

Tasks and Solutions

We designed the worksheets of the chapter in such a way that it beautifully illustrated how Maple participated in the process of the solution of the tasks. More precisely, how it participates in the interpretation and the demonstration of the selected task and the solution.

While we are trying to understand the behaviour and the inner properties of the mathematical objects which come up as a solution, new questions keep on arising. For example, would we think about which parabolas out of the class of parabolas received as results of the task in 2.1 are upstanding and reverse parabolas if we did not start to draw the elements of the class of parabolas and found an upstanding and a reverse parabola accidentally? We are afraid that the answer is no. Especially as this was not part of the original task. The arising questions shed new light on the whole topic in many cases. And in these cases the merits of Maple are exceptional.

2.1 The Representation of a Class of Parabolas

Let's determine the parameters a , b and c in parabola $a x^2 + b x + c$ such a way that

- a) the extremum of the parabola is on the $y=x$ line and
- b) the parabola goes through the point $[0,1]$.

Before we get down to solve the task we have to do some introductory preparations. Namely, we have to calculate the critical points of the polynomial $a x^2 + b x + c$. Of course, we could do it without Maple but it is practical to execute the following instructions, especially because it offers an opportunity for getting to know another input method of the commands, namely the usage of the context menus.



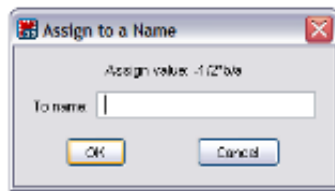
The **context menu** is a dropdown list which can be activated by the mouse right click. We navigate the pointer above the object on which we want to execute an operation and thus start the **context menu**. Its characteristic is that it offers the execution of different instructions depending on the object located under the mouse pointer.

The **context menu** can be launched from any parts of the worksheet: from text, from input line and even from the output of the instruction. This functionality mentioned previously gives us the opportunity to execute operations sequentially in a way that practically we do not have to type commands. We choose the operation to be executed from the context menu, or maybe we can also give the necessary parameters in the open dialogue window and the system inserts the command generated by itself subsequent to the Maple prompt.

The left column of the following table shows the command in question and its output. It gives you a guide to the usage of the **context menu**. The right column shows that what kind of actions are to be executed following the correctly executed steps.

Enter the parabola.

<pre>> parab := a x^2 + b x + c parab := a x^2 + b x + c</pre> <p>Take the pointer above the output line next to the letter c and then right click.</p>	<p>(1)</p>
---	------------

<pre>></pre>	
<pre>> diff((1), x)</pre> $2ax + b$ <p>Take the pointer above the output line, open context menu with right click and choose the Apply Command. Enter the solve character sequence in the Applicable Command field of the appearing dialogue window then enter x in the Applicable Arguments field.</p>	
<pre>> solve((2), x)</pre> $-\frac{1}{2} \frac{b}{a}$ <p>Open the context menu again above the output line and choose the Assign to Name command. Type the <i>szhely</i> character sequence into the To name field.</p>	
<pre>> szhely := (3)</pre> $szhely := -\frac{1}{2} \frac{b}{a}$ <pre>></pre>	

We met both **diff** and **solve** commands in the first chapter. The **solve** command solves the result of the previous command for x which is the derivative of the parabola. According to this, *szhely* contains the critical point of the parabola.

According to the second specification of the task, the value of the parabola(s) we are looking for is 1 at the point $x = 0$. We can give value to the variable appearing in an expression with the **subs** command and the system executes the simplification of the expression to the given value of the variable.

```
> e1 := subs(x=0, parab) = 1
```

$$e_1 := c = 1$$

So we substituted zero in the place of x in the *parab* expression and Maple simplified the solution to c . We made this solution equal to 1 and we gave the received equation as a value to the variable e_1 . Maybe at first sight the two equal signs can seem strange in the result of the command. But notice that the first

equal sign is really $:=$ which is the sign of the assignment. So please understand that the value of e_1 is an equality the left side of which is c and right side is 1.

