問 1：

(1) まず台形公式のプログラムは

```
program daikei

real(8),external::f
real(8)::eps,a,b,h,s,tn,t,exact
eps=1.0d-15
b=1.0_8
a=0.0_8
exact=atan(1.0_8)
n=1
h=b-a
t=h*(f(a)+f(b))/2.0_8

do  k=1,25

    n=2*n
    h=h/2.0_8
    s=0.0_8

      do  i=1,n-1,2

              s=s+f(a+i*h)

      enddo

    tn=t/2.0+h*s
    if (abs(tn-t) < eps*abs(t)) exit
    t=tn
    print*,h,abs(tn-exact)
enddo



end

real(8) function  f(x)
real(8)::x
f=1.0_8/(1.0_8+x*x)
end
```

結果は以下のようになる。1 列目が h で 2 列目が正しい値との差の絶対値。

```
  0.500000000000000        1.039816339744826E-002
  0.250000000000000        2.604045750389528E-003
  0.125000000000000        6.510397746760654E-004
  6.250000000000000E-002   1.627603871010574E-004
  3.125000000000000E-002   4.069010370466586E-005
  1.562500000000000E-002   1.017252603430219E-005
  7.812500000000000E-003   2.543131510379659E-006
  3.906250000000000E-003   6.357828774561369E-007
  1.953125000000000E-003   1.589457194750565E-007
  9.765625000000000E-004   3.973643003529759E-008
  4.882812500000000E-004   9.934107536579972E-009
  2.441406250000000E-004   2.483527050678447E-009
  1.220703125000000E-004   6.208820124697922E-010
  6.103515625000000E-005   1.552205031174481E-010
  3.051757812500000E-005   3.880629151353787E-011
  1.525878906250000E-005   9.701572878384468E-012
  7.629394531250000E-006   2.427835710250292E-012
  3.814697265625000E-006   6.088463067044358E-013
  1.907348632812500E-006   1.443289932012704E-013
  9.536743164062500E-007   3.264055692397960E-014
  4.768371582031250E-007   1.687538997430238E-014
  2.384185791015625E-007   6.883382752675971E-015
  1.192092895507813E-007   4.640732242933154E-014
  5.960464477539063E-008   7.671641100159832E-014
  2.980232238769531E-008   1.709743457922741E-014
  2.980232238769531E-008   1.709743457922741E-014
```

次にシンプソン公式を用いたプログラム

```
program simpson

real(8),external::f
real(8)::eps,a,b,h,xi,t,ss,tn,sn,s,exact
eps=1.0d-15
n=2
b=1.0_8
a=0.0_8
```

```
exact=datan(1.0_8)
h=(b-a)/2.0_8
xi=(a+b)/2.0_8
t=h*(f(a)+f(b)+2.0_8*f(xi))/2.0_8
ss=h*(f(a)+f(b)+4.0_8*f(xi))/3.0_8


do k=1,15

   n=2*n
   h=h/2.0_8
   s=0.0_8

   do  i=1,n-1,2

       s=s+f(a+i*h)

   enddo

   tn=t/2.0_8+h*s
   sn=(4.0_8*tn-t)/3.0_8
   if(abs(sn-ss) < eps*abs(sn)) exit
   print*,h,abs(sn-exact)
   t=tn
   ss=sn

enddo

end

real(8)function f(x)
real(8)::x
f=1.0_8/(1.0_8+x*x)
end
```

そして結果。さきほどと同じように 1 列目が h で 2 列目が正しい値との差の絶対値。

```
  0.250000000000000       6.006534703284494E-006
  0.125000000000000       3.778277157806542E-008
  6.250000000000000E-002  5.912427214482818E-010
```

