2011 年度卒業研究レポート

S-W 近似による楕円領域での波動方程式のシミュレーション

明治大学理工学部数学科
浜　勇樹
指導教員　桂田祐史　准教授

2012 年 3 月 1 日

目 次

1　概要

インターネット上からのコピーだが以下の様な話がある。

> 「シャーロックホームズはこうやって泥棒をつかまえた。」[1]
> コナンドイルの名作の主人公、シャーロック・ホームズが、ある日酒場に入りました。よくみると泥棒が何人かでテーブルを囲んでひそひそなにやら相談をしていました。ところが、声が小さくてよく聞き取れません。よくみるとこの酒場は、天井が楕円形をしていてその一方の焦点に泥棒たちのテーブルがあるのに気づきました。そこでホームズはこの焦点と反対の位置（焦点）にテーブル席を取りました。すると、泥棒たちの話が手にとるように聞こえてくるではありませんか。こうしてなんなく泥棒の現場をおさえることが出来たといいます。このように、音には、光のように反射していくというような性質も備えています。

この現象が2次元波動方程式を解くことによって再現できるかということに挑戦した。2007年度桂田研卒業, 久保田祥史先輩の「S-W 近似によって様々な領域の熱方程式を解く」[2] のS-W 近似を参考にし、C 言語を用いてプログラミングを行い、計算結果を GLSC を用いて可視化した。

2　楕円盤領域でのS-W近似について

2.1　イントロ

S-W近似の正式な名前は「Shortley-Weller(ショートリィ・ウェラー)近似」と言う。以下、これをS-W近似と呼ぶ。通常、差分法は等間隔格子点を用いて行われる。これは領域が長方形の場合には問題ないが、円盤領域等には合わない。これらの領域で問題を解くには、変数変換によって領域を長方形に変換することで差分法を適用するのが普通である。しかし、このS-W近似ではそのままの領域に対して差分法を行なうことが出来る。ここでは求める領域Ω($\mathbb{R}^2$の有界領域）を

$$\Omega = \left\{ (x,y); \frac{x^2}{a^2} + \frac{y^2}{b^2} < 1 \right\}$$

とし、以下の波動方程式の初期値境界値問題を考える。

$$\begin{cases} \frac{1}{c^2} u_{tt}(x,y,t) = \triangle u(x,y) \quad ((x,y) \in \Omega, t > 0) \\ u(x,y,t) = \phi(x,y) \quad ((x,y) \in \Omega, t = 0) \\ u_t(x,y,t) = \psi(x,y) \quad ((x,y) \in \Omega, t = 0) \\ u(x,y,t) = 0 \quad ((x,y) \in \partial\Omega, t > 0) \end{cases}$$

cは正定数であり、$\phi(x,y), \psi(x,y)$は既知関数である。

2.2　求める領域を差分格子で覆う

楕円領域Ωを覆う正方領域(もしくは長方領域)の左辺のx座標をx_{min}、右辺のx座標をx_{max}、下辺のy座標をy_{min}、上辺のy座標をy_{max}、x方向の分割数をN_x、y方向の分割数をN_y、x方向の格子点の間隔をh_x、y方向の格子点の間隔をh_y、時間の刻み幅$\tau > 0$、とおく。このとき

$$h_x = \frac{x_{max} - x_{min}}{N_x}$$
$$h_y = \frac{y_{max} - y_{min}}{N_y}$$

が成り立つ。さらに、x_i, y_jを

$$x_i = x_{min} + ih_x \quad (0 \leqq i \leqq N_x)$$
$$y_j = y_{min} + jh_y \quad (0 \leqq j \leqq N_y)$$

とする。

2.3 格子点の領域内外の判断

楕円領域 Ω に対して、関数 F を $F(x,y) = a^2b^2 - (b^2x^2 + a^2y^2)$ で定めると、任意の $P = (x,y)$ に対して

$$\begin{cases} F(x,y) > 0 & \text{のとき領域内部} \\ F(x,y) = 0 & \text{のとき境界上} \\ F(x,y) < 0 & \text{のとき領域外部} \end{cases}$$

が成り立つ。この F により任意の点が領域の内部、外部、境界のどこにあるか判断できる。

2.4 境界上の不等間隔格子点を取る

等間隔の格子点によって分けられた領域内のある格子点 $P = (x,y)$ に対して、その左右上下の等間隔格子点が領域外に出てしまう場合、境界上に新しく不等間隔格子点を取る。左側、右側、上側、下側の点をそれぞれ

$$P_w = (x_w, y), P_e = (x_e, y), P_n = (x, y_n), P_s = (x, y_s)$$

と置く。新たに出来た左右上下の格子点の間隔をそれぞれ

$$h_w = x - x_w = \varepsilon_w h_x, \quad h_e = x_e - x = \varepsilon_e h_x$$

$$h_n = y_n - y = \varepsilon_n h_y, \quad h_s = y - y_s = \varepsilon_s h_y$$

$$(0 < \varepsilon_w, \varepsilon_e, \varepsilon_n, \varepsilon_s \leqq 1)$$

とおく。

2.5 境界の座標の計算

格子点の間に存在する境界の座標を計算する。楕円領域 Ω の境界上では

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

が満たされ、x, y それぞれについて解くと

$$x = \pm a\sqrt{1 - \frac{y^2}{b^2}}, \quad y = \pm b\sqrt{1 - \frac{x^2}{a^2}}$$

