Vol.2016-CE-134 No.20
2016/3/6

R による数値計算法

作花一志[†1]　江見圭司[†1]

概要：R プログラミングを用いた数値計算法について述べる．例として非線形方程式の解法，微分方程式の解法，３D 図形描画などを挙げる．

キーワード：R 言語，数値計算，非線形方程式，積分，微分方程式，３D 図形

Numerical Computing by R

KAZUSHI SAKKA　　KEIJI EMI

Abstract: Methods of numerical computing by using R programming are described. Some examples are presented such as differential and integral calculus , differential equations and 3D figures.

Keywords: I R Numerical Computing, Nom-linear Equation, Integral, Differential Equation, 3D Figures

1. はじめに

R 言語は統計解析用に開発された有名なオープンソース・フリーウェアである.Windows，MacOS，Linux など多種の OS に対応していて，自分の PC に簡単にインストールできる．R は非常にたくさんの関数，ライブラリを持ち，しかも毎月のように増補されている.グラフィックス機能が充実して 3D 描画も容易にできるので,シミュレーションに適している．そのため近年データサイエンスのツールとして注目され適用範囲は．画像解析，生命現象，経営，金融など多方面にわたり，それらのアプリケーションも多数開発されている．

この小文では大学初年度の数学である微積分学の学習への応用を試みる.それらの教科書のかなりの部分は極限値，微分，積分などの計算手法で占められ，特異な技巧を必要とするものも少なくないが，R を使えば簡単に求まるものが多い．また微分積分全般を修得してから学ぶ微分方程式の解なども短いステップで容易に解くことができる．

次節以降で，方程式の解法，微分，積分,微分方程式の解法，３D 図形の描画などについて述べ，最後にサンプルプログラムを載せた.アルゴリズムについてはこれまでに優良な参考書が名数出版されているので省略する.主に参考したサイトは[1]，[2]である．なお結果のカラー図などはウェブサイト[3]を参照されたい.筆者はこれまでフリーウェアを用いた数値計算・統計解析教材開発について述べてきたが[4][5],この小文もその延長上にある.

２　方程式の解法

非線形方程式　$x^4-4^x=0$ を解いてみよう．解析的方法では求まらないが，直感的に x=2 が解になることはすぐにわかり，また x=4 も明らかに解である．解はこの 2 つだけだろうか．これ以上は直観でも解析でもわからない．そこでグラフを描いて x 軸との交点を調べる．するともう一つ負の解があることがわかるが，これは数値的にしか求まらない.幸い R では uniroot という便利な関数が装備されていて，方程式 f(x)=0 の a<x<b の間にある解は

```
Sol<-uniroot(f, c(a, b))
```

で得られる．

この場合 $f(x)= x^4-4^x$ ，a=-1， b=0 であり，解は Sol$root で与えられ -0.7666825 である．

この方法は f(x) が連続関数であれば適用できるし,不連続点があっても有限個の場合は可能である．

整方程式の場合は x の係数を昇べき順にベクトルで与えて polyroot 関数により虚根も含め簡単に求まる．

２次方程式 $x^2-2x+3=0$ の解は

```
polyroot(c(3,-2,1))
```

より 1+1.414214i ， 1-1.414214i である．

†1 京都情報大学院大学
The Kyoto College of Graduate Studies for Informatics

ⓒ2016 Information Processing Society of Japan

Vol.2016-CE-134 No.20
2016/3/6

3 微分　極値

前節の関数の極値を求めよう．そのためには元の関数を微分しその導関数=0 という方程式を解くことになる．
関数 f1 とその導関数 f2 を求めるには

```
f1 <- deriv(~＊＊＊＊＊, "x", func=T)
f2 <-function(x) attr(f1(x),"gradient")
```

とすればよい．f1(0)より x=0 における関数値を，f2(0)より微分係数を求めることができる．
この２つの関数のグラフを同一座標に描いた．ただし導関数のグラフは破線で描いてあり，0 と 2 の間および 3 と 4 の間で x 軸と交わることがわかる．前節と同様にしてその解は 1.253741，3.434586 でありこの x に対する関数値が極値である．このように煩雑な計算をしないで容易に求めることができる．