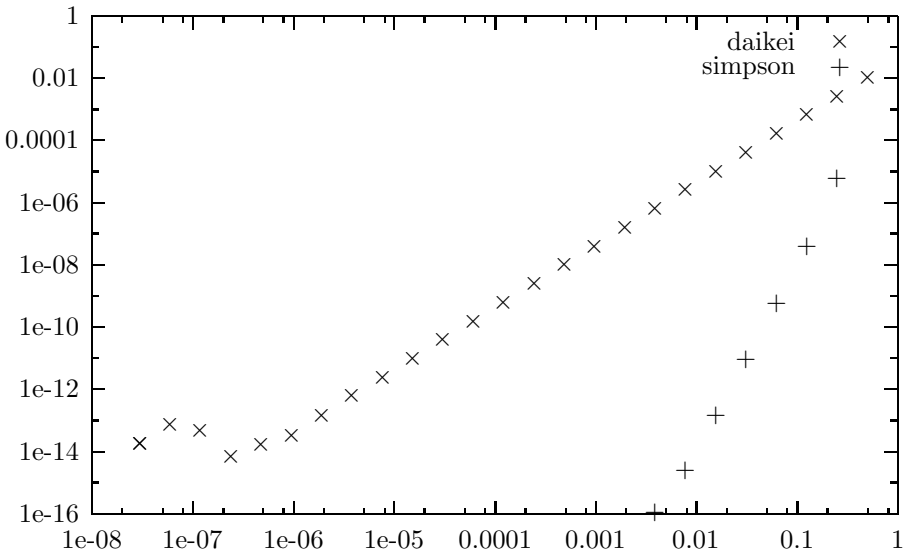

図 1: 台形公式とシンプソン公式の刻幅対誤差のグラフ (1)

```
3.125000000000000E-002  9.239164988628090E-012
1.562500000000000E-002  1.442179708988078E-013
7.812500000000000E-003  2.331468351712829E-015
3.906250000000000E-003  1.110223024625157E-016
```

図 1 に台形公式とシンプソン公式の結果をグラフにあらわす。横軸が h、縦軸が正しい値との差の絶対値。両軸とも log スケールである。

(2)

プログラムは (1) の被積分関数と積分範囲を変えただけなので省略。

台形公式の結果

```
0.500000000000000       3.472222222222221E-002
0.250000000000000       8.993764172335661E-003
0.125000000000000       2.270850326336560E-003
6.250000000000000E-002  5.691701269964211E-004
3.125000000000000E-002  1.423845908211652E-004
1.562500000000000E-002  3.560191675611168E-005
7.812500000000000E-003  8.900839995940046E-006
3.906250000000000E-003  2.225232553110246E-006
1.953125000000000E-003  5.563095479832469E-007
9.765625000000000E-004  1.390774748699641E-007
4.882812500000000E-004  3.476937437962846E-008
```

```
 2.441406250000000E-004  8.692344177774203E-009
 1.220703125000000E-004  2.173086266488156E-009
 6.103515625000000E-005  5.432714278441608E-010
 3.051757812500000E-005  1.358176904275865E-010
 1.525878906250000E-005  3.395406178441362E-011
 7.629394531250000E-006  8.491207736938122E-012
 3.814697265625000E-006  2.126410159064562E-012
 1.907348632812500E-006  5.310196726782124E-013
 9.536743164062500E-007  1.350031197944190E-013
 4.768371582031250E-007  3.708144902248023E-014
 2.384185791015625E-007  8.881784197001252E-016
 1.192092895507813E-007  6.883382752675971E-015
 5.960464477539063E-008  2.564615186884112E-014
 2.980232238769531E-008  9.992007221626409E-016
 2.980232238769531E-008  9.992007221626409E-016
```

シンプソン公式の結果

```
 0.250000000000000       4.176114890400706E-004
 0.125000000000000       2.987904433682242E-005
 6.250000000000000E-002  1.943393883041544E-006
 3.125000000000000E-002  1.227454293761809E-007
 1.562500000000000E-002  7.692067760523003E-009
 7.812500000000000E-003  4.810759568485423E-010
 3.906250000000000E-003  3.007216697881177E-011
 1.953125000000000E-003  1.879607580690390E-012
 9.765625000000000E-004  1.171285290979540E-013
 4.882812500000000E-004  7.549516567451064E-015
 2.441406250000000E-004  7.771561172376096E-016
```

図 2 に台形公式とシンプソン公式の結果をグラフにあらわす。横軸が h、縦軸が正しい値との差の絶対値。両軸とも log スケールである。

(3)

ルンゲクッタ法を用いたプログラム例を以下にしめす。

n：粒子個数

mmm：出力回数

$tend$：終了時間 (始まりは時刻 0)