となる。これによって境界上の x 座標 (x_w, x_e) と y 座標 (y_n, y_s) が分かるので、格子点の左右上下のどの等間隔格子点が境界外に出るかに合わせて、P_w, P_e, P_n, P_s を決める。

2.6 S-W 近似の差分方程式

通常の差分法では、ラプラシアンを

$$\triangle u \cong \frac{1}{h^2}(u_w + u_e + u_n + u_s - 4u)$$

と近似する。S-W 近似では、

$$\triangle u \cong \frac{2}{h_w + h_e}\left(\frac{u_e - u}{h_e} - \frac{u - u_w}{h_w}\right) + \frac{2}{h_n + h_s}\left(\frac{u_n - u}{h_n} - \frac{u - u_s}{h_s}\right)$$

と近似する。波動方程式

$$\frac{1}{c^2}u_{tt} = \triangle u$$

の左辺を 2 階の中心差分で近似すると、差分方程式は

$$\frac{1}{c^2}\frac{u_{i,j}^{n+1} - 2u_{i,j}^{n} + u_{i,j}^{n-1}}{\tau^2} = \frac{2}{h_w + h_e}\left(\frac{u_e - u_{i,j}^{n}}{h_e} - \frac{u_{i,j}^{n} - u_w}{h_w}\right) + \frac{2}{h_n + h_s}\left(\frac{u_n - u_{i,j}^{n}}{h_n} - \frac{u_{i,j}^{n} - u_s}{h_s}\right)$$

という形になる。u^{n+1} について整理すると

$$u_{i,j}^{n+1} = 2u_{i,j}^{n} - u_{i,j}^{n-1} + \frac{2\tau^2 c^2}{h_w + h_e}\left(\frac{u_e - u_{i,j}^{n}}{h_e} - \frac{u_{i,j}^{n} - u_w}{h_w}\right) + \frac{2\tau^2 c^2}{h_n + h_s}\left(\frac{u_n - u_{i,j}^{n}}{h_n} - \frac{u_{i,j}^{n} - u_s}{h_s}\right)$$

となる。これは $h_w = h_e = h_n = h_s = h$ のとき、通常の差分方程式に帰結する。

2.7 差分スキームの安定性条件,CFL 条件

Wikipedia[4] には

> CFL 条件（Courant-Friedrichs-Lewy Condition じょうけん）とは、コンピュータシミュレーションの計算（数値解析）において、「情報が伝播する速さ」を「実際の現象で波や物理量が伝播する速さ」よりも早くしなければならないという必要条件のことである。

とある。

今回の波動方程式の場合に当てはめて考えると、等間隔格子を使った差分法の場合

$$\left(\frac{c\tau}{h_x}\right)^2 + \left(\frac{c\tau}{h_y}\right)^2 \leqq 1$$

となる。これを τ について解くと

$$\tau \leqq \sqrt{\frac{h_x^2 h_y^2}{c^2(h_x^2 + h_y^2)}}$$

という τ の条件が得られる。

次に、S-W 近似における安定性条件について考える。まず、

$$\triangle u \cong \frac{2}{h_w + h_e}\left(\frac{u_e - u}{h_e} - \frac{u - u_w}{h_w}\right) + \frac{2}{h_n + h_s}\left(\frac{u_n - u}{h_n} - \frac{u - u_s}{h_s}\right)$$

に、$h_w = \varepsilon_w h_x, h_e = \varepsilon_e h_x, h_n = \varepsilon_n h_y, h_s = \varepsilon_s h_y$ を代入すると

$$\triangle u \cong \frac{2}{\varepsilon_w hx + \varepsilon_e hx}\left(\frac{u_e - u}{\varepsilon_e hx} - \frac{u - u_w}{\varepsilon_w hx}\right) + \frac{2}{\varepsilon_n hy + \varepsilon_s hy}\left(\frac{u_n - u}{\varepsilon_n hy} - \frac{u - u_s}{\varepsilon_s hy}\right)$$

となる。これを変形すると

$$\triangle u \cong \frac{2(\varepsilon_w u_e + \varepsilon_e u_e)}{h_x^2 \varepsilon_w \varepsilon_e(\varepsilon_w + \varepsilon_e)} + \frac{2(\varepsilon_s u_n + \varepsilon_n u_s)}{h_y^2 \varepsilon_s \varepsilon_n(\varepsilon_s + \varepsilon_n)} - \frac{2(\varepsilon_w \varepsilon_e h_x^2 + \varepsilon_s \varepsilon_n h_y^2)}{\varepsilon_w \varepsilon_e \varepsilon_n \varepsilon_s h_x^2 h_y^2} u$$

