Project 4 Help

In order to use *Mathematica* to explore series solutions of differential equations, we will use the Big-O Notation which comes predefined in *Mathematica*. We begin by defining a generic power series expansion:

```
y[m_, a0_, a1_] := a0 + a1*x + Sum[a[i]*x^i, {i, 2, m}] + O[x]^(m+1)
```

The first two coefficients of the power series (a0 and a1) are given as function arguments while the remaining coefficients are implemented as a list (*a*[*i*]). The argument *m* specifies the maximum order of the power series that *Mathematica* will actually allocate memory for. If we call this function with a specific value for *m*, we find

```
y[10, a0, a1]
```

$$a0 + a1\,x + a[2]\,x^2 + a[3]\,x^3 + a[4]\,x^4 + a[5]\,x^5 + a[6]\,x^6 + a[7]\,x^7 + a[8]\,x^8 + a[9]\,x^9 + a[10]\,x^{10} + O[x]^{11}$$

The Big - O notation at the end of the series tells *Mathematica* that the object is indeed a power series, but only terms up to order 10 have been given explicit coefficients. MAKE SURE YOU TYPE IN A "BIG-OH" AND NOT A ZERO! We will use this to find power series solutions of the differential equation

$$\left(2 + x^2\right) \frac{d^2 y}{dx^2} - x\,\frac{dy}{dx} + 4\,y = \cos(x)$$

up to order 100 (which is 101 terms counting a0). We will begin with the initial data *y*(0) = 1, *y*' (0) = 0. Since the initial data give us the coefficients a0 and a1, we will be using the series *y*[100, 1, 0]. To see what is happening in the program, we will begin with a more modest number of terms - 10 terms to be specific.

```
Terms = (2 + x^2) D[y[10, 1, 0], x, x] - x*D[y[10, 1, 0], x] + 4*y[10, 1, 0] == Cos[x]
```

$$(4 + 4\,a[2]) + 12\,a[3]\,x + (4\,a[2] + 24\,a[4])\,x^2 + (7\,a[3] + 40\,a[5])\,x^3 + (12\,a[4] + 60\,a[6])\,x^4 + (19\,a[5] + 84\,a[7])\,x^5 + (28\,a[6] + 112\,a[8])\,x^6 + (39\,a[7] + 144\,a[9])\,x^7 + (52\,a[8] + 180\,a[10])\,x^8 + O[x]^9 == \mathrm{Cos}[x]$$

Notice that we have just taken the differential equation above and inserted *y*[10, 1, 0] for the function y (*Mathematica*'s D[f,x] command instructs the program to differentiate the expression f with respect to x). The output is a power series set equal to the polynomial on the left. To get coefficients, we use the LogicalExpand function followed by the Solve command. The % sign references the last computation *Mathematica* performed (in this case, the series expansion above).

```
Solve[LogicalExpand[Terms]]
```

