An Introduction to Using Maple in Math 23 *

R. C. Daileda

The purpose of this document is to help you become familiar with some of the tools the Maple software package offers for visualizing solutions to differential equations. We'll start by describing the basics of Maple and then move on to the specific commands that we'll find most useful. In every case, we'll only be scraping the surface. For much more detailed information on the commands and their syntax and options see Maple's extensive "help" documentation. As you read this, you might find it useful to have Maple running on your computer so that you can try things out as you read about them.

1 General

The command prompt in Maple is the greater than symbol: `>`. Everything entered by you will follow one of these prompts. You'll never have to actually type in the command prompt: Maple does that part for you. After typing in a command, to end and execute it, type a semicolon (`;`) and hit "enter". For example

```
3 + 4;
```

followed by "enter" will ask Maple to add 3 to 4, producing 7 on the following line. You can also end a command with a colon (`:`). The difference is that the colon tells Maple to suppress its output. This is useful when you want Maple to perform a computation but when you don't want to see the results right away (or at all). We'll see some instances of its use later.

Addition, subtraction and division all use the usual symbols: `+`,`-`, `/`. Multiplication is done using the asterisk: `*`. Maple *does not* interpret concatenation as multiplication. So, to enter $3y$ you need to type `3 * y` or `y*3` and *not* `3y` or `y3`. Exponentiation is achieved by using the caret: `^`. So, for example, to compute 18 multiplied by 2 to the seventh power, you would type

```
18*2^7;
```

followed by "enter".

Overuse of parentheses is never a bad idea when you aren't sure how Maple might choose to associate your operations. For example, if we wanted to enter

$$\frac{4x^2+5}{y-7}$$

we'd type

```
(4*x^2)/(y-7).
```

Note that we have enclosed the numerator and denominator in parentheses, with the division symbol `/` between them. Any complex fraction should be entered this way, otherwise Maple will get confused and give you funny results. Parentheses should be used in a similar way with exponents. y^{2x-4z} would be entered

```
y^(2*x - 4*z).
```

Such a use of parentheses is necessary any time the exponent consists of more that a single symbol.

Maple also knows most of your basic algebraic and transcendental functions. It knows how to compute values of the exponential function, natural log, sine, cosine, etc. . Most of these use the usual names you'd

*Last updated on May 8, 2006.

see in a math book or on a programmable calculator. For example, to compute $\cos 8$ you'd type `cos(8);` and hit "enter." To find the natural log of $\sqrt{2}$ you'd type `ln(2^(1/2));`. Note that we have used exponential notation for the square root. We could also have typed `sqrt(2)`. Be aware, however, that to evaluate the exponential function e^x you *do not* simply enter `e^x` as you might expect. You actually type `exp(x)`. So to compute $e^{4/3}$ you would type `exp(4/3);`. If you were to try this out, Maple would give you the output

$$e^{\left(\frac{4}{3}\right)}$$

which makes it look like Maple didn't do anything! If you want to know the decimal expansion of this or any other "symbolic" number you can use the command `evalf`. So, continuing with our exponential example, typing in `evalf(exp(4/3))` gives the output 3.793667893.

Okay, now for a few more interesting (and perhaps less obvious) commands. To assign values to variables you use the assignment operator `:=`. So, to create a variable named x and set it equal to 8, you type

```
x := 8;
```

end hit "enter". To create a variable name *george* and give it the value of x, you'd type

```
george := x;
```

and hit "enter". Variables that have numerical values assigned to them can be used in any expression the same way a number would be. For example, you could ask Maple to find the natural log of the variable *george* the same way you would ask it to find the natural log of 8: by typing `ln(george)`.

Variables can actually hold almost any kind of data that Maple knows how to process. In fact, we will frequently use variables to store equations or even entire plots. Encapsulating data using variables makes it easier to deal with many "large" objects at once. We'll see exactly how later.

2 Direction Fields

To plot the direction field for a first order differential equation of the form

$$\frac{dy}{dt} = f(t, y)$$

you can use the command `dfieldplot`. Before you can use this command, however, you must load the `DEtools` package. This is done by typing `with(DEtools):` at the command prompt and pressing enter. Even if you plan on having Maple draw several direction fields, you only need to load the `DEtools` package *once*.

Let's demonstrate the use of `dfieldplot` with an example. Suppose you want to plot the direction field of the differential equation

$$\frac{dy}{dt} = \frac{t^2 + 5t}{3y - 7y^2}. \tag{1}$$

At the command prompt type

```
dfieldplot(diff(y(t),t) = (t^2 + 5*t)/(3*y(t) - 7*y(t)^2),
                    y(t), t=-3..3, y=-3..3, arrows=line );
```
(2)

and hit "enter." This produces the output shown in Figure 1.