となる、ここで

$$u_{tt} \cong \frac{u_{i,j}^{n+1} - 2u_{i,j}^n + u_{i,j}^{n-1}}{\tau^2}$$

であるから、差分方程式は

$$\frac{1}{c^2}\frac{u_{i,j}^{n+1} - 2u_{i,j}^n + u_{i,j}^{n-1}}{\tau^2} = \frac{2(\varepsilon_w u_e + \varepsilon_e u_e)}{h_x^2 \varepsilon_w \varepsilon_e(\varepsilon_w + \varepsilon_e)} + \frac{2(\varepsilon_s u_n + \varepsilon_n u_s)}{h_y^2 \varepsilon_s \varepsilon_n(\varepsilon_s + \varepsilon_n)} - \frac{2(\varepsilon_w \varepsilon_e h_x^2 + \varepsilon_s \varepsilon_n h_y^2)}{\varepsilon_w \varepsilon_e \varepsilon_n \varepsilon_s h_x^2 h_y^2} u_{i,j}^n$$

となる。u^{n+1} について解くと

$$u^{n+1} = -u^{n-1} + \left(\frac{2(\varepsilon_w u_e + \varepsilon_e u_e)}{h_x^2 \varepsilon_w \varepsilon_e(\varepsilon_w + \varepsilon_e)} + \frac{2(\varepsilon_s u_n + \varepsilon_n u_s)}{h_y^2 \varepsilon_s \varepsilon_n(\varepsilon_s + \varepsilon_n)}\right) c^2\tau^2 + \left(2 - \frac{2c^2\tau^2(\varepsilon_w \varepsilon_e h_x^2 + \varepsilon_s \varepsilon_n h_y^2)}{\varepsilon_w \varepsilon_e \varepsilon_n \varepsilon_s h_x^2 h_y^2}\right) u_{i,j}^n$$

となる。第三項の係数が正になるようにすると、

$$2 \geqq \frac{2c^2\tau^2(\varepsilon_w \varepsilon_e h_x^2 + \varepsilon_s \varepsilon_n h_y^2)}{\varepsilon_w \varepsilon_e \varepsilon_n \varepsilon_s h_x^2 h_y^2}$$

となる。これを τ について整理すると

$$\tau \leqq \sqrt{\frac{\varepsilon_w \varepsilon_e \varepsilon_n \varepsilon_s h_x^2 h_y^2}{c^2(\varepsilon_w \varepsilon_e h_x^2 + \varepsilon_s \varepsilon_n h_y^2)}}$$

となる。$\varepsilon_w = \varepsilon_e = \varepsilon_n = \varepsilon_s = 1$ のとき、この式は上に述べた等間隔格子の CFL 条件に一致しているので、2 次元波動方程式を S-W 近似によって解く場合の CFL 条件であると考えられる。

また、

$$\varepsilon = \min\{\varepsilon_w, \varepsilon_e, \varepsilon_n, \varepsilon_s\}$$

とおく。

2.8 丸め誤差

格子点が境界上にある場合に、プログラムが丸め誤差のせいで領域の内部だと判断することがある。よってその対策をプログラムに組み込む必要がある。何の対策もしなければ、その等

間隔格子点上を領域の内部だと勘違いをして ε を極端に小さい値として返す。*double* 型では、だいたい 10^{-14}〜10^{-15} ぐらいである。この対策として $\varepsilon_r = 10^{-13}$〜10^{-14} というものを置き、これを用いて点 (x, y) が領域の内部、外部、境界のどこに位置するのか判断する。

$$\begin{cases} F(x,y) > \varepsilon_r & \text{のとき領域内部} \\ |F(x,y)| \leqq \varepsilon_r & \text{のとき境界上} \\ F(x,y) < \varepsilon_r & \text{のとき領域外部} \end{cases}$$

と判断することにする。また、この誤差は、領域の形や N_x, N_y の大小によって多少変化する。なので、もし ε が ε_r を下回るとき、そのことを出力するようにプログラムを組み込んでおく。

2.9 差分方程式を解くために

差分方程式を見ると $n+1$ での値を求めるために、一段前の n での値のみならず、もう一段前の $n-1$ での値が必要になる。これは、もとの方程式が時刻 t に関して 2 階であることに対応している。そのため計算を出発させるためには、$n = 0$ での値だけでなく、$n = 1$ での値も必要になる。[3] を参考に、そのための計算を行う。x, y を固定した 1 変数関数 $t \mapsto u(x, y, t)$ の $t = 0$ におけるテーラー展開を行う。

$$u(x,y,\tau) \cong u(x,y,0) + \tau \frac{\partial u}{\partial t}(x,y,0) + \frac{\tau^2}{2}\frac{\partial^2 u}{\partial t^2}(x,y,0)$$