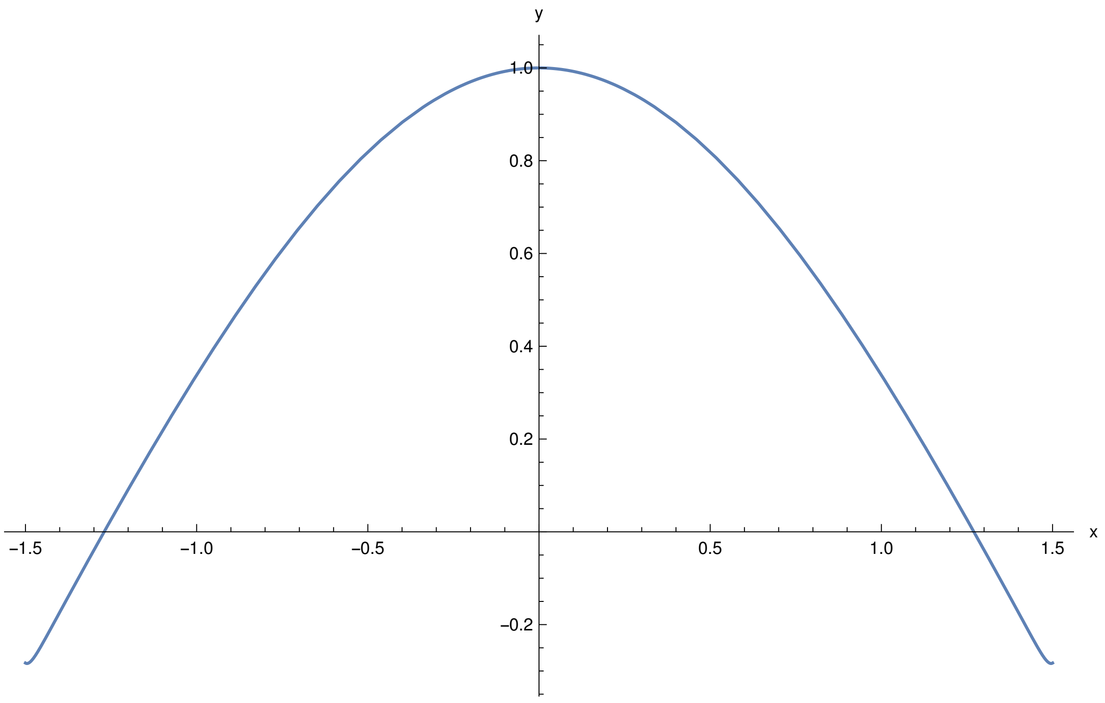
$$\left\{\left\{a[2] \to -\frac{3}{4},\ a[3] \to 0,\ a[4] \to \frac{5}{48},\ a[5] \to 0,\ a[6] \to -\frac{29}{1440},\ a[7] \to 0,\ a[8] \to \frac{9}{1792},\ a[9] \to 0,\ a[10] \to -\frac{10529}{7257600}\right\}\right\}$$

This list gives us replacement rules for all coefficients up to a[10]! Of course, we will not want to see all of the details when we compute 101 coefficients. So, we will repeat the computation above but use semi-colons to suppress the output.

```
TermsA = (2 + x^2) D[y[100, 1, 0], x, x] - x * D[y[100, 1, 0], x] + 4 * y[100, 1, 0] == Cos[x];
A = Solve[LogicalExpand[TermsA]];
```

We have stored the list of replacement rules to the variable A so that we can use them to define a function which we then plot.

```
Y1[x_] := 1 + 0 * x + Sum[a[i] * x^i, {i, 2, 100}] /. A
Plot[Y1[x], {x, -1.5, 1.5}, AxesLabel → {"x", "y"}]
```



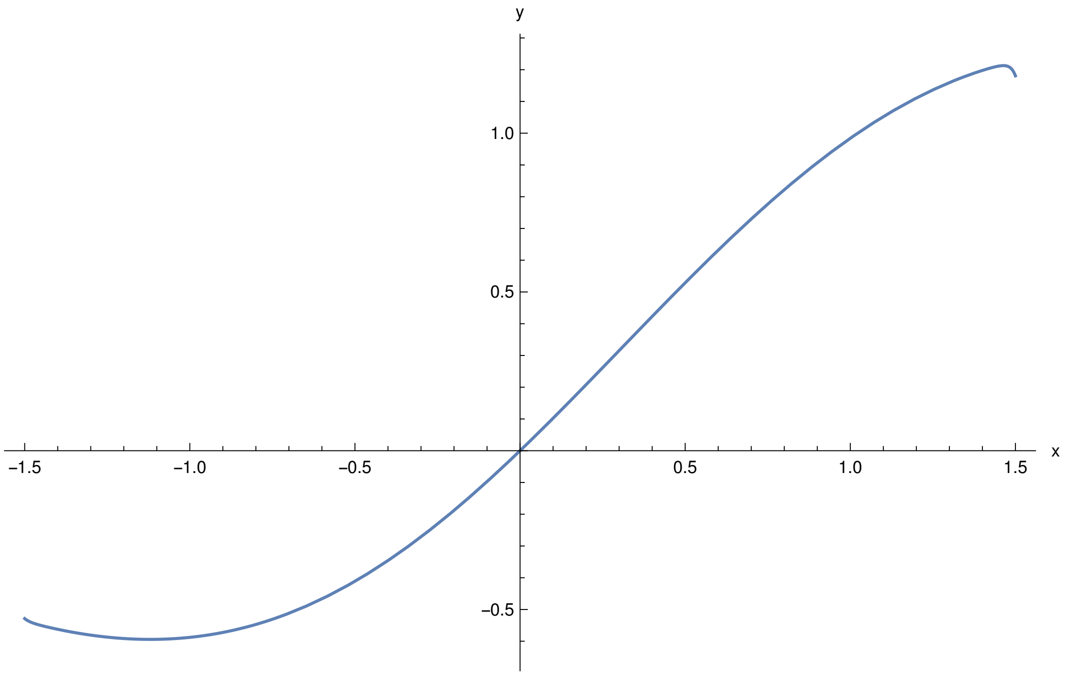
The plot range chosen above ($-1.5 \le x \le 1.5$) was chosen due to the fact that the power series centered at 0 for this differential equation will definitely diverge for $|x| > \sqrt{2}$ (can you see why?). We now repeat all of the computations with the initial data y(0) = 0, y'(0) = 1. Be sure to notice the changes in the functions below. First, we print out the first ten coefficients explicitly.

```
Terms = (2 + x^2) D[y[10, 0, 1], x, x] - x * D[y[10, 0, 1], x] + 4 * y[10, 0, 1] == Cos[x];
Solve[LogicalExpand[Terms]]
```

