空気中の放物運動について

著者名　学生番号

平成 24 年 1 月 14 日

概 要

本レポートでは、数値計算を用いて、空気中のピンポン球の示す放物運動に関する考察を行った。空気抵抗を受ける物体の運動を記述する運動方程式を導出し、オイラー法による微分方程式の数値解法を C 言語を用いて実装した。さらに、このプログラムを実行することで、斜方投射の飛距離が初期速度にどのように依存するかを計算した。その結果、空気抵抗の影響で、初期の速さが大きいほど、最大飛距離をあたえる投射角が小さくなることが分かった。これらの結果は解析的には求められない数値計算によって初めて得られる結果である。

1　空気からの抵抗をうける物体の運動

空気中で運動する半径 R の球体は、物体の速さに比例する粘性抗力と、速さの二乗に比例する慣性抗力から力を受ける。空気の動粘性係数 $\eta = 1.8 \times 10^{-5}$ N sec/m ^{2} および空気密度 $\rho = 1.2\mathrm{kg/m}^3$ を用いて

$$F_{\text{粘性}} = 6\pi\eta R v, \quad F_{\text{慣性}} = \frac{C_d \pi \rho R^2}{2} v^2 \tag{1}$$

と表されることが知られている [1]。C_d は空気抵抗係数とよばれ、球状の物体の場合 $C_d = 0.3$ である。

ピンポン球の質量 m は 2.7g で半径 R は 20 mm である。慣性抗力と粘性抗力との比はレイノルズ数 Re と呼ばれ、運動の速さが 10 m/s の場合の Re の値は、おおよそ

$$Re \sim \frac{\rho (Rv)^2}{\eta R v} \sim 10^4 \tag{2}$$

となる。つまり、慣性抗力 $\gg$ 粘性抗力である。これより、本レポートでは空気抵抗として慣性抗力だけが働くとして、ピンポン球の運動を考察す

る。運動方程式は以下の通りである：

$$m\frac{d^2x}{dt^2} = -bv\frac{dx}{dt} \tag{3}$$

$$m\frac{d^2y}{dt^2} = -mg - bv\frac{dy}{dt} \tag{4}$$

ここで g は重力加速度 9.8 m/s^2、v は速さ $\sqrt{(dx/dt)^2 + (dy/dt)^2}$、$b$ は慣性抗力の係数 $C_d\pi\rho R^2/2 = 0.000226195$ kg/m を表す。

2　オイラー法による数値解

運動方程式をオイラー法で解くために、まず次のように速度 (v_x, v_y) を用いて 1 階の常微分方程式に書き換える：

$$\frac{dx}{dt} = v_x,\ \frac{dy}{dt} = v_y,\ \frac{dv_x}{dt} = -\frac{b}{m}vv_x,\ \frac{dv_y}{dt} = -g - \frac{b}{m}vv_y. \tag{5}$$

オイラー法は時刻 $t_n = ndt$ の x, y, v_x, v_y の値 $x_n, y_n, v_{x,n}, v_{y,n}$ を

$$x_n = x_{n-1} + v_{x,n-1}dt, \tag{6}$$

$$y_n = y_{n-1} + v_{y,n-1}dt, \tag{7}$$

$$v_{x,n} = v_{x,n-1} - \frac{b}{m}v_{n-1}v_{x,n-1}dt, \tag{8}$$

$$v_{y,n} = v_{y,n-1} - (g + \frac{b}{m}v_{n-1}v_{y,n-1})dt \tag{9}$$

から決定する 1 次の数値解法である。ここで、

$$v_{n-1} = \sqrt{(v_{x,n-1})^2 + (v_{y,n-1})^2}$$

である。

初期条件として、物体は初期時刻 $t = 0$ s で原点から v_0 m/s の速さで x 軸からの角度が $\theta°$ の方向に投げ出される場合を考える。このとき、

$$x_0 = 0,\ y_0 = 0,\ v_{x,0} = v_0\cos\theta,\ v_{y,0} = v_0\sin\theta \tag{10}$$