において、関係式

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2}(x,y,0) = \triangle u(x,y,0) = \triangle \phi$$

が成立する。

$$\begin{aligned} u(x,y,\tau) &\cong u(x,y,0) + \tau \frac{\partial u}{\partial t}(x,y,0) + \frac{\tau^2 c^2}{2}\triangle u(x,y,0) \\ &\cong \phi(x,y) + \tau\psi(x,y) \\ &\quad + \frac{\tau^2 c^2}{2}\left(\frac{2}{h_w + h_e}\left(\frac{u_e - u}{h_e} - \frac{u - u_w}{h_w}\right) + \frac{2}{h_n + h_s}\left(\frac{u_n - u}{h_n} - \frac{u - u_s}{h_s}\right)\right). \end{aligned}$$

これを整理すると

$$\begin{aligned} u^1_{i,j} &= \phi(x_i, y_j) + \tau\psi(x_i, y_j) \\ &\quad + \left\{\frac{\tau^2 c^2}{h_w + h_e}\left(\frac{u_e - u^0_{i,j}}{h_e} - \frac{u^0_{i,j} - u_w}{h_w}\right) + \frac{\tau^2 c^2}{h_n + h_s}\left(\frac{u_n - u^0_{i,j}}{h_n} - \frac{u^0_{i,j} - u_s}{h_s}\right)\right\} \end{aligned}$$

という式が得られる。これで数値計算の準備が出来た。

3　数値実験

3.1　楕円領域の焦点

領域 Ω で波動方程式を解くにあたって、$(x_0, 0)$ を $a > b$ の場合の x 軸上にある正の値の焦点の座標、$(0, y_0)$ を $b > a$ の場合の y 軸上にある正の値の焦点の座標とする。以下の実験では $a = 4, b = 3$ とする。すなわち

$$\Omega = \left\{ (x, y); \frac{x^2}{16} + \frac{y^2}{9} < 1 \right\}$$

$x_0 = \sqrt{a^2 - b^2} = \sqrt{7}$ であり、$(\sqrt{7}, 0)$ は楕円の焦点となる。

3.2　初期値の設定

初期値とは、波動方程式

$$\begin{cases} \frac{1}{c^2} u_{tt}(x, y) = \triangle u(x, y, t) \quad ((x, y) \in \Omega, t > 0) \\ u(x, y, t) = \phi(x, y) \quad ((x, y) \in \Omega, t = 0) \\ u_t(x, y, t) = \psi(x, y) \quad ((x, y) \in \Omega, t = 0) \\ u(x, y, t) = 0 \quad ((x, y) \in \partial\Omega, t > 0) \end{cases}$$

の、ϕ と ψ に当たる。プログラム上では $\phi = f$ と $\psi = g$ と置くことにした。検証したいのはシャーロックホームズの話が再現できるか、つまり、片方の焦点から発せられた波がもう片方の焦点に向かって反射し収束するのかということである。よって、片方の焦点上にピークをもち領域内のいたるところで 0 になるような初期値をあたえる。

$a > b$ の場合、$H(x, y) = K \exp(-M\{(x - x_0)^2 + y^2)\})$ を用意し、

$$f(x, y) = \begin{cases} H(x, y) - 0.1 & (H(x, y) \geqq 0.1) \\ 0 & (H(x, y) < 0.1) \end{cases}$$

