Plotting Graphs in LaTeX Using LyX and PGFPlots

Steve Smith

21 February 2013 : Version 0.04

Abstract

LyX is a document processor that encourages an approach to writing based on the structure of your documents (WYSIWYM) and not simply their appearance (WYSIWYG).

LyX combines the power and flexibility of TeX/LaTeX with the ease of use of a graphical interface. This results in world-class support for creation of mathematical content (via a fully integrated equation editor) and structured documents like academic articles, theses, and books. In addition, staples of scientific authoring such as reference list and index creation come standard. But you can also use LyX to create a letter or a novel or a theatre play or film script. A broad array of ready, well-designed document layouts are built in.

LyX is for people who want their writing to look great, right out of the box. No more endless tinkering with formatting details, "finger painting" font attributes or futzing around with page boundaries. You just write. On screen, LyX looks like any word processor; its printed output — or richly cross-referenced PDF, just as readily produced — looks like nothing else.

1 Table of Contents

Contents

2 Document History

Date	Version	Comments
28 September 2012	0.01	The initial version of the document.
9 October 2012	0.02	Removing an introduction to LyX; improving the initial tutorials; putting the main options into tables.
18 October 2012	0.03	General tidying. Correcting the pin style error. Correcting outer usage for legends.
21 February 2013	0.04	Adding an example of drawing a graph parametrically.

Part I
Using LyX

3 YouTube Video Introductions

This document is not intended to provide an introduction to LyX. That can be found elsewhere. The LyX website is a good place to start, obviously, but to get a flavour of LyX if you are new to it, there are some really good YouYube videos that provide a good introduction to using LyX. Check them out. In a very brief search I have found the following two series. They'll get you going.

3.1 Dr. Ed Deveney : Bridgewater State University, Bridgewater, Massachusetts, USA

- Part 1 (of 2)
- Part 2 (of 2)

3.2 Dave Hall : Maths Teacher, El Paso, Texas

- Part 1 (of 5)
- Part 2 (of 5)
- Part 3 (of 5)
- Part 4 (of 5)
- Part 5 (of 5)

Part II
Plotting Graphs in LyX

4 Preliminaries

The main essence of this document is to provide an introduction to drawing graphs of functions in a LyX document as simply as possible.

In order to plot graphs in LyX, we need to use the PGFPlots package that is already bundled in the LaTeX distribution that you will be running once you have LyX working. That means that there is nothing else to download and install: we can just use it.

And it's very simple to use. If you want to plot a graph in a LyX document, you must first go to:

Document ⇒ Settings ⇒ Latex Preamble

and type into the big white space the following text:

\usepackage{pgfplots}

and click OK.

Quite simply, this instruction tells LyX that in this document, you want to use the PGFPlots package from your LaTeX distribution. That's not so bad, is it?

There is a general rule here: in LaTeX, software libraries are called "packages". And if you want to use a package that is not part of the standard LaTeX bundle, you have to download it, put it somewhere LaTeX can find it, and tell LaTeX that you want to use it by inserting the appropriate "\usepackage{}" statement in the preamble.

Later on in this document, we will be adding more lines to our preamble when we want to use different packages.

5 Plotting Graphs in LyX

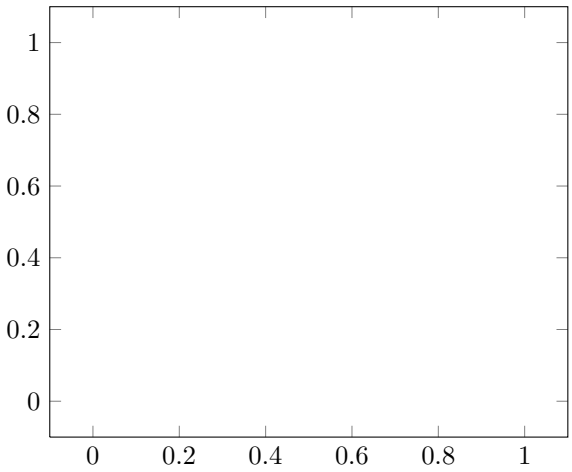
5.1 How to Insert a PGFPlots Graph in LyX

PGFPlots graphs are inserted into LyX documents as TeX. To insert TeX into a LyX document, either click on the "TeX" toolbar button, or use the appropriate keyboard shortcut. You can find the appropriate keyboard shortcut if you hover your mouse over the "TeX" toolbar button. On the Mac (which is what I use), for example, it's COMMAND-L.
Once you've created a TeX block, type this into it:

```
\begin{tikzpicture}
    \begin{axis}
    \end{axis}
\end{tikzpicture}
```

This is the basis of the graph. Don't worry about what the "\begin{tikzpicture}" and the "\end{tikzpicture}" do. We just need them.

Now, the "\begin{axis}" and "\end{axis}" statements define what the axes on your graph will look like. With what we have above, we will have normal linear axes on our graph. [It is also possible to have log-normal and log-log axes. They will be covered later.] OK, so what sort of graph will you get by simply typing this basic stuff? Well, this is what you get:



We have a box, with axes, and not much else. That of course, is because you haven't said what you want to plot! And what the axis labels will be etc. etc.

5.2 Drawing the Tangent to a Function at a Point

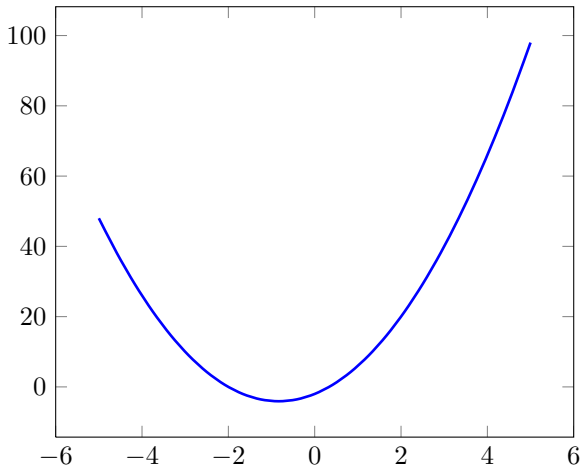
Right, so let's try and plot a function. Let's say we wanted to plot the function

$$y = 3x^2 + 5x - 2$$

So how do we do this? Well, add the following to your TEX:

```
\begin{tikzpicture}
   \begin{axis}
      \addplot [blue, line width = 1, smooth] {3*x^2 + 5*x - 2};
   \end{axis}
\end{tikzpicture}
```

and this is now what the graph becomes:

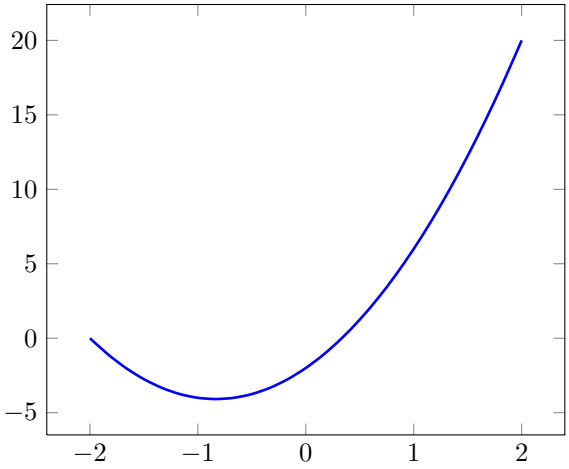


Notice that even though you haven't specified what the range of x or y values will be, it has still drawn the graph of this function! Cool! And as you can see, we have drawn a blue line, with a particular thickness (I'll leave you to experiment with changing those variables), which is smooth (normally what you want).

