MathGraph32 help

Table des matières

Software presentation

Introduction

MathGraph32 Java is a software for plane geometry and analysis specially adapted for teaching purposes.

MathGraph32 is free under Gnu GPL license 3 (or higher).
The project is now a project of Sesamath association.

The software is now written in JavaScript and the files created are compatible with the Java version.

Expandable icon menus are available for a user-friendly usage. Icons are grouped through types. For example the first expandable bar allows creation or modification of points.

The size of icons and interface are now adapted to the window size. If you resize the application window, icon sizes will be adapted on next start of the software.

With MathGraph32 you can easily create any figure of plane geometry, and change dynamically this figure using the mouse .

The figure is drawn in a vectorial format (SVG) for a very good visual result.

You can also draw function curves, parametric or polar curves, graph recurrent sequences type of u(n+1) = f[u(n)] (real or complex).

MathGraph32 implements complex numbers calculus and graphic representation..

Very sophisticated figures are quickly achieved. For instance you can create a point of coordinates (4/3; cos(2**a*)) where *a* is a value already defined or measured.

MathGraph32 Java is visual. No need of learning complicated syntax for a command line. Most objects are created through mouse clicks and dialog boxes.

Figures can be saved in file images (PNG, JPEG, SVG) , copied in the clipboard. You ca even specify a length in cm for the length unity of the figure in order to get a figure with the exact dimensions you need when copied in your favorite text processor.
You ca also generate a HTML file to export the figure on a web page and get the LaTeX-Tikz code of the figure.

MathGraph32 is the only software of his kind allowing creation of objects locus. This allows the creation of very sophisticated figures.

MathGraph32 macros are buttons allowing an action on the figure to be achieved :starting a random experience or an animation, unmasking objects and many other features. Macros are very useful for teaching purposes.

You can choose the level of functioning (four levels available). You can even allow the use of your preferred tools, which will allow teachers to adapt the software for their personal use.

Exported figures are dynamic : You can capture points, launch macros or change formulas directly on a web page.

MathGraph32 is a multi-window software.

To get more information on MathGraph32 please browse to [MathGraph32 website](#)..

The left toolbar

Since version 5.0, a much more friendly-user interface is available.
Icons on the left toolbar are now grouped by type.
Clicking on the blue arrow on the right (or double-clicking an icon) of an icon makes pop-up an horizontal menu with all the icons available for the type of object. The clicked tool replaces the icon on the left.
Here is an example of the first toolbar relative to points. :

The  icon ending some of these toolbars offers other tools of less common usage.

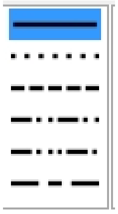
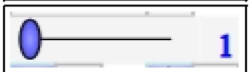
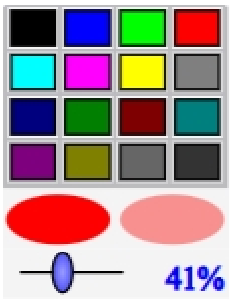
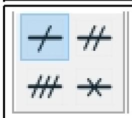
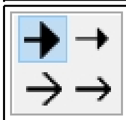
	Select this tool to move a point (free point or linked point), a display or increase or decrease the size of an angl You can also use this tool to translate the whole figure.
	Groups all the icons for the creation and modification of points. Some of these icons may not appear if the tool i
	Groups all the icons for lines creation.
	Groups icons for creation of segments, rays, vectors.
	Groups all the icons for the creation of circles and arcs of circle.
	Groups all the icons for the creation of lines and polygons.
	Groups all the icons for the creation of segment marks or angle marks.
	Groups all the icons for the creation of point locus, function curves, recurrent sequence graphs and object locus
	Groups all the icons for the use of geometrical transformations and virtual protractor.
ou	Groups all the icons for the creation of measure.

	Groups all the icons for text, LaTeX, value display, image or macro.
	Groups all the icons for the creation of calculations, cursors, functions, frames, sequences (real or complex).
	Groups all the icons for the creation of surfaces and half-planes.

To be noted : Some icons will be available only if the context is adapted.

For instance, the  icon will not be available if the figure doesn't include an unity length.

The right toolbar

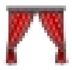
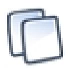
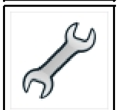
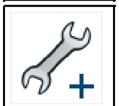
	For the line style selection of future objects created.
	Selection of line thickness.
	The 16 icons are for selecting predefined colors. Click the right ellipse  if you want to use an owner defined color. The ellipse will then take the chosen color. The cursor is for the choice of transparency level of the objects. The right ellipse indicates the rendering of a surface with the chosen color and transparency.
	For the style of created points.
	For the style of segment marks.
	For the style of arrows (vectors and oriented angle marks).
	For the style of angle marks.

	For the filling style of surfaces. The top left icon is for surfaces with transparency level. The black icon is for opaque filling. The rectangle underneath gives a preview of the filling style.
	To end some actions (such as finishing a polygon creation).

To be noticed : Palette tool allows you to modify the color and style of an object (just click on the object).

The upper toolbar

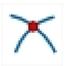
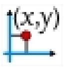
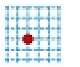
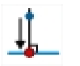
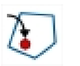
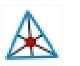
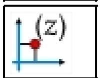
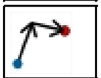
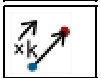
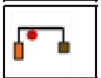
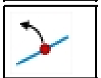
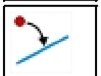
	To display again the last indication (top right corner of the figure)
	For a new figure creation. You will be asked for a figure with or without frame or predefined figures
	To open a figure from a mgj file
	To save the current figure.
	Undo last action.
	Redo last action.
	Deletion of a graphical object. The object will be deleted as well as dependent objects.
	Modify a graphical object when this object needs a dialog box to be changed. For instance, clicking on a point image of another through a rotation will get a popup dialog box appear allowing the change of angle. Simply click on the object to be modified.
	Zoom in.
	Zoom out.
	To list, modify or reclassify numerical objects.
	To change the color, transparency or drawing style of an already created object (line style, point style, arrow style, filling style).
	To select the graphic style of an object (color, transparency, line style, thickness, fill style) in order to paste the graphic style to another object(Palette tool is automatically selected after the click on the first object)
	To rename a point or a line. Simply click on the point or the line.It is also possible to name a point or a line on the fly immediatly after the creation.
	To move the name of a point or a line..
	Toggles activation or deactivation of mode Trace. When mode Trace is on, traces you get via menu Move - Objects trace or via a macro generating a trace remain permanently as long as

	mode Trace is on and the window containing the figure is not resized.
	New calculus and drawing of the whole figure, including new values pour rand() function.
	Magnifying lens. This is on on-off swich button. If selected, when you capture a mobile point, a magnifying glass appears showing the surroundings of the mouse pointer. Particularly useful on tactile devices.
	To mask an object (the object still exists but is no longer visible).
	To unmask a hidden object. Most of hidden objects are in dotted style for this tool.
	When this icon is activated, some tools automatically create segments and angle marks to complete the (perpendicular line, midpoint, perpendicular bisector, bisector). Before using these tools, be sure to select the proper segment mark and angle mark.
	To display the figure protocol. Point and lines without names are given a temporary name and each object of the figure is described in a list. The objects created via a dialog box may be modified. It is also possible to change the order of objects with the arrows on the right panel.
	To execution a macro clicking on the title.
	To export the figure : Copy the figure in the clipboard, export the figure in an image (png, jpeg or svg),get the Base64 code of the figure, get the LaTeX-Tikz code of the figure.
	To get a dialog box popup allowing to change options : Change the angle unity of the figure, change the level of use of the software, personalize available tools, choose to display or not measures when creating the measures, use a selection frame on the figure, choose animation options for tool  (linked point animation), choose a background image for the figure (or delete it), choose the language (English, Spanish and French).
	Toggle tool making visible or hidden an upper toolbar with some other tools of less common usage.
	To reclassify an object towards the end of the list of the created objects, as far as possible. Just click on the object you want to reclassify.
	To reclassify an object towards the beginning of the list of the created objects, as far as possible. Just click on the object you want to reclassify.
	To increase the font-size of all the displays of the figure (one pixel step).
	To decrease the font-size of all the displays of the figure (one pixel step).
	To manage constructions of the figure (i any are present).
	To create a new construction.

Construction icons

[Icons for the creation and modification of points]
[Icons for the creation of lines]
[Icons for the creation of segments, arrays and vectors]

Creation and modification of points.

	Free point creation.
	Intersection of two objects creation (lines, rays, circles, circle arcs).
	Creation of a linked point (linked to line, ray, segment, vector, circle, circle arc, broken line or polygon)
	Midpoint creation.
	Coordinates-defined point creation in a frame.
	Creation of a free point with integer coordinates in a frame.
	Creation of an orthogonally projected point on a line, ray, segment, vector.
	Creation on a point inside a polygon. This point can be moved but will stay inside the polygon.
	Gravity center of a triangle creation.
	Creation of a point defined through abscissa relative to two other points.
	Affix-defined point creation in a frame.
	Creation of a point image of another through a translation vector of which is the sum of two vectors.
	Creation of a point image of another through a translation vector of which is the product of a vector and a number.
	Barycenter creation.
	To suppress the link between a linked point and an object. The point becomes a free point.
	To create a link between a free point and an object. The point becomes a linked point.
	Pinning-down a mobile point. It won't be movable until released.
	Releasing of a pinned point.

	To get a point marked for trace. If Trace icon is activated, the points marked will leave a trace when moving.
	To get a point unmarked for trace. The point will not leave a trace when moving.

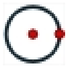
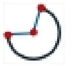
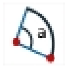
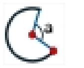
Lines creation.

	Line creation through two points.
	Parallel line creation. Click first on a line, ray, vector or segment then on a point.
	Perpendicular line creation. Click first on a line, ray, vector or segment then on a point.
	Angle bisector creation.
	Perpendicular bisector creation.
	Line defined through point and slope.
	Line defined by an equation in a frame. The equation may be of any kind, for instance 2*x+3*y-5 = 0 or x = rac(2) or y = rac(2)*x+1
	Horizontal line creation.
	Vertical line creation.
	Tangent creation on a curve by specifying abscissa of the point. It is the result of an internal macro-construction.
	Line of linear regression of a set of points. Click on the points then click on the red button STOP. To be noted : A matrix is created along with the line. This matrix contains the coordinates of the points clicked. If your figure contains more than one frame, you will be asked to choose the frame (and confirm the matrix name).

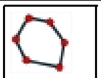
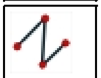
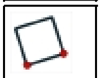
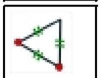
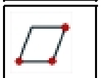
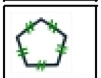
Segments, rays and vectors creation.

	Segment creation.
	Ray creation.
	Vector creation (it is a segment with an arrow not a real vector).
	To create a segment with a given length.

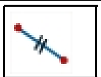
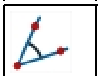
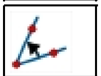
Circles and arcs of circle creation.

	Circle through center and point creation.
	Circle through center and radius creation.
	Little circle arc creation through three points.
	Big circle arc creation through three points.
	Direct circle arc creation through three points.
	Indirect circle arc creation through three points.
	Little circle arc creation through center, starting point and angle value.
	Big circle arc creation through center, starting point and angle value.
	Direct circle arc creation through center, starting point and angle value
	Indirect circle arc creation through centre, starting point and angle value.
	Creation of a little arc of circle with a virtual protractor (only if figure angle unity is degree).
	Creation of a big arc of circle with a virtual protractor (only if figure angle unity is degree).

Polygons and broken lines creation.

	Polygon creation. To end , either click on the red STOP button on the right or click again on the first vertex.
	Broken line creation. To end, click on the red STOP button on the right or click again on the first vertex.
	Direct square creation.
	Direct equilateral triangle creation.
	To create a parallelogram by clicking on three vertexes.
	Regular polygon creation.

Segment marks and angle marks creation.

	Segment mark creation.
	Non-oriented angle mark creation.
	Oriented angle mark creation.

	Oriented angle mark creation always in the direct direction.
	Non oriented angle mark creation always in the direct direction.
	Pinning-down an angle mark. It won't be movable until released.
	Releasing a pinned angle mark

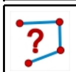
Point locus, function curves, object locus and sequences graphs creation.

	Creation of a linked point locus generated through a linked point.
	Creation of a linked point locus generated through a variable.
	Creation of a not connected point locus generated through a linked point.
	Creation of a not connected point locus generated through a variable.
	Graph of a real function creation in a frame (on a segment or an interval).
	Graph of a real function on an interval with one or two brackets at the interval edge.
	Graph of a function going through *n* given points giving the tangent slope at these points (*n* ranging from 2 to 10) If you check the checkbox **Use only one polynomial function**, an unique polynomial function filling the required conditions will be created and you can get the coefficients of this function in a matrix name of which starts with *coef.* If you don't check the checkbox **Use only one polynomial function**, on each interval changing the formula, the formula of a polynomial function of degree 3 will be used. In both cases, a function is created, it's name starting with *f.* **Beware :** If you don't check **Use only one polynomial function,** the order of the x-coordinates of the clicked points must not be changed for the curve to remain accurate.
	Graph of a polynomial function curve going through *n* points (*n* ranging from 2 to 8).
	Creation of the graph of a real recurrent sequence kind of u(n+1)=f[u(n)].
	Creation of the graph of a complex recurrent sequence kind of z(n+1)=f[z(n)]., where *f* is a complex function.
	Creation of an object locus generated through a linked point.
	Creation of an object locus generated through a variable.

Geometrical transformations.

	Creation of image objects through a translation. Click first on the starting point and ending point of the translation vector. Vector of translation blinks. Then click on an object to create it's image (line; ray, segment, circle, circle arc, polygon, broken line). You can click on several objects (as long as the vector is blinking).
	Creation of image objects through a axial symmetry. Click first on the symmetry axis (line, ray, segment). Axis blinks. Then click on an object to create it's image (line; ray, segment, circle, circle arc, polygon, broken line). You can click on several objects (as long as the axis is blinking).
	Creation of image objects through a central symmetry. Click first on the symmetry center (point). Center blinks. Then click on an object to create it's image (line; ray, segment, circle, circle arc, polygon, broken line). You can click on several objects (as long as the center of symmetry is blinking).
	Creation of image objects through a rotation. Click first on the rotation center (point). A dialog box pops up. Enter the angle of rotation. Then click on an object to create it's image (line; ray, segment, circle, circle arc, polygon, broken line). You can click on several objects (as long as the center of rotation is blinking).
	Use of a virtual protractor in order to rotate a point around another one.
	Creation of image objects through a translation by giving the translation vector coordinates in frame. Enter first the vector coordinates in the dialog box then click on an object to create it's image (line; ray, segment, circle, circle arc, polygon, broken line). You can click on several objects (as long as the translation vector is blinking).
	Creation of image objects through a dilation. Click first on the dilation center (point). A dialog box pops up. Enter the dilation ratio. Then click on an object to create it's image (line; ray, segment, circle, circle arc, polygon, broken line). You can click on several objects (as long as the center of dilation is blinking).
	Creation of image objects through a direct similitude. Click first on the dilation center (point). A dialog box pops up. Enter the similitude ratio and angle. Then click on an object to create it's image (line; ray, segment, circle, circle arc, polygon, broken line). You can click on several objects (as long as the center of similitude is blinking).

