
   - C(C++) の使い方 (主に数値計算)
   - Python の使い方 (データ解析と作図)
   - 数値計算の理念
   - 桁落ち
   - 科学計算における無次元化
2. 常微分方程式の数値解法：減衰振動や調和振動子を例に
   - 微分方程式の数値積分
   - 軌道の安定性と保存則
3. １粒子系のブラウン運動
   - ランジュバン方程式（確率微分方程式）
   - 正規乱数の生成法
   - オイラー・丸山法
   - 時間平均とアンサンブル平均
4. 多粒子系のブラウン運動
   - 相互作用力の計算方法
   - 非平衡系のシミュレーション：相分離現象を例に
5. 多粒子系の分子動力学シミュレーション
   - 位置ベルレ法と速度ベルレ法
   - 多粒子系における保存則
6. モンテカルロ法
   - 統計力学の復習
   - マルコフ連鎖モンテカルロ法
   - メトロポリス判定法

目次

2. 第 2 回自主課題解説

減衰運動の数値計算と解析計算の比較

1 次元中を運動する溶媒中の粒子の運動を考える．この粒子の運動方程式は

$$m\frac{\mathrm{d}v(t)}{\mathrm{d}t} = -\zeta v(t),$$

であり，時刻 $t = 0$ において速度 $10a\zeta/m$ の速さで運動しているとする. この時，以下の各問に答えよ．なお，無次元化に際し，長さの単位は a, 時間の単位は $t_0 = m/\zeta$ とおくこと．

(1) この微分運動方程式の解析解を求めよ．さらに問題文中にて指定した単位系を用いることで無次元化せよ．

(2) この粒子の速度の時間発展を数値的に計算し解析解と比較せよ．ただし運動方程式の離散化は簡単のため Euler 法を用いるものとし，離散化における時間分解能を $\Delta t/t_0 = 0.1, 0.01, 0.001$ と変えたものを比較してみよ．

[解説]

2. 第 2 回自主課題解説 (2)

(1) 微分方程式の理論解は

$$v(t) = 10\frac{a\zeta}{m}\exp(-t\zeta/m) \tag{1}$$

である．数値的に得られた解は無次元化されたものであるから，この理論解も無次元化してみよう．上式の両辺を $\frac{a\zeta}{m}$ で割ると，

$$v(t)\frac{m}{a\zeta} = 10\exp(-t\zeta/m) \tag{2}$$

を得る．いま，$\tilde{v}(t) = v(t)\frac{m}{a\zeta}$ かつ $\tilde{t} = t\zeta/m$ であるので，上式は

$$\boxed{\tilde{v}(t) = 10\exp(-\tilde{t})} \tag{3}$$

となり無次元化された．つまりこの解と数値計算の解を比較すればよい．

2. 第 2 回自主課題解説 (3)

(2) 数値計算の主計算プログラムをリスト 1 に載せた”damping.cpp”．以下の GitHub リポジトリより取得可能．
[リンク]

- (ポイント) 前回学んだ計算機イプシロンの値より速度の値が小さくなった際に 0 とみなし終了するようにした．
- (ポイント) 関数（サブルーチン）に主な計算を押し込むことにより main 文は 4 行に抑えた． 変数を関数内で初期化できるので便利．

- 図示の python プログラムをリスト 2 ”velo_plot.py”にまとめた．先の GitHub リポジトリより取得可能．
  - 図示をする際に，for 文や if 文を駆使することで，前回より複数のグラフを並べる際にスリムなコードになった．

2. 第 2 回自主課題解説 (4)

