1.2 外部ファイルの出入力　　新バージョン改訂版 ver. 0.3

これまでは、プログラムの結果を画面に表示プリントすることをやってきました。たいていの場合、画面に出た結果をコピー＆ペーストしてやれば、レポートは作成できます。もちろん、それだけでも十分なのですが、プログラムの構造をしっかり学ぶために結果をファイルに直接出す方法を学びましょう。

プログラム

```
cls;
output file = D:/sample2.txt reset;
print “Howdy,folks.”;
print “How is it going? ”;
/*          without return          */
print “Howdy,folks.”;;
print “How is it going? ”;
```

前と同じプログラムですが、２行目にアウトプットファイルを指定してみました。＝のマーク以下がディレクトリーを含むファイル名となっています。そして、スペースをあけて、reset というオプションをつけました。この一連の命令の意味は、アウトプットファイルを D:ディレクトリーの sample2.txt というファイルに定義して、これまでそのファイルに何か書かれていれば消去してリセットするということです。もし、ファイルが D:ディレクトリーの中の gauss というフォルダーにあるとすれば、この行は、

```
output file = D:/gauss/sample2.txt reset;
```

となります。さらに深いレベルのフォルダーに保存する場合には、/　/を続けていくことになります。なお、日本語フォントでは、GAUSS からファイルの位置を確認した際に、/のマークの代わりに、¥のマークに置き換わっているかもしれませんが、プログラムはスラッシュ/で統一して方が現代の Web 社会では有効かもしれません。Reset は普通のオプションと違って、付けないと何もアウトプットされないばかりか、ファイル名さえもつくられませんのでご注意を。オプションは普通はスラッシュ/で添えられることが多いのですが、この場合は、フォルダーのレベルの表示と混同しないために、/reset とはなりません。なお、ログファイルをつくるように、前のものを消さずに、アウトプットをつけ足していく場合には、

```
output file = D:/gauss/sample2.txt on;
```

と on としてやれば、アウトプットが積み重なっていきます。前の結果を消したくない場合 on は威力を発揮します。一時的にアウトプットを止めるには、on のところを off にします。

ポイント　出力ファイル定義は、output file = [**詳しい位置表示つきファイル名**] reset;
ディレクトリーには X:**の形式、その後のフォルダーはスラッシュ/で表す。**

最新バージョンでは X:のところの大文字小文字の区別は存在する。先頭のスラッシュも必ず必要になりました。もちろん、バックスラッシュのアジアフォントの￥でも同じことです。

画面表示もアウトプットファイルと同様にオンオフのコントロールができます。

プログラム

```
cls;
screen off;
output file = D:/sample2.txt reset;
print “Howdy,folks.”;
print “How is it going? ”;
/*              without return              */
print “Howdy,folks.”;;
print “How is it going? ”;
```

といった具合に、cls でスクリーン消去をした次にでも、screen off の命令を入れてやりましょう。画面出力は停止します。ただし、気をつけておきたいことは、１度 off にしたものは、screen off の命令を消しても、有効であり続けてしまうことです。元通りに表示されるようにするには、screen on;とあらためて書いてやらなければなりません。何もしないと、規定値は、screen on のままになっています。

ポイント　画面表示を停止したいときには、screen off**。元にもどすには、**screen on**。**
**　　１度** off **にしてしまうと、あらためて** on **にするまで画面表示はされない。**

さて、少しはやいかもしれませんが、実際のデータファイルを外部から読み込んで、１つの変数の中に格納することをやってみましょう。まだ行列の話についてはふれていませんから、行列の話がでてきてから、ここをもう１度読みなおしてみてください。

今、datafile1.txt というファイルが D:ドライブにあるとします。ファイルはＷＥＢからダウンロードした上で、D:ドライブにそのままテキスト形式で保存してください。そのデータファイルに入っているデータ行列をそのまま GAUSS 上の変数に格納してみましょう。

プログラム

```
cls;
load data1[29,2] = D:/datafile1.txt;
print data1;
```

ポイント　M行N列のデータファイルを読み取るには

load **変数名**[M,N] = **ファイル名**

例　load abc[100,3] = D:/gauss/datafile1.txt;

ポイント　変数の内容を画面に表示するには、　print **変数名**

画面表示

```
     +DEN          +DEN
493.00000     8.2000000
410.00000     7.3900000
        .
        .
        .
263.00000     4.4100000
549.00000     8.6000000
267.00000     4.7100000
```