So what, exactly, is all of the stuff in the line of input (2) and how is it related to the equation (1)? The first thing you see here after the word `dfieldplot` is

```
diff(y(t),t) = (t^2 + 5*t)/(3*y(t) - 7*y(t)^2).
```

This specifies the differential equation. The part to the left of the equal sign, `diff(y(t),t)`, is Maple's notation for the derivative of $y(t)$ with respect to t, i. e. dy/dt. To the right of the equal sign we have `(t^2 + 5*t)/(3*y(t) - 7*y(t)^2)`, which is just the rest of the differential equation. Notice that we have used

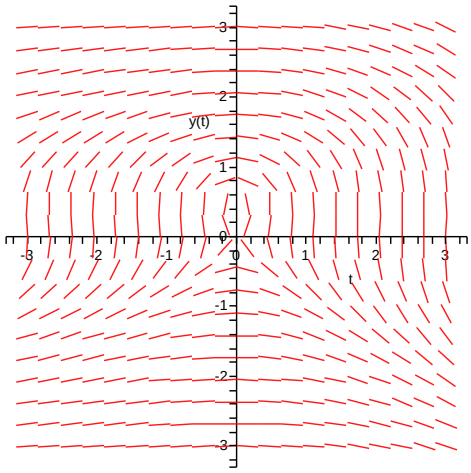

Figure 1: Output of command (2)

`y(t)` everywhere that y appears in equation (1). Although Maple still seems to get the right picture if we just use `y`, it will complain at us first. So better to just stick to the preferred syntax. You can enter the equation however you'd like, as long as you remember that all y's are entered as `y(t)` and dy/dt is entered as `diff(y(t),t)`. So, we could just as well have typed

```
dfieldplot((3*y(t) - 7*y(t)^2)*diff(y(t),t) = t^2 + 5*t,
                    y(t), t=-3..3, y=-3..3, arrows=line );
```

which corresponds the the equivalent differential equation

$$(3y - 7y^2)\frac{dy}{dt} = t^2 + 5t.$$

Okay, so what about the rest of the stuff in the `dfieldplot` command? The next thing you see is `y(t)` by itself. This just tells Maple that y is the dependent variable (the "unknown" function) and that it is a function of t. The next two things you see are `t=-3..3` and `y=-3..3`. These just specify the rectangle in the ty-plane in which the direction field should be plotted. In this example we have t running from -3 to 3 and y in the same range. To change the window, just change the numbers -3 or 3 to whatever you'd like. For example, if you wanted to "zoom in" around the point $(0, -1)$ to see what the direction field looked like there, you might try `t=-1..1, y=-2..0` instead. Finally, we come to `arrows=line`. This tells Maple that it should use small line segments (as opposed to arrows) when it draws your direction field. Should you leave this off you'll get a direction field that looks more like a vector field, with each line segment replaced by an arrow.

Should you want to use different independent and dependent variables that y and t, just change them (everywhere that they occur!) to whatever you'd like.

Theres one additional option that I didnt include in the example above but that I should mention: its possible to specify the color of the line segments of the direction field. Not only do colors help fight boredom, but they can be useful if you plan to overlay a plot of a solution curve and a direction field (see below). To tell Maple that you want your direction field to have blue line segments, the `dfieldplot` command (2) should end with

```
... arrows=line, color=blue);
```

You can replace `blue` with any of the following colors: `aquamarine, black, blue, navy, coral, cyan, brown, gold, green, gray, grey, khaki, magenta, maroon, orange, pink, plum, red, sienna, tan, turquoise, violet, wheat, white, yellow`.

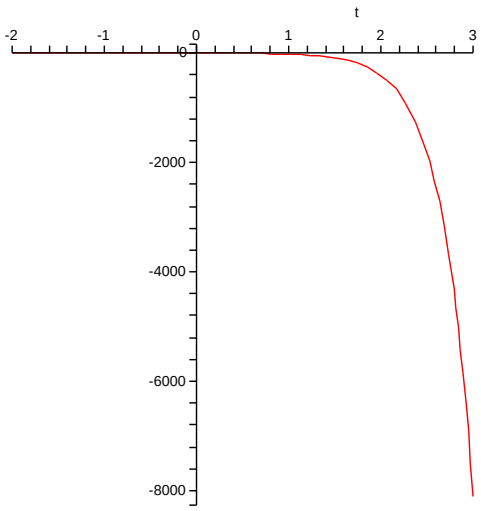

Figure 2: Output of command (3)

3 Solution Curves

There are several ways to plot solution curves of a differential equation. We will describe two commands here that plot solutions once you have solved for them by hand. There are commands to compute numerical approximations to solutions, but we'll wait to describe them until they're needed for our classwork. Also, we will limit ourselves here to the plotting of solutions to *ordinary* differential equations. We'll save the plotting of multivariable functions for later as well.

First, let's talk about the `plot` command. To use it you will need to load the `plots` package by typing `with(plots):` and pressing "enter." As before, this only needs to be done once. The command `plot` will graph any function of the form $y =$ (formula in x). For example, to have Maple draw the graph of $y = x^3 + 2x - 7$ in the region $-3 \leq x \leq 2$ you'd type

```
plot(x^3 + 2*x - 7, x = -3..2 );
```

Notice that we typed `x^3 + 2*x - 7` and not `y = x^3 + 2*x - 7`. The `plot` command does *not* require you to give it the dependent variable, and in fact it will complain if you do enter it. Also, notice that we specified the x range as in `dfieldplot` but that we did not specify the y range. Maple adjusts the y range so that the the graph fits in the window. But, if you want to, you can tell Maple the y range as we did in the previous section.