とし、g については

$$g(x, y) = 0$$

とした。K, M で初期値の波のサイズ（幅、高さ）を変更できる。今回は $K = 3, M = 10$ とした。

4 付録

4.1 hadouD.c

```
/*
 * hadouD.c --- 楕円領域での波動方程式を SW 近似で解く
 */

#include <stdio.h>
#include <math.h>
#define M (10)
#define K (3)
#define pai (3. 141592653589)
double a, b, c, xmin, xmax, ymin, ymax, distance, x0, y00;

/* to use matrix, new_matrix() */
#include <matrix.h>

/* to use GLSC */
#define G_DOUBLE
#include <glsc.h>
#include <glsc2.h>

/* 2 乗の関数  */
double d(double x){
  return x*x;
}

/* 領域の内部・境界・外部の判定用 */
double F(double x, double y){
  return d(a)*d(b) - (d(b)*d(x) + d(a)*d(y));
}

/* 東側の境界 */
double E(double x, double y){
  return a * sqrt(1 - d(y)/d(b));
}

/* 西側の境界 */
double W(double x, double y){
  return - a * sqrt(1 - d(y)/d(b));
}

/* 北側の境界 */
double N(double x, double y){
  return b* sqrt(1 - d(x)/d(a));
}

/* 南側の境界 */
double S(double x, double y){
  return - b * sqrt(1 - d(x)/d(a));
}

/* 初期条件 */
double f(double x, double y){
  double H;
  if(a>b){
```

```
    H= K*exp(-M*(d(x-x0)+d(y)));
  }
  else if(b>a){
    H= K*exp(-M*(d(x)+d(y-y00)));
  }
  else{
    H= K*exp(-M*(d(x)+d(y)));
  }
  if(H-0.1>=0) return H-0.1;
  else return 0;
}

double g(double x, double y){
  return 0;
}

/* 大小比較の関数 */
double min(double x, double y){
  if(x < y){
  return x;
  }
  else return y;
}

int main()
{
  int    Nx, Ny, i, j, n, skip, nMax;
  double hx, hy, tau, Tmax, t, dt, ew, es, ee, en, er, mine, maxtau;
  double x, y, hw, he, hn, hs;
  double xi, xiw, xie, yj, yjs, yjn;
  double uw,ue,un,us;
  matrix u1,u2,u3;
  double W_MARGIN, H_MARGIN, WIDTH, HEIGHT;
  int k;
  char str[100];
  double sheta;

  /* 波の速さ */
  c = 1.0;

  /* 領域の x 軸、y 軸の長さ */
  xmin = -5;
  xmax = 5;
  ymin = -5;
  ymax = 5;

  /* 分割数 */
  printf("Nx=  "); scanf("%d",&Nx);
  printf("Ny=  "); scanf("%d",&Ny);

  /* 楕円の長軸、短軸 */
  a=4.0;
  b=3.0;

  /* 初期条件のピークの位置 */
  x0 = sqrt(d(a)-d(b));
  y00 = sqrt(d(b)-d(a));
```

```
/* 距離 */
distance = 60;

/* 格子間の長さ */
hx = (xmax-xmin)/Nx;
hy = (ymax-ymin)/Ny;
printf("hx=%g, hy=%g\n", hx, hy);

/* 誤差 */
er= 1.0e-13;

mine= 1.0;
maxtau= 1.0;

/*****************最小εと最大τを求める ******************/

for (i = 0;i <= Nx; i++){
  xi = xmin+i*hx;
  xiw = xmin+(i-1)*hx;
  xie = xmin+(i+1)*hx;

  for (j = 0;j <= Ny; j++){
    yj = ymin+j*hy;
    yjs = ymin+(j-1)*hy;
    yjn = ymin+(j+1)*hy;

    if(F(xi, yj) >= er){
      /* WEST */
      if(F(xiw, yj) >= er){
        hw= hx;
        ew= 1.0;
      }
      else if(fabs(F(xiw, yj)) < er){
        ew= 1.0;
        hw= hx;
      }
      else{
        hw= xi - W(xi,yj);
        ew= hw/hx;
      }
      /* SOUTH */
      if(F(xi, yjs) >= er){
        hs= hy;
        es= 1.0;
      }
      else if(fabs(F(xi, yjs)) < er){
        hs= hy;
        es= 1.0;
      }
      else{
        hs= yj - S(xi,yj);
        es= hs/hy;
      }
      /* EAST */
      if(F(xie, yj) >= er){
        he= hx;
        ee= 1.0;
      }
```

```
      else if(fabs(F(xie, yj)) < er){
        ee= 1.0;
        he= hx;
      }
      else {
        he= E(xi,yj) - xi;
        ee= he/hx;
      }
      /* NORTH */
      if(F(xi, yjn) >= er){
        en= 1.0;
        hn= hy;
