`pst-euclide`

A PSTricks package for drawing geometric pictures; v.1.65

Dominique Rodriguez
Herbert Voß

October 10, 2019

The `pst-eucl` package allow the drawing of Euclidean geometric figures using LaTeX macros for specifying mathematical constraints. It is thus possible to build point using common transformations or intersections. The use of coordinates is limited to points which controlled the figure.

I would like to thanks the following persons for the help they gave me for development of this package:

- Denis Girou pour ses critiques pertinentes et ses encouragement lors de la découverte de l'embryon initial et pour sa relecture du présent manuel;
- Michael Vulis for his fast testing of the documentation using VTeX which leads to the correction of a bug in the `PostScript` code;
- Manuel Luque and Olivier Reboux for their remarks and their examples.
- Alain Delplanque for its modification propositions on automatic placing of points name and the ability of giving a list of points in `\pstGeonode`.

Thanks to: Manuel Luque; Thomas Söll.

Contents

Part I. The package

1. Special specifications

1.1. PSTricks Options

The package activates the `\SpecialCoor` mode. This mode extend the coordinates specification. Furthermore the plotting type is set to `dimen=middle` , which indicates that the position of the drawing is done according to the middle of the line. Please look at the user manual for more information about these setting.

At last, the working axes are supposed to be (ortho)normed.

1.2. Conventions

For this manual, I used the geometric French conventions for naming the points:

- O is a centre (circle, axes, symmetry, homothety, rotation);
- I defined the unity of the abscissa axe, or a midpoint;
- J defined the unity of the ordinate axe;
- A, B, C, D are points ;
- M' is the image of M by a transformation ;

At last, although these are nodes in PSTricks, I treat them intentionally as points.

2. Basic Objects

2.1. Points

`\pstGeonode [Options]` $(x_1,y_1)\{A_1\}(x_2,y_2)\{A_1\}\ldots(x,y)\{A_n\}$

This command defines one or more geometrical points associated with a node in the default cartesian coordinate system. Each point has a node name A_i which defines the default label put on the picture. This label is managed by default in mathematical mode, the boolean parameter `PtNameMath` (default `true`) can modify this behavior and let manage the label in normal mode. It is placed at a distance of `PointNameSep` (default 1em) of the center of the node with a angle of `PosAngle` (default 0). It is possible to specify another label using the parameter `PointName=default` , and an empty label can be specified by selecting the value `none`, in that case the point will have no name on the picture.

The point symbol is given by the parameter `PointSymbol=*` . The symbol is the same as used by the macro `\psdot`. This parameter can be set to `none`, which means that the point will not be drawn on the picture.

Here are the possible values for this parameter:

- `*`: ●
- `o`: ○
- `+`: +
- `x`: ·
- `asterisk` : ⋆
- `oplus`: ¯
- `otimes`: ˜
- `triangle`: △
- `triangle*`: ▲
- `square`: □
- `square*`: ■
- `diamond`: ◊
- `diamond*`: ◆
- `pentagon`: ⬠
- `pentagon*`: ⬟
- `|`: |

Furthermore, these symbols can be controlled with some others PSTricks, several of these are :

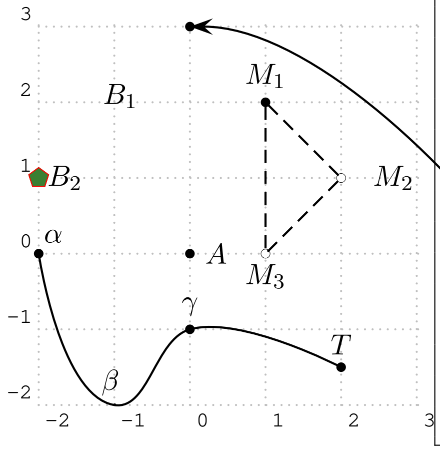
- their scale with dotscale, the value of whom is either two numbers defining the horizontal and vertical scale factor, or one single value being the same for both,
- their angle with parameter dotangle.

Please consult the PSTricks documentation for further details. The parameters PosAngle, PointSymbol, PointName and PointNameSep can be set to :

- either a single value, the same for all points ;
- or a list of values delimited by accolads { ... } and separated with comma *without any blanks*, allowing to differenciate the value for each point.

In the later case, the list can have less values than point which means that the last value is used for all the remaining points. At least, the parameter setting CurveType=none can be used to draw a line between the points:

- opened polyline ;
- closed polygon ;
- open and curved curve.



```
\begin{pspicture}[showgrid=true](-2,-2)(3,3)
\pstGeonode{A}
\pstGeonode[PosAngle=-135, PointNameSep=1.3](0,3){B_1}
\pstGeonode[PointSymbol=pentagon, dotscale=2, fillstyle=solid,
         fillcolor=OliveGreen, PtNameMath=false,
         PointName=$B_2$, linecolor=red](-2,1){B2}
\pstGeonode[PosAngle={90,0,-90}, PointSymbol={*,o},
         linestyle=dashed, CurveType=polygon,
         PointNameSep={1em,2em,3mm}]
  (1,2){M_1}(2,1){M_2}(1,0){M_3}
\pstGeonode[PosAngle={50,100,90}, PointSymbol={*,x,*},
         PointNameSep=3mm, CurveType=curve,
         PointName={\alpha,\beta,\gamma,default}]
  (-2,0){alpha}(-1,-2){beta}(0,-1){gamma}(2,-1.5){T}
\end{pspicture}
```

Obviously, the nodes appearing in the picture can be used as normal PSTricks nodes. Thus, it is possible to reference a point from here.

After v1.65, we add macros \pstAbscissa and \pstOrdinate to get the abscissa and ordinate of the specified node, so it is possible to define a new node from an already constructed node with them.

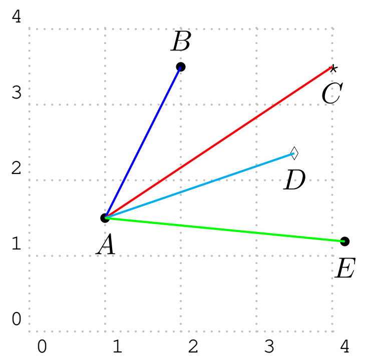
```
\pstAbscissa{A}
\pstOrdinate{A}
```

Note that the value of abscissa or ordinate are transformed to the User coordinate, and then put into the stack of PostScript, so they can be used to do some compound arithmetic(e.g, \pscalculate) without concerned the xunit and yunit in the PSTricks SpecialCoor function.

The macro \pstMoveNode use them to move node A by abscissa increment dx and ordinate increment dy to get the target node B.

```
\pstMoveNode [Options] (dx,dy){A}{B}
```

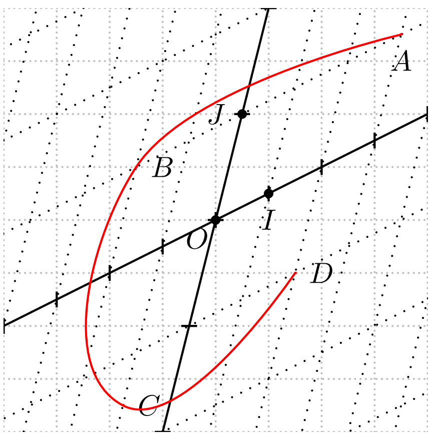
for example:



```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\def\ra{3.0}\def\rb{4.0}
\pstGeonode[PosAngle=-90](1.0,1.5){A}
\pstGeonode[PosAngle=90](! \pstAbscissa{A} 1 add \pstOrdinate{A}
    2 add){B}
\pstLineAB[linecolor=blue]{A}{B}
\pstMoveNode[PosAngle=-90,PointSymbol=asterisk](3,2){A}{C}
\pstLineAB[linecolor=red]{A}{C}
\pstMoveNode[PosAngle=-90,PointSymbol=diamond](\pscalculate{sqrt
    (\ra*\ra+\rb*\rb)/2},\pscalculate{\ra*\rb/(2*(\ra+\rb))}){A
    }{D}
\pstLineAB[linecolor=cyan]{A}{D}
\pstMoveNode[PosAngle=-90](\pstAbscissa{B} 3 div,\pstOrdinate{B}
    neg 3 div){D}{E}
\pstLineAB[linecolor=green]{A}{E}
\end{pspicture}
```

`\pstOIJGeonode` creates a list of points in the landmark $(O; I; J)$. Possible parameters are `PointName`, `PointNameSep`, `PosAngle`, `PointSymbol`, and `PtNameMath`.

`\pstOIJGeonode` `[Options]` $(x_1,y_1)\{A_1\}\{O\}\{I\}\{J\}$ $(x_2,y_2)\{A_2\}\ldots(x,y)\{A_n\}$



```
\psset{unit=.7}
\begin{pspicture*}[showgrid=true](-4,-4)(4,4)
  \pstGeonode[PosAngle={-135,-90,180}]{O}(1,0.5){I}(0.5,2){J}
  \pstLineAB[nodesep=10]{O}{I}
  \pstLineAB[nodesep=10]{O}{J}
  \multips(-5,-2.5)(1,0.5){11}{\psline(0,-.15)(0,.15)}
  \multips(-2,-8)(0.5,2){9}{\psline(-.15,0)(.15,0)}
  \psset{linestyle=dotted}%
  \multips(-5,-2.5)(1,0.5){11}{\psline(-10,-40)(10,40)}
  \multips(-2,-8)(0.5,2){9}{\psline(-10,-5)(10,5)}
  \psset{PointSymbol=x, linestyle=solid}
  \pstOIJGeonode[PosAngle={-90,0}, CurveType=curve,
   linecolor=red] (3,1){A}{O}{I}{J}(-2,1){B}(-1,-1.5){C}(2,-1){D}
\end{pspicture*}
```

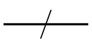
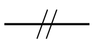
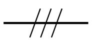
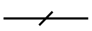
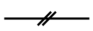
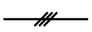
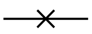
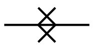
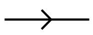
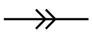
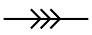
2.2. Segment mark

A segment can be drawn using the `\ncline` command. However, for marking a segment there is the following command:

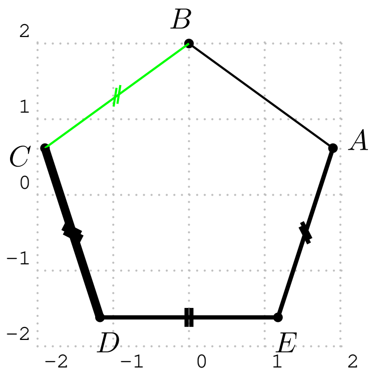
`\pstSegmentMark` `[Options]` $\{A\}\{B\}$

The symbol drawn on the segment is given by the parameter `SegmentSymbol`. Its value can be any valid command which can be used in math mode. Its default value is `MarkHashh`, which produced two slashes on the segment. The segment is drawn.

Several commands are predefined for marking the segment:

- `pstslash` 
- `pstslashh` 
- `pstslashhh` 
- `MarkHash` 
- `MarkHashh` 
- `MarkHashhh` 
- `MarkCros` 
- `MarkCross` 
- `MarkArrow` 
- `MarkArroww` 
- `MarkArrowww` 

The three commands of the family MarkHash draw a line whose inclination is controled by the parameter MarkAngle (default is 45). Their width and colour depends of the width and color of the line when the drawing is done, as shown is the next example.



```
\begin{pspicture}[showgrid=true](-2,-2)(2,2)
 \rput{18}{%
  \pstGeonode[PosAngle={0,90,180,-90}](2,0){A}(2;72){B}
   (2;144){C}(2;216){D}(2;288){E}}
 \pstSegmentMark[SegmentSymbol=none]{A}{B}
 \pstSegmentMark[linecolor=green]{B}{C}
 \psset{linewidth=2\pslinewidth}
 \pstSegmentMark[linewidth=2\pslinewidth]{C}{D}
 \pstSegmentMark[MarkAngle=90]{D}{E}
 \pstSegmentMark{E}{A}
\end{pspicture}
```

The length and the separation of multiple hases can be set by MarkHashLength and MarkHashSep.

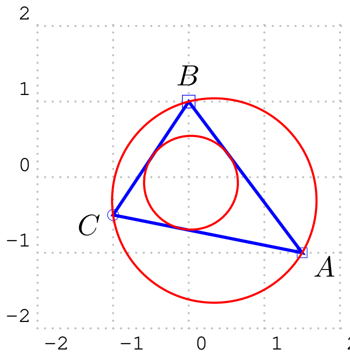
2.3. Triangles

The more classical figure, it has its own macro for a quick definition:

\pstTriangle [Options] $(x_1,y_1)\{A\}(x_2,y_2)\{B\}(x_3,y_3)\{C\}$
\pstTriangleIC [Options] $\{A\}\{B\}\{C\}$
\pstTriangleOC [Options] $\{A\}\{B\}\{C\}$

Valid optional arguments are PointName, PointNameSep, PointSymbol, PointNameA, PosAngleA, PointSymbolA, PointNameB, PosAngleB, PointSymbolB, PointNameC, PosAngleC, and PointSymbolC. In order to accurately put the name of the points, there are three parameters PosAngleA, PosAngleB and PosAngleC, which are associated respectively to the nodes $\langle A\rangle$, $\langle B\rangle$ and $\langle C\rangle$. Obviously they have the same meaning as the parameter PosAngle. If one or more of such parameters is omitted, the value of PosAngle is taken. If no angle is specified, points name are placed on the bissector line.

In the same way there are parameters for controlling the symbol used for each points: PointSymbolA, PointSymbolB and PointSymbolC. They are equivalent to the parameter PointSymbol. The management of the default value followed the same rule.



```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\pstTriangle[PointSymbol=square,PointSymbolC=o,
 linecolor=blue,linewidth=1.5\pslinewidth]
 (1.5,-1){A}(0,1){B}(-1,-.5){C}
\pstTriangleIC[linecolor=red]{A}{B}{C}
\pstTriangleOC[linecolor=red]{A}{B}{C}
\end{pspicture}
```

The center of the inner circle is called IC_O and the outer circle OC_O. They are only defined, if the macros \pstTriangleIC and \pstTriangleOC are used.

2.4. Angles

Each angle is defined with three points. The vertex is the second point. Their order is important because it is assumed that the angle is specified in the direct order. The first command is the marking of a right angle:

```
\pstRightAngle [Options] {A}{B}{C}
```

Valid optional arguments are `RightAngleType`, `RightAngleSize`, and `RightAngleSize`

The symbol used is controlled by the parameter `RightAngleType default`. Its possible values are :

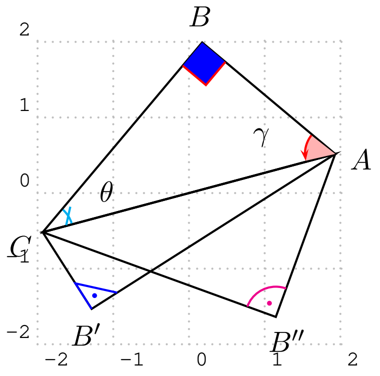
- $*$: standard symbol ;
- `german` : german symbol (given by U. Dirr) ;
- `suisseromand` : swiss romand symbol (given P. Schnewlin).

The only parameter controlling this command, excepting the ones which controlled the line, is `RightAngleSize` which defines the size of the symbol (by default 0.28 unit).

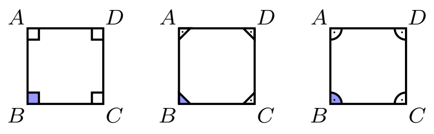
For other angles, there is the command:

```
\pstMarkAngle [Options] {A}{B}{C}
```

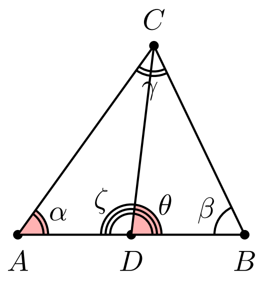
Valid optional arguments are `MarkAngleRadius`, `LabelAngleOffset`, `MarkAngleType` and `Mark` The `label` can be any valid TEX box, it is put at `LabelSep` (by default 1 unit) of the node in the direction of the bisector of the angle modified by `LabelAngleOffset`(by default 0) and positioned using `LabelRefPt` (by default c). Furthermore the arc used for marking has a radius of `MarkAngleRadius` (by default .4 unit). At least, it is possible to place an arrow using the parameter `arrows`.Finally, it is possible to mark the angle by specifying a TEX command as argument of parameter `Mark`.



```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psset{PointSymbol=none}
\pstTriangle(2;15){A}(2;85){B}(2;195){C}
\psset{PointName=none}
\pstTriangle[PointNameA=default](2;-130){B'}(2;15){A'}(2;195){C'}
\pstTriangle[PointNameA=default](2;-55){B''}(2;15){A''}(2;195){C''}
\pstRightAngle[linecolor=red,fillstyle=solid,fillcolor=blue]{C}{B}{A}
\pstRightAngle[linecolor=blue, RightAngleType=suisseromand]{A}{B'}{C}
\pstRightAngle[linecolor=magenta, RightAngleType=german]{A}{B''}{C}
\psset{arcsep=\pslinewidth}
\pstMarkAngle[linecolor=cyan, Mark=MarkHash]{A}{C}{B}{$\theta$}
\pstMarkAngle[linecolor=red, arrows=->,fillcolor=red!30,
  fillstyle=solid]{B}{A}{C}{$\gamma$}
\end{pspicture}
```



```
\begin{pspicture}(-0.5,-0.5)(9,3)
\psset{PointSymbol=none,PointNameMathSize=\scriptstyle,PointNameSep=6pt,
    RightAngleSize=0.15,PosAngle={135,225,-45,45}}
\pstGeonode(1,2){A}(1,1){B}(2,1){C}(2,2){D}%
\pstRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\pstRightAngle{D}{C}{B} \pstRightAngle{A}{D}{C}
\pstRightAngle{B}{A}{D} \pspolygon(A)(B)(C)(D)
\psset{RightAngleType=suisseromand}
\pstGeonode(3,2){A}(3,1){B}(4,1){C}(4,2){D}%
\pstRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\pstRightAngle{D}{C}{B} \pstRightAngle{A}{D}{C}
\pstRightAngle{B}{A}{D} \pspolygon(A)(B)(C)(D)
\psset{RightAngleType=german}
\pstGeonode(5,2){A}(5,1){B}(6,1){C}(6,2){D}%
\pstRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\pstRightAngle{D}{C}{B} \pstRightAngle{A}{D}{C}
\pstRightAngle{B}{A}{D} \pspolygon(A)(B)(C)(D)
\end{pspicture}
```



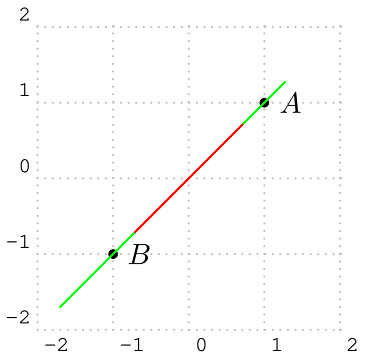
```
\begin{pspicture}[showgrid=false](-1.0,-1.0)(4,4)
\pstGeonode[PosAngle=-90](0.0,0.0){A}
\pstGeonode[PosAngle=-90](3.0,0.0){B}
\pstGeonode[PosAngle=90](1.8,2.5){C}
\pstGeonode[PosAngle=-90](1.5,0.0){D}
\pstMarkAngle[LabelSep=.6,MarkAngleRadius=.4,MarkAngleType=double]{A}{C}{B}{$\gamma$}
\pstMarkAngle[LabelSep=.6,MarkAngleRadius=.4,MarkAngleType=default]{C}{B}{A}{$\beta$}
\pstMarkAngle[LabelSep=.6,MarkAngleRadius=.4,MarkAngleType=double,fillcolor=red!30,fillstyle=solid]{B
    }{A}{C}{$\alpha$}
\pstMarkAngle[LabelSep=.6,MarkAngleRadius=.4,MarkAngleType=triple,fillcolor=red!30,fillstyle=solid]{B
    }{D}{C}{$\theta$}
\pstMarkAngle[LabelSep=.6,MarkAngleRadius=.4,MarkAngleType=triple]{C}{D}{A}{$\zeta$}
\pstLineAB{A}{B}
\pstLineAB{B}{C}
\pstLineAB{C}{A}
\pstLineAB{C}{D}
\end{pspicture}
```

2.5. Lines, half-lines and segments

The classical line $(\overline{AB})$!

```
\pstLineAB [Options] {A}{B}
```

In order to control its length[1], the two parameters `nodesepA` et `nodesepB` specify the abscissa of the extremity of the drawing part of the line. A negative abscissa specify an outside point, while a positive abscissa specify an internal point. If these parameters have to be equal, `nodesep` can be used instead. The default value of these parameters is equal to 0.



```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\pstGeonode(1,1){A}(-1,-1){B}
\pstLineAB[nodesepA=-.4,nodesepB=-1,
        linecolor=green]{A}{B}
\pstLineAB[nodesep=.4,linecolor=red]{A}{B}
\end{pspicture}
```

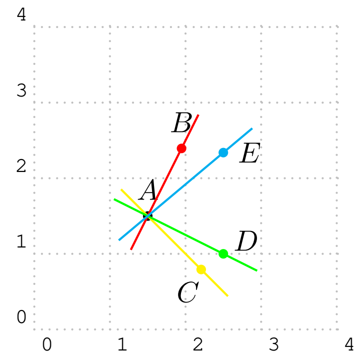
The macro `\pstLine` draws a new line with two nodes, or two coordinate or one node and one coordinate. This macro is similar with `\pstLineAB`, but more compatible.

```
\pstLine [Options] {A}{B}
\pstLine [Options] {A}(x,y)
\pstLine [Options] (x,y){B}
\pstLine [Options] (x,y)(x,y)
```

The macros `\pstLineAA` and `\pstLineAS` draw a new line with one node, the slope `angle` between the line and the horizontal axis, or the slope `gradient` of the line, and create a new node B on the line.

```
\pstLineAA [Options] {A}{angle}{B}
\pstLineAA [Options] (x,y){angle}{B}
\pstLineAS [Options] {A}{gradient}{B}
\pstLineAS [Options] (x,y){gradient}{B}
```

Here are some examples:



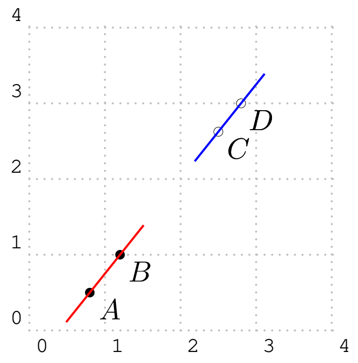
```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\pstGeonode[PosAngle=90](1.5,1.5){A}
% draw a line with angle atan(2/1), about 63.43 degree.
\pstLineAA[linecolor=red,nodesep=-0.5,PosAngle=90]{A}{2 1 atan}{B
    }
\pstLineAA[linecolor=yellow,nodesep=-0.5,PosAngle=-120]{A}{-45}{C
    }
\pstLineAS[linecolor=green,nodesep=-0.5,PosAngle=30]{A}{-0.5}{D}
% draw a line with gradient (cos50/sin50).
\pstLineAS[linecolor=cyan,nodesep=-0.5]{A}{50 cos 50 sin div}{E}
\end{pspicture}
```

1 which is the comble for a line!