The command `plot` will allow you to graph explicit solutions of differential equations, i.e. those solutions that express y explicitly as a function of t. So, for example, if we wanted to plot the particular solution

$$y = \frac{3}{2} - e^{3t}$$

to the differential equation

$$y' = 3y - 2$$

in the region $-2 \leq t \leq 3$ we would type

```
plot(3/2 - exp(3*t), t = -2..3);
```
(3)

yielding the output of Figure 2.

Since $y \to -\infty$ very rapidly as $t \to \infty$ in this example, the plot we get is a little skewed: our t range has a width of 5 but the y range is close to 8000! So in this case it might not be a bad idea to tell Maple the y

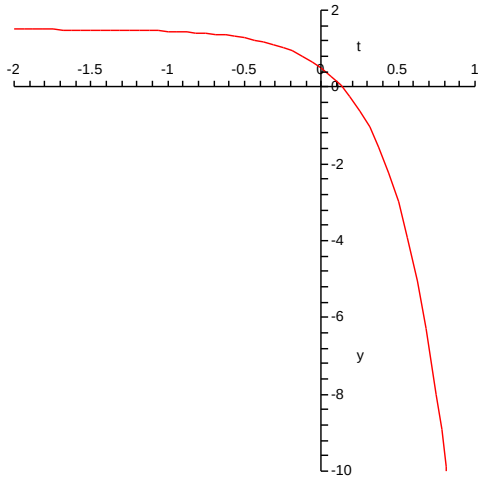

Figure 3: Output of command (4)

range we want. We adjust the t range a little to and try

```
plot(3/2 - exp(3*t), t = -2..1, y = -10..2);
```
(4)

yielding the less "stretched" output of Figure 3.

When we solve some types of equations (e.g. separable equations) our solutions are more naturally expressed implicitly, that is, in the form $f(t, y) = 0$. To plot such solutions we need the command `implicitplot`. As with `plot`, it can only be used once the `plots` package has been loaded, and this can be done by typing `with(plots):` and pressing "enter." But if you have already loaded the `plots` package, there's no need to do it again.

To see how `implicitplot` works, we again use an example. To graph the curve $y^2 + 3y = x^3 - 2x^2 + 1$ in the region $-2 \leq x \leq 3$, $-4 \leq y \leq 2$ we type

```
implicitplot(y^2 + 3*y = x^3 -2*x^2 + 1, x=-2..3, y=-4..2);
```
(5)

and hit "enter." This produces the output shown in Figure 4. As with `dfieldplot`, the equation can be entered in any form you like. You could use `y^2 + 3*y - x^3 +2*x^2 = 1` instead if you choose to. Also as before, `x=-2..3, y=-4..2` specify the window, and can be changed to anything you like.

As with direction fields, you can specify the color of your curves using the `color` option, as described above. Another useful thing you can do is increase the thickness of your curves, so they stand out a little more. This is done by using the `thickness` option. Its tacked on just like the `color` option. To set the thickness to 3 and the color to green in the preceding example, you would end the command with

```
... y = -4..2, thickness = 3, color = green);
```

The thickness can be any integer ≥ 0. You should try out a few different values to find one that suits you.

4 Combining Plots

So now you can plot direction fields and solution curves. But what if you want to put multiple plots on the same graph? For instance, you might want to plot several solution curves together. Or maybe you want to plot a solution curve on top of a direction field. Both of these tasks are easily achieved by saving your plots in variables and then using the `display` command. Once we see how to use `display` in this way we'll discuss a few shortcuts to using the `plot` command to graph several solutions simultaneously.

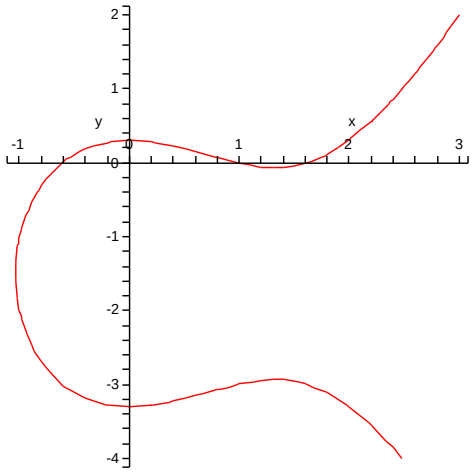

Figure 4: Output of command (5)

4.1 The `display` Command

As with `plot` and `implicitplot`, the `display` command can only be used after loading the `plots` package (see above). The easiest way to use `display` is to compute your plots first, store them as variables and then tell `display` to show them to you.