である。

C 言語を用いたプログラミングを行い、この初期条件からの物体の運動をオイラー法により計算した。プログラムのソースコードは以下の通りである：

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define  PI 3.141592
```

```
int main(int argc,char *argv[]){
  double dt,t1;
  double v0,th;
  /* コマンド引数（文字列）を数値に変換して代入 */
  v0=atof(argv[1]); /* 1 番目の（初速度）を v0 に代入 */
  th=atof(argv[2]); /* 2 番目の（角度）を th に代入 */
  t1=atof(argv[3]); /* 3 番目の（終了時刻）を t1 に代入 */
  dt=atof(argv[4]); /* 4 番目の（時刻の刻み）を dt に代入 */

  double g(9.8); /* 重力加速度 */
  double R(2.0e-2); /* ピンポン球の半径 */
  double m(2.7e-3); /* ピンポン球の質量 */
  double Cd(0.3); /* 球の Cd 値 */
  double b(0.5*Cd*PI*1.2*R*R); /* 慣性抗力の係数 */
  double bm(b/m); /* 慣性抗力の係数/m */

  double x,y,xa,ya;
  double v,u,va,ua; /* u, v はそれぞれ x, y の速度を表す変数 */
  double v_abs;     /* 速さ */
  x=0.0;            /* 初期位置と初期速度の設定 */
  y=0.0;
  v=v0*cos(PI/180*th);
  u=v0*sin(PI/180*th);