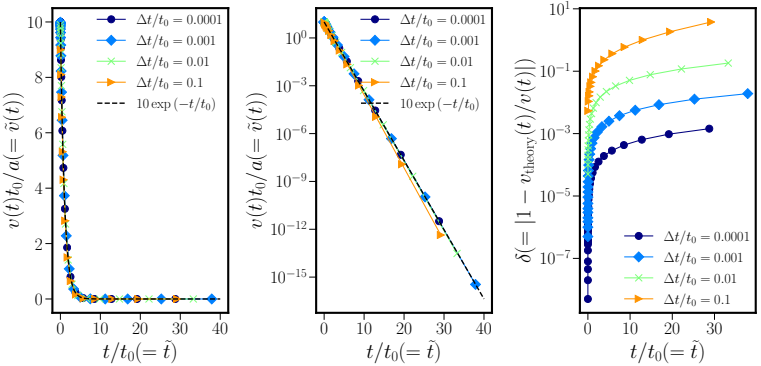

図 1: 時間刻み（時間分解能）を変化させた際の，(左) 無次元化速度の時間発展 (線形図)，(中) 無次元化速度の時間発展（片対数図），(右)　速度の誤差の時間発展．

2. 第 2 回自主課題解説 (5)

- 以下，C(C++) を用いた計算プログラムである．

リスト 1: 第 2 回自主課題　主計算用サンプルプログラム “damping.cpp”. 以下の GitHub リポジトリより取得可能．　[リンク]

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <math.h> // for mathmatical functions
4  #include <iostream> // for input or output, i.e., std::cout
5  #include <fstream> // for  std::ofstream file
6  #include <cfloat> // for DBL_EPSILON, a.k.a. numerical epsilon
7  //using namespace std;
8
9  int damp(double dt){
10   char filename[128];
11   std::ofstream file;
12   int i=0;
13   double v=10.,out=1.;
14   sprintf(filename,"velo_%.4f.dat",dt);
15   file.open(filename);
16   while(v > DBL_EPSILON){ //continue while v > 2.22044604925031e-16.
17     v-=v*dt;  // as same as v = v - v*dt
18     i++;      // as same as i += 1 or i =i + 1;
19     if((double)i >= out){
```

2. 第 2 回自主課題解説 (6)

```
20        file << (double)i*dt << "    " << v <<std::endl;
21        std::cout << (double)i*dt << "    " << v <<std::endl;
22        out*=1.5;  // as same as out = out * 1.5
23      }
24    }
25    file.close();  // should be closed when all input process has been finised.
26    return 0;
27  }
28
29  int main(){
30    double dt;
31    for(dt=1.e-4;dt<=1.e-1;dt*=10.)
32      damp(dt);
33    return 0;
34  }
```

2. 第 2 回自主課題解説 (7)

- 以下，図示の際のサンプルプログラムである．

リスト 2: Python サンプルプログラム “velo_plot.py” GitHub リポジトリより取得可能．[リンク] jupyter notebook ファイル “_jupyter.ipynb”にも同等のプログラムを掲載．

```
1  import matplotlib
2  import matplotlib.pyplot as plt
3  %matplotlib inline
4  %config InlineBackend.figure_format = 'retina'
5  import matplotlib.cm as cm  # colormap
6  import numpy as np
7  #plt.rcParams["text.usetex"] =True
8  plt.rcParams['font.family'] = 'Arial' 使用するフォント名#
9  plt.rcParams["font.size"] = 25
10
11 fig = plt.figure(figsize=(18,8))
12
13 dt=[0.0001,0.0010,0.0100,0.1000]
14 symbol =['o-','D-','x-','>-']
15
16 for j in range (1,4):
17     #ax = fig.add_subplot("13{}".format(j))
18     ax = fig.add_subplot(1,3,j)
19     if(j>1):
```

2. 第 2 回自主課題解説 (8)

```
20          plt.yscale('log')
21
22      for i in range (0,4):
23          print(i,symbol[i],dt[i])  # for check on the arrays
24          time,vel= np.loadtxt("./Lecture3/velo_{:.4f}.dat".format(dt[i]), comments='#',
                unpack=True)
25          if(j!=3):
26              plt.plot(time,vel, "{}".format(symbol[i]) ,markersize=10,color=cm.jet(i/4),
                    label=r"$\Delta t/t_0={}$".format(dt[i]))
27          else:
28              plt.plot(time,np.abs(1 -10.*np.exp(-time)/vel), "{}".format(symbol[i]) ,
                    markersize=10,color=cm.jet(i/4),label=r"$\Delta t/t_0={}$".format(dt[i]))
29
30
31      ###Drawing a line ######
32      if(j<3):
33          time= np.linspace(1e-4, 4e1, 1000)
34          vel= 10.*np.exp(-time)
35          plt.plot(time,vel, "--",markersize=3,linewidth = 2.0, color="k",label=r"$10\exp{(-
                t/t_0)}$")
36      #########図の書式設定
37      #
38      plt.tick_params(which='major',width = 1, length = 10)
39      plt.tick_params(which='minor',width = 1, length = 5)
```

2. 第 2 回自主課題解説 (9)