なお関数を expression で定義し D を使うとその導関数が得られるが，関数形だけで関数値は求まらないのでグラフは描けない．

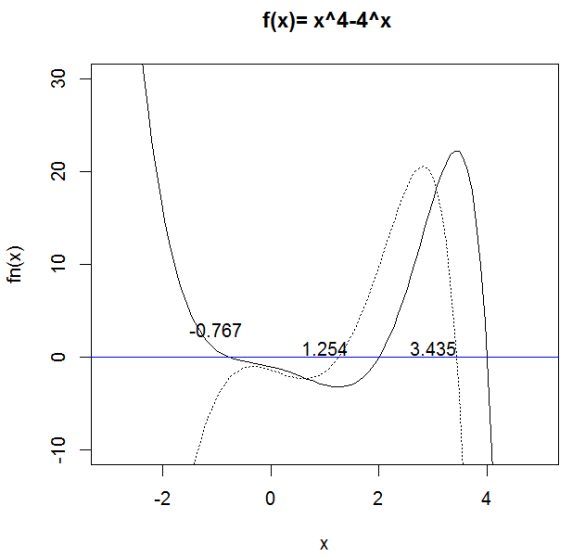

図 1　　関数のグラフと方程式の解
$f(x)= x^4 - 4^x$ のグラフは実線で，その導関数 f'(x) のグラフは破線で示す．
f(x)=0 の解，f'(x)=0 の解も記されている

4 積分

ある関数 f(x)を a から b までの定積分した結果は

```
f<-function(x)　******
integrate(f,a,b)
```

で得られる．関数としては初等関数で表されるものなら何でもよく，また積分の上限下限として Inf（∞）も使用できる．

表 1　定積分

f a,b	sin(x) [0,1]	1/x [0.001,1]	exp(-x) [1,∞)	$2/(1+x^2)$ [-1,1]	√x/cos(x) [0,1]
解析値	1－cos(1) 0.459698	－ln(0.001) 6.907755	1/e 0.367879	π	?
R	0.45969	6.907755	0.36787	3.14159	0.86417

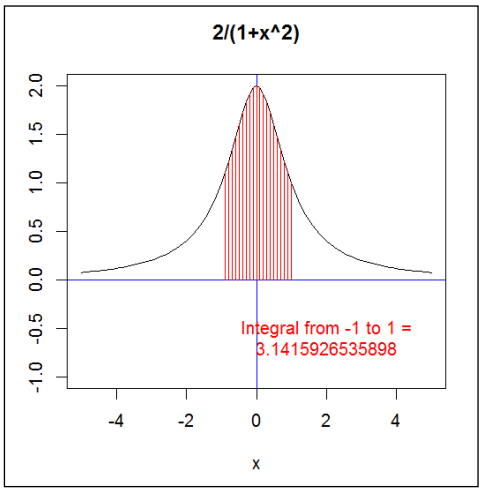

図 2　$2/(1+x^2)$ を -1 から 1 まで積分

ある関数 f(x)とその原始関数のグラフを同一座標に描くことはできないものか？
f(x)を 0 から t までの積分しさらにその値をプロットする関数を作り，t をある値からある値まで動かしてみよう．
下記は f(x)=cos(x)を 0 から t まで積分した値を y とし，(t,y)をプロットする関数を f1 として，t を 0.02 刻みで－2 から 10 までプロットするプログラムの結果である．この場合原始関数は当然 sin(x) である．

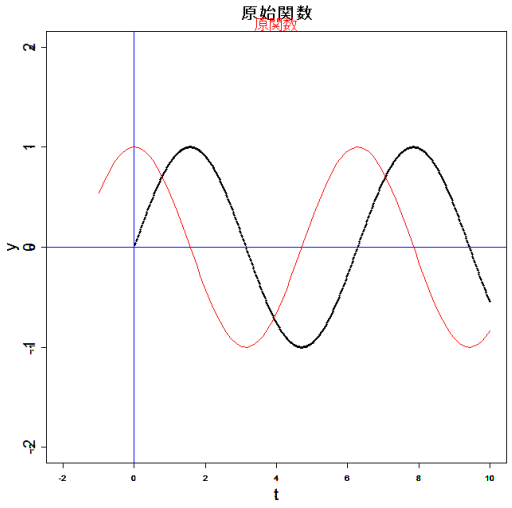

図 3　cos(x)とその原始関数のグラフ
原始関数は太線で描かれている．

ⓒ2016 Information Processing Society of Japan

一般に原始関数は存在しても初等関数では表されないことが多い．しかしこの方法だとほとんどの関数の原始関数は関数形はわからなくてもグラフは描くことができる．

5 微分方程式の解

微分方程式を解くとは x，y，y’，y” などを含む方程式から y を x の関数として表すことで，微分積分学修得の後で学ぶのが通例である．その起源はニュートンの運動方程式に始まり，これまでさまざまな解法が研究されているが，解析的方法は一般に非常に難解・技巧的であり，数値的にしか解けない場合も多い．