OK, so let's choose a range of values of x (known in the trade as the *domain*) that we want to draw. Let's say that we want the function to go from x = -2 to x = 2.

```
\begin{tikzpicture}
    \begin{axis}
        \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
    \end{axis}
\end{tikzpicture}
```

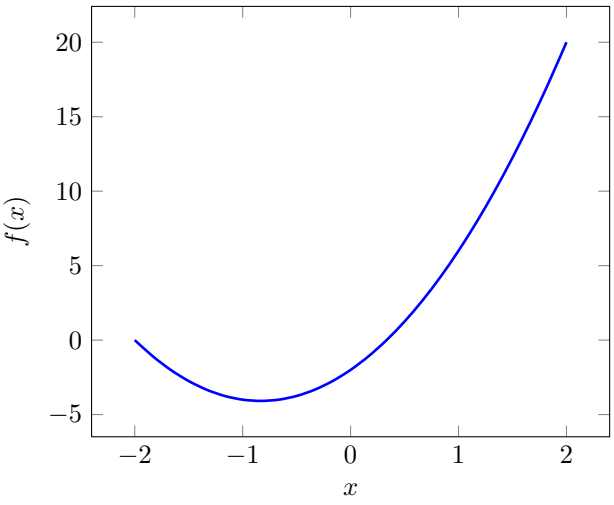
And this is now what the graph becomes:



So now, let's add labels to our axes:

```
\begin{tikzpicture}
    \begin{axis} [xlabel=$x$, ylabel=$f(x)$]
        \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
    \end{axis}
\end{tikzpicture}
```

And we get:

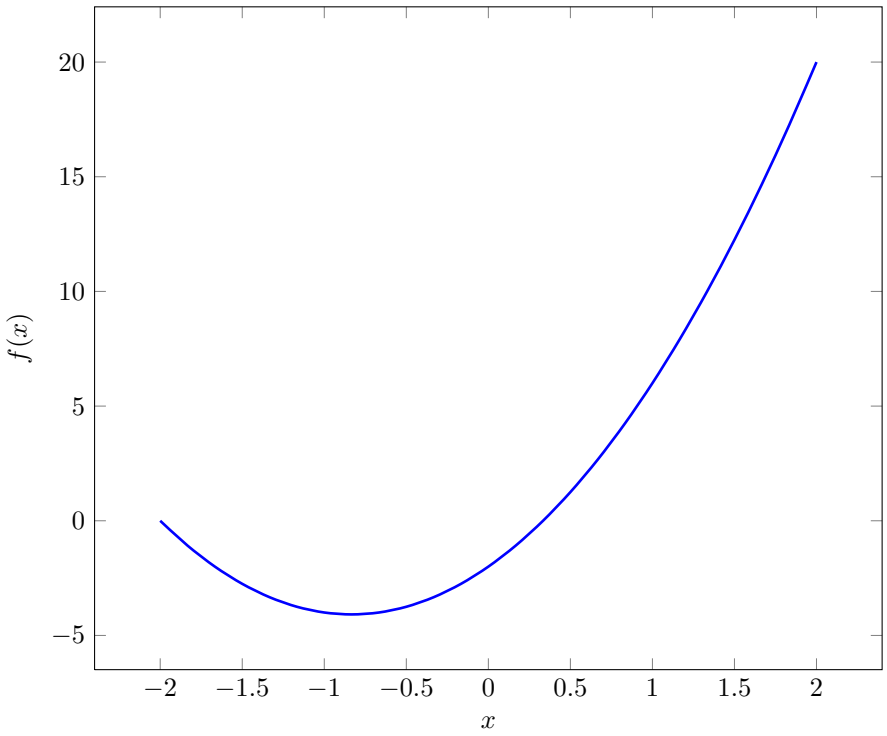


Notice that the *options* for the axes go in square brackets after the *\begin{axis}* command.

Maybe we want the picture to be a bit bigger:

```
\begin{tikzpicture}
   \begin{axis} [xlabel=$x$, ylabel=$f(x)$, width=12cm]
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
   \end{axis}
\end{tikzpicture}
```

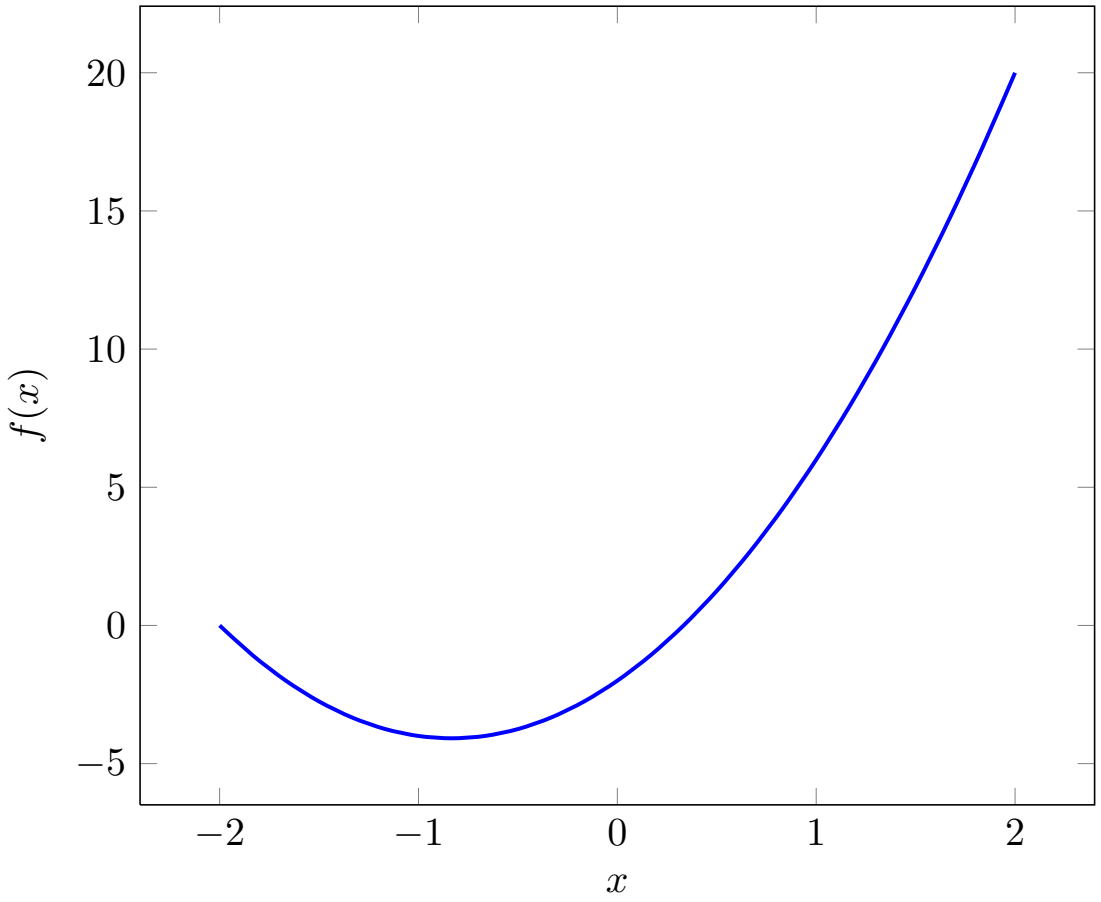
when we get:



Or scaled:

```
\begin{tikzpicture} [scale=1.5]
   \begin{axis} [xlabel=$x$, ylabel=$f(x)$, width=10cm]
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
   \end{axis}
\end{tikzpicture}
```

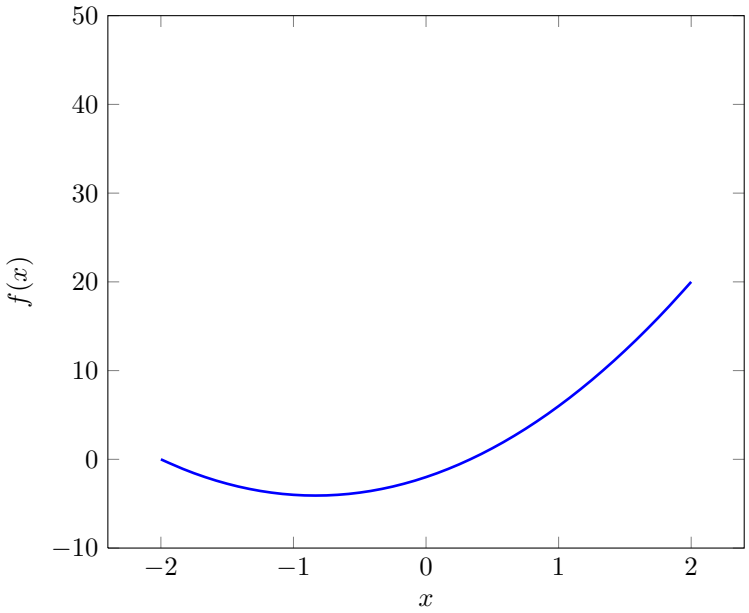
when we get:



Setting the size of the graph using *width* (and *height*), then the graph is resized, but all the texts are kept the same size. Changing the *scale* affects the font sizes, too.

