X.5 R(S,S-Plus) to GAUSS の基礎　　ver. 0.1

ここでは、リソースの制約の関係からＲについて取り上げます。Ｒは特に Mac で動作する完全な言語体系なので関係者の方は注目してください。マニュアルの日本語化、統計学者の広範な活用、ソースコードの完全公開などと、単にクーロンの領域を越えた「良心をもった」研究開発用の無料ソフトウエアと言えます。

これまで取り上げた諸言語との根本的な違いは、代入格納する意味を表すのに＜－の矢印を２文字で表現することです。方向は左右どちらでもいいのですが、とにかく代入ということが色濃く表されます。また、Ｒは、言語として、小文字大文字の区別および行の概念があり、複数行にまたがるプログラムの場合、冒頭に自動的に＋のマークがつきます。

Ｒのプロンプト

＞	通常の入力待ち
＋	複数行にわたる印（２行目から）

代入と大小文字の区別

R	GAUSS
a<-1; A<-2	a=1;

Ｒでのセミコロンは、複数の命令を同じ１行に書く場合のみに必要であって、通常の 1 行ごとの入力では特に必要ではありません。GAUSS では命令ごとに必須です。変数に入っている内容を表示するのには、ただ単に変数を置くだけか、print( )形式で表示できます。

画面表示

R	GAUSS
print(a) または a	print a;

Ｒは丸括弧系の言語です。通常使用では、変数（オブジェクト）をそのまま置いて表示させることを行ないます。一方 GAUSS はスペースを入れて直接続ける系統の言語です。変数 a の中味を表示するのではなくて、OK という文字列を表示するのには、Ｒでは丸括弧の中にで文字列を包みます。一方、GAUSS では二重の引用符” “で包みます。エラーログで abort する場合も同様な違いがあります。なおコメント行は R では＃で示します。

コメント行

R	GAUSS
# This is a comment.	/* This is a comment. */
k<-1 # This is a comment.	k=1; @ This is a comment. @

行列表記

R	GAUSS
a<-c(1,2,3)	a={1 2 3};
b<-matrix(c(1,2,3,4,5,6),2,byrow=TRUE)	b={1 2 3,
b<-matrix(c(1,2,3,4,5,6),2)	4 5 6};

GAUSS では、まず行列とベクトルの区別はありません。というよりも、スケラーも１×１の行列として基本的に認識されます。GUASS では｛　｝の中に行列の内容を入れます。カンマで区切られていたならば、そこまでが１行目です。一方Ｒでは、ベクトルを代入するには c( )という関数を使って、丸括弧の中にカンマで区切って代入します。この際のカンマは必須です。なければシンタックスエラーになります。Ｒではこのカンマは通常の数学で用いるような単に数値と数値を区切る記号です。一方、GAUSS ではカンマが常に「そこまで１行」という意味を持ちます。Ｒでは、行列の要素を直接代入する方法に、matrix( )関数を用いて、そこにベクトルで c(1,2,3,4,5,6)などと数値を入れた後に、行数を指定する形をとるのが１つの方法です。GAUSS では、｛　｝の中に行列の内容を１行ごとにカンマをつけて代入します。

配列表記

R	GAUSS
b[2,3]	b[2,3]

Ｒも GAUSS も配列は四角括弧で表すことができます。２行３列目なら b[2,3]とすることは同じです。なお、Ｒでは Array などの機能を用いると３次元以上の配列を扱うことができます。GAUSS では計量データの次元だけを考えるので、１次元か２次元のみです。

サブ行列

MATLAB	GAUSS
b[1:2,2:3]	b[1:2,2:3]

RもGAUSSもコロンで「何々から何々まで」を表します。同様に扱えます。

行列のワイルドカード

R	GAUSS
b[,2]	b[.,2]

Rでも GAUSS でもカンマで区切って行または列を指定することは同じです。ただし、RではGAUSSのようにドットはつけません。ワイルドカードはただ単に何も書きません。