Lets look at an example. Lets plot the direction field and several solutions to

$$\frac{dy}{dt} = \frac{t+2}{y^3+3y-1}. \tag{6}$$

Start by plotting the direction field and storing the result in the variable `p1` (for "plot 1"). Type

```
p1 := dfieldplot(diff(y(t),t) = (t + 2)/(y(t)^3 + 3*y(t) - 1),
  y(t), t = -3..3, y = -3..3, arrows = line, color = black):
```
(7)

and hit "enter." Notice that the color of the field has been set to black (so it will contrast with our solution curves) and that the input is ended with a colon. This prevents Maple from actually displaying the plot right now, since youre going to look at it later. The differential equation (6) is separable so you know how to find the solutions (go ahead, its good practice). They satisfy

$$y^4 + 6y^2 - 4y = 2t^2 + 8t + c.$$

Since this is an implicit equation, you need to use `implicitplot` to plot the curve it represents. Lets do this for two different choices of c, say $c = -1$ and $c = 7$. As with the direction field, we store the plots in variables for later use by `display`. Type

```
p2 := implicitplot(y^4 + 6*y^2 - 4*y = 2*t^2 + 8*t - 1, t = -3..3, y = -3..3):
```

and hit "enter", then type

```
p3 := implicitplot(y^4 + 6*y^2 - 4*y = 2*t^2 + 8*t + 7, t = -3..3, y = -3..3):
```

followed by "enter." Again, notice the colon suppressing the output of these computations.

Finally, you can put the three plots together with `display` by typing

```
display(p1,p2,p3);
```
(8)

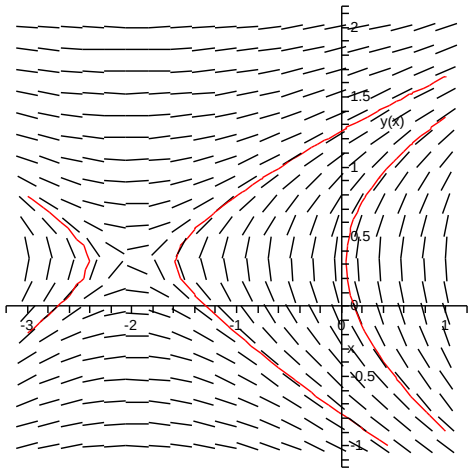

Figure 5: Output of command (8)

and hitting "enter." The resulting output is shown in Figure 5. If the curves dont stand out enough against the direction field, go back to the `implicitplot` commands you have entered earlier and try changing the `thickness` setting. And if you dont like the view youre getting, go back and change (all of) the x and y ranges.

You can overlay as many plots as you would like with display. If, for example, you had generated plots `p1, p2, p3, p4, p5` and `p6`, you could put them all together by typing

```
display(p1,p2,p3,p4,p5,p6);
```

and pressing "enter."

4.2 Multiple Graphs with `plot`

In the preceding examples we plotted several curves simultaneously by storing each in a separate variable and then using `display` to put them all together. As you can imagine, if you've got quite a few curves you want to see, this procedure can become rather tedious. You can easily find yourself entering the same line of input, with small modifications, over and over again: the viewing window and thickness might stay the same each time, but the functions and colors might be different.

Fortunately the `plot` and `implicitplot` commands are designed to accept several functions at once! To plot several functions (on the same graph) with a *single* call to `plot`, simply replace the single function you would normally enter with a comma separated list of functions enclosed in brackets (`[ and ]`). For example, if we wanted to plot the functions $\sin x$ and $\cos x$ together we would type

```
plot([sin(x), cos(x)], x = 0..4*Pi);
```  (9)

and press "enter." [1] This produces the output displayed in Figure 6. Note that $\sin x$ has been plotted in red while $\cos x$ is plotted in green. If we chose to plot more than 2 functions, Maple would continue to provide different colors for each function. Should you want all of your curves to be the same color, simply use the `color` option as described above.

Almost everything we have said above applies equally well to the `implicitplot` command. The only exception is that Maple does *not* automatically provide different colors for the different curves drawn in this case.

[1] The constant $\pi = 3.14159\ldots$ is stored in Maple under the variable name `Pi`. Note the capitalization.

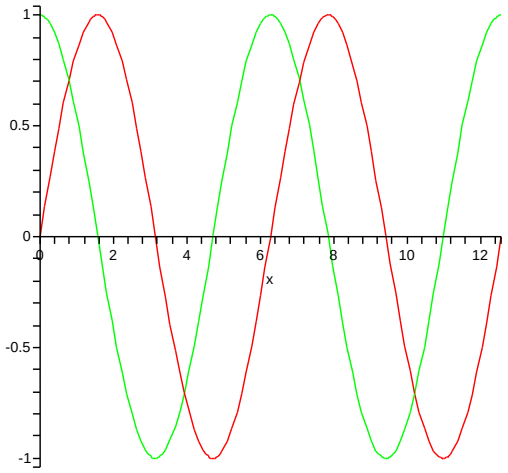

Figure 6: Output of command (9)

Although passing a list of functions to a single call of the `plot` command is a nice short-cut to plotting several graphs at once, there are times when even this can be tedious. In Math 23 we will usually be interested in plotting several solutions to a single differential equation. As we have seen, these solutions offer differ only by a constant, so plotting several of them at once amounts to typing the same formula over and over again, changing only one symbol each time. Fortunately for us Maple provides a simple way to input such a list.

