

Mathematica Notes

Math 421-A

Fall, 2001

Principles for Using *Mathematica*

- Built-in commands, functions (such as `Sin[x]`) and variables (such as $\pi = \text{Pi}$, $e = \text{E}$, and $i = \text{I}$) are always capitalized.
- A command is not executed until you do a shift-return.
- Parentheses are used only for grouping; functions and commands both use square brackets.
- Vectors use set braces. To access an element of a vector v , use `v[[index]]`. The indices of a vector of length n run from 1 to n .
- To define a function, you need to use an underscore with the variable on the left-hand side: `f[x_] := x^2`. The underscore tells *Mathematica* to replace any x by whatever the input is to the function. Otherwise `f[x] := x^2` will only work when the input to f is x .
- Ranges for plotting, integration, etc. are always written as lists in braces, e.g., `Plot[x^2, {x, -2, 3}]` is the command for plotting x^2 with x ranging from -2 to 3 .
- When in doubt, use the *Help Browser* to do a keyword search. You will see examples of the various commands this way.

Common Errors

- xy is not the product of x and y ; it is a single variable xy . You need to use either `x y` or `x*y` to get the product of two variables.
- If you edit entries in your *Mathematica* notebook, it is very easy for the entries to get out of synchronization spatially versus temporarily, i.e., later entries will appear above earlier entries. It is important that if you are turning a notebook in for an assignment that you try saving it, opening it, and verifying that the commands work in the order that they appear on the screen.

- `Sin(x)` is the product of two variables, *Sin* and *x*. Use `Sin[x]` to compute a sine. In general, make sure that you always use square brackets for function evaluations.

Commands Used In Math 321

Manipulating Expressions

- `Expand[expr]` expands an algebraic expression into its simplest terms.
Example: `Expand[(x+2)^3]`.
- `Factor[expr]` factors an algebraic expression as far as possible.
Example: `Factor[x^6-1]`.
- `Simplify[expr]` attempts to write an expression in the simplest possible way without making any assumptions about the variables.
Example: `Simplify[3x+2x]`
- `Simplify[expr, assumption(s)]` attempts to write an expression in the simplest possible way making only the given assumption(s).
Example: `Simplify[Sqrt[x^4], Element[x,Reals]]`.
Example: `Simplify[Sqrt[x^2], x>0]`
Example: `Simplify[Sqrt[x^6]+Log[y^2], {Element[x,Reals], y>0}]`

Solving Equations

All of the following commands return lists of rewrite rules of the form $\{x \rightarrow 2\}$ or $\{x \rightarrow 3, y \rightarrow 4\}$ as their solutions. If you need to apply these rules use the `/.` operator.

Example: if `rules` contained $\{\{x \rightarrow 2, y \rightarrow 3\}, \{x \rightarrow 4, y \rightarrow 5\}, \{x \rightarrow 6, y \rightarrow 7\}\}$, and you wanted to apply the second rewrite rule to `f[x,y]`, you would type `f[x,y] /. rules[[2]]` and the result would be the value of `f[4,5]`.

- `Solve[eqn(s), var(s)]` attempts to solve symbolically the given equation(s) for the variable or variables listed. This command may not be successful if the equation is too difficult.
Example: `Solve[x^3-3x+1 == 0, x]`.
Example: `Solve[{x^3-y^3 == x-y, x^2+y^2 == 4}, {x,y}]`.
- `NSolve[eqn(s), var(s)]` attempts to solve symbolically, then numerically approximate the given equation for the variable or variables listed. This command may not be successful if the equation is too difficult.
Example: `NSolve[x^3-3x+1 == 0, x]`.
Example: `NSolve[{x^3-y^3 == x-y, x^2+y^2 == 4}, {x,y}]`.

- `FindRoot[eqn, {var, initial guess}]` will use Newton's method to find a root for a given equation. It is likely to work in situations where `Solve` and `NSolve` fail, but will only give a numerical (as opposed to symbolic) solution. It can fail at places where Newton's method is inapplicable.

Example: `FindRoot[x^3-3x+1 == 0, {x,2}]`.

Plotting

- `Plot[expr, range]` plots a function of one variable over a given range of values.

Example: `Plot[x^2, {x, -2, 3}]`.

- `Plot3D[expr, range1, range2]` plots a function of two variables over given ranges of values.

Example: `Plot3D[x^2+y^2, {x, -2, 3}, {y, -1, 3}]`.

- `ParametricPlot[Evaluate[expr], range]` plots a two dimensional parametric curve.

Example: `ParametricPlot[Evaluate[{t^2, t^3-3t}], {t, 1, 2}]`.

- `ParametricPlot3D[Evaluate[expr], range]` plots a three dimensional parametric curve.

Example: `ParametricPlot3D[Evaluate[{t^2, t^3, t^4}], {t, 1, 2}]`.

- `ParametricPlot3D[Evaluate[expr], range1, range2]` plots a three dimensional parametric surface.

Example: `ParametricPlot3D[Evaluate[{t^2+u, t^3-u, t^4-u^2}], {t, 1, 2}, {u, 3, 4}]`.

Differentiation and Integration

- `D[expr, var]` finds the partial derivative of the given expression with respect to the given variable.

Example: `D[Sin[x^2*y], x]`.

- `D[expr, {var, order}]` finds the partial derivative of the given expression with respect to the given variable to the given order.

Example: `D[E^(2x)*y^2, {x, 3}]`.

- `D[expr, {var1, order1}, {var2, order2}, ...]` finds the partial derivative of the given expression with respect to the given variables to the given orders.

Example: `D[x^3*Sin[y], {x, 2}, {y, 2}]`.

- `Integrate[expr, {var, lower limit, upper limit}]` finds the symbolic integral of the given expression with the given variable and limits. This command will fail if the integration is too difficult.

Example: `Integrate[x^2, {x, a, b}]`.

- `Integrate[expr, {var1, a1, b1}, {var2, a2, b2}, ...]` will try to find the symbolic integral of the given expression with the given variables and limits. The order of the variables is the order of the integration signs, which is the reverse of the order of the differentials. This command will fail if the integration is too difficult.

Example: `Integrate[x^2*E[y], {x, a, b}, {y, c, d}]`.

- `NIntegrate[expr, {var1, a1, b1}, {var2, a2, b2}, ...]` finds the numerical integral of the given expression with the given variables and limits. The order of the variables is the order of the integration signs, which is the reverse of the order of the differentials.

Example: `NIntegrate[Sin[x^2+y^2], {x, 0, Pi}, {y, 0, x}]`.

Vector Operations

Vectors are lists of elements.

Note: we usually defined our own length operator `len[v_] := Sqrt[v.v]`.

- `v.w` finds the dot product of the two vectors.

Example: `{1,2,3}.{3,4,5}`.

- `Cross[v,w]` finds the cross product of the two vectors. This is only defined if each vector has three components.

Example: `Cross[{1,2,3},{3,4,5}]`.

Vector Calculus

These commands all require that you load the following package and set the coordinate system.

```
<<Calculus`VectorAnalysis`
```

```
SetCoordinates[Cartesian[x,y,z]
```

You should only have to do this once in a given *Mathematica* session.

- `Grad[expr]` will find the gradient for the given function.

Example: `Grad[x*y^2*z^3]`.

- `Curl[field]` will find the curl of the given vector field.

Example: `Curl[{x+y,y+z,z+x}]`.

- `Div[field]` will find the divergence of the given vector field.

Example: `Div[{x+y,y+z,z+x}]`.