The other specification of the task is that the extremum of the parabola should be on the $y = x$ line. Since we know that the critical point of the parabola is at the *szhely*, this is what we have to substitute into the variable *parab* in order to get the extremum of the parabola. The extremum will be on the line $y = x$ if it is equal to the value of *szhely*.

$$\begin{array}{l} \text{> } e_2 := \text{subs}(x = \text{szhely}, \text{parab}) = \text{szhely} \\ \qquad \qquad \qquad e_2 := -\frac{1}{4} \frac{b^2}{a} + c = -\frac{1}{2} \frac{b}{a} \end{array} \quad (6)$$

Notice that e_1 and e_2 provide two equations to the variables a , b and c . We can solve the system of equations to a and c considering b as a parameter or a free variable, if you like. Although so far we have used the solve procedure to solve equations for one unknown, it can also be used to solve the equation systems. In this case a set of equalities must be given as the first parameter and the set of those variables the solutions of which we are looking for must be given as the second parameter.

$$\begin{array}{l} \text{> } \text{solve}(\{e_1, e_2\}, \{a, c\}) \\ \qquad \qquad \qquad \left\{ a = \frac{1}{4} b^2 - \frac{1}{2} b, c = 1 \right\} \end{array} \quad (7)$$

We have received the solution of the system of equations in the form of a set in which two equalities can be found: the solution to variables a and c expressed with parameter b . What do you think will be the value of the variables a and c ? In other words, does the solve procedure give value to the variables to which it has solved the system of equations? The following command gives the answer:

$$\begin{array}{l} \text{> } a, c \\ \qquad \qquad \qquad a, c \end{array} \quad (8)$$

Both variables have returned their own names. These kinds of variables are called **unbounded** variables. The name derives from the fact that we bounded the own value of the variable that has a value to the variable itself, thus this kind of variable is called a **bounded** variable. So if the variable does not have a value then it is obviously **unbounded**.

The **solve** procedure has only generated the solution set and it has not given value to the variable. The **assign** procedure, to which we can give the output of the solve procedure as the parameter, ensures this.

$$\text{> } \text{assign}((7))$$

The command itself does not return an output. Its response is empty, which means its output is the NULL constant, that is, an empty sequence. However, if we ask for the value of the variables a and c , then we get the wanted solution.

$$\begin{array}{l} \text{> } a, c \\ \qquad \qquad \qquad \frac{1}{4} b^2 - \frac{1}{2} b, 1 \end{array} \quad (9)$$