ところが R には ode と言う便利な関数が装備されていて，y’と初期値を与えれば，非線形でも連立でも高階微分の場合でも容易に解くことができる．

まず dfn で微分方程式 y ’ を定義する．times では 0 から 5 まで 0.1 刻みで y，y’ の値を計算し out という matrix に収める．out の第 1 列，第 2 列は t と y で，head でその最初の 3 行だけをコンソールに出力される．

```
      time        1
[1,]   0.0 2.000000
[2,]   0.1 1.760983
[3,]   0.2 1.560480
```

また plot によりグラフを描いたものが図 4 である．

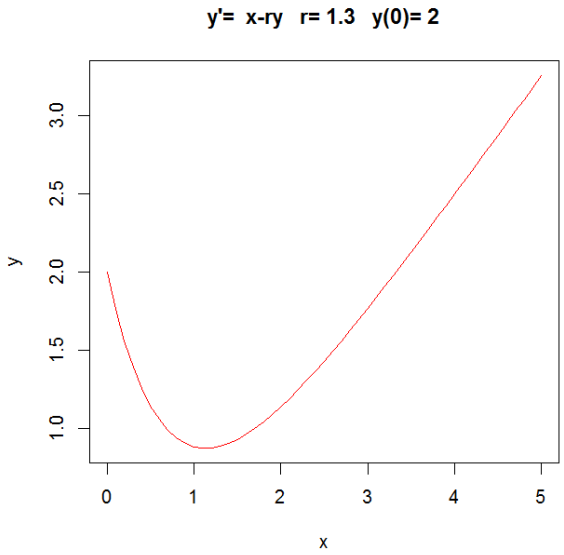

図 4　1 階微分方程式の解

2 階微分方程式の場合は y→y[1] ， y'→y[2] と置換すると y''=y[2]' だから 2 元連立 1 階微分方程式に変換される．

out の各列は t，y，y' であり，前述のようにその最初の 3 行だけをコンソールに出力し，また plot により値をプロットした．図 5 は y”+y=0　y(0)=0　y’(0)=1 の解を図示したもので実線は y，破線は y ’ の値を表している．なお解析解は y = sin(x) である．

```
      time        1         2
[1,]   0.0 0.00000000 1.0000000
[2,]   0.1 0.09983342 0.9950042
[3,]   0.2 0.19866933 0.9800666
```

なおこのプログラムにはパッケージ deSolve をインストールしておく必要がある．

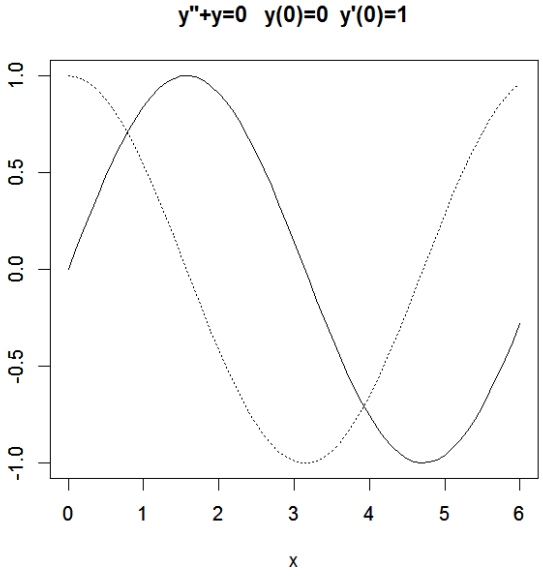

図 5　2 階微分方程式の解
実線は y，破線は y’