  double t;
  for(t=0.0; t<=t1; t+=dt){
    printf("%f %f %f\n",t,x,y);
    v_abs=sqrt(v*v+u*u);
    x+=v*dt;
    y+=u*dt;
    v-=bm*v_abs*v*dt;
    u-=g*dt+bm*v_abs*u*dt;
  }
  return 0;
}
```

このプログラムは、4 つのコマンドライン引数をもつ。第 1 の引数は物体の投げ出される初速度 v_0、第 2 の引数は投げ出される角度 θ、第 3 の引数は時間発展を計算する終了時刻 t、第 4 の引数は時間の刻み幅 dt である。これらの物理量は SI 単位系の数値でプログラムに与える。

このプログラムを用いて、$v = 10$ m/s, $\theta = 45°$, $t = 1.5$ s の条件下で、$dt = 2^{-3}, 2^{-6}, 2^{-9}$ s の 3 つの時間刻みの場合について運動する物体の運動を計算した。次のグラフは、dt ごとの座標値 x, y のデータからその軌跡を描いたものである。このグラフから、dt を小さくすると同じ曲線に収束していくことが分かる。

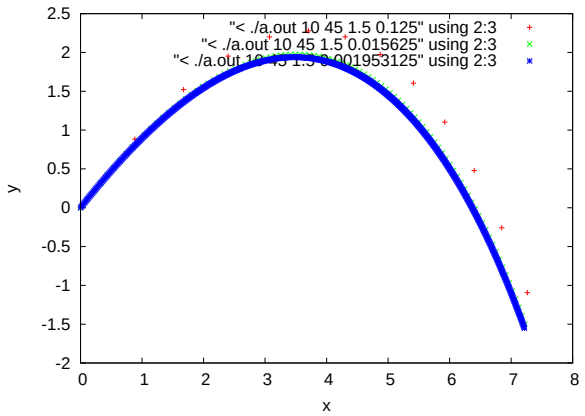

図 1: 初速度 $v = 10$ m/s、$\theta = 45°$ で打ち出されたピンポン球の運動の軌跡（$t \leq 1.5$ s）。時間刻みは $dt = 0.125, 0.0015625, 0.0001953125$ s。

3　数値解の精度

ここでは、数値解の精度について考察する。1 次解法のオイラー法を採用したので時刻 t での数値解の誤差 Δ は、C をある定数として

$$\Delta \equiv \| \text{時間刻み } dt \text{ での数値解} - \text{真の解} \| \sim Cdt \tag{11}$$

なる dt 依存性を持つ。ここで、$\|(x, y)\| = \sqrt{x^2 + y^2}$ である。実際には、式 (6)–(9) の微分方程式は解析的に解を求めることが不可能なので、時間刻みが最小の数値解を真の解の代わりに用いて式 (11) を評価した。

次の図は Δ を dt の関数として示したグラフである。このグラフより

$$\log \Delta \sim \log 4 + \log dt \tag{12}$$

の関係が成立していることが分かる。つまり、オイラー法の収束性 $\Delta \propto dt$ が実際の計算で保たれていることが確かめられた。

4　最大飛距離について

ここでは、最大飛距離を数値的に調べる。最大飛距離とは、原点 $(x, y) = (0, 0)$ から投げ出されたピンポン球が再び $y = 0$ に戻ってきたときの x 座標のことである。空気抵抗のない真空中では、$45°$ で最大飛距離となる。空気抵抗を考慮に入れると、最大飛距離となる角度は、$45°$ とは異なる値になることが予想される。そこで、本章では、初速度 $v_0 = 1$ m/s, 10 m/s の場合に最大飛距離となる角度を調べる。運動の数値計算には、2 章で説明したプログラムを用いた。

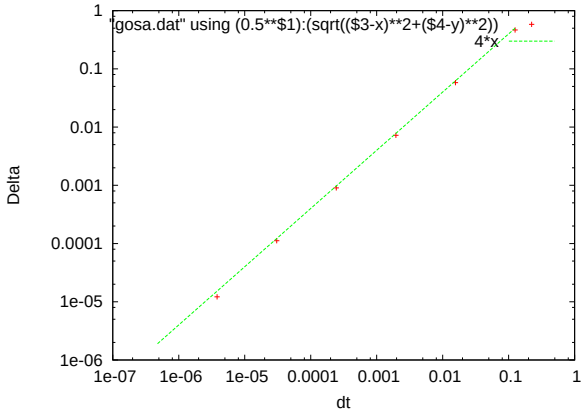

図 2: $v_0 = 10$ m/s、$\theta = 45°$、終了時刻 $= 1.5$ s の場合の誤差 Δ を dt の関数としてプロットした。式 (12) を確かめるため直線 $y = 6x$ も重ねてプロットした。時間の刻み dt として、$0.125(= 2^{-3})$, $0.0015625(= 2^{-6})$, $0.0001953125(= 2^{-9})$, $0.000244140625(= 2^{-12})$, $0.000030517578125(= 2^{-15})$, $0.000003814697265625(= 2^{-18})$, $0.000000476837158203125(= 2^{-21})$ の場合を計算した。真の値を最も高精度の $dt = 2^{-21}$ の計算結果で代用した。

まず、初期速度が $v_0 = 1$、角度 $\theta = 44, 45, 46$ の場合の運動の軌跡を図 3 に示した。このグラフより、$\theta = 45°$ のときに、真空中の最大飛距離

$$x = \frac{v_0^2}{g} \simeq 0.1 \tag{13}$$

と同程度の所に到達していることが分かる。