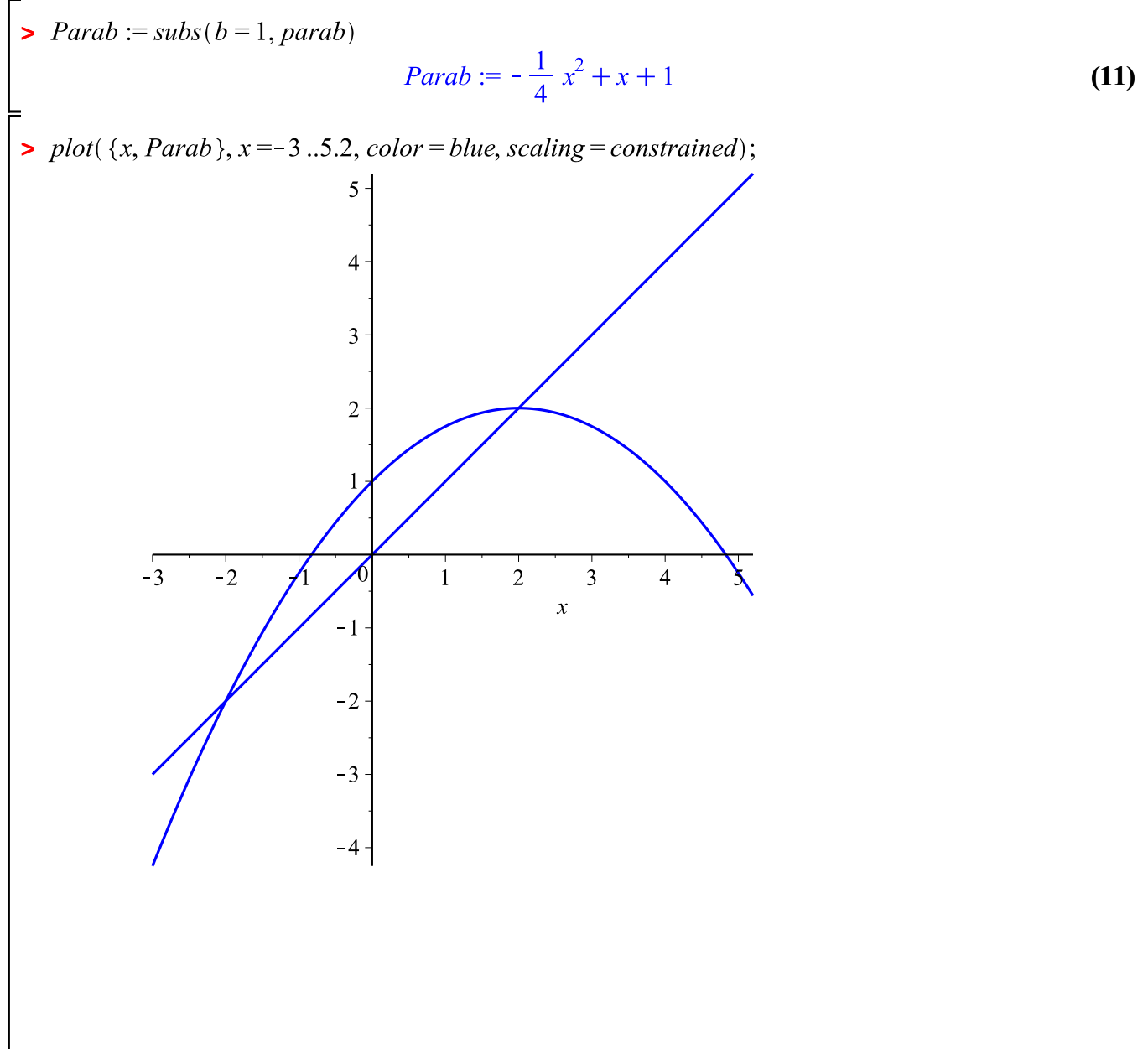
So the parabolas which we are looking for are described by the one-parameter family of curves.

$$\begin{array}{l} \text{> } \text{parab} \\ \qquad \qquad \qquad \left(\frac{1}{4} b^2 - \frac{1}{2} b \right) x^2 + b x + 1 \end{array} \quad (10)$$

Just a side question: Why do we find it natural that Maple evaluates the variable *parab* to

$\left(\frac{1}{4}b^2 - \frac{1}{2}b\right)x^2 + bx + 1$ when its value was $ax^2 + bx + c$ at the time of its creation? The answer is that in the meantime variables a and c got a value which Maple obviously substituted into the original formula. This is the **complete evaluation** process which is typical of the computer-algebra systems and of which we are going to talk about later in details.

We can consider the task is solved. But the advantages of Maple can be seen clearly at this point. It would be good to see how the solutions of the task look like. The plot command is very expressive.