下図は有名な非線形 3 元連立微分方程式であるローレンツの微分方程式の結果で out は 10000 行 4 列の matrix となる．

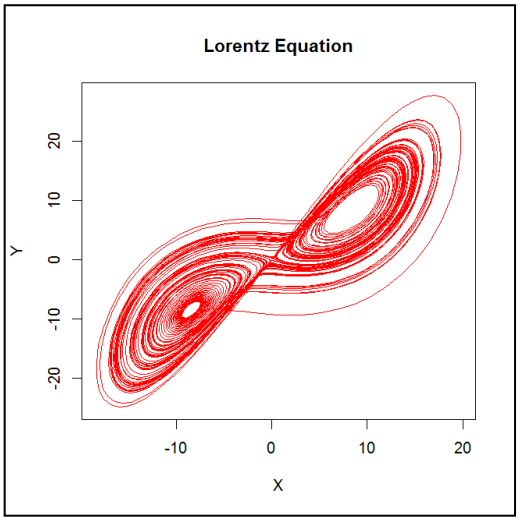

図 6　ローレンツの微分方程式の解

Vol.2016-CE-134 No.20
2016/3/6

６　３D 図形

z = f(x,y)　をプロットするにはライブラリ fields と rgl をインストールしておくとよい．関数 f を定義し

f <- function(x, y) sin(x^2+y^2)

x,y の範囲を指定し

x <- seq(-3, 3, length = 60);y=x

z <- outer(x, y, f)

persp(x，y，z，オプション)　で図 7 を描くことができる．

persp3d(x，y，z，オプション) を使えば描かれた図をマウスドラッグで拡大縮小回転できる．

さらに play3d(spin3d(c(xx，yy，zz), 速度))で自動回転させることができる．ただし xx, yy, zz は回転軸ベクトル．

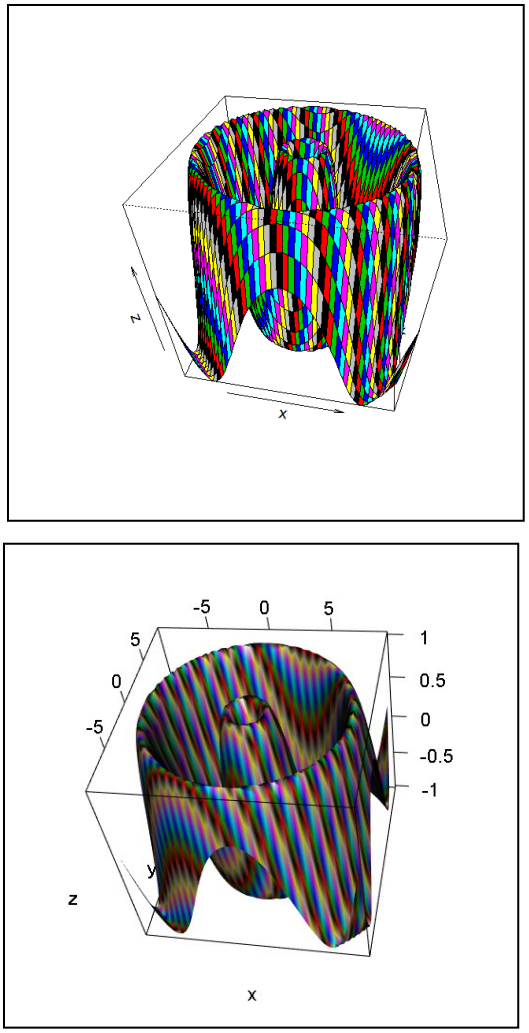

図 7　　$z=x^2+y^2$

上図　persp で描かれた

下図　persp3d で描かれた

R の作業フォルダーの下に WG というフォルダーを作り

writeWebGL(dir="WG",width=700,height=400)

命令を加えると WG の中に index.html が作られこの画像をブラウザで閲覧できる. しかもマウスで伸縮･回転できる.

正の曲率をもつ３D 図形である球面，楕円面は容易に想起できるし描くこともできる．負の曲率をもつ３D 図形を描くのは難しいが，この機能では双曲面 $z=x^2-y^2$　（鞍形またはポテトチップス形）と小球 3 個を重ねて描ける．

