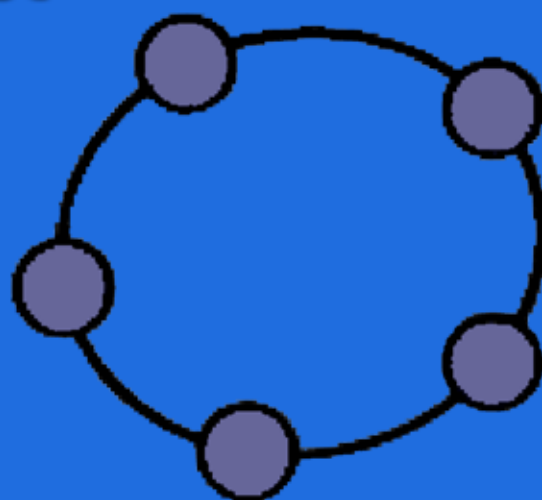


# An Introduction to GeoGebra



Produced By:

Regional Centre for Excellence in Mathematics Teaching and Learning

University of Limerick

[www.ul.ie/cemtl/](http://www.ul.ie/cemtl/)

Project Coordinators:

John O'Donoghue,  
Olivia Gill

Writing Team:

Tim Brophy  
Olivia Gill

Peer Reviewer(s):

Paddy Johnson

# Contents

<b>Foreword</b>	<b>ii</b>
Downloading and Installing . . . . .	1
Acquiring GeoGebra . . . . .	1
Installing . . . . .	1
Necessary Files . . . . .	1
Running GeoGebra . . . . .	1
Launching . . . . .	1
The GeoGebra Toolbar . . . . .	3
Algebra Input . . . . .	6
Commands . . . . .	6
Direct Input . . . . .	7

## Foreword

The Regional Centre for Excellence in Mathematics Teaching and Learning (CEMTL) is one of six projects that form the Regional Approach to Outstanding Teaching, Learning and Learner Support for the Shannon Consortium. CEMTL aims to develop and implement a comprehensive regional mathematics and numeracy enhancement strategy. To that end the following material has been created not only to assist the mathematics education of all mathematics learners in second and third level institutions but also to enhance the instruction of mathematics by teachers and lecturers in Ireland.

Please be advised that the material contained in this document is for information purposes only. The information has been peer-reviewed and is correct and accurate at the time of publishing. CEMTL will endeavour to update the information contained in this document on a regular basis.

Finally, CEMTL holds copyright on the information contained in this document, unless otherwise stated. Copyright on any third-party materials found in this document or on the CEMTL website must also be respected. If you wish to obtain permission to reproduce CEMTL materials please contact us via the CEMTL website.

# An Introduction to GeoGebra

## Downloading and Installing

### Acquiring GeoGebra

GeoGebra is an application for exploring and demonstrating Geometry and Algebra. It is an open source application and is freely available for non-commercial use. There are currently versions available for Windows, Mac OS X, Linux and other java-enabled platforms. To start GeoGebra go to <http://www.geogebra.org> where you will see links to *Webstart* or *Download*. The *Webstart* option downloads the necessary java files to your computer and starts the application immediately. The advantage of choosing this option is that the application is always up to date. The *Download* option downloads files to your computer which you must then install. The big advantage here is that you can continue to work off-line.

### Installing

The installation process is very straightforward. After you have downloaded on a Windows machine or on a Mac just double-click the downloaded file. An Install Wizard will guide you through every step. It is strongly advised that you select the *typical* configuration when given the choice. Full instructions are given on the GeoGebra site.

### Necessary Files

GeoGebra is a java application. You must have at least version 1.4.2 of Java. If you are using Windows or Linux you can get this from Sun's Java website. If you are using a Mac you can get it from Apple's website.

## Running GeoGebra

### Launching

Double click the GeoGebra icon on your desktop to start the application. You will be presented with a launch screen as shown in Figure 1. The exact details may depend on the type of Operating System that you are using but the buttons along the top, the clear space at the left, the drawing pad at the right, the input field, symbols ( $^{\circ}$ ), Greek letters ( $\alpha$ ) and the Command menu will be common to all systems.