$q(i)$：x_i

$p(i)$：v_i

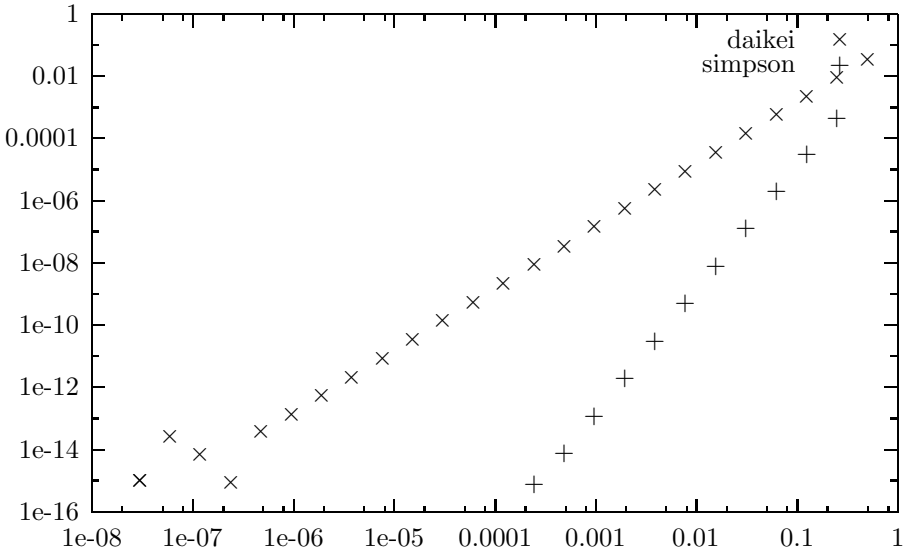

図 2: 台形公式とシンプソン公式の刻幅対誤差のグラフ (2)

```
module com

integer,parameter::n=3,mmm=100
real(8),parameter::dt=5.d-5

end module

program bane

use com

real(8)::q(1:n),p(1:n),qd(1:n),pd(1:n),t,tend

integer::ntime,ncycle


open(1,file='report3x1.dat')
open(2,file='report3x2.dat')
open(3,file='report3x3.dat')
open(4,file='report3-5.dat')
```

```
tend=5.d0
ntime=int(tend/dt)/mmm
ncycle=mmm

t=0.0_8

!!!initial condition!!!

do j=1,n

  q(j)=0.0_8
  p(j)=0.0_8

enddo

q(1)=1.0_8

!!!!!!!!!!!!!!!!!!!!!



write(1,*)q(1),p(1)
write(2,*)q(2),p(2)
write(3,*)q(3),p(3)


do j=1,n-1

   write(4,*)t,dble(j),q(j+1)-q(j)+1.0_8

enddo

do l=1,ncycle



   do i=1,ntime

      t=t+dt
```

```
      call rk4(q,p,qd,pd)

      do j=1,n

         q(j)=qd(j)
         p(j)=pd(j)

      enddo

   enddo


write(1,*)q(1),p(1)
write(2,*)q(2),p(2)
write(3,*)q(3),p(3)

do j=1,n-1

   write(4,*)t,dble(j),q(j+1)-q(j)+1.0_8

enddo

   write(4,*)


enddo


close(1)
close(2)
close(3)
close(4)

end


!!!subroutine!!!
```

```
subroutine rk4(q,p,qm,pm)

use com

real(8),intent(in)::q(1:n),p(1:n)
real(8),intent(out)::qm(1:n),pm(1:n)
real(8)::qd(1:n),pd(1:n)
real(8)::k1q(1:n),k1p(1:n),k2q(1:n),k2p(1:n)
real(8)::k3q(1:n),k3p(1:n),k4q(1:n),k4p(1:n)

call onestep(q,p,k1q,k1p)

do j=1,n

   qd(j)=q(j)+k1q(j)/2.0_8
   pd(j)=p(j)+k1p(j)/2.0_8

enddo

call onestep(qd,pd,k2q,k2p)

do j=1,n

   qd(j)=q(j)+k2q(j)/2.0_8
   pd(j)=p(j)+k2p(j)/2.0_8

enddo

call onestep(qd,pd,k3q,k3p)

do j=1,n

   qd(j)=q(j)+k3q(j)/2.0_8
   pd(j)=p(j)+k3p(j)/2.0_8

enddo

call onestep(qd,pd,k4q,k4p)
```

```
do j=1,n

   qm(j)=q(j)+(k1q(j)+2.0_8*k2q(j)+2.0_8*k3q(j)+k4q(j))/6.0_8
   pm(j)=p(j)+(k1p(j)+2.0_8*k2p(j)+2.0_8*k3p(j)+k4p(j))/6.0_8

enddo


end


!!!subroutine!!!


subroutine onestep(q,p,qq,pp)

use com

real(8),intent(in)::q(1:n),p(1:n)
real(8),intent(out)::qq(1:n),pp(1:n)

qq(1)=p(1)*dt
pp(1)=(q(2)-q(1))*dt
qq(n)=p(n)*dt
pp(n)=(q(n-1)-q(n))*dt

do j=2,n-1

   qq(j)=p(j)*dt
   pp(j)=(q(j+1)+q(j-1)-2.0_8*q(j))*dt

enddo


end
```