Consider the task of entering the functions

$$y = \frac{t^2}{4} - \frac{t}{3} + \frac{1}{2} + \frac{c}{t^2}$$

for several values of the variable c, say $c = -1, -1/3, 0, 1/3, 1$. Start by creating a list containing all of the values of c we want to use and storing to a variable called `L` (for "list"). Do this by typing

```
L := {-1, -1/3, 0, 1/3, 1}:
```

and pressing "enter." We have chosen to suppress the output of this command (using `:`) since there's no need for us to see the list again. We now use the `seq` command, which is short for "sequence." Type

```
seq(t^2/4 - t/3 + 1/2 + c/t^2, c in L);
```

followed by "enter." Maple outputs

$$\frac{1}{4}t^2 - \frac{1}{3}t + \frac{1}{2} - \frac{1}{t^2}, \frac{1}{4}t^2 - \frac{1}{3}t + \frac{1}{2}, \frac{1}{4}t^2 - \frac{1}{3}t + \frac{1}{2} + \frac{1}{t^2}, \frac{1}{4}t^2 - \frac{1}{3}t + \frac{1}{2} + \frac{1}{3t^2}, \frac{1}{4}t^2 - \frac{1}{3}t + \frac{1}{2} - \frac{1}{3t^2}$$

which, aside from a reordering of the values of c, is the list of functions we wanted!

So now let's use `seq` and `plot` together to (quickly!) graph the functions

$$y = \frac{t^2}{4} - \frac{t}{3} + \frac{1}{2} + \frac{c}{t^2}$$

for the values $c = -1, -1/3, 0, 1/3, 1$, simultaneously. Start by creating the list of c's as above: type

```
L := {-1, -1/3, 0, 1/3, 1}:
```

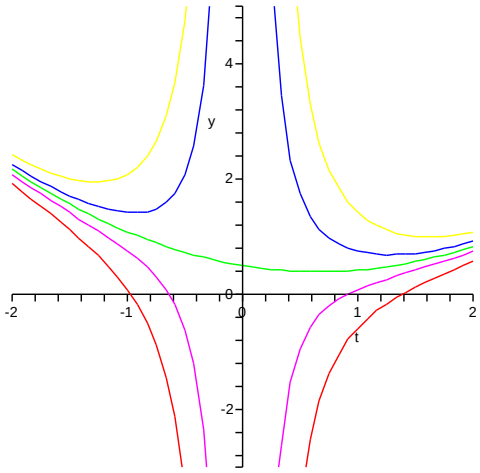

Figure 7: Output of command (10)

and hit "enter." Now put the `seq` command into the `plot` command where the list of functions would normally go, being sure to enclose it in brackets: type

```
plot([seq(t^2/4 - t/3 + 1/2 + c/t^2,c in L)], t = -2..2, y = -3..5);
```
(10)

and hit "enter." The result is shown in Figure 7.

In the same way, `seq` can be used in combination with `implicitplot` to quickly plot "families" of implicitly defined curves. Try it out by plotting the curves

$$y^2 - y = x^3 + c(x^2 + 1)$$

for the values $c = -2, -1, 0, 1, 2$.

5 The `sum` Command

Suppose you want to use Maple to compute the following sum:

$$\sum_{k=1}^{20} \frac{k^2 + 3}{e^k}.$$

One (very inefficient) way to do this would be to type all 20 of the terms in this sum

`(1^2 + 3)/exp(1) + (2^2 + 3)/exp(2) + (3^2 + 3)/exp(3) + (4^2 + 3)/exp(4) +` *etc.*

and hit enter. This, naturally, would take forever, and should there be a typo in your input it would be very difficult to spot. Fortunately Maple provides the command `sum` to get around this problem. As usual, we proceed by example. The sum considered above would be entered as

```
sum((k^2 + 3)/exp(k), k=1..20)
```

The sum

$$\sum_{n=1}^{1000} \frac{1}{n^2}$$

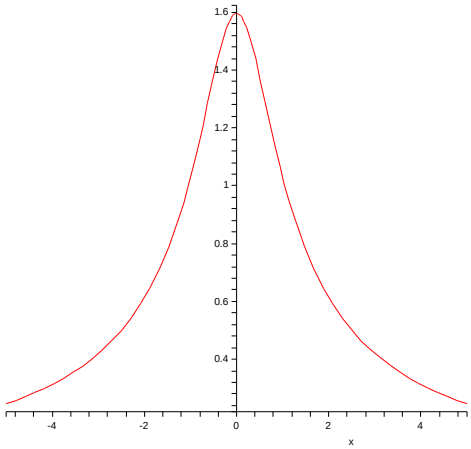

Figure 8: Output of command (11)

would be entered as

```
sum(1/n^2, n=1..1000)
```

As is clear from these two examples, the first quantity that comes after the word `sum` is the formula for the general term of the series. Next we indicate the indexing variable and its range. In the first example `k=1..20` tells Maple the indexing variable is k and that it should add the terms from $k = 1$ to $k = 20$. In the second example `n=1..1000` tells Maple the indexing variable is n and that it should add the terms from $n = 1$ to $n = 1000$. You are, of course, free to use any indexing variable you like, and the range need not start with 1 every time.