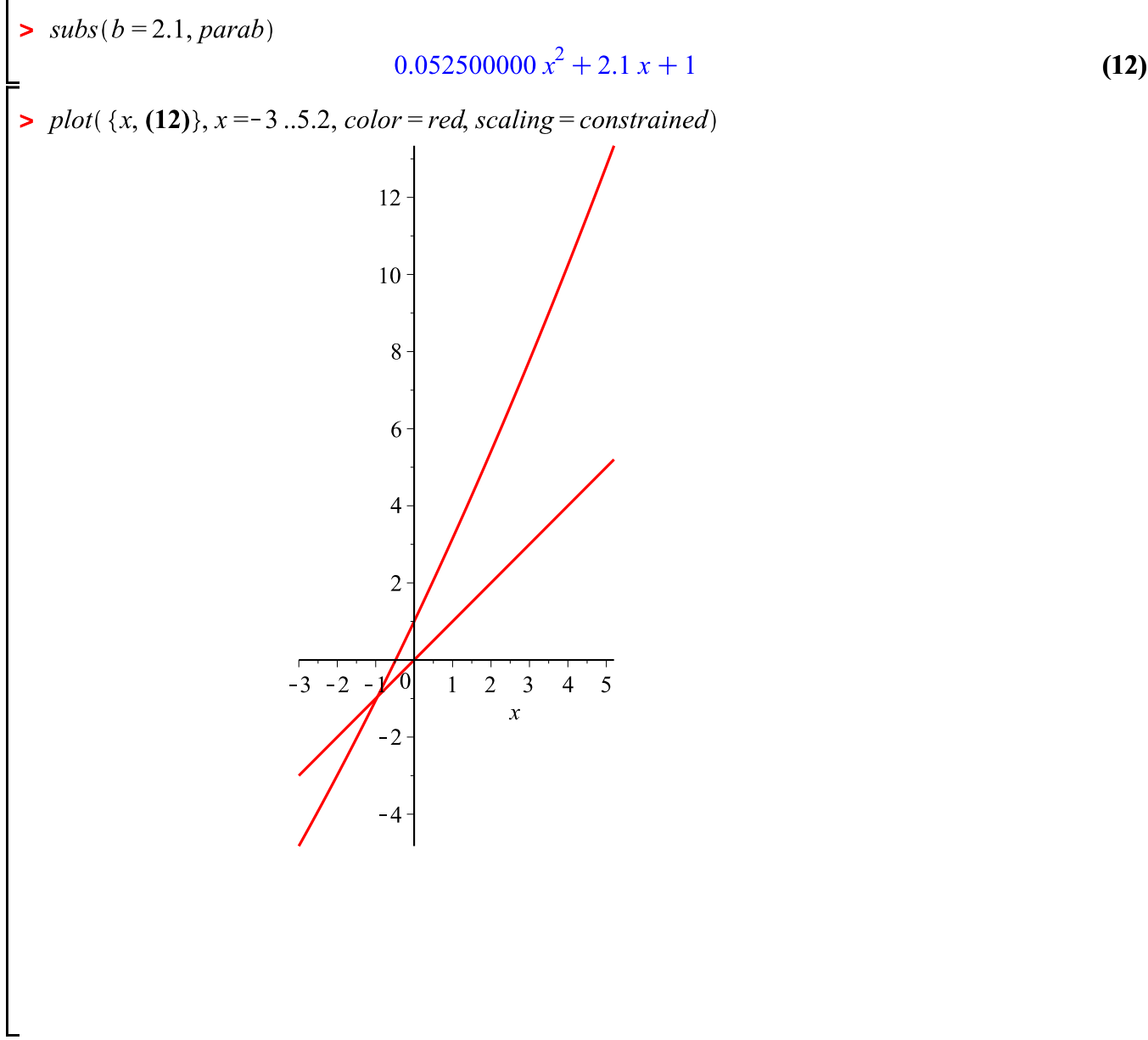


The graph speaks for itself. It beautifully illustrates the parabola, the $y = x$ line and the fact that the parabola crosses the y -axis in the point 1 and its extremum lies really on the line. Notice that first we created the parabola in *Parab* which is the element belonging to the $b = 1$ parameter values of the group of parabolas. It is very important to notice that the variables *parab* and *Parab* are different variables. Maple is sensitive to lower case and capital letters.

