


Appendix A

INTRODUCTION TO GAUSS

The purpose of this appendix is to give a quick introduction to GAUSS. For more complete information on GAUSS, see the GAUSS manual.


A.1 Starting and Exiting GAUSS



A.1.1 The Windows Version

For the Windows version of GAUSS, click the icon  for GAUSS and you will be in the COMMAND mode of GAUSS. In the COMMAND mode, you can execute screen-resident programs.


To exit GAUSS, click  at the right upper corner.

A.1.2 The DOS Version


For the DOS version of GAUSS, from the DOS prompt, type GAUSS  to start GAUSS (for some versions of GAUSS, you may have to type GAUSSI or GAUSS386 instead of GAUSS). You will be in the COMMAND mode of GAUSS. You will see the GAUSS prompt, >>. In the COMMAND mode, you can execute screen-resident programs.




To exit the DOS version of GAUSS, press , then type  at the prompt.

A.2 Running a Program Stored in a File from the COMMAND Mode

From the command mode type `RUN FILENAME.EXP` and hit  to run a file named `FILENAME.EXP`, for example.

A.3 Editing a File

From the COMMAND mode, type `EDIT FILENAME.EXP` and hit  to edit a file named `FILENAME.EXP`, for example. You will be in the EDIT mode of GAUSS. You can edit the file with a full screen editor.

You can move around the file using arrow keys and   keys that are usually at the right of the keyboard. You can edit the file by deleting letters using  key and typing in letters.

A.4 Rules of Syntax

This section lists some of the general rules of syntax for GAUSS programs.

A.4.1 Statements

A GAUSS program consists of a series of statements. A statement is a complete expression or command. Statements in GAUSS end with a semicolon.

A.4.2 Case

GAUSS does not distinguish between upper and lower case except inside double quotes.

A.4.3 Comments

Comments can be placed inside `/*` and `*/`, which can nest other comments or inside `@` and `@`, which cannot nest other comments.

A.4.4 Symbol Names

The names of matrices, strings, procedures, and functions can be up to eight characters long. The characters must be alphanumerical or the underscore. The first character must be alphabetic or an underscore. Note that you cannot use some names that are already used by GAUSS. It is often a good idea to use unusual names in your programs to avoid potential problems.

A.5 Reading and Storing Data

```
LOAD X[n,m]=FILENAME.DAT;
```

reads in data stored in a ASCII file named FILENAME.DAT, for example. This data file should contain data separated by spaces in the form of an $n \times m$ matrix.

If X is a matrix of numbers in GAUSS,

```
SAVE XFILE=X;
```

stores X into a file named XFILE.FMT. Then you can read in the data again by

```
LOAD X=XFILE;
```

A.6 Operators

A.6.1 Operators for Matrix Manipulations

Assignment operator:

```
Y=3;
```

assigns the value 3 to the 1×1 matrix Y .

Indexing operator:

Brackets [] are used to index matrices. It is very important to note that parentheses () are used for different purposes in GAUSS such as indicating the dimensions of a matrix or to take arguments for commands or functions.

```
Y=X[3,3];
```

assigns the 3-3 element of X to Y . Commas are used to separate row indices from column indices. A vector can take one argument.

Period:

Dots are used in brackets to signify “all rows” or “all columns”.

```
Y=X[.,3];
```

assigns the third column of X to Y .

Colon:

A colon is used within brackets to create a continuous range of indices.

```
Y=X[1:5, .];
```

Transpose operator:

' transposes matrices.

Vertical Concatenation:

| is used to concatenate two matrices vertically.

```
Z=X|Y;
```

Horizontal Concatenation:

\sim is used to concatenate two matrices horizontally.

$$Z=X\sim Y;$$

A.6.2 Numeric Operators

Usual Operators:

Usual operators in GAUSS work according to standard rules of matrix algebra. For example, $*$ is the operator for matrix multiplication, and

$$Y=X*Z;$$

performs matrix multiplication when X and Z are conformable in the sense of matrix algebra.

Usual operators include $*$, $+$, $-$, and Kronecker product ($.*.$). $y=x.*.z$ results in a matrix in which every element in x has been multiplied by the matrix z . For example,

$$x = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, z = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \text{ then } x.*.z = \begin{bmatrix} 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \\ 3 & 0 & 4 & 0 \\ 0 & 3 & 0 & 4 \end{bmatrix}.$$

Element by Element Operators:

In some applications, X is an $m \times n$ matrix, and Y is an $m \times 1$ vector, and it is convenient to multiply each row of Y with each of the n elements in each row of X . The Element by Element Operators allow you to perform such operations. For example, $.*$ is the element by element multiplication operator, and

$$Z=X.*Y$$

performs the operation described above.

Other element by element operators are the following:

- ./ Element by element division:
 $Y=X./Z;$
- ^ Element by element exponentiation:
 $Y=X^Z;$
- + Element by element addition.
 $Y=X+Z;$
- - Element by element subtraction.
 $Y=X-Y;$

A.7 Commands

A.7.1 Functions

The following is a short list of useful functions. See the GAUSS manual for other useful functions.

- `cols(x)`: with a matrix x returns the number of columns of x .
- `diag(x)`: with an $M \times M$ matrix x returns a column vector of the diagonal of x .
- `eye(N)`: returns an $N \times N$ identity matrix.
- `ln(x)`: with an $M \times N$ matrix x returns an $M \times N$ matrix of the natural logarithm of all elements in x .
- `meanc(x)`: with an $M \times N$ matrix x returns an $N \times 1$ vector of the means of the columns of x .
- `int(x)`: with an $M \times N$ matrix x returns the $M \times N$ matrix of the largest integer which is smaller than or equal to each element of x .

- `invpd(x)`: with a symmetric, positive definite $N \times N$ matrix x returns the inverse of x .
- `inv(x)`: with an $N \times N$ matrix x returns the inverse of x .
- `ones(M,N)`: returns an $M \times N$ matrix of ones. For example, `x=ones(3,2)`; will create $x = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$.
- `sqrt(x)`: with an $M \times N$ matrix x returns an $M \times N$ matrix of the square roots of all elements of x .
- `reshape(x,r,c)`: with an $N \times K$ matrix x , and two scalars r and c , returns an $r \times c$ matrix created from the elements of x . The elements in x are first stored in row major order, and then the first c elements are put into the first row of the created matrix, the second in the second row, and so on. For example, `y=reshape(x,4,3)` for $x = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 8 & 9 & 10 & 11 & 12 \end{bmatrix}$ creates $y = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$.
- `rows(x)`: with a matrix x returns the number of rows of x .
- `zeros(M,N)`: returns an $M \times N$ matrix of zeros.

A.7.2 Printing

```
PRINT X Y;
```

will print matrices X and Y to the screen. Instead of matrices, you can print words inside double quotes:

```
PRINT "This will be printed";
```

You can use ? instead of PRINT:

```
? X Y;
```

A.7.3 Preparing an Output File

```
OUTPUT FILE = FILENAME.OUT RESET;
```

allows you to write the output of PRINT statements to a file named FILENAME.OUT, for example. To print out or edit the output file, you have to close the output file by the

```
OUTPUT OFF;
```

command. Most of the programs in the GMM package contain this statement near their end. However, if your program does not reach its end because of errors, you have to issue this command from the COMMAND mode to close the file to check the output file.

A.8 Procedure

A.9 Examples