The macro `\pstLineAbsNode` creates a new node C whose abscissa is the given value x_1 on the line AB. The macro `\pstLineOrdNode` creates a new node C whose ordinate is the given value y_1 on the line AB. You can input x_1 or y_1 as any number(e.g, 2.0), or use `\pscalculate` to generate the value, or use `\pstAbscissa` and `\pstOrdinate` to get the abscissa and ordinate of any other node.

```
\pstLineAbsNode [Options] {A}{B}{x1}{C}
\pstLineOrdNode [Options] {A}{B}{y1}{C}
```

For example,



```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\pstGeonode[PosAngle=-40](0.8,0.5){A}
\pstGeonode[PosAngle=-40](1.2,1.0){B}
\pstLineAB[linecolor=red,nodesep=-0.5]{A}{B}
\pstLineAbsNode[PosAngle=-40,PointSymbol=o]{A}{B}{2.5}{C}
\pstLineOrdNode[PosAngle=-40,PointSymbol=o]{A}{B}{3.0}{D}
\pstLineAB[linecolor=blue,nodesep=-0.5]{C}{D}
\end{pspicture}
```

2.6. Circles

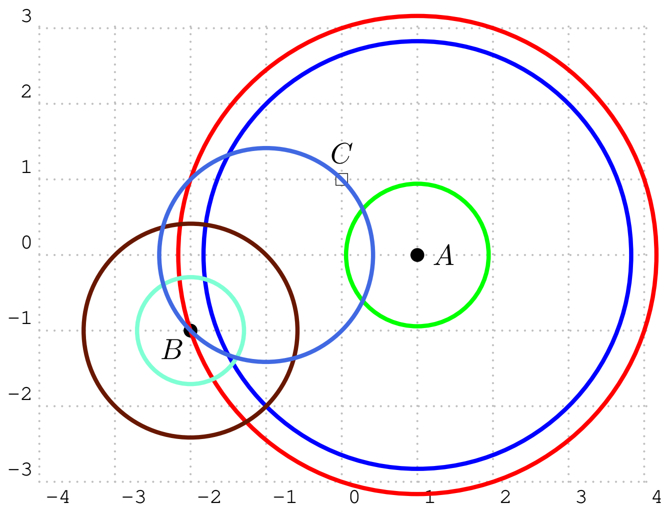
A circle can be defined either with its center and a point of its circumference, or with two diameterly opposed points. There is two commands :

```
\pstCircleOA [Options] {O}{A}
\pstCircleAB [Options] {O}{A}
\pstDistAB [Options] {A}{B}
\pstDistVal [Options] {x}
```

For the first macro, it is possible to omit the second point and then to specify a radius or a diameter using the parameters `Radius` and `Diameter`. The values of these parameters must be specified with one of the two following macros :

The first specifies a distance between two points. The parameter `DistCoef` can be used to specify a coefficient to reduce or enlarge this distance. To be taken into account this last parameter must be specified before the distance. The second macro can be used to specify an explicit numeric value. We will see later how to draw the circle crossing three points. With this package, it becomes possible to draw:

- the circle of center A crossing B;
- the circle of center A whose radius is AC;
- the circle of center A whose radius is BC;
- the circle of center B whose radius is AC;
- the circle of center B of diameter AC;
- the circle whose diameter is BC.

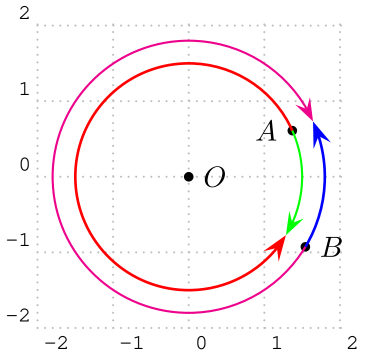


```
\begin{pspicture}[showgrid](-4,-3.3)(4,3)
\psset{linewidth=2\pslinewidth}
\pstGeonode[PosAngle={0,-135,90},PointSymbol={*,*,square}](1,0){A}(-2,-1){B}(0,1){C}
\pstCircleOA[linecolor=red]{A}{B}
\pstCircleOA[linecolor=green, DistCoef=2 3 div, Radius=\pstDistAB{A}{C}]{A}{}
\pstCircleOA[linecolor=blue, Radius=\pstDistAB{B}{C}]{A}{}
\pstCircleOA[linecolor=Sepia, Radius=\pstDistAB{A}{C}]{B}{}
\pstCircleOA[linecolor=Aquamarine, Diameter=\pstDistAB{A}{C}]{B}{}
\pstCircleAB[linecolor=RoyalBlue]{B}{C}
\end{pspicture}
```

2.7. Circle arcs

```
\pstArcOAB [Options] {O}{A}{B}
\pstArcnOAB [Options] {O}{A}{B}
```

These two macros draw circle arcs, O is the center, the radius defined by OA, the beginning angle given by A and the final angle by B. Finally, the first macro draws the arc in the direct way, whereas the second in the indirect way. It is not necessary that the two points are at the same distance of O.



```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\pstGeonode[PosAngle={180,0}](1.5;24){A}(1.8;-31){B}
\pstGeonode{O}
\psset{arrows=->,arrowscale=2}
\pstArcOAB[linecolor=red,linewidth=1pt]{O}{A}{B}
\pstArcOAB[linecolor=blue,linewidth=1pt]{O}{B}{A}
\pstArcnOAB[linecolor=green]{O}{A}{B}
\pstArcnOAB[linecolor=magenta]{O}{B}{A}
\end{pspicture}
```

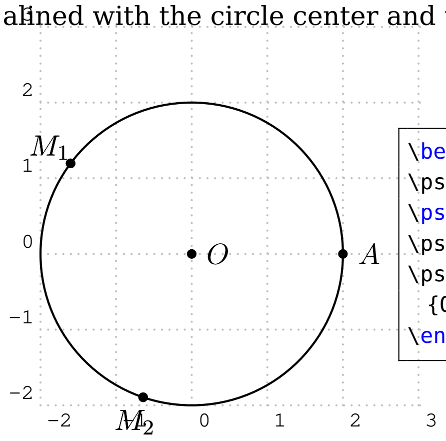
2.8. Curved abscissa

A point can be positioned on a circle using its curved abscissa.

```
\pstCurvAbsNode [Options] {O}{A}{B}{Abs}
```

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, `PtNameMath`, and `CurvAbsNeg`. The point $\langle B\rangle$ is positioned on the circle of center $\langle O\rangle$ crossing $\langle A\rangle$, with the curved abscissa $\langle \text{Abs}\rangle$. The origin is $\langle A\rangle$ and the direction is anti-clockwise by default. The parameter `CurvAbsNeg` (by default false) can change this behavior.

If the parameter `PosAngle` is not specified, the point label is put automatically in oirder to be alined with the circle center and the point.

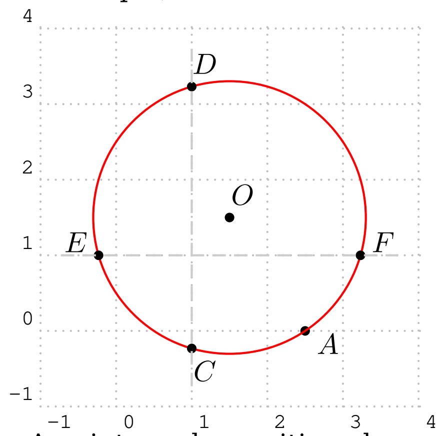


```
\begin{pspicture}[showgrid](-2.5,-2.5)(2.5,2.5)
\pstGeonode{O}(2,0){A}
\pstCircleOA{O}{A}
\pstCurvAbsNode{O}{A}{M_1}{\pstDistVal{5}}
\pstCurvAbsNode[CurvAbsNeg=true]%
 {O}{A}{M_2}{\pstDistAB{A}{M_1}}
\end{pspicture}
```

A point can be positioned on a circle using its absolute abscissa or ordinate too. You can input x_1 or y_1 as any number(e.g, 2.0), or use `\pscalculate` to generate the value, or use `\pstAbscissa` and `\pstOrdinate` to get the abscissa and ordinate of any other node.

`\pstCircleAbsNode [Options]` $\{O\}\{A\}\{x_1\}\{C\}\{C\}$
`\pstCircleOrdNode [Options]` $\{O\}\{A\}\{y_1\}\{C\}\{C\}$

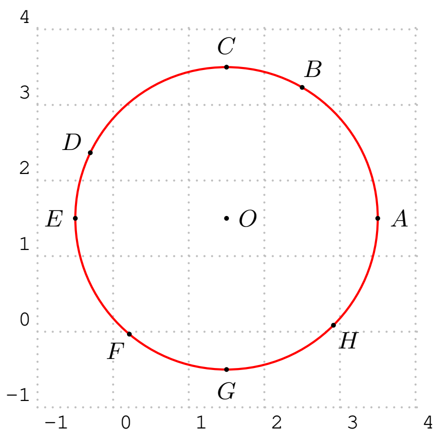
for example,



```
\begin{pspicture}[showgrid=true](-1,-1)(4,4)
\pstGeonode[PosAngle=60](1.5,1.5){O}
\pstGeonode[PosAngle=-30](2.5,0){A}
\pstCircleOA[linecolor=red]{O}{A}
\pstCircleAbsNode[PosAngleA=-60,PosAngleB=60,PointSymbol=*]{O}{A
    }{1.0}{C}{D}
\pstCircleOrdNode[PosAngleA=150,PosAngleB=30,PointSymbol=*]{O}{A
    }{1.0}{E}{F}
\pstLineAB[linestyle=dashed,linecolor=gray!40,nodesep=-0.5]{C}{D}
\pstLineAB[linestyle=dashed,linecolor=gray!40,nodesep=-0.5]{E}{F}
\end{pspicture}
```

A point can be positioned on a circle using its rotation angle by macro `\pstCircleRotNode`. The rotation angle should be passed by the `RotAngle` in the `Options`. The circle is defined by center O and point A on the circle or `Radius` in parameter. If you not set `RotAngle`, the default value is $60°$.

`\pstCircleRotNode [Options]` $\{O\}\{A\}\{X\}$



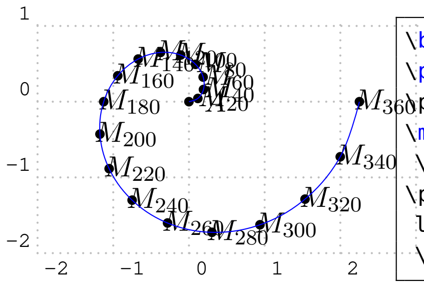
```
\begin{pspicture}[showgrid=true](-1,-1)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}
\pstGeonode[PosAngle=0](1.5,1.5){O}
\pstCircleOA[linecolor=red,Radius=\pstDistVal{\ra}]{O}{}
\pstCircleRotNode[PosAngle=0,RotAngle=0,Radius=\pstDistVal{\ra}]{
    O}{}{A}
\pstCircleRotNode[PosAngle=60,Radius=\pstDistVal{\ra}]{O}{}{B} %
    default 60 degree
\pstCircleRotNode[PosAngle=90,RotAngle=90,Radius=\pstDistVal{\ra
    }]{O}{}{C}
\pstCircleRotNode[PosAngle=150,RotAngle=\pscalculate{3*360/7},
    Radius=\pstDistVal{\ra}]{O}{}{D}
\pstCircleRotNode[PosAngle=180,RotAngle=180,Radius=\pstDistVal{\
    ra}]{O}{}{E}
\pstCircleRotNode[PosAngle=230,RotAngle=230,Radius=\pstDistVal{\
    ra}]{O}{}{F}
\pstCircleRotNode[PosAngle=270,RotAngle=270,Radius=\pstDistVal{\
    ra}]{O}{}{G}
\pstCircleRotNode[PosAngle=-45,RotAngle=-45,Radius=\pstDistVal{\
    ra}]{O}{}{H}
\end{pspicture}
```

2.9. Generic curve

It is possible to generate a set of points using a loop, and to give them a generic name defined by a radical and a number. The following command can draw a interpolated curve crossing all such kind of points.

`\pstGenericCurve [Options]` {*Radical*}{n_1}{n_2}

Possible optional arguments are `GenCurvFirst`, `GenCurvInc`, and `GenCurvLast` The curve is drawn on the points whose name is defined using the radical ⟨Radical⟩ followed by a number from $\langle n_1 \rangle$ to $\langle n_2 \rangle$. In order to manage side effect, the parameters `GenCurvFirst` et `GenCurvLast` can be used to specified special first or last point. The parameter `GenCurvInc` can be used to modify the increment from a point to the next one (by default 1).



```
\begin{pspicture}[showgrid](-2.5,-2.5)(2.5,1)
\psset{unit=.00625}
\pstGeonode{A}
\multido{\n=20+20}{18}{%
 \pstGeonode[PointName=M_{\n}](\n;\n){M_\n}}
\pstGenericCurve[GenCurvFirst=A,GenCurvInc=20,
 linecolor=blue,linewidth=.5\pslinewidth]{M_}{20}{360}
 \end{pspicture}
```

3. Conics

3.1. Standard Ellipse

The Standard Ellipse with coordinate translation E is defined by center $O(x_0, y_0)$, the half of the major axis $max(abs(a), abs(b))$, the half of the minor axis $min(abs(a), abs(b))$, the equation as following:

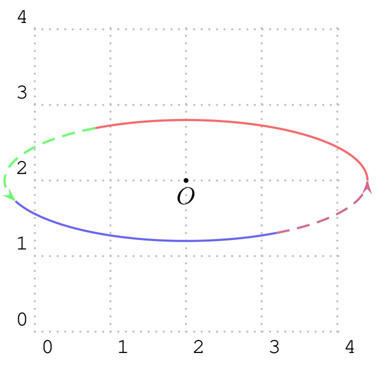
$$\frac{(x-x_o)^2}{a^2} + \frac{(y-y_o)^2}{b^2} = 1 \qquad (1)$$

Sometimes we use the parametric function of the Standard Ellipse with coordinate translation:

$$\begin{cases} x = a\cos\alpha + x_o \\ y = b\sin\alpha + y_0 \end{cases} \tag{2}$$

The Macro `\pstEllipse` is used to draw a Standard Ellipse with center O from `angleA` to `angleB`, going counter clockwise. It combines the function like `\psellipse` and `\psellipticarc` in `PSTricks`. If `angleA` and `angleB` are not specified, the macro will draw the whole ellipse.

`\pstEllipse [Options] (`O`)(`a, b`) [angleA] [angleB]`

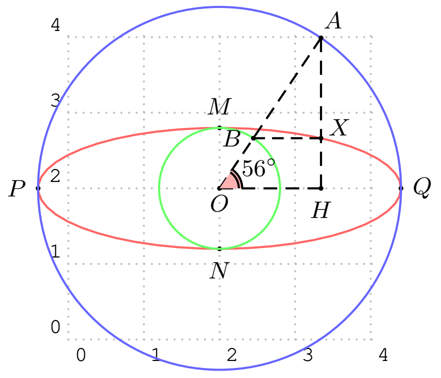


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{0.8}\def\rot{56}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
%\psellipse[linecolor=red!60](O)(\ra,\rb)
\pstEllipse[linecolor=red!60](O)(\ra,\rb)[0][120]
\pstEllipse[linecolor=green!60,linestyle=dashed,arrows=->,
    arrowscale=1.2](O)(\ra,\rb)[120][200]
\pstEllipse[linecolor=blue!60](O)(\ra,\rb)[200][300]
\pstEllipse[linecolor=purple!60,linestyle=dashed,arrows=->,
    arrowscale=1.2](O)(\ra,\rb)[300][360]
\end{pspicture}
```

Now you can draw some points on this Ellipse using macro `\pstEllipseNode` or `\pstEllipseRotNode`. The macro `\pstEllipseNode` requires an explicit parameter t as α in equation (2) to calculate the point; but the macro `\pstEllipseRotNode` requires an implicit parameter `RotAngle` as α in equation (2) to calculate the point.

`\pstEllipseNode [Options] (`O`)(`a, b`){`t`}{`P`}`
`\pstEllipseRotNode [Options] (`O`)(`a, b`){`P`}`

The following is the example, note that the `RotAngle` is not $\angle HOX$ in geometrical, but $\angle HOA$ or $\angle HOB$.

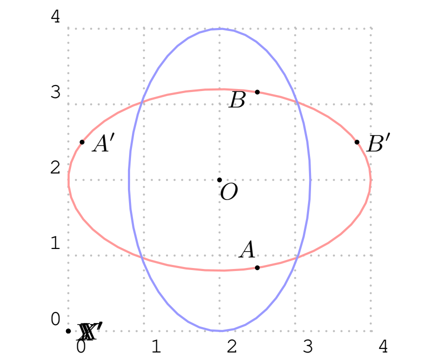


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{0.8}\def\rot{56}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
%\psellipse[linecolor=red!60](O)(\ra,\rb)
\pstEllipse[linecolor=red!60](O)(\ra,\rb)
\pstEllipseNode[PosAngle=180](O)(\ra,\rb){180}{P}
\pstEllipseRotNode[PosAngle=0,RotAngle=0](O)(\ra,\rb){Q}
\pstEllipseRotNode[PosAngle=90,RotAngle=90](O)(\ra,\rb){M}
\pstEllipseRotNode[PosAngle=-90,RotAngle=-90](O)(\ra,\rb){N}
\pstCircleOA[linecolor=blue!60,Radius=\pstDistVal{\ra}]{O}{}
\pstCircleRotNode[PosAngle=\rot,RotAngle=\rot,Radius=\pstDistVal
    {\ra}]{O}{}{A}
\pstCircleOA[linecolor=green!60,Radius=\pstDistVal{\rb}]{O}{}
\pstCircleRotNode[PosAngle=180,RotAngle=\rot,Radius=\pstDistVal{\
    rb}]{O}{}{B}
\pstEllipseRotNode[PosAngle=30,RotAngle=\rot](O)(\ra,\rb){X}
\pstProjection[PosAngle=-90]{P}{Q}{A}[H]
\pstLineAB[linestyle=dashed]{A}{O}
\pstLineAB[linestyle=dashed]{A}{H}
\pstLineAB[linestyle=dashed]{B}{X}
\pstLineAB[linestyle=dashed]{O}{H}
\pstMarkAngle[LabelSep=.6,MarkAngleRadius=.3,MarkAngleType=double
    ,fillcolor=red!30,fillstyle=solid]{H}{O}{A}{$\rot^\circ$}
\end{pspicture}
```

The macros `\pstEllipseAbsNode` and `\pstEllipseOrdNode` are used to get the two nodes A and B whose abscissas or ordinates are the given value x_1 or y_1 on the Standard Ellipse E.

If there is no such point satisfied this condition, then the nodes A and B will be put at the origin.

\pstEllipseAbsNode [Options] (O)(a, b){x_1}{A}{B}
\pstEllipseOrdNode [Options] (O)(a, b){y_1}{A}{B}

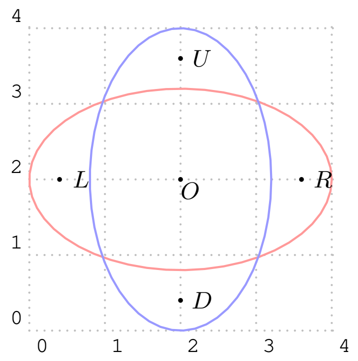


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](O)(\ra,\rb)
\pstEllipse[linecolor=blue!40](O)(\rb,\ra)
\pstEllipseAbsNode[PosAngle={120,200}](O)(\ra,\rb){2.5}{A}{B}
\pstEllipseAbsNode(O)(\ra,\rb){6}{X}{Y} % not exist
\pstEllipseOrdNode(O)(\ra,\rb){2.5}{A'}{B'}
\pstEllipseOrdNode(O)(\ra,\rb){6}{X'}{Y'} % not exist
\end{pspicture}
```

Here we find the focus node of Standard Ellipse! Please use macro `\pstEllipseFocusNode` to do this work.

\pstEllipseFocusNode [Options] (O)(a, b){A}{B}

For example:

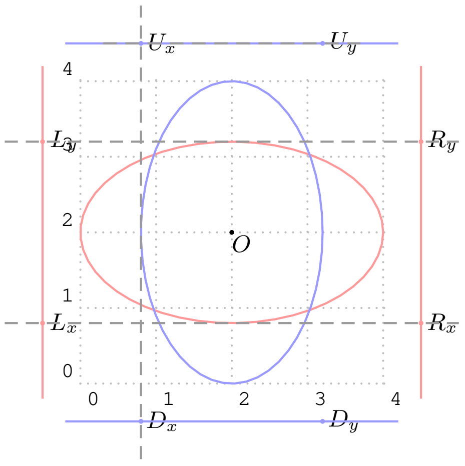


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](O)(\ra,\rb)
\pstEllipse[linecolor=blue!40](O)(\rb,\ra)
\pstEllipseFocusNode(O)(\ra,\rb){L}{R}
\pstEllipseFocusNode(O)(\rb,\ra){D}{U}
\end{pspicture}
```

The macro `\pstEllipseDirectrixLine` is used to draw the two directrix lines of Standard Ellipse, and create two new nodes on each of them. The nodes L_x, L_y are on the left/down directrix line, and R_x, R_y are on the right/up directrix line. They are lie on the tangent line of the vertex on the other axis.

`\pstEllipseDirectrixLine [Options] (`O`)(`a,b`){`L_x`}{`L_y`}{`R_x`}{`R_y`}`

For example:

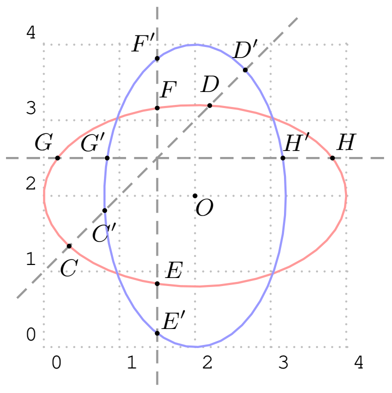


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](O)(\ra,\rb)
\pstEllipse[linecolor=blue!40](O)(\rb,\ra)
\pstEllipseDirectrixLine[PointName={L_x,L_y,R_x,R_y},nodesep=-1,
    linecolor=red!40](O)(\ra,\rb){Lx}{Ly}{Rx}{Ry}
\pstEllipseDirectrixLine[PointName={D_x,D_y,U_x,U_y},nodesep=-1,
    linecolor=blue!40](O)(\rb,\ra){Dx}{Dy}{Ux}{Uy}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{Lx}{Rx
    }
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{Ly}{Ry
    }
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{Dx}{Ux
    }
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{Ux}{Uy
    }
\end{pspicture}
```

Sometimes we need to find the intersection of Ellipse and line, the Macro `\pstEllipseLineInter` can do this work, and it can handle any type of line, i.e, horizontal, vertical or others lines. It get the two intersection C and D of the Standard Ellipse E and the given line AB. When there is none intersection, C and D are both put at the origin; When there is only on intersection, it will be saved at node C, and D will be put at the origin.

`\pstEllipseLineInter [Options] (`O`)(`a,b`){`A`}{`B`}{`C`}{`D`}`

Here is examples:



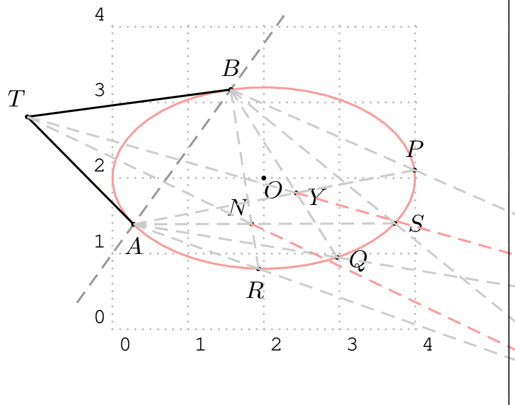
```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](O)(\ra,\rb)
\pstEllipse[linecolor=blue!40](O)(\rb,\ra)
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed
    ]{0,1}{3,4}
\pstEllipseLineInter[PosAngle={-90,90}](O)(\ra,\rb){0,1}{3,4}{C}{
    D}
\pstEllipseLineInter[PosAngle={-90,90}](O)(\rb,\ra){0,1}{3,4}{C
    '}{D'}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed
    ]{1.5,0}{1.5,4}
\pstEllipseLineInter[PosAngle={40,60}](O)(\ra,\rb){1.5,0}{1.5,4}{
    E}{F}
\pstEllipseLineInter[PosAngle={40,130}](O)(\rb,\ra)
    {1.5,1}{1.5,4}{E'}{F'}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed
    ]{4,2.5}{0,2.5}
\pstEllipseLineInter[PosAngle={130,50}](O)(\ra,\rb)
    {4,2.5}{0,2.5}{G}{H}
\pstEllipseLineInter[PosAngle={130,50}](O)(\rb,\ra)
    {4,2.5}{0,2.5}{G'}{H'}
\end{pspicture}
```

The macro `\pstEllipsePolarNode` is use to draw the tangent line of a point A or B on the Standard Ellipse. It draws the every tangent line through the point A and B on the Standard Ellipse E and get the insection node T of the two tangent lines. We call T as the polar point of chord AB as normal.

`\pstEllipsePolarNode [Options] ($O$)($a,b$){$A$}{$B$}{$T$}`

We use the following theorem to find the node T:

Theorem 1 *Give chord AB on the ellipse, we draw any other two chords PQ and RS, AB and PQ intersect at I, AQ and BP intersect at X, AP and BQ intersect at Y, we call XY is the polar line of point I. Also AB and RS intersect at J, AR and BS intersect at M, AS and BR intersect at N, we call MN is the polar line of point J. Then the intersection T of XY and MN is the polar point of chord AB, i.e. TA is the tangent line through A and TB is the tangent line through B.*



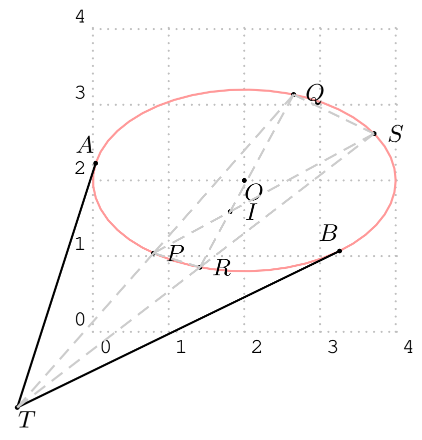
```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](O)(\ra,\rb)
\pstLine[nodesep=-0.8,linecolor=black!40,linestyle=dashed
    ]{0,1}{1.8,3.5}
\pstEllipseLineInter[PosAngle={-90,90}](O)(\ra,\rb)
    {0,1}{1.8,3.5}{A}{B}
\pstEllipsePolarNode[PosAngle=120](O)(\ra,\rb){A}{B}{T}
% Here are the auxiliary lines to explain Theorem 1.
\pstEllipseRotNode[PosAngle=90,RotAngle=5](O)(\ra,\rb){P}
\pstEllipseRotNode[PosAngle=-10,RotAngle=-61](O)(\ra,\rb){Q}
\pstEllipseRotNode[PosAngle=-100,RotAngle=-92](O)(\ra,\rb){R}
\pstEllipseRotNode[PosAngle=0,RotAngle=-30](O)(\ra,\rb){S}
\pstInterLL[PosAngle=-60]{A}{Q}{B}{P}{X}
\pstInterLL[PosAngle=-10]{A}{P}{B}{Q}{Y}
\pstInterLL[PosAngle=0]{A}{R}{B}{S}{M}
\pstInterLL[PosAngle=130]{A}{S}{B}{R}{N}
\psset{linestyle=dashed,linecolor=gray!40}
\pstLine{A}{Q}\pstLine{B}{P}\pstLine{A}{P}\pstLine{B}{Q}
\pstLine{A}{R}\pstLine{B}{S}\pstLine{A}{S}\pstLine{B}{R}
\pstLine{Q}{X}\pstLine{Q}{Y}\pstLine{P}{X}\pstLine{P}{Y}
\pstLine{R}{M}\pstLine{S}{M}\pstLine{T}{Y}\pstLine{T}{N}
\pstLine[linestyle=dashed,linecolor=red!40]{X}{Y}
\pstLine[linestyle=dashed,linecolor=red!40]{M}{N}
\end{pspicture}
```

The macro `\pstEllipseTangentNode` is use to draw the tangent line of a point T out of the Standard Ellipse E. It draw the two tangent lines through the point T to the Standard Ellipse E and get the node A and B on the Ellipse.

`\pstEllipseTangentNode [Options]` $(O)(a,b)\{T\}\{A\}\{B\}$

We use the following theorem to find the tangent node of the given T:

Theorem 2 *Give point T outside of the ellipse, we draw any other two chords TPQ and TRS, let PS and QR intersect at I, PR and QS intersect at X, XI and Ellipse intersect at A and B, then TA is the tangent line through A and TB is the tangent line through B.*