The *color = blue* option of the plot procedure ensures the representation in blue and the option *scaling = constrained* ensures that the axes are scaled according to equal measure. We suggest that our

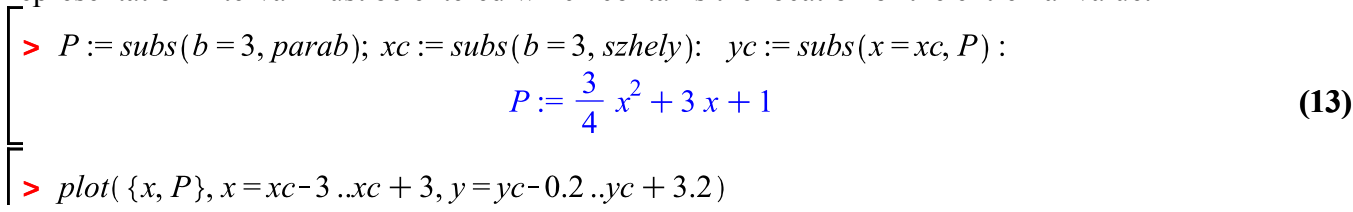
readers should draw other solutions with the other values of the parameter b and with the changing of the domain.

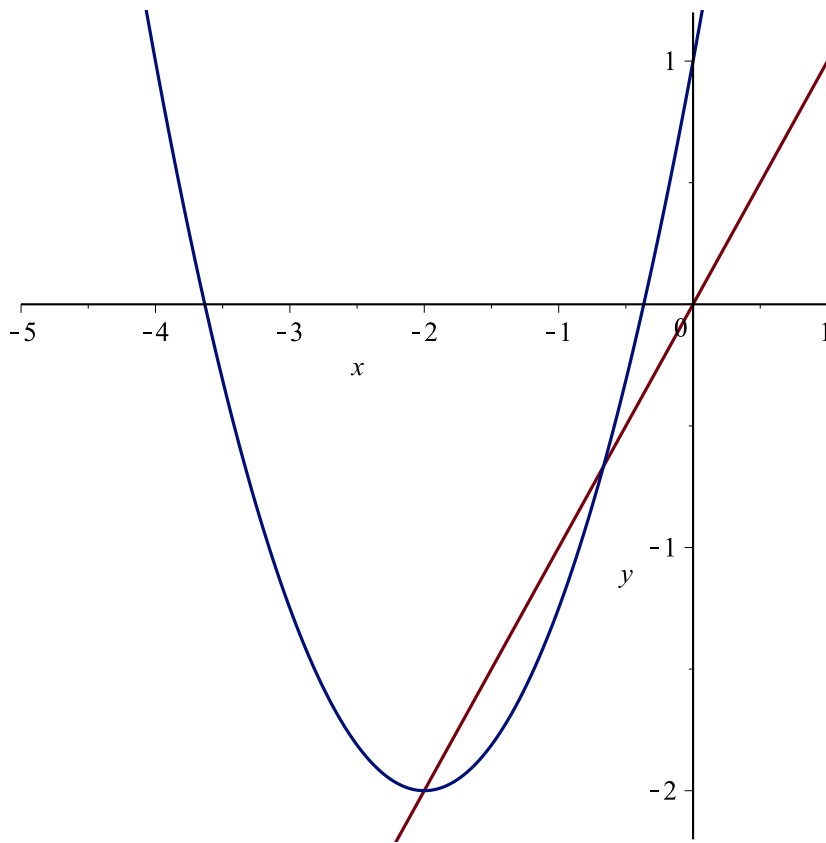
If you took our advice and you really have experimented then we are pretty sure that a graph similar to the one below has been created.



It does not really illustrate the behaviour of the parabola. It is also difficult to decide that out of the two functions which one is the line and which one is the parabola.

A question can emerge at this point: how can it be ensured besides experimenting that the graph shows the essence to the arbitrary value of the parameter b ? That is, how can it show that the parabola is a curve, the line is a straight and that the extremum of the parabola is on the line? By all means such a representation interval must be entered which contains the location of the extremal value.