```
40      ax.spines['top'].set_linewidth(3)
41      ax.spines['bottom'].set_linewidth(3)
42      ax.spines['left'].set_linewidth(3)
43      ax.spines['right'].set_linewidth(3)
44      plt.xlabel(r"$t/t_0(=\tilde{t})$",color='k', size=30)
45      if(j!=3)  :
46          plt.ylabel(r"$v(t)t_0/a(=\tilde{v}(t))$",color='k', size=30)
47      else:
48          plt.ylabel(r"$\delta(=|1-v_{\rm theory}(t)/v(t)|)$",color='k', size=30)
49      if(j<3):
50          plt.legend(ncol=1, loc=1, borderaxespad=0, fontsize=20,frameon=False)
51      else:
52          plt.legend(ncol=1, loc=4, borderaxespad=0, fontsize=20,frameon=False)
53
54  ################################
55  #図のマージン設定
56  plt.subplots_adjust(wspace=0.5, hspace=0.25)
57  #各自ファイルのパスを変えること.
58  plt.savefig('./Lecture3/velo_dt_lin_log.png')
59  plt.savefig('./Lecture3/velo_dt_lin_log.pdf')
60  plt.show()
```

目次

3. 第 3 回課題・自主課題：減衰振動，過減衰，臨界減衰の数値計算

第 3 回課題

一端を固定した，ばね定数 k のばねに，質点とみなせる質量 m の小球をつけ，滑らかな水平面上を 1 次元運動させる．小球にはさらに，粘性抵抗係数を ζ とする粘性抵抗力が働くものとする．この時，適当な座標系を敷き，時刻 t における小球の位置を $x(t)$，速度を $v(t)$ とすれば，物体の運動方程式は，

$$m\frac{\mathrm{d}v(t)}{\mathrm{d}t} = -\zeta v(t) - kx(t) \tag{4}$$

と表される．いま正値パラメータ (m，ζ，k) を変化させ，これらの大小関係から質点の運動が大きく変化する様子を考察する．

[設問]

(1) 小球の運動が減衰振動（不足減衰），過減衰，臨界減衰を示す条件をそれぞれ，m，ζ，k を用いて解析的に表せ．

(2) $\tilde{x}$，$\tilde{t}$ を長さと時間に関する無次元の変数であるとし，問題にで与えた実際の物理量との関係は $x = a\tilde{x}$，$t = t_0\tilde{t}$，$\dot{x} = v = (a/t_0)\tilde{v}$ であるとする．ここで a[m]，t_0[s] は適当に定めた長さと時間である．一方，運動方程式で与えた物理量を組み合わせていくつかの物理的に意味のある時間スケールを抽出することができる．ここでは，減衰運動の時間スケール $t_d = \frac{m}{\zeta}$，ばねの振動を特徴付ける時間スケール $t_s = \sqrt{m/k}$ が挙げられる．いまこれらを用いて運動方程式を無次元化し，さらに時間刻み $\Delta\tilde{t}$ に関する半陰的 Euler 法 (Semi-implicit Euler method) で離散化することで 2 項間漸化式を，$\tilde{v}(\tilde{t} + \Delta\tilde{t})$，$\tilde{x}(\tilde{t} + \Delta\tilde{t})$ に対してそれぞれ立てよ．