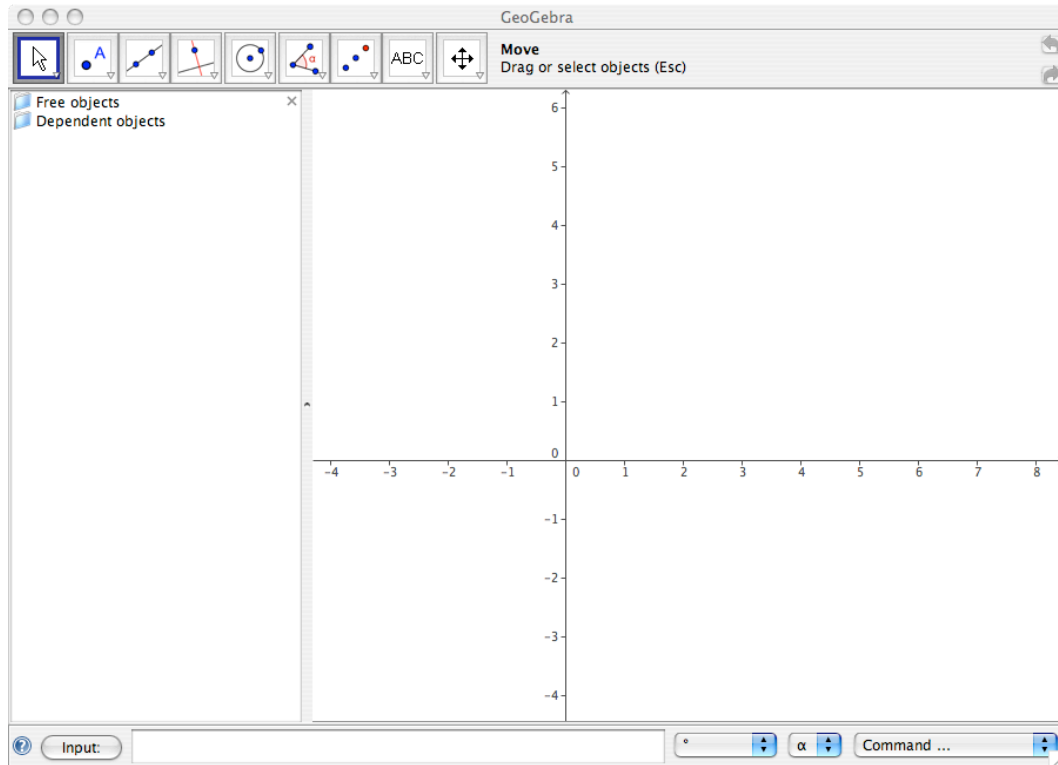


Figure 1: Launch Screen for GeoGebra

The button menu along the top (see Figure 2) contains a submenu of actions. By clicking on the down-pointing arrow at the bottom right corner of any of these buttons the submenu is displayed. The following section explains each of these submenus in more detail.

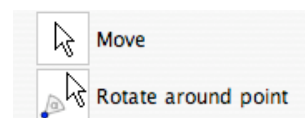


Figure 2: Buttons for GeoGebra

## The GeoGebra Toolbar

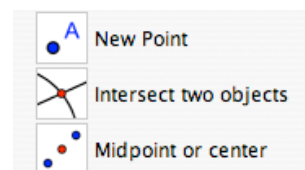
### Move

The first button here allows you to select a previously drawn object and move it around the drawing area. The second button in this menu allows you to select a point as a centre of rotation and rotate any object around it.



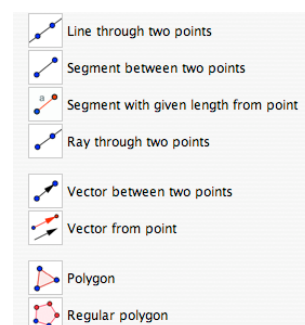
### New Point

The three buttons in the next menu allow you to construct different types of points. The first button allows you to put a point anywhere in the drawing area. The second button allows you to select two different curves and find their point of intersection. The third button constructs the midpoint of a line segment.