つぎに、初期速度が $v_0 = 10$、角度 $\theta = 40, 41, 45, 46$ の場合の運動の軌跡を図 4 に示した。この左図より、真空中の放物運動と異なり、運動の軌跡が頂点に関して左右非対称であることがわかる。これは空気抵抗により失速することが原因で v_x の値が小さくなることが原因と考えられる。さらに、図 4 右図からは、最大飛距離はおおよそ $\theta = 41°$ のときに 6.4 m であることがわかる。この値は真空中の最大飛距離 $10^2/9.8 \sim 10$ m の 6 割の長さに相当している。

以上の計算より、初速度 v_0 の小さな場合には、最大飛距離となる角度はほぼ 45° で飛距離も真空中の結果と等しいが、$v_0 \gtrsim 10$ の比較的速い運動ではこの値から有意にずれた角度 $\theta < 45°$ で、大幅に割り引かれた最大飛距離となることが分かった。

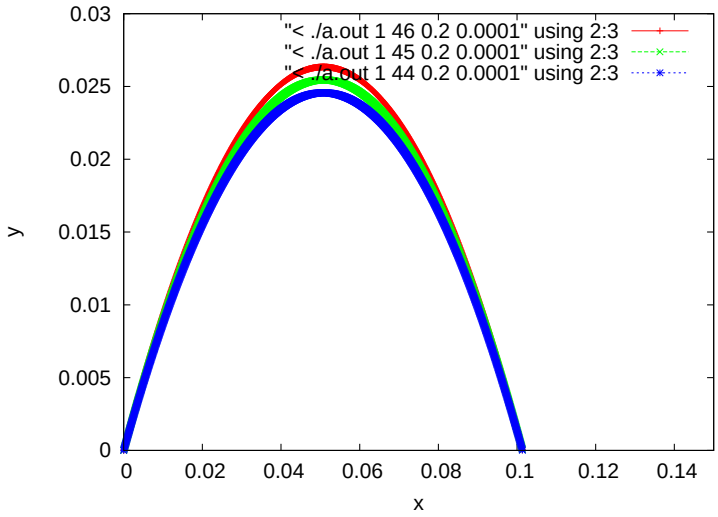

図 3: $v_0 = 1$, 角度 $\theta = 42, 43, 44, 45, 46, 47°$ の場合の運動の軌跡。運動の数値計算にはオイラー法を用いた。時間の刻みは $dt = 0.0001$ で計算した。

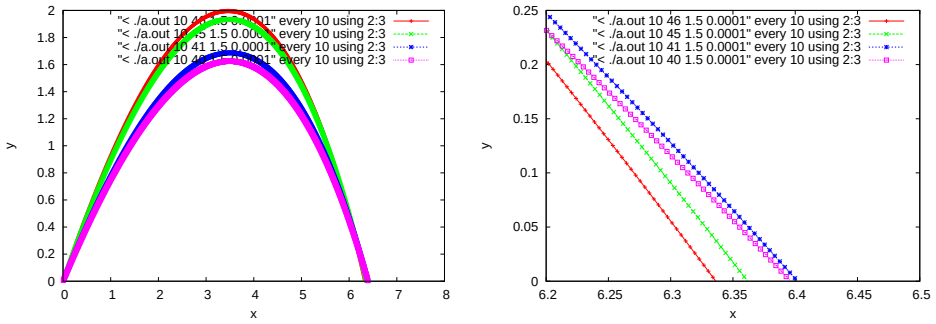

図 4: $v_0 = 10$, 角度 $\theta = 42, 43, 44, 45, 46, 47°$ の場合の運動の軌跡。運動の数値計算にはオイラー法を用いた。時間の刻みは $dt = 0.0001$ で計算した。

5　飛距離を求めるプログラム

前章では gnuplot のグラフから飛距離を読み取ることで、飛距離と初期速度との関係を定性的に理解することができた。ここでは、より精度の高い分析をするため、飛距離を高精度で計算する C 言語プログラムを作成した。

オイラー法の時間発展において、時刻 t_{n-1} において $y_{n-1} > 0$ であって、時刻 t_n で初めて $y_n \leq 0$ となったとする。このとき、(x_{n-1}, y_{n-1}) と

(x_n, y_n) を結ぶ直線は、

$$x = \frac{|y_{n-1}|x_n + |y_n|x_{n-1}}{|y_{n-1}| + |y_n|} \tag{14}$$

で、x 軸をよこぎる。時間刻み dt が十分に小さいときには、式 (14) は飛距離のよい近似になっている。

式 (14) を計算するC言語プログラムのソースコードは以下の通りである:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define  PI 3.141592

int main(int argc,char *argv[]){
  double dt,t1;
  double v0,th;
  v0=atof(argv[1]);
  th=atof(argv[2]);
  t1=atof(argv[3]);
  dt=atof(argv[4]);

  double g(9.8); /* 重力加速度 */
  double R(2.0e-2); /* ピンポン球の半径 */
  double m(2.7e-3); /* ピンポン球の質量 */
  double Cd(0.3); /* 球の Cd 値 */
  double b(0.5*Cd*PI*1.2*R*R); /* 慣性抗力の係数 */
  double bm(b/m); /* 慣性抗力の係数/m */

  double x,y,xa,ya;
  double v,u,va,ua; /* u, v はそれぞれ x, y の速度を表す変数 */
  double v_abs;     /* 速さ */
  x=0.0;            /* 初期位置と初期速度の設定 */
  y=0.0;
  v=v0*cos(PI/180*th);
  u=v0*sin(PI/180*th);

  ya=-1.0;
  xa=0.0;
  double t;
  for(t=0.0; (y>0.0 || ya<=0.0)&&t<=t1; t+=dt){
    ya=y; /* 1step まえの y を格納 */
    xa=x;

    v_abs=sqrt(v*v+u*u);
    x+=v*dt;
    y+=u*dt;
    v-=bm*v_abs*v*dt;
    u-=g*dt+bm*v_abs*u*dt;
  }
```

```
    double hikyori;
    hikyori=(fabs(y)*xa+fabs(ya)*x)/(fabs(y)+fabs(ya));
    printf("%f %f %f\n",v0,th,hikyori);

  return 0;
}
```

