The \texttt{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:
\begin{itemize}
\item Denis Girou pour ses critiques pertinentes et ses encouragement lors de la découverte de l’embryon initial et pour sa relecture du présent manuel;
\item Michael Vulis for his fast testing of the documentation using \TeX\ which leads to the correction of a bug in the PostScript code;
\item Manuel Luque and Olivier Reboux for their remarks and their examples.
\item Alain Delplanque for its modification theorems on automatic placing of points name and the ability of giving a list of points in \texttt{\pstGeonode}.
\end{itemize}

Thanks to: Manuel Luque; Thomas Söll.
## Contents

1. The package  
   1. Special specifications  
      1.1. PSTricks Options  
      1.2. Conventions  
   2. Basic Objects  
      2.1. Points  
      2.2. Segment mark  
      2.3. Segment labels  
      2.4. Triangles  
      2.5. Angles  
      2.6. Lines, half-lines and segments  
      2.7. Distance  
      2.8. Circles  
      2.9. Circle arcs  
      2.10. Circle nodes  
      2.11. Circle tangent  
      2.12. Circle radical axis  
      2.13. Regular polygons  
      2.14. Generic curve  
   3. Conics  
      3.1. Standard Ellipse  
      3.2. General Ellipse  
      3.3. Standard Parabola  
      3.4. Standard Conjugate Parabola  
      3.5. General Parabola  
      3.6. General Conjugate Parabola  
      3.7. Standard Hyperbola  
      3.8. Standard Conjugate Hyperbola  
      3.9. General Hyperbola  
      3.10. General Conjugate Hyperbola  
      3.11. General Conics  
   4. Geometric Transformations  
      4.1. Central symmetry  
      4.2. Orthogonal (or axial) symmetry  
      4.3. Rotation  
      4.4. Translation  
      4.5. Homothetie  
      4.6. Orthogonal projection  
   5. Special object  
      5.1. Midpoint  
      5.2. Triangle center of gravity  
      5.3. Centre of the circumcircle of a triangle

## 1. The package

1. Special specifications
   1.1. PSTricks Options
   1.2. Conventions

## 2. Basic Objects

2.1. Points
2.2. Segment mark
2.3. Segment labels
2.4. Triangles
2.5. Angles
2.6. Lines, half-lines and segments
2.7. Distance
2.8. Circles
2.9. Circle arcs
2.10. Circle nodes
2.11. Circle tangent
2.12. Circle radical axis
2.13. Regular polygons
2.14. Generic curve

## 3. Conics

3.1. Standard Ellipse
3.2. General Ellipse
3.3. Standard Parabola
3.4. Standard Conjugate Parabola
3.5. General Parabola
3.6. General Conjugate Parabola
3.7. Standard Hyperbola
3.8. Standard Conjugate Hyperbola
3.9. General Hyperbola
3.10. General Conjugate Hyperbola
3.11. General Conics

## 4. Geometric Transformations

4.1. Central symmetry
4.2. Orthogonal (or axial) symmetry
4.3. Rotation
4.4. Translation
4.5. Homothetie
4.6. Orthogonal projection

## 5. Special object

5.1. Midpoint
5.2. Triangle center of gravity
5.3. Centre of the circumcircle of a triangle
5.4. Perpendicular bisector of a segment ........................................... 82
5.5. Bisectors of angles ................................................................. 82

6. Intersections 83
   6.1. Line-Line ............................................................................... 83
   6.2. Circle–Line ........................................................................... 84
   6.3. Circle–Circle ......................................................................... 84
   6.4. Function–function ................................................................. 85
   6.5. Function–line ......................................................................... 86
   6.6. Function–Circle ..................................................................... 86

7. Helper Macros 87
II. Examples gallery

A. Basic geometry
A.1. Drawing of the bissector .................................................. 89
A.2. Transformation de polygones et courbes ................................. 90
A.3. Triangle lines ................................................................. 91
A.4. Euler circle ................................................................. 92
A.5. Orthocenter and hyperbola .............................................. 93
A.6. 17 sides regular polygon .................................................. 94
A.7. Circles & tangents ......................................................... 96
A.8. Fermat’s point .............................................................. 97
A.9. Escribed and inscribed circles of a triangle ......................... 98

B. Some locus points
B.1. Parabola ................................................................. 100
B.2. Hyperbola ................................................................. 101
B.3. Cycloid ................................................................. 101
B.4. Hypocycloids (Astroid and Deltoid) .................................. 103

C. Lines and circles envelope
C.1. Conics ................................................................. 105
C.2. Cardioid ................................................................. 106

D. Homotethy and fractals .................................................... 107

E. hyperbolic geometry: a triangle and its altitudes ...................... 108

F. List of all optional arguments for pst-eucl ............................... 109

References ................................................................. 110
Part I.
The package

1. Special specifications

1.1. PStricks Options

The package activates the \SpecialCoor mode. This mode extends 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 settings.