What if we want a particular range of values of y? No problem!

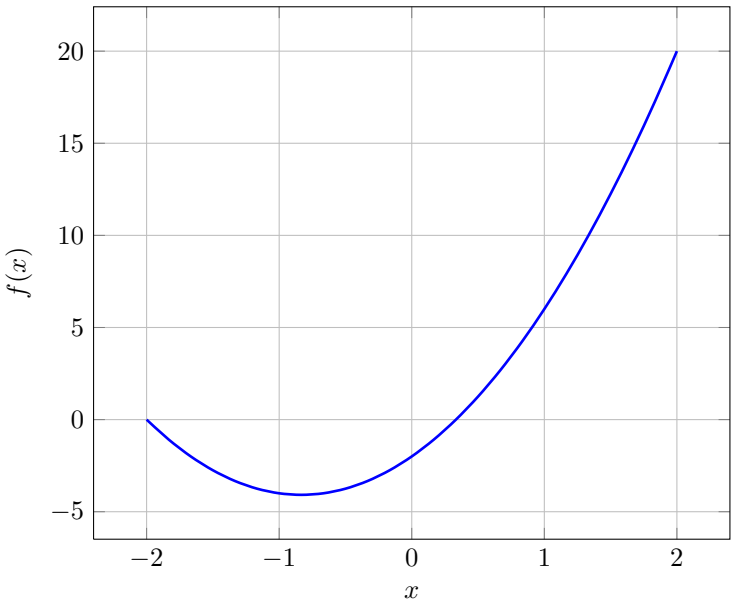
```
\begin{tikzpicture} [scale=1.0]
   \begin{axis} [xlabel=$x$, ylabel=$f(x)$, width=10cm, ymin=-10, ymax=50]
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
   \end{axis}
\end{tikzpicture}
```



Well, that seems easy enough. I was happier with it when the graph sorted it's own y-range out though! So I'm going to go back to that!

Right. Let's put a background grid in:

```
\begin{tikzpicture} [scale=1.0]
   \begin{axis} [xlabel=$x$, ylabel=$f(x)$, width=10cm, grid=major]
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
   \end{axis}
\end{tikzpicture}
```

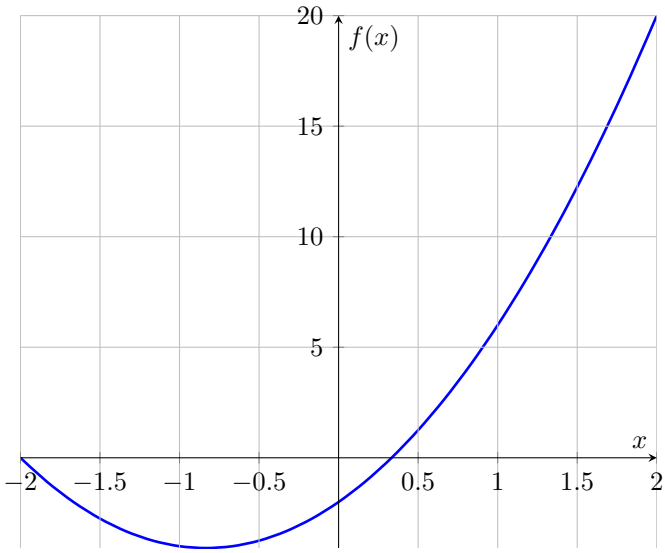


It would be nice if we could emphasize the x- and y-axes, wouldn't it?

```
\begin{tikzpicture} [scale=1.0]
   \begin{axis}
      [
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=true,
         axis x line=middle,
         axis y line=middle
      ]

      % Draw the function
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
   \end{axis}
\end{tikzpicture}
```

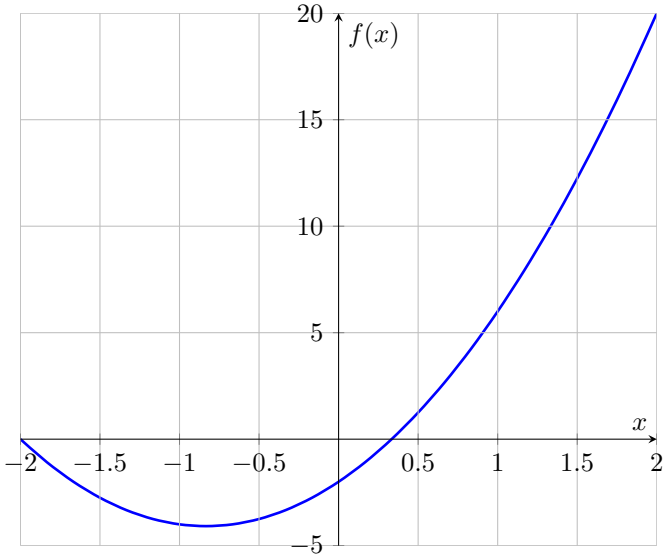
Notice that as my list of axis options is getting larger, I have put each of them on a separate line. That way I can read them more easily. *axis on top* makes sure that your axes are drawn on top of anything else that's drawn in the graph. *axis x line* indicates where the x-axis will be drawn. Using the *middle* value for this option draws it along the y=0 line. There are other values this option can take. See later. The same applies to the *axis y line* option. Using the *middle* value draws this axis along the x=0 line.

Notice also that by the use of the axis options the box has gone, and the axis labels are nicely adjacent to the axes themselves. We are seeing the kind of graph that we were perhaps expecting from the start. We are getting there!



I've noticed though that the minimum of the graph is a bit thin, so I'm going to specify a *ymin* value.

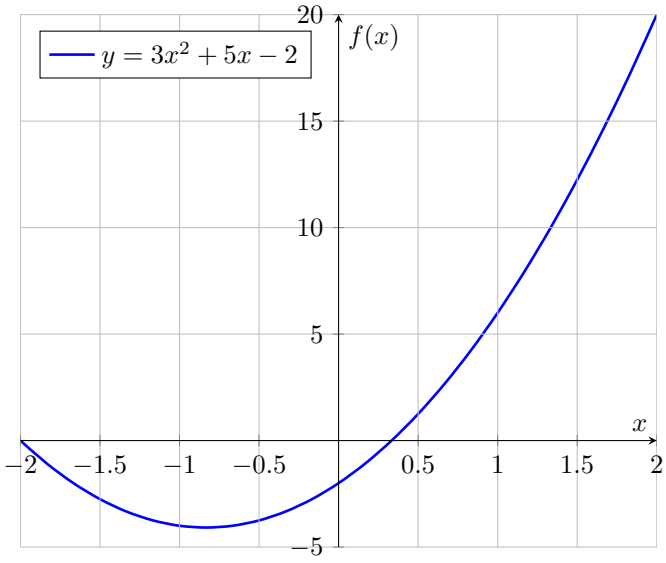
```
\begin{tikzpicture} [scale=1.0]
   \begin{axis}
      [
         ymin=-5,
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=true,
         axis x line=middle,
         axis y line=middle
      ]

      % Draw the function
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
   \end{axis}
\end{tikzpicture}
```



Lovely!