となるはずです。０がいっぱいついていますが、とりあえずは気にしないでください。一番上の行とその次の行、それから一番最後の行を確認してください。１行目は変数のラベルがついていたはずですが、なにか違う文字に置き換わっています。２行目（データのはじめの行）は同じです。そして、最後の行も一致しています。とりあえず、うまくデータ行列が data1 という変数の中に取りこまれました。

load data1[29,2] のところの意味は、D:ドライブにある datafile1.txt というデータファイルの内容を、data1 という GAUSS 上に自分で作った data1 という２９行２列の行列変数の中に格納するということです。[　,　]の中の数字は、データのラベル行も含めた行と列の合計の数をそれぞれ入れます。このままでは、ラベルの行に名前を勝手につけられて、計算などに持ちこめません。そこで、２行目から２９行目までを取り出して、新たな変数に格納するという作業をします。

print 文は、いままでは、”　“の中の文字を表示するものでしたが、もちろん、このように、print のあとにスペースをあけてそのまま変数を書くことによって、その変数の内容を画面に表示することができます。変数が行列の形をしていれば画面表示は行列ですし、変数が１×１の単なる数であれば画面表示は１つの数字です。GAUSS は、行列計算用の統計関数に富んだ C 言語のような性格があって、画面表示もそうですが、変数計算や操作も、行列をスケラーとかわりのないように取り扱えるようになっています。

プログラム

```
cls;
load data1[29,2] = D:/datafile1.txt;
```

```
data2=data1[2:29,.];
print data2;
```

こうすると、２行目から２９行目まで data1 からデータを取り出して data2 という変数に格納することができます。data1[2:29,.]の 2:29 は二行目から２９行目までという意味。:は何から何までという記号です。カンマ,の後のドット.はすべてということを意味します。ここで注意してほしいのは、これまで幾度かイコールの印＝がでてきましたが、プログラム上のイコール＝の意味は、「後ろの変数を前の変数に格納する」または「後ろの変数で前の変数を定義する」ということであって、「前の変数が後ろの変数と等しい」ということではありません。後ろのものを前に代入、格納、あるいは定義するという意味であるということをきっちりとおさえておいてください。つまり、data1[2:29,.] = data2;と書くと完全な論理ミスということになります。後ろを前にいれるという記号＝の意味は、プログラム言語に共通の決まりですので、通常のイコールの意味とは区別していきましょう。特に、その意味がはっきりするのは、前と後ろの変数が同じという下の例のようなプログラムの場合においてです。

ポイント　格納された行列データのU行目からD行目までを利用するには
変数名 = 変数名[U:D, .]
記号：はU［から］Dということを表し、記号 . は「すべて」の列を表す。
なお、変数名は前後同じであってもかまわない。
例　load abc[100,3] = D:/gauss/datafile1.txt;
abc = abc[2:100, .];

なお、面倒であれば、データファイルの１行目を切りとって１行まるまる上にあげてしまったデータを使うことも原始的ですが、賢明かもしれません。その場合には、単純にプログラム

```
cls;
load data3[28,2] = D:/datafile1.txt;
print data3;
```

２９行から１を引いて、２８行２列の data3 という変数をつくることができます。３０行目以降になにか書かれていても、たいていの場合には、上の２つのやり方の両方とも通用します。ただし、行数と列数がわかっている場合にかぎります。もし、１行目にラベルではなくて、なにかファイル名とか出所とかが書かれてあって、列数に文字のかたまりの数が一致してない場合には、手作業で原始的に切り取ってください。

ただ、列数は簡単にわかっても、行数は多すぎて数えられなかったり、数えるのが面倒である場合もあります。そんな時には、少し裏技を使って行数を計算する方法があります。

プログラム

```
cls;
load data4[ ] = D:/datafile1.txt;
print data4;
r=rows(data4);
print “total=” r;
```

のように行列を[ ]と空白にしてやると、GAUSS ではデータをｒ×１の列ベクトルとして格納されます。これを利用して、組込み関数の rows( )を用いて、変数ｒに data4 の行数を格納して、それを印刷すれば、全体のラベルも含んだデータ数がわかります。これを列数で割れば、実際のラベルを含む行数が計算できます。この場合、total= 58 になるはずですから、これを列数２で割ると、２９がラベルも含んだ行数になります。こうした補助プログラムで、あらかじめ行数を計算した上で、先にやったプログラムで M 行 N 列データを取りこむ方法を行えばよいことになります。