$$\left\{\left\{a[2] \to \frac{1}{4},\ a[3] \to -\frac{1}{4},\ a[4] \to -\frac{1}{16},\ a[5] \to \frac{7}{160},\ a[6] \to \frac{19}{1440},\ a[7] \to -\frac{19}{1920},\ a[8] \to -\frac{89}{26\,880},\ a[9] \to \frac{247}{92\,160},\ a[10] \to \frac{6943}{7\,257\,600}\right\}\right\}$$

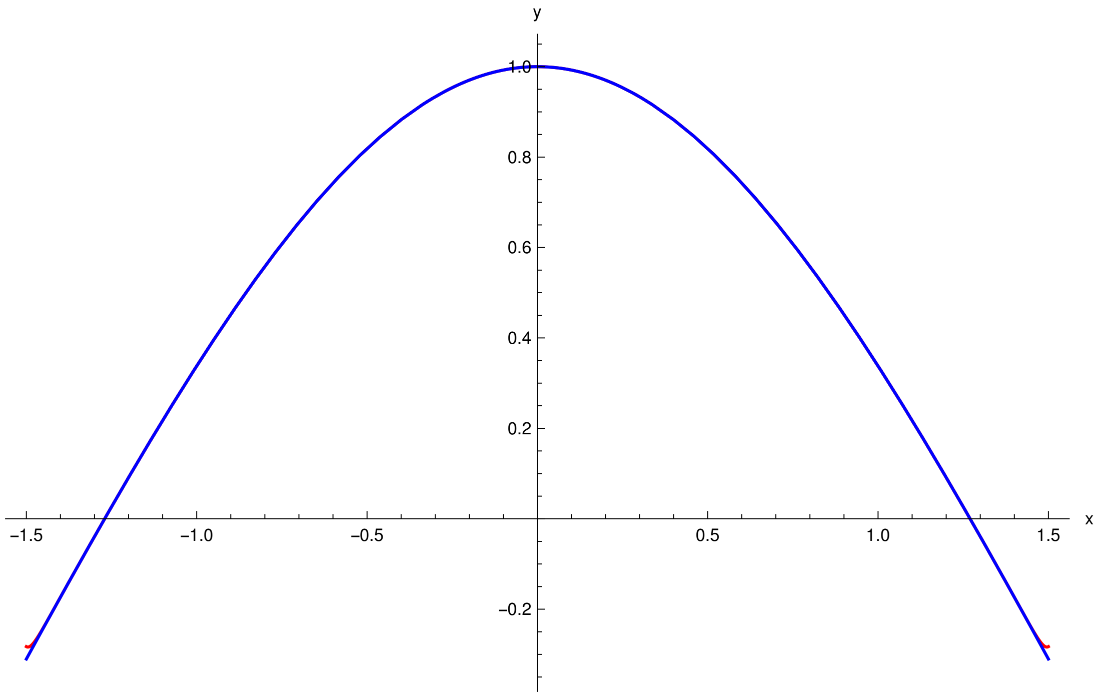
Next, we find the first 101 coefficients, and use them to give a plot of the function.

```
TermsB = (2 + x^2) D[y[100, 0, 1], x, x] - x * D[y[100, 0, 1], x] + 4 * y[100, 0, 1] == Cos[x];
B = Solve[LogicalExpand[TermsB]];
Y2[x_] := 0 + 1 * x + Sum[a[i] * x^i, {i, 2, 100}] /. B
Plot[Y2[x], {x, -1.5, 1.5}, AxesLabel → {"x", "y"}]
```