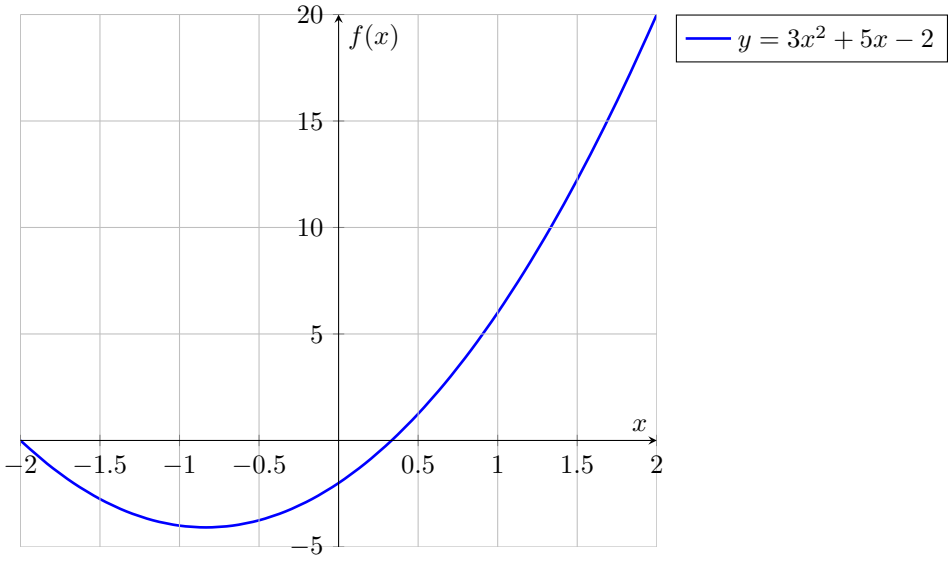
It would be nice if we had a legend!

```
\begin{tikzpicture} [scale=1.0]
   \begin{axis}
      [
         ymin=-5,
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=true,
         axis x line=middle,
         axis y line=middle,
         legend pos=north west
      ]

      % Draw the function
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};

      % And finally the legend
      \legend {$y = 3x^2 + 5x - 2$};

   \end{axis}
\end{tikzpicture}
```



If you feel that the legend is in the way when it is positioned on the graph, you can always have it outside the graph by specifying *outer north east* as the legend's position:

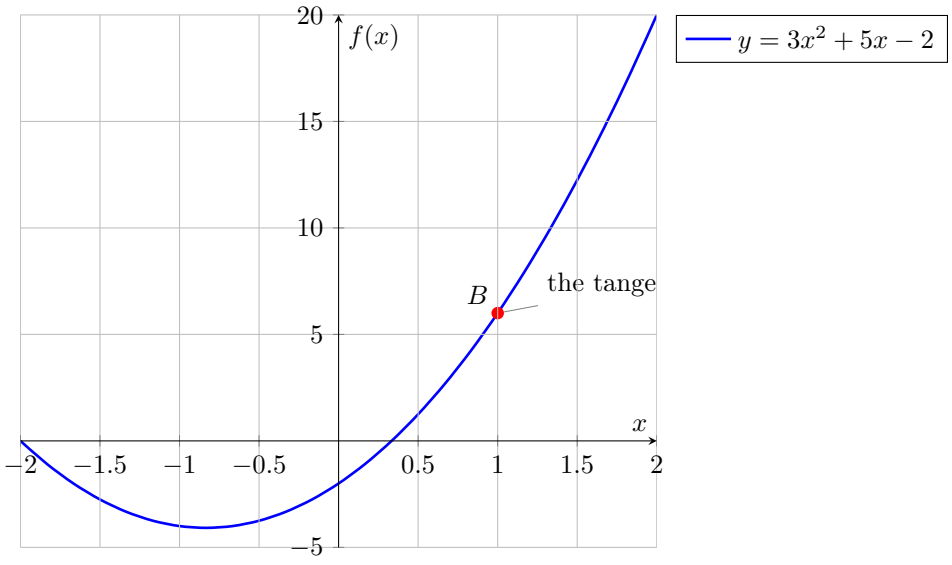
```
\begin{tikzpicture} [scale=1.0]
   \begin{axis}
      [
         ymin=-5,
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=true,
         axis x line=middle,
         axis y line=middle,
         legend pos=outer north east
      ]

      % Draw the function
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};

      % And finally the legend
      \legend {$y = 3x^2 + 5x - 2$};

   \end{axis}
\end{tikzpicture}
```



This is starting to look good! Let's plot the point (1,6), give it a name, and a bit of blurb:

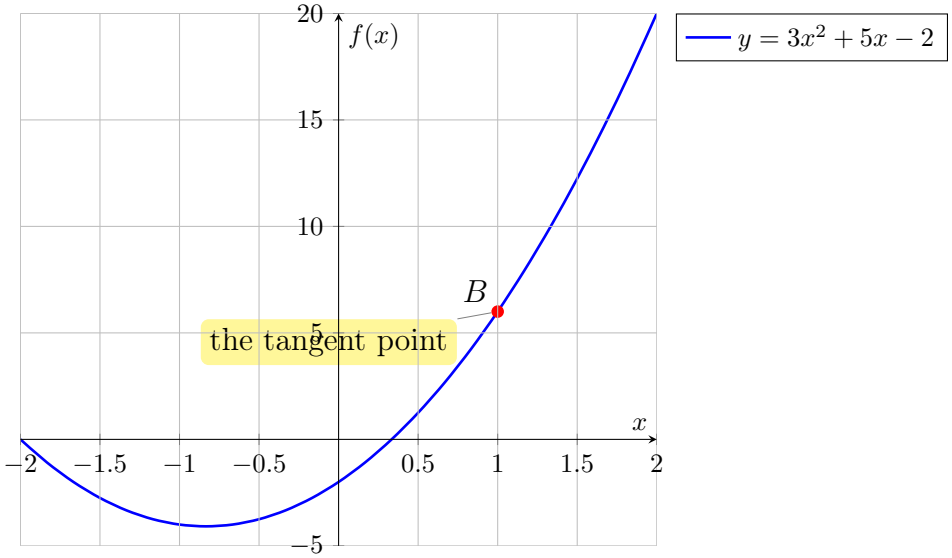
```
\begin{tikzpicture} [scale=1.0]
   \begin{axis}
      [
         ymin=-5,
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=true,
         axis x line=middle,
         axis y line=middle,
         legend pos=outer north east
      ]

      % Draw the function
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};

      % And now the interesting point
      \node [fill=red, circle, scale=0.5, pin=10:{the tangent point}] at (axis cs: 1,6) {};
      \node [above left] at (axis cs: 1,6) {$B$};

      % And finally the legend
      \legend {$y = 3x^2 + 5x - 2$};

   \end{axis}
\end{tikzpicture}
```

The first three options for the first *\node* command (*fill*, *circle* and *scale*) are pretty self-evident. The *pin* option is much more complicated. Don't worry about this yet, but suffice it to say that a pin option draws a line from the object you are drawing (in this case the filled circle) to a description of the object. The *-70* value indicates where the line gets drawn. It's associated to the angle you want the line to be drawn at.



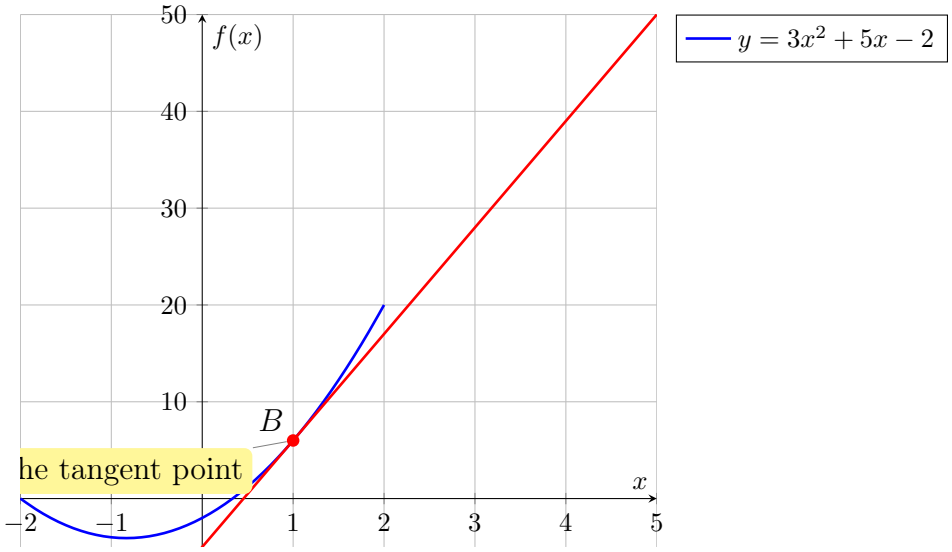
Now let's add a touch of *style*, and move the text on the *pin* around a bit, because we can't see all of it:

```
\begin{tikzpicture} [scale=1.0]

    % The Styles
    \tikzstyle {every pin} = [ fill=yellow!50!white, rectangle, rounded corners=3pt, font=\large ]
    \tikzstyle {smallDot} = [ fill=red, circle, scale=0.5 ]

    % The Graph
    \begin{axis}
        [
            ymin=-5,
            xlabel=$x$,
            ylabel=$f(x)$,
            width=10cm,
            grid=major,
            axis on top=true,
            axis x line=middle,
            axis y line=middle,
            legend pos=outer north east
        ]

        % Draw the function
        \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};

        % And now the interesting point
        \node [smallDot, pin=-170:{the tangent point}] at (axis cs: 1,6) {};
        \node [above left, font=\large] at (axis cs: 1,6) {$B$};

        % And finally the legend
        \legend {$y = 3x^2 + 5x - 2$};

    \end{axis}
\end{tikzpicture}
```