問題に与えられた初期条件のもとで 3 つの粒子の (位置 x、速度 v) を図 3、図 4、図 5 にしめす。

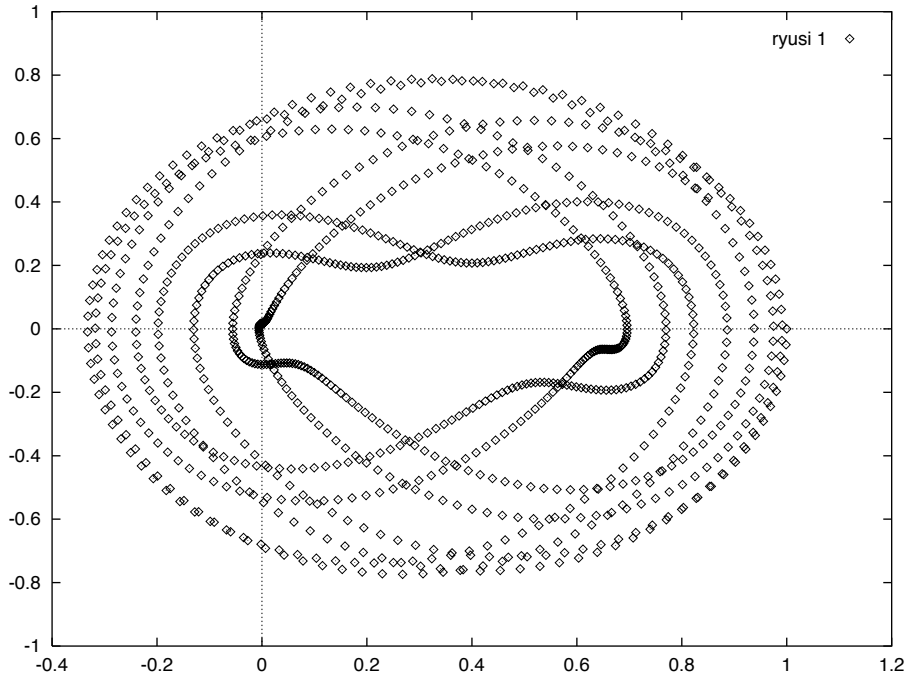

図 3: 粒子 1 の位置と速度のグラフ

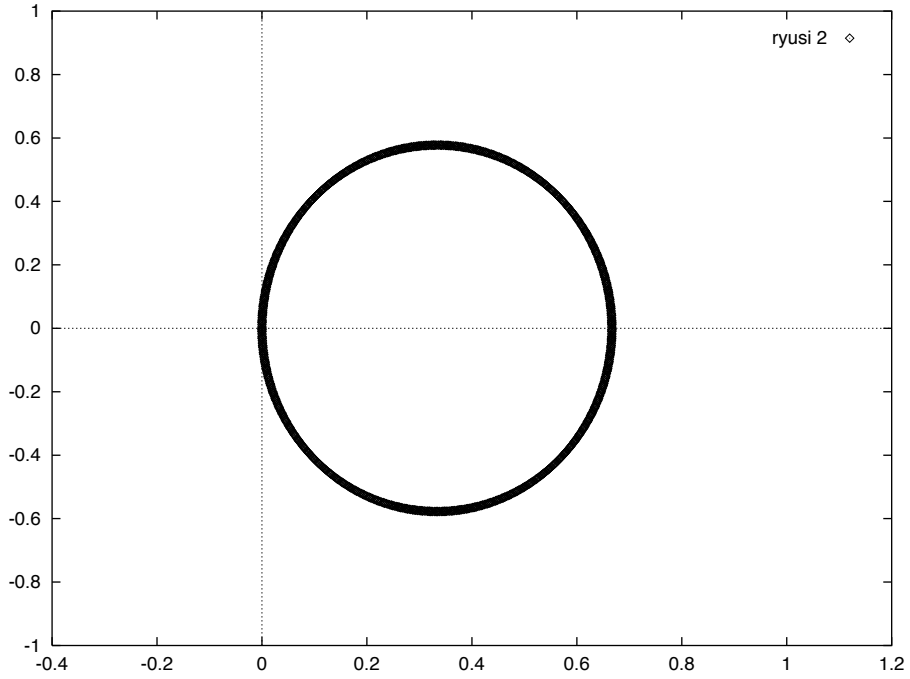

図 4: 粒子 2 の位置と速度のグラフ

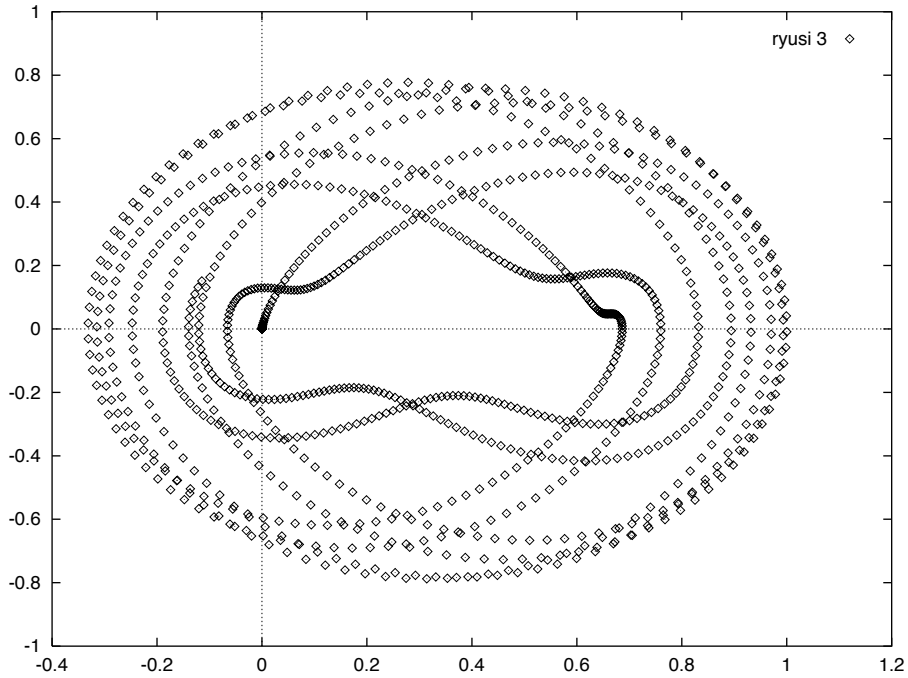

図 5: 粒子 3 の位置と速度のグラフ