行列のマージ

R	GAUSS
cbind(b,c)	b~c
rbind(b,c)	b|c

GAUSSでは、～は水平方向、｜は垂直方向のマージを表すが、Rではcbindでcolumn方向のマージを、rbindでrow方向のマージを関数形の形でする。

組込み関数

R	GAUSS
x<-c(1,2,3,4,5,6,7,8)	x={1,2,3,4,5,6,7,8}
m<-mean(x)	m=meanc(x);
s<-sd(x)	s=stdc(x);
sm<-sum(x)	sm=sumc(x);

cs<-cumsum(x)	cs=cumsumc(x);

組込み関数（違いがあるもの）

R	GAUSS
n<-length(x)	n=rows(x);
x1<-log(x)	x1=ln(x);
x2<-log10(x)	x2=log(x);

Ｒではベクトルにはっきりした列と行の区別はありませんが、その長さを測るときには、length( )を用います。Ｒでは log は e がベースです。GAUSS では ln を使います。

シークエンス数列

R	GAUSS
i<-1:10	i=seqa(1,1,10)'
j<-seq(1,2,0.1)	j=seqa(1,0.1,11)'

Ｒでは、コロンを用いて、例えば１から 10 までは１：10 として１から 10 までの数列を作成します。また、ステップを指定した数列にするには、seq( )を用いて、始点、終点、ステップ幅の順に指定します。Ｒではベクトルには列と行の区別は特にありません。GAUSS ではその点厳格です。数列は列としてでてきます。１から２までの 0.1 ステップの数列を作成するには、seqa で始点、ステップ幅、個数の順番で指定します。ポールと距離の関係の問題と同じように個数は 10+1=11 個になりますので GAUSS では注意してください。

関数の作り方

R	GAUSS
simpleols<-function(y,x) {	proc(3)=simpleols(y,x); local b,e,s2hat,varb,se,t;
. . . . .	. . . . .
return(b)	retp(b,se,t);

}	endp;

Ｒではｃライクに｛　から始まり　｝で終わります。GAUSS では endp;でその代わりを果たします。リターンがない内部での表示やライブラリや関数の実行がＲではよくなされますが、リターンがある場合には return( )で対応します。GAUSS では複数リターンもあって、その数は０の時も冒頭で指定しなければいけません。（ただしリターン１個の場合は省略して proc simpleols(y,x);という具合になる。）

条件分岐

R	GAUSS
if (b>1)	if b>0;
y<-1	y=1;
else	else;
y<-0	y=0;
	endif;

Ｒでは条件文は丸括弧の中に書く。さらに条件が続く場合には else if で対応する。なお、GAUSS では丸括弧の代わりに条件文の終わりにもセミコロンが必要なことに注意されたい。また、endif;も厳密に必須となる。

ループ

R	GAUSS
k<-0	k=0;
for (i in 1:10) {	for i (1,10,1);
k<-k+i	k=k+i;
}	endfor;

上の例は for ループのケースである。この他に同じようなやり方で while (条件文) {　}がＲにはある。注意すべきは｛　｝で最初と最後を包むことである。GAUSS ではその代わりに endfor;または endo;があります。

プロット

R	GAUSS
	library pgraph; graphset; title(“xy plot”); xlabel(“t”);
plot(x,y)	xy(x,y);
plot(x,y,main=”xy plot”, xlab=”t”, ylab=”A”)	

R は直接 plot を内部で呼び出せます。GAUSS は、ライブラリーで呼び出す外部式です。したがって、GAUSS では library の呼出しで、pgraph を呼んできて、そのグローバル変数を初期化するため graphset;としなくてはなりません。なお、Rではグラフのラベルをつけるには、関数内部で main および xlab および ylab にそれぞれ文字列を割り当てます。

この他にもRには美しい文法体系とライブラリが標準で装備されています。分布関係が網羅されているだけでなく、豊富なライブラリ群、特に最適化のアルゴリズムは GAUSS とはまた違う観点から作られていて興味を引かれます。本章とは別に、さらに GAUSS でできないことをRにさせて、GAUSS とRとの融合をめざした章または節を、将来的には、ご説明したいと思います。