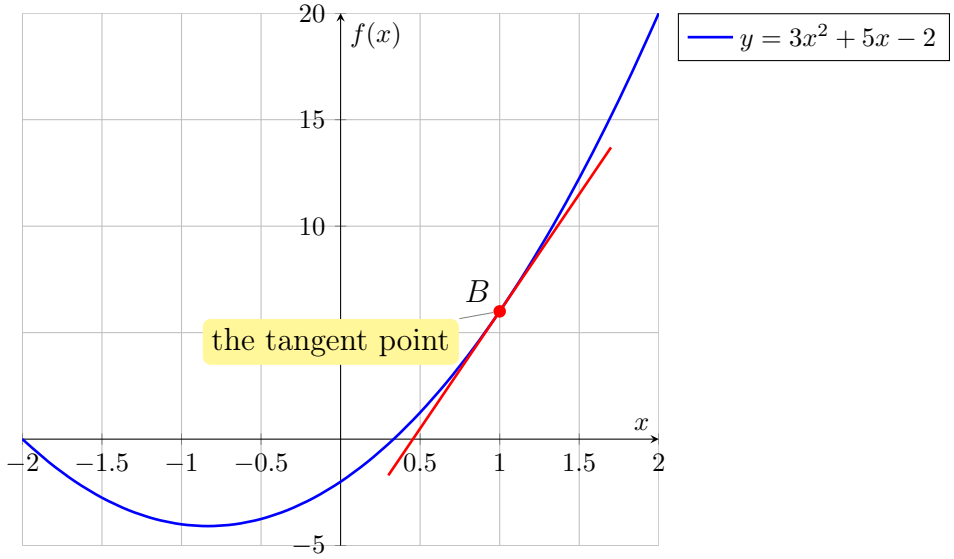
Now let's add the tangent to the curve at B. I want to set the *axis on top* option to *false* as well, so that my text is drawn over the y-axis (rather than the other way around).

```
\begin{tikzpicture} [scale=1.0]
   % The Styles
   \tikzstyle {every pin} = [ fill=yellow!50!white, rectangle, rounded corners=3pt, font=\large ]
   \tikzstyle {smallDot} = [ fill=red, circle, scale=0.5 ]

   % The Graph
   \begin{axis}
      [
         ymin=-5,
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=false,
         axis x line=middle,
         axis y line=middle,
         legend pos=outer north east
      ]

      % Draw the function
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};

      % Draw the Tangent Function
      \addplot [red, line width=1] {11*x - 5};

      % And now the interesting point
      \node [smallDot, pin=-170:{the tangent point}] at (axis cs: 1,6) {};
      \node [above left, font=\large] at (axis cs: 1,6) {$B$};

      % And finally the legend
      \legend {$y = 3x^2 + 5x - 2$};

   \end{axis}
\end{tikzpicture}
```



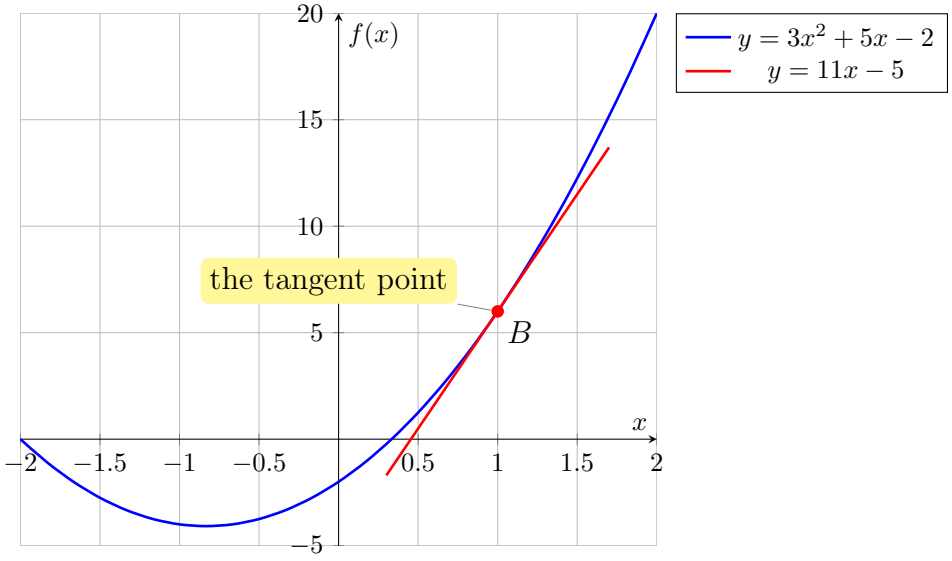
I want my tangent line to be smaller. So I can set the *domain* of the function:

```
\begin{tikzpicture} [scale=1.0]
   % The Styles
   \tikzstyle {every pin} = [ fill=yellow!50!white, rectangle, rounded corners=3pt, font=\large ]
   \tikzstyle {smallDot} = [ fill=red, circle, scale=0.5 ]

   % The Graph
   \begin{axis}
      [
         ymin=-5,
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=false,
         axis x line=middle,
         axis y line=middle,
         legend pos=outer north east
      ]

      % Draw the function
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};

      % Draw the Tangent Function
      \addplot [red, line width=1, domain=0.3:1.7] {11*x - 5};

      % And now the interesting point
      \node [smallDot, pin=-170:{the tangent point}] at (axis cs: 1,6) {};
      \node [above left, font=\large] at (axis cs: 1,6) {$B$};

      % And finally the legend
      \legend {$y = 3x^2 + 5x - 2$};

   \end{axis}
\end{tikzpicture}
```



Finally, let's tweek our point labelling a bit, and add the tangent to the legend:

```
\begin{tikzpicture} [scale=1.0]
   % The Styles
   \tikzstyle {every pin} = [ fill=yellow!50!white, rectangle, rounded corners=3pt, font=\large ]
   \tikzstyle {smallDot} = [ fill=red, circle, scale=0.5 ]

   % The Graph
   \begin{axis}
      [
         ymin=-5,
         xlabel=$x$,
         ylabel=$f(x)$,
         width=10cm,
         grid=major,
         axis on top=false,
         axis x line=middle,
         axis y line=middle,
         legend pos=outer north east
      ]

      % Now draw the functions
      \addplot [blue, line width = 1, smooth, domain=-2:2] {3*x^2 + 5*x - 2};
      \addplot [red, line width = 1, domain=0.3:1.7] {11*x - 5};

      % And now the interesting point
      \node [smallDot, pin=170:{the tangent point}] at (axis cs: 1,6) {};
      \node [below right, font=\large] at (axis cs: 1,6) {$B$};

      % And finally the legend
      \legend {$y = 3x^2 + 5x - 2$, $y = 11x - 5$};
   \end{axis}
\end{tikzpicture}
```



Now I'm quite happy with that!!