Since you explicitly tell Maple which variable is the indexing variable for the summation, you can use any other variables in the expression for the general term that you like. This is very useful for plotting functions given by series. For example, the 20^{th} partial sum of the series

$$\sum_{n=1}^{\infty} \frac{1}{x^2 + n^2}$$

would be entered into Maple as

```
sum(1/(x^2 + n^2), n=1..20)
```

This expression is perfectly valid as an argument to the `plot` command. Thus, if we wanted to plot the function given by the 20^{th} partial sum of the series above, on the interval $-5 \leq x \leq 5$, we would type

```
plot(sum(1/(x^2 + n^2), n=1..20), x=-5..5);
```
(11)

and hit enter. The result is displayed in Figure 8

6 Autonomous Systems of ODEs

There are several ways to use Maple to visualize solutions to autonomous systems of first order ODEs. Some of these make use of the tools we have already introduced but with modified syntax. There is also another command, `phaseportrait`, that can be used to accomplish all of our visualizations "at once," and it is especially useful when we cannot find explicit solutions to the system at hand.

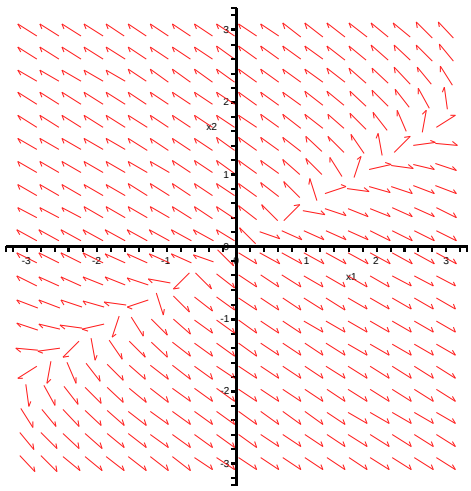

Figure 9: Output of command (12)

6.1 Direction Fields Revisited

While we only used it previously to generate direction fields for first order autonomous *equations*, it is also possible to get `dfieldplot` to draw direction fields for first order autonomous *systems* as well. As an example, consider the homogeneous constant-coefficient system

$$\begin{aligned} x_1' &= 2x_1 - 3x_2 \\ x_2' &= -x_1 + 2x_2 \end{aligned}$$

which can also be written in matrix notation as

$$\mathbf{x}' = \begin{pmatrix} 2 & -3 \\ -1 & 2 \end{pmatrix} \mathbf{x}.$$

To draw the direction field for this system in the region $-3 \le x_1 \le 3$, $-3 \le x_2 \le 3$ of the phase plane you would type

```
dfieldplot([diff(x1(t),t) = 2*x1(t) - 3*x2(t), diff(x2(t),t) = -x1(t) + 2*x2(t)],
 [x1(t), x2(t)], t = 0..1, x1 = -3..3, x2 = -3..3);                    (12)
```

and hit "enter," the result being displayed in Figure p10.

Much of the syntax of this command is analogous to our previous use of `dfieldplot`. We begin by specifying the equations of the system, enclosing them in a pair of square brackets: `[diff(x1(t),t) = 2*x1(t) - 3*x2(t), diff(x2(t),t) = -x1(t) + 2*x2(t)]`. As before, you use `diff` to indicate differentiation and you must explicitly specify that x_1 and x_2 are functions of t by typing `x1(t)` and `x2(t)` everywhere that they occur. As above, we must specify which variables are independent and which are dependent, and this is the purpose served by the start of the second line, `[x1(t), x2(t)]`. Even though it will not show up in your plot, you must also specify a range of the independent variable. In this case we have used $0 \le t \le 1$, as indicated by `t = 0..1`. In reality, *any* range will suffice. Finally, `x1 = -3..3, x2 = -3..3` specifies the region of the phase (x_1x_2) plane to be shown, which in this case is $-3 \le x_1 \le 3$, $-3 \le x_2 \le 3$. Naturally, you can modify these values as needed to show any part of the phase plane that interests you.

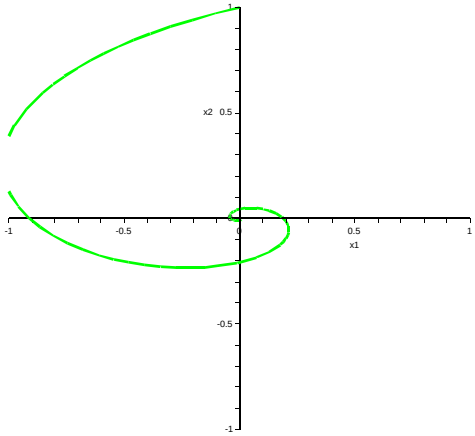

Figure 10: Output of command 13

6.2 Trajectories and `plot`

The trajectories of solutions to a system of differential equations are examples of *parametric curves*, and it is rather easy to modify our eariler use of `plot` to get Maple to draw these as well. In fact, only the part of the `plot` command that specifies the curve to be drawn needs to be modified: the portion specifying the various window, color, thickness, etc. options remains unchanged.