組込み関数を用いた行列のくわしい操作方法については、あとでくわしく章をもうけて説明することにします。なにかすでに決まった名前がついている組込み関数というのがあって、その括弧の中に変数を入れると、値がでてくるというくらいに思ってください。組込み関数には、そのほかに逆行列をとったり、転置をしたり、ほとんどの行列操作ができるほか、スケラーを入れて統計値が出てくるもの GAUSS には数多く登録されています。繰り返しになりますが、r=rows(data4);のところでもわかるように、後ろの組込み関数で出てきた変数を前の変数 r に代入格納しているのであって、ここでの＝のマークはいわゆる我々が使うイコールではないことに注意してください。あくまでも、後ろのものを前へというのが＝のマークの基本となります。

最後に、print 文は” “の中に説明する文字やイコールのなどのマークを入れて、その後に、スペースをおいて変数を置くことによって、文字やイコールの後に変数の内容が画面表示されます。もちろん、アウトプットファイルを on または reset に設定していると、同じ結果がファイルにも出力されます。

ポイント　文字と変数を同時に表示するには、　pirnt “文字” 変数名
ポイント　組込み関数を使うには、　組込み関数名(変数名)
のように組込み関数のあとにある( )の中に変数を入れる。
ポイント　＝のマークの意味は、後ろの変数（または計算結果）を前の変数に代入格納。

その他に、データ数が少ない場合には、直接プログラム上で変数に行列の値を代入格納することもできます。例えば、５×３の下のような行列であれば

```
 1   2    3
 4   5    6
 7   8    9
10   11   12
13   14   15
```

プログラム

```
cls;
data = {1   2   3,
        4   5   6,
        7   8   9,
        10 11 12,
        13 14 15};
print “data=” data;
```

というように、{　　} の中に、スペースをあけて、各行の終わりにはカンマ，のマークをつけて行列の内容を書き入れれば、手軽に行列の内容をプログラム上で書き入れることもできます。画面表示の結果は、０が多数ついた上の数字が５×３に並ぶはずです。なお、GAUSS には行の概念がないですから、data = {1 2 3,4 5 6,7 8 9,10 11 12,13 14 15};
と書いても上と同じ５行３列のデータが data という変数の中に格納されます。したがって、注意してほしいのが、data = {1,2,3,4,5};とプログラムすると、変数 data の内容は、１×５の行ベクトルではなくて、５×１の列ベクトルになることです。この点は、特に行列の扱いについては GAUSS では厳密ですから、他の数学ソフトも使用している方は特に注意してください。一部の日本の識者の間には、GAUSS はベクトル計算に融通性があまりないので使いづらく他の数学ソフトから乗りかえられないという意見もありますが、それは正しくはありません。GAUSS は厳密な行列言語です。行ベクトルにすれば、行ベクトル。列ベクトルにすれば列ベクトル。計算をする際に自動的に置き換わるという厳密でないことまでは GAUSS はしません。実際のデータが列で縦にならんでいるということを念頭において、GAUSS の行列は、ごく自然に表現されているのです。最後に、ファイルの入出力でててくる/　/のフォルダーの深さのレベルからくる困難さとプログラムの本質には何の相関関係はありません。もし複雑なレベルにユーザーフォルダーが設定されている場合には、指導者は、くわしく説明するか、より簡単なディレクトリーに設定しなおしてください。

バージョン相互の注意事項

以下の章では、バージョン３時代に書いた原稿が多く含まれているので、最新５のバージョンでは、次のように変更してファイルの入出力をしてほしい。

（旧バージョン）

load data1[29,2] = d:datafile1.txt;

（新旧両バージョン）

load data1[29,2] = **D:/**datafile1.txt;

または

load data1[29,2] = **D:¥**datafile1.txt;

新バージョンからはドライブ名が大文字の場合、GAUSS 外部では大文字小文字の区別があるようで d:ではいけない。大文字にする必要がある。また、従来まではそのドライブのトップの場合、/または¥のしるしなしに直接書けたが、新バージョンからはこれがないと動かないでエラーを発する。（無論、上のような書き換えをしなくとも、GAUSS の Working Directory を変更してファイルのある場所と同じにすればディレクトリは上書きされて無視されるのでエラーなしに動くはずである。）