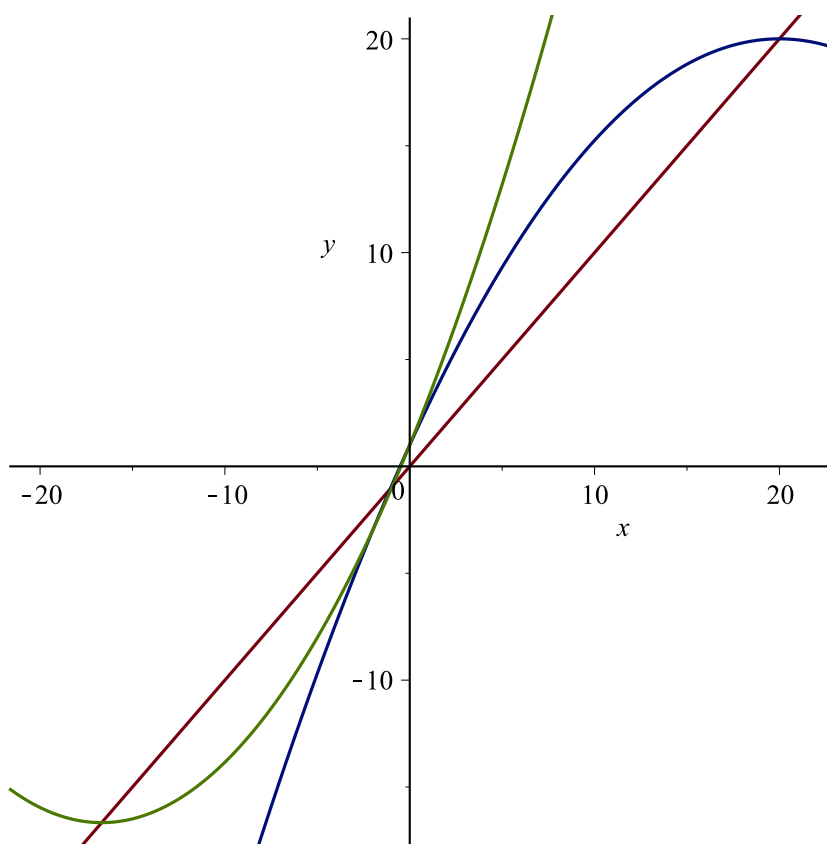
First, we calculated the expressions that determines the parabola in the variables P , x_c and y_c at the parameter value $b = 3$. Then its extreme value location and the extreme value itself were also calculated. The latter two coincide developing our trust in the correctness of the solutions. The domain on the x -axis was chosen in such a way that it would clearly show the curve of the parabola. The y -axis was restricted so that the significantly different values of the line would not distort the curve.

As long as we motivate the readers to make further attempts until let's continue our investigation of the class of parabolas. Let's draw more parabolas in one graph.

$$\begin{aligned} > f := \text{subs}(b = 1.9, \text{parab}); \quad x_f := \text{subs}(b = 1.9, \text{szhely}); \quad y_f := \text{subs}(x = x_f, f); \\ & \quad \quad \quad f := -0.0475000000 x^2 + 1.9 x + 1 \end{aligned} \quad (14)$$

$$\begin{aligned} > g := \text{subs}(b = 2.12, \text{parab}); \quad x_g := \text{subs}(b = 2.12, \text{szhely}) \quad : y_g := \text{subs}(x = x_g, g); \\ & \quad \quad \quad g := 0.0636000000 x^2 + 2.12 x + 1 \end{aligned} \quad (15)$$

$$> \text{plot}(\{x, f, g\}, x = \min(x_f, x_g) - 5 .. \max(x_f, x_g) + 3, y = \min(y_f, y_g) - 1 .. \max(y_f, y_g) + 1);$$



Notice that the color option was not set in the **plot** command and in this case Maple represents the functions with different colours.