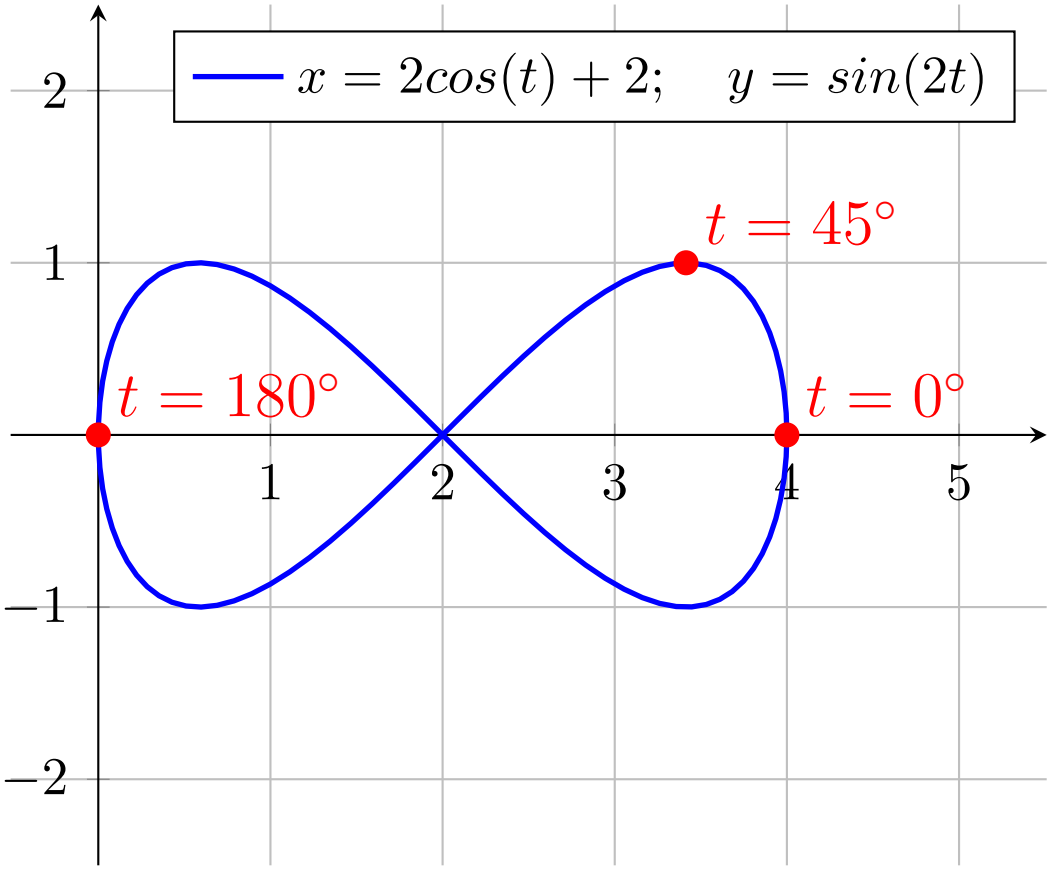
5.3 Drawing Parametric Graphs

Here's an example of how to draw graphs of functions specified parametrically. I'm going to draw the graph of the function defined by:

$$x = 2\cos(t) + 2$$
$$y = \sin(2t)$$

```
\begin{tikzpicture} [scale=2.0]
% The Styles
    \tikzstyle {every pin} = [ fill=yellow!50!white, rectangle, rounded corners=3pt, font=\large ]
    \tikzstyle {smallDot} = [ fill=red, circle, scale=0.5 ]

% The Graph
\begin{axis}
    [
        ymin=-3,
        ymax=3,
        xmin=-1,
        xmax=5,
        grid=major,
        axis on top=false,
        axis x line=middle,
        axis y line=middle,
        axis equal,
        legend pos=north east
    ]

    \def \xradius {2.0};
    \def \yradius {1.0};
    \def \centreX {2.0};
    \def \centreY {0.0};

    % Draw the curve
    \addplot [blue, line width=1, samples=100, domain=0:360]
        % ({the x = 2 cos(t) + 2 bit}, {the y = sin(2t) bit})
        % both defined in terms of x, unfortunately, rather than t!
        ({\xradius * cos(x) + \centreX}, {\yradius * sin(2.0 * x) + \centreY});

    % And now the interesting points
    \node [smallDot] at (axis cs: 4,0) {};
    \node [red, above right, font=\large] at (axis cs: 4,0) {$t=0^{\circ}$};

    \node [smallDot] at (axis cs: 3.414,1) {};
    \node [red, above right, font=\large] at (axis cs: 3.414,1) {$t=45^{\circ}$};

    \node [smallDot] at (axis cs: 0,0) {};
    \node [red, above right, font=\large] at (axis cs: 0,0) {$t=180^{\circ}$};

    % And finally the legend
    \legend {$x=2cos(t)+ 2; \quad y=sin(2t)$};
\end{axis}
\end{tikzpicture}
```

And here's the graph:



Nice!

5.4 Drawing the Shaded Region Under a Function

One of the other common diagrams that we might want to draw is one showing an area under a curve. This kind of thing crops up when you are talking about integration, for example. Let's see how we can do that. This time, let's use a different function. Let's take this one, Newton's gravitation formula:

$$F = \frac{GMm}{r^2}$$

The first thing I want to do is to plot the function curve. In order to do that, I'll need to know what the G, M and m are. If we use the Earth as the big mass, and Kylie Minogue as the small mass, then using the values provided by NASA at the time of writing,

- G, the gravitational constant, is $6.673 \times 10^{-11}\ m^3 kg^{-1} s^{-2}$
- M, the mass of the Earth, is $5.9736 \times 10^{24}\ kg$
- m, the mass of Kyle Minogue, is (according to the prestigious website http://www.howmuchdotheyweigh.com at the time of writing) $46\ kg$

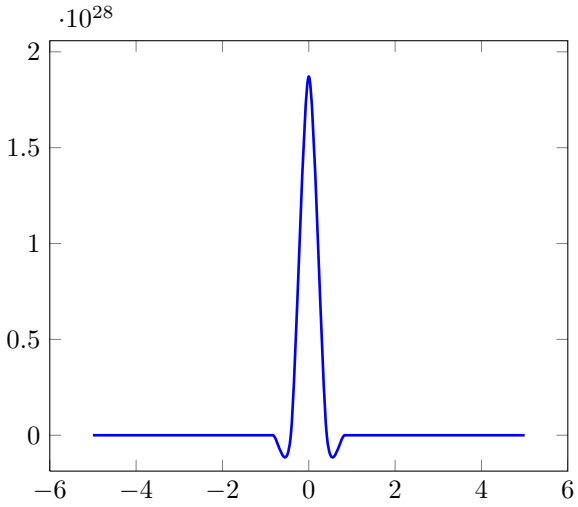
So we now have everything we want to put into this equation:

$$F = \frac{6.673 \times 10^{-11} \cdot 5.9736 \times 10^{24} \cdot 46}{r^2}$$

So as a first step, let's draw this function:

```
\begin{tikzpicture}
   % Constants
   \def \G {6.673E-11}
   \def \M {5.9736E24}
   \def \m {46}

   \begin{axis}
      \addplot [blue, line width = 1, smooth] {$\G * \M * \m / x^2$};
   \end{axis}
\end{tikzpicture}
```

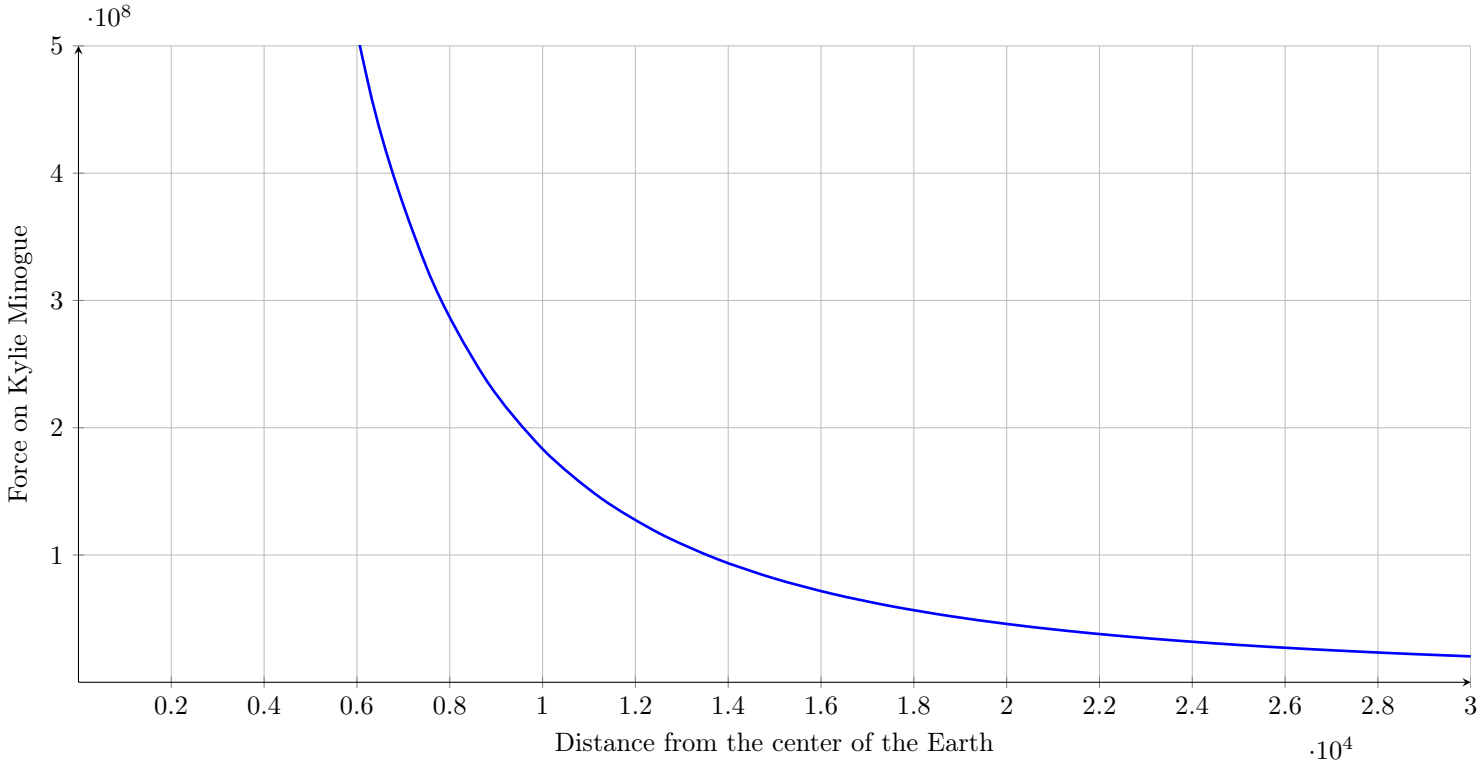
A new thing here is the use of the *\def* command to declare constants that we are going to use in the rest of the picture. That's neat. It means that we don't have "magic numbers" all over our TEX: if the mass of Kylie Minogue changed, for example, then we would only have to make the change to our TEX in one place.