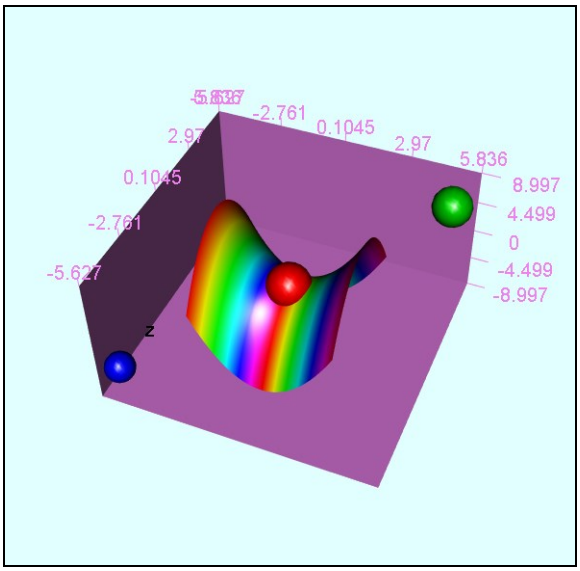

図８　　$z=x^2-y^2$　と３小球

x,y,z 値として数式ではなく数値を与えることもできる．volcano はニュージーランドの火山の高さのデータで R にデフォルトに装備されている．このデータを persp3d で描くと下図が得られる．

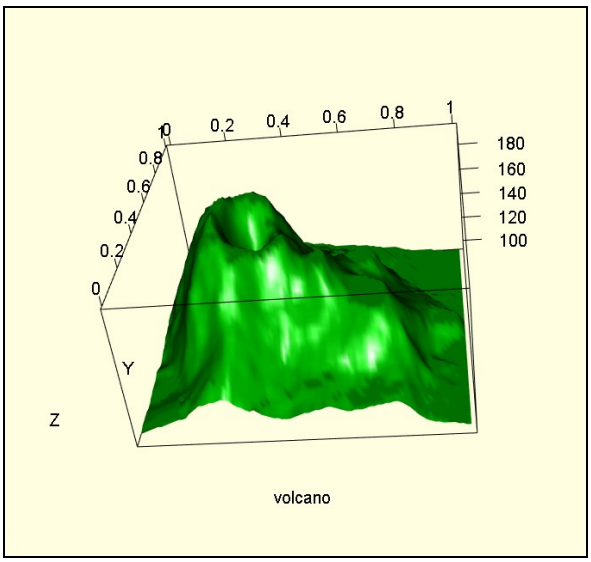

図９　　数値データ valcano　の表示

ⓒ2016 Information Processing Society of Japan

7．プログラムコード

```
#######　方程式　#############
fn <- function(x) x^4-4^x
curve(fn,xlim=c(-3,5),ylim=c(-10,30))
abline(h = 0,　col = 4)
Sol <- uniroot(fn, c(-2, 0))
text (-1,3,round(Sol$root,3))

#######　微分　極値　#############
f1=deriv(~x^4-4^x,"x",func=T)
f2=function (x) attr(f1(x),"gradient")
curve(f2(x),-3,5,add=T,lty=3)
Sol <- uniroot(f2, c(0,2))
text (1,1,round(Sol$root,3))
Sol <- uniroot(f2, c(3,4))
text (3,1,round(Sol$root,3))
title(main=" f(x)= x^4-4^x")
f1(Sol$root)                    #極小値
##########
f1=deriv(~cos(x)/sqrt(1+x^2),"x",func=T)
f2=function (x) attr(f1(x),"gradient")
curve(f1(x),xlim=c(-8,8),ylim=c(-2,2))
curve(f2(x),xlim=c(-8,8),ylim=c(-2,2),add=T,lty=3)
abline(h = 0,　col = 4)
title(main=" f(x)= cos(x)/sqrt(1+x^2)")

############　積分　############
f <-function(x) cos(x)
f1 <-function(t,x1,x2) {
        y=integrate(f,0,t)$value
        par(cex=0.5)
        plot(t,y,xlim=c(x1,x2),ylim=c(-2,2),pch=20)
      }
x1=-2;x2=10;h=0.02;n=x2/h
for (i in 0:n) {
  t=h*i;f1(t,x1,x2);par(new=T)
}
abline(h=0,col=4);abline(v=0,col=4)
par(cex=1.0)
curve(f(x),-1,10,col=2,add=T)
title("原始関数")
mtext("原関数",3,0,col=2)
```

```
############ 微分方程式##########
library(deSolve)
###　1 階微分方程式の数値解法
#　 y’=x-ry　　y(0)=y0　　r, y0 はパラメータ
r=1.3;y0=2;fm="x-ry"
    dfn <- function(x, y, parms){list(x-y*r)}
times <- seq(from = 0, to = 5, by = 0.1)
out <- ode(y = y0, times = times, func = dfn,parms=null)
head (out, n = 3)
plot(out[,1], out[,2], col=2 ,type="l",xlab="x",ylab="y")
title(paste("y'= ",fm,"　r=",r,"　y(0)=",y0))

# ##2 階微分方程式の数値解法
#　　y''+y=0　　y(0)=0　y'(0)=1
#　　y→y[1] y'→y[2] とおくと　y''=y[2]' だから
#　　y[1]'=y[2]
#　　y[2]'=-y[1]
y0 <- c( 0, 1)　　# 初期条件　t=0 にて y=0 v=1
df <- function(t, y, parms) {
  dy1 <- y[2]
  dy2 <-　-y[1]
  list(c(dy1, dy2))}
times <- seq(from = 0, to = 6, by = 0.1)
out <- ode (times = times, y = y0, func = df, parms = NULL,
method = rkMethod("rk45ck"))
head (out, n = 3)
plot(out[,1], out[,2], lty =1,type="l",xlab="",ylab="")
par(new=T);plot(out[,1],out[,3],lty=3　,type="l",xlab="x",yla
b="")
#mtext("y",2,-1,col=2);mtext("y'",4,-1,col=3)
title("y"+y=0　　y(0)=0　y'(0)=1")

############ 3D 図形 ##############
library(fields)
f <- function(x, y) sin(sqrt(x^2+y^2))
x <- seq(-9, 9, length = 50)
y =x
z <- outer(x, y, f)
persp(x,y,z,theta=10,phi=40,axes=T,　col=1:8)

#### mouse ##########
library(rgl)
par3d(windowRect=c(300,200,800,700))
persp3d(x,y,z,col=1:8)
### spin ###
play3d(spin3d(c(1, 1, 0), 10))　# spin start　stop ESC
```