(3) いま，時間の単位を $t_0 = t_s$ と定めれば，問題 **(2)** で求めた漸化式における係数が 1 つ落ち，$\boxed{t_s/t_d}$ がここでの運動の特徴を制御する唯一のパラメータになることがわかる．そこで，問題 **(1)** において見積もった条件から，$\boxed{t_s/t_d}$ に適当な値を入れることで，減衰振動（不足減衰），過減衰，臨界減衰を，問題 **(2)** までに議論した漸化式を数値的に解くことで再現し，$x(t)$ のグラフにプロットせよ．なお，初期条件は $x(0) = a$，$\dot{x}(0) = 0$ と与えよ．計算する時間は $t > 0$ で，運動の特徴が見られる範囲で適当に定めてよい．時間刻み $\Delta\tilde{t}$ の大きさについても数値解が収束する範囲で適当に定めてよいことにする．

3. 第 3 回課題・自主課題：減衰振動，過減衰，臨界減衰の数値計算 (2)

[部分解説]

(1) 解として $x(t) = Ae^{\lambda t}$ を代入すると，λ は以下の特性方程式を満たす必要がある．

$$m\lambda^2 + \zeta\lambda + k = 0 \tag{5}$$

この 2 次方程式の判別式 $D = \zeta^2 - 4mk$ より，

- 過減衰: $\zeta^2 - 4mk > 0$
- 臨界減衰: $\zeta^2 - 4mk = 0$
- 不足減衰: $\zeta^2 - 4mk < 0$

(2) 運動方程式を以下のように無次元化する

$$m\frac{a}{t_0^2}\dot{\tilde{v}}(t + \Delta t) = -\zeta\frac{a}{t_0}\zeta\tilde{v}(t) - ka\tilde{x}(t) \tag{6}$$

両辺を $m\frac{a}{t_0^2}$ で割れば，無次元方程式となり

$$\dot{\tilde{v}}(t) = -\frac{\zeta t_0}{m}\tilde{v}(t) - \frac{kt_0^2}{m}\tilde{x}(t) \tag{7}$$

3. 第 3 回課題・自主課題：減衰振動，過減衰，臨界減衰の数値計算 (3)

を得る．これより右辺第１項と第２項から時間スケール $t_d = \frac{m}{\zeta}$ と $t_s = \sqrt{\frac{m}{k}}$ を見出すことができる．これらを用いれば，運動方程式は

$$\dot{\tilde{v}}(\tilde{t}) = -\frac{t_0}{t_d}\tilde{v}(\tilde{t}) - \frac{t_0^2}{t_s^2}\tilde{x}(\tilde{t}) \tag{8}$$

と表すことができる．これらを，以下で説明する半陰 Euler 法 (Semi-implicit Euler method) で離散化すると，$\tilde{v}(\tilde{t}+\Delta\tilde{t})$，$\tilde{x}(\tilde{t}+\Delta\tilde{t})$ に関する漸化式は，

$$\tilde{v}(\tilde{t}+\Delta\tilde{t}) = \tilde{v}(\tilde{t}) - \frac{t_0}{t_d}\tilde{v}(\tilde{t})\Delta\tilde{t} - \frac{t_0^2}{t_s^2}\tilde{x}(t)\Delta\tilde{t} \tag{9}$$

$$\tilde{x}(\tilde{t}+\Delta\tilde{t}) = \tilde{x}(\tilde{t}) + \boxed{\tilde{v}(\tilde{t}+\Delta\tilde{t})\Delta\tilde{t}} \tag{10}$$

となる．さらに，時間スケールを $t_0 = t_s$ と選べば，$\tilde{v}(\tilde{t}+\Delta\tilde{t})$ に関する漸化式は

$$\tilde{v}(\tilde{t}+\Delta\tilde{t}) = \tilde{v}(\tilde{t}) - (t_s^2/t_d^2)\tilde{v}(\tilde{t})\Delta\tilde{t} - \tilde{x}(t)\Delta\tilde{t} \tag{11}$$

となる．つまり，t_s/t_d が本系における $\boxed{\text{パラメータ}}$ となる．

3. 第 3 回課題・自主課題：減衰振動，過減衰，臨界減衰の数値計算 (4)

半陰 Euler 法 (Semi-implicit Euler method)

- 陽的 Euler 法と陰的 Euler 法を連立させる方法．
- 調和振動子系ではシンプレクティック（$t \to t + \Delta t$ の時間前進軌道と $t + \Delta t \to t$ の時間後退軌道が同一になる）になることが知られている．