(2) 初期条件を $x_1 = 1, x_i = 0, v_j = 0, 2 \leq i \leq 20, 1 \leq j \leq 20$ としたとき、時刻 50 までの $r_i (i = 1, 2, \cdots, 20)$ の時間発展を図 6 にしめす。ただし図 6 では $r_i + 1 (i = 1, 2, \cdots, 20)$ をプロットしてある。

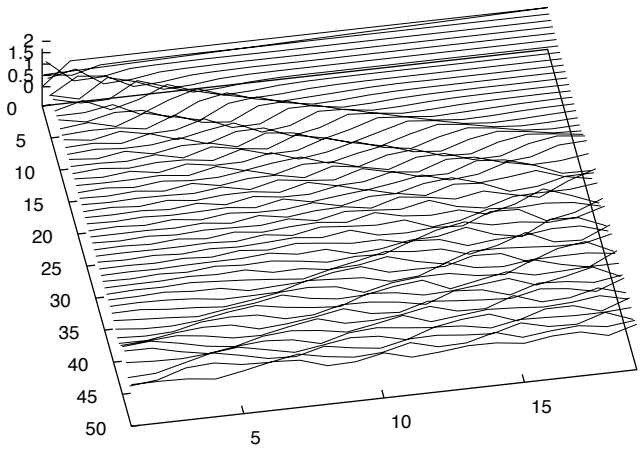

図 6: $r_i(i = 1, 2, \cdots, 20)$ の時間発展