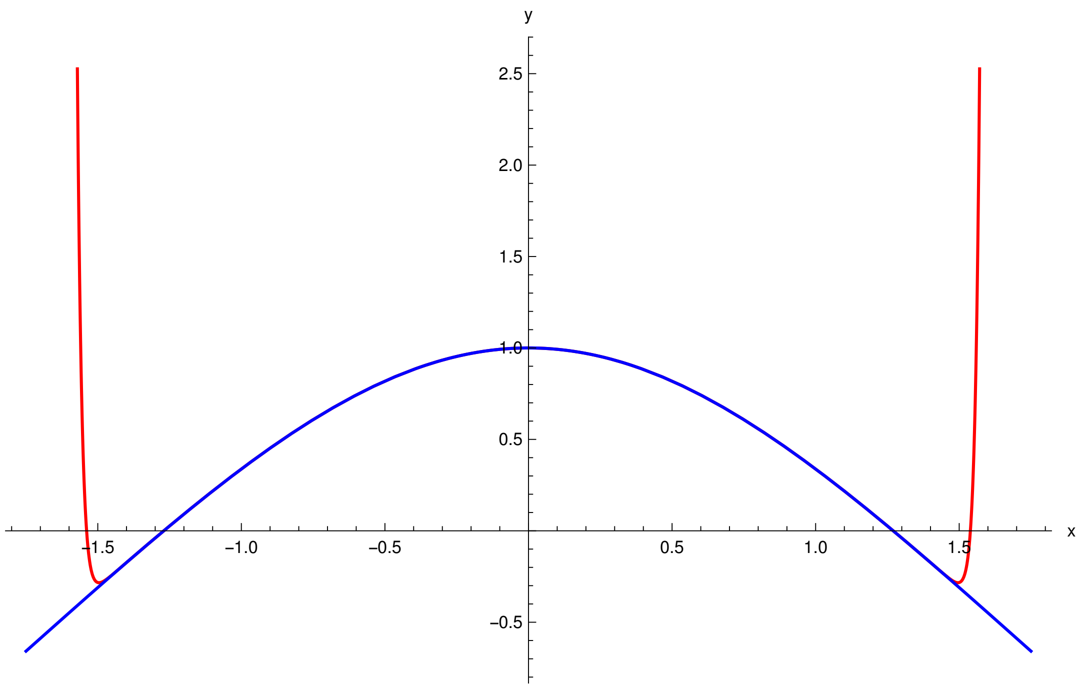
This is a good place to mention the limitations of power series. Power series solutions to differential equations often converge fairly rapidly around their center (here $x = 0$). However, if a power series only has a finite radius of convergence, the behavior can be quite bad at the boundary of the interval of convergence. To see this, we will compare the solution from our power series approach with *Mathematica*'s numerical solution with NDSolve.

```
A1 = NDSolve[{(2 + x^2) * Y''[x] - x * Y'[x] + 4 * Y[x] == Cos[x], Y[0] == 1, Y'[0] == 0},
    Y[x], {x, -10, 10}];
Plot[{Y1[x], Evaluate[Y[x] /. A1]}, {x, -1.5, 1.5},
 AxesLabel → {"x", "y"}, PlotStyle → {Red, Blue}]
```



The two approximations to the solution (i.e. our 100 terms of the power series and *Mathematica*'s numerical solution from NDSolve) agree quite well! Note however that the two solutions seem to diverging after about 1.45 units from the origin. If we expand the plot region a bit, the difference become dramatic.

```
Plot[{Y1[x], Evaluate[Y[x] /. A1]}, {x, -1.75, 1.75},
 AxesLabel → {"x", "y"}, PlotStyle → {Red, Blue}]
```



The blue curve is *Mathematica*'s numerical approximation. Note that our power series looks as though it will become divergent, and in fact it is possible to show that the power series cannot converge outside the interval $|x| \leq \sqrt{2}$. So, if we want a decent representation of the solution outside this interval, the power series solution will not work! Of course, *Mathematica*'s NDSolve command has error associated with it also! But, the programmers have written very sophisticated algorithms into NDSolve which try to minimize the error as much as possible over the specified interval (in this case $-10 \leq x \leq 10$). Always be aware of the limitations of whatever approximation scheme you choose to implement!