      }
      else if(fabs(F(xi, yjn)) < er){
        en= 1.0;
        hn= hy;
      }
      else{
        hn= N(xi,yj) - yj;
        en= hn/hy;
      }
    }
    else {
      ew = ee = en = es = 1.0;
      hw = he = hx;
      hn = hs = hy;
    }

    mine = min(mine, min(min(ew,ee),min(en,es)));

    maxtau = min(maxtau,
          sqrt((ew*es*ee*en*d(hx)*d(hy))/(d(hx)*ew*ee+d(hy)*es*en)));
  }
}

printf("ε の最小値= %g\n", mine);
printf("τ の最大値=%g\n", maxtau);
if(mine < er){
  printf("mine=%g: ******* ε が小さすぎます********\n", mine);
}

if ((u1 = new_matrix(Nx + 1, Ny + 1)) == NULL) {
  fprintf(stderr, "配列 u1 を確保できませんでした。");
  exit(1);
}
if ((u2 = new_matrix(Nx + 1, Ny + 1)) == NULL) {
  fprintf(stderr, "配列 u2 を確保できませんでした。");
  exit(1);
}
if ((u3 = new_matrix(Nx + 1, Ny + 1)) == NULL) {
  fprintf(stderr, "配列 u3 を確保できませんでした。");
  exit(1);
}

printf("Tmax=  "); scanf("%lf", &Tmax);
printf("τ (≦ %g )==  ", maxtau); scanf("%lf", &tau);
printf("Δ t=  "); scanf("%lf",  &dt);
```

```
skip = rint(dt / tau);
if (skip == 0) {
  printf("Δ t が小さすぎるので、Δ t=  τ  とします。\n");
  skip = 1;
  dt = skip * tau;
}

/* 初期値の入力 */
t=0.0;
for (i = 0; i <= Nx; i++){
  xi = xmin+i*hx;
  for (j = 0; j <= Ny; j++){
    yj = ymin+j*hy;
    u1[i][j] = f(xi, yj);
  }
}

/* u2 の計算 */
for (i = 0; i <= Nx; i++){
  xi = xmin+i*hx;
  xiw = xmin+(i-1)*hx;
  xie = xmin+(i+1)*hx;
  for (j = 0; j <= Ny; j++){
    yj = ymin+j*hy;
    yjn = ymin+(j+1)*hy;
    yjs = ymin+(j-1)*hy;

    if(F(xi, yj) >= er){
      /* WEST */
      if(F(xiw, yj) >= er){
        hw= hx;
        uw= u1[i-1][j];
      }
      else if(fabs(F(xiw, yj)) < er){
        hw= hx;
        uw= u1[i-1][j];
      }
      else{
        hw= xi - W(xi,yj);
        uw= 0;
      }
      /* EAST */
      if(F(xie, yj) >= er){
        he=hx;
        ue=u1[i+1][j];
      }
      else if(fabs(F(xie, yj)) < er){
        he=hx;
        ue=u1[i+1][j];
      }
      else{
        he= E(xi,yj) - xi;
        ue= 0;
      }
      /* NORTH */
      if(F(xi, yjn) >= er){
        hn=hy;
        un=u1[i][j+1];
```

```
      }
      else if(fabs(F(xi, yjn)) < er){
        hn=hy;
        un=u1[i][j+1];
      }
      else{
        hn= N(xi,yj) - yj;
        un= 0;
      }
      /* SOUTH */
      if(F(xi, yjs) >= er){
        hs=hy;
        us=u1[i][j-1];
      }
      else if(fabs(F(xi, yjs)) < er){
        hs=hy;
        us=u1[i][j-1];
      }
      else{
        hs=yj - S(xi,yj);
        us=0;
      }
      u2[i][j] = u1[i][j]+tau*g(xi,yj)
      + d(tau)*d(c)*((ue-u1[i][j])/he - (u1[i][j]-uw)/hw) /(he+hw)
      + d(tau)*d(c)*((un-u1[i][j])/hn - (u1[i][j]-us)/hs) /(hn+hs);
    }
    else{
      u1[i][j] = 0;
      u2[i][j] = 0;
      hw= he= hx;
      hn= hs= hy;
      uw= ue= un= us= 0;
    }
  }
}
W_MARGIN = 10.0; H_MARGIN = 10.0; WIDTH = 100.0; HEIGHT = 100.0;
g_init("Meta", 2*WIDTH+3*W_MARGIN, HEIGHT + 2*H_MARGIN);
g_device(G_BOTH);
g_def_scale(0, xmin, xmax, ymin, ymax, W_MARGIN, H_MARGIN, WIDTH, HEIGHT);
g_def_line(0, G_BLACK, 0, G_LINE_SOLID);
g_sel_scale(0);

g_text(WIDTH+2*W_MARGIN, H_MARGIN, "t = 0.0");
g_move( a , 0 );
    for(k = 0; k < 63; k++){
      sheta=k*0.1;