このプログラムは、4 つのコマンドライン引数をもつ。第 1 の引数は物体の投げ出される初速度 v、第 2 の引数は投げ出される角度 θ、第 3 の引数は時間発展を計算する終了時刻 t、第 4 の引数は時間の刻み幅 dt である。これらの物理量は SI 単位系の数値でプログラムに与える。

図は、様々な v_0 の値に対して、飛距離を投射角 θ の関数としてプロットしたグラフである。このグラフより、$v_0 = 1$ の低速の投射では $\theta \simeq 45°$ あたりに飛距離のピークが認められるのに比して、v_0 が大きくなるなればなるほど、θ がより小さな値で最大飛距離となることが明らかである。縦軸は空気抵抗のない放物運動の最大飛距離 v_0^2/g を単位としたので、その値が 1 に近ければ近いほど空気抵抗の影響が小さく、0 に近ければ近いほど空気抵抗による散逸の効果が大きいことが見て取れる。

最後に、図 5 から計算される最大飛距離をあたえる θ の値を初期速度 v_0 の関数としてプロットした（図 6)。このグラフより、$v_0 \gtrsim 1$ の領域では、v_0 を 10 m/s 増やすと、最大飛距離を得るためには斜方投射角を-5°

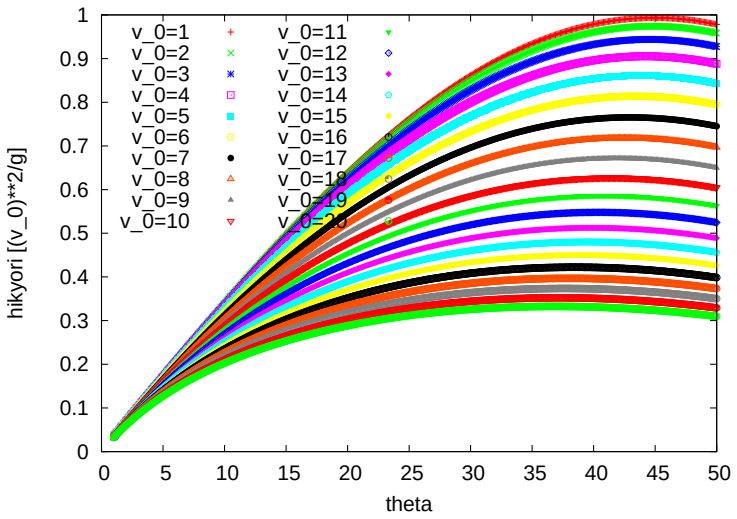

図 5: 初期速度が $v_0 = 1, 2, \ldots, 20$ の場合の飛距離を θ の関数として計算した。時間の刻みは $dt = 0.000001$ を採用した。$v_0 = 1$ では $\theta = 45°$ で、縦軸の値が 1 に達する。この場合は空気抵抗の効果はほとんどない。縦軸は空気抵抗のない放物運動の最大飛距離 v_0^2/g を単位とした。