ⓒ2016 Information Processing Society of Japan

```
########## 双曲面　###########
library(rgl)
 f <-　function(x, y) x^2-y^2
 x <- seq(-3, 3, length = 60); y =x
 z <- outer(x, y, f)
par3d(windowRect=c(100,100,700,700))
rgl.bg(color=c("lightcyan"))
persp3d(x, y, z,　expand = 1.0, col = rainbow(30))
####　3 小球 ####
rgl.spheres(0,0,0,1.8,color=rainbow(2))
rgl.spheres(5,5,5,1.6,color=3)
rgl.spheres(-5,-5,-5,1.2,color=4)
rgl.bbox(color="violet")

###　　WGL ###
writeWebGL(dir="WG",width=700,height=400)

### volcano ###
par3d(windowRect=c(100,100,800,800))
rgl.bg(color=c("lightyellow"))
persp3d(volcano,col = 3,aspect = c(0.8,0.8, 0.4))
play3d(spin3d(c(1, 1, 0), 10))　# spin start　　stop ESC
```

８．おわりに

R を使って統計解析，データマイニングを学習する図書やウェブサイトは多数あるが，線形代数や微積分について記されたものは少ない.この小文では微積分の演習問題解法を述べたが，次は行列，一次変換，固有値問題などについてまとめる準備をしている.さらにカオス・フラクタル図形，地理情報，天文シミュレーションなどにも適用していく予定である.

R の学習においては他のプログラミング言語のように文法を理解してから例題・演習問題を解くという順序はとらずに，数学的内容を理解してからサンプルプログラムを実行していく方法を勧める.疑問が出たらその都度参考書を開けばいい.C や Java に比べ文法自体は簡単だが, 細部にこだわると本来の目的を見失う恐れがある.R は発展途上で現在 web との関連は希薄だが，第 6 節の末尾に記した WebGL 機能は最近公開されたもので将来の発展が期待される.

参考文献

[1] http://cse.naro.affrc.go.jp/takezawa/r-tips/r.html
[2] http://itbc-world.com/home/rfm/home/
[3] http://web1.kcg.edu/~sakka//num/R/web/index.htm
[4] **作花一志** フリーソフトを用いた数値計算～Scilab のすすめ
NAIS.J. Vol.9,pp.34-37,(2014)
[5] **作花一志　胡明** R による数値計算と統計解析
NAIS.J. Vol.10,pp.61-69,(2015)

ⓒ2016 Information Processing Society of Japan