Let's say that you want to plot the parametric curve

$$\mathbf{x}(t) = \begin{pmatrix} -2e^{-t}\sin 2t \\ e^{-t}\cos 2t \end{pmatrix}$$

in the window $-1 \le x_1 \le 1$, $-1 \le x_2 \le 1$. You would type

```
plot([-2*exp(-t)*sin(2*t), exp(-t)*cos(2*t), t = 0..5],
x1 = -1..1, x2 = -1..1, color = green, thickness = 3);        (13)
```

followed by "enter." This produces Figure 10. The first thing just inside the `plot` command is `[-2*exp(-t)*sin(2*t), exp(-t)*cos(2*t),t=0..5]`. This specifies the parametric equations of the curve as well as the t range that Maple should use when drawing it. So in this case, we will see all the points on the curve with t values between 0 and 5. If you like, you can think of the parametric curve as specifying the location of a particle at time t and we are drawing the complete path of the particle as t varies from 0 to 5.

The remainder of the things inside the `plot` command (13) have exactly the same meaning as they did before. They specify the viewing window (`x1=-1..1, x2=-1..1`), the color of the curve (`color = green`) and the thickness of the curve (`thickness = 3`). Moreover, any options you used with `plot` previously can be used in this setting as well, in exactly the same way that you used them before.

6.3 Phase Portraits

A phase portrait for a system of autonomous ODEs is a diagram that includes a direction field for that system as well as several representative solution trajectories. One way to produce such a diagram would be to proceed as follows. Plot the direction field using `dfieldplot` and store the result in a variable. Then, solve the system and use `plot` to produce several trajectories, storing each in a variable. Then use `display`[2] to put all of these pictures together, yielding your phase portrait.

[2]Don't forget: to use `display` you need to load the `plots` library first using the command `with(plots):`

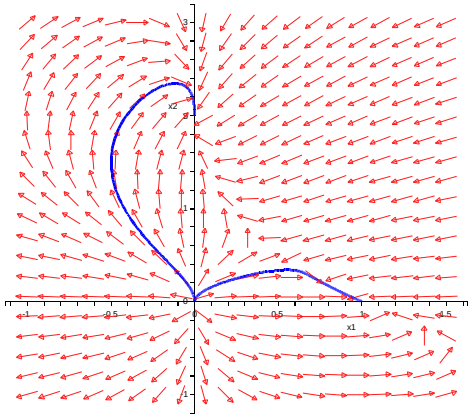

Figure 11: Output of `phaseportrait`

While there is nothing wrong with this approach, it has two shortcomings. The first is that if the solution curves have complicated mathematical expressions, entering them into the `plot` command can be tiresome (and error prone!). The second (and more fundamental) drawback is that it might not be possible to find explicit formulae for the solutions in the first place. Fortunately, the `phaseportrait` command allows you to draw phase portraits of systems of differential equations, whether or not you know the solutions. It simultaneously produces a direction field and as many solution curves as you ask it to. In order to use `phaseportrait`, the `DEtools` package must first be loaded by typing `with(DEtools):` and pressing "enter."

As with the other commands weve talked about, its easiest to understand how to use `phaseportrait` by looking at an example. Suppose we would like to draw a phase portrait for the system

$$\begin{aligned} \frac{dx_1}{dt} &= x_1 - x_1^2 - x_1x_2 \\ \frac{dx_2}{dt} &= \frac{1}{2}x_2 - \frac{1}{4}x_2^2 - \frac{3}{4}x_1x_2. \end{aligned}$$

This system has critical points at $(0,0), (0,2), (1/2,1/2), (1,0)$, as is easily verified. Accordingly, we want to plot the system in a window that doesn't include too much more than the critical points, say $-1 \le x_1 \le 3/2$, $-1 \le x_2 \le 3$. We would type in the following:

```
phaseportrait([diff(x1(t),t) = x1(t) - x1(t)^2 - x1(t)*x2(t),
diff(x2(t),t) = 1/2*x2(t) - 1/4*x2(t)^2 - 3/4*x1(t)*x2(t)],
[x1(t),x2(t)], t=-10..40, [[x1(0)=-.2, x2(0)=.5], [x1(0)=.25, x2(0)=.25]],
x1=-1..3/2, x2=-1..3, arrows = slim, stepsize = .05, linecolor = blue);
```

The output of this command is displayed in Figure 11.

Lets try and decipher this command. The first thing we see inside the parentheses is `[diff(x1(t),t) = x1(t) - x1(t)^2 - x1(t)*x2(t), diff(x2(t),t) = 1/2*x2(t) - 1/4*x2(t)^2 - 3/4*x1(t)*x2(t)]`. These are just the equations of the system. Notice two things: as with `dfieldplot`, the equations of the system are enclosed in a pair of square brackets and instead of `x1` and `x2` we have written `x1(t)` and `x2(t)`, specifying explicitly that x_1 and x_2 are functions of the independent variable t. As we have seen several times now, the syntax for dx/dt is `diff(x(t),t)`.