By the way, parabolas with a minimum and a maximum have been found for two positive and not overly different parameter values. This can raise the question that to which values of the parameter b we receive an upstanding parabola? The answer is given by the behaviour of the coefficient of the squared term.

Let's take again the p which describes the one-parameter family of parabolas.

`> parab`

$$\left(\frac{1}{4} b^2 - \frac{1}{2} b \right) x^2 + b x + 1 \quad (16)$$

`> lcoeff(parab, x)`

$$\frac{1}{4} b^2 - \frac{1}{2} b \quad (17)$$

The command $lcoeff(y, x)$ considers the expression y as the polynomial of x and it gives the coefficient of the highest degree term. Its behaviour, that is, its sign determines the position of the parabola. The

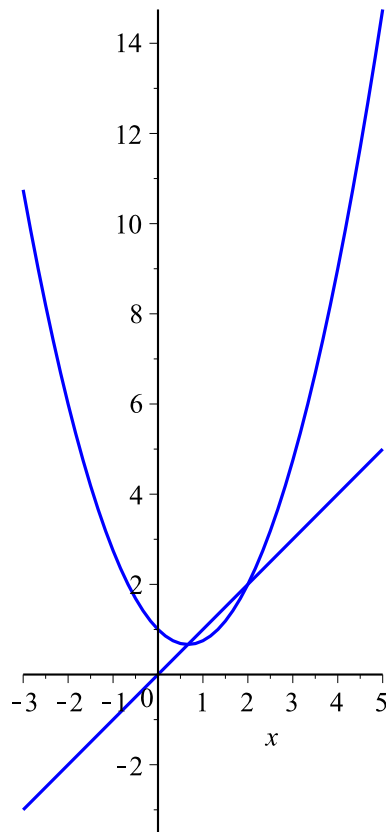
parabola $\frac{1}{4} b^2 - \frac{1}{2} b$ is upstanding with the 0 and 2 roots, that is, it has negative values in the (0,2)

open interval. According to this, the polynomial p is going to be an upstanding parabola to those values

of the parameter b which are lower than zero or bigger than two. We would like to draw the readers' attention to the $b = 0$ and $b = 2$ values. What kinds of curves do we get for these parameter values? Are they the solutions of our original task?

Perhaps our readers may be fed up with the interpretation of the result of our task but it is not over. There are more exciting things to come. Now we know that to which parameter values we get upstanding or reverse parabolas and to which we get an abnormal solution. We can also visualize these parabolas. However, we have not seen yet the change of the family of parabolas depending on the change of the parameter b . That is, we want to generate movement on the screen. Let's see the next command.

```
> plots[animate]({x, parab}, x=-3..5, b=-1..0.8, color=blue, scaling=constrained, frames=20);
```



First of all, the name of the procedure seems strange. So far we have met such names like **diff**, **solve** or **assign**. In the case of the command above, the **plots[animate]** itself is the name of the procedure.

A part of the procedures of Maple are available after the system was booted and so far we have met this phenomenon. Other procedures are located in the so-called packages and in the case of these procedures not only the name of the function but also the name of the package must be given to the system. In the case of the example above, **plots** is the name of the package, **animate** is the name of the procedure and **plots[animate]** is the long name of the procedure. But the usage of the long name can become inconvenient that's why Maple offers the **with** procedure which offers all the procedures of the package

given as a parameter. Thus we can refer to them with their short names. So, for example, after entering the **with(plots)** command, we can use the short name of the **animate** procedure.

The first parameter of the **animate** procedure is the same as that of the **plot** procedure. The function to be drawn or the set of these and its second parameter is the domain. But notice that in this case the function p has two variables in reality because the parameter b appears in it. This is what the **animate** procedure takes advantage of: it puts the function p sequentially on the screen while the parameter b runs through the $1 \dots 1.6$ interval with 20 division points.

The window opened by the **animate** procedure is completed with additional controllers. These provide the step-by-step and automatic play of the animated pictures, the direction and the speed of the play and the set of the single or cyclical play.