```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](O)(\ra,\rb)
\pstGeonode[PosAngle=-50,PointNameSep=0.2](-1,-1){T}
\pstEllipseTangentNode[PosAngle=120](O)(\ra,\rb){T}{A}{B}
% Here are the auxiliary lines to explain Theorem 2.
\pstEllipseRotNode[PointName=none,RotAngle=71](O)(\ra,\rb){P0}
\pstEllipseRotNode[PointName=none,RotAngle=31](O)(\ra,\rb){R0}
\pstEllipseLineInter[PosAngle=0](O)(\ra,\rb){T}{P0}{P}{Q}
\pstEllipseLineInter[PosAngle=0](O)(\ra,\rb){T}{R0}{R}{S}
\pstInterLL[PosAngle=0]{P}{S}{Q}{R}{I}
\pstInterLL[PosAngle=0]{P}{R}{Q}{S}{X}
\psset{linestyle=dashed,linecolor=gray!40}
\pstLine{T}{P}\pstLine{P}{Q}\pstLine{T}{R}\pstLine{R}{S}
\pstLine{P}{S}\pstLine{Q}{R}\pstLine{P}{R}\pstLine{Q}{S}
\end{pspicture}
```

3.2. General Ellipse

Now we will introduce some macros for the General Ellipse as same as the Standard Ellipse. The General Ellipse E with coordinate translation and rotation is defined by center $O(x_0, y_0)$, the half of the major axis $max(abs(a), abs(b))$, the half of the minor axis $min(abs(a), abs(b))$, and the gradient angle θ of the major axis.

The equation can be got from the parametric function of the ellipse equation (2), using the rotation transform formula:

$$\begin{cases} x' = x\cos\theta - y\sin\theta \\ y' = x\sin\theta + y\cos\theta \end{cases} \tag{3}$$

then we have

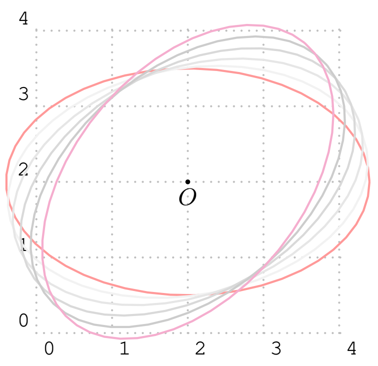
$$\begin{cases} x' = (a\cos\alpha + x_o)\cos\theta - (b\sin\alpha + y_o)\sin\theta = a\cos\alpha\cos\theta - b\sin\alpha\sin\theta + x'_o \\ y' = (a\cos\alpha + x_o)\sin\theta + (b\sin\alpha + y_o)\cos\theta = a\cos\alpha\sin\theta + b\sin\alpha\cos\theta + y'_o \end{cases} \tag{4}$$

where the x'_o and y'_o are the coordinate of the given center O after rotation. So we get the parametric function of the General Ellipse with coordinate translation and rotation as following:

$$\begin{cases} x = a\cos\alpha\cos\theta - b\sin\alpha\sin\theta + x_o \\ y = a\cos\alpha\sin\theta + b\sin\alpha\cos\theta + y_o \end{cases} \tag{5}$$

The Macro `\pstGeneralEllipse` is used to draw a General Ellipse with center O from `angleA` to `angleB`, going counter clockwise. If `angleA` and `angleB` are not specified, the macro will draw the whole ellipse. If you not input gradient angle θ, the default value is $0°$, at this time, the result of this macro is same as `\pstEllipse`. That is, `\pstGeneralEllipse` is more complex than `\pstEllipse`!

\pstGeneralEllipse [Options] (O)(a, b) [θ] [angleA] [angleB]

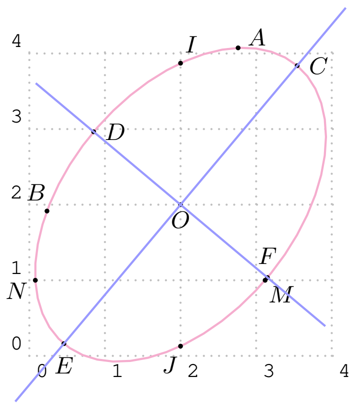


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{-1.5}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=red!40](O)(\ra,\rb)[0]
\pstGeneralEllipse[linecolor=gray!10](O)(\ra,\rb)[10]
\pstGeneralEllipse[linecolor=gray!20](O)(\ra,\rb)[20]
\pstGeneralEllipse[linecolor=gray!30](O)(\ra,\rb)[30]
\pstGeneralEllipse[linecolor=gray!40](O)(\ra,\rb)[40]
\pstGeneralEllipse[linecolor=magenta!40](O)(\ra,\rb)[50]
\end{pspicture}
```

Similarly, we can location the points on the General Ellipse using the macros `\pstGeneralEllipseNode`, `\pstGeneralEllipseRotNode`, `\pstGeneralEllipseAbsNode` and `\pstGeneralEllipseOrdNode` as following.

\pstGeneralEllipseNode [Options] (O)(a, b) [θ] {t}{A}
\pstGeneralEllipseRotNode [Options] (O)(a, b) [θ] {A}
\pstGeneralEllipseAbsNode [Options] (O)(a, b) [θ] {x_1}{A}{B}
\pstGeneralEllipseOrdNode [Options] (O)(a, b) [θ] {y_1}{A}{B}

Some examples all together:

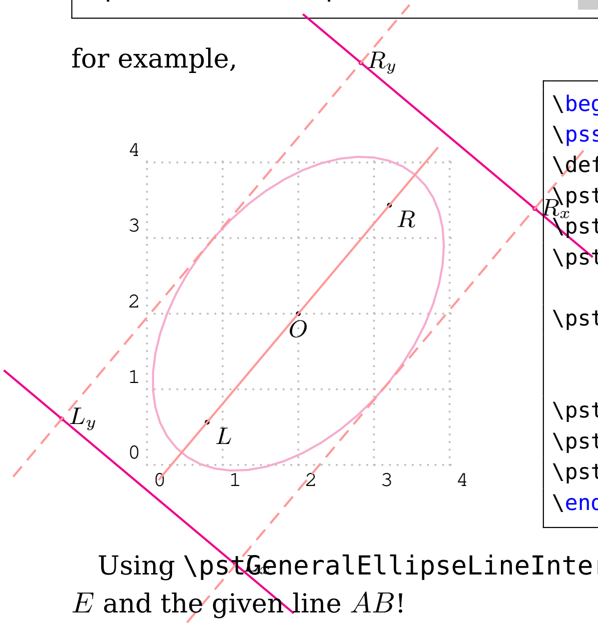


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{-1.5}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=magenta!40](O)(\ra,\rb)[50]
\pstGeneralEllipseNode[PosAngle=30](O)(\ra,\rb)[50]{30}{A}
\pstGeneralEllipseRotNode[PosAngle=120,RotAngle=120](O)(\ra,\rb)
    [50]{B}
\pstGeneralEllipseRotNode[PosAngle=0,RotAngle=0](O)(\ra,\rb)[50]{
    C}
\pstGeneralEllipseRotNode[PosAngle=0,RotAngle=90](O)(\ra,\rb)
    [50]{D}
\pstGeneralEllipseRotNode[PosAngle=-90,RotAngle=180](O)(\ra,\rb)
    [50]{E}
\pstGeneralEllipseRotNode[PosAngle=90,RotAngle=-90](O)(\ra,\rb)
    [50]{F}
\pstGeneralEllipseAbsNode[PosAngle={60,240}](O)(\ra,\rb)[50]{2}{I
    }{J}
\pstGeneralEllipseOrdNode[PosAngle={-40,210}](O)(\ra,\rb)[50]{1}{
    M}{N}
\pstLineAB[nodesep=-1,linecolor=blue!40]{C}{E}
\pstLineAB[nodesep=-1,linecolor=blue!40]{D}{F}
\end{pspicture}
```

Using macro `\pstGeneralEllipseFocusNode` to find the two focus nodes, and macro `\pstGeneralEllipseDirectrixLine` to get the two directrix lines.

`\pstGeneralEllipseFocusNode [Options]` $(O)(a,\,b)$ `[`θ`]` $\{t\}\{A\}$
`\pstGeneralEllipseDirectrixLine [Options]` $(O)(a,\,b)$ `[`θ`]` $\{A\}$

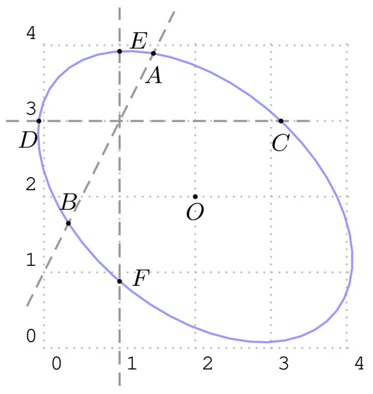
for example,



```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{-1.5}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=magenta!40](O)(\ra,\rb)[50]
\pstGeneralEllipseFocusNode[PosAngle={-40,-40}](O)(\ra,\rb)[50]{L
    }{R}
\pstGeneralEllipseDirectrixLine[PointName={L_x,L_y,R_x,R_y},
    nodesep=-1,linecolor=magenta](O)(\ra,\rb)[50]{Lx}{Ly}{Rx}{Ry
    }
\pstLine[nodesep=-1,linecolor=red!40]{L}{R}
\pstLine[nodesep=-1,linecolor=red!40,linestyle=dashed]{Lx}{Rx}
\pstLine[nodesep=-1,linecolor=red!40,linestyle=dashed]{Ly}{Ry}
\end{pspicture}
```

Using `\pstGeneralEllipseLineInter` to get the two intersections C and D of the General Ellipse E and the given line AB!

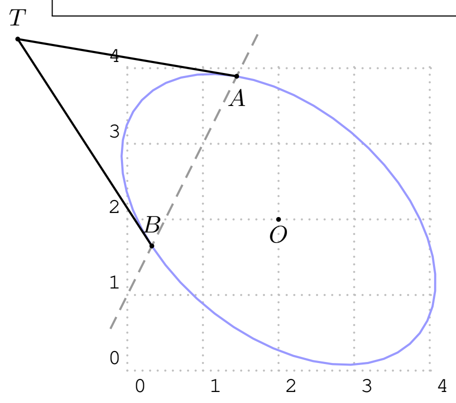
`\pstGeneralEllipseLineInter [Options]` $(O)(a,\,b)$ `[`θ`]` $\{A\}\{B\}\{C\}\{D\}$



```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.5}\def\rb{-2.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=blue!40](O)(\ra,\rb)[50]
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed
    ]{0,1}{1.5,4}
\pstGeneralEllipseLineInter[PosAngle={-90,90}](O)(\ra,\rb)
    [50]{0,1}{1.5,4}{A}{B}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed
    ]{0,3}{3,3}
\pstGeneralEllipseLineInter[PosAngle={-90,240}](O)(\ra,\rb)
    [50]{0,3}{3,3}{C}{D}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed
    ]{1,0}{1,4}
\pstGeneralEllipseLineInter[PosAngle={30,10}](O)(\ra,\rb)
    [50]{1,1}{1,4}{E}{F}
\end{pspicture}
```

Using `\pstGeneralEllipsePolarNode` to find the polar point T of chord AB, please refer to Theorem 1.

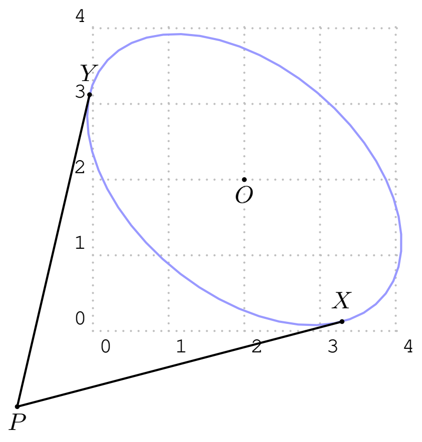
`\pstGeneralEllipsePolarNode [Options] (`O`)(`a, b`) [`θ`] {`A`}{`B`}{`T`}`



```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.5}\def\rb{-2.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=blue!40](O)(\ra,\rb)[50]
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed
    ]{0,1}{1.5,4}
\pstGeneralEllipseLineInter[PosAngle={-90,90}](O)(\ra,\rb)
    [50]{0,1}{1.5,4}{A}{B}
\pstGeneralEllipsePolarNode[PosAngle=90](O)(\ra,\rb)[50]{A}{B}{T}
\end{pspicture}
```

Using `\pstGeneralEllipseTangentNode` to find the tangent point A and B of outside point T, please refer to Theorem 2.

`\pstGeneralEllipseTangentNode [Options] (`O`)(`a, b`) [`θ`] {`T`}{`A`}{`B`}`



```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.5}\def\rb{-2.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=blue!40](O)(\ra,\rb)[50]
\pstGeonode[PosAngle=-90,PointNameSep=0.2](-1,-1){P}
\pstGeneralEllipseTangentNode[PosAngle=90](O)(\ra,\rb)[50]{P}{X}{
    Y}
\end{pspicture}
```

3.3. Standard Parabola

The Standard Parabola P with coordinate translation is defined by vertex $O(x_0, y_0)$, the half of the focus chord axis $abs(p)$. Note that the sign of p indicates the direction of the parabola.

The equation can be written as:

$$(x - x0)^2 = 2p(y - y0) \tag{6}$$

and the parametric function can be written as:

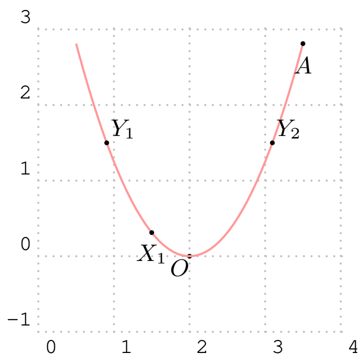
$$\begin{cases} x = t + x_o \\ y = \dfrac{t^2}{2p} + y_o \end{cases} \tag{7}$$

The macro `\pstParabola` is used to draw a Parabola from x_1 to x_2 with Vertex O, the half of the focus chord axis $abs(p)$.

```
\pstParabola [Options] (O)(a, b){x1}{x2}{p}
```

The macro `\pstParabolaNode` is used to draw a node whose parameter is the given value t on parabola, please refer to equation (7). The macro `\pstParabolaAbsNode` is used to draw a node whose abscissa is the given value x_1 on parabola. The macro `\pstParabolaOrdNode` is used to draw a node whose ordinate is the given value y_1 on parabola. Note that `\pstParabolaOrdNode` will create two nodes A and B at most time.

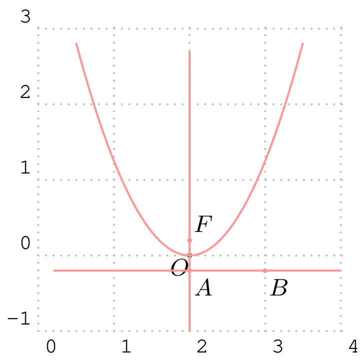
```
\pstParabolaNode [Options] (O)(a, b){p}{t}{A}
\pstParabolaAbsNode [Options] (O)(a, b){p}{x1}{A}
\pstParabolaOrdNode [Options] (O)(a, b){p}{y1}{A}{B}
```



```
\begin{pspicture}[showgrid=true](0,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\semifocalchord{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){-1.5}{1.5}{\semifocalchord}
\pstParabolaNode[PosAngle=-90](O){\semifocalchord}{1.5}{A}
\pstParabolaAbsNode[PosAngle=-90,PointName=X_1](O){\
    semifocalchord}{1.5}{X1}
\pstParabolaOrdNode[PosAngle=40,PointName={Y_1,Y_2}](O){\
    semifocalchord}{1.5}{Y1}{Y2}
\end{pspicture}
```

The macro `\pstParabolaFocusNode` is used to find the focus of the parabola, and the macro `\pstParabolaDirectrixLine` is used to find the directrix line of the parabola.

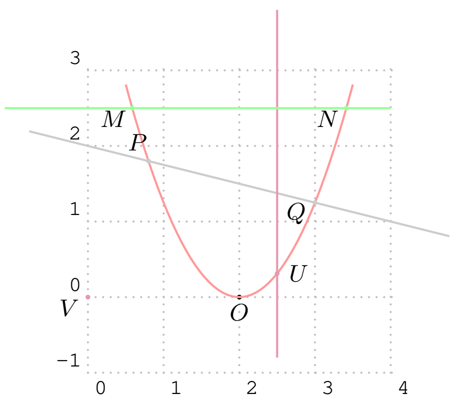
```
\pstParabolaFocusNode [Options] (O)(a, b){p}{F}
\pstParabolaDirectrixLine [Options] (O)(a, b){p}{Lx}{Ly}
```



```
\begin{pspicture}[showgrid=true](0,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\semifocalchord{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){-1.5}{1.5}{\semifocalchord}
\pstParabolaFocusNode[linecolor=red!40,PosAngle=50](O){\
    semifocalchord}{F}
\pstParabolaDirectrixLine[linecolor=red!40,nodesepA=-1.8,nodesepB
    =-1,PosAngle={-50,-50}](O){\semifocalchord}{A}{B}
\pstLine[linecolor=red!40,nodesepA=-0.8,nodesepB=-2.5]{A}{F}
\end{pspicture}
```

The macro `\pstParabolaLineInter` is used to find the intersections C and D of the parabola and the given line AB.

`\pstParabolaLineInter` `[Options]` $(O)(a,b)\{p\}\{A\}\{B\}\{C\}\{D\}$



```
\begin{pspicture}[showgrid=true](0,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\semifocalchord{0.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){-1.5}{1.5}{\semifocalchord}
\pstLine[linecolor=gray!40,nodesepA=-0.8,nodesepB=-0.8]{0,2}{4,1}
\pstParabolaLineInter[linecolor=gray!40,PosAngle={120,210}](O){\
    semifocalchord}{0,2}{4,1}{P}{Q}
\pstLine[linecolor=purple!40,nodesepA=-0.8,nodesepB
    =-0.8]{2.5,0}{2.5,3}
\pstParabolaLineInter[linecolor=purple!40,PosAngle={0,210}](O){\
    semifocalchord}{2.5,0}{2.5,3}{U}{V}
\pstLine[linecolor=green!40,nodesepA=-2.5,nodesepB
    =-1.6]{1.5,2.5}{0.5,2.5}
\pstParabolaLineInter[linecolor=green!40,PosAngle={210,210}](O){\
    semifocalchord}{1.5,2.5}{0.5,2.5}{M}{N}
\end{pspicture}
```

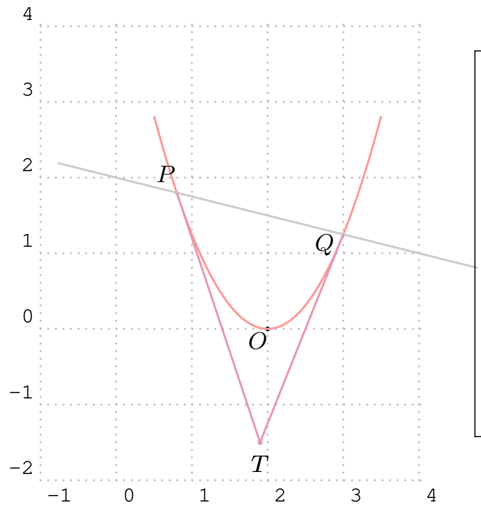
The macro `\pstParabolaPolarNode` is used to find the polar point T of chord AB on Parabola P.

`\pstParabolaPolarNode` `[Options]` $(O)(a,b)\{p\}\{A\}\{B\}\{T\}$
`\pstParabolaPolarNode` `[Options]` $(O)(a,b)\{p\}(F)\{A\}\{B\}\{T\}$
`\pstParabolaPolarNode` `[Options]` $(O)(a,b)\{p\}(F)$ $[L_x]$ $[L_y]$ $\{A\}\{B\}\{T\}$

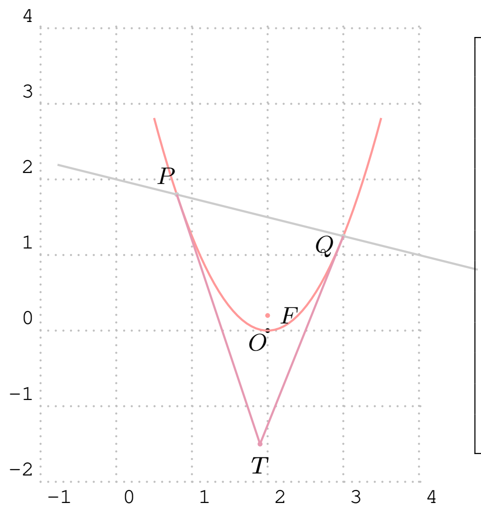
We use the following propositton to find the polar point T of chord AB:

Theorem 3 *Give any chord AB on parabola, drawing two focal chord AFC and BFD, where F is the focus of parabola, then drawing FX which is perpendicular to AFC at point F, and intersect with the directrix line at X; also drawing FY which is perpendicular to BFD at point F, and intersect with the directrix line at Y. Then the intersection T of AX and BY is the polar point of chord AB.*

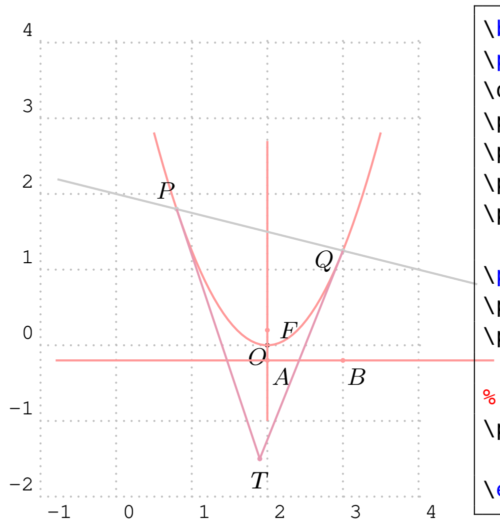
If you don't know the focus F, or the directrix line, we will find them automated, otherwise you can pass them to this macro.