- 調和振動子系での適用例 (v は陽解法，x は陰解法)：

$$
\begin{aligned}
v(t + \Delta t) &= v(t) - \omega^2 x(t)\Delta t && (12)\\
x(t + \Delta t) &= x(t) + v(t + \Delta t)\Delta t \\
&= x(t) + v(t)\Delta t - \omega^2 x(t)(\Delta t)^2 && (13)
\end{aligned}
$$

- これを位相空間に関する時間発展行列で書くと

$$
\begin{pmatrix} v(t + \Delta t) \\ x(t + \Delta t) \end{pmatrix} = \begin{pmatrix} 1 & -\omega^2 \Delta t \\ \Delta t & 1 - \omega^2 (\Delta t)^2 \end{pmatrix} \begin{pmatrix} v(t) \\ x(t) \end{pmatrix} = \mathbf{A}(t + \Delta t | t) \begin{pmatrix} v(t) \\ x(t) \end{pmatrix} \quad (14)
$$

- 一方，時間後退方向に関して同様の式から得られる $\mathbf{A}(t|t + \Delta t)$ は $\mathbf{A}(t + \Delta t|t)$ の逆行列となる．このような性質をシンプレクティック (symplectic) といい，軌道の収束性がよくなる．
- シンプレクティックな場合 $\det \mathbf{A}(t|t + \Delta t) = 1$(位相空間が膨張・収縮しない).
- 陽解法のみを用いた場合はシンプレクティックにならず軌道が閉じなくなる．

(3) 自主課題: 次週解説．

目次

4.1. オイラー法の収束性の評価

- 減衰型の微分方程式

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \lambda x \tag{15}$$

は，オイラー法で離散化すると

$$x_{n+1} = (1 + \lambda\Delta t)x_n \tag{16}$$

と書ける．この時，数列の収束条件は

$$|x_{n+1}/x_n| = |1 + \lambda\Delta t| < 1 \tag{17}$$

であるから，

$$\Delta t < -1/\lambda \tag{18}$$

を満たす Δt を選べば数値計算は安定に回ることがわかる．

- ここでは $\lambda = -1$ なら $\Delta t < 1$ で安定．
- しかしそれだけでは十分でなく Δt を変化させた時の誤差の程度も議論するのが一般的である．
- 収束値からのずれはオイラー法の場合 $O(\Delta t)$ で系統的に変化する．
- 見たい現象がどの程度誤差に寛容であるかに応じて時間刻み Δt を変化させる必要がある．

目次

5.1. 端末上でよく使うコマンド集（Unix 系および Windows）

- Unix 環境よく使うコマンド:

Unix 環境よく使うコマンド

pwd　現在ディレクトリのフルパス表示
cd (dir)　ディレクトリ移動
cd ~　ホームディレクトリへ移動
mkdir (dir)　新規ディレクトリを作成
rm -r (dir)　指定ディレクトリを中のファイルごと削除
cp -r (dirA) (dirB)　ディレクトリのコピー
mv (file) (dir)　ファイルをディレクトリへ移動
ls　現在ディレクトリのファイル一覧表示
cat (file)　ファイルの内容表示
cat (file1) (file2) > (file3)　file1 と file2 を連結させて file3 を生成
rm (file)　ファイルを削除
cp (fileA) (fileB)　ファイルコピー
mv (fileA) (fileB)　ファイル名変更
touch (file)　空のファイルを新たに作成 (本来はタイムスタンプを押す)
find (dir)　指定ディレクトリ以下のファイルを列挙
more (file)　ファイルの内容表示（ページごとに止まる）
diff (fileA) (fileB)　ファイル間の差分（変更点）を表示
history　コマンドの履歴リストを表示