Remembering that we are only interested in positive r values (the distance from the center of the Earth needs to be positive), and doing a lot of other tweeking that I'm not going to bore you with, this is what I've come up with:

```
\begin{tikzpicture}
   % Constants
   \def \G {6.673E-11}
   \def \M {5.9736E24}
   \def \m {46}

   \begin{axis}
      [
         ymin=0,
         ymax=0.5E9,
         xmin=0,
         grid=major,
         axis on top=false,
         axis x line=middle,
         axis y line=middle,
         xlabel=Distance from the center of the Earth,
         xlabel near ticks,
         ylabel=Force on Kylie Minogue,
         ylabel near ticks,
         width=20cm,
         height=10cm
      ]

      \addplot [blue, line width = 1, smooth, domain=0:30000] {$\G * \M * \m / x^2$};
   \end{axis}
\end{tikzpicture}
```

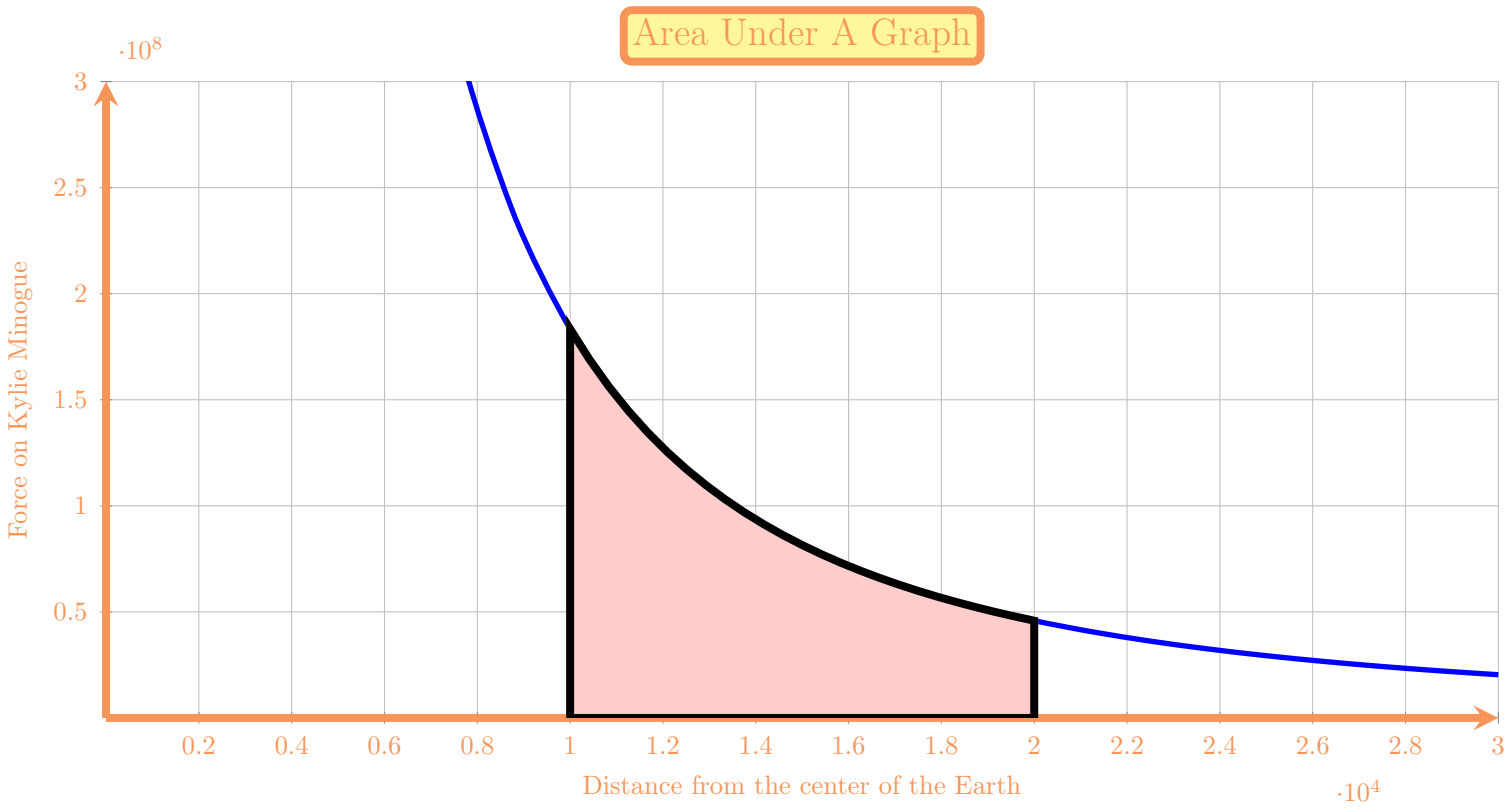
and this is now what the graph becomes:



OK, so let's try and shade an area under this graph. And give it a title.

```
\begin{tikzpicture}

    % The Styles
    \tikzstyle {titleStyle} = [draw, fill=yellow!50!white, rectangle, rounded corners=3pt, font=\Large]
    \tikzstyle {materia} = [draw, fill=blue!20, text width=10cm, text centered, minimum height=1cm, drop shadow]

    % Constants
    \def \G {6.673E-11}
    \def \M {5.9736E24}
    \def \m {46}

    \begin{axis}
        [
            ymin=0,
            ymax=0.3E9,
            xmin=0,
            grid=major,
            axis on top=false,
            axis x line=middle,
            axis y line=middle,
            xlabel=Distance from the center of the Earth,
            xlabel near ticks,
            ylabel=Force on Kylie Minogue,
            ylabel near ticks,
            width=20cm,
            height=10cm,
            title=Area Under A Graph,
            title style={titleStyle},
            Peach,
            line width=3
        ]

        % Draw the graph
        \addplot [blue, line width = 2, smooth, domain=0:30000] {\G * \M * \m / x^2};

        % Draw the shaded region under the graph
        \addplot [red!20!white, fill, domain=10000:20000] {\G * \M * \m / x^2} \closedcycle;
        \addplot [black, domain=10000:20000, line width=3] {\G * \M * \m / x^2} \closedcycle;

    \end{axis}
\end{tikzpicture}
```

and this is now what the graph becomes:



Vile! But you get the idea!

Part III
Graphing Options

6 Plotting Functions

Use the *addplot* command to draw functions:

```
\addplot [blue, line width = 1, smooth] {3*x^2 + 5*x - 2};
```

7 Plotting Points

Points can be plotted using the *node* command. For example, to draw a red point of a particular size, with a label "the tangent point" attached, at the point (1,6), do this:

```
\node [fill=red, circle, scale=0.5, pin=-70:{$the tangent point$}] at (axis cs: 1,6) {};
```

The pin option works like this. The pin command is of the form:

```
pin=angle:{$text$}
```

where angle determines where the line from the pin to the text will be displayed. Play around with it to see how it works.