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}... (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:

- $*$: ●
- 0: ○
- +: ⊕
- 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 differentiate 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.

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.

\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){B_2}
\pstGeonode[PosAngle=90,0,-90, PointSymbol={$\ast$,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={$\ast$,x,\ast}, PointNameSep=3mm, CurveType=curve, PointName={$\alpha$,$\beta$,$\gamma$,default} ](-2,0){alpha}(-1,-2){beta}(0,-1){gamma}(2,-1.5){T}
\end{pspicture}

\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){B_2}
\pstGeonode[PosAngle=90,0,-90, PointSymbol={$\ast$,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={$\ast$,x,\ast}, PointNameSep=3mm, CurveType=curve, PointName={$\alpha$,$\beta$,$\gamma$,default} ](-2,0){alpha}(-1,-2){beta}(0,-1){gamma}(2,-1.5){T}
\end{pspicture}

\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){B_2}
\pstGeonode[PosAngle=90,0,-90, PointSymbol={$\ast$,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={$\ast$,x,\ast}, PointNameSep=3mm, CurveType=curve, PointName={$\alpha$,$\beta$,$\gamma$,default} ](-2,0){alpha}(-1,-2){beta}(0,-1){gamma}(2,-1.5){T}
\end{pspicture}

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 without concerned the xunit and yunit in the PSTricks SpecialCoor function. You need the other third package to do float arithmetic operation, like \pscalculate \footnote{Provided by package pst-calculate, sometimes it results the numbers more than 9 fraction digits, which are not supported good by PSTricks with ‘! number too big’ issue.} to generate the numerical values, or the expandable command \fpeval \footnote{Provided by package xfp, it can truncate the fraction part digits using the trunc function perfectly, e.g. \fpeval{trunc(18/7,3)}.} to get a purely numerical result.

The macro \pstMoveNode use them to move node A by abscissa increment $dx$ and ordinate increment $dy$ to get the target node B.
\begin{pspicture}[showgrid=true](0,0)(4,4)
\pstMoveNode[dx=1,dy=2]{A}{B}
\pstLineAB{A}{B}
\pstLineAB{B}{C}
\pstLineAB{C}{D}
\pstLineAB{D}{E}
\end{pspicture}

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

\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{unit=.7}
\begin{pspicture*}[showgrid=true](-4,-4)(4,4)
\pstMoveNode[dx=1,dy=2]{A}{B}
\pstLineAB{A}{B}
\pstLineAB{B}{C}
\pstLineAB{C}{D}
\pstLineAB{D}{E}
\end{pspicture*}
\end{pspicture}

2.2. Segment mark

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

\begin{pspicture}[showgrid=true](0,0)(4,4)
\pstSegmentMark[dx=1,dy=2]{A}{B}
\end{pspicture}

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

Several commands are predefined for marking the segment:

- \texttt{psslash} \quad \texttt{psslashh} \quad \texttt{MarkHashh}
- \texttt{psslash} \quad \texttt{psslashh} \quad \texttt{MarkHashh}
\begin{pspicture}[-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[MarkHashSep=90]{E}{A} 
\end{pspicture}

The three commands of the family \texttt{MarkHash} draw a line whose inclination is controled by the parameter \texttt{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.

\subsection*{2.3. Segment labels}

According to the manual of \texttt{PSTricks}, you can use the macros \texttt{\naput}, \texttt{\ncput} and \texttt{\nbput} to put the label \texttt{above}, \texttt{cover}, \texttt{below} the segment. The macro \texttt{\psetLabelAB} just use them to draw a ruler bar and put the label on the ruler bar.

\begin{verbatim}
\psetLabelAB [+] [Options] {A}{B}{label}
\end{verbatim}

You can use the parameters of \texttt{\ncline} to control the ruler bar, such as \texttt{linestyle}, \texttt{linecolor}, \texttt{linewidth}, \texttt{arrows}, \texttt{nodelinewidth} etc; and use the parameters of \texttt{\ncput} to control the label position, such as \texttt{nrot}, \texttt{npos} etc; and use the parameters of \texttt{\ncline} to control the separation between the ruler bar and the segment.

It does not display the ruler bar as default, and you need to setup \texttt{linestyle} to display it. The star version uses also the star version of the put macro (white background).

\begin{verbatim}
\begin{pspicture}[-2,-2](4,4) 
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize 
\pstGeonode[PosAngle=\{-90\}]{0,1.5}{A} 
\pstGeonode[PosAngle=\{-90\}]{2.5,1.5}{B}\pstLineAB{A}{B} 
\psset{linestyle=\linewidth}{a}{b} 
\pstGeonode[PosAngle=\{0\}]{0.5,0.5}{C} 
\pstGeonode[PosAngle=\{0\}]{3,3}{D}\pstLineAB{C}{D} 
\psset{linestyle=dashed}{D}{E}\sqrt{a^2+b^2}\psetLabelAB{D}{E}{\sqrt{a^2+b^2}} 
\psset{linestyle=dashed,arrows=\{<\}}\psset{\linewidth=\pslinewidth}{\psframebox[\pslinewidth]{A}} 
\end{pspicture}
\end{verbatim}
2.4. Triangles

The more classical figure, it has its own macro \texttt{\textbackslash pstTriangle} for a quick definition:

\begin{verbatim}
\pstTriangle[Options] (x_1, y_1){A}(x_2, y_2){B}(x_3, y_3){C}
\end{verbatim}

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 (A), (B) and (C). 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 bisector 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.