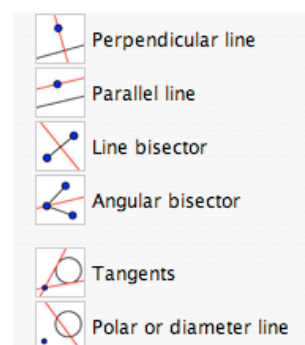
### Line through two points

The next tool has eight different buttons. These construct different types of lines and polygons. The first button draws an infinite line through two selected points. The second button draws a line segment between two selected points. The third button draws a line segment of a given length from a specific point. The fourth button draws a ray through two points. The fifth button draws a vector between two points. The sixth button allows you to draw a vector parallel to a given vector. The seventh button, the polygon tool, allows you to construct a closed polygon. The last button, the regular polygon tool, allows you to draw a regular polygon.



### Perpendicular line

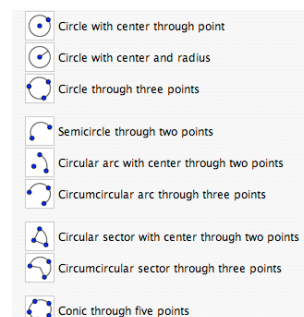
There are six buttons under the Perpendicular line tool. These allow you to draw specific types of lines. The first button allows you to draw a line perpendicular to a given line. The second button draws a line parallel to a previously constructed line. The third button, the **Line bisector**, is used to bisect a line segment. The fourth button, **Angular bisector**, bisects a previously constructed angle. The fifth button is **Tangents**. This button can draw tangents to a circle, a conic or the graph of a previously defined function. The sixth button is the **Polar or diameter line** button.



## Circle with center through point

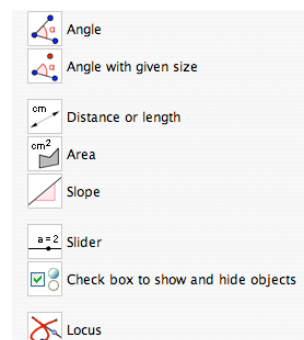
The nine buttons in the next menu allow you to construct circles, arcs, sectors and conics. The first button, **Circle with center through point**, works by either clicking on the point you want to be the centre of the circle or clicking on some blank space to create such a point. Release the mouse button. Move the mouse and you will see a circle in the process of construction. When you click the mouse again the circle is finished. The second button in this menu is **Circle with center and radius**. Click on the required centre. A menu appears asking you for the radius. Enter the required value and press the Return key. The desired circle is drawn. The third button is called **Circle through three points**. Select this tool and click on three different points. The circle through these points is constructed.

The fourth button, **Semicircle through two points**, does exactly what it says. Click with it in two different places to construct the semicircle with that line segment as diameter. The fifth button here is **Circular arc with center through two points**. Click somewhere with this button to construct the centre of the circle. Click again to create the first point of the arc. If you move the mouse anti-clockwise you will see an arc being constructed. When you click again the arc is completed. The sixth button, **Circumcircular arc through three points**, draws an arc through any three points. To use this button click in three different places. Three points and the arc passing through them are constructed. The seventh and eighth buttons do for sectors what the fifth and sixth buttons do for arcs. The final button, **Conic through five points**, works by allowing you to select 5 different points on the screen. It then calculates and plots the conic section passing through all 5 points.



## Angle

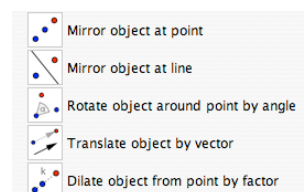
The eight buttons in this menu deal mainly with measurement. The first button, **Angle**, is used to construct and measure the angle between three points. The second button is **Angle with given size**. Select this tool and click to create the vertex of the arm of an angle. The next click gives the point at the angle. At this point a menu appears where you will be asked to select angle size and direction. The third button, **Distance or length**, has several functions. If you use this button and click on two previously constructed points it will give you the distance between them. The function of this button is to measure the length of a line segment. The fourth button is called **Area**. Use this button on a polygon, circle or conic to display the area of the figure. The fifth button, **Slope**, will display the slope of a selected line or line segment.