And to give the point a name, "B", which is placed both above and left of the point at (1,6), do this:

```
\node [above left] at (axis cs: 1,6) {$B$};
```

You might be able to guess other *node* options.

8 Drawing Straight Lines

There are two ways of drawing straight lines. First, you could just draw the function, if you know the equation for it:

```
\addplot [red, line width = 1, domain=0.3:1.7] {11*x - 5};
```

Or, if you know two points that are on the line, you could just join them:

```
\draw (axis cs: -3,2) -- (axis cs: 5,4);
```

9 Drawing Circles and Ellipses

Use the draw command to draw circles:

```
\draw [blue, line width=2] (axis cs: 3,4) circle [radius=5];
```

The same kind of thing can be done to draw ellipses:

```
\draw [red, line width=2] (axis cs: 5,5) ellipse [x radius=1, y radius=2];
```

In order to get the circles and ellipses to draw properly, you need to have

```
\pgfplotsset{compat=newest}
```

in the document preamble.

10 Definitions

It is possible to define variables that represent numbers. That way if you want to change the value of a particular number, you only have to do it in one place. For example,

```
% Constants
\def \G {6.673E-11}
\def \M {5.9736E24}
\def \R {6.371E6}
```

could represent the gravitational constant, the mass of the Earth and the radius of the Earth respectively. They could then be used like this:

```
\addplot [red!20!white, fill, domain=10000:20000] {\G * \M * 46.0 / x^2};
```

Note that definitions and *styles* (see Section 11) need to go between the picture and axis *begin* statements:

```
\begin{tikzpicture}

    % The Styles
    \tikzstyle {titleStyle} = [draw, fill=yellow!50!white, rectangle, rounded corners=3pt, font=\Large]
    \tikzstyle {materia} = [draw, fill=blue!20, text width=10cm, text centered, minimum height=1cm, drop shadow]

    % Constants
    \def \G {6.673E-11}
    \def \M {5.9736E24}
    \def \m {46}

    \begin{axis}
...
```

11 Styles

It is possible to define styles for lines, points, pins, text, etc. Here are a couple of examples:

```
\tikzstyle {titleStyle} = [ draw, fill=yellow!50!white, rectangle, rounded corners=3pt, font=\Large ]
\tikzstyle {materia} = [ draw, fill=blue!20, text width=10cm, text centered, minimum height=1cm, drop shadow ]
```

which could then be used like this:

```
\begin{axis} [ title style={titleStyle} ]
\end{axis}
```

for example. Oh, and if you are going to use shadows, you have to add

```
\usetikzlibrary{shadows}
```

to the document preamble. And note that styles need to go between the picture and axis *begin* statements (see Section 10).

If you want to style a *pin*, you have to do this (where the name of the style is *every pin* - and you have to write that exactly: all lower case, and with a space):

```
\tikzstyle {every pin} = [ draw, fill=yellow!50!white, rectangle, rounded corners=3pt, font=\Large ]
```

12 General Options

These are options that apply to more than one type of command. For example, you may well be interested in changing the colours not only of axes, but also of plotted lines, points, circles, etc.

12.1 List of General Options

Option	Values	Comment
colours	See Section 12.2	Axes, points, lines, circles etc can all be drawn in different colours.
line width	number	This is pretty straightforward. The line (whether axis or plot) will have the given width.

12.2 Colours

Colours can be applied just about anywhere. Anything you can draw using Tikz and PGFPlots can have a colour.

12.2.1 Basic Colours

If you want to specify colours other than black for your axes, you can. You do this by adding the option

```
color=blue
```

say, to your list of options. [Sorry about the American spelling of *color*.] For example:

```
\begin{axis} [ color=blue ]
```

Because colouring things is such a common thing to do, you can even just write the colour:

```
\begin{axis} [ blue ]
```

12.2.2 dvips Colours

As well as the standard ones that come with LaTeX (white, black, red, green, blue, cyan, magenta, yellow), it is possible to get hold of a whole load of colours by typing this into your preamble:

```
\usepackage[usenames,dvipsnames]{xcolor}
```

and interestingly, this won't work unless you put this line *before* the `\usepackage{pdfplots}` line.

Now we can have all these colours:

Apricot Aquamarine Bittersweet Black Blue BlueGreen BlueViolet BrickRed

Brown BurntOrange CadetBlue CarnationPink Cerulean CornflowerBlue Cyan Dandelion

DarkOrchid Emerald ForestGreen Fuchsia Goldenrod Gray Green GreenYellow

JungleGreen Lavender LimeGreen Magenta Mahogany Maroon Melon MidnightBlue

Mulberry NavyBlue OliveGreen Orange OrangeRed Orchid Peach Periwinkle

PineGreen Plum ProcessBlue Purple RawSienna Red RedOrange RedViolet

Rhodamine RoyalBlue RoyalPurple RubineRed Salmon SeaGreen Sepia SkyBlue

SpringGreen Tan TealBlue Thistle Turquoise Violet VioletRed White

WildStrawberry Yellow YellowGreen YellowOrange

12.2.3 Creating Your Own Colours!

And if the above 68 different colours are not enough for you, it is also possible to create your own colours. You can do this by following this syntax:

```
Blue!20
```

This will create a colour that is 20% blue. Which looks like this:

Then this

```
Blue!40!Apricot
```

will create a colour that is 40% Blue and 60% Apricot. That will look like this:

And then

```
Orchid!50!Turquoise!30!YellowOrange
```

will create a colour that is (50 * 0.3)% Orchid, (30 * 0.3)% Turquoise, and 70% YellowOrange!!! And if you are wondering, this colour turns out to be

Now the possibilities are endless...

13 Picture Options

13.1 List of Picture Options

Option	Values	Comment
scale	number	The *scale* option allows you to create a larger version of the picture that you want to draw.

14 Axis Options

14.1 List of Axis Options

Option	Values	Comment
xlabel	$text$	The label associated to the x-axis.
ylabel	$text$	The label associated to the y-axis.
xlabel near ticks	N/A	Puts the x-axis label on the outside of the graph aligned with the axis.
ylabel near ticks	N/A	Puts the y-axis label on the outside of the graph aligned with the axis.
width	length	Sets the width of the picture.
height	length	Sets the height of the picture.
grid	*minor*, *major*, *both*, *none*	Sets the background grid for the picture. *minor* puts the grid lines at minor tick positions; *major* puts the gridlines at major tick positions; *both* puts gridlines at both minor and major tick positions; *none* does not include gridlines.
axis x line	*box*, *top*, *middle*, *center*, *bottom*, *none*	*box* draws the box; *top* draws the x-axis along $y = y_{max}$; *middle* and *center* draw the x-axis along $y = 0$; *bottom* draws the x-axis along $y = y_{min}$; *none* does not draw the axis.
axis y line	*box*, *left*, *middle*, *center*, *right*, *none*	*box* draws the box; *left* draws the y-axis along the left edge of the domain; *middle* and *center* draw the y-axis along $y = 0$; *right* draws the y-axis along the right edge of the domain; *none* does not draw the axis.
axis on top	*true*, *false*	The *axis on top* option enables you to draw the axes on top of other stuff drawn in the picture. Or not, of course!
axis equal	N/A	Makes sure that the x- and y- scales are the same size. This is useful when drawing circles, for example.
legend pos	*south west*, *south east*, *north west*, *north east*, outer north east	The position of legends can be specified using this option.
xmin	number	Specifies the lowest value for the domain of the graph or function plot.
xmax	number	Specifies the highest value for the domain of the graph or function plot.
ymin	number	Specifies the lowest value for the range of the graph or function plot.
ymax	number	Specifies the highest value for the range of the graph or function plot.
title	$text$	Specifies the title for the graph.
title style	{style name}	Specifies the style in which the title will be displayed.

15 Function Plot Options

15.1 List of Function Plot Options

Option	Values	Comment
smooth	N/A	Smooth plots are interpolated smoothly between plotted points.
sharp plot	N/A	This joins coordinates with straight lines.
domain	number : number	This gives the domain of the function in a range of x values. For example: *domain=-5:5*