```
\begin{pspicture}[showgrid=true](-1,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\semifocalchord{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){-1.5}{1.5}{\semifocalchord}
\pstLine[linecolor=gray!40,nodesepA=-0.8,nodesepB=-0.8]{0,2}{4,1}
\pstParabolaLineInter[linecolor=gray!40,PosAngle={120,210}](O){\
    semifocalchord}{0,2}{4,1}{P}{Q}
% if you don't know focus F or directrix line
\pstParabolaPolarNode[linecolor=purple!40,PosAngle=-90](O){\
    semifocalchord}{P}{Q}{T}
\end{pspicture}
```



```
\begin{pspicture}[showgrid=true](-1,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\semifocalchord{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){-1.5}{1.5}{\semifocalchord}
\pstParabolaFocusNode[linecolor=red!40](O){\semifocalchord}{F}
\pstLine[linecolor=gray!40,nodesepA=-0.8,nodesepB=-0.8]{0,2}{4,1}
\pstParabolaLineInter[linecolor=gray!40,PosAngle={120,210}](O){\
    semifocalchord}{0,2}{4,1}{P}{Q}
% if you know focus F, but don't known directrix line
\pstParabolaPolarNode[linecolor=purple!40,PosAngle=-90](O){\
    semifocalchord}(F){P}{Q}{T}
\end{pspicture}
```



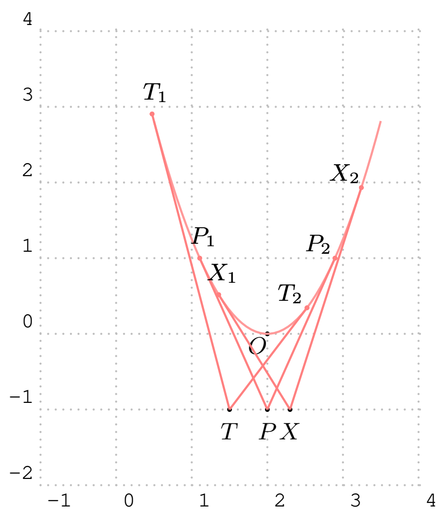
```
\begin{pspicture}[showgrid=true](-1,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\semifocalchord{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){-1.5}{1.5}{\semifocalchord}
\pstParabolaFocusNode[linecolor=red!40](O){\semifocalchord}{F}
\pstParabolaDirectrixLine[linecolor=red!40,nodesepA=-2.8,nodesepB
    =-2,PosAngle={-50,-50}](O){\semifocalchord}{A}{B}
\pstLineAB[linecolor=red!40,nodesepA=-0.8,nodesepB=-2.5]{A}{F}
\pstLine[linecolor=gray!40,nodesepA=-0.8,nodesepB=-0.8]{0,2}{4,1}
\pstParabolaLineInter[linecolor=gray!40,PosAngle={120,210}](O){\
    semifocalchord}{0,2}{4,1}{P}{Q}
% if you know focus F and also directrix line
\pstParabolaPolarNode[linecolor=purple!40,PosAngle=-90](O){\
    semifocalchord}(F)[A][B]{P}{Q}{T}
\end{pspicture}
```

The macro `\pstParabolaTangentNode` is used to find the two nodes A and B on the Parabola through the point T.

`\pstParabolaTangentNode [Options]` $(O)(a,b)\{p\}\{T\}\{A\}\{B\}$

We use the following proposition to find the tangent node A and B of outside point T:

Theorem 4 *Give point T outside of the parabola, we draw any other two chords TPQ and TRS, PS and QR intersect at I, PR and QS intersect at X, XI and Parabola intersect at A and B, then TA is the tangent line through A and TB is the tangent line through B.*



```
\begin{pspicture}[showgrid=true](-1,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\semifocalchord{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){-1.5}{1.5}{\semifocalchord}
\pstGeonode[PosAngle=-90](1.5,-1){T}
\pstParabolaTangentNode[linecolor=red!50,PosAngle={80,140},
    PointName={T_1,T_2}](O){\semifocalchord}{T}{T1}{T2}
\pstGeonode[PosAngle=-90](2,-1){P}
\pstParabolaTangentNode[linecolor=red!50,PosAngle={80,140},
    PointName={P_1,P_2}](O){\semifocalchord}{P}{P1}{P2}
\pstGeonode[PosAngle=-90](2.3,-1){X}
\pstParabolaTangentNode[linecolor=red!50,PosAngle={80,140},
    PointName={X_1,X_2}](O){\semifocalchord}{X}{X1}{X2}
\end{pspicture}
```

3.4. Standard Inversion Parabola

The Inversion Parabola P with coordinate translation is defined by vertex $O(x_0, y_0)$, the half of the focus chord axis $abs(p)$. Note that the sign of p indicates the direction of the parabola. The equation can be written as:

$$(y - y0)^2 = 2p(x - x0) \tag{8}$$

and the parametric function can be written as:

$$\begin{cases} x = \dfrac{t^2}{2p} + x_o \\ y = t + y_o \end{cases} \tag{9}$$

3.5. General Parabola

The General Parabola P with coordinate translation and rotation is defined by vertex $O(x_0, y_0)$, the half of the focus chord axis $abs(p)$, the sign of p indicates the direction of the parabola, and the gradient angle θ of the symmetrical axis.

The equation can be got from the parametric function of the parabola equation (7), using the rotation transform formula (3), then we have

$$\begin{cases} x' = (t + x_o)\cos\theta - (t^2/(2p) + y_o)\sin\theta = x'_o + t\cos\theta - t^2/(2p)\sin\theta \\ y' = (t + x_o)\sin\theta + (t^2/(2p) + y_o)\cos\theta = y'_o + t\sin\theta + t^2/(2p)\cos\theta \end{cases} \tag{10}$$

where the x'_o and y'_o are the coordinate of the given vertex O after rotation. So we get the parametric function of the General Parabola with coordinate translation and rotation as following:

$$\begin{cases} x = x_o + t\cos\theta - t^2/(2p)\sin\theta \\ y = y_o + t\sin\theta + t^2/(2p)\cos\theta \end{cases} \tag{11}$$

3.6. General Inversion Parabola with coordinate translation and rotation

3.7. Standard Hyperbola with coordinate translation

3.8. Standard Inversion Hyperbola with coordinate translation

3.9. General Hyperbola with coordinate translation and rotation

3.10. General Inversion Hyperbola with coordinate translation and rotation

4. Geometric Transformations

The geometric transformations are the ideal tools to construct geometric figures. All the classical transformations are available with the following macros which share the same syntaxic scheme end two parameters.

The common syntax put at the end two point lists whose second is optional or with a cardinal at least equal. These two lists contain the antecedent points and their respective images. In the case no image is given for some points the a default name is build appending a `'` to the antecedent name.

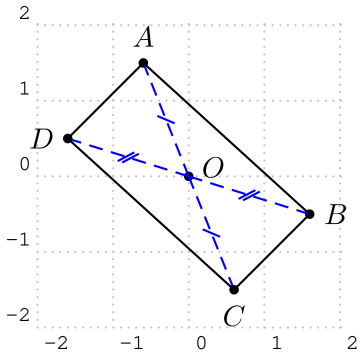
The first shared parameter is `CodeFig` which draws the specific constructions lines. Its default value is `false`, and a `true` value activates this optional drawing. The drawing is done using the line style `CodeFigStyle` (by default dashed), with the color `CodeFigColor` (by default cyan).

Their second shared parameter is `CurveType` which controls the drawing of a line crossing all images, and thus allow a quick description of a transformed figure.

4.1. Central symmetry

`\pstSymO [Options]` $\{O\}\{M_1, M_2, \cdots, M_n\}$ $[M'_1, M'_2, \cdots, M'_p]$

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, `PtNameMath`, `CodeFig`, `CodeFigColor`, and `CodeFigStyle`. Draw the symmetric point in relation to point O. The classical parameter of point creation are usable here, and also for all the following functions.

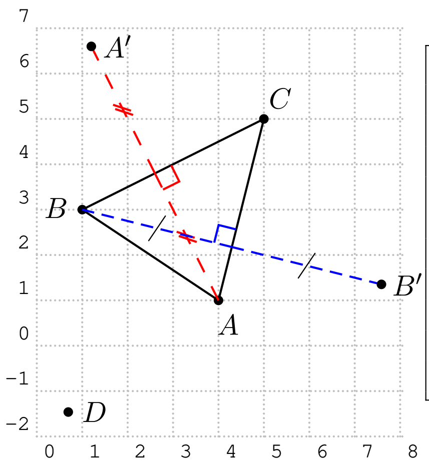


```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psset{CodeFig=true}
\pstGeonode[PosAngle={20,90,0}]{O}(-.6,1.5){A}(1.6,-.5){B}
\pstSymO[CodeFigColor=blue,
  PosAngle={-90,180}]{O}{A, B}[C, D]
\pstLineAB{A}{B}\pstLineAB{C}{D}
\pstLineAB{A}{D}\pstLineAB{C}{B}
\end{pspicture}
```

4.2. Orthogonal (or axial) symmetry

`\pstOrtSym [Options]` $\{A\}\{B\}\{M_1, M_2, \cdots, M_n\}$ $[M'_1, M'_2, \cdots, M'_p]$

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, `PtNameMath`, `CodeFig`, `CodeFigColor`, and `CodeFigStyle`. Draws the symmetric point in relation to line (AB).



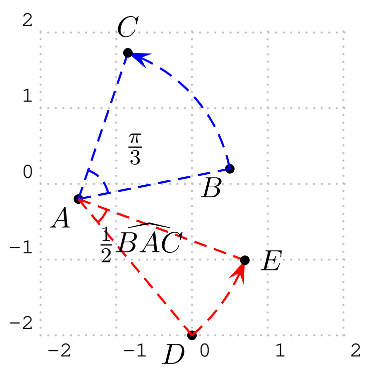
```
\psset{unit=0.6}
\begin{pspicture}[showgrid](0,-2)(8,7)
\pstTriangle(1,3){B}(5,5){C}(4,1){A}
\pstOrtSym{A}{B}{C}[D]
\psset{CodeFig=true}
\pstOrtSym[dash=2mm 2mm,CodeFigColor=red]%
  {C}{B}{A}
\pstOrtSym[SegmentSymbol=pstslash,
  linestyle=dotted,dotsep=3mm,CodeFigColor=blue]%
  {C}{A}{B}
\end{pspicture}
```

4.3. Rotation

`\pstRotation [Options] {`O`}{`$M_1, M_2, \cdots, M_n$`} [`$M'_1, M'_2, \cdots, M'_p$`]`
`\pstAngleAOB{`A`}{`O`}{`B`}`

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, `PtNameMath`, and `RotAngle` for `\pstRotation` and `AngleCoef`, `RotAngle` for `\pstAngleABC`. Draw the image of M_i by the rotation of center O and angle given by the parameter `RotAngle`. This later can be an angle specified by three points. In such a case, the following function must be used:

Never forget to use the rotation for drawing a square or an equilateral triangle. The parameter `CodeFig` puts a bow with an arrow between the point and its image, and if `TransformLabel` (by default none) contain some text, it is put on the corresponding angle in mathematical mode.



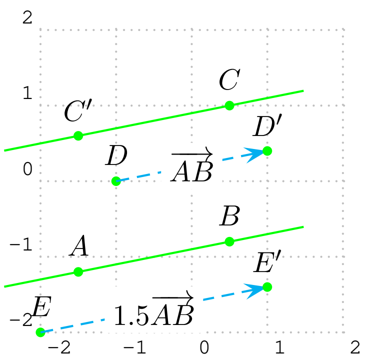
```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psset{arrowscale=2}
\pstGeonode[PosAngle=-135](-1.5,-.2){A}%
  (.5,.2){B}(0,-2){D}
\pstRotation[PosAngle=90,RotAngle=60,
  CodeFig,CodeFigColor=blue,
  TransformLabel=\frac{\pi}{3}]{A}{B}[C]
\pstRotation[AngleCoef=.5,
  RotAngle=\pstAngleAOB{B}{A}{C},
  CodeFigColor=red, CodeFig,
  TransformLabel=\frac{1}{2}\widehat{BAC}]{A}{D}[E]
\end{pspicture}
```

4.4. Translation

`\pstTranslation [Options] {`A`}{`B`}{`$M_1, M_2, \cdots, M_n$`} [`$M'_1, M'_2, \cdots, M'_p$`]`

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, `PtNameMath`, and `DistCoef` Draws the translated M'_i of M_i using the vector $\vec{AB}$. Useful for drawing a parallel line.

The parameter `DistCoef` can be used as a multiplicand coefficient to modify the translation vector. The parameter `CodeFig` draws the translation vector le vecteur de translation between the point and its image, labeled in its middle defaultly with the vector name or by the text specified with `TransformLabel` (by default none).

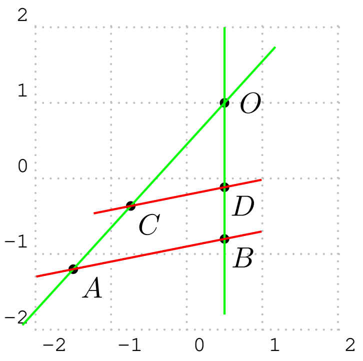


```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psset{linecolor=green,nodesep=-1,
 PosAngle=90,arrowscale=2}
\pstGeonode(-1.5,-1.2){A}(.5,-.8){B}(.5,1){C}(-1,0){D}(-2,-2){E}
\pstTranslation{B}{A}{C}
\psset{CodeFig,TransformLabel=default}
\pstTranslation{A}{B}{D}
\pstTranslation[DistCoef=1.5]{A}{B}{E}
\pstLineAB{A}{B}\pstLineAB{C}{C'}
\end{pspicture}
```

4.5. Homothetie

\pstHomO [Options] $\{O\}\{M_1, M_2, \cdots, M_n\}$ $[M'_1, M'_2, \cdots, M'_p]$

Possible optional arguments are HomCoef, PointSymbol, PosAngle, PointName, PointNameSep, PtNameMath, and HomCoef. Draws M'_i the image of M_i by the homotethy of center O and coefficient specified with the parameter HomCoef.

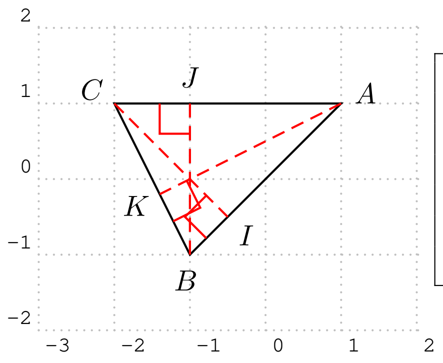


```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\pstGeonode[PosAngle={0,-45}](.5,1){O}%
     (-1.5,-1.2){A}(.5,-.8){B}
\pstHomO[HomCoef=.62,PosAngle=-45]{O}{A,B}[C,D]
\psset{linecolor=green,nodesep=-1}
\pstLineAB{A}{O}\pstLineAB{B}{O}
\psset{linecolor=red,nodesep=-.5}
\pstLineAB{A}{B}\pstLineAB{C}{D}
\end{pspicture}
```

4.6. Orthogonal projection

\pstProjection [Options] $\{A\}\{B\}\{M_1, M_2, \cdots, M_n\}$ $[M'_1, M'_2, \cdots, M'_p]$

Possible optional arguments are PointSymbol, PosAngle, PointName, PointNameSep, PtNameMath, CodeFig, CodeFigColor, andCodeFigStyle Projects orthogonally the point M_i on the line (AB). Useful for the altitude of a triangle. The name is aligned with the point and the projected point as shown in the exemple.



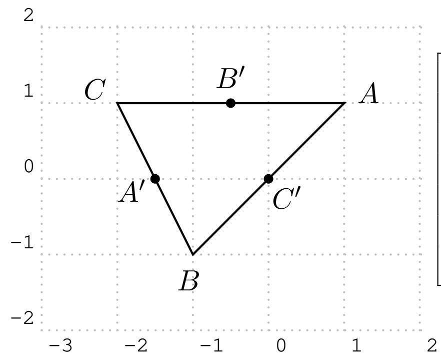
```
\begin{pspicture}[showgrid](-3,-2)(2,2)
\psset{PointSymbol=none,CodeFig,CodeFigColor=red}
\pstTriangle(1,1){A}(-2,1){C}(-1,-1){B}
\pstProjection{A}{B}{C}[I]
\pstProjection{A}{C}{B}[J]
\pstProjection{C}{B}{A}[K]
\end{pspicture}
```

5. Special object

5.1. Midpoint

```
\pstMiddleAB [Options] {A}{B}{I}
```

`PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, `PtNameMath`, `SegmentSymbol`, `CodeFig`, `CodeFigColor`, and `CodeFigStyle` Draw the midpoint I of segment $[AB]$. By default, the point name is automatically put below the segment.

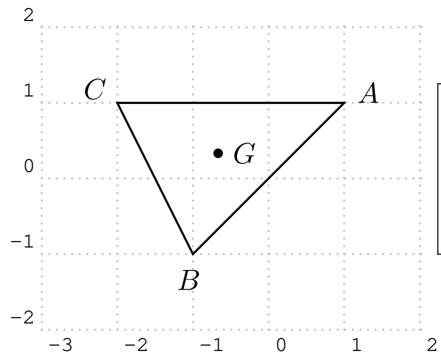


```
\begin{pspicture}[showgrid](-3,-2)(2,2)
\pstTriangle[PointSymbol=none]%
  (1,1){A}(-1,-1){B}(-2,1){C}
\pstMiddleAB{A}{B}{C'}
\pstMiddleAB{C}{A}{B'}
\pstMiddleAB{B}{C}{A'}
\end{pspicture}
```

5.2. Triangle center of gravity

```
\pstCGravABC [Options] {A}{B}{C}{G}
```

Possible optional arguments are `PointName`, `PointNameSep`, `PosAngle`, `PointSymbol`, and `PtNameMath` Draw the ABC triangle centre of gravity G.

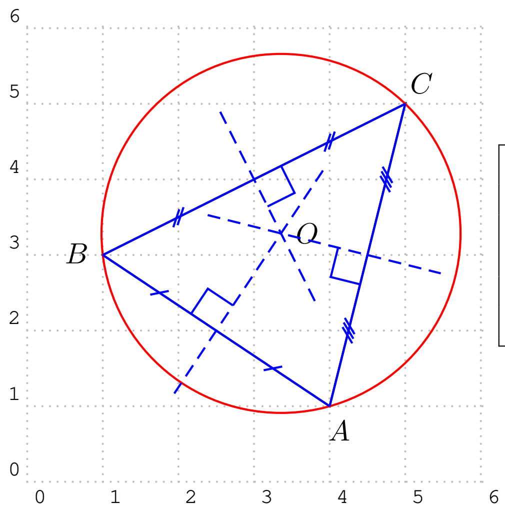


```
\begin{pspicture}[showgrid](-3,-2)(2,2)
\pstTriangle[PointSymbol=none]%
  (1,1){A}(-1,-1){B}(-2,1){C}
\pstCGravABC{A}{B}{C}{G}
\end{pspicture}
```

5.3. Centre of the circumcircle of a triangle

```
\pstCircleABC [Options] {A}{B}{C}{O}
```

Possible optional arguments are `PointName`, `PointNameSep`, `PosAngle`, `PointSymbol`, `PtNameMath`, `DrawCirABC`, `CodeFig`, `CodeFigColor`, `CodeFigStyle`, `SegmentSymbolA`, `SegmentSymbolB`, and `SegmentSymbolC`. Draws the circle crossing three points (the circum circle) and put its center O. The effective drawing is controlled by the boolean parameter `DrawCirABC` (by default true). Moreover the intermediate constructs (mediator lines) can be drawn by setting the boolean parameter `CodeFig`. In that case the middle points are marked on the segemnts using three different marks given by the parameters `SegmentSymbolA`, `SegmentSymbolB` et `SegmentSymbolC`.



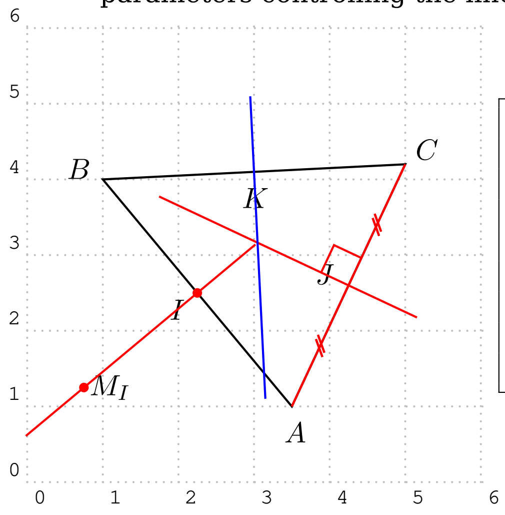
```
\begin{pspicture}[showgrid](6,6)
\pstTriangle[PointSymbol=none]%
  (4,1){A}(1,3){B}(5,5){C}
\pstCircleABC[CodeFig,CodeFigColor=blue,
  linecolor=red,PointSymbol=none]{A}{B}{C}{O}
\end{pspicture}
```

5.4. Perpendicular bisector of a segment

```
\pstMediatorAB [Options] {A}{B}{I}{M}
```

Possible optional arguments are `PointName`, `PointNameSep`, `PosAngle`, `PointSymbol`, `PtNameMath`, `CodeFig`, `CodeFigColor`, `CodeFigStyle`, and `SegmentSymbol`. The perpendicular bisector of a segment is a line perpendicular to this segment in its midpoint. The segment is $[AB]$, the midpoint I, and M is a point belonging to the perpendicular bisector line. It is build by a rotation of B of 90 degrees around I. This mean that the order of A and B is important, it controls the position of M. The command creates the two points M end I. The construction is controlled by the following parameters:

- `CodeFig`, `CodeFigColor` and `SegmentSymbol` for marking the right angle ;
- `PointSymbol` et `PointName` for controlling the drawing of the two points, each of them can be specified separately with the parameters `...A` and `...B` ;
- parameters controlling the line drawing.

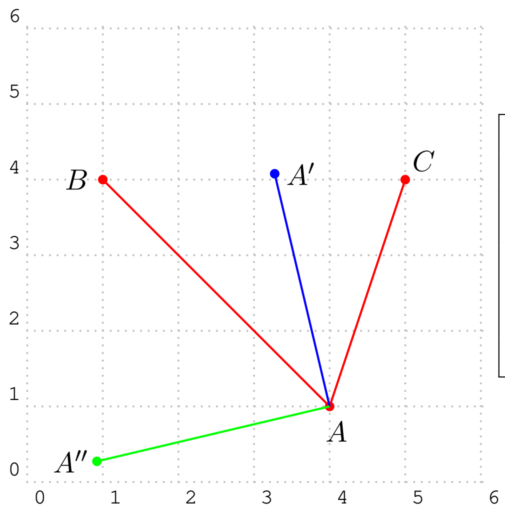


```
\begin{pspicture}[showgrid](6,6)
\pstTriangle[PointSymbol=none](3.5,1){A}(1,4){B}(5,4.2){C}
\psset{linecolor=red,CodeFigColor=red,nodesep=-1}
\pstMediatorAB[PointSymbolA=none]{A}{B}{I}{M_I}
\psset{PointSymbol=none,PointNameB=none}
\pstMediatorAB[CodeFig=true]{A}{C}{J}{M_J}
\pstMediatorAB[PosAngleA=45,linecolor=blue]
      {C}{B}{K}{M_K}
\end{pspicture}
```

5.5. Bisectors of angles

```
\pstBissectBAC [Options] {B}{A}{C}{N}
\pstOutBissectBAC [Options] {B}{A}{C}{N}
```

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, and `PtNameMath`. There are two bisectors for a given geometric angle: the inside one and the outside one; this is why there is two commands. The angle is specified by three points specified in the trigonometric direction (anti-clockwise). The result of the commands is the specific line and a point belonging to this line. This point is built by a rotation of point B.