For the sake of the readers who are interested in showing another animate command we give it.

```
[> plots[animate]({x, parab}, x=-3..7, b = 1..1.6, color = blue, frames = 20) :  
[>
```

The `example(animate)` gives further perfect examples of the animations that can be executed on the screen. Attention! Don't forget to enter the `with(plots)` command when you try out the examples offered by `example(animate)`.

Ladies and gentlemen, please experiment! Have a good time!

What Have You Learnt About Maple?

- The **context menus** is design to facilitate the users of typing the commands. The context indicator refers to the fact that the offer of the menu depends on the object from which it was activated. We put the mouse pointer above the selected object and then right click. The dropdown menu offers different commands which can be executed on the chosen object. We can choose from these with mouse click and if needed, we can give further data concerning the command to be executed. As a result, Maple itself generates the wanted command and it pastes it after the next prompt.
- Maple is sensitive to lower case and capital letters. Thus *poly*, *Poly*, *PoLy*, *poLY* and *POLY* all represent different variables.
- The `solve({ e_1, e_2 }, { x, y })` command solves the system of equations containing in variables e_1, e_2 for the variables x and y . The result of the command is a sequence of sets. The syntax of all the sets are $\{x = k_1, y = k_2\}$ where k_1 and k_2 are expressions and each of them represents one solution of the system of equations. Of course there is no limitation to the number of the equations and the unknowns.
- The **assign** procedure can be used to give value to the variables. We can give the result of the **solve** command to it. More precisely we can give one element of the sequence returned by the **solve** procedure and in this case the variables add the values generated by **solve**.
- In the **plot** procedure we can set the colour of the curves of the functions with the **color** option. If this option is not used then Maple represents each curve of functions with a different colour. We get the colours available with the command `help("color")`.
- The scaling option of the **plot** procedure determines the scale of the scaling on the x - and y -axes. In the case of `scaling = constrained` this scale is 1:1, that is, the two axes are scaled in equal proportions. If this option is not used then Maple takes advantage of the biggest space of the screen when it scales the axes.
- The `lcoeff(y, x)` command considers the y expression as the polynomial of x , thus it returns with the

leading coefficient of y that is the term of x which has the biggest exponent.

- The `animate(f(x, t), x = a .. b, t = c .. d, frames = n)` command represents the function $f(x, t)$ as the function of x in the real interval $x = a .. b$ in such a way that divides the $c .. d$ interval into n parts and t gets the $t_0 = c, t_1 = c + \frac{d-c}{n}, t_2 = c + \frac{2 \cdot (d-c)}{n}, \dots$ values. The graphs of the functions $f(x, t_0), f(x, t_1), f(x, t_2), \dots$ created are shown in the same coordinate system sequentially and it simulates movement on the screen.
- The non-library Maple procedures are located in packages. One of them is the package named **plots**. The **animate** procedure can be found here. Its full name is created by writing the name of the procedure between square brackets subsequent to the name of the package: **plots[animate]**. The **with** procedure offers an alternative solution with which all the procedures of the package given as its parameter can be made available. Thus, for example, we can refer to all the procedures of the **plots** package with their short names following the execution of the **with(plots)** command.

Exercises

1. Find such a one-parameter family of parabolas all the elements of which goes through the point $(0,1)$ and touches the $y = x$ line. Draw such curves as well.
2. Visualize the family of parabolas that you have received as a result of the 1. exercise with the methods mentioned in this worksheet.
3. Solve the following system of equations.

$2x_1 - x_2 - x_3 = 4$	$x_1 + 2x_2 + 4x_3 = 31$	$ax + by + cz + dt = 0$
$3x_1 + 4x_2 - 2x_3 = 11$	$5x_1 + x_2 + 2x_3 = 29$	$bx - ay + dz - ct = 0$
$3x_1 - 2x_2 + 4x_3 = 11$	$3x_1 - x_2 + 4x_3 = 10$	$cx - dy - az + bt = 0$
		$dx + cy - bz - at = 0$