Measures.

	Length measure.
	Non-oriented angle measure.
	Oriented angle measure.
	x-coordinate in a frame measure.
	y-coordinate in a frame measure.
	Abscissa measure relative to two other points. Points have to be aligned.
	Line slope measure in a frame measure.
	Polygon or broken line length measure.

	Area of a polygon measure.
	Scalar multiplication measure.
	Affix in a frame measure.

Value, text, LaTeX displays and formula editors.

	Free value display creation.
	Linked to point value display creation.
	Free text display creation.
	Linked to point text display creation.
	Free LaTeX formula display.
	LaTeX formula display linked to a point.
	Creation a a free image display.
	Creation of an image display linked to a point.
	Creation of a formula editor associated with a calculation or a function.
	Creation of a link between a display (text, LaTeX, value display, macro) and a point.
	Suppression of the link between a text and a point.
	Pinning-down a display (text, LaTeX, value display, macro, formula editor). It won't be movable until released. To be noted : a pinned down display can't be modified. To get it modified you ave to unpin it first.
	Releasing a pinned display (text, LaTeX, value display, macro, formula editor). To be noted : a pinned down display can't be modified. To get it modified you ave to unpin it first.

Cursors, calculations, variables, functions and sequences.

	Creation of a cursor. The cursor may be horizontal or vertical, with integer or not integer values.
	New real calculus creation.
	Creation of a real function of one variable and if wished drawing of it's curve.

7→a	Creation of a variable (defined through mini, maxi, actual values and incrementation step).
	Frame creation.
	Calculation of an approximate solution of an equation.
	Calculation of the abscissa of the maximum of a function on a segment.
	Calculation of the abscissa of the minimum of a function on a segment.
$f'(x)$	Creation of the derivative function o a real function
$u_{n+1} = f(u_n)$	Real recurrent sequence creation type of u(n+1)=*f*[u(n)] where *f* is a real function.
$z_{n+1} = f(z_n)$	Complex recurrent sequence creation type of u(n+1)=*f*[u(n)] where *f* is a complex function.
z=	New complex calculation.
f(z)=	New complex function of one variable.

Surfaces.

	Surface creation delimited by a circle, polygon, circle arc, point locus (closed). See the surface.
	Surface creation delimited by an arc of circle and it's chord.
	Surface creation delimited by a point locus and a line. Locus extremities are projected on the line to define the edge.
	Surface creation delimited by a point locus and two points.
	Surface creation delimited by two point locus.
	Crown creation (delimited by circles or polygons)
	Half-plane creation.

Visual help for objects creation

MathGraph32 gives graphic information on a graphic object while creating the object.

For instance, when using the icon of circle creation through two points, after clicking on the center, a temporary circle appears, which center is the first point clicked and including a temporary point located at the mouse pointer position.

How to modify an object

Modification of a graphical object.

To modify the graphical characteristics of a graphical object, select it's characteristics in the right toolbar then activate the palette tool and click on all the objects to be modified.

When a graphical object has been created via a dialog box, you have to select the graphical object modification tool the click on the object. The dialog box will pop-up and allow you to modify the object.

Modification of a numerical object.

To modify a numerical object, you can use :

- Tool of numerical object modification).
- Tool of figure protocol.

Renaming a numerical object.

To rename a numerical object, the toolbar relative to calculations (third from the bottom) and click on icon . Choose *Rename a calculation* in the list.

Modification of a a frame.

With tool (graphical object modification) : Click on the frame in the list then click on button *Modify* (or double click) .

With tool (figure protocol) : Click on the frame in the list then click on icon (oor double click).

Masking ou unmasking an object.

To mask an object, click on tool the click on the object you want to hide.

To unmask a masked object, click on tool and click on a blinking object.

Pinning-down a free point or a linked point.

In the points icons, select icon and click on the point (free point or linked point). It won't be movable until it is unpinned.

Unpinning a pinned-down point.

In the points icons, activate tool then click on a point which has been pinned-down. This point can now be captured.

Marking or unmarking a point for traces.

A point marked for trace will write down a trace during the capture of a point (or some macro executions).

For this trace to be visible, you must activate the trace for the active figure, clicking on icon (which must be highlighted).

To get a point marked for trace trace, use tool and click on the point.

To get a point unmarked for trace trace, use tool and click on the point.

Adding scales to a frame.

expand the toolbar relative to calculations (thirs from the bottom), click on icon then choose *Add scale to a frame*.

Figure protocol

Clicking on icon , a dialog box pops up on the right side of the figure.
This dialog box displays all the created objects in a list.
Before the creation of the dialog box, points and lines without names are given a temporary name so that all the objects of the figure can be fully described.
The left side keys allow the user to navigate through the objects of the list.
If graphical, the selected object of the list is blinking, as well as the objects it is created with.

The right side arrows allow the user to modify the relative order of the objects :

: Reclassify the selected object of one step towards the beginning of the list. If necessary, the objects it is depending on will also be reclassifed.
: Reclassify the selected object of one step towards the beginning of the list. If necessary, the objects depending on it on will also be reclassifed.
: Reclassify as far as possible the selected object towards the beginning of the list.
: Reclassify as far as possible the selected object towards the end of the list.

The other icons :

: Destroys the selected object and the objects depending on it.

: Destroys the selected object and the following objects.

: Unmask the selected object (if it was masked).

: Allows to modify the selected object if it was created through a dialog box. For instance, if the selected object is an image point through a rotation, a dialog box will popup allowing the change of the rotation angle.

The check boxes :

Intermediary objects : If checked, intermediary objects of the figure will be displayed in the list. Uncheked by default.

Masked objects visible : If checked, masked objects remain visible (unchecked by default).

Naming a point or a line

Naming a point or a line on the fly.

Only points and lines can be given a name.

When a point or a line is created, an editor appears on the figure, allowing you to name it.
If you don't wish this point or line to be named, just select another tool or click on the figure.

Naming a point or a line after it's creation.

Use tool of the upper toolbar, click on the point or the line then fill in the dialog box.

Moving a name of a point or a line.

Use icon of the upper toolbar, the drag the name when the mouse pointer is a hand.

Object deletion

Deleting a a graphical object.

Two possibilities with the upper toolbar :

Tool : Click on the object which is to be suppressed. If the object deletion implies the deletion of other objects you will be prompted for agreement.

Protocol tool . A dialog box will popup with all the created objects (numerical and graphical) displayed in a list. Select the object in the list the click on icon .

Deleting a numerical object.

Two possibilities with the upper toolbar :

Tool of numerical objects modification. A dialog box will popup with all the numerical objects displayed in a list. You can click on the **Delete** button to get the selected object suppressed.

Protocol tool . A dialog box will popup with all the created objects (numerical and graphical) displayed in a list. You can click on the button to get the selected object suppressed.

New figure and predefined figures

To create a new figure, use icon .

New figure with frame :

A dialog box pops up displaying choices for the frame.
You can create an frame of type orthonormal, orthogonal, oblique or oblique withh equal length unities.
You can also choose different types of grid, of scaling (including trigonometry scaling).
You can ask to get vectors on each axis..
You can also choose the x-coordinate origin, the y-coordinate origine, the unit on each axis..
Length OI is set as length-unity of the figure.

New figure with millimetered frame :

For the creation of a figure with a frame and a millimetered grid.

Figure without frame and with length unity :

A new figure is created with a segment [UV] on the top left corner of the figure. The length of this segment is the length-unity of the figure.

Figure without frame and with length unity :

A new figure is created without legth unity. So you won't be able to measure lengths unless you choose a length unity afterwards.

To choose a length unity for a figure, use icon at the botoom of the left toolbar and choose *Unity Length Creation.*

Basic figure :

Choose the type of basic figure in the dialog box and validate.

Figure with alterable frame without vectors :

You get a figure with a frame without vectors on the axis
Macros on the figure allow the user to transform the frame :

- orthonormal
- orthogonal
- oblique
- oblique with equal length unities on each axis

Figure with alterable frame with vectors :

Same as above with vectors on each axis.

Figures with graphs of functions or sequences :

You can choose one of the following figures :

- Function curves : Up to 3 curves in a frame.
- Parametric curve.
- Polar curve.
- un = f(n) sequence graph.
- u(n+1) = f (u(n)] sequence with first term movable .The first term of the sequence can me modified moving a linked point. Click on + and - buttons to make terms appear gradually.
- u(n+1) = f (u(n)] sequence with first term editable. The first term can be changed using a formula editor. Click on + and - buttons to make terms appear gradually.

Figure of probability :

- Binomial distribution.
- Binomial distribution with confidence interval.
- Binomial distribution approximated through normal distribution.
- Exponential distribution.
- Normal distribution.
- Normal distribution with computing of u_{alpha}.
- Normal distribution inversed.

Basic 3D Figures :

15 figures are available.
One of the allowing the graph of a z = f(x, y) surface.
Movable points allow the user to turn the figure and change the viewing angle.

Levels of use

The four predefined levels of use.

MathGraph32 can run under four predefined levels :

- Elementary level.
- Mid-level.
- Advanced level without complex numbers calculation.
- Advanced level with complex numbers calculation.

On the first start of MathGraph32, the level in use is advanced with complex numbers.

To change the level of use, use menu item *Options - Preferences* et select the chosen level.

The available tools are adapted to the chosen level.

To change the level of use, use icon on the top toolbar and select one of the four predefined levels.
Check the checkbox *Use this choice for next start* and validate.
The chosen level will be active for next start.

Personalizing the level of use.

Use icon of the top toolbar.
To filter the available tools, select on of the four predefined levels.
Check the checkbox ***Personalize available tools****.* A new button **Tools choice** appears.Click on this button.
A dialog box pops up. Choose the tools you want to allow or forbid.
Check **Use this choice for next start** if you want this choice to be memorized.

You can also use an already created figure (in which a tools choice has already be done) as base for the tools choice by clicking on button **Tools choice from figure**. Then you have to browse to the directory containing this figure, select the then click on button **Open**. Then you can modify the tools forbidden or allowed.

Créer une figure avec des outils personnalisés lors de son ouverture.

Use icon of the top toolbar.
To filter the available tools, select on of the four predefined levels.
Check the checkbox *Personalize available tools.* A new button *Tools choice* appears.Click on this button.
A dialog box pops up. Choose the tools you want to allow or forbid and validate.
The tools available in the left toolbar are now the chosen tools.
If you save your figure, these tools will be the available tools when opening next the figure.

Figure options and preferences

Options for the current figure and preferences are available by clicking on icon of the top toolbar .
A dialog box pops up.

Angle unity tab changes the angle unity of the current
.
Underneath you can choose the level of use :

- Elementary level.
- Mid-level.
- Advanced level without complex numbers calculation.

- Advanced level with complex numbers calculation.

If the checkbox **Use chosen level for next start** is checked, this level will be used on next start of the software.

If the checkbox **Use dys mode for next start** is checked, on next start, the default point style will be a big point, the default line thickness will be 2 pixels and indications on the figure will be bigger. The creation of point on click will first be deactivated.

Starting figure is for the figure displayed on start of MathGraph 32 : Frame with grid or Frame dotted or Empty figure with unity length.

When the check box **Automatic display of measures** is checked, measures are automatically displayed on the figure.

If the checkbox **Display a frame of given dimensions** is checked, a selection rectangle will appear on the figure. Enter the rectangle dimensions in pixels (starting frm the top left corner of the figure). This rectangle is not part of the figure, just a help.

Scale factor for export of images : If you choose a factor bigger than 1 the quality of the figure will be better (but the image will be bigger).

If you click on :

Fixed objects : You will choose if angle marks or displays are movable or not by tools  and .

Animation options, you will choose animation options when using tool  of linked point animation (in the first left expandable bar).

Background image choice, you will choose an image file for the background image of the figure (Click on button **browse**).

Background image deletion, you will be able to delete the current background image (if present).

Figure background color, you can choose a background color for the figure (default is white)..

Language, you can change the language of the software for next start (French, English or Spanish).

About MathGraph, an about dialog box pops up.

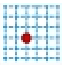
MathGraph32 website, your browser opens a window on MathGraph32 website.

Graphical objects

The free point	The free point with integer coordinates
The constructed point	The linked point
The point through two vectors sum	The point through vector multiplication by a number

The free point :

It is a point which can be freely moved on the figure.

The free point with integer coordinates : 

It is a point which can be freely moved on the figure but with coordinates remaining integer values.
You can restrain the position in the two axis.

The constructed point :

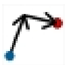
It is a point which has been defined through intersection of two objects or a point image of another through a geometrical transform.

The linked point : 

That kind of point can be moved but must remain on the object he is linked tot.
You can link a point to a line, ray, segment, segment, circle, polygon, broken line, circle arc or point locus.

A free point can be turned in linked-point through tool  .

A linked point can be turned in a free point through tool  .

The point through two vectors sum : 

C'est un point qui est l'image d'un autre par une translation dont le vecteur est la somme de deux vecteurs.

The point through vector multiplication by a number : 

This points is image of another point through a translation vector of which is the multiplication of a vector and a number.

The coordinate defined point : 

It is a point defined through it's coordinates in a frame.

The point inside a polygon or a circle : 

This kind of point is created by clicking inside a polygon or a circle.
This point can be captured with the mouse through the tool but it will stay inside the polygon.

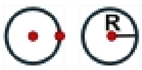
This kind of point may be turned into a free point through tool .
A free point can be turned into a point inside a polygon through menu item *Modify - Link creation - Between point and the inside of a polygon*.

The line : 

A line ca be defined through two points, as parallel, perpendicular, angle-bisector, perpendicular-bisector, horizontal line, vertical line, defined through point and slope, defined by a line equation or a line of linear regression.
You can also create the image of a line trough geometrical transformations (inversion excepted).

The segment : 

Un segment est créé en désignant ses deux extrémités.
On peut aussi créer l'image d'un segment par une transformation (excepté l'inversion).
A segment is created by clicking on two points.
You can also create the image of a segment trough geometrical transformations (inversion excepted).

The circle : 

A circle is created trough it's center and a point or through it's center and radius.
You can also create the image of a circle trough geometrical transformations (inversion excepted).

The ray : 

A circle is created trough it's origin and a point.
You can also create the image of a ray trough geometrical transformations (inversion excepted).

The vector : 

A vector is created through it's starting and ending points.

The circle arc : 

A circle arc is created :

- either trough three points : you click first on the center point, the on the starting point, finally on a point giving the direction of the end of the arc (first four icons).
- either through two points and the value of the oriented angle giving the end of the arc (last four icons).

The polygone :

A polygon is created by clicking on each vertex. Click on the red buttton STOP when the last vertex has been specified (or click again on the first vertex).
You can also create the image of a polygon trough geometrical transformations (inversion excepted).

The broken line :

A broken line is created by clicking on each vertex. Right click when the last vertex has been specified.
You can also create the image of a broken line trough geometrical transformations (inversion excepted).