      g_plot(a*cos(sheta),b*sin(sheta));
    }

g_line_color (G_BLUE);
for (k = -10; k <= 0; k++){
  g_contln2 (xmin, xmax, ymin, ymax, u1, Nx+1, Ny+1, k * 0.2 );
}
g_line_color (G_RED);
for (k = 0; k <= 10; k++){
  g_contln2 (xmin, xmax, ymin, ymax, u1, Nx+1, Ny+1, k * 0.2 );
}
```

```
g_hidden2(xmax-xmin, ymax-ymin, 10.0, 0.0, 10.0,
              distance, 10.0, 20.0,
              WIDTH+2*W_MARGIN, H_MARGIN, WIDTH, HEIGHT,
              u1, Nx+1, Ny+1, 1, G_SIDE_NONE, 1, 1);
g_sleep(1.0);

nMax = rint(Tmax/tau);
for (n = 2; n <= nMax ; n++){
/**********************領域**********************/
  for (i = 0; i <= Nx; i++){
    xi = xmin+i*hx;
    xiw = xmin+(i-1)*hx;
    xie = xmin+(i+1)*hx;

    for (j = 0; j <= Ny; j++){
      yj = ymin+j*hy;
      yjn = ymin+(j+1)*hy;
      yjs = ymin+(j-1)*hy;

      if(F(xi, yj) >= er){
        /* WEST */
        if(F(xiw, yj) >= er){
          hw= hx;
          uw= u2[i-1][j];
        }
        else if(fabs(F(xiw, yj)) < er){
          hw= hx;
          uw= u2[i-1][j];
        }
        else{
          hw= xi - W(xi,yj);
          uw= 0;
        }
        /* EAST */
        if(F(xie, yj) >= er){
          he=hx;
          ue=u2[i+1][j];
        }
        else if(fabs(F(xie, yj)) < er){
          he=hx;
          ue=u2[i+1][j];
        }
        else{
          he= E(xi,yj) - xi;
          ue= 0;
        }
        /* NORTH */
        if(F(xi, yjn) >= er){
          hn=hy;
          un=u2[i][j+1];
        }
        else if(fabs(F(xi, yjn)) < er){
          hn=hy;
          un=u2[i][j+1];
        }
        else{
          hn= N(xi,yj) - yj;
          un= 0;
```

```
      }
      /* SOUTH */
      if(F(xi, yjs) >= er){
        hs=hy;
        us=u2[i][j-1];
      }
      else if(fabs(F(xi, yjs)) < er){
        hs=hy;
        us=u2[i][j-1];
      }
      else{
        hs= yj - S(xi,yj);
        us= 0;
      }
      u3[i][j] = 2.0*u2[i][j]-u1[i][j]
      + 2.0*d(tau)*d(c)*((ue-u2[i][j])/he - (u2[i][j]-uw)/hw) /(he+hw)
      + 2.0*d(tau)*d(c)*((un-u2[i][j])/hn - (u2[i][j]-us)/hs) /(hn+hs);
    }
    else{
      u1[i][j] = 0;
      u2[i][j] = 0;
      u3[i][j] = 0;
      hw= he= hx;
      hn= hs= hy;
      uw= ue= un= us= 0;
    }
  }
}

for (i = 0; i <= Nx; i++){
  for (j = 0; j <= Ny; j++){
    u1[i][j] = u2[i][j];
    u2[i][j] = u3[i][j];
  }
}
t = tau * n;
if (n % skip == 0){
  g_cls();
  g_line_color(G_BLACK);

  g_move( a , 0 );
  for(k = 0; k < 65; k++){
    sheta=k*0.1;
    g_plot(a*cos(sheta),b*sin(sheta));
  }

  sprintf(str, "t = %lf", t);
  g_text(WIDTH+2*W_MARGIN, H_MARGIN, str);

  g_line_color (G_BLUE);
  for (k = -10; k <= 0; k++){
    g_contln2 (xmin, xmax, ymin, ymax, u2, Nx+1, Ny+1, k * 0.2 );
  }

  g_line_color (G_RED);
  for (k = 0; k <= 10; k++){
    g_contln2 (xmin, xmax, ymin, ymax, u2, Nx+1, Ny+1, k * 0.2 );
  }
```

```
      g_hidden2(xmax-xmin, ymax-ymin, 10.0, 0.0, 10.0,
                distance, 10.0, 20.0,
                WIDTH+2*W_MARGIN, H_MARGIN, WIDTH, HEIGHT,
                u2, Nx+1, Ny+1, 1, G_SIDE_NONE, 1, 1);

    }
  }
  /* マウスでクリックされるのを待つ */
  g_sleep(-1.0);
  /* ウィンドウを消す */
  g_term();
  return 0;
}
```