```
\begin{pspicture}[showgrid](6,6)
\psset{CurveType=polyline,linecolor=red}
\pstGeonode[PosAngle={180,-75,45}]%
  (1,4){B}(4,1){A}(5,4){C}
\pstBissectBAC[linecolor=blue]{C}{A}{B}{A’}
\pstOutBissectBAC[linecolor=green,PosAngle=180]%
  {C}{A}{B}{A’’}
\end{pspicture}
```

6. Intersections

Points can be defined by intersections. Six intersection types are managed:

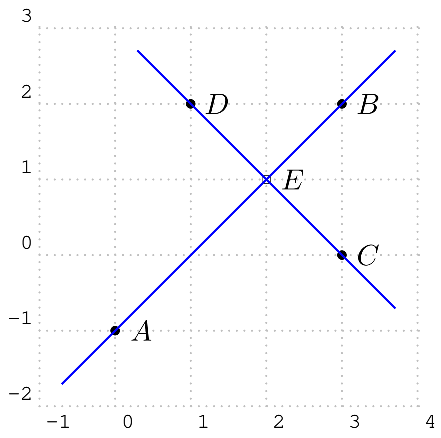
- line-line;
- line-circle;
- circle-circle;
- function-function;
- function-line;
- function-circle.

An intersection can not exist: case of parallel lines. In such a case, the point(s) are positioned at the origin. In fact, the user has to manage the existence of these points.

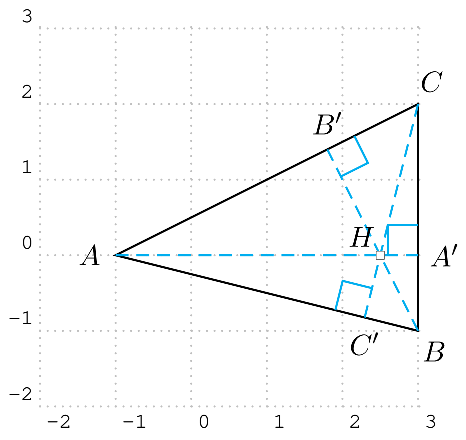
6.1. Line-Line

```
\pstInterLL [Options] {A}{B}{C}{D}{M}
```

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, and `PtNameMath`. Draw the intersection point between lines (AB) and (CD).



```
\begin{pspicture}[showgrid](-1,-2)(4,3)
\pstGeonode(0,-1){A}(3,2){B}(3,0){C}(1,2){D}
\pstInterLL[PointSymbol=square]{A}{B}{C}{D}{E}
\psset{linecolor=blue, nodesep=-1}
\pstLineAB{A}{B}\pstLineAB{C}{D}
\end{pspicture}
```



```
\begin{pspicture}[showgrid](-2,-2)(3,3)
\psset{CodeFig,PointSymbol=none}
\pstTriangle[PosAngleA=180](-1,0){A}(3,-1){B}(3,2){C}
\pstProjection[PosAngle=-90]{B}{A}{C}
\pstProjection{B}{C}{A}
\pstProjection[PosAngle=90]{A}{C}{B}
\pstInterLL[PosAngle=135,PointSymbol=square]{A}{A'}{B}{B'}{
    H}
\end{pspicture}
```

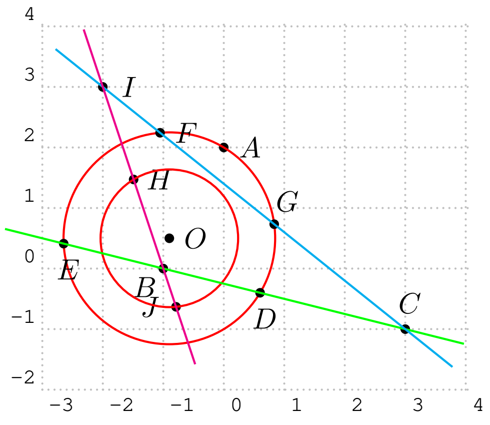
6.2. Circle–Line

\pstInterLC [Options] $\{A\}\{B\}\{O\}\{C\}\{M_1\}\{M_2\}$

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, `PtNameMath`, `PointSymbolA`, `PosAngleA`, `PointNameA`, `PointSymbolB`, `PosAngleB`, `PointNameB`, `Radius`, and `Diameter`. Draw the one or two intersection point(s) between the line (AB) and the circle of centre O and with radius OC.

The circle is specified with its center and either a point of its circumference or with a radius specified with parameter `radius` or its diameter specified with parameter `Diameter`. These two parameters can be modify by coefficient `DistCoef`.

The position of the wo points is such that the vectors $\vec{AB}$ abd $M_1\vec{M}_2$ are in the same direction. Thus, if the points definig the line are switch, then the resulting points will be also switched. If the intersection is void, then the points are positionned at the center of the circle.

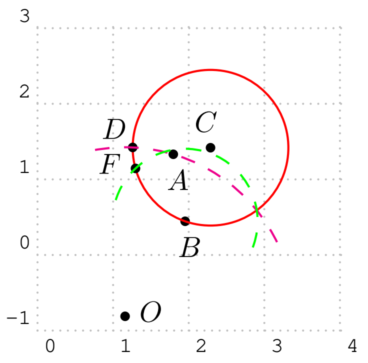


```
\psset{unit=0.8}
\begin{pspicture}[showgrid](-3,-2)(4,4)
\pstGeonode[PosAngle={-135,80,0}](-1,0){B}(3,-1){C}(-.9,.5){O
    }(0,2){A}
\pstGeonode(-2,3){I}
\pstCircleOA[linecolor=red]{O}{A}
\pstInterLC[PosAngle=-80]{C}{B}{O}{A}{D}{E}
\pstInterLC[PosAngleB=60, Radius=\pstDistAB{O}{D}]{I}{C}{O}{}{F}{
    G}
\pstInterLC[PosAngleB=180,DistCoef=1.3,Diameter=\pstDistAB{O}{D}]
  {I}{B}{O}{}{H}{J}
\pstCircleOA[linecolor=red,DistCoef=1.3,Diameter=\pstDistAB{O}{D
    }]{O}{}
\psset{nodesep=-1}
\pstLineAB[linecolor=green]{E}{C}
\pstLineAB[linecolor=cyan]{I}{C}
\pstLineAB[linecolor=magenta]{J}{I}
\end{pspicture}
```

6.3. Circle–Circle

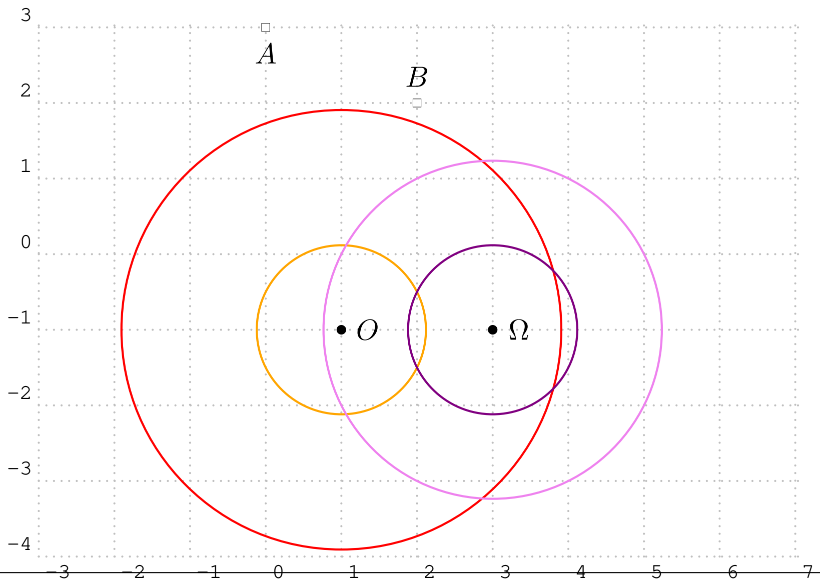
\pstInterCC [Options] $\{O_1\}\{B\}\{O_2\}\{C\}\{M_1\}\{M_2\}$

This function is similar to the last one. The boolean parameters `CodeFigA` et `CodeFigB` allow the drawing of the arcs at the intersection. In order to get a coherence `CodeFig` allow the drawing of both arcs. The boolean parameters `CodeFigAarc` and `CodeFigBarc` specified the direction of these optional arcs: trigonometric (by default) or clockwise. Here is a first example.



```
\begin{pspicture}[showgrid](0,-1)(4,3)
\psset{dash=2mm 2mm}
\rput{10}{%
 \pstGeonode[PosAngle={0,-90,-90,90}]
   (1,-1){O}(2,1){A}(2,0.1){B}(2.5,1){C}}
\pstCircleOA[linecolor=red]{C}{B}
\pstInterCC[PosAngleA=135, CodeFigA=true, CodeFigAarc=false,
 CodeFigColor=magenta]{O}{A}{C}{B}{D}{E}
\pstInterCC[PosAngleA=170, CodeFigA=true,
  CodeFigAarc=false,
  CodeFigColor=green]{B}{E}{C}{B}{F}{G}
\end{pspicture}
```

And a more complete one, which includes the special circle specification using radius and diameter. For such specifications it exists the parameters `RadiusA`, `RadiusB`, `DiameterA` and `DiameterB`.



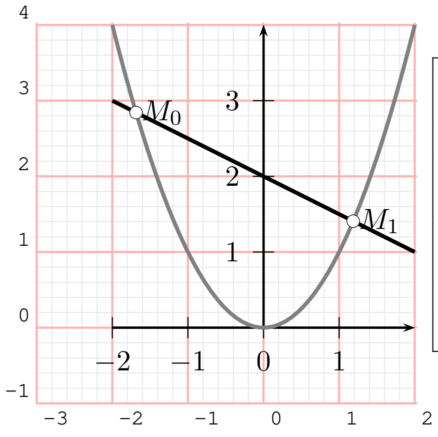
```
\begin{pspicture}[showgrid](-3,-4)(7,3)
\pstGeonode[PointName={\Omega,O}](3,-1){Omega}(1,-1){O}
\pstGeonode[PointSymbol=square, PosAngle={-90,90}](0,3){A}(2,2){B}
\psset{PointSymbol=o}
\pstCircleOA[linecolor=red, DistCoef=1 3 10 div add, Radius=\pstDistAB{A}{B}]{O}{}
\pstCircleOA[linecolor=Orange, Diameter=\pstDistAB{A}{B}]{O}{}
\pstCircleOA[linecolor=Violet, Radius=\pstDistAB{A}{B}]{Omega}{}
\pstCircleOA[linecolor=Purple, Diameter=\pstDistAB{A}{B}]{Omega}{}
\pstInterCC[DistCoef=1 3 10 div add, RadiusA=\pstDistAB{A}{B},
        DistCoef=none, RadiusB=\pstDistAB{A}{B}]{O}{}{Omega}{}{D}{E}
\pstInterCC[DiameterA=\pstDistAB{A}{B}, RadiusB=\pstDistAB{A}{B}]{O}{}{Omega}{}{F}{G}
\pstInterCC[DistCoef=1 3 10 div add, RadiusA=\pstDistAB{A}{B},
        DistCoef=none, DiameterB=\pstDistAB{A}{B}]{O}{}{Omega}{}{H}{I}
\pstInterCC[DiameterA=\pstDistAB{A}{B}, DiameterB=\pstDistAB{A}{B}]{O}{}{Omega}{}{J}{K}
\end{pspicture}
```

6.4. Function–function

`\pstInterFF [Options]` $\{f\}\{g\}\{x_0\}\{M\}$

This function put a point at the intersection between two curves defined by a function. x_0 is an intersection approximated value of the abscissa. It is obviously possible to ise this function several time if more than one intersection is present. Each function is describerd in `PostScript`in the same way as the description used by the \psplot macro of PSTricks. A constant function can be specified, and then seaching function root is possible.

The Newton algorithm is used for the research, and the intersection may not to be found. In such a case the point is positionned at the origin. On the other hand, the research can be trapped (in a local extremum near zero).

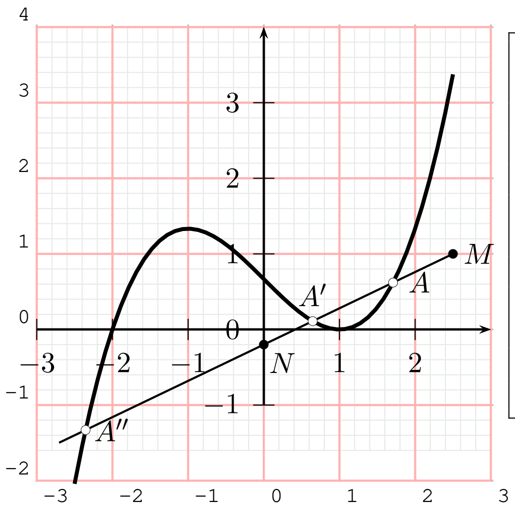


```
\begin{pspicture}[showgrid](-3,-1)(2,4)
\psaxes{->}(0,0)(-2,0)(2,4)
\psset{linewidth=1.5pt,algebraic}
\psplot[linecolor=gray]{-2}{2}{x^2}
\psplot{-2}{2}{2-x/2}
\psset{PointSymbol=o}
\pstInterFF{2-x/2}{x^2}{1}{M_1}
\pstInterFF{2-x/2}{x^2}{-2}{M_0}
\end{pspicture}
```

6.5. Function–line

\pstInterFL [Options] $\{f\}\{A\}\{B\}\{x_0\}\{M\}$

Puts a point at the intersection between the function f and the line (AB).

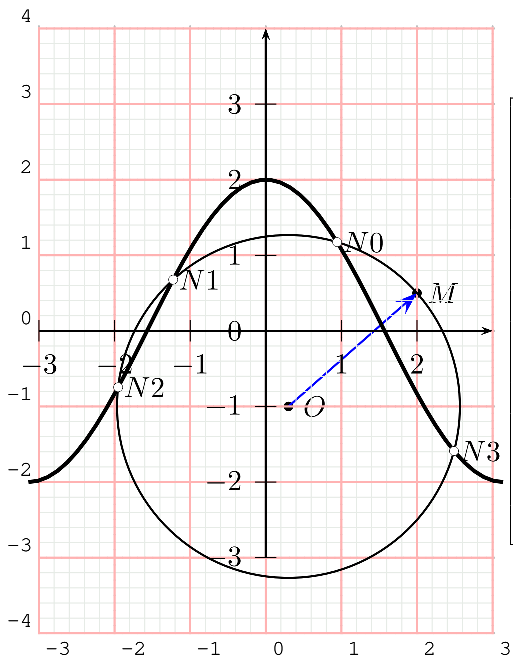


```
\begin{pspicture}[showgrid](-3,-1.5)(3,4)
\def\F{x^3/3 - x + 2/3 }
\psaxes{->}(0,0)(-3,-1)(3,4)
\psplot[linewidth=1.5pt,algebraic]{-2.5}{2.5}{\F}
\psset{PointSymbol=*}
\pstGeonode[PosAngle={-45,0}](0,-.2){N}(2.5,1){M}
\pstLineAB[nodesepA=-3cm]{N}{M}
\psset{PointSymbol=o,algebraic}
\pstInterFL{\F}{N}{M}{2}{A}
\pstInterFL[PosAngle=90]{\F}{N}{M}{0}{A'}
\pstInterFL{\F}{N}{M}{-2}{A''}
\end{pspicture}
```

6.6. Function–Circle

\pstInterFC [Options] $\{f\}\{O\}\{A\}\{x_0\}\{M\}$

Puts a point at the intersection between the function f and the circle of centre O and radius OA.

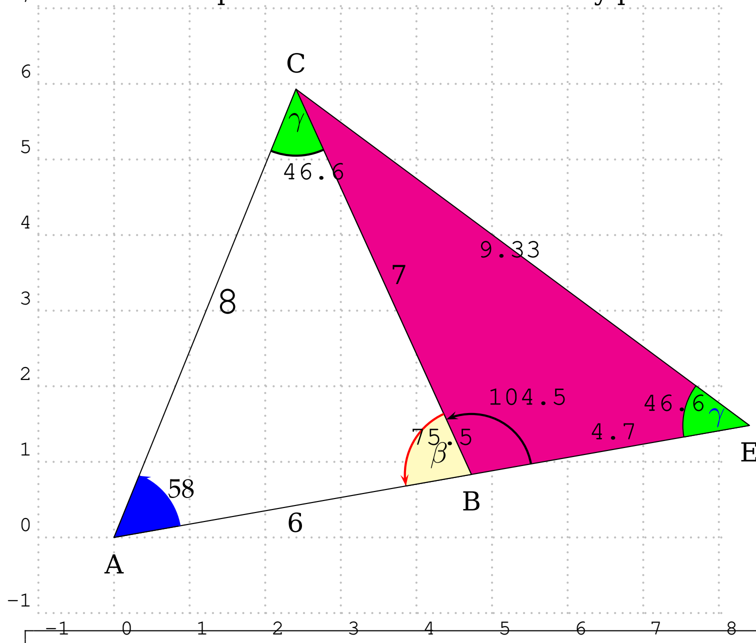


```
\begin{pspicture}[showgrid](-3,-4)(3,4)
\def\F{2*cos(x)}
\psset{algebraic}
\pstGeonode(0.3,-1){O}(2,.5){M}
\ncline[linecolor=blue, arrowscale=2]{->}{O}{M}
\psaxes{->}(0,0)(-3,-3)(3,4)
\psplot[linewidth=1.5pt]{-3.14}{3.14}{\F}
\pstCircleOA[PointSymbol=*]{O}{M}
\psset{PointSymbol=o}
\pstInterFC{\F}{O}{M}{1}{N0}
\pstInterFC{\F}{O}{M}{-1}{N1}
\pstInterFC{\F}{O}{M}{-2}{N2}
\pstInterFC{\F}{O}{M}{2}{N3}
\end{pspicture}
```

7. Helper Macros

```
\psGetDistanceAB [Options] (x1,y1)(x2,y2){<name>}
\psGetAngleABC [Options] (x1,y1)(x2,y2)(x3,y3){<symbol>}
```

Galculates and prints the values. This is only possible on PostScript level!



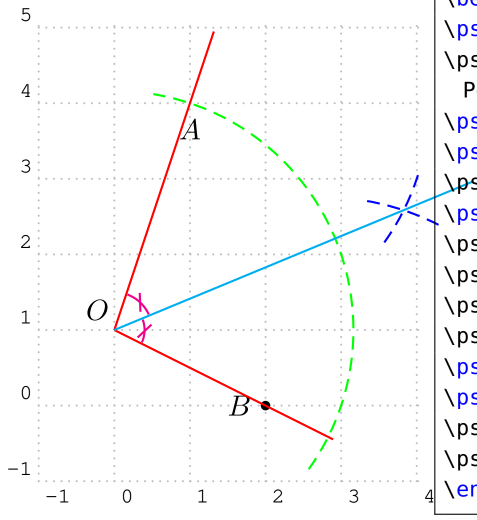
```
\begin{pspicture}(-1,0)(11,8)
\psgrid[gridlabels=0pt,subgriddiv=2,gridwidth=0.4pt,subgridwidth=0.2pt,gridcolor=black!60,
    subgridcolor=black!40]
\def\sideC{6} \def\sideA{7} \def\sideB{8}
\psset{PointSymbol=none,linejoin=1,linewidth=0.4pt,PtNameMath=false,labelsep=0.07,MarkAngleRadius
    =1.1,decimals=1,comma}
\pstGeonode[PosAngle={-90,-90}](0,0){A}(\sideC;10){B}
\pstInterCC[RadiusA=\pstDistVal{\sideB},RadiusB=\pstDistVal{\sideA},PosAngle=90,PointNameA=C]{A}{}{B
    }{}{C}{C-}
\pstInterCC[RadiusA=\pstDistAB{A}{B},RadiusB=\pstDistAB{B}{C}]{C}{}{A}{}{D-}{D}
\pstInterLC[Radius=\pstDistAB{A}{C}]{C}{D}{C}{}{A'-}{A'}
```

```
\pstInterCC[RadiusA=\pstDistAB{A}{B},RadiusB=\pstDistAB{B}{C}]{A'}{}{C}{}{B'}{B'-}
\pstInterLL[PosAngle=90,PointName=default]{B'}{C}{A}{B}{E}
\pspolygon(A)(B)(C)
\pspolygon[fillstyle=solid,fillcolor=magenta,opacity=0.1](C)(E)(B)
%
\psGetAngleABC[ArcColor=blue,AngleValue=true,LabelSep=0.8,arrows=->,decimals=0,PSfont=Palatino-Roman
    ](B)(A)(C){}
\psGetAngleABC[AngleValue=true,ArcColor=red,arrows=->,WedgeOpacity=0.6,WedgeColor=yellow!30,LabelSep
    =0.5](C)(B)(A){$\beta$}
\psGetAngleABC[LabelSep=0.8,WedgeColor=green,xShift=-6,yShift=-10](A)(C)(B){$\gamma$}
\psGetAngleABC[LabelSep=0.8,AngleArc=false,WedgeColor=green,arrows=->,xShift=-15,yShift=0](C)(E)(B)
    {\color{blue}$\gamma$}
\psGetAngleABC[AngleValue=true,MarkAngleRadius=1.0,LabelSep=0.5,ShowWedge=false,xShift=-5,yShift=7,
    arrows=->](E)(B)(C){}
%
\pcline[linestyle=none](A)(B)\nbput{\sideC}
\pcline[linestyle=none](C)(B)\naput{\sideA}
\psGetDistanceAB[xShift=-8,yShift=4](B)(E){MW}
\psGetDistanceAB[fontscale=15,xShift=4,decimals=0](A)(C){MAC}
\psGetDistanceAB[xShift=-17,decimals=2](E)(C){MEC}
\end{pspicture}
```

Part II. Examples gallery

A. Basic geometry

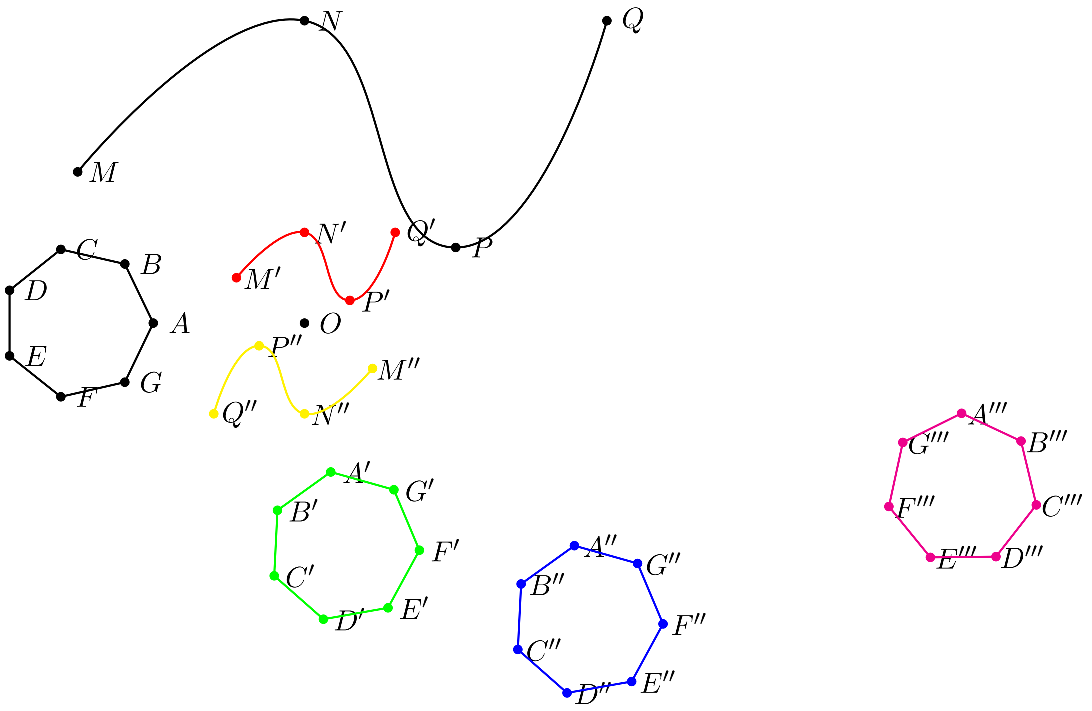
A.1. Drawing of the bissector