The sixth button, **Slider**, allows us to construct variables which can be changed in Dynamic worksheets. The seventh button, **Check box to show and hide objects**, is another button to control what is displayed on the screen. The final button in this section is called **Locus**. This tool allows you to select a point, B, that depends in some way on a previously created point, A. Using the locus tool on B first and then A will construct the locus of B subject to the given conditions.

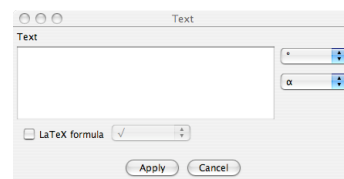
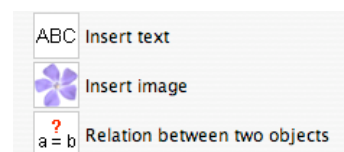
### Mirror object at point

These five buttons are concerned with geometric transformations. The first button, **Mirror object at point**, performs a central symmetry. Select the object that you want to perform the symmetry on, choose the centre of symmetry, the mirror image is then constructed. The button, **Mirror object at line**, works the same way for an axial symmetry. The next button, **Rotate object around point by angle**, needs you to select the object to be rotated and the centre of rotation. A menu will then appear asking for the size of the angle. You can enter a number or the variable name of a previously constructed angle. The default rotation is anti-clockwise. **Translate object by vector** performs the translation of an object while **Dilate object from point by factor** performs a dilation.



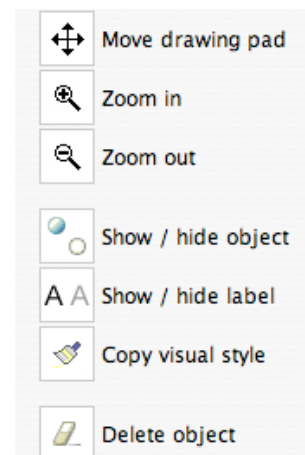
### Insert text

There are three buttons in this menu. The first, **Insert text** allows you to place text anywhere on the page. When you select this tool and click on the drawing pad a menu like the one on the bottom right appears. The button with the degree symbol helps you to enter mathematical symbols while the button with  $\alpha$  on it contains the Greek alphabet. These two allow you to enter most things that you will need. The **Insert image** button allows you to import various types of images into your worksheet. Once the image is imported it is treated as an ordinary object and you can use various tools on it. In particular, all the geometric transformation tools will work. This allows you to prepare an image in your favourite drawing application and then use it in GeoGebra. The **Relation between two objects** button tells you the relation between a limited set of objects. It will let you find out if two objects are equal, if a point lies on a line or a conic, or if a line is tangent or secant to a conic.



## Move drawing pad

Most of the buttons under this tool are self explanatory. The **Move drawing pad** button allows you to drag the whole drawing pad wherever you like. The **Zoom in** button zooms in on portions of your drawing. The **Zoom out** button zooms out on portions of your drawing. The **Show / hide object** is a little more complicated. Use the button to click on the object that you want to show or hide. Nothing appears to happen. Now select any other tool. At this point the change you wanted occurs. If you click on an object with the **Show / hide label** button the label will disappear or appear. The **Copy visual style** button is a very convenient one. Suppose that you have drawn a diagram where you assigned a particular colour to an object. Later you may want to use the same colour on another object, maybe a descriptive piece of text. If you use this style to click on the first object it will apply that object's style to the object you click on next. **Delete object** deletes any object you click on when using it. Be very careful before you use this tool. It will delete not only the object you select but also any objects that depended on the selected object during their construction. Unless you are sure that you want to delete an object it is usually a much safer option to hide it.



## Algebra Input

### Commands

GeoGebra contains a set of internal commands and functions. These can be entered directly by using the box beside the word **Input**. This is called the **Command** window. The internal commands that GeoGebra currently accepts are given in Table 1.

There are other commands available also that you can examine at your leisure in the GeoGebra help file. With these commands you can draw many types of graph that are not possible by just using the Geometry toolbar buttons. For example if you enter  $\sin(x)$  into the **Command** window and press the **Enter** key you will find that the graph of  $\sin x$  is drawn on the screen. The command

$$\text{Function}[\sin(x),0,2 \pi]$$

will draw the graph of  $\sin x$  from 0 to  $2\pi$ . Note the use of the square brackets and the space between the number 2 and the constant pi. We are now going to look at the ways that we can use direct input, entered into the **Command** window, to perform many different constructions and operations.

Operation	Command
Addition/Subtraction	+/-
Multiplication or Scalar Product	* or space
Division	/
Exponentiation	^
Factorial	!
Parentheses	()
x/y-coordinate of the point $A$	x(A)/y(A)
Absolute value	abs( )
Sign	sign( )
Square root	sqrt( )
Cube root	cbt( )
Random number between 0 and 1	random( )
Exponential function	exp( ) or $e^x$
Logarithm (natural)	ln( ) or log( )
$\log_2$	ld( )
$\log_{10}$	lg( )
Cosine/Sine/Tangent functions	cos( )/sin( )/tan( )
$\cos^{-1}$	acos( )
$\sin^{-1}$	asin( )
$\tan^{-1}$	atan( )
cosh	cosh( )
sinh	sinh( )
tanh	tanh( )
$\cosh^{-1}$	acosh( )
$\tanh^{-1}$	atanh( )
Greatest integer less than or equal to $x$	floor(x)
Least integer greater than or equal to $x$	ceil(x)
round	round( )

Table 1: Internal GeoGebra Commands

## Direct Input

### Numbers and Angles

You can enter numbers and angles directly. Type into the **Command** window **radius = 4.5**. From here on whenever I want you to type something into the **Command** window I will just say “Type”. Nothing appears in the drawing area but in the **Algebra** window you will see the equation appear. The word, **radius**, is a container that holds the number 4.5. You can use the expression **radius** as a variable in any formula. It can be changed quite easily. You can double click the

formula and edit it. A nice feature is that you can just highlight the formula and then use the up-down or left-right arrow keys to increment or decrement the number by 0.1. This is the default value, it can be changed. This can be used for animation later on. You can get  $\pi$  or  $e$  from the drop-down menu under the  $^\circ$  symbol. Angles can be entered in the same way as degrees or radians. As you already know, you can use Tools to enter numbers and angles as sliders.

### Special Commands associated with Numbers

#### *Length*

**Length**[ **v** ] Length of a vector **v**

**Length**[ **A** ] Length of the position vector of the point **A**

**Length**[ **f**,  $x_1$ ,  $x_2$  ] Length of the function **f** between  $f(x_1)$  and  $f(x_2)$

**Length**[ **f**, **A**, **B** ] Length of the function **f** between the points **A** and **B**

**Length**[ **c**,  $t_1$ ,  $t_2$  ] Length of the parametrised curve, **c**, between the two values of the parameter  $t_1$  and  $t_2$

**Length**[ **c**, **A**, **B** ] Length of the parametrised curve, **c**, between the two points **A** and **B** on the curve

**Length**[ **L** ] Length of a list **L** *ie* the number of elements in **L**

#### *Area*

**Area**[ **A**, **B**, **C**, ... ] Area of the polygon defined by the points **A**, **B**, **C**, ...

**Area**[ **c** ] Area of the circle or ellipse, **c**.

#### *Distance*

**Distance**[ **A**, **B** ] Distance between two points **A** and **B**.

**Distance**[ **A**, **g** ] Perpendicular distance of the point **A** from the line **g**.

**Distance**[ **g**, **h** ] Distance between two lines **g** and **h**. This will return 0 if the lines intersect. It is therefore a method of checking whether two lines are parallel.

*Modulo and Integer Division*

**Mod**[ **a**, **b** ] Returns the remainder when the number a is divided by the number b.

**Div**[ **a**, **b** ] Returns the integer part of  $\frac{a}{b}$ .

*Slope*

**Slope**[ **g** ] Returns the slope of a line g. This command also draws a small right-angled triangle on the line g. The size of this triangle can be changed with the Properties dialog. It can also be hidden.

*Radius, Circumference and Perimeter*

**Radius**[ **c** ] Returns the radius of a circle c.

**Circumference**[ **c** ] Returns the circumference of the circle or ellipse c.

**Perimeter**[ **P** ] Returns the perimeter of the polygon P.

*Conic sections*

**FirstAxisLength**[ **c** ] The length of the principal axis of a conic section c.

**SecondAxisLength**[ **c** ] The length of the second axis of a conic section c.

**Excentricity**[ **c** ] The eccentricity of a conic section c. Note the spelling.

*Integrals and Sums*

**Integral**[ **f**, **a**, **b** ] Returns  $\int_a^b f(x)dx$ . This command also draws the area between the function and the  $x$ -axis. You can select it and from the properties dialog change the filling to 0.

**Integral**[ **f**, **g**, **a**, **b** ] Returns  $\int_a^b (f(x) - g(x)) dx$ . The command also shades in the area between the two curves.

**Integral**[ **f** ] Returns and draws the curve  $y = \int f(x)dx$ , if possible.

**LowerSum**[ **f**, **a**, **b**, **n** ] Returns and displays the Lower sum of the function  $f$  over the interval  $[a, b]$  using  $n$  rectangles.

**UpperSum**[ **f**, **a**, **b**, **n** ] Returns and displays the Upper sum of the function  $f$  over the interval  $[a, b]$  using  $n$  rectangles.

*Iteration*

**Iteration**[ **f**,  $x_0$ , **n** ] Iterates the function  $f$  on the number  $x_0$   $n$  times. If  $f(x) = x^2$  then  $\text{Iteration}[f, 2, 1]$  will return  $2^2 = 4$  while  $\text{Iteration}[f, 2, 2]$  will return  $2^{2^2} = 16$  and so on.

*Minimum and Maximum*

**Min**[ **a**, **b** ] Returns the lower of the two numbers  $a$  and  $b$ .

**Min**[ { **a**, **b**, ... } ] Returns the lowest member of the list inside the curly brackets.

**Max**[ **a**, **b** ] Returns the greater of the two numbers  $a$  and  $b$ .

**Max**[ { **a**, **b**, ... } ] Returns the greatest element of the list between the curly brackets.

### Special Commands associated with Angles

**Remember that all angles are drawn anti-clockwise**

**Angle**[  $v_1, v_2$  ] Draws the angle between the two vectors  $v_1$  and  $v_2$ .

**Angle**[ **g**, **h** ] Draws the angle between the two lines  $g$  and  $h$ .

**Angle**[ **A**, **B**, **C** ] Draws the angle between the lines  $BA$  and  $BC$  where  $A$ ,  $B$  and  $C$  are points.  $B$  is the vertex of the angle.

**Angle**[ **A**, **B**,  $\alpha$  ] Draws an angle of size  $\alpha$  at the point  $B$  with  $AB$  as one arm of the angle. This command also draws the image of the point  $A$  by a rotation of  $\alpha$  about the point  $B$ .

**Angle**[  $v$  ] Draws the angle between the  $x$ -axis and the vector  $v$ .

**Angle[ A ]** Draws the angle between the  $x$ -axis and the position vector of the point A.

**Angle[ n ]** Creates a variable which is an angle that corresponds to the number  $n$ . Note that  $n$  will be assumed to be radian measure and the created variable will be converted to an angle between  $0^\circ$  and  $360^\circ$ .

**Angle[ poly ]** Draws and returns as variables all the angles of the polygon poly. Note that these angles will be interior or exterior depending on whether the polygon was drawn anti-clockwise or clockwise.

### Special Commands associated with Points

Remember that there are three ways to draw a point. You can use the **New Point** tool. You can enter the coordinates directly into the **Command** window as, say,  $(2, -3)$  and press Return. You can give the point a name as you create it by entering into the **Command** window  $P = (-2, 4)$  and pressing Return.

**Point[ g ]** This draws a random point on the line  $g$ . You can select this point and move it with the arrow but it will be confined to the line  $g$ .

**Point[ c ]** Draws a point on a conic section  $c$ .

**Point[ f ]** Draws a point on the graph of the function  $f$ .

**Point[ poly ]** Draws a point on the polygon poly.

**Point[ v ]** Draws a point on the vector  $v$ .

**Point[ P, v ]** Draws the image of the point  $P$  by the translation represented by the vector  $v$ .

**Midpoint[ A, B ]** Draws the midpoint of the line segment  $AB$  where  $A$  and  $B$  are two points.

**Midpoint[ s ]** Draws the midpoint of the line segment  $s$ .

**Center**[  $c$  ] Draws the centre of a conic section. Note the spelling.

**Focus**[  $c$  ] Draws the foci of the conic section  $c$ .

**Vertex**[  $c$  ] Draws the vertices of the conic section  $c$ .

**Centroid**[  $\text{poly}$  ] Draws the centroid of the polygon  $\text{poly}$ .

**Intersect**[  $g, h$  ] Draws the intersection point of the lines  $g$  and  $h$ .

**Intersection**[  $g, c$  ] Draws up to two intersection points of the line  $g$  and the conic section  $c$ .

**Intersection**[  $g, c, n$  ] Draws the  $n^{\text{th}}$  intersection point of the line  $g$  and the conic section  $c$ .

**Intersection**[  $c_1, c_2$  ] Draws up to 4 intersection points of the conics  $c_1$  and  $c_2$ .

**Intersection**[  $c_1, c_2, n$  ] Draws the  $n^{\text{th}}$  intersection point of the conic sections  $c_1$  and  $c_2$ .

**Intersection**[  $f_1, f_2$  ] Draws all the intersection points of the polynomials  $f_1$  and  $f_2$ .

**Intersection**[  $f_1, f_2, n$  ] Draws the  $n^{\text{th}}$  intersection point of the polynomials  $f_1$  and  $f_2$ .

**Intersection**[  $f, g$  ] Draws all the intersection points of the polynomial  $f$  and the line  $g$ .

**Intersection**[  $f, g, n$  ] Draws the  $n^{\text{th}}$  intersection point of the polynomial  $f$  and the line  $g$ .

**Intersection**[  $f_1, f_2, A$  ] Draws the intersection point between the functions  $f_1$  and  $f_2$  using the point  $A$  as the initial value in Newton's method.

**Intersection**[  $f$ ,  $g$ ,  $A$  ] Draws the intersection point of the function  $f$  and the line  $g$  using  $A$  as the initial point in Newton's method.

**Root**[  $f$  ] Draws the points that are the roots of the equation  $f(x) = 0$ .

**Root**[  $f$ ,  $x_0$  ] Draws the point corresponding to the root of  $f(x) = 0$  calculated by Newton's method using  $x_0$  as the initial value.

**Root**[  $f$ ,  $a$ ,  $b$  ] Draws the root of  $f(x) = 0$ , if it exists, where  $a \leq x \leq b$ .

**Extremum**[  $f$  ] Draws the points which are the local extrema of the polynomial  $f$ .

**InflectionPoint**[  $f$  ] Draws the inflection points of the polynomial  $f$ .

### Special Commands associated with Vectors

**Vector**[  $A$ ,  $B$  ] Draws the vector from the point  $A$  to the point  $B$ .

**Vector**[  $A$  ] Draws the position vector of the point  $A$ .

**Direction**[  $g$  ] Draws the direction vector of the line  $g$ .

**UnitVector**[  $g$  ] Draws the unit vector parallel to the direction vector of the line  $g$ .

**UnitVector**[  $v$  ] Draws the unit vector parallel to the vector  $v$ .

**PerpendicularVector**[  $g$  ] Draws the perpendicular vector to the line  $g$ .

**PerpendicularVector**[  $v$  ] Draws the perpendicular vector to the vector  $v$ .

**UnitPerpendicularVector**[  $g$  ] Draws the unit vector perpendicular to the line  $g$ .

**UnitPerpendicularVector**[  $v$  ] Draws the unit vector perpendicular to the vector  $v$ .

**Special commands associated with lines, line segments, Rays and Polygons**

**Segment**[ **A**, **B** ] Draws the line segment between the points A and B.

**Segment**[ **A**,  $n$  ] Draws the line segment of length  $n$  starting at A. The end-point of the segment is also constructed.

**Ray**[ **A**, **B** ] Draws the ray (half-line) starting at the point A and passing through the point B.

**Ray**[ **A**,  $v$  ] Draws the ray starting at the point A parallel to the vector  $v$ .

**Polygon**[ **A**, **B**, **C**, ... ] Draws the polygon defined by the points A, B, C, ...

**Polygon**[ **A**, **B**,  $n$  ] Draws the regular polygon with  $n$  sides beginning with the side AB.

**Line**[ **A**, **B** ] Draws the line through the points A and B.

**Line**[ **A**,  $g$  ] Draws the line through the point A and which is parallel to the line  $g$ .

**Line**[ **A**,  $v$  ] Draws the line through the point A parallel to the vector  $v$ .

**Perpendicular**[ **A**,  $g$  ] Draws the line through the point A which is perpendicular to the line  $g$ .

**Perpendicular**[ **A**,  $v$  ] Draws the line through the point A which is perpendicular to the vector  $v$ .

**LineBisector**[ **A**, **B** ] Draws the perpendicular bisector of the line segment connecting the points A and B.

**LineBisector**[  $s$  ] Draws the perpendicular bisector of the line segment  $s$ .

**AngularBisector**[ **A, B, C** ] Draws the line bisecting the angle  $\angle ABC$ .

**AngularBisector**[ **g, h** ] Draws the two bisectors of the angles between the lines **g** and **h**.

**Tangent**[ **A, c** ] Draws all the tangents through the point **A** to the conic section **c**.

**Tangent**[ **g, c** ] Draws all the tangents to the conic section **c** which are parallel to the line **g**.

**Tangent**[  $x_0, f$  ] Draws a tangent to the function  $f(x)$  at the point  $(x_0, f(x_0))$

**Tangent**[ **A, f** ] Draws a tangent to the function  $f(x)$  at the point whose  $x$ -coordinate is that of **A**. The point **A** need not be on the graph of  $f$ .

**Asymptote**[ **h** ] Draws the two asymptotes to the hyperbola **h**.

**Directrix**[ **p** ] Draws the directrix of the parabola **p**.

**Axes**[ **c** ] Draws the two axes of the conic section **c**.

**FirstAxis**[ **c** ] Draws the principal axis of the conic section **c**.

**SecondAxis**[ **c** ] Draws the second axis of the conic section **c**.

**Polar**[ **A, c** ] Draws the polar line of the point **A** in the conic section **c**.

### **Special commands associated with the construction of Conic Sections**

**Circle**[ **C, r** ] Draws the circle with centre **C** and radius **r**.

**Circle**[ **C, s** ] Draws the circle with centre **C** and radius equal to the length of the line segment **s**.

**Circle**[ **C, P** ] Draws the circle with centre **C** passing through the point **P**.

**Circle**[  $P, Q, R$  ] Draws the circle passing through the three points  $P$ ,  $Q$  and  $R$ . If the points are collinear GeoGebra still tries to construct a circle of infinite radius.

**OsculatingCircle**[  $P, f$  ] Draws the osculating circle to the function  $f(x)$  at the point  $P$ . Note that GeoGebra does not check to see if  $P$  is on the curve. You must ensure that yourself. This is easy to do by constructing a slider,  $a$ . Then define the point  $P$  as  $P = (a, f(a))$ .

**OsculatingCircle**[  $P, c$  ] Draws the osculating circle to parametrically defined curve  $c$  at the point  $P$ . The same note as previously also applies here.

**Ellipse**[  $F_1, F_2, a$  ] Draws the ellipse with foci  $F_1$  and  $F_2$  and whose principal axis is of length  $a$ . Note that the distance  $2a$  must be greater than the distance between  $F_1$  and  $F_2$ .