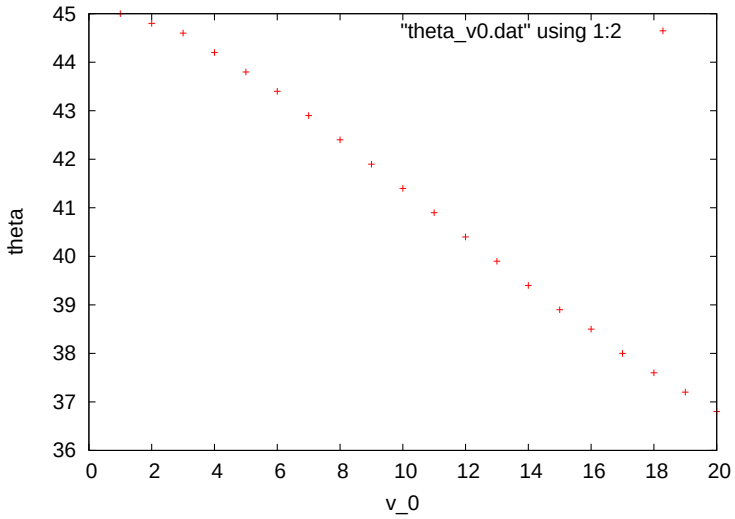

図 6: 最大飛距離を与える θ の値を v_0 の関数としてプロットした。

程度小さくする必要があることが分かった。一方、$v_0 \lesssim 1$ の低速領域では、$\theta \simeq 1 -$ 定数 v_0^2 となり、慣性抗力はほとんど効かない。この領域では本レポートでは無視した粘性抗力を取り入れた数値計算が重要となるであろう。

6 まとめ

計算機を用いて C 言語プログラムを作成し実行することで、空気抵抗を受けるピンポン球の最大飛距離を調べた。

本レポートの数値計算結果は、$F_{\text{慣性}} \propto v^2$ から、定性的に理解可能であった。なぜなら、速ければ速いほど慣性力の効果が大きくなり減速が顕著になる。このときに最大飛距離を得るためには投射角を小さくして減速して失墜するまえに距離を稼ぐ必要があるからである。

高精度の数値計算を行ったおかげで、図 6 などの、より定量的な結果が得られた。これらの結果により実際にどれくらいの角度が最大飛距離となるのかが明確になった。

さらに、考察の結果、今後課題として、$v_0 \lesssim 1$ の初速度では、粘性抗力の効果を取り込んだ計算を行う必要があることが分かった。

A　便利な bash コマンド

パラメタを振って大量の数値計算する場合、ひとつひとつのコマンドをターミナルから入力するのは大変です。以下の内容のファイルをファイル名 hikyori.sh で作成して、

```
for ((v=1;v<=20;++v)) do
for ((th=10;th<=500;++th))
do ./a.out $v $(echo "scale=200;0.1*$th"|bc) 50.0 0.000001;
done >hikyori_v$v.dat
done
```

ターミナルから

```
% bash hikyori.sh
```

とすると、一挙に $v = 1, 2, \ldots, 20$ の計算結果が格納されたファイル hikyori_v1.dat, hikyori_v2.dat, ...,hikyori_v20.dat が作成され、大変便利。

例えば、hikyori_v1.dat の各列に v_0 の値、 θ の値 、飛距離の値が格納されているとき、飛距離が最大の行を取り出すには、

```
% sort -k3 -gr hikyori_v1.dat |head -n1
```

で出来る。すべての hikyori_v*.dat に対して飛距離最大の行を取り出すには、ターミナルから

```
% for f in $(ls hikyori_v*.dat)
>do
>sort -k3 -gr $f |head -n1
>done > saidai_theta.dat
%
```

とすれば簡単に計算できる。

B　便利な gnuplot の使い方

たくさんのファイルを一つのグラフに重ね書きする場合には、ひとつひとつファイル名を入力するのは大変。

```
% gnuplot
%
....
gnuplot> plot for [i=1;10;2] sprintf("data%d.dat",i) using 1:2
```

とすると、

```
gnuplot> plot "data1.dat" using 1:2, "data3.dat" using 1:2, \
> "data5.dat" using 1:2, "data7.dat" using 1:2, "data9.dat" using 1:2
```

を実現できる。

参考文献

[1] パリティ物理学コース「一般物理学（上)」、太田信義