4.2 実行結果

楕円は平面上の2つの焦点からの距離の和が一定であるという性質を持っている。長軸$a = 4$、短軸$b = 3$、波の速さ$c = 1$という条件から片方の焦点から出た波が境界で反射し、もう片方の焦点へ行くのにかかる時間は8と予想される。したがって$t = 8$と$t = 16$の場合の結果を選び載せた。

同時に等高線も表示している。等高線の赤は正を意味し、青が負を意味する。

結果は以下の通りとなった。

片方の焦点上にピークをもち領域内のいたるところで0になるような初期値が楕円内を反射しもう片方の焦点に集まっている現象は何となくというレベルでしか見られなかった。計算の精度の問題かと思い、分割数を増やしてみたが結果は大きくは変わらなかった。これは2次元の波動方程式の解では、ホイヘンスの原理が成り立たないためかと考えられるが、実際は分からない。3次元での数値計算の必要性を感じた。

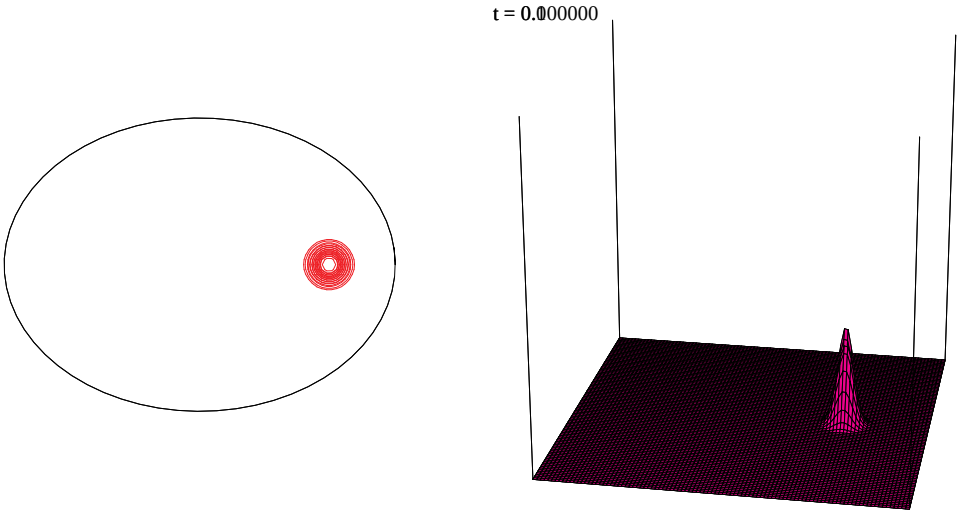

図 1: $t = 0$ の状態

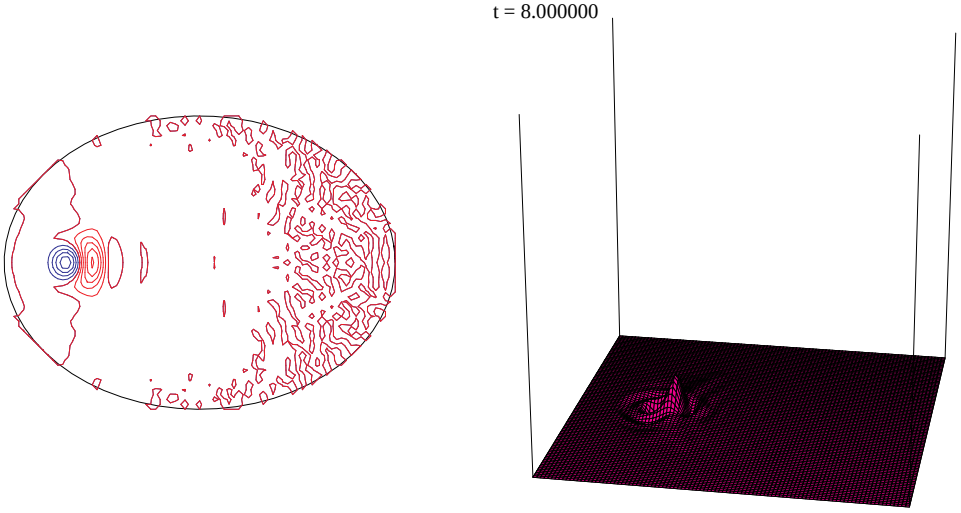

図 2: $t = 8$ の状態

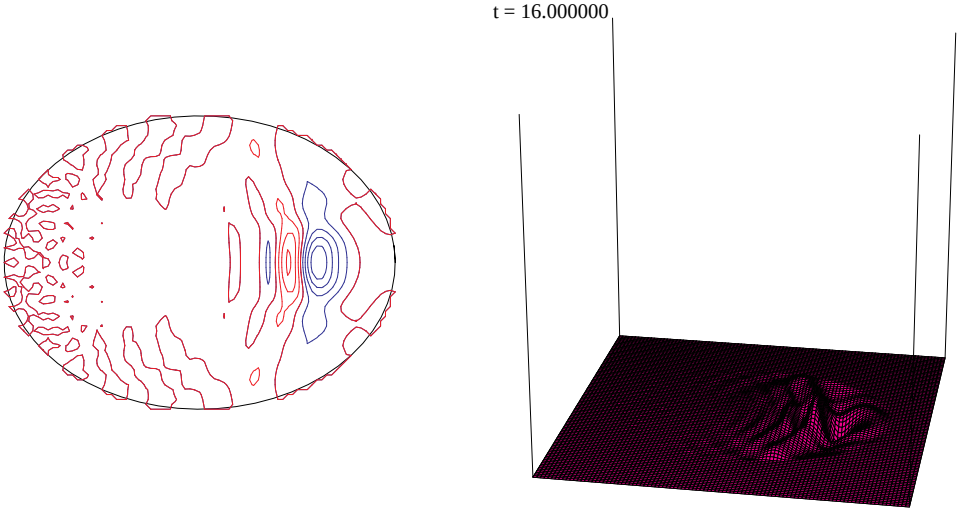

図 3: $t = 16$ の状態

参考文献

[1] 音のエピソード
http://www13.plala.or.jp/sound-planner/episo-do.html

[2] 久保田　祥史,「S-W 近似によって様々な領域の熱方程式を解く」
2007 年度桂田研卒業研究レポート
http://www.math.meiji.ac.jp/~mk/labo/report/open/2007-kubota.pdf

[3] 桂田 祐史, 波動方程式に対する差分法
http://www.math.meiji.ac.jp/~mk/labo/text/wave.pdf

[4] *wikipedia,Courant-Friedrichs-Lewy condition*（2012 年 3 月 1 日）
http://en.wikipedia.org/wiki/Courant%E2%80%93Friedrichs%E2%80%93Lewy_condition