The next thing we see is `[x1(t),x2(t)]`. This tells Maple (again!) that x_1 and x_2 are functions of the independent variable t, and that the phase portrait should be drawn in the x_1x_2-plane. Notice that, as above, this part is enclosed in square brackets.

Next we come to `t=-10..40`. This is the t range that Maple uses when plotting the solution curves. It is usually a good idea to have the t range include both positive and negative values (here we have $-10 \le t \le 40$).

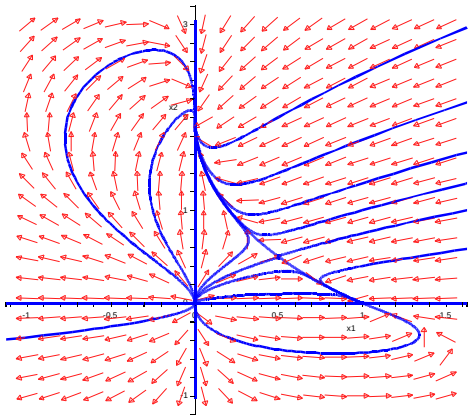

Figure 12: Output of `phaseportrait`

You should try various choices for the range and see what produces the nicest phase portrait for the system at hand. For example, if your solution curves dont seem to be getting very close to what you know is an asymptotically stable point, try extending the t range. You'll see what I mean if you change the range above to `t=-10..10`.

Now we come to what, in many cases, will probably be the longest single part of the command:

```
[[x1(0)=-.2, x2(0)=.5], [x1(0)=.25, x2(0)=.25]]
```

This is a list of initial conditions. Here we are asking Maple to draw two solution curves, one with the initial conditions $x_1(0) = -.2, x_2(0) = .5$ and another with $x_1(0) = .25, x_2(0) = .25$. Notice that each pair of initial conditions is enclosed in square brackets, each pair is separated by a comma and the entire list is enclosed in another set of square brackets. More often than not, two solution curves don't give a very complete phase portrait. Therefore, it is usually necessary to specify (many!) more initial conditions. A list such as

```
[[x1(0)=.5, x2(0)=.1], [x1(0)=0, x2(0)=3], [x1(0)=.25, x2(0)=2.5],
[x1(0)=-.2, x2(0)=.5], [x1(0)=-.5, x2(0)=-.25], [x1(0)=.25, x2(0)=.25],
[x1(0)=-.5, x2(0)=-.25], [x1(0)=.5, x2(0)=.49], [x1(0)=.25, x2(0)=.25],
[x1(0)=.5, x2(0)=.51], [x1(0)=.5, x2(0)=1], [x1(0)=-.25, x2(0)=1],
[x1(0)=-.75, x2(0)=1.5], [x1(0)=.25, x2(0)=.5],[x1(0)=1, x2(0)=1],
[x1(0)=0, x2(0)=.1], [x1(0)=1, x2(0)=.4], [x1(0)=.5, x2(0)=-.5],
[x1(0)=.5, x2(0)=2], [x1(0)=.5, x2(0)=1.4], [x1(0)=1.5, x2(0)=0],
[x1(0)=.5, x2(0)=0], [x1(0)=-.1, x2(0)=0], [x1(0)=0, x2(0)=-.5]]
```

would not be inappropriate for this system. The result of replacing the initial conditions above with this new list is shown in Figure 12.

How should you come up with such a list? By looking at the phase plane and direction field! One technique is to proceed as follows. Use `phaseportrait` to plot the phase plane and direction field with one or two randomly chosen initial conditions. Then look at the picture you get and decide where youd like to see more curves. For example, if you felt like you wanted a solution curve through the point $(1, 1)$ then you would add `[x1(0)=1, x2(0)=1]` to the list of initial conditions. If you wanted a solution curve through the point $(-.5, .25)$ then you'd add `[x(0)=-.5, y(0)=.25]` to the list. It's likely that you'll need to do this several times until you get exactly the picture you want. It may be tedious to do so, but it certainly isn't difficult.

The rest of the stuff in the `phaseportrait` command is pretty straightforward. `x1=-1..3/2` and `x2=-1..3` specify the x_1 and x_2 ranges of the plotting window, respectively. `arrows = slim` tells Maple to draw the direction field using slim arrows. You can leave this option off entirely if you want to (the arrows just look a little different) or you can replace it with `arrows = none` if you dont want the phase portrait to include a direction field. Without getting into the details of how Maple draws the solution curves, lets just say that `stepsize = .05` controls the smoothness of the curves it draws. If you find yourself getting jagged curves, try decreasing this value. In most cases .05 should suffice. Finally, `linecolor = blue` is the color change command. Here we are specifying that the curves should be drawn in blue. Any of the colors mentioned earlier would work in place of `blue`. If you want the arrows in the direction field to be a different color (red is the default) then use the `color` option as above, e.g. add `color=green` for green arrows.

I should point out that if you have a long list of initial conditions and a wide t range then don't be surprised if it takes Maple a little while to draw the phase portrait. To give you some frame of reference, it took Maple roughly 15 seconds (on my office G5 iMac) to draw the phase portrait of Figure 12.