The angle mark :

You can create :

- a non-oriented angle mark :
- an oriented-angle mark in the shortest way :
- a direct-oriented angle mark : .
- a indirect-oriented angle mark : .

The segment mark :

A segment mark must be associated with a segment.
You simply click on the segment.
The angle mark is created in the current color and line-style.

The text display :

A text display is able to contain references to calculus or variables previously defined.
It is also able to display greek or special characters.

A text display can be free ( ) or linked to a point ( ).

You can choose the horizontal alignment (left, centered, right aligned) and the vertical alignment (top, centered, bottom)
For the future characters to be in italic, enter the sequence *#I*.
For the future characters to be bold, enter the sequence *#G*.
For the future characters to be underlined, enter the sequence *#U*.
To return to the normal font enter the sequence *#N*.
To write *chaine in exponent* use #H(*chaine)*.
To write *chaine in index* use #H(*chaine)*.
You can insert LaTeX code inside a text display by surrounding the LaTeX code between two characters $.
Buttons on top of the dialog box insert the most common formulas.

A free text display can be linked to a point with tool .

A text display linked to a pointed can be freed with tool .
Since version 6.0, you can specify the angle of the display. This angle is relative to the horizontal and must be set in the figure angle unity.
The rotation is relative to the anchor point of the display.

The value display :

Allows you to display a calculated or measured value on the figure.

A value display can be free ( ) or linked to a point ( ).
You can specify the number of digits.
You can also specify :
a starting string that will be displayed before the value.
an-ending string that will be displayed after the value.

You can choose the horizontal alignment (left, centered, right aligned) and the vertical alignment (top, centered, bottom).

A free value display can be linked to a point with tool .

A value display linked to a pointed can be freed with tool .Since version 6.0, you can specify the angle of the display. This angle is relative to the horizontal and must be set in the figure angle unity.
The rotation is relative to the anchor point of the display.

The LaTeX display:

Allows you to display LaTeX code on the figure.

A LaTeX display can be free ( ) our linked to a point ( ).

You can choose the horizontal alignment (left, centered, right aligned) and the vertical alignment (top, centered, bottom)

The LaTeX code may contain dynamic references to the value of a variable or a calculus.
The LaTeX creation dialog box contains special icons for inserting the most popular LaTex sequences.
Buttons on top of the dialog box insert the most common formulas.

A free LaTeX display can be linked to a point with tool .

A LaTeX display linked to a pointed can be freed with tool .
Since version 6.0, you can specify the angle of the display. This angle is relative to the horizontal and must be set in the figure angle unity.
The rotation is relative to the anchor point of the display.

The formula editor :

Allows you to display a text editor for the formula of a calculation or a function (real or complex).

You click first at the spot the editor will be located to.
Then a dialog box pops up.
You can choose the calculation or the function associated with the editor, the size of the editor an a starting string to be displayed in front of the editor.
If the starting string is empty, the name of the calculation or the function will be displayed followed by the signe = (for instance if the calculation is a fonction of the *x* variable, *f(x)* = will be displayed in front of the text editor).
If the *Latex display of the formula* checkbox is checked, the formula entered in the editor will be displayed in a natural form on the right size of the editor (LaTeX style).

The connected point locus :  

It is drawn by linking the positions of a point through segments.
The positions can be generated :

- either through the positions of a point linked to an object :  .
- either through the values of a variable :  .

You can specify :

- the number of points calculated.
- If the locus is closed or not

The dotted point locus :  

It is drawn by tracing the positions of a point.
The positions can be generated :

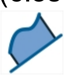
- either through the positions of a point linked to an object :  .
- either through the values of a variable :  .

You can specify the number of points calculated.

The surface :    

A surface is drawn by filling a part of the figure.

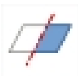
 : Creation of a surface delimited by a polygon, a circle, a circle arc or a point locus (closed).



 : Creation of a surface delimited by a point locus and two points.

 : Creation of a surface delimited by a point locus and a line.

The surface is drawn in the active color and filling style.

The half-plan : 

It is created by clicking on a line an on a point belonging to the half-plan.
The half-plan is drawn in the active color and filling style.

The graph of real u(n+1)=f[u(n)] recurrent sequence :

It is drawn in the active color and line-style.
It is the traditional web-graph of a real sequence type of u(n+1)=f (u(n)] where f is a function.
You can link the points to the x-axis or not.

The graph of complex u(n+1)=f[u(n)]l recurrent sequence :

It is drawn in the active color and line-style.
It is the graph of a real sequence type of u(n+1)=f (u(n)] where f is a function.
Each complex term is represented by a point and linked to the next point trough a segment.
The points are drawn in the active color, active point-style and active line-style.

The image display :

Allows you to display in the figure an image located in a file.

It can be free ( ) or linked to a file ( ).

You can choose the horizontal alignment (left, centered, right aligned) and the vertical alignment (top, centered, bottom).
Since version 6.0, you can specify the angle of the display. This angle is relative to the horizontal and must be set in the figure angle unity.
The rotation is relative to the anchor point of the display.

The object locus :

The object locus is a set of traces of an object.

It can be generated through the positions of a linked point ( ) or through the values of a variable ( ).
The number *N* of object traces is limited to 100000. This number can be dynamically defined.
If the object locus is defined trough a linked point :
The software simulate *N* positions of the linked point on the object it is linked to. For each position, a trace of the object is drawn.
If the object locus is defined trough a variable :
The software simulate *N* values of the variable from the mini to the maxi value. For each value, a trace of the object is drawn.
To designate a n object locus, simply clic on one of the traces.
To be noticed :

> To change the color of an object locus, use the palette tool ( ).
> An object locus doesn't own a line-style. The line style in which it is drawn is the line style of the object generating the traces.
> You can create locus of locus of objects (and so on) but the total number of traces must not exceed 100000.

The duplicated of object :

Use icon at the bottom of the left toolbar.
In a few cases, on object can be hidden by objects defined afterwards.
You can then create a duplicate of this object which function is to redraw the object.
Only one duplicate can be created for an object in a figure.

The clone of object :
It is an object playing exactly the same role as an another object previously defined.
This kind of object is particularly useful for iterative and recursive constructions appeared in 4.8 version of MathGraph32.
I is possible to create a clone of the following objects : point, line, half line, segment and circle.

The cursors

MathGraph32 cursors are not real objects but are the result of internal constructions.
A cursor gives the ability to get a numerical value vary by capturing a point linked to a segment.

MathGraph32 cursors are more powerful than cursors of other software because the mini and maxi values of the cursor may be dynamic.

A cursor can be horizontal or vertical.
The values of the cursor may be continuous or evenly spaced with a step.

A cursor is obtained via the tool .

A dialog box pops up.

Choose the characteristics of the cursor.
You can ask for scales.
Enter the mini and maxi values of the cursor. You can use an existing value or enter a calculus. In the second case a calculus will be added to the figure, with a name beginning with min or max.

Cursors are created in the active color and line style.

Point and object traces

When icon is activated, points marked for trace leave a trace when moving.
Some macros also leave traces of objects when executed. When trace mode is activated, these traces remain on the figure.

To get a point marked for trace *use icon* .

To get a point unmarked for trace, use *icon* .

Export of the figure

With tool of the top toolbar you can get the figure under different formats :

- Copy the figure in the clipboard.
- Copy the figure in the clipboard giving a value in cm for the length unity of the figure (if

present).
- Get a PNG image file.
- Get a JPEG image file.
- Get a SVG image file.
- Get a PNG image file giving a value in cm for the length unity of the figure (if present).
- Get the Base64 code of the figure (a big String of characters with witch a figure v=can be displayed online)
- Get the LaTeX-Tikz code of the figure.

Pour images export and clipboard use, you can improve the resolution of the image using a scale factor.

For this, use icon of the top toolbar and enter a value in the editor*Scale factor for export of images* (ranging from 0.25 to 0.5). Generally a value of 2 will improve the quality of the exported image when the figure will be pasted in your word processor.

To be noticed : When exporting the figure with unity length in cm, if the size of the image is not accurate after pasting it (it may happen under Libre-Office), you can manually specify the figure dimensions. When the dialog box asking for the unity length in cm is opened, check the dimensions of the figur displayed under the editor.
When you paste the figure in your word processor, give to this image the dimensions you noted in the dialog box (right click on the image and Format under Libre-Office).

Important : To use a selection frame, use icon on the top toolbar.
Check *Display a frame of given dimensions*.
Enter the dimensions in pixels.
The frame appears on the figure.
If a selection frame is present, Only the inside of this frame will be exported.

Images

You can create a free image display by clicking on icon .

You can create an image display linked to a point by clicking on icon .

You will be asked to fetch the image on your computer.

A MathGraph32 figure can also contain a background image. This image is also directly included in the mgj file generated.

You can choose the background image with tool of the top toolbar.

The background image can be momentarily hidden with this tool.

Base64 code and export

When a MathGraph32 figure is included in a web page, she is included as a big string of characters (named Base64 string).
It is possible to create a new figure from this Base64 code).

Let us say you are interested in a MathGraph32 figure included in a web page.

- Ask your browser to display ths source code of the html page.
- In this source code, look for an instruction kinf of var st = "*TWF0aEdyYXBoSmF2YTEuMAAAAA8 +TMzNAANmcmH###8BAP8BAAAAAAAAAAACigAAAa4AAAAAAAAAAAAAAAEA*"; only the beginning of the string is displayed here).
- Copy this Base64 string in your clipboard.
- In MathGraph32, use icon  (new Figure). Click on *Figure through Base64 code.*
- A dialog box pops up.
- Paste the Base64 string in the editor of teh dialog box ans validate.
- A new figure appears. You can now modify it and save it or export it

On the contrary, you can also get the Base64 code of a figure through  (export icon).

Links between objects

Link point-objet.

The icons underneath can be found in the first expandable bar of the left toolbar.

A linked point is a point linked to an object. It can me moved but must stay on the object.

Turning a linked point in a free point : 

The link between a point and an object can be deleted with this tool .

Turning a free point in a linked point : 

A free point can be turned in a linked point with this tool.
Click first on the free point which is to become a linked point. Then click on the object the point must be linked to. The link creation is impossible if the object the free point is to be linked to depends on the free point.

Liaison display-point.

The icons underneath can be found in the fourth expandable bar of the left toolbar sgtarting fro the bottom.

Text displays, value displays, LaTeX displays and macros can be linked to a point.

Turning a linked display in a free display : 

The link between a display and an object can be deleted with this icon.

Turning a free display in a linked display : 

A free display can be turned in a display linked to a point with this icon.
Click first on the free display. Click the on the point the display is to be linked to.

Special displays

In text displays, value displays and titles of macros you can get special displays useful for writing mathematics.

Special sequences of characters :

- **#G** will display following characters in bold .
- **#I** will display following characters in italic .
- **#N** will return to normal characters (neither italic nor bold).
- **#H(***chaine***)** will display *chaine* in exponent.
- **#L(***chaine***)** will display *chaine* in index.

Dynamic display of values in text displays :

You can insert dynamic displays of values in text displays :

Dynamic values that can be dynamically displayed in text displays are calculus (real or complex) and variables.

To insert inside a text display the value of a calculus or variable, insert the following sequence of characters : #Val(*Name of the variable or calculus)*.
The default number of digits displayed will be 2.
If you want to specify the number of digits to display, insert sequence : #Val(*Name of the variable or calculus*, *Number of digits)*.
If you wand the display to begin with a plus sign when the value is positive or null, insert sequence #Val(*Name of the variable or calculus*, *Number of digits*, +*)*.

To be noticed :

- You can use the **Dynamic value insertion** button to get a dialog box that will get the special sequence for the dynamic display of value**.**
- A text display can only display values defined before the text display. If necessary, use object reclassification.
- You can display the result of a measure in a text display. To do so, you must create a calculus containing the measure as a formula and ask for the dynamic display of the calculus.

LaTeX code display

MathGraph32 allows the user to display mathematics formulas written in LaTeX code in text displays et in LaTeX displays.

To display LaTeX code in a text display (even in a text display with several lines), get the LaTeX code surrounded by two characters $.

To display LaTeX code in a LaTeX display, simply write the LaTeX code.

In both cases, the result is visible and blinking when you type the LaTeX display in the dialog box.

Buttons at the top of the edit dialog box allow the user to get the most usual LaTeX codes. You just have to complete the code inside the brackets. Pressing the TAB key will send the caret inside the next pair of brackets.

MathGraph32 uses specific LaTeX codes :

- \Val allowing the user to display dynamically the value of a calculus or a varaible in a formula.
- \FracRed allowing the user to display an irreducible fraction equaling a given fraction by specifying the numerator and denominator of the fraction.
- \If : allowing a conditional display inside LaTeX code.
- \For : allowing the display of a formula with LaTeX style.
- \ForSimp : allowing the display of a formula with LaTeX style with replacement of the values used in the formula and simplification of addition of 0 and multiplication, or division by 1.
- \ForSimpFrac : allowing the display of a formula with LaTeX style with replacement of the values used in the formula by approximated rational fractions and simplification of addition of 0 and multiplication, or division by 1.
- \Calc : allowing the display of a formula with inline style (with calculation signs).
- \CalcSimp : allowing the display of a formula with inline style (with calculation signs) with replacement of the values used in the formula and simplification of addition of 0 and multiplication, or division by 1.
- \Decomp : returns the primary decomposition of an integer using exponents.
- \DecompFull : returns the primary decomposition of an integer without using exponents.

To be noticed : the name of these specific LaTeX functions begin with a upper case character.

\Val : To dynamically display a value in a LaTeX formula.

This LaTeX function allows you to dynamically display the value of a calculus (real or complex) or a variable in a MaTeX formula.

If you want to dynamically display another numerical object (a measure for instance), create first a calculus containing this value.

Let us assume that *num* is a calculus or a variable already defined and you wish to display a fraction with *num* as numerator and 2 as denominator.

Click on icon LaTeX to create a display of a LaTeX formula.

Click on button a/b that inserts code LaTeX \frac{ }{ }.
Set the caret position inside the last pair of brackets and type 2, the set the caret position inside the first pair of brackets and press button **Dynamic value insertion**.
Click on *num* and validate.You get the following LaTeX code : \frac{\Val{*num*} {2}} that will display the display of *num* value as numerator for the fraction with two digits.

You can also get another number of digits or get a + sign before the value when it is positive. For instance, code will display a fraction with *num* as numerator (with three digits) and a + sign at the beginning if the value is positive.

A more sophisticated use is available :

- \Val{*num, dec,+0*} displays *num* value with *dec* digits, preceded with a + sign before if it is positive or null, display invisible if *num* value equals 0.
- \Val{*num, dec,0*} displays *num* value with *dec* digits, display invisible if *num* value equals 0.
- \Val{*num, dec,+1*} displays *num* value with *dec* digits, display replaced by a + sign if *num* equals 1 and a signe - if *num* equals - 1.
- \Val{*num,dec,1*} displays *num* value with *dec* digits, display invisible if *num* equals 1 and by a - sign if *num* equals -1.

\ValFrac : To dynamically display a value in a LaTeX formula in the form of an approximated fraction.

The syntax is : \ValFrac{*number*}.

number is replaced by the approximated rational fraction gotten from the continue fraction algorithm (approximated to 10^(-12)).

To display a matrix *mat* with the terms approximated by rational fractions, use syntax \ValFrac{mat}

If *number* is a real value, you can also use syntax \ValFrac{*number*,+} if you wish the number representation to start with a plus sign when the number is positive or null.

\FracRed : To display a rational fraction in the most simple way*.*

The syntax is as follows :

\FracRed {*num*, *den*} ou \FracRed {*num*, *den*, *code*}

Both will display an irreducible fraction equaling fraction of numerator *num* and denominator *den*.
If the fraction equals an integer number, this integer number will be displayed (not a fraction).
If *num* or *den* are not the names of calculuses, or are nor of integer value, or if *den* value is 0, *Frac Error* will be displayed.

If *code* equals +0, the fraction will be preceded with a + sign when the result is positive and will disappear when the result is 0.
If *code* equals 0, the fraction will disappear when the result is zero.
If *code* equals +1, the fraction the fraction will be preceded by a + sign when the result is positive and a - sign when it is negative.
If *code* equals 1, the fraction will disappear when the result equals 1 and will be replaced by a - sign when the result is -1.

\If : to create a conditional display of LaTeX code inside a LaTeX display.

The syntax is : \If {*test*} {code1} {code2} where *test* is a real calculus or a variable already created.

If *test* value is 1 then LaTeX code *code1* is dispalyed et otherwise LaTeX code *code2* is displayed.

code1 et *code2* may themselves use the command \If and this with a some interweaving level.

\For : to create the display of the formula of a calculation or function (complex or real).

The syntax is : \For {*calc*} where *calc* is the name of a calculation or a function.

\ForSimp : To display dynamically the simplified formula of a calculation or a function (real or complex).

The syntax is : \ForSimp {*calc*} where *calc* is a calculation or a function already defined. The calls to variables or calculations are replaced by their numerical value and the formula displayed is simplified as much as possible (for instance the null terms of a sum are dismissed and 1*x is replaced with x).

\ForRep : To display dynamically the formula of a calculation or a function (real or complex) without simplification.

The syntax is : \ForRep {*calc*} where *calc* is a calculation or a function already defined. The calls to variables or calculations are replaced by their numerical value and the formula displayed without any simplification.

\ForSimpFrac : To display dynamically the formula of a calculation or a function (real or complex) with real values replaced by an approximated rational fraction.

The syntax is : \ForSimpFrac {*calc*} where *calc* is a calculation or a function already defined.

Variables and calculations used in the formula of *calc* are replaced par their approxiamted rational fraction (rounded to 10^(-12)) and the formula displayed is simplified as much as possible (for instance the null terms of a sum are dismissed and 1*x is replaced with x).

\Calc : To display the inline formula of a calculation or a function.

The syntax is : \Calc {calc} where *calc* is the name of a calculation or a function.

\CalcSimp : To display the simplified inline formula of a calculation or a function.

The syntax is : \CalcSimp {calc} where *calc* is the name of a calculation or a function. The calls to variables or calculations are replaced by their numerical value and the formula displayed is simplfied as much as possible (for instance the null terms of a sum are dismissed and 1*x is replaced with x).

\Decomp : to display the primary decomposition of an integer using exponents.

The syntax is : \Decomp {*number*} where *number* is a real calculation or a variable already defined.

If *number* is not an integer, returns the decimal display of the number (with 12 digits max).

If *number* is -1, 1 or 0 returns -1, 1 or 0.

\DecompFull : to display the primary decomposition of an integer without using exponents.

The syntax is : \DecompFull {*number*} where *number* is a real calculation or a variable already defined.

If *number* is not an integer, returns the decimal display of the number (with 12 digits max).

If *number* is -1, 1 or 0 returns -1, 1 or 0.

Geometrical transformations

Translation tool : 

To create image of objects through a translation by clicking on the origin and end point of the translation's vector.

- Click on the starting point of the translation vector.
- Click on the ending point of the translation vector..
- As long as vector blinks you can click on an object to get it's image created.

So you can get the image of a line, segment, ray, circle, circle arc, polygon or broken line.

Translation through coordinates tool : 

To create image of objects through a translation by clicking on the translation's vector.
Click on a vector.
Then click on objects to get their images through the translation.

Axial symmetry tool : 

To create image of objects through an axial symmetry.

- Click on the symmetry axis (line, segment, ray or vector).
- As long as the symmetry axis blinks you can click on an object to get it's image created.

So you can get the image of a line, segment, ray, circle, circle arc, polygon or broken line.

Central symmetry tool : 

To create image of objects through an central symmetry.

- Click on the symmetry center (point).
- As long as the symmetry center blinks you can click on an object to get it's image created.

So you can get the image of a line, segment, ray, circle, circle arc, polygon or broken line.

Rotation tool : 

To create image of objects through an rotation.

- Click on the rotation center (point).
- A dialog box pops up : choose the angle.
- As long as the rotation center blinks you can click on an object to get it's image created.

So you can get the image of a line, segment, ray, circle, circle arc, polygon or broken line.

Dilation tool : 

To create image of objects through an dilation.

- Click on the dilation center (point).
- A dialog box pops up : choose the ratio of dilation.
- As long as the dilation center blinks you can click on an object to get it's image created.

So you can get the image of a line, segment, ray, circle, circle arc, polygon or broken line.

Direct similitude tool : 

To create image of objects through an dilation.

- Click on the dilation center (point).
- A dialog box pops up : choose the angle and ratio.
- As long as the similitude center blinks you can click on an object to get it's image created.

So you can get the image of a line, segment, ray, circle, circle arc, polygon or broken line.

Inversion tool :

Use tool  at the right edge of the transformations toolbar .

To create image of points only through an inversion.

- Click on the center point of inversion.
- a dialog box pops up. Enter the ratio.

- As long as the center of inversion blinks you can click on a point to get it's image created.

Frames of coordinates

You can define in MathGraph32 as many frames as desired.

To create a frame, use the tool or the menu *Create - Frame*.
If you want to create frame (O,I,J) simply click on O, I and J in that order.

It is possible to create a new figure with a frame.

For this use tool of the top toolbar.

Different sorts of frame are available :

- orthonormal frame.
- orthogonal frame.
- regular oblique frame.
- oblique frame with equal length unities

A frame can be created with grid and dots at integer coordinates.

To modify a frame already created, use icon of the top toolbar (numerical object modification), select the frame in the listand click on button *Modify* or use icon (figure protocol), select the frame in the list and click on modification icon (you cal also double-click on the frame in the list).

When a new figure is created with a frame, radio boxes at the bottom of the dialog box allow the user to give scales to the axis.

A trigonometry scale is available to scale the x-axis with multiples of pi.

The scales on each half-axis can be moved with capture tool .

It is possible to add scales to the axis of a frame after the creation of the frame.

For this use *icon* of the calculations expandable bar (third starting from the button).

Two possibilities :

- A simple scale (with the ability to choose the step and the number of scales on each axis).
- A trigonometry scale with multiple of pi on the x-axis ans a step of scaling on y-axis to be given.

Object reclassification

In certain circumstances, you will need to change the order objects were created in.

Here is an example :

In the back color, you create two free points O and O then the circle center O through B point. The create the line the line joining points O and A.
You activate red color, and choose the full filling style in the fill style palette .

Click the on tool . Then click on the circle to get it filled.
The surface created hides a part of the circle.

Then use tool (available by a click on at the right of the top toolbar) and click on line AB.
The part of the line which was hidden is now visible because the line has been reclassified as far as possible in the list of created objects.

If you use the protocol tool ( ) you will notice that the line is now the last of objects created.

In a similar way it is possible to reclassify a graphical object or a numerical object as far as possible either towards the beginning of the list or towards the start of the list using tool .

The protocol tool also offers the possibility to reclassify objects relatively to other objects in a very precise way.

In the protocol dialog box :

To reclass an objet as far as possible towards the beginning of the object list : Select this object in the list the click on icon .

To reclass an objet as far as possible towards the end of the object list : Select this object in the list the click on icon .

To reclass an object in an single step towards the beginning of the object list : Select this object in the list the click on icon .

To reclass an object in an single step towards the end of the object list : Select this object in the list the click on icon

To reclass an object before or after another object :

First you have to select these two objects in the list (Use Ctrl key + click to select the second object).

Once the two objects selected, if we name object1 the object that is defined first and object2 the object that is defined last :

- A click on icon will reclass object2 before object1.
- A click on icon will reclass object1 after object2.

To be noted :

- Reclassing an object implies reclassing other objects it depends on.
- In some circumstances , the reclassing of an object will be impossible and will not be accepted.

Keyboard Shortcuts

Les raccourcis clavier ne fonctionnent pas avec la version en ligne de MathGraph32.

F1	Launch MathGraph32 help
F2	Launch the dialog box allowing to see and modify numerical objects .
F3	Palette tool .
F4	Name a point or a line .
F5	Move the name of a point or a line .
F6	Mask an object .
F7	Unmask an object .
F8	Execute a macro .
F9	Protocol tool .
Ctrl + Suppr	Graphical object deletion tool .
Ctrl + Ins	Point through coordinates with a frame .
Ctrl + A	Segment .
Ctrl + B	Circle through center and point .
Ctrl + Shift + B	Circle through center and radius .
Ctrl + C	Copy the figure in the clipboard.
Ctrl + D	Line through two points .
Ctrl + E	Real calculation .
Ctrl + Shift + E	Complex calculation .
Ctrl + F	Real function creation .
Ctrl + Shift + F	Complex function creation .
Ctrl + J	Cursor creation .

Ctrl + K	Polygone .
Ctrl + Shift + K	Surface of polygon or circle .
Ctrl + L	Length measure .
Ctrl + Shift + L	Non oriented angle measure .
Ctrl + M	Segment mark creation .
Ctrl + Shift + M	Non orientd angle mark creation .
Ctrl + N	New figure .
Ctrl + O	MathGraph32 file opening .
Ctrl + P	Free point creation .
Ctrl + Shift + P	Linked point creation .
Ctrl + S	Save the figure in a file .
Ctrl + T	Free text display creation .
Ctrl + Shift + T	Linked text display creation .
Ctrl + U	Free value display creation .
Ctrl + Shift + U	Value display linked to point creation .
Ctrl + V	Free LaTeX display creation .
Ctrl + Shift + V	Linked to point LaTeX display creation .
Ctrl + Z	Cancel last action
Ctrl + Y	Redo last action

Numerical objects

Real calculation syntax

Complex calculation syntax

[Matrix calculation syntax](#)

[Numerical objects of calculation type](#)

[Measures](#)

Real calculation syntax

A calculus may use operators + (addition), - (subtraction), * (multiplication), / (division), ^ (power of a number), ² (square), (by priority decreasing priority order).
To be noticed : Operators must be written and spaces are allowed. No implicit multiplication is allowed.
Parenthesis may be used.
Function arguments must lay between braces.

The predefined functions are :

abs	absolute value.
sqrt	square root.
int	integer part.
sin	sine.
cos	cosine.
tan	tangent.
ln	neperian logarithm.
exp	exponential function.
asin	arcsine.
acos	arccosine.
atan	arctangent.
cosh	Hyperbolic cosine.
sinh	Hyperbolic sine.
tanh	Hyperbolic tangent.
asinh	Hyperbolic arcsine.
acosh	Hyperbolic arccosine.
atanh	Hyperbolic arctangent.
rand	returns a pseudo random double laying between 0 and 1 (0 included and 1 excluded).
²	square of a number.
fact(*x*)	returns *x* !. *x* must be integer, positive or null.
left(*x*)	If *x* is a calculation containing a test or an operation, returns the left member. Otherwise the calculation containing this formula will return the same calculation as *x*.
right(*x*)	If *x* is a calculation containing a test or an operation, returns the right. Otherwise the calculation containing this formula will return the same calculation as *x*.
core(x)	Returns the calculation equivalent to x where all calls to functions left, right and if are replaced by their effective argument and where all calls to user defined functions are replaced with the formula of the function.

The predefined functions of two variables are :

max(*x, y*)	returns the greater number of *x* and *y*.
min(*x, y*)	returns the lesser number of *x* and *y*.

pgcd(*n*, *p*)	returns the greater common divisor of *n* and *p*. *n* and *p* must be integer, positive and not null both.
ppcm(*n*, *p*)	returns the lesser common multiple of *n* and *p*. *n* and *p* must be integer positive.
mod(*n*, *p*)	returns the rest of euclidian division of *n* through *b*. *n* and *p* must be integer, positive and *p* not null .
ncr(*n*, *p*)	returns the number of subsets of *p* elements in a set of *n* elements. *n* and *p* must be integer, positive with *p* lower or equal to *n*.
npr(*n*, *p*)	returns the number of permutations of *p* elements in a set of *n* elements. *n* ans *p* must be integer, positive with *p* lower or equal to *n*.
divmaxp(n,p)	Returns the highest positive integer k such as n is divisible by k^p with 1 < p < 256 et 1 < |n| < 1000000

The predefined functions of three variables are :

if(*cond*, *x*, *y*)	returns *x* if *cond* equals 1 and *y* otherwise.

The predefined functions of four variables are :

integrale(*expr*, *var*, *a*, *b*)	returns an appoximated value of the integral of *expr* between *a* and *b, var* is the variable of integration. The integral is calculated through Simpson's method with 400 intervals.
primitive(e*xpr*, *var, start, a*, *b*)	expr is a function of variable var. Returns f(b) - f(a).

The predefined functions of five variables are :

sum(e*xpr*, *var, start, end*, *step*)	returns sum of expression *expr* when all integer values in the range *start-end* with an increment of *step* are given to variable *var*. *expr* may use all values or functions already defined. *start, end* and *step* must have integer values.
prod(e*xpr*, *var, start, end*, *step*)	returns product of expression *expr* when all integer values in the range *start-end* with an increment of *step* are given to variable *var* . *expr* may use all values or functions already defined. *start, end* and *step* must have integer values.

Tests : They return a value which is 1 when result of the test is true, 0 otherwise.

a **>** *b* returns 1 if *a* is superior to b, 0 otherwise .
a **<** *b* returns 1 if *a* is inferior to b, 0 otherwise
a **>=** *b* returns 1 if *a* is superior to b or equal, 0 otherwise .
a **<=** *b* returns 1 if *a* is inferior to b or equal, 0 otherwise .
a **=** b returns 1 if *a* is equal to b, 0 otherwise .
a **<>** *b* returns 1 if *a* is different of b, 0 otherwise .

Boolean operators :

a&b : Returns 1 if a = 1 et b = 1 and 0 otherwise.
a|b : Returns 1 if a = 1 ou b = 1 et 0 otherwise.

Complex calculation syntax

A complex calculus is allowed to use operators + (addition), - (subtraction), * (multiplication), / (division), ^ (power of a number), ² (square), (by priority decreasing priority order).
To be noticed : Operators must be written and spaces are allowed. No implicit multiplication is allowed.
Parenthesis may be used.
Function arguments must lay between braces.

The predefined complex functions are :

abs	Module of a complex. Result is a complex number of imaginary part 0.
conj	Complex conjugate.
angle	Returns a complex number imaginary part of which is zero and real part of which is principal argument of the complex given as argument.
re	Returns a complex number imaginary part of which is zero and real part of which is the real part of the complex given as argument.
im	Returns a complex number imaginary part of which is zero and imaginary part of which is the real part of the complex given as argument.
sqrt	Square root. If the argument is not real or is strictly negative, the result doesn't exist.
int	Integer part. If the argument is not real, the result doesn't exist.
sin	complex sine.
cos	complex cosine.
tan	complex tangent.
ln	complex neperian logarithm.
exp	complex exponential.
asin	arcsine. If the argument is not real, the result doesn't exist.
acos	arcsine. If the argument is not real or doesn't lay between -1 and 1, the result doesn't exist.
atan	arctangent. If the argument is not real, the result doesn't exist.
cosh	Hyperbolic cosine
sinh	Hyperbolic sine
tanh	Hyperbolic tangent
asinh	Hyperbolic arcsine. If the argument is not real, the result doesn't exist.
acosh	Hyperbolic arccosine.If the argument is not real is strictly inferior to 1, the result doesn't exist..
atanh	Hyperbolic arctangent. If the argument is not real or doesn't lay strictly between -1 and +1, the result doesn't exist.
rand	Returns a complex number containing a real value which is a pseudo-random real number lying between 0 and 1 (0 excluded and 1 included).
²	Square.
fact(z)	returns z !. z must be real, integer, positive or null.
left(z)	If *z* is a calculation containing a test or an operation, returns the left member. Otherwise the calculation containing this formula will return the same calculation as *z*.
right(z)	If *x* is a calculation containing a test or an operation, returns the right member. Otherwise the calculation containing this formula will return the same calculation as *x*.

The predefined complex functions of two variables are :

max(*z, z'*)	returns the greater number of *z* and *z'*. *z* and *z'* must be real.
min(*z, z'*)	returns the lowest number of *z* and *z'*. *z* and *z'* must be real.
pgcd(*z, z'*)	returns the GCD of *z* and *z'*. *z* and *z'* must be real, integer positive and not null.
ppcm(*z, z'*)	returns the LCM of *z* and *z'*. *z* and *z'* must be real, integer positive and not null.
mod(*z, z'*)	returns the rest of the integer division of *z* by *z'*. *z* and *z'* must be real and integer positive with b not null.
ncr(*z, z'*)	returns the number of subsets of *z'* elements in a set of *z* elements. *z* and z' must be real, integer positive with *z'* lower or equal to *z*.
npr(*z, z'*)	returns the number of permutations of *z'* elements in a set of *z* elements. *z* and z' must be real, integer positive with *z'* lower or equal to *z*.
divmaxp(n,p)	Returns the highest positive integer k such as n is divisible by k^p with 1 < p < 256 et 1 < |n| < 1000000. n and p must be real.

The predefined functions of three variables are :

if(*cond*, *x*, *y*)	returns *x* if *cond* equals 1 and *y* otherwise.

The predefined functions of four variables are :

integral(*expr*, *var*, *a*, *b*)	returns an appoximated value of the integral of *expr* between *a* and *b, var* is the variable of integration. *a* and *b* must be real. The integral is calculated through Simpson's method with 400 intervals.
primitive(*expr*, *var*, *a*, *b*)	expr is a function of variable var. Returns f(b) - f(a).

The predefined functions of five variables are :

sum(e*xpr*, *var, start, end*, *step*)	returns sum of expression *expr* when all integer values in the range *start-end* with an increment of *step* are given to variable *var*. *expr* may use all values or functions already defined. *start, end* and *step* must have integer values.
prod(e*xpr*, *var, start, end*, *step*)	returns product of expression *expr* when all integer values in the range *start-end* with an increment of *step* are given to variable *var* . *expr* may use all values or functions already defined. *start, end* and *step* must have integer values.

The tests: In complex calculus,the test exist only if the two arguments are real. They return a complex value which is 1 when the result is true, zero otherwise.

a **>** *b* return 1 if a > b and zero otherwise.
a **<** *b* return 1 if a < b and zero otherwise.
a **>=** *b* return 1 if a >= b and zero otherwise.
a **<=** *b* return 1 if a <= b and zero otherwise.
a **=** b return 1 if a = b and zero otherwise.
a **<>** *b* return 1 if a is not equal to b and zero otherwise.

Boolean operators :

a&b : Returns 1 if a = 1 et b = 1 and 0 otherwise.
a|b : Returns 1 if a = 1 ou b = 1 et 0 otherwise.

Matrix calculation syntax

A matrix can be defined through a matrix calculation.

A matrix calculation can use all items allowed for a real calculation along with operations specific to matrices.

Reference to a matrix term :

A matrix calculation can use a reference of a term belonging to a matrix already defined. For instance, if A is a matrix 3x2, A(3,2) will refer to the first term of the third row of matrix A.

Sum and difference of two matrices :

If A and B are two matrices of the same dimension, A+B returns the matrix sum and A - B returns the difference matrix of A and B.

Product of two matrices :

If A is a matrix with *n* rows and *p* columns and B a matrix with p rows and q columns, A*B returns the matrix product of A and B (matrix with *n* rows and *q* columns).

Inverse of a matrix :

if A is an invertible square matrix, A^(-1) returns the inverse matrix of A.

Power of a matrix :

If A is a matrix and *n* a positive integer, A^n returns the *n* power of matrix A. If $n \geq 256$, the result doesn't exist

Determinant of a matrix :

If A is a square matrix, deter(A) refers to the determinant of A.

Transposed of a matrix :

If A is a matrix with *n* rows and *p* columns, transp(A) returns the transposed matrix of A (matrix with *p* rows et *n* columns).

Term to term inverse of a matrix :

If A is a matrix, inv(A) returns the matrix where each term of A is replaced by the inverse. If one of the terms of A is zero, the result doesn't exist.

Term to term product :

If A and B are two matrices of the same dimensions, dotmult(A, B) returns the matrix terms of which are the products of the matching terms of matrix A and B.

Division term by term :

If A and B are two matrices of the same dimensions and if all the terms of B are not null, A/B returns the matrix terms of which are the result of the division of A terms by the matching terms of matrix B.

Sort by row :

If A est a matrix, sortbyrow(A, indrow) returns a matrix columns of which are the columns of A, but sorted according to the increasing order of the elements of row of index indrow.

Sort by column :

If A est a matrix, sortbycol(A, indcol) returns a matrix rows of which are the rows of A, but sorted according to the increasing order of the elements of column of index indcol.

Image through a predefined function :

For instance, if A is a matrix, cos(A) returns the matrix terms of which are the image of A terms through function cosinus.

Image through a user defined function :

If A is a matrix and *f* a user-defined real function, *f*(A) returns the matrix terms of which are the image of A terms through function *f*.

Addition of a constant :

If A is a matrix and *k* a real number, *k* + A ou A + *k* returns the matrix terms of which are the terms of A added with *k*.

Subtraction of a constant :

If A is a matrix and *k* a real number, A - *k* returns the matrix terms of which are the result of the substraction of A terms and *k*.

If *k* is a real number and A a matrix, *k* - A returns the matrix terms of which are the result of the substraction of *k* and A terms.

Multiplication by a constant :

If A is a matrix and *k* a number, *k**A ou A**k* returns the matrix terms of which are the result of the multiplication of A terms and *k*.

Division by a constant :

If A is a matrix and *k* a real number different from zero, A/*k* returns the matrix terms of which are the result of the division of A terms by *k*.

if *k* is a real number and A a matrix, *k*/A returns the matrix terms of which are the result of the division of *k* by A terms. The result doesn't exist of one of A terms is zero.

Approximation by a rational fraction :

If *a* is a real number, frac(a) returns a matrix with one row and two columns. The first term of the returned matrix is the numerator of the approximated rational fraction of this number (with 10^(-12) uncertainty) and the second term is the denominator of the fraction.

If a is a one column matrix, frac(a) returns a matrix with two columns, the first column containing the numerators and the second column the denominators of the approximated rational fractions of each term of matrix *a.*

If a is a one row matrix, frac(a) returns a matrix with two rows, the first row containing the numerators and the second row the denominators of the approximated rational fractions of each term of matrix *a.*

To be noticed :

If the result of a matrix calculation A is a matrix of one row and one column, a reference of A in a matrix calculation will be considered as a real number.

To get the result of the determinant in a real calculation, use the tool provided in the calculation menu (expanding the calculation toolbar). The derminant will be the result of a macro-construction.

Numerical objects of calculation type

These tools are available in the third expandable toolbar starting from the bottom.

The real or complex calculus

A calculus is an object defined through it's name and a formula.

A real calculus can be created through icon.

A complex calculus is created through icon.

A calculus can be modify with icon of the top toolbar.
Name *i* is not available for a calculus or a function. It is reserved for complex calculus.

Real calculus creation :

A real calculus can use all real values previously defined (including measures) but is not allowed to use any complex calculus or complex function.
In the dialog box opening when creating a real calculus :

- A click on the **List of values** button allows to get a dialog box displaying all the real values previously defined.
- A click on the **List of functions** button allows to get a dialog box displaying all the real functions previously defined

Complex calculus creation :

A real calculus can use all real or complex values previously defined (including measures) or any complex function but is not allowed to use owner-defined real functions.

In the dialog box opening when creating a real calculus :

- A click on the **List of values** button allows to get a dialog box displaying all the real or complex values previously defined.
- A click on the **List of functions** button allows to get a dialog box displaying all the complex functions previously defined

The variable

A variable is a real numerical value lying between a *mini* and a *maxi value* ad defined with an step value.
The value returned in a calculus is named *current value*.

A variable is created through icon 7→a .

A variable can modified through icon .
In the dialog box opened when creating a variable you can associate to the variable a special display in the lower right side of the figure as follows :

When clicking on the + button, the variable will be incremented by the step value.
When clicking on the - button, the variable will be decremented by the step value.
Clicking on the = button opens a dialog box allowing the user to modify the variable.

A variable ca be used to generate a linked point locus, a dotted point locus or an object locus.
A variable can also been used to create a macro :

- incrementing or decrementing a variable.
- of automatic trace through variable
- of loop with animation
- of loop with trace
- giving a value to a variable
- modifying a variable.

The real function

It is a real function of a real variable.

A real function is defined through icon f(x)= .
The calculus may be modified afterwards with tool .

For instance, to define a function with formula f(x) = sin(2x) enter *f* as name, *x* as formal variable and enter sin(2*x) for formula but you can also choose *t* as formal variable and enter sin(2*t) for formula.

See also : Real calculus syntax.

The derivative function

MathGraph32 is able to calculate the formal derivative function of a real function.

It is created via icon  .

A dialog box appears displaying all available functions (except function derivative function of which is already defined).

Select the function and specify a name for the derivative.

To be noted : It is the result of a formal calculus but the formula of the derivative is not available to the user.

The complex function

It is a complex function of a real function of a variable.

It is created via icon  of the numerical objects expandable bar (third form the bottom).

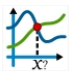
A real function can by modified with icon  of the top toolbar.

For instance, if you want to define a complex function *f* with f(z) = (z + 1/z)^2, you can choose *t* as formal variable and enter formula (*t* + 1/*t*)^2.

See also : Complex calculus syntax.

The approximated equation root

Allows the user to create an approximation root of an equation on an interval [a; b].
This root will be characterized by a name, the name of the unknown, an equation and the values of a, b, and uncertainty.
MathGraph32 creates a function by subtraction of the two members of the equation. Let's call *f* this function.
The solution will exist only if f(a) and f(b) are of different signs.
You are responsible of the existence and unicity of the solution on the given interval.

The approximated root is defined via icon 

To be noticed :

In no way the root is an exact solution..
Asking a very high precision may result in slowing the update of the figure.

The function maximum

This object is created via icon  .

Allows the user to calculate the abscissa of the maximum of a function on an interval [*a*; *b*].
The value is ascertained to be accurate only if the function is increasing from *a* up to the maximum and then decreasing until *b*.

In the dialog box popping up you must :

- Select the function in the list of functions already created..

- Choose the *a* value.
- Choose the b value.
- Choose the uncertainty value.

The function minimum

This object is created via icon .
Allows the user to calculate the abscissa of the minimum of a function on an interval [*a*; *b*].
The value is ascertained to be accurate only if the function is decreasing from *a* up to the maximum and then increasing until *b*.

In the dialog box popping up you must :

- Select the function in the list of functions already created.
- Choose the *a* value.
- Choose the b value.
- Choose the uncertainty value.

The approximated integral value

f referring to a previously defined function, you can create the approximated value of the integral of *f* from *a* to *b* through Simpson method.

This integral value will be characterized by a name, the associated function, *a* and *b* values and value of *n* (number of subdivisions of the interval will be 2*n*)).
Values of *a*, *b* and *n* may be dynamic.

An integral value is created via icon at the right of the calculations expandable bar (third from the bottom).

To be noticed :

Practically, for an interval of reasonable length and a function rather regular, the default value *n* = 20 gives good results.

The test of value existence

A test of value existence is a value allowing to know if a value (calculus, measure or function) previously created exists or not.
The value returned is 1 if the associated value exists and 0 otherwise.
For isntance, let's suppose you have created a figure with a unity length have created a point M intersection of circle de center O with a line.
For some positions of the line, M may not exist. If you want to know if point M exists, , measure first length OM, then create a test of value existence of this length.

A test of existence is created via icon of the calculations expandable bars (third from the bottom).

The test of equivalence of calculations

A test of equivalence of calculations is a value allowing the comparison of two calculations or two functions.
The returned value is 1 if the two calculations are equivalent and 0 otherwise.
This type of object is used to publish online exercises with MathGraph32 JavaScript library.
For instance will be considered equivalent *-2*sqrt(2)/3+1* and *1-2*sqrt(2)/3* and *1-2/3*sqrt(2)* and *1-1/3*2*srqt(2)*
but not *-4*sqrt(2)/6+1*.

To create such an object you must use menu item icon of calculations expandable bar (third from the bottom).
A dialog box pops up.
You must select in the left and right list boxes the calculations to be compared. If you check the **Replacement of values through comparison** checkbox, the calls to variables or calculations will be replaced with their numerical value before the comparison. Additions of 0 will be suppressed, divisions and multiplications by 1 will be suppressed too. If the checkbox **Suppression of multiplications by 1** is unchecked, the multiplications by 1 will be preserved.

For instance, if *a* et *b* are two real calculations containing the values 2 and 3, if *calc1* is a calculation with formula *a/b + sqrt(2)* and *calc2* is a calculation with formula *sqrt(2)+1/3*2*, an equivalence test of these two calculations obtained by checking the checkbox **Replacement of values through comparison** for *calc1* will return value 1 because these two calculations are equivalent.

The test of factorization

A test of factorization is a value allowing to know if the formula of a calculation or a function is factorized by the formula of another calculation or function.
The returned value is 1 if the factorization is possible and else 0.
This type of object is used to publish online exercises with MathGraph32 applet or JavaScript library.
For instance, if the function *f* contains the formula *2*(2*x+3)*(1-x^2)* and if the function *g* contains the formula *3+x*2*, a test of factorization of *f* through *g* will return 1 for value. But if *g* contains the formula *1+x*, it will return 0.

To create such an object you must use icon of the calculations expandable bar (third from the bottom).
A dialog box pops up.
You must select in the left and right list boxes the calculations to be compared. If you check the **Replacement of values through comparison** checkbox, the calls to variables or calculations will be replaced with their numerical value before the comparison. Additions of 0 will be suppressed, divisions and multiplications by 1 will be suppressed too. If the checkbox **Suppression of multiplications by 1** is unchecked, the multiplications by 1 will be preserved

For instance, if *a* et *b* are two real calculations containing the values 2 and 3, if *calc1* is a calculation containing formula *sqrt(3)*5*sqrt(2)*3 and if calc2* is a calculation containing formula *b*sqrt(a)*, a test of factorization created by checking the checkbox **Replacement of values through comparison for** *calc2* will return value 1.

The indexed sum

The indexed sum is the sum of a numerical object, named *value to be summed* obtained when a variable (called the index variable) takes all the integer values between the start and end values of index specified.
The sum is calculated as follows :
Variable index is given all the integer values from mini value to maxi value. Each time, all the elements elements of the figure depending on value to be summed and depending on variable index are re-calculated. The *value to be summed* is added to the sum and so on.

Let us have a simple example :
We have already defined a real function *f* , a variable *N* current value of which is 10.
We wish to calculate the sum of all *f(k)* for *k* in the range from 1 to *N*.
We will use a variable *k* with minimal value of 1, maximal value of 100, step 1 and current value 2. We will notice that mini, maxi values and the step values of the variable are not used by the sum
Then we create a calculus named *v* (the *value to be summed*) that will contain the formula *f(k)*.

Then we will use icon of the calculations expandable bar (third from the bottom).
Enter *S* as name for the sum.
Choose v as value to be summed.
Choose *k* for index variable.
Choose 1 for start index.
Choose *N* for end index.

To be noticed : This sum is dynamic.Modifying N value will modify the sum value.

The indexed multiplication

The indexed multiplication is the sum of a numerical object, named *value to be multiplied* obtained when a variable (called the index variable) takes all the integer values between the start and end values of index specified.
The result is calculated as follows :
Variable index is given all the integer values from mini value to maxi value. Each time, all the elements elements of the figure depending on value to be multiplied and depending on variable index are re-calculated. The *value to be multiplied* is multiplied to the result and so on.

Let us have a simple example :
We have already defined a real function *f* , a variable *N* current value of which is 10.
We wish to calculate the multiplication of all *f(k)* for *k* in the range from 1 to *N*.
We will use a variable *k* with minimal value of 1, maximal value of 100, step 1 and current value 2. We will notice that mini, maxi values and the step values of the variable are not used by the sum
Then we create a calculus named *v* (the *value to be multiplied*) that will contain the formula *f(k)*.

Then we will use icon of the calculations expandable bar (third from the bottom).
Enter *P* as name for the multiplication.
Choose v as value to be summed.
Choose *k* for index variable.
Choose 1 for start index.
Choose *N* for end index.

To be noticed : This multiplication is dynamic.Modifying N value will modify the result.

The real recurrent sequence

This is a sequence defined through a formula type of u(n+1) = f[u(n)] where f refers to a real function already defined.

Such a sequence *u* is defined via icon $u_{n+1} = f(u_n)$.
A dialog box pops up.

You are asked for :

- The name of the sequence.
- Selecting the function in a list.
- Choosing the starting term of the sequence (you can enter a constant value or a calculus).
- Choosing the number of terms of the sequence (you can enter a constant value or a calculus).

The indexes of the sequence start to zero.

When a sequence named *u* has been created, you can refer to term of index *n* of the sequence in a calculus as *u(n)*. This term will exist only if *n* is in the range from 0 to *N* - 1, where *N* is the number of terms of the sequence (assuming the term exist).

See also : The real recurrent sequence graph.

The complex recurrent sequence

This is a sequence defined through a formula type of u(n+1) = f[u(n)] where f refers to a complex function already defined.

Such a sequence *u* is created via icon .

A dialog box pops up.

You are asked for :

- The name of the sequence.
- Selecting the function in a list.
- Choosing the starting term of the sequence (you can enter a constant value or a calculus).
- Choosing the number of terms of the sequence (you can enter a constant value or a calculus).

The indexes of the sequence start to zero.

When a sequence named *u* has been created, you can refer to term of index *n* of the sequence in a calculus as *u(n)*. This term will exist only if *n* is in the range from 0 to *N* - 1, where *N* is the number of terms of the sequence (assuming the term exist).

See also : The complex recurrent sequence graph.

The real function of two variables

A real function of two variables is created through icon of the calculations expandable bar (third from the bottom).

Such a function may be modified via icon .

For instance,you can define a real function of two variables by $f(x, y) = 2x^2y + 3x + y + 1$. Enter *f* for the name, *x* and *y* for the variables names and 2x^2**y*+ 3**x* + *y* + 1 for formula.

See also : Real calculus syntax.

The real function of three variables

A real function of three variables is created through icon of the calculations expandable bar (third from the bottom).

Such a function may be modified through icon .

For instance,you ca define a real function of three variables by $f(x, y, z) = 2x^2yz + 3x + z + 1$. Enter *f* for the name, *x, y* and z for the variables names and 2**x*^2**y*+ 3**x* + *y* + 1 for formula.

See also : Real calculus syntax.

The complex function of two variables

A complex function of two variables is created via icon of the calculations expandable bar (third from the bottom).

Such a function may be modified through icon .
For instance,you can define a real function of two variables by choosing *f* for name, *z* and *z'* for the variables names and 2*Re(*z*^2)**z'*+ 3**z* + *z'* + 1 fro formula.

See also : [Complex calculus syntax](#).

The complex function of three variables

A complex function of three variables is created icon of the calculations expandable bar (third from the bottom).

Such a function may be modified through icon .

For instance,you can define a real function of two variables by f(*u*, *v*, w*)* = 2*u*²*v* + 3Im(*w*). Enter *f* for the name, *u* and *v and* w for the variables names and 2*u^2*v+3*Im(w) for formula.

See also : [Complex calculus syntax](#).

The modulus of complex

The formula of a real calculus is not allowed to refer to a complex number. Thus if you want to use the modulus of a complex number you have first to create a modulus of complex

You can then refer to this calculus (which is a real calculus) in another real calculus.
Here is an example :

Let's assume you have measured the affix of a point M in a frame (O;I,J) and named this affix *aff*. You want to create a circle radius of which will be the half of the modulus of this affix.

You have to use icon of the calculations expandable bar (third from the bottom).
A dialog box pops up.
Select the affix *aff*.
In the edit field **Name of the calculus**, enter *a*.
Click **OK**.
When you will create the circle you will enter in the radius fields the formula *a/2*.

The principal argument of complex

The principal argument of complex is a real value lying between -π (excluded) and +π (included) if the angle unit of the figure is the radian and between -180 (excluded) and 180 (included) if the angle unit of the figure is the degree.
The formula of a real calculus is not allowed to refer to a complex number. Thus if you want to use the principal argument of a complex number you have first to create a principal argument of complex.
You can then refer to this calculus (which is a real calculus) in another real calculus.
Here is an example :
A complex number named *z* has been created (not null) and you wish to create the image of a point through a rotation angle of which is the double of double of *z* principal argument.

You must use of the calculations expandable bar (third from the bottom).
A dialog box pops up.
Select complex *z* in the list.
In the field **Name of the calculus**, type *a*.
Press **OK**.
When you will create the 'image of a point point through a rotation you will enter 2**a in the field*

angle of rotation.

The real part of complex

The formula of a real calculus is not allowed to refer to a complex number. Thus if you want to use the real part of a complex number you have first to create a modulus of complex

You can then refer to this calculus (which is a real calculus) in another real calculus.

Here is an example :

Let's a complex number named *z* has been created.
You wish to create a point coordinates of which will be the double of *z* real part.

You must use  of the calculations expandable bar (third from the bottom)..
In the field **Name of the calculus**, type *a*.
Press **OK**.
When you will create a point by it's coordinates you will specify 2**a in the fields x-coordinate* and y-coordinate.

The imaginary part of complex

The formula of a real calculus is not allowed to refer to a complex number. Thus if you want to use the imaginary of a complex number you have first to create a modulus of complex

You can then refer to this calculus (which is a real calculus) in another real calculus.
Here is an example :

Let's a complex number named *z* has been created.
You wish to create a point coordinates of which will be the double of *z* real part.

You must use  of the calculations expandable bar (third from the bottom).
In the field **Name of the calculus**, type *a*.
Press **OK**.
When you will create a point by it's coordinates you will specify 2**a in the fields x-coordinate* and y-coordinate.

The partial derivative function

MathGraph32 is able to calculate the partial formal derivative function of a real function of two or three variables.

It is created via icon  of the calculations expandable bar (third from the bottom).

A dialog box appears displaying all available functions.

Select the function, specify a name for the derivative and choose the variable of derivation.

To be noted : It is the result of a formal calculus but the formula of the derivative is not available to the user.

The test of function variable dependence

This object purpose is mainly for displaying inline figures for the teaching of calculation.
It's goal is to know if a function of one, two or three variables depend on one of it's variables.
The test is made without simplifying the calculation of the function.
The value returned is 1 if the formula depends on the chosen variable and else 0.

For example, a function defined by f(x,y) = x^2 +2*y - x^2 will be considered dependant of *x* and *y*.
But a function defined by f(x,y,z)=x^2+y will be considered as non dependent on *z* and dependent on *x* and *y*.

To define such an object, use icon of the calculations expandable bar (third from the bottom)..

The real matrix

This object is characterized by the number of rows, the number of columns and the real terms of each cell.

A real matrix can be defined :

By giving a formula to each term of the matrix.

Use icon at the right of the calculations toolbar and choose **Matrix**.

By providing a formula of a two variable function of variable (i, j) where *i* is the row index and *j* the column index (starting from 1)

Use icon at the right of the calculations toolbar and choose **Matrix by formula.**

For instance, the formula if(i=j, 1, 0) will provide an identity matrix.

By a matrix calculation :

Use icon at the right of the calculations toolbar and choose **Matrix calculation.**

If, for instance,A is an invertible matrix of dimensions 3x3 and B is a 3x2 matrix, the matrix calculation defined by the formula A^(-1)*B will return a 3x2 matrix (A^(-1) returning the inverse of matrix A).

But a matrix calculation defined by the formula B*A won't exist (invalid matrix product).

By defining a matrix with random integer values :

Use icon at the right of the calculations toolbar and choose **Matrix with random integer values.**

Such a matrix is defined through the number of rows, number of lines, a minimal and a maximum values (integer).

The minimum value must be inferior to the maximum value.

When such a matrix is updated, all the cell of the matrix will be given distinct integer values ranging form *min* value to *max* value. If the difference between max and min value is not big enough, there will be repetitions (the lines being filled first)

By including data pasted from a worksheet :

Select the desired numerical data in the worksheet then use icon at the right of the calculations toolbar and choose **Matrix by worksheet data.**

By creating a two columns matrix giving the primary decomposition superior or equal to 2.

Use icon  at the right of the calculations toolbar and choose **Matrix of primary decomposition.**

The first column of the matrix will contain the primary factors and the second one will provide the mathing exponents.

The number to be decomposed must be in the range 2 -> 10 000 000.

By creating a two columns matrix columns of which are coordinates pf points of the figure.

Use icon  at the right of the calculations toolbar and choose **Matrix of coordinates.**
To be noted : You ca also create a set of points coordinates of which are contained in a two columns matrix. For this use icon  at the bottom of the left toolbar ans choose Set of points by 2 columns matrix. This point creation is dynamic : The matrix changes when the points move.

Measures

Les mesures se font via les icônes de la cinquième barre d'icônes à partir du bas.

Mesure de longueur

Mesure d'une abscisse

Mesure d'angle non orienté

Mesure d'angle orienté

Mesure d'abscisse dans un repère

Mesure d'ordonnée dans un repère

Mesure de coefficient directeur

Mesure d'affixe dans un repère

Mesure de longueur de ligne

Mesure d'aire de polygone

Mesure de produit scalaire

Length measure

For this tool to be selected, the figure must contain a unity length.

To measure the distance between two points, click on icon  then click on the two points. The points clicked must be named. If they are not named, they will be named by the tool.

Abscissa measure

You can measure the abscissa of a point on a line relative to two points via icon.

This abscissa will not exist if the points are nor aligned.

In order to measure abscissa of M relative to (A;B) :

- Click on A (point of abscissa 0).
- Click on B (point of abscissa 1).
- Click on M to measure abscissa of M (M must be aligned with avec A and B).

The three points clicked must be named.
If they aren't they will be named by the tool.

MathGraph32 will ask you a name for this measure.

Non oriented angle measure

In order to measure non-oriented angle type of AÔB, click on icon .

Click on A, O then B.

If the unity angle of the figure is degree, this measure will lie in the range 0 to 180.
If the unity angle of the figure is radian, this measure will lie in the range 0 to π.

The three points clicked must be named.
If they aren't they will be named by the tool.

In future calculations, this measure will be represented by AOB.

Oriented angle measure

In order to measure the oriented-angle from OA vector towards OB vector, click on icon.

Click on A, O then B.
If the unity angle of the figure is degree, this measure will lie in the range -180 (excluded) to 180.
If the unity angle of the figure is radian, this measure will lie in the range - π (excluded) to π.

The three points clicked must be named.
If they aren't they will be named by the tool.

MathGraph32 will ask you a name for this measure.

Abscissa measure in a frame

In order to measure abscissa of a point relative to a frame, click on icon.

Simply click on the point the abscissa of which has to be measured.

The point clicked must be named.
If it isn't it will be named by the tool.

MathGraph32 will ask you a name for this measure et the choice of the frame in a dialog box.

Ordinate measure in frame

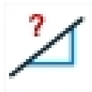
In order to measure abscissa of a point relative to a frame, click on  icon.

Simply click on the point the ordinate of which has to be measured.

The point clicked must be named.
If it isn't it will be named by the tool.

MathGraph32 will ask you a name for this measure et the choice of the frame in a dialog box.

Line slope measure

In order to measure the slope of a line in a frame, click on  icon.

At least one frame must have been defined.
Click first on the line.
A dialog box pops up.
Specify the name chosen for the measure and click on the frame the measure is relative to.

Affix measure in a frame

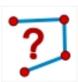
Click on  icon to get the complex affix of a point measured.

The point clicked must be named.
If it isn't it will be named by the tool.

MathGraph32 will ask you a name for this measure et the choice of the frame in a dialog box.

Line length measure

If you wish a line length (polygon or broken line) to be measured, your figure must have an unity length.

Use  icon .

Click on the line (polygon or broken line) measure of which has to be created.
A dialog box pops up asking you for a name for the measure.

The vertexes of the line must be named. If they aren't they will be named by the tool.

Polygon area measure

If you wish a polygon area to be measured, your figure must own an unity length.

Use  icon.

Click on the polygon area of which has to be created.
A dialog box pops up asking you for a name for the measure.

The vertexes of the line must be named. If they aren't they will be named by the tool.

Scalar multiplication measure

If you want to measure a scalar multiplication, your figure must have a unity length.

Click on icon  of scalar multiplication measure.
Click then on the two vectors scalar multiplication is to be measured.

To be noticed : The four vectors vertexes have to be named. Otherwise a name will be attributed to them.

Curves and graphs

[Courbe de fonction](#)

[Graphe de suite récurrente](#)

[Tangente à une courbe de fonction](#)

Function curve

The function curve is not an object.
MathGraph32 draws a function curve as a point locus.
Your figure must own a frame.

You must first create a function must be created through icon  .
If your figure contains one or several frames, at the bottom of the dialog box the checkbox **Draw curve** is selected.
If you keep the checkbox **Draw curve** selected, the function curve will be added to your figure (defined on R).
If you wish to draw this function curve only on an interval (or don't want the curve to be created) uncheck **Draw curve** checkbox.

If you have chosen not to check **Draw curve** checkbox, you can draw the curve afterwards using icon .
A dialog box pops up.
Select the function and the frame the curve will be traced in.
Select the type of interval on which the curve is to be calculated.

The function curve is created as follows :

MathGraph32 creates a point linked to a line, ray or segment according to the type of interval chosen.
Then are created :

- The measure of the abscissa of the linked point in the frame.
- A calculus containing this abscissa. We will refer to it as *x*.
- Another calculus containing the image of the abscissa through the function. We will refer to it as *y*.
- A point of coordinates (*x*; *y*) in the frame.
- A point locus of this last point through the positions of the linked point.

In the dialog box you can change the number of points used to draw the point locus (curve).

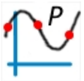
You can also ask for :

- the linked point to be hidden or not.
- the point generating the curve to be hidden or not.
- the automatic management of discontinuity

Three more tools allow function curves creation.

Tool  allows the creation of a function curve with one or two brackets at the edge of the curve.

Tool  allow the creation of a curve going through *n* given points and the tangent slope at these points. (*n* in the range 2 to 10).

Tool  allows the creation of a polynomial function curve (of degree less or equal *n*) going through *n* given points (n valant de 2 à 8).

To be noticed : A function curve is a point locus and therefore you can create a point linked to it.

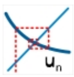
Recurrent sequence graph

MathGraph32 is able to graph recurrent sequences type of u(n+1) = *f*[u(n)], where *f* represents a real or complex function.

Real recurrent sequence graph.

It is the type of graph often called web-graph.

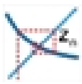
Before creating the graph you must create the real recurrent sequence with tool  in the calculations expandable bar.
.

The graph is the created via *icon*  in the graph expandable bar
A dialog box pops up.
You select :

- the sequence associated.
- the frame in which the graph occurs.
- if the points of the graph must be joined to the abscissa axis or not.

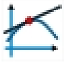
Complex recurrent sequence graph.

Before creating the graph you must create the complex recurrent sequence with tool  in the calculations expandable bar.
Each complex term of the sequence is represented through a point affix of which is the term.
This point is drawn in the active point style.
Each point is linked to the next one through a segment drawn in the active color and line style.

The graph is created *via icon*  in the graph expandable bar.
A dialog box pops up for the choice of the sequence and frame.

Function curve tangent

This is not an object but the result of an internal construction allowing you creating a tangent to a unction curve through the abscissa of the point.

Click on the icon  in the lines expandable bar.
A dialog box pops up.
Select the function and the frame.
Enter the abscissa of the point. It may a reference to an already existing value or a calculus. In the second case a new calculus will be created name of which will start with *abs*.
Enter a name for the calculus that will contain the slope of the line created.

To be noticed : For the tangent to be created, MathGraph32 computes the formal derivative of the function. Sometimes, it may happen that the tangent will not exist while it should. For example, if f(x)=x*sqrt(x), the tangent at point of abscissa 0 will not exist while it should.

Macros

MathGraph32 macros are buttons that fire an action on the figure when pressed.
This kind of button starts with ⇒ character.

Some macros manage a list of objects. It is possible to remove or add objects to such macros through menus *Macros - Add objects to macro* or *Macro - Remove objects from macro*.

Once created, a macro ca be modified or reclassified through menu *Macros - Modification or reclassification*.

Different kind of macros

Adding objects to a macro

Removing objects from a macro

Different types of macros

Masking objects macro
Unmasking objects macro
Linked point animation
Linked point animation with trace
Assigning value to variable macro
Variable modification macro
Variable increment macro
Variable decrement macro
Auto trace through linked point macro
Auto trace through variable macro
Moving linked point macro
Object blinking macro
Updating figure macro
Pause macro
Playing sound file macro
Loop with animation macro
Loop with trace macro
Sequence of macros execution macro
Mode trace activation macro

Mode trace deactivation macro

Masking objects macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).
It's purpose is to move automatically a linked point of the figure.
It's purpose is to make hidden masked objects.

Clicking on the macro button will make these objects become invisible. First activate tool (macro execution on the top toolbar).

Two possibilities :

- If radio button **Click for next object** is selected, you must click as many times as the macro manages objects.
- If radio button **Click for next object** is not selected, a single click all the objects appear.

Unmasking objects macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).
It's purpose is to move automatically a linked point of the figure.
It's purpose is to make visible hidden objects.

Clicking on the macro button will make these objects become visible. First activate tool (macro execution on the top toolbar).

Two possibilities :

- If radio button **Click for next object** is selected, you must click as many times as the macro manages objects.
- If radio button **Click for next object** is not selected, a single click all the objects appear.

Linked point animation macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).
It's purpose is to move automatically a linked point of the figure.

The animation starts by clicking on the button of the macro. First activate tool (macro execution on the top toolbar).

Click on the figure or select another tool to stop the animation..

Variable animation macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).
It's purpose is to make a variable vary and update the objects of the figure depending on this variable.

The animation starts by clicking on the button of the macro. First activate tool (macro execution on the top toolbar).

Click on the figure or select another tool to stop the animation..

Linked point animation macro with trace

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

.

It's purpose is to move automatically a linked point of the figure while a list of objects leave a trace of their position.

The animation starts by clicking on the button of the macro.

Click on the figure to stop animation. First activate tool (macro execution on the top toolbar).

See also :

Adding objects to a macro

Remove objects from a macro

Variable animation macro with trace

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

.

It's purpose is to make a variable vary and update the objects of the figure depending on this variable. while a list of objects leave a trace of their position.

The animation starts by clicking on the button of the macro.

Click on the figure to stop animation. First activate tool (macro execution on the top toolbar).

See also :

Adding objects to a macro

Remove objects from a macro

Assigning value to variable macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

It's purpose is to give a value to a variable and recalculate the figure.

Variable modification macro

This macro is created via icon of the displays expandable toolbar (fourth from the

bottom).
It's purpose is to modify all the variable attributes and recalculate the figure.

You have to specify the new values of :
- mini value of the variable
- maxi value of the variable
- step value of the variable
- current value of the variable

Variable increment macro

This macro is created via *icon* of the displays expandable toolbar (fourth from the bottom).

When activated, such a macro adds the step of the variable to the current value (if the new value doesn't exceed the maxi value) and puts the result in the current value.

Variable decrement macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

When activated, such a macro subtracts the step of the variable to the current value (if the new value exceeds the mani value) and puts the result in the current value.

Auto trace through linked point macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

When activated, this macro leaves on the figure the trace of a list of objects when a linked point is moving on the object it is linked to.

To be noticed : The trace is not permanent and will disappear with tool .

See also :
- Adding objects to a macro
- Removing objects from a macro

Auto trace through variable macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

When activated, this macro leaves on the figure the trace of a list of objects when a variable takes values from it's mini to maxi values.

To be noticed : The trace is not permanent and will disappear with tool .

See also :
- Adding objects to a macro

- [Removing objects from a macro]

Moving linked point macro

This macro is created via con of the displays expandable toolbar (fourth from the bottom).

When activated, this macro moves the linked point to the position of another point linked to the same object.

Object blinking macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

When activated this macro starts the blinking of the chosen objects.
The blinking duration can be of determined length or stopped by a mouse click.

See also :

- Adding objects to a macro
- Removing objects from a macro

Updating figure macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

When activated such a macro erases the display and redraws the figure. It is useful to erase traces produced by another macro.

To be noticed :

- Only one macro of this type can be created in a figure.
- If the figure uses calls of *rand()* function, activating this macro will cause a new value to be affected for all the calls of random function, thus allowing a random experience to be updated.

Pause macro

This macro is created via *icon* of the displays expandable toolbar (fourth from the bottom).

The only purpose of this macro is to be inserted in a macro executing a sequence of macros and execute a pause before executing the following macros of the sequence.

Playing sound file macro

This macro is created via icon of the displays expandable toolbar (fourth from the bottom).

It's purpose is to play the sound file when activated.

The sound file must absolutely lay in the same directory as the figure file or in a sub-directory.

Click first at the spot the button is to be displayed to.
A dialog box pops up.
Enter the characteristics of the macro and the relative path to the figure (the path is relative to the place where the figure will be).

Click **OK** to validate.

Loop with animation macro

This macro is created via icon  of the displays expandable toolbar (fourth from the bottom).

This macro purpose is to recalculate and display the figure for all values of a variable, from mini to maxi value, with step increment.
This kind of macro is very useful to simulation random experiences.
You may specify a macro to be executed before all the loops to run. For example, this macro will initialize counters.
You may also define a macro to be executed at the end of each loop. For example, this macro will update counters.

When a loop with animation macro is executed :

The variable is given it's mini value.
The macro to be executed before the loops is executed.
The figure is recalculated and updated.
The macro to be executed at the end of each loop is executed.
The step of the variable is added to the current value.
And so on while current value of the variable is inferior to it's max value.

To be noticed : The macro to be executed before the loops and at the end of each loop will often be macros executing a sequence of macros.

Loop with trace macro

This macro is created via icon  of the displays expandable toolbar (fourth from the bottom).

This macro purpose is to recalculate and display the figure for all values of a variable, from mini to maxi value, with step increment. At the end of each loop, a list of objects will draw a trace on the figure.
This kind of macro is very useful to simulation random experiences.
You may specify a macro to be executed before all the loops to run. For example, this macro will initialize counters.
You may also define a macro to be executed at the end of each loop. For example, this macro will update counters.

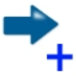
When a loop with animation macro is executed :

The variable is given it's mini value.
The macro to be executed before the loops is executed.
The figure is recalculated and updated, the objects of the chosen list leaving a trace on the figure.
The macro to be executed at the end of each loop is executed.
The step of the variable is added to the current value.

And so on while current value of the variable is inferior to it's max value.

To be noticed : The macro to be executed before the loops and at the end of each loop will often be macros executing a sequence of macros.

Sequence of macros execution macro

This macro is created via icon  of the displays expandable toolbar (fourth from the bottom).

This macro purpose is to execute a sequence of other macros previously defined.

In the dialog box popping up while creation, two frames are available.
The left frame presents all the exiting macro titles.
In the left list, select a macro.
Click on the button **Insert to end position** to insert this macro title at the end of the right list.
Click on the button **Insert to start position** to insert this macro title at the beginning of the right list.

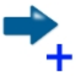
To get a macro title inserted between two others in the right list :

- In the right list, click on the macro above which the new macro is to be inserted.
- In the left list, select the macro to be inserted.
- Click on the **Insert button**.

To withdraw macros in the right list, select them in the right list the click on the **Withdraw** button.

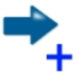
To reinitialize the right list, click on the button **Withdraw all**.

Trace mode activation macro

This macro is created via  of the displays expandable toolbar (fourth from the bottom).

This macro purpose is to set Trace Mode on.
When Traces mode is on, traces you get via menu Move - Objects trace or via a macro generating a trace remain permanently as long as mode Trace is on and the window containing the figure is not resized.
This kind of macro may be useful in a macro executing a sequence of macros.

Trace mode deactivation macro

This macro is created via  of the displays expandable toolbar (fourth from the bottom).

This macro purpose is to set Trace mode off.
When Traces mode is on, traces you get via menu Move - Objects trace or via a macro generating a trace remain permanently as long as mode Trace is on and the window containing the figure is not resized.
This kind of macro may be useful in a macro executing a sequence of macros.

Adding objects to a macro

Some kinds of macros maintain a list of objects : Unmasking macros, Masking macros, Macro of animation with trace, of trace auto or macros of loops.

To add new objects after the creation of the macro, use tool at the bottom of the left toolbar and choose *Add objects to macro*.

A dialog box pops up displaying available macros.

Click on the macro in the list and validate.

The objects of this macro are now blinking.

Click on other objects to add the to the macro.

Click on the red button STOP to finish.

Remove objects from a macro

Some kinds of macros maintain a list of objects : Unmasking macros, Masking macros, Macro of animation with trace, of trace auto or macros of loops.

To remove objects from the macro after the creation of the macro, use tool at the bottom of the left toolbar and choose *Remove objects from macro*.

A dialog box pops up displaying available macros.

Click on the macro in the list and validate.

The objects of this macro are now blinking.

Click on the objects you want to be remove from the macro.

Click on the red button STOP to finish.

Keyboard shortcuts

Les raccourcis clavier ne fonctionnent pas avec la version en ligne de MathGraph32.

F1	Launch of MathGraph32 help in browser.
F2	Reactivate the last tool.
F3	Open the dialog box showing numerical objects and allowing their modification .
F4	Activation of tool for naming point or line .
F5	Activation of tool for moving the name of a point or a line .
F6	Activation of tool for masking an object (the object still exists but is not visible).
F7	Activation of tool for unmasking a hidden object (curtain).
F8	Activation of tool for execution a macro .
F9	Open the protocol dialog box displaying all the objects of the figure with information on each and modifying possibilities.
F10	Displays again the last indication for active tool (top right corner)

Ctrl + Suppr	Graphic tool deletion .
Ctrl + Ins	Creation of a point via it's coordinate .
Ctrl + A	Segment tool .
Ctrl + B	Circle trough two points creation tool .
Ctrl + Shift + B	Circle through center and radius creation tool .
Ctrl + C	Copy the figure in the clipboard.
Ctrl + D	Line through two points creation .
Ctrl + E	Real calculation creation .
Ctrl + Shift + E	Complex calculation creation .
Ctrl + F	Real function creation (one variable) .
Ctrl + Shift + F	Complex function creation (one variable) .
Ctrl + J	Cursor creation .
Ctrl + K	Polygon creation .
Ctrl + Shift + K	Surface creation (delimited by polygon or circle).
Ctrl + L	Length measure .
Ctrl + Shift + L	Non oriented angle measure .
Ctrl + M	Segment mark creation .
Ctrl + Shift + M	Non oriented angle mark creation .
Ctrl + N	New figure creation .
Ctrl + O	Open new figure (mgj file).
Ctrl + P	Free point creation .
Ctrl + Shift + P	Linked point creation .
Ctrl + S	Save actual figure .

Ctrl + T	Free text display creation .
Ctrl + Shift + T	Linked to point text display creation .
Ctrl + U	Free display value creation .
Ctrl + Shift + U	Linked to point display value creation .
Ctrl + V	Free LaTeX display creation .
Ctrl + Shift + V	Linked to point LaTeX display creation .
Ctrl + Z	Cancel last action on the figure
Ctrl + Y	Redo last action on the figure

Constructions

Presentation of constructions

Constructions are one of the most powerful features of MathGraph32.
Constructions are some sort of figures which can be incorporated in other figures.
A construction utilizes objects belonging to the figure the construction is to be implemented in called *sources objects*.
It uses internal objects (called *intermediary objects*) to create new objects (called *final objects*).
The *final objects* will be available to the user who will be able to use them for creating other objects.
Sources and final objects cab be numerical or graphical objects.
Frames are taken as a numerical object.

Numerical source object are allowed to be :

- a real calculus
- a complex calculus
- a real function
- a complex function
- a variable
- a real recurrent sequence
- a complex recurrent sequence.

Graphical sources objects are allowed to be :

- a free point
- a linked point
- a line

- a ray
- a segment
- a circle
- a circle arc
- a polygon
- a broken line
- a vector

To be valid, a construction must obey the following rule :

Final objects are only allowed to be constructed with the sources objects selected and every source object must be used at least one time for the creation of a final object.

Just a single exception to this rule : if final objects are constructed using sources objects and a variable which is not a source object, the construction will be accepted and the variable will be part of intermediary objects.

Constructions are saved in files with extension *mgf*.

Icons to create and manage constructions are available when clicking on icon on the right of the top toolbar.

Icon purpose is to manage constructions. Clicking on this icon makes a choice dialog box popping up.

Icon purpose is create a construction. Clicking on this icon makes a choice dialog box popping up.

You can load a construction in the figure from a file with icon and choosing *Incorporate construction from file*.

You implement a construction of the figure via icon et choosing *Implement construction of the figure*.

You save a construction in a file with icon and choosing *Save a construction of the figure*.

When a construction has been implemented in a figure, it is possible to get all intermediary objects becoming normal objects using icon and choosing *Merge constructions of the figure*.

How to create a construction.

For a construction to be defined, sources objects must be chosen first.
It is important to understand that, when a construction is implemented numerical objects (if any) will have to be chosen first.

You choose sources objects via icon and *Numerical sources objects choice* or *Graphical sources objects choice.* Don't forget to click on the red STOP button when all the objects have been clicked on.

When sources objects have been chosen, you must choose final objects. It is only possible to select final objects constructed only through sources objects.

The choice of final objects is made via icon  and *Numerical final objects choice* or *Numerical final objects choice*.

Final and sources objects choice can be canceled via *icon*  *and Reset current construction*.

To finalize the construction once sources and final objects chosen, use icon  then choose *Finish current construction*.

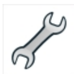
In the commentary of the construction you can enter explanations and indications that will be available to the user when implementing the macro (refer to examples).

Let us say that your construction has three sources objects : a real calculation (representing an angle value), a point and a circle.
You can add the following lines to the commentary of the macro :

#1:the angle value
#2:a point
#3:a circle

How to implement a construction.

You can implement a construction only if this construction has been created in the figure or incorporated in the figure from a file.

To implement a construction, use icon  then choose *Implement a construction of the figure*.

If the construction contains both graphical and non graphical sources objects, non graphical objects are to be chosen first.

To be noticed : If a construction uses a single non graphical object which is a frame and if only a frame is present in the figure, this frame will be automatically chosen as first source object and the non graphical sources objects dialog box will not appear.

Construction. Example 1.

We will here explain how to create a construction and how to use it.

We wish to create a construction. It's purpose will be, given two points A and B and a numerical value *ang*, to create a triangle AMB right-angled in M with angle BAM of measure *ang*.

Create a new figure with icon . Choose a figure without frame and with an unity length.

If necessary, use menu icon  to set degree as angle unity for the current figure.

Use icon to create a calculation named *ang* with formula *30*.

Create two free points with icon et name them *A* and *B*.

Use icon to create midpoint of segment [*AB*].

With icon create a circle of center the midpoint and going through point *A*.

Let's now create the image of B through rotation center A and angle *ang*.

Click on tool . Then click on A (rotation center). *A* starts blinking. Click on point *B* to get it's image created.

Use tool to create the ray with origin A and going through the last point created (image point).

Click on intersection tool , then click at the intersection of the last ray and the circle. We will call the new point created C.

Use tool to create segments [*AB*], [*BC*] and [*CA*] and tool to mark the right angle in C (click on A, C and B in this order).
Our figure is now ready to create our construction.

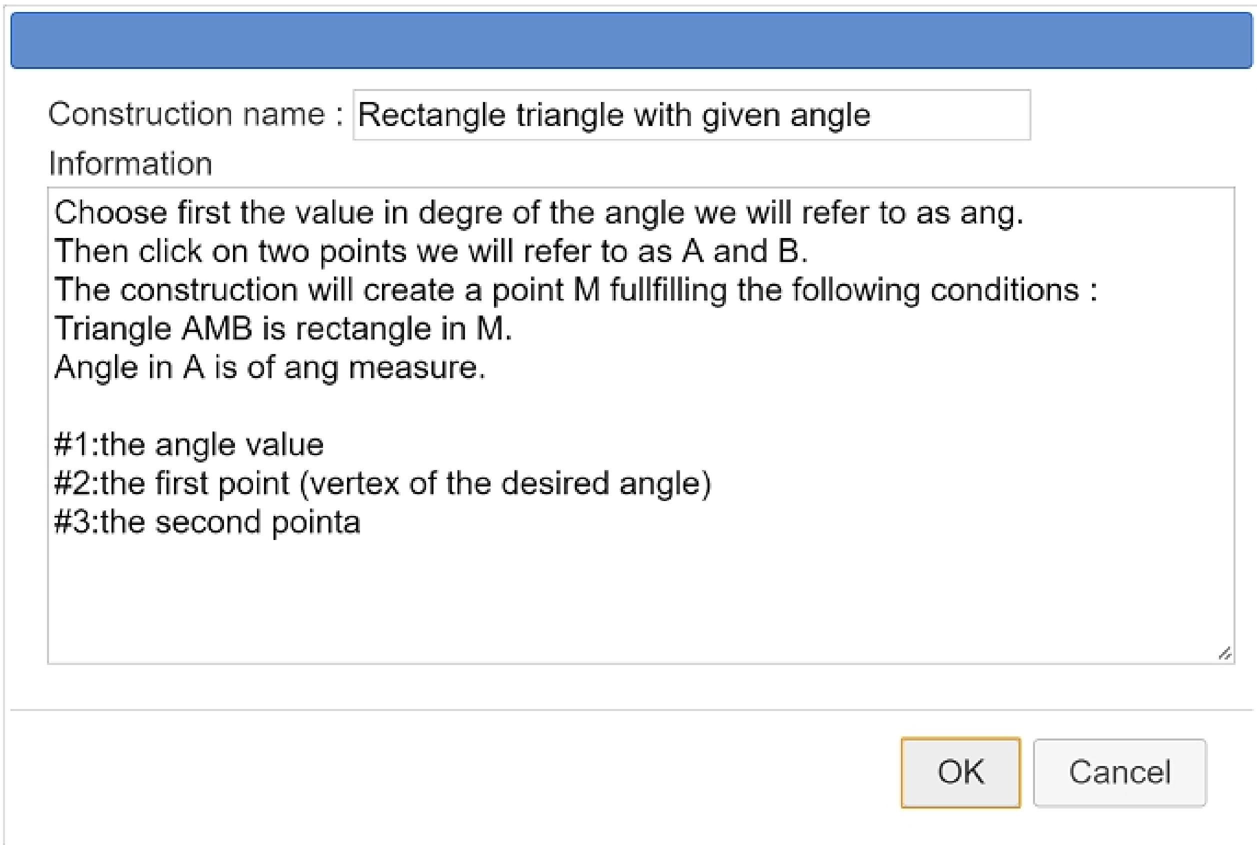
Use icon then choose *Graphical sources objects choice*. Click on *A* and *B*, then click on the red STOP button on the right bottom side of the window.

Use icon then choose *Numerical sources objects choice*.
A dialog box pops up.
The left list contains all numerical objects which are potential object sources.
Click on *ang* then click on button **Insert** (you can also double click on *ang*) then validate.
Now we have to choose the final objects of our figure. In this example, all the final objects are numerical.
Let's point out that it is only possible to choose as final objects objects exclusively created through sources objects.

Use icon then choose *Graphical final objects choice*.
Click on point *C*, on the three segments and on the angle mark then click on the red STOP button.

To finalize the construction, la construction, use icon then choose *Finish current construction*.

Fill in the dialog box as underneath :



Let us explain the information entered here.
The first five lines are general explanations for the future user. These explanations will pop up when pressing key **F7** while implementing the construction.
The last three lines will be displayed in the indication line(at the bottom of MathGraph32 window) when the user will be asked to click on sources objects.
Each line starts with character # followed with the index of the source object.

To be noticed : When implementing a macro, numerical sources objects must always be specified first.

If you don't delete this construction, it will be saved with the figure when you save the figure to a file. But it is better to save the construction in a separate file with extension *mgc*.

For this use icon  and choose *Save a construction of the figure.*

Let us now implement this construction in another figure.

Use menu icon  and choose a figure with or without a frame.

Use icon  and choose *Incorporate construction from file* to incorporate the construction you just saved in a file.

Use icon 

(toolbar of calculations and cursors) to create a new variable named *a*, with mini value of - *90*, maxi value of *90*, step value of *10* , current value of *30* and select checkbox **Associated dialog**.

Create two free points with tool and get them named C and D.

Use now icon and choose *Implement a construction of the figure*.
A dialog box pops up with a list of the constructions of the figure. Click on *Tangents* then click on button **Implement**.
Another dialog box pops up for the choice of numerical objects.
In the right list, click on variable *a* to associate *a* to the first source object (only source object in this example).
Then in the indication line we see that the construction is waiting for us to click on two points.
Click on point C and D.
You see now new objects appear : the final objects of the construction.
Click on buttons + and - in the little dialog box associated with the variable to change the angle in C.

In more complicated constructions you can also have numerical final object sources.

Using icon (figure protocol) and checking the checkbox **Intermediary objects**, you will be able to see all the objects created, including intermediary objects.

Intermediary objects cannot be used by other objects, except using icon and choosing *Merge constructions implemented in figure*.

To be noticed :
In current use, free points are only sources objects.
But you may want a free point to become a final object of a construction.
To do so :
Click on the free point while choosing graphical sources objects.
Click on the same point when clicking on final graphical objects.
A confirmation will be asked and your point will become a final object.

Construction. Example 2.

Second example of a construction creation.

We want to create a construction with the purpose of creating the two tangents to a circle through a given point (exterior to the circle).

Use icon to create a new figure without unity length.

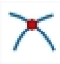
Use tool to create three free points we will refer to as O, A and M.

Use tool to create a circle with center O and through point A.
Our construction must use the circle center but must not use point O. Indeed final objects must only be constructed through the circle and point M. They are not allowed to use point O which was used to create the circle.

Use tool to get the point O masked.

Use icon at the right edge of the points expandable bar and choose *Circle center creation* then click on the circle. A new point appears.

Use tool to create the midpoint of the segment joining the point *M* to the circle center. We will call it *I*.
Now create the circle with center *I* and going through point *O*.

Use intersection tool  to create the intersection of the two circles. Two points are created we will refer to as P and Q.

Use tool  to create lines (MP) and (MQ). These are our two tangents.

If necessary use icon  to get access to more tools.

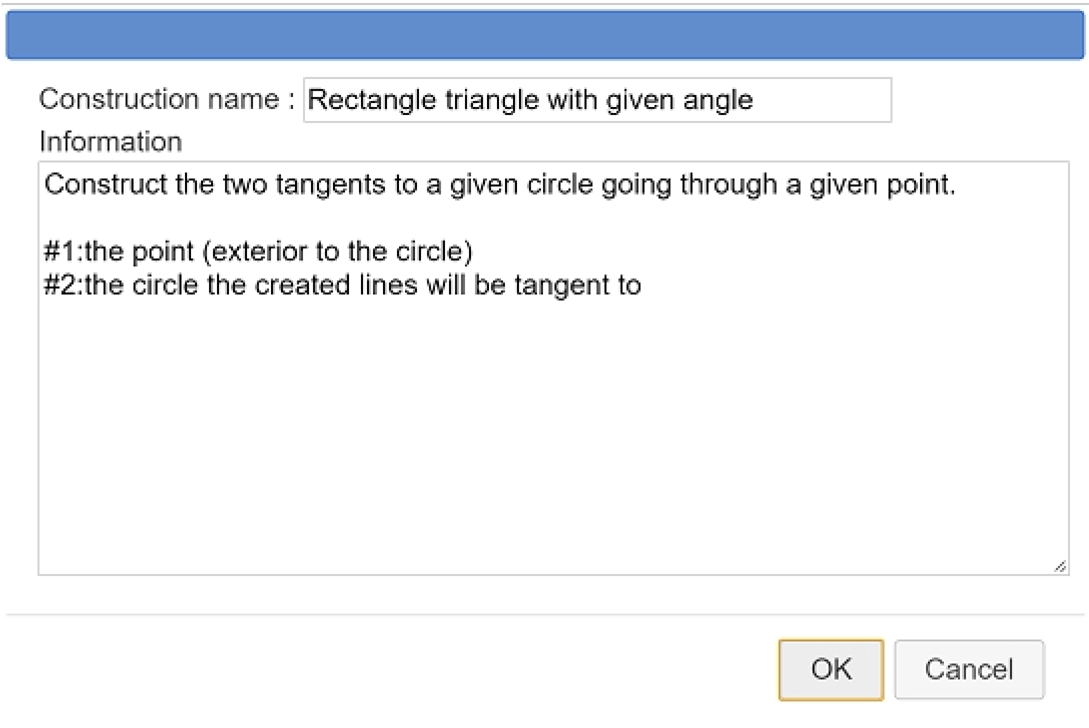
Use icon  and choose *Graphical sources objects choice*.
First click on M then click on the first circle.

Again use icon  and choose *Graphical final objects choice*.
Click on the two tangents (lines (MP) and (MQ)).

Now we have to finalize our construction.

Use icon  then choose *Finish current construction*.

A dialog box pops up. Fill it as below :



The first line give information about the construction.
The last two lines purpose is to give what indication will be displayed when the user will choose his sources objects.

Validate the dialog box. The construction is created. If you save your figure in a file, the construction will be saved along with the figure.

Now let us save this construction in a *mgc* file.

Use icon  and choose *Save a construction of the figure.*

A dialog box pops up. The only construction is already selected. Click on the button **Save file** and save your construction in a *mgc* file (you must choose a name for the file).

Let us now use this construction in a another figure.

Use icon  to create a new figure (with or without a frame).

Create a circle and a point exterior to the circle.

Use icon  and choose *Incorporate construction from file*. Browse to the place you saved your construction to before and incorporate this *mgc* file in your figure.

Now use icon  and choose *Implement a construction of the figure*. A dialog box pops up with the construction already selected. Click on button **Implement**.

The first indication displayed is to click on the point (exterior to the circle). Click on the point.
Then click on the circle.
The construction is now implemented et the final objects are visible (the two tangents).