```
\begin{pspicture}[showgrid](-1,-1)(4.4,5)
\psset{PointSymbol=none,PointName=none}
\pstGeonode[PosAngle={180,130,-90},PointSymbol={*,none},
  PointName=default](2,0){B}(0,1){O}(1,4){A}
\pstLineAB[nodesepB=-1,linecolor=red]{O}{A}
\pstLineAB[nodesepB=-1,linecolor=red]{O}{B}
\pstInterLC[PosAngleB=-45]{O}{B}{O}{A}{G}{C}
\psset{arcsepA=-1, arcsepB=-1}
\pstArcOAB[linecolor=green,linestyle=dashed]{O}{C}{A}
\pstInterCC[PosAngleA=100]{A}{O}{C}{O}{O'}{OO}
\pstArcOAB[linecolor=blue,linestyle=dashed]{A}{O'}{O'}
\pstArcOAB[linecolor=blue,linestyle=dashed]{C}{O'}{O'}
\pstLineAB[nodesepB=-1,linecolor=cyan]{O}{O'}
\psset{arcsep=1pt,linecolor=magenta,Mark=MarkHash}
\pstMarkAngle{C}{O}{O'}{}
\pstMarkAngle[MarkAngleRadius=.5]{O'}{O}{A}{}
\end{pspicture}
```

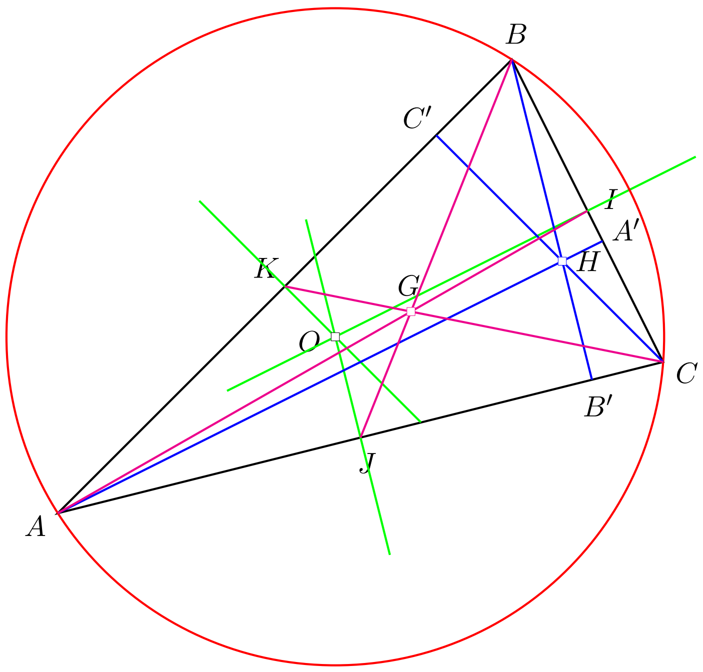
A.2. Transformation de polygones et courbes

Here is an example of the use of `CurveType` with transformation.



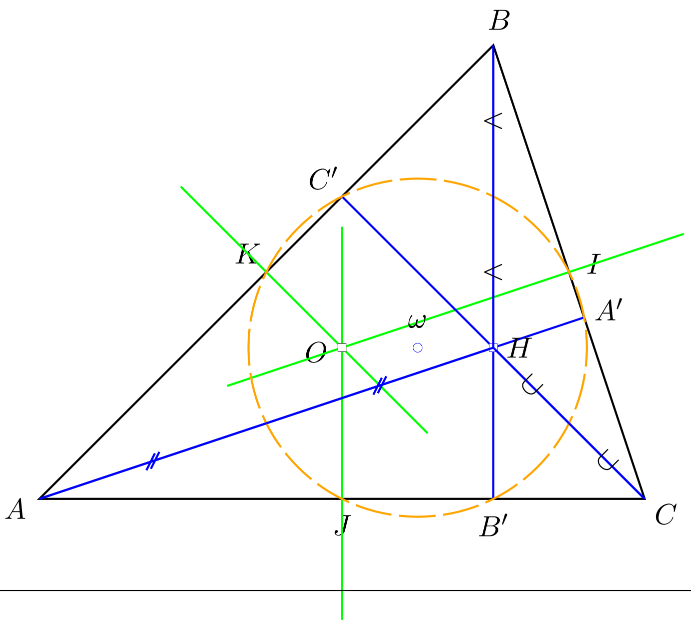
```
\begin{pspicture}(-5,-5)(10,5)
\pstGeonode{O}
\rput(-3,0){\pstGeonode[CurveType=polygon](1,0){A}(1;51.43){B}(1;102.86){C}
  (1;154.29){D}(1;205.71){E}(1;257.14){F}(1;308.57){G}}
\rput(-4,-1){\pstGeonode[CurveType=curve](1,3){M}(4,5){N}(6,2){P}(8,5){Q}}
\pstRotation[linecolor=green, RotAngle=100, CurveType=polygon]{O}{A, B, C, D, E, F, G}
\pstHomO[linecolor=red, HomCoef=.3, CurveType=curve]{O}{M,N,P,Q}
\pstTranslation[linecolor=blue, CurveType=polygon]{C}{O}{A’, B’, C’, D’, E’, F’, G’}
\pstSymO[linecolor=yellow, CurveType=curve]{O}{M’,N’,P’,Q’}
\pstOrtSym[linecolor=magenta, CurveType=polygon]{Q}{F’’}
  {A’, B’, C’, D’, E’, F’, G’}[A’’’, B’’’, C’’’, D’’’, E’’’, F’’’, G’’’]
\end{pspicture}
```

A.3. Triangle lines



```
\psset{unit=2}
\begin{pspicture}(-3,-2)(3,3)
\psset{PointSymbol=none}
\pstTriangle[PointSymbol=none](-2,-1){A}(1,2){B}(2,0){C}
{ \psset{linestyle=none, PointNameB=none}
  \pstMediatorAB{A}{B}{K}{KP}
  \pstMediatorAB[PosAngleA=-40]{C}{A}{J}{JP}
  \pstMediatorAB[PosAngleA=75]{B}{C}{I}{IP}
}% fin
\pstInterLL[PointSymbol=square, PosAngle=-170]{I}{IP}{J}{JP}{O}
{% encapsulation de modif parametres
  \psset{nodesep=-.8, linecolor=green}
  \pstLineAB{O}{I}\pstLineAB{O}{J}\pstLineAB{O}{K}
}% fin
\pstCircleOA[linecolor=red]{O}{A}
% pour que le symbol de O soit sur et non sous les droites
\psdot[dotstyle=square](O)
% les hauteurs et l'orthocentre
\pstProjection{B}{A}{C}
\pstProjection{B}{C}{A}
\pstProjection{A}{C}{B}
\psset{linecolor=blue}\ncline{A}{A'}\ncline{C}{C'}\ncline{B}{B'}
\pstInterLL[PointSymbol=square]{A}{A'}{B}{B'}{H}
% les medianes et le centre de gravite
\psset{linecolor=magenta}\ncline{A}{I}\ncline{C}{K}\ncline{B}{J}
\pstCGravABC[PointSymbol=square, PosAngle=95]{A}{B}{C}{G}
\end{pspicture}
```

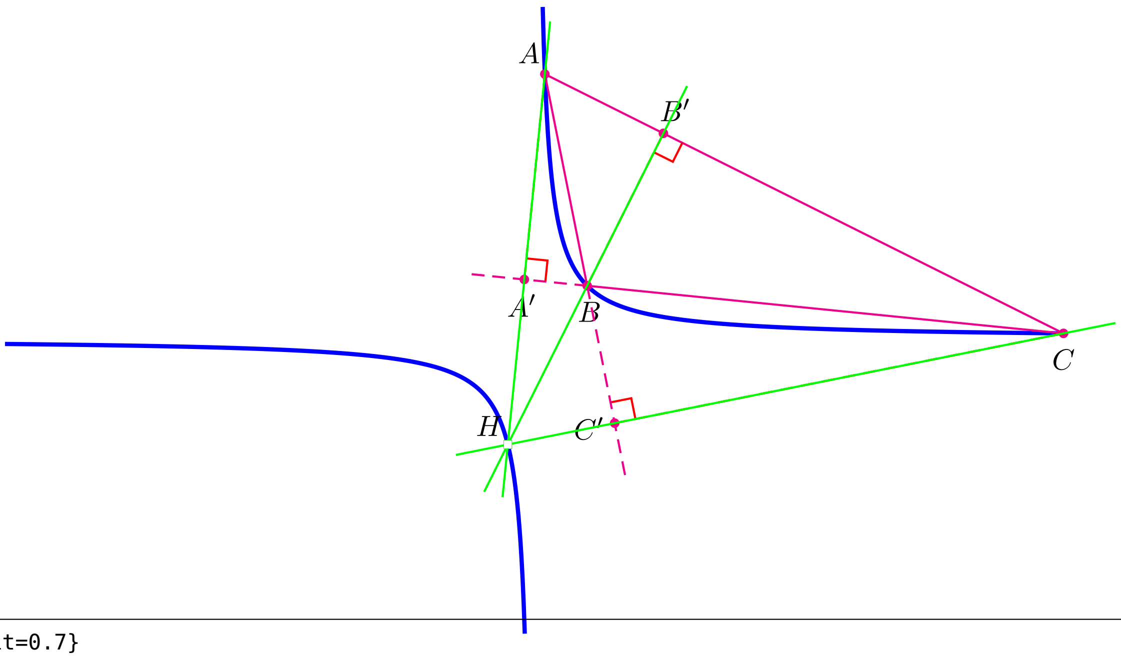
A.4. Euler circle



```
\psset{unit=2}
\begin{pspicture}(-3,-1.5)(3,2.5)
\psset{PointSymbol=none}
\pstTriangle(-2,-1){A}(1,2){B}(2,-1){C}
{% encapsulation de modif parametres
  \psset{linestyle=none, PointSymbolB=none, PointNameB=none}
  \pstMediatorAB{A}{B}{K}{KP}
  \pstMediatorAB{C}{A}{J}{JP}
  \pstMediatorAB{B}{C}{I}{IP}
}% fin
\pstInterLL[PointSymbol=square, PosAngle=-170]{I}{IP}{J}{JP}{O}
{% encapsulation de modif parametres
  \psset{nodesep=-.8, linecolor=green}
  \pstLineAB{O}{I}\pstLineAB{O}{J}\pstLineAB{O}{K}
}% fin
\psdot[dotstyle=square](O)
\pstProjection{B}{A}{C}
\pstProjection{B}{C}{A}
\pstProjection{A}{C}{B}
\psset{linecolor=blue}\ncline{A}{A'}\ncline{C}{C'}\ncline{B}{B'}
\pstInterLL[PointSymbol=square]{A}{A'}{B}{B'}{H}
% le cercle d'Euler (centre au milieu de [OH])
\pstMiddleAB[PointSymbol=o, PointName=\omega]{O}{H}{omega}
\pstCircleOA[linecolor=Orange, linestyle=dashed, dash=5mm 1mm]{omega}{B'}
\psset{PointName=none}
% il passe par le milieu des segments joignant l'orthocentre et les sommets
\pstMiddleAB{H}{A}{AH}\pstMiddleAB{H}{B}{BH}\pstMiddleAB{H}{C}{CH}
\pstSegmentMark{H}{AH}\pstSegmentMark{AH}{A}
\psset{SegmentSymbol=wedge}\pstSegmentMark{H}{BH}\pstSegmentMark{BH}{B}
\psset{SegmentSymbol=cup}\pstSegmentMark{H}{CH}\pstSegmentMark{CH}{C}
\end{pspicture}
```

A.5. Orthocenter and hyperbola

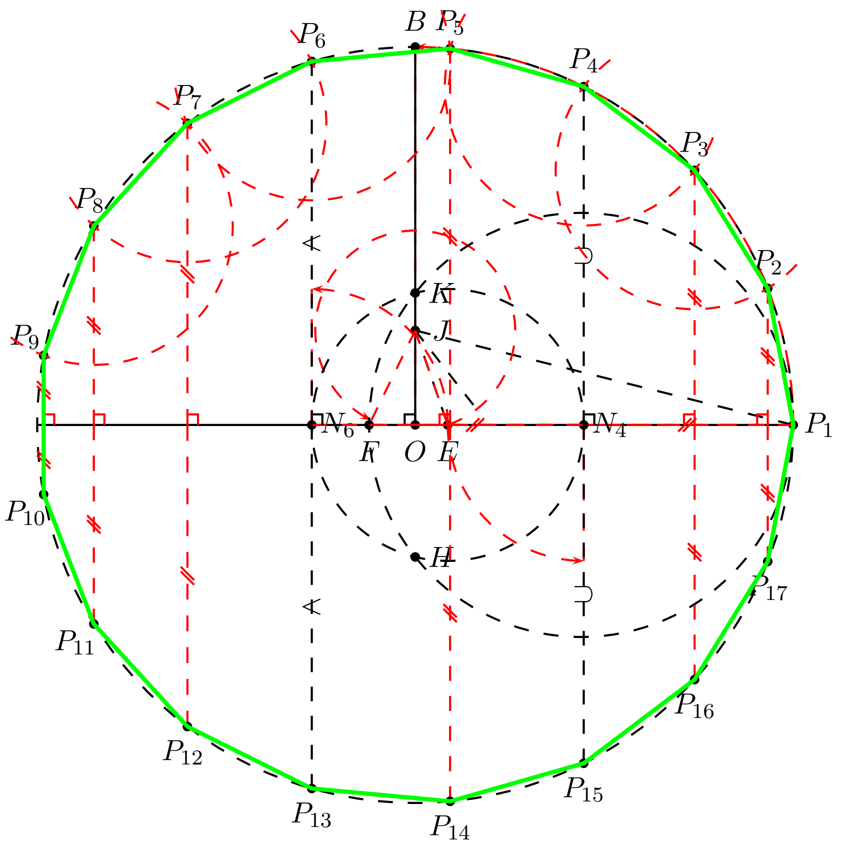
The orthocenter of a triangle whose points are on the branches of the hyperbola $\mathcal{H} : y = a/x$ belong to this hyperbola.



```
\psset{unit=0.7}
\begin{pspicture}(-11,-5)(11,7)
\psset{linecolor=blue, linewidth=2\pslinewidth}
\psplot[yMaxValue=6,plotpoints=500]{-10}{-.1}{1 x div}
\psplot[yMaxValue=6,plotpoints=500]{.1}{10}{1 x div}
\psset{%PointSymbol=none,
linewidth=.5\pslinewidth}
\pstTriangle[linecolor=magenta, PosAngleB=-85, PosAngleC=-90](.2,5){A}(1,1){B}(10,.1){C}
\psset{linecolor=magenta,CodeFig=true, CodeFigColor=red}
\pstProjection{B}{A}{C}
\ncline[nodesepA=-1,linestyle=dashed,linecolor=magenta]{C'}{B}
\pstProjection{B}{C}{A}
\ncline[nodesepA=-1,linestyle=dashed,linecolor=magenta]{A'}{B}
\pstProjection{A}{C}{B}
\pstInterLL[PosAngle=135,PointSymbol=square]{A}{A'}{B}{B'}{H}
\psset{linecolor=green, nodesep=-1}
\pstLineAB{A}{H}\pstLineAB{B'}{H}\pstLineAB{C}{H}
\psdot[dotstyle=square](H)
\end{pspicture}
```

A.6. 17 sides regular polygon

Striking picture created by K. F. Gauss. he also prooved that it is possible to build the regular polygons which have $2^{2^p}+1$ sides, the following one has 257 sides!

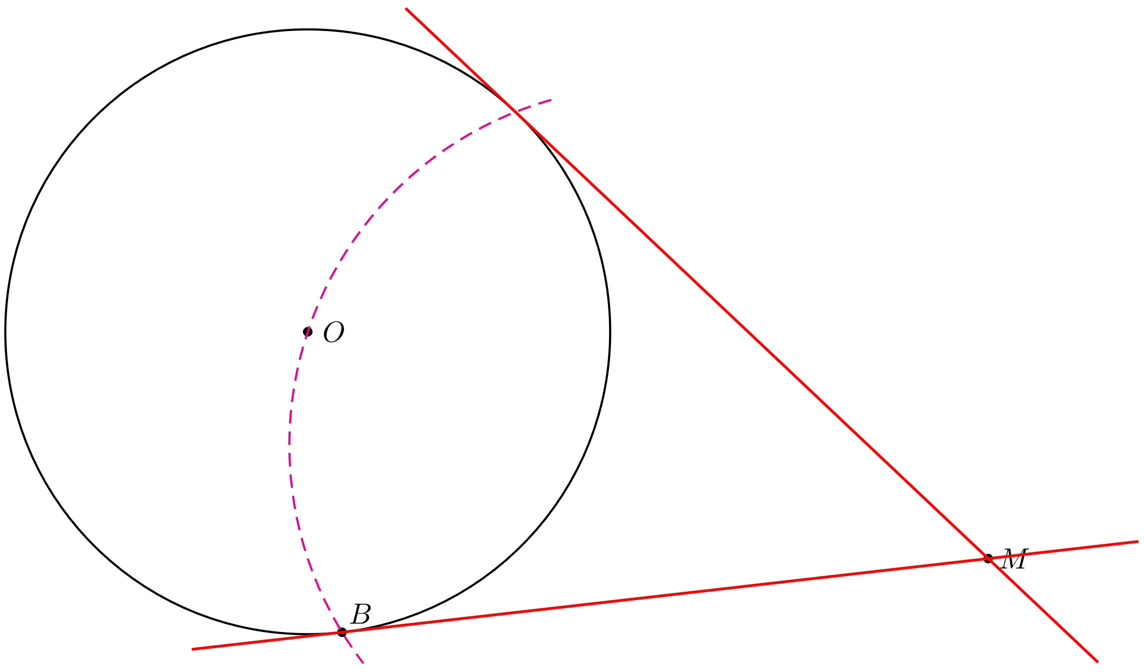


```
\begin{pspicture}(-5.5,-5.5)(5.5,6)
 \psset{CodeFig, RightAngleSize=.14, CodeFigColor=red,
   CodeFigB=true, linestyle=dashed, dash=2mm 2mm}
 \pstGeonode[PosAngle={-90,0}]{O}(5;0){P_1}
 \pstCircleOA{O}{P_1}
 \pstSymO[PointSymbol=none, PointName=none, CodeFig=false]{O}{P_1}[PP_1]
 \ncline[linestyle=solid]{PP_1}{P_1}
 \pstRotation[RotAngle=90, PosAngle=90]{O}{P_1}[B]
 \pstRightAngle[linestyle=solid]{B}{O}{PP_1}\ncline[linestyle=solid]{O}{B}
 \pstHomO[HomCoef=.25]{O}{B}[J] \ncline{J}{P_1}
 \pstBissectBAC[PointSymbol=none, PointName=none]{O}{J}{P_1}{PE1}
 \pstBissectBAC[PointSymbol=none, PointName=none]{O}{J}{PE1}{PE2}
 \pstInterLL[PosAngle=-90]{O}{P_1}{J}{PE2}{E}
 \pstRotation[PosAngle=-90, RotAngle=-45, PointSymbol=none, PointName=none]{J}{E}[PF1]
 \pstInterLL[PosAngle=-90]{O}{P_1}{J}{PF1}{F}
 \pstMiddleAB[PointSymbol=none, PointName=none]{F}{P_1}{MFP1} \pstCircleOA{MFP1}{P_1}
 \pstInterLC[PointSymbolA=none, PointNameA=none]{O}{B}{MFP1}{P_1}{H}{K}
 \pstCircleOA{E}{K} \pstInterLC{O}{P_1}{E}{K}{N_6}{N_4}
 \pstRotation[RotAngle=90,PointSymbol=none, PointName=none]{N_6}{E}[PP_6]
 \pstInterLC[PosAngleA=90,PosAngleB=-90, PointNameB=P_{13}]{PP_6}{N_6}{O}{P_1}{P_6}{P_13}
 \pstSegmentMark[SegmentSymbol=wedge]{N_6}{P_6}
 \pstSegmentMark[SegmentSymbol=wedge]{P_13}{N_6}
 \pstRotation[RotAngle=90,PointSymbol=none, PointName=none]{N_4}{E}[PP_4]
 \pstInterLC[PosAngleA=90,PosAngleB=-90,PointNameB=P_{15}]{N_4}{PP_4}{O}{P_1}{P_4}{P_15}
 \pstSegmentMark[SegmentSymbol=cup]{N_4}{P_4}
 \pstSegmentMark[SegmentSymbol=cup]{P_15}{N_4}
```

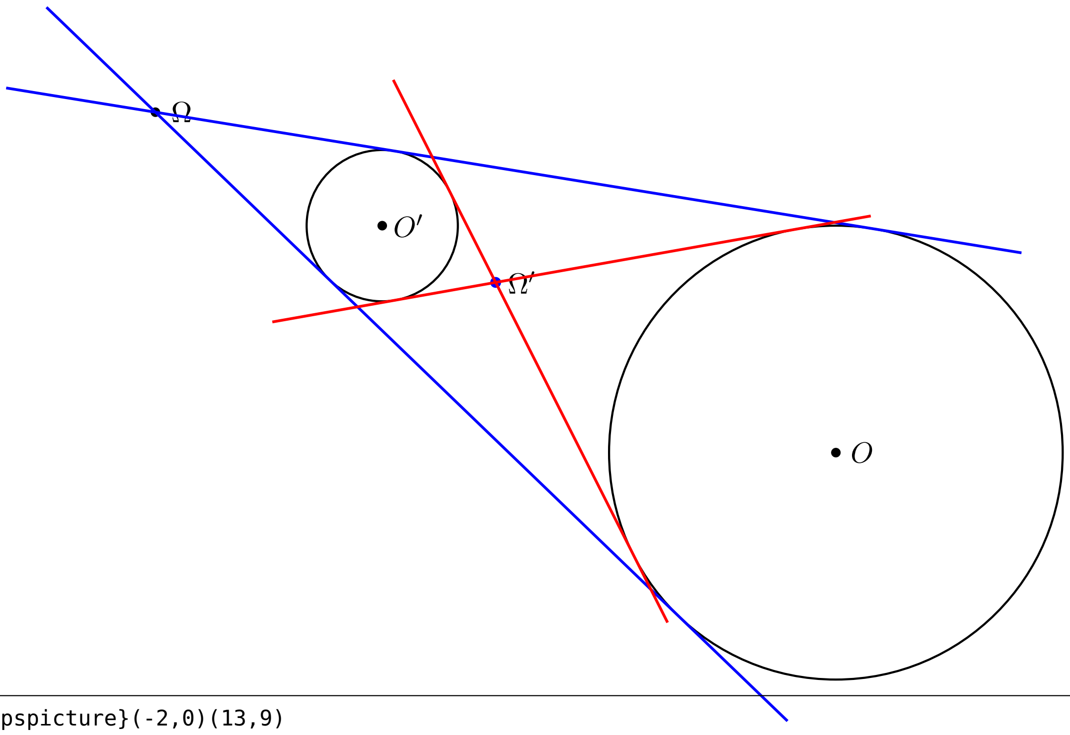
```
  \pstRightAngle[linestyle=solid]{P_1}{N_6}{P_6}
  \pstRightAngle[linestyle=solid]{P_1}{N_4}{P_4}
  \pstBissectBAC[PosAngle=90, linestyle=none]{P_4}{O}{P_6}{P_5}
  \pstInterCC[PosAngleB=90, PointSymbolA=none, PointNameA=none]{O}{P_1}{P_4}{P_5}{H}{P_3}
  \pstInterCC[PosAngleB=90, PointSymbolA=none, PointNameA=none]{O}{P_1}{P_3}{P_4}{H}{P_2}
  \pstInterCC[PosAngleA=90, PointSymbolB=none, PointNameB=none]{O}{P_1}{P_6}{P_5}{P_7}{H}
  \pstInterCC[PosAngleA=100, PointSymbolB=none, PointNameB=none]{O}{P_1}{P_7}{P_6}{P_8}{H}
  \pstInterCC[PosAngleA=135, PointSymbolB=none, PointNameB=none]{O}{P_1}{P_8}{P_7}{P_9}{H}
  \pstOrtSym[PosAngle={-90,-90,-90,-100,-135},PointName={P_{17},P_{16},P_{14},P_{12},P_{11},P_{10}}]
          {O}{P_1}{P_2,P_3,P_5,P_7,P_8,P_9}[P_17,P_16,P_14,P_12,P_11,P_10]
  \pspolygon[linecolor=green, linestyle=solid, linewidth=2\pslinewidth]
   (P_1)(P_2)(P_3)(P_4)(P_5)(P_6)(P_7)(P_8)(P_9)(P_10)(P_11)(P_12)(P_13)(P_14)(P_15)(P_16)(P_17)
\end{pspicture}
```

A.7. Circles & tangents

The drawing of the circle tangents which crosses a given point.



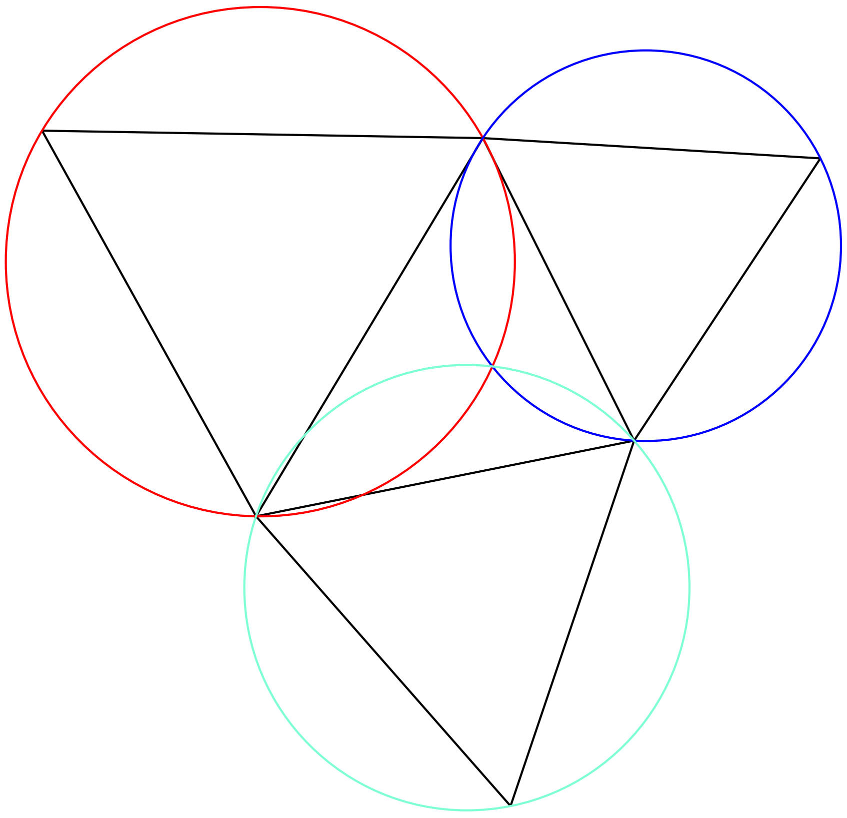
```
\begin{pspicture}(15,10)
\pstGeonode(5, 5){O}(14,2){M}
\pstCircleOA[Radius=\pstDistVal{4}]{O}{}
\pstMiddleAB[PointSymbol=none, PointName=none]{O}{M}{O'}
\pstInterCC[RadiusA=\pstDistVal{4}, DiameterB=\pstDistAB{O}{M},
         CodeFigB=true, CodeFigColor=magenta, PosAngleB=45]{O}{}{O'}{}{A}{B}
\psset{linecolor=red, linewidth=1.3\pslinewidth, nodesep=-2}
\pstLineAB{M}{A}\pstLineAB{M}{B}
\end{pspicture}
```