The macros \texttt{\textbackslash pstTriangleSSS}, \texttt{\textbackslash pstTriangleSAS}, \texttt{\textbackslash pstTriangleAAS} and \texttt{\textbackslash pstTriangleASA} are used to draw the triangle according the specified sides or angles.

\begin{verbatim}
\pstTriangleSSS[Options] (pos){A}(a,b,c){B}{C}
\end{verbatim}

- \texttt{\textbackslash pstTriangleSSS} create a triangle $ABC$ with given $A(x_1, y_1)$, and the three sides $a, b, c$, it output $B(x_2, y_2)$ and $C(x_3, y_3)$.
- \texttt{\textbackslash pstTriangleSAS} create a triangle $ABC$ with given $A(x_1, y_1)$, the angle of $\angle A$, and the other two sides $b, c$, it output $B(x_2, y_2)$ and $C(x_3, y_3)$.
- \texttt{\textbackslash pstTriangleAAS} create a triangle $ABC$ with given $A(x_1, y_1)$, the angle of $\angle C$, the angle of $\angle A$, and the side of $AB = c$, it output $B(x_2, y_2)$ and $C(x_3, y_3)$.
- \texttt{\textbackslash pstTriangleASA} create a triangle $ABC$ with given $A(x_1, y_1)$, the angle of $\angle A$, the angle of $\angle B$, and the side of $AB = c$, it output $B(x_2, y_2)$ and $C(x_3, y_3)$.

\begin{verbatim}
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}
\psset{PointSymbol=none}
\footnotesize
\pstGeonode*[PosAngle=-90](0,0){A}
\pstTriangleSSS*[linecolor=red!60,PosAngle={-90,90}]{A}(3,4,5){B}{C}
\pstTriangleSAS*[linecolor=blue!60,PosAngle={-90,90}]{A}(2,4,5,3.8){B}{E}
\pstTriangleSAS*[linecolor=green!60,PosAngle={-90,90}]{A}(3,4,2,8){F}{G}
\pstTriangleSAS*[linecolor=black!60,PosAngle={-90,90}]{A}(40,50,1.8){H}{I}
\pstTriangleAAS*[linecolor=orange!60,PosAngle={-90,90}]{A}(70,1.8,60){J}{K}
\end{pspicture}
\end{verbatim}

The optional parameter pos setup the position of the first node $A$, it should be \texttt{':L'} for left, \texttt{':R'} for right, \texttt{':U'} for up and \texttt{':D'} for down. You also can specify one another node $X$ to draw a triangle base on the line $AX$. If you don’t input this parameter, the default value is \texttt{':L'}. The following example explains how to draw an isosceles triangle with the given isosceles sides and the vertex angle.
2. Basic Objects

The macros \texttt{pstTriangleIC} and \texttt{pstTriangleOC} are used to draw the inner circle and outer circle of triangle $ABC$.

\begin{pspicture}(-2,-2)(2,2)
\pstTriangleIC[\texttt{Options}] {A}{B}{C}[I][H]
\end{pspicture}

\begin{pspicture}(-2,-2)(2,2)
\pstTriangleOC[\texttt{Options}] {A}{B}{C}[0]
\end{pspicture}

The center of the inner circle is called IC, as default and the outer circle OC, as default, but you can change the node names by the optional parameters [I] [H] and [0]. The optional node name H is a node on the inner circle, so you can operate the inner circle by center I and node H later.

The inner center I, node H and outer circle center O are not printed out as default, but you can setup \texttt{PointSymbol} and \texttt{PointName} to display them. For example:

\begin{verbatim}
\pstTriangleIC[\texttt{PosAngle=-90,160}, \texttt{PointName=I,none}, \texttt{PointSymbol=*,none}]{A}{B}{C}[I][D]
\pstTriangleIC[\texttt{PosAngle=-90,PointName=I,PointSymbol=*,none}]{A}{B}{C}[I][D]
\pstTriangleOC[\texttt{PosAngle=90,PointSymbol=*,PointName=X}]{A}{B}{C}[X]
\end{verbatim}

The macros \texttt{pstTriangleGC}, \texttt{pstTriangleHC}, \texttt{pstTriangleEC}, \texttt{pstTriangleNC}, \texttt{pstTriangleLC} are used to draw the barycenter $G$, the orthