

GETTING STARTED WITH *MATHEMATICA*

Martin Erickson

March 7, 2001

The purpose of this manual is to help you begin using *Mathematica*, a mathematical exploration system in which you can perform calculations, evaluate functions, create graphics, and develop programs. The manual contains simple examples for you to try. It covers just the basics, enough to get you started. To learn more about *Mathematica*, you may want to consult the sources listed in Section 7 and in the references that follow.

Contents

1. **What is *Mathematica* ?**
2. **How to use *Mathematica* as a calculator**
3. **How to compute functions**
4. **How to graph functions**
5. **How to do simple programming**
6. **How to develop a *Mathematica* notebook**
7. **How to learn more**

References

1 What is *Mathematica* ?

Mathematica, created by Stephen Wolfram, is a software system in which you can investigate mathematics, perform calculations, create graphics, and write programs.

Mathematica commands are typed on a graphical user interface containing menu options (see Figure 1).

2 How to use *Mathematica* as a calculator

You can use *Mathematica* as a powerful calculator. Simply type the expression you wish to evaluate and press SHIFT+ENTER (or just the ENTER key on the lower-right corner of an extended keyboard).

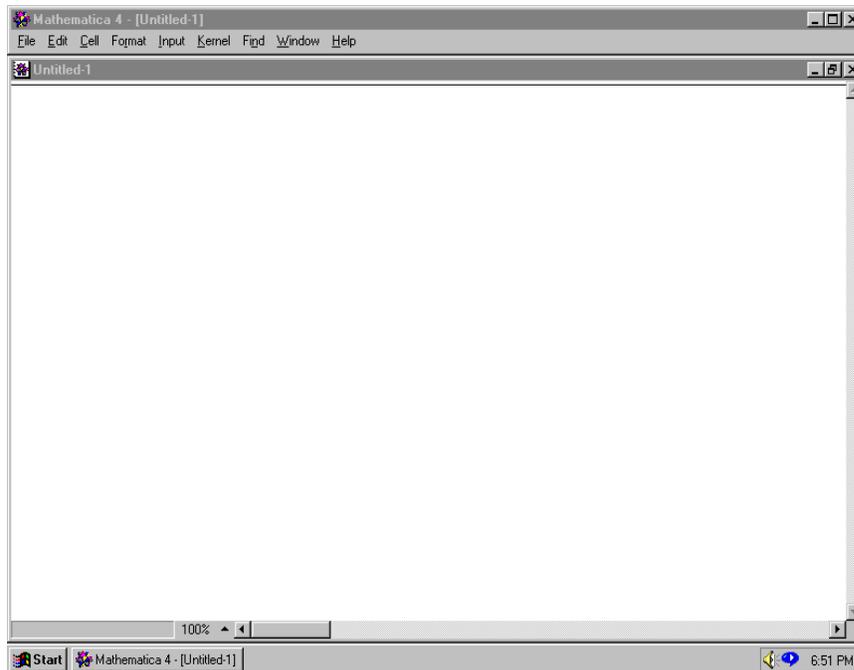


Figure 1: The *Mathematica* command screen.

Example 2.1. Here we add 2 and 2 to get ... 4.

```
2 + 2
```

```
4
```

Note. The first time you ask *Mathematica* to perform a calculation, *Mathematica* starts up the *kernel* (its calculating engine).

Note. *Mathematica* assigns line numbers to the input and output, but we suppress the numbers in this manual.

Mathematica can handle very large numbers. The following example shows off *Mathematica*'s fantastic calculating power.

Example 2.2. We evaluate 3^{100} .

```
3^(100)
```

```
515377520732011331036461129765621272702107522001
```

Note. *Mathematica* can easily work with numbers hundreds of digits long.

To multiply two numbers, type the numbers with a space between them and enter.

Example 2.3. Here we calculate $1024 \cdot 59049$.

```
1024 59049  
  
60466176
```

The values of useful mathematical constants such as π , e , and i are stored in *Mathematica*. To obtain the numerical value of an expression, use the function `N`.

Example 2.4. We calculate π and e .

```
N[Pi]  
  
3.14159  
  
N[E]  
  
2.71828
```

If you want a numerical result given to a high degree of accuracy, use the command `N[_,_]`. The first argument of this function is the number to be calculated; the second argument is the number of decimal places to which the number is computed.

Example 2.5. We calculate π to 100 decimal places.

```
N[Pi,100]  
  
3.1415926535897932384626433832795028841971693993751058209749  
44592307816406286208998628034825342117068
```

In addition to making numerical calculations, *Mathematica* performs algebraic operations.

Example 2.6. We set a equal to 17 and then calculate with a .

```
a = 17  
  
17  
  
a^3 + a - 15  
  
4915
```

If you want to work with the output of the previous command, use `%` (percentage sign).

Example 2.7. We compute the square of the output of the previous example (4915).

```
%^2  
  
24157225
```

Mathematica performs matrix calculations. Matrices are stored as sets of row vectors.

Example 2.8. We define two matrices,

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \text{ and } N = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

```
m = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};  
n = {{0, 1, 0}, {0, 0, 1}, {1, 0, 0}};
```

Note. The ; (semicolon) symbol is used to separate commands. If you end a command with a semicolon, the output will not be displayed.

We add the matrices.

```
m + n  
  
{{1, 3, 3}, {4, 5, 7}, {8, 8, 9}}
```

And we multiply them.

```
m . n  
  
{{3, 1, 2}, {6, 4, 5}, {9, 7, 8}}
```

We may want to use the same variables (e.g., a , m , and n in the above computations) later in a different context. Therefore, it is a good idea to “clear” the values of variables when we are finished using them.

```
Clear[a,m,n]
```

We check that the values of these variables have disappeared.

```
{a, m ,n}  
  
{a, m, n}
```

Note. You can obtain information about a specific command by typing a question mark followed by the name of the command. For example, to find out about the function `N`, type `? N`.

```
? N
```

`N[expr]` gives the numerical value of `expr`. `N[expr, n]` attempts to give a result with `n`-digit precision.

We have only scratched the surface of *Mathematica*'s calculating capabilities. For more information, see *Mathematica*'s help index (click on `HELP` on the menu bar) and the references listed at the end of this manual.

3 How to compute functions

The operator `N` in Examples 2.4 and 2.5 is a function. *Mathematica* contains many such built-in functions. Usually, a function's name is what you expect it to be. For example, `Sin[x]` computes $\sin x$.

Note. In *Mathematica*, every function name begins with a capital letter.

Example 3.1. We calculate $\sin(\pi/2)$.

```
Sin[Pi/2]
```

```
1
```

Some functions take more than one argument.

Example 3.2. We calculate the binomial coefficient $\binom{7}{2}$.

```
Binomial[7,2]
```

```
21
```

Some functions have outputs that are strings.

Example 3.3. The command `FactorInteger` determines the prime factorization of an integer. Here we find the prime factorization of the number 60466176 (obtained in Example 2.3 as the product of 1024 and 59049).

```
FactorInteger[60466176]
```

```
{{2, 10}, {3, 10}}
```

The output tells us that $60466176 = 2^{10} \cdot 3^{10}$.

In the next example, we calculate and display a table of function values.

Example 3.4. The function `Prime[n]` gives the n th prime number. Using this function, we construct a table of the first 100 prime numbers.

```
Table[Prime[n], {n, 1, 100}]
```

```
{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151,
157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233,
239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317,
331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419,
421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503,
509, 521, 523, 541}
```

Mathematica can evaluate functions both arithmetically and symbolically.

Command	Meaning	Example input	Meaning
<code>Sqrt[]</code>	square root	<code>Sqrt[5]</code>	$\sqrt{5}$
<code>Exp[]</code>	exponential	<code>Exp[x]</code>	e^x
<code>Log[]</code>	natural logarithm	<code>Log[10]</code>	$\ln 10$
<code>Log[,]</code>	logarithm to a specified base	<code>Log[10,5]</code>	$\log_{10} 5$
<code>Sin[]</code>	sine	<code>Sin[x]</code>	$\sin x$
<code>Cos[]</code>	cosine	<code>Cos[x]</code>	$\cos x$
<code>Tan[]</code>	tangent	<code>Tan[x]</code>	$\tan x$
<code>Sum[,]</code>	sum	<code>Sum[i, {i, 1, n}]</code>	$\sum_{i=1}^n i$
<code>Product[,]</code>	product	<code>Product[i, {i, 1, n}]</code>	$\prod_{i=1}^n i$
<code>Mod[,]</code>	modulo	<code>Mod[10,3]</code>	$10 \bmod 3$

Table 1: Some useful functions.

Example 3.5. Here we calculate $\sum_{i=1}^{10} i^2$.

```
Sum[i^2, {i, 1, 10}]
```

```
385
```

Now we compute the same sum with the upper limit 10 replaced by n .

```
Sum[i^2, {i, 1, n}]
```

```
(1/6) n (1+n) (1+2n)
```

This result tells us that $\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$ (a well-known formula).

Table 1 displays some useful *Mathematica* functions.

You can define your own functions. To create a function $f(x)$, write `f[x_]:=` followed by the definition of f .

Example 3.6. We define a function $f(x) = x^3 + \sin x$.

```
f[x_] := x^3 + Sin[x]
```

Now we evaluate the function.

```
f[Pi/2]
```

```
1 + Pi^3/8
```

We differentiate the function.

```
D[f[x], x]
```

```
3 x^2 + Cos[x]
```

And we integrate it.

```
Integrate[f[x],x]
```

```
x^4/4 - Cos[x]
```

Note. *Mathematica* does not supply an additive constant ($+C$) for indefinite integrals. We compute a definite integral of our function.

```
Integrate[f[x], {x, 0, Pi}]
```

```
2 + Pi^4/4
```

You can define functions recursively.

Example 3.7. We define the Fibonacci sequence.

```
f[0] = 1;  
f[1] = 1;  
f[n_] := f[n-1] + f[n-2]
```

```
Table[f[n], {n, 0, 10}]
```

```
{1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89}
```

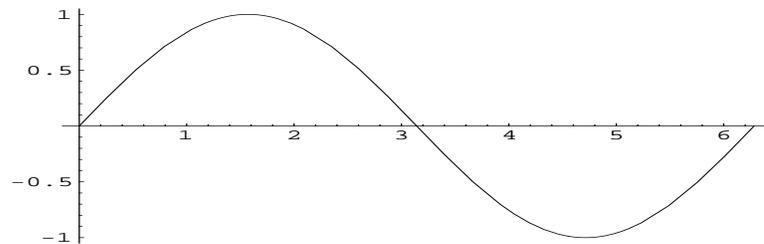
4 How to graph functions

Mathematica offers many graphing options. We show a few examples here.

You can create graphs of functions using *Mathematica*'s `Plot` command.

Example 4.1. Here we graph the function $y = \sin x$, for $0 \leq x \leq 2\pi$.

```
Plot[Sin[x], {x, 0, 2 Pi}]
```



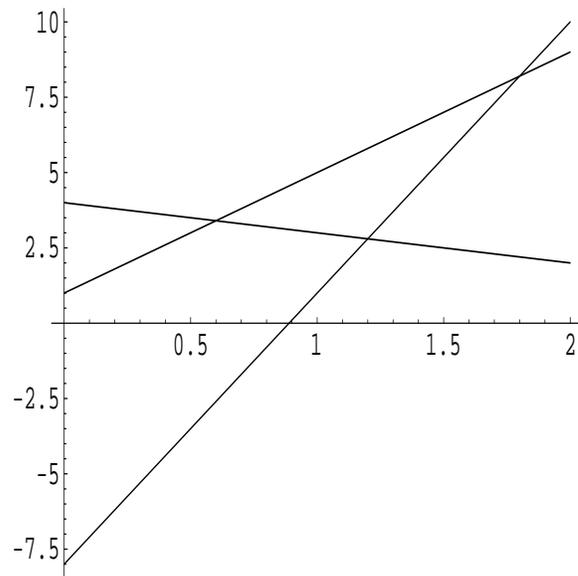
You can graph several curves together.

Example 4.2. We graph the three lines $y = 4x + 1$, $y = -x + 4$, and $y = 9x - 8$.

```

f[x_] := 4 x + 1;
g[x_] := -x + 4;
h[x_] := 9 x - 8;
Plot[{f[x], g[x], h[x]}, {x, 0, 2}]

```



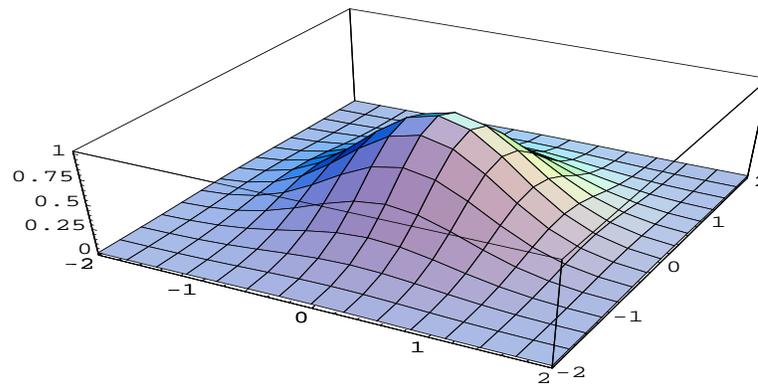
You can represent 3-dimensional graphs using the `Plot3D` command.

Example 4.3. We graph $z = e^{-(x^2+y^2)}$, for $-2 \leq x, y \leq 2$.

```

Plot3D[E^(-(x^2 + y^2)), {x, -2, 2}, {y, -2, 2}]

```



5 How to do simple programming

Mathematica supports a full spectrum of programming paradigms, including procedural, transformational, functional, and object-oriented approaches. We give a sampling here.

We use a Do loop to cause a procedure to occur multiple times.

Example 5.1. Here we make some computations related to the fractal known as the *Mandelbrot set*. We set $c = -0.5 + 0.5i$ and $z = 0 + 0i$. Then we iterate the function $f(z) = z^2 + c$ ten times.

```
c = -0.5 + 0.5 I;
z = 0 + 0 I;
Do[
  z = z^2 + c,
  {10}
];
z
-0.11932 + 0.219608 I

Clear[c,z]
```

Perhaps we wish to run the above program several times, using different values of c and different numbers of iterations. To do this, we create a module, which is procedure containing local variables.

Example 5.2. We define a module containing the local variable z . The values of c and i (the number of iterations) are input when the module is called.

```
f[c_, i_] := Module[{z},
  z = 0 + 0 I;
  Do[
    z = z^2 + c,
    {i}
  ];
  z
]
```

We test the module.

```
f[-0.5 + 0.5 I, 10]
-0.11932 + 0.219608 I
```

We check to see that z has no value outside the module.

```
z
z
```

It is good programming practice to use modules, and to make them small and easy to understand.

Functions are “threaded” over lists automatically.

Example 5.3. We thread addition and cubing operations over the list $\{a, b, c\}$.

```
1000 + {a, b, c}^3
```

```
{1000 + a^3, 1000 + b^3, 1000 + c^3}
```

Sometimes functions are complex enough to be called programs.

Example 5.4. We define a function

$$f(n) = \frac{1}{n} \sum_{k|n} \phi(k) 2^{n/k}.$$

Note. From the Pólya theory of counting, $f(n)$ is the number of distinct (up to rotation and flipping) necklaces formed by n beads of two types.

The summation is over a set of numbers, namely, the set of positive divisors of n . This set is obtained in *Mathematica* as `Divisors[n]`. We need to apply the summand, $\phi(k)2^{n/k}$, to each element of this set. The summand contains a “dummy variable,” k . To define the summand as a *Mathematica* function, we replace each instance of the dummy variable with the marker `#`.

```
EulerPhi[#]2^(n/#)&
```

The `&` identifies the function as a “pure function” in which the argument is denoted by `#`.

Now we apply the function to the set `Divisors[n]` as follows.

```
EulerPhi[#]2^(n/#)&/@Divisors[n]
```

(The construction `f/@s` applies a function `f` to a set `s`.)

Finally, we add the elements of the set produced by this process. The expression `Plus@@s` adds the elements of the set `s`. Thus, we can now define our function in *Mathematica*.

```
f[n_] := (1/n)Plus@@(EulerPhi[#]2^(n/#)&/@Divisors[n])
```

We test our function.

```
f[4]
```

```
6
```

It is easy to verify by inspection that there are indeed exactly six different necklaces made of six beads of two types.

And now we compute a value that we could never verify by hand.

```
f[100]
```

```
12676506002282305273966813560
```

6 How to develop a *Mathematica* notebook

With a combination of input, output, and comments, you can use *Mathematica* to create a document that presents and explains your mathematical work.

Mathematica files are called *notebooks*. When you save your file as a notebook, *Mathematica* gives it an `nb` extension (creating, for example, `myfile.nb`).

Notebooks are composed of *cells*. The *cell brackets* at the right edge of a notebook show the extent of each cell. A cell may contain commands, output, graphics, or text.

To create a new cell, place the cursor below the last cell and begin typing. To evaluate a cell, place the cursor on the cell to be evaluated and press SHIFT+ENTER (or ENTER on an extended keyboard). To delete a cell, place the cursor on the cell to be deleted and enter CTRL+X.

If you want to include some comments (text) in your notebook, click on the cell bracket of the cell to contain the comments. Then go the MENU BAR and click on FORMAT, drag the cursor to STYLE, and enter TEXT. Similarly, you can select TITLE, SECTION, etc.

7 How to learn more

There are many aspects of *Mathematica* that are not discussed in this manual, such as standard and add-on packages, sound, and animated graphics. Here are some resources for you to investigate to learn more.

The definitive book about *Mathematica* is [9]. Good beginning books are [2] and [7]. For informative examples of *Mathematica* in a wide variety of settings, see [1], [8], and [6]. The books [4] and [3] show many applications of *Mathematica* to calculus. For a comprehensive guide to add-on packages, see [5].

For the most complete information describing *Mathematica*, you may want to visit the web site <http://www.wolfram.com/>. Another interesting site, concerning the *Mathematica in Education and Research* journal, is <http://www.telospub.com/journal/MIER/>.

References

- [1] K. R. Coombes, B. R. Hunt, R. L. Lipsman, J. E. Osborn, and G. J. Stuck. *The Mathematica Primer*. Cambridge University Press, New York, 1998.
- [2] J. Glynn and T. Gray. *The Beginner's Guide to Mathematica*. Cambridge University Press, New York, fourth edition, 2000.
- [3] S. Hollis. *Multivariable CalcLabs with Mathematica for Stewart's Multivariable Calculus*. Brooks/Cole, New York, fourth edition, 1999.
- [4] S. Hollis. *Single Variable CalcLabs with Mathematica for Stewart's Calculus, Single Variable*. Brooks/Cole, New York, fourth edition, 1999.
- [5] E. Martin, editor. *Mathematica 4 Standard Add-on Packages*. Cambridge University Press, New York, first edition, 1999.
- [6] H. Ruskeepää. *Mathematica Navigator: Graphics and Methods of Applied Mathematics*. Academic Press, New York, first edition, 1998.
- [7] B. F. Torrence and E. A. Torrence. *The Student's Introduction to Mathematica: A Handbook for Precalculus, Calculus, and Linear Algebra*. Cambridge University Press, New York, 1999.
- [8] S. Wagon. *Mathematica in Action*. Springer-Verlag, New York, second edition, 1999.
- [9] S. Wolfram. *The Mathematica Book*. Cambridge University Press, New York, fourth edition, 1999.