```
\begin{pspicture}(-2,0)(13,9)
\pstGeonode(9,3){O}(3,6){O'}\psset{PointSymbol=none, PointName=none}
\pstCircleOA[Radius=\pstDistVal{3}]{O}{}\pstCircleOA[Radius=\pstDistVal{1}]{O'}{}
\pstInterLC[Radius=\pstDistVal{3}]{O}{O'}{O}{}{M}{toto}
\pstInterLC[Radius=\pstDistVal{1}]{O}{O'}{O'}{}{M'}{toto}
\pstRotation[RotAngle=30]{O}{M}[N]
\pstRotation[RotAngle=30]{O'}{M'}[N']
\pstInterLL[PointSymbol=*, PointName=\Omega]{O}{O'}{N}{N'}{Omega}
\pstMiddleAB{O}{Omega}{I} \pstInterCC{I}{O}{O}{M}{A}{B}
\psset{nodesepA=-1, nodesepB=-3, linecolor=blue, linewidth=1.3\pslinewidth}
\pstLineAB[nodesep=-2]{A}{Omega}\pstLineAB[nodesep=-2]{B}{Omega}
\pstRotation[RotAngle=-150]{O'}{M'}[N'']
\pstInterLL[PointSymbol=*, PointName=\Omega']{O}{O'}{N}{N''}{Omega'}
\pstMiddleAB{O}{Omega'}{J}
\pstInterCC{J}{O}{O}{M}{A'}{B'}
\psset{nodesepA=-1, nodesepB=-3, linecolor=red}
\pstLineAB{A'}{Omega'}\pstLineAB{B'}{Omega'}
\end{pspicture}
```

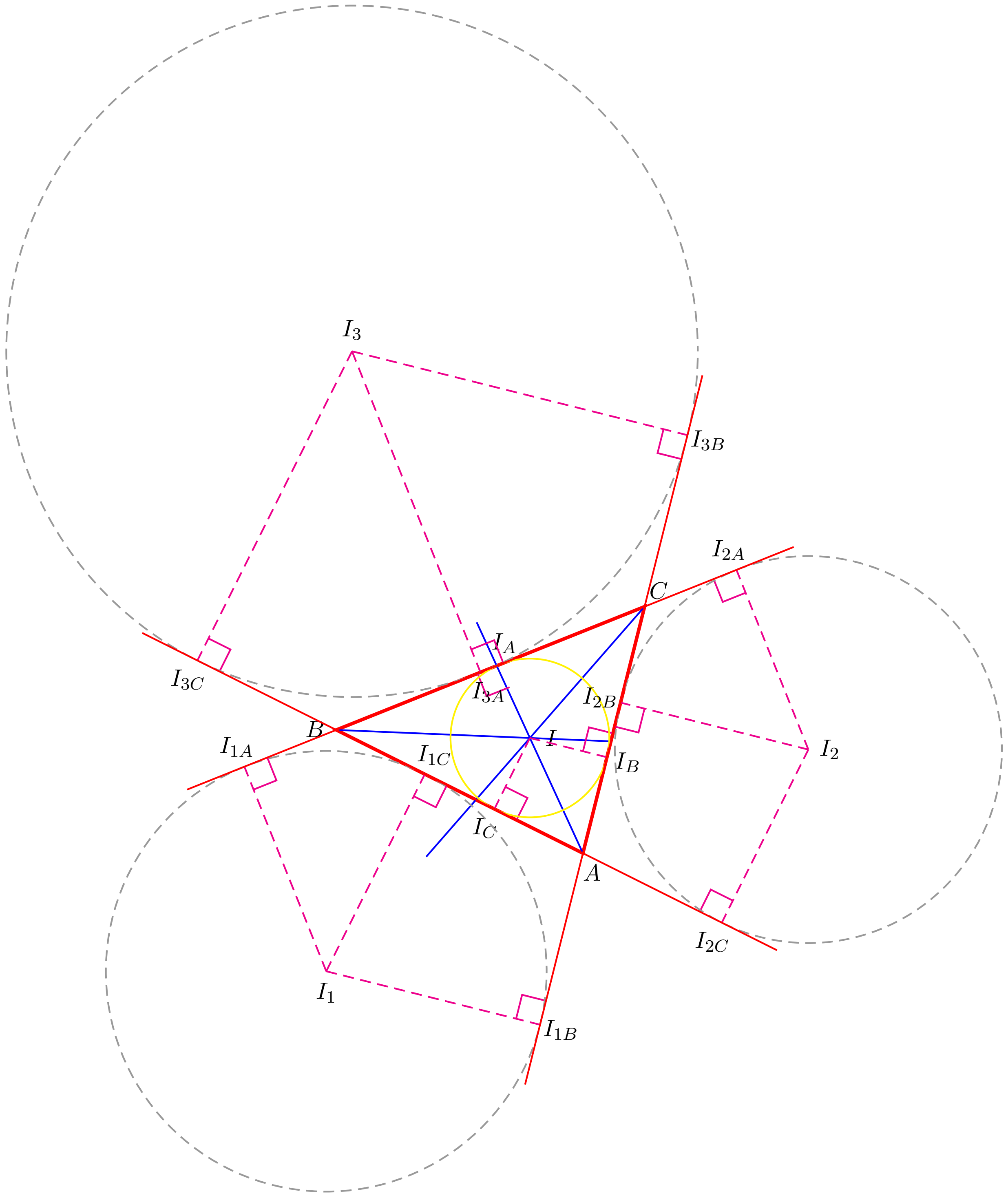
A.8. Fermat’s point

Drawing of Manuel Luque.



```
\begin{pspicture}(-7,-6)(5,5)
\psset{PointSymbol=none, PointName=none}
\pstTriangle[PosAngleA=-160,PosAngleB=90,PosAngleC=-25](-3,-2){B}(0,3){A}(2,-1){C}%
\psset{RotAngle=-60}
\pstRotation[PosAngle=-90]{B}{C}[A’]
\pstRotation{C}{A}[B’]
\pstRotation[PosAngle=160]{A}{B}[C’]
\pstLineAB{A}{B’}
\pstLineAB{C}{B’}
\pstLineAB{B}{A’}
\pstLineAB{C}{A’}
\pstLineAB{B}{C’}
\pstLineAB{A}{C’}
\pstCircleABC[linecolor=red]{A}{B}{C’}{O_1}
\pstCircleABC[linecolor=blue]{A}{C}{B’}{O_2}
\pstCircleABC[linecolor=Aquamarine]{A’}{C}{B}{O_3}
\pstInterCC[PointSymbolA=none]{O_1}{A}{O_2}{A}{E}{F}
\end{pspicture}
```

A.9. Escribed and inscribed circles of a triangle



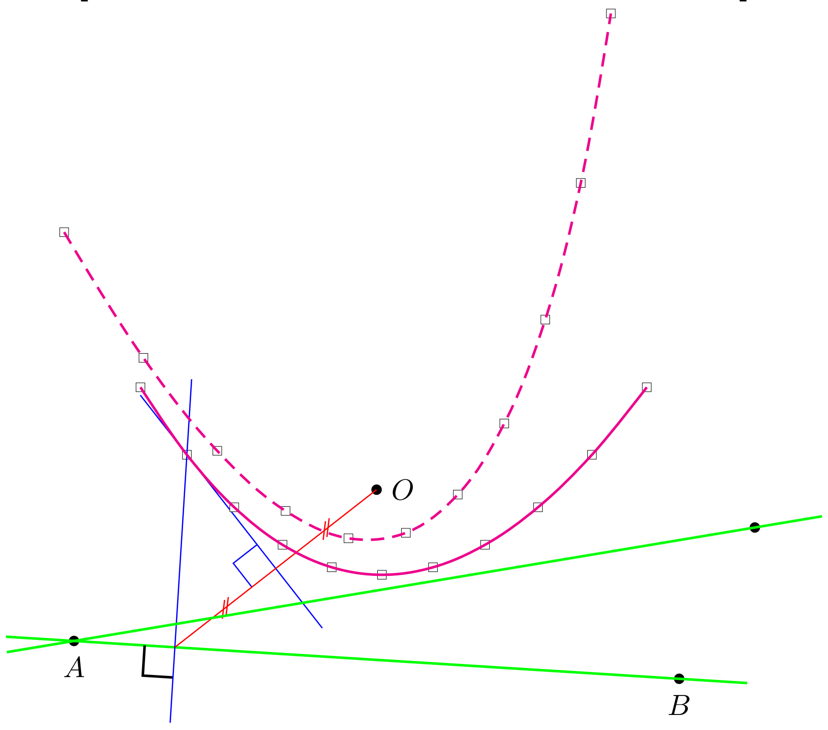
```
\begin{pspicture}(-6,-5)(11,15)
\psset{PointSymbol=none}
\pstTriangle[linewidth=2\pslinewidth,linecolor=red](4,1){A}(0,3){B}(5,5){C}
\psset{linecolor=blue}
\pstBissectBAC[PointSymbol=none,PointName=none]{C}{A}{B}{AB}
\pstBissectBAC[PointSymbol=none,PointName=none]{A}{B}{C}{BB}
\pstBissectBAC[PointSymbol=none,PointName=none]{B}{C}{A}{CB}
\pstInterLL{A}{AB}{B}{BB}{I}
```

```
\psset{linecolor=magenta, linestyle=dashed}
\pstProjection{A}{B}{I}[I_C]
\pstLineAB{I}{I_C}\pstRightAngle[linestyle=solid]{A}{I_C}{I}
\pstProjection{A}{C}{I}[I_B]
\pstLineAB{I}{I_B}\pstRightAngle[linestyle=solid]{C}{I_B}{I}
\pstProjection[PosAngle=80]{C}{B}{I}[I_A]
\pstLineAB{I}{IA}\pstRightAngle[linestyle=solid]{B}{I_A}{I}
\pstCircleOA[linecolor=yellow, linestyle=solid]{I}{I_A}
\psset{linecolor=magenta, linestyle=none}
\pstOutBissectBAC[PointSymbol=none,PointName=none]{C}{A}{B}{AOB}
\pstOutBissectBAC[PointSymbol=none,PointName=none]{A}{B}{C}{BOB}
\pstOutBissectBAC[PointSymbol=none,PointName=none]{B}{C}{A}{COB}
\pstInterLL[PosAngle=-90]{A}{AOB}{B}{BOB}{I_1}
\pstInterLL{A}{AOB}{C}{COB}{I_2}
\pstInterLL[PosAngle=90]{C}{COB}{B}{BOB}{I_3}
\psset{linecolor=magenta, linestyle=dashed}
\pstProjection[PointName=I_{1C}]{A}{B}{I_1}[I1C]
\pstLineAB{I_1}{I1C}\pstRightAngle[linestyle=solid]{I_1}{I1C}{A}
\pstProjection[PointName=I_{1B}]{A}{C}{I_1}[I1B]
\pstLineAB{I_1}{I1B}\pstRightAngle[linestyle=solid]{A}{I1B}{I_1}
\pstProjection[PointName=I_{1A}]{C}{B}{I_1}[I1A]
\pstLineAB{I_1}{I1A}\pstRightAngle[linestyle=solid]{I_1}{I1A}{C}
\pstProjection[PointName=I_{2B}]{A}{C}{I_2}[I2B]
\pstLineAB{I_2}{I2B}\pstRightAngle[linestyle=solid]{A}{I2B}{I_2}
\pstProjection[PointName=I_{2C}]{A}{B}{I_2}[I2C]
\pstLineAB{I_2}{I2C}\pstRightAngle[linestyle=solid]{I_2}{I2C}{A}
\pstProjection[PointName=I_{2A}]{B}{C}{I_2}[I2A]
\pstLineAB{I_2}{I2A}\pstRightAngle[linestyle=solid]{C}{I2A}{I_2}
\pstProjection[PointName=I_{3A}]{C}{B}{I_3}[I3A]
\pstLineAB{I_3}{I3A}\pstRightAngle[linestyle=solid]{C}{I3A}{I_3}
\pstProjection[PointName=I_{3C}]{A}{B}{I_3}[I3C]
\pstLineAB{I_3}{I3C}\pstRightAngle[linestyle=solid]{A}{I3C}{I_3}
\pstProjection[PointName=I_{3B}]{C}{A}{I_3}[I3B]
\pstLineAB{I_3}{I3B}\pstRightAngle[linestyle=solid]{I_3}{I3B}{A}
\psset{linecolor=yellow, linestyle=solid}
\pstCircleOA{I_1}{I1C} \pstCircleOA{I_2}{I2B} \pstCircleOA{I_3}{I3A}
\psset{linecolor=red, linestyle=solid, nodesepA=-1, nodesepB=-1}
\pstLineAB{I1B}{I3B}\pstLineAB{I1A}{I2A}\pstLineAB{I2C}{I3C}
\end{pspicture}
```

B. Some locus points

B.1. Parabola

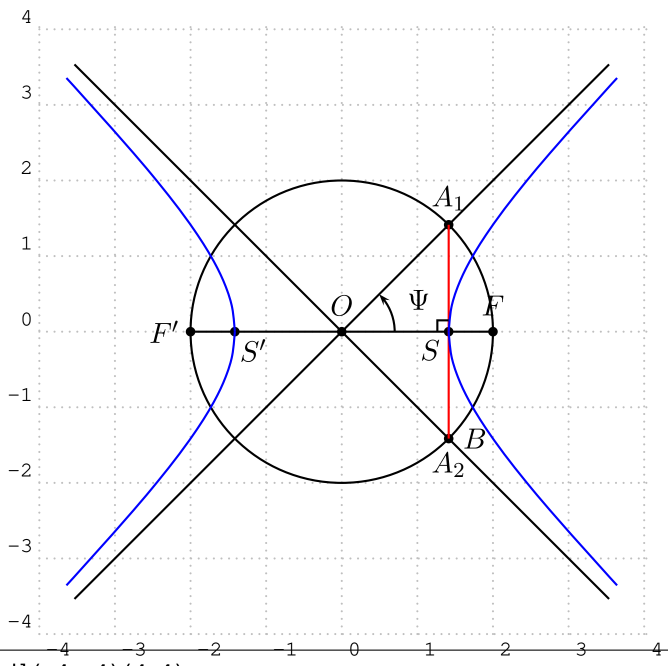
The parabola is the set of points which are at the same distance between a point and a line.



```
\def\NbPt{11}
\begin{pspicture}(-0.5,0)(11,10)
\psset{linewidth=1.2\pslinewidth}\renewcommand{\NbPt}{11}
\pstGeonode[PosAngle={0,-90}](5,4){O}(1,2){A}(9,1.5){B}
\newcommand\Parabole[1][100]{%
  \pstLineAB[nodesep=-.9, linecolor=green]{A}{B}
  \psset{RotAngle=90, PointSymbol=none, PointName=none}
  \multido{\n=1+1}{\NbPt}{%
    \pstHomO[HomCoef=\n\space \NbPt\space 1 add div]{A}{B}[M\n]
    \pstMediatorAB[linestyle=none]{M\n}{O}{M\n_I}{M\n_IP}
    \pstRotation{M\n}{A}[M\n_P]
    \pstInterLL[PointSymbol=square, PointName=none]{M\n_I}{M\n_IP}{M\n}{M\n_P}{P_\n}
    \ifnum\n=#1 \bgroup
      \pstRightAngle{A}{M\n}{M\n_P}
      \psset{linewidth=.5\pslinewidth, nodesep=-1, linecolor=blue}
      \pstLineAB{M\n_I}{P_\n}\pstLineAB{M\n}{P_\n}
      \pstRightAngle{P_\n}{M\n_I}{M\n}
      \psset{linecolor=red}\pstSegmentMark{M\n}{M\n_I}\pstSegmentMark{M\n_I}{O}
      \egroup \fi}}
\Parabole[2]\pstGenericCurve[linecolor=magenta]{P_}{1}{\NbPt}
\pstGeonode[PointSymbol=*, PosAngle=-90](10,3.5){B}
\Parabole\pstGenericCurve[linecolor=magenta,linestyle=dashed]{P_}{1}{\NbPt}
\end{pspicture}
```

B.2. Hyperbola

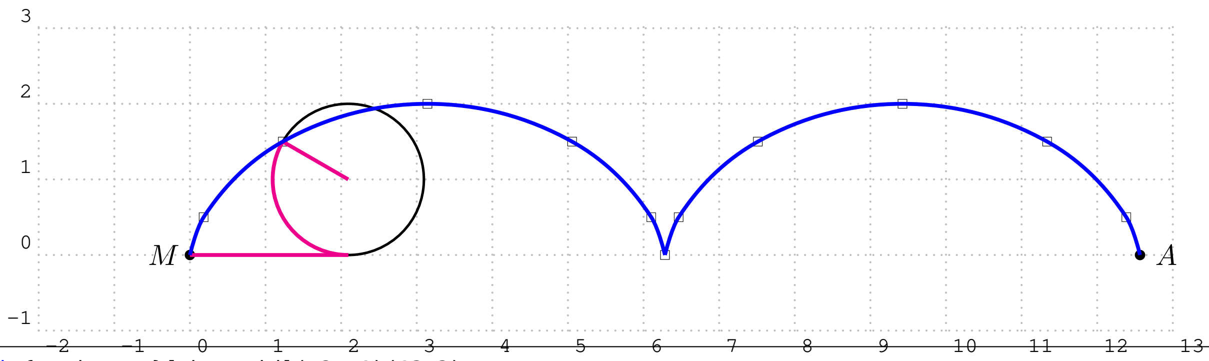
The hyperbola is the set of points whose difference between their distance of two points (the focus) is constant.



```
\begin{pspicture}[showgrid](-4,-4)(4,4)
\newcommand\Sommet{1.4142135623730951 } \newcounter{i} \setcounter{i}{1}
\newcommand\PosFoyer{2 } \newcommand\HypAngle{0}
\newcounter{CoefDiv}\setcounter{CoefDiv}{20}
\newcounter{Inc}\setcounter{Inc}{2} \newcounter{n}\setcounter{n}{2}
\newcommand\Ri{ \PosFoyer \Sommet sub \arabic{i}\space\arabic{CoefDiv}\space div add }
\newcommand\Rii{\Ri \Sommet 2 mul add }
\pstGeonode[PosAngle=90]{O}(\PosFoyer;\HypAngle){F}
\pstSymO[PosAngle=180]{O}{F}\pstLineAB{F}{F'} \pstCircleOA{O}{F}
\pstGeonode[PosAngle=-135](\Sommet;\HypAngle){S}
\pstGeonode[PosAngle=-45](-\Sommet;\HypAngle){S'}
\pstRotation[RotAngle=90, PointSymbol=none]{S}{O}[B]
\pstInterLC[PosAngleA=90, PosAngleB=-90]{S}{B}{O}{F}{A_1}{A_2}
\pstLineAB[nodesepA=-3,nodesepB=-5]{A_1}{O}\pstLineAB[nodesepA=-3,nodesepB=-5]{A_2}{O}
\pstMarkAngle[LabelSep=.8,MarkAngleRadius=.7,arrows=->,LabelSep=1.1]{F}{O}{A_1}{$\Psi$}
\ncline[linecolor=red]{A_1}{A_2} \pstRightAngle[RightAngleSize=.15]{A_1}{S}{O}
\psset{PointName=none}
\whiledo{\value{n}<8}{%
  \psset{RadiusA=\pstDistVal{\Ri},RadiusB=\pstDistVal{\Rii},PointSymbol=none}
  \pstInterCC{F}{}{F'}{}{M\arabic{n}}{P\arabic{n}}
  \pstInterCC{F'}{}{F}{}{M'\arabic{n}}{P'\arabic{n}}
  \stepcounter{n}\addtocounter{i}{\value{Inc}}
  \addtocounter{Inc}{\value{Inc}}}%% fin de whiledo
\psset{linecolor=blue}
\pstGenericCurve[GenCurvFirst=S]{M}{2}{7}
\pstGenericCurve[GenCurvFirst=S]{P}{2}{7}
\pstGenericCurve[GenCurvFirst=S']{M'}{2}{7}
\pstGenericCurve[GenCurvFirst=S']{P'}{2}{7}
\end{pspicture}
```

B.3. Cycloid

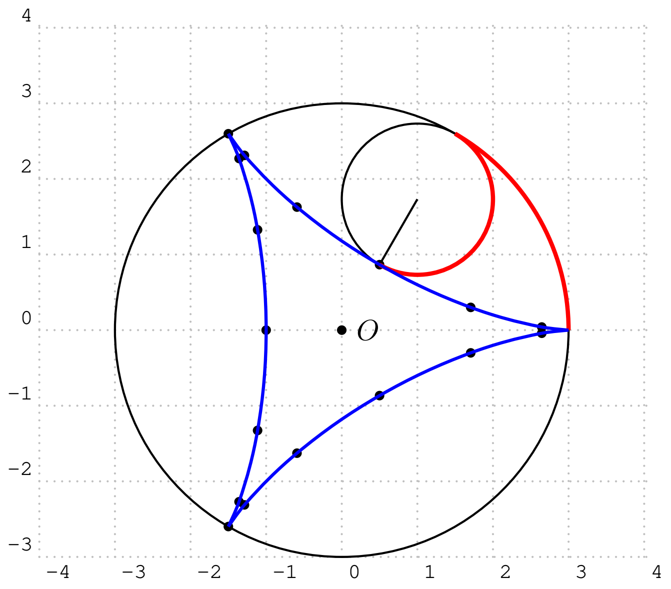
The wheel rolls from M to A. The circle points are on a cycloid.



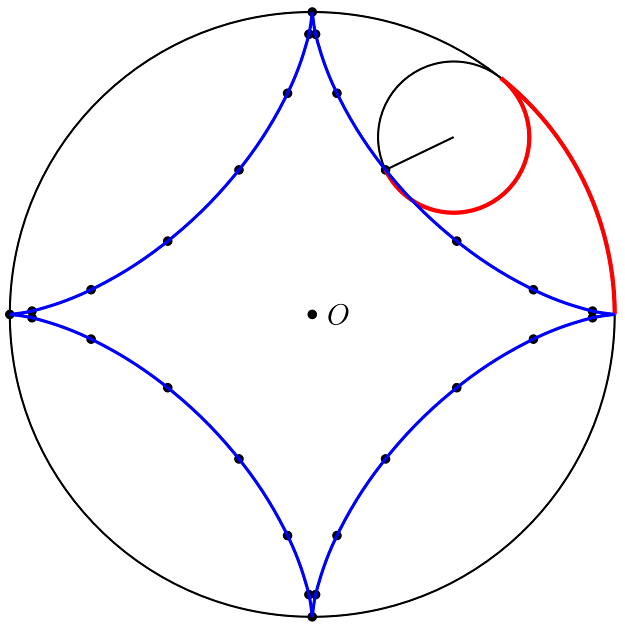
```
\begin{pspicture}[showgrid](-2,-1)(13,3)
\providecommand\NbPt{11}
\psset{linewidth=1.2\pslinewidth}
\pstGeonode[PointSymbol={*,none}, PointName={default,none}, PosAngle=180]{M}(0,1){O}
\pstGeonode(12.5663706144,0){A}
\pstTranslation[PointSymbol=none, PointName=none]{M}{A}{O}[B]
\multido{\nA=1+1}{\NbPt}{%
  \pstHomO[HomCoef=\nA\space\NbPt\space 1 add div,PointSymbol=none,PointName=none]{O}{B}[O\nA]
  \pstProjection[PointSymbol=none, PointName=none]{M}{A}{O\nA}[P\nA]
  \pstCurvAbsNode[PointSymbol=square, PointName=none,CurvAbsNeg=true]%
    {O\nA}{P\nA}{M\nA}{\pstDistAB{O}{O\nA}}
  \ifnum\nA=2 \bgroup
    \pstCircleOA{O\nA}{M\nA}
    \psset{linecolor=magenta, linewidth=1.5\pslinewidth}
    \pstArcnOAB{O\nA}{P\nA}{M\nA}
    \ncline{O\nA}{M\nA}\ncline{P\nA}{M}
    \egroup \fi
  }% fin du multido
\psset{linecolor=blue, linewidth=1.5\pslinewidth}
\pstGenericCurve[GenCurvFirst=M]{M}{1}{6} \pstGenericCurve[GenCurvLast=A]{M}{6}{\NbPt}
\end{pspicture}
```

B.4. Hypocycloids (Astroid and Deltoid)

A wheel rolls inside a circle, and depending of the radius ratio, it is an astroid, a deltoid and in the general case hypo-cycloids.



```
\newcommand\HypoCyclo[4][100]{%
  \def\R{#2}\def\petitR{#3}\def\NbPt{#4}
  \def\Anglen{\n\space 360 \NbPt\space 1 add div mul}
  \psset{PointSymbol=none,PointName=none}
  \pstGeonode[PointSymbol={*,none},PointName={default,none}, PosAngle=0]{O}(\R;0){P}
  \pstCircleOA{O}{P}
  \pstHomO[HomCoef=\petitR\space\R\space div]{P}{O}[M]
  \multido{\n=1+1}{\NbPt}{%
    \pstRotation[RotAngle=\Anglen]{O}{M}[M\n]
    \rput(M\n){\pstGeonode(\petitR;0){Q}}
    \pstRotation[RotAngle=\Anglen]{M\n}{Q}[N]
    \pstRotation[RotAngle=\n\space -360 \NbPt\space 1 add div
    mul \R\space\petitR\space div mul,PointSymbol=*,PointName=none]{M\n}{N}[N\n]
    \ifnum\n=#1
      \pstCircleOA{M\n}{N\n}\ncline{M\n}{N\n}%
      {\psset{linecolor=red, linewidth=2\pslinewidth}
      \pstArcOAB{M\n}{N\n}{N}\pstArcOAB{O}{P}{N}}
    \fi}}%fin multido-newcommand
\begin{pspicture}[showgrid](-3.5,-3.4)(3.5,4)
\HypoCyclo[3]{3}{1}{17}
\psset{linecolor=blue,linewidth=1.5\pslinewidth}
\pstGenericCurve[GenCurvFirst=P]{N}{1}{6}
\pstGenericCurve{N}{6}{12}
\pstGenericCurve[GenCurvLast=P]{N}{12}{17}
\end{pspicture}
```



```
\newcommand\HypoCyclo[4][100]{%
  \def\R{#2}\def\petitR{#3}\def\NbPt{#4}
  \def\Anglen{\n\space 360 \NbPt\space 1 add div mul}
  \psset{PointSymbol=none,PointName=none}
  \pstGeonode[PointSymbol={*,none},PointName={default,none}, PosAngle=0]{O}(\R;0){P}
  \pstCircleOA{O}{P}
  \pstHomO[HomCoef=\petitR\space\R\space div]{P}{O}[M]
  \multido{\n=1+1}{\NbPt}{%
    \pstRotation[RotAngle=\Anglen]{O}{M}[M\n]
    \rput(M\n){\pstGeonode(\petitR;0){Q}}
    \pstRotation[RotAngle=\Anglen]{M\n}{Q}[N]
    \pstRotation[RotAngle=\n\space -360 \NbPt\space 1 add div
    mul \R\space\petitR\space div mul, PointSymbol=*, PointName=none]{M\n}{N}[N\n]
    \ifnum\n=#1
      \pstCircleOA{M\n}{N\n}\ncline{M\n}{N\n}%
      {\psset{linecolor=red, linewidth=2\pslinewidth}
      \pstArcOAB{M\n}{N\n}{N}\pstArcOAB{O}{P}{N}}
    \fi}}%fin multido-newcommand
\begin{pspicture}(-4.5,-4)(4.5,4.5)
\HypoCyclo[4]{4}{1}{27}
\psset{linecolor=blue, linewidth=1.5\pslinewidth}
\pstGenericCurve[GenCurvFirst=P]{N}{1}{7}
\pstGenericCurve{N}{7}{14}\pstGenericCurve{N}{14}{21}
\pstGenericCurve[GenCurvLast=P]{N}{21}{27}
\end{pspicture}
```

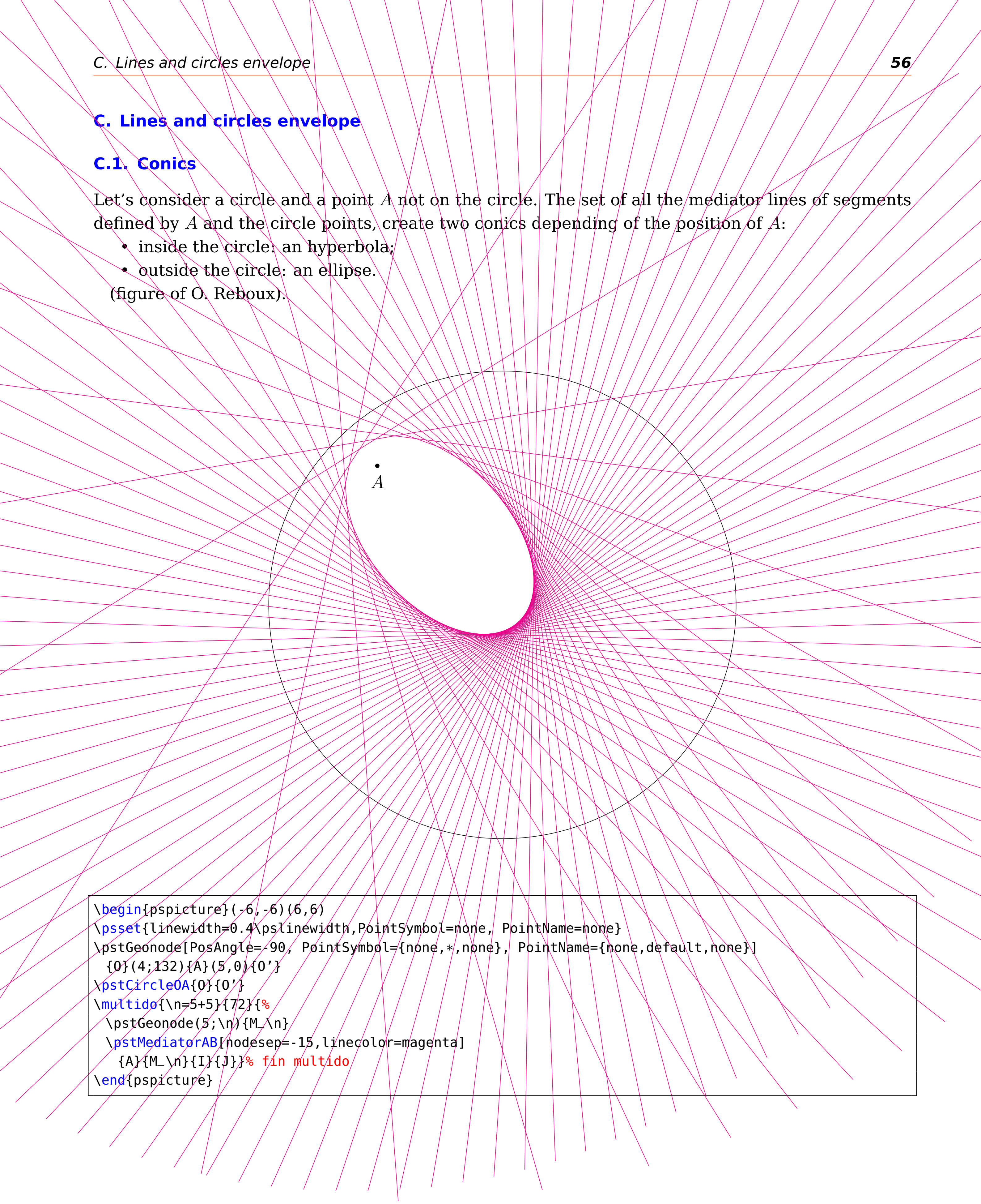
C. Lines and circles envelope

C.1. Conics

Let's consider a circle and a point A not on the circle. The set of all the mediator lines of segments defined by A and the circle points, create two conics depending of the position of A:

- inside the circle: an hyperbola;
- outside the circle: an ellipse.

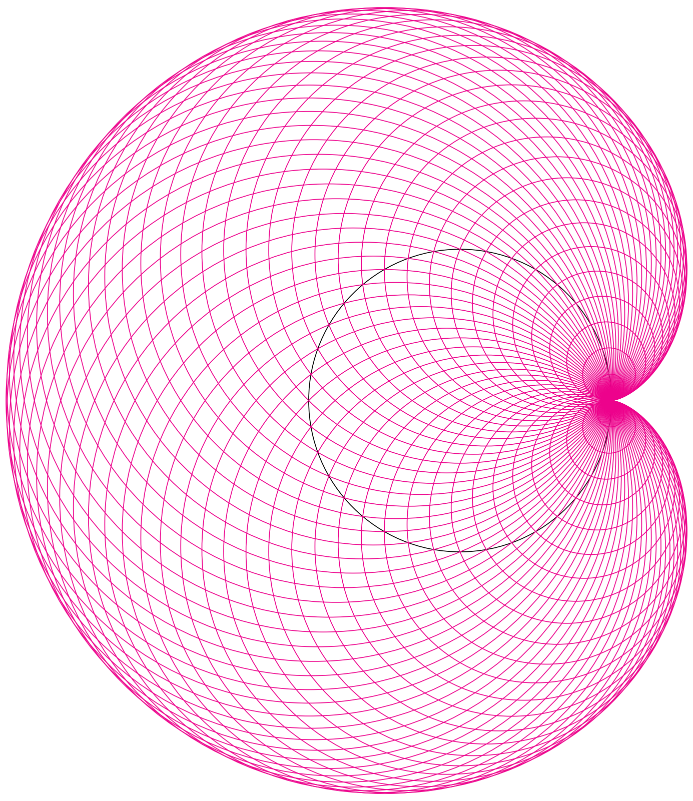
(figure of O. Reboux).



```
\begin{pspicture}(-6,-6)(6,6)
\psset{linewidth=0.4\pslinewidth,PointSymbol=none, PointName=none}
\pstGeonode[PosAngle=-90, PointSymbol={none,*,none}, PointName={none,default,none}]
  {O}(4;132){A}(5,0){O'}
\pstCircleOA{O}{O'}
\multido{\n=5+5}{72}{%
  \pstGeonode(5;\n){M_\n}
  \pstMediatorAB[nodesep=-15,linecolor=magenta]
    {A}{M_\n}{I}{J}}% fin multido
\end{pspicture}
```

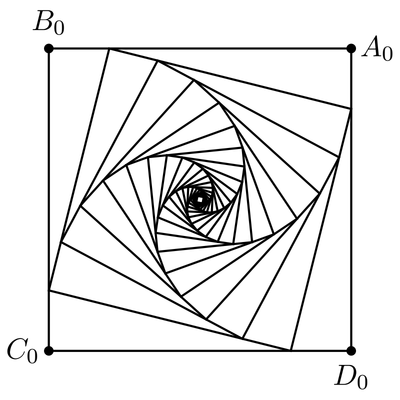
C.2. Cardioid

The cardioid is defined by the circles centered on a circle and crossing a given point.



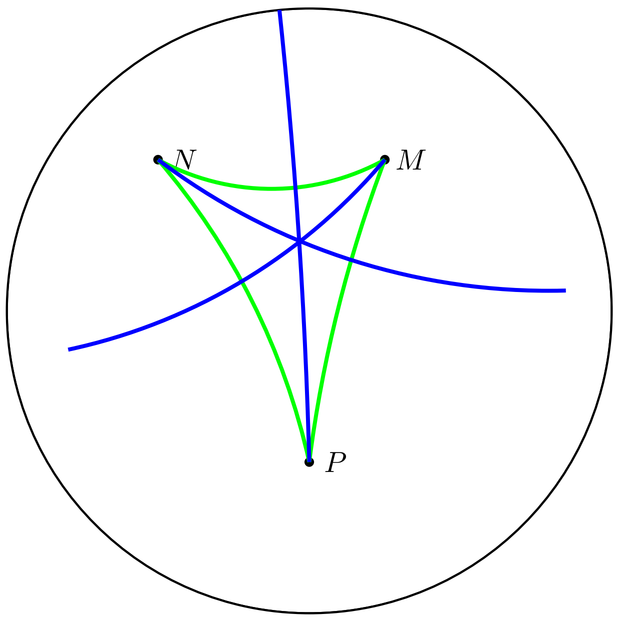
```
\begin{pspicture}(-6,-6)(3,5)
\psset{linewidth=0.4\pslinewidth,PointSymbol=x,nodesep=0,linecolor=magenta}
\pstGeonode[PointName=none]{O}(2,0){O'}
\pstCircleOA[linecolor=black]{O}{O'}
\multido{\n=5+5}{72}{%
  \pstGeonode[PointSymbol=none, PointName=none](2;\n){M_\n}
  \pstCircleOA{M_\n}{O'}}
  \end{pspicture}
```

D. Homotethy and fractals



```
\begin{pspicture}(-2.8,-3)(2.8,3)
\pstGeonode[PosAngle={0,90}](2,2){A_0}(-2,2){B_0}%
\psset{RotAngle=90}
\pstRotation[PosAngle=270]{A_0}{B_0}[D_0]
\pstRotation[PosAngle=180]{D_0}{A_0}[C_0]
\pspolygon(A_0)(B_0)(C_0)(D_0)%
\psset{PointSymbol=none, PointName=none, HomCoef=.2}
\multido{\n=1+1,\i=0+1}{20}{%
  \pstHomO[PosAngle=0]{B_\i}{A_\i}[A_\n]
  \pstHomO[PosAngle=90]{C_\i}{B_\i}[B_\n]
  \pstHomO[PosAngle=180]{D_\i}{C_\i}[C_\n]
  \pstHomO[PosAngle=270]{A_\i}{D_\i}[D_\n]
  \pspolygon(A_\n)(B_\n)(C_\n)(D_\n)}% fin multido
\end{pspicture}
```

E. hyperbolic geometry: a triangle and its altitudes



```
\begin{pspicture}(-5,-5)(5,5)
\psclip{\pscircle(0,0){4}}
  \pstGeonode(1, 2){M}\pstGeonode(-2,2){N}\pstGeonode(0,-2){P}
  \psset{DrawCirABC=false, PointSymbol=none, PointName=none}%
  \pstGeonode(0,0){O}\pstGeonode(4,0){A}\pstCircleOA{O}{A}
  \pstHomO[HomCoef=\pstDistAB{O}{A} 2 mul \pstDistAB{O}{M} sub
    \pstDistAB{O}{M} div]{O}{M}[M']%
  \pstHomO[HomCoef=\pstDistAB{O}{A} 2 mul \pstDistAB{O}{P} sub
    \pstDistAB{O}{P} div]{O}{P}[P']%
  \pstHomO[HomCoef=\pstDistAB{O}{A} 2 mul \pstDistAB{O}{N} sub
    \pstDistAB{O}{N} div]{O}{N}[N']%
  \psset{linecolor=green, linewidth=1.5pt}%
  \pstCircleABC{M}{N}{M'}{OmegaMN}\pstArcOAB{OmegaMN}{N}{M}
  \pstCircleABC{M}{P}{M'}{OmegaMP}\pstArcOAB{OmegaMP}{M}{P}
  \pstCircleABC{N}{P}{P'}{OmegaNP}\pstArcOAB{OmegaNP}{P}{N}
  \psset{linecolor=blue}
  \pstHomO[HomCoef=\pstDistAB{OmegaNP}{N} 2 mul \pstDistAB{OmegaNP}{M} sub %% M
    \pstDistAB{OmegaNP}{M} div]{OmegaNP}{M}[MH']
  \pstCircleABC{M}{M'}{MH'}{OmegaMH}\pstArcOAB{OmegaMH}{MH'}{M} %% N
  \pstHomO[HomCoef=\pstDistAB{OmegaMP}{M} 2 mul \pstDistAB{OmegaMP}{N} sub
    \pstDistAB{OmegaMP}{N} div]{OmegaMP}{N}[NH']
  \pstCircleABC{N}{N'}{NH'}{OmegaNH}\pstArcOAB{OmegaNH}{N}{NH'} %% P
  \pstHomO[HomCoef=\pstDistAB{OmegaMN}{M} 2 mul \pstDistAB{OmegaMN}{P} sub
    \pstDistAB{OmegaMN}{P} div]{OmegaMN}{P}[PH']
  \pstCircleABC{P}{P'}{PH'}{OmegaPH}\pstArcOAB{OmegaPH}{P}{PH'}
\endpsclip
\end{pspicture}
```

F. List of all optional arguments for pst-eucl

Key	Type	Default
PointSymbol	ordinary	*
PointSymbolA	ordinary	*
PointSymbolB	ordinary	*
PointSymbolC	ordinary	*
PointName	ordinary	default
PointNameA	ordinary	undef
PointNameB	ordinary	undef
PointNameC	ordinary	undef
PtNameMath	ordinary	false
PointNameSize	ordinary	\normalsize
PointNameMathSize	ordinary	\textnormal
SegmentSymbol	ordinary	MarkHashh
SegmentSymbolA	ordinary	MarkHashh
SegmentSymbolB	ordinary	MarkHashh
SegmentSymbolC	ordinary	MarkHashh
Mark	ordinary	undef
mark	ordinary	undef
MarkAngle	ordinary	undef
MarkHashLength	ordinary	1.25mm
MarkHashSep	ordinary	0.625mm
PointNameSep	ordinary	[none]
PosAngle	ordinary	[none]
PosAngleA	ordinary	undef
PosAngleB	ordinary	undef
PosAngleC	ordinary	undef
RightAngleSize	ordinary	4
RightAngleType	ordinary	default
MarkAngleRadius	ordinary	0.4
MarkAngleType	ordinary	default
LabelAngleOffset	ordinary	0
LabelSep	ordinary	1
LabelRefPt	ordinary	c
CurveType	ordinary	none
HomCoef	ordinary	0.5
RotAngle	ordinary	60
TransformLabel	ordinary	none
Central@Sym	ordinary	false
DrawCirABC	ordinary	true
CodeFig	boolean	true
CodeFigA	ordinary	undef
CodeFigB	ordinary	undef
CodeFigColor	ordinary	cyan
CodeFigStyle	ordinary	dashed
CodeFigAarc	ordinary	true

Continued on next page

Continued from previous page

Key	Type	Default
`CodeFigBarc`	`ordinary`	`true`
`Radius`	`ordinary`	`none`
`RadiusA`	`ordinary`	`undef`
`RadiusB`	`ordinary`	`undef`
`Diameter`	`ordinary`	`none`
`DiameterA`	`ordinary`	`undef`
`DiameterB`	`ordinary`	`undef`
`DistCoef`	`ordinary`	`none`
`AngleCoef`	`ordinary`	`none`
`CurvAbsNeg`	`ordinary`	`false`
`GenCurvFirst`	`ordinary`	`none`
`GenCurvLast`	`ordinary`	`none`
`GenCurvInc`	`ordinary`	`1`
`AngleValue`	`boolean`	`false`
`AngleArc`	`boolean`	`true`
`ShowWedge`	`boolean`	`true`
`ArcColor`	`ordinary`	`[none]`
`ArcLinestyle`	`ordinary`	`[none]`
`ArcLinewidth`	`ordinary`	`[none]`
`WedgeColor`	`ordinary`	`[none]`
`WedgeFillstyle`	`ordinary`	`[none]`
`WedgeOpacity`	`ordinary`	`[none]`

References

[1] Victor Eijkhout. *TEX by Topic – A TEXnician Reference*. 1st ed. Heidelberg/Berlin: DANTE – lehmanns media, 2014.

[2] Denis Girou. “Présentation de PSTricks”. In: *Cahier GUTenberg* 16 (Apr. 1994), pp. 21–70.

[3] Michel Goosens et al. *The LATEX Graphics Companion*. second. Boston, Mass.: Addison-Wesley Publishing Company, 2007.

[4] Nikolai G. Kollock. *PostScript richtig eingesetzt: vom Konzept zum praktischen Einsatz*. Vaterstetten: IWT, 1989.

[5] Timothy Van Zandt, Herbert Voß, and Rolf Niepraschk. *The Multido package. A loop facility for Generic TeX*. Version 1.42. URL: `macros/latex/multido` (visited on 09/01/2018).

[6] Herbert Voß. “Die mathematischen Funktionen von Postscript”. In: *Die TEXnische Komödie* 1/02 (Mar. 2002), pp. 40–47.

[7] Herbert Voß. *Presentations with LATEX*. 1st ed. Heidelberg/Berlin: DANTE – Lehmanns Media, 2012.

[8] Herbert Voß. *PSTricks – Grafik für TEX und LATEX*. 7th ed. Heidelberg/Hamburg: DANTE – Lehmanns, 2016.

[9] Herbert Voß. *PSTricks – Graphics and PostScript for LATEX*. 1st ed. Cambridge – UK: UIT, 2011.

[10] Herbert Voß. *LATEX quick reference*. 1st ed. Cambridge – UK: UIT, 2012.

[11] Timothy Van Zandt and Denis Girou. “Inside PSTricks”. In: *TUGboat* 15 (Sept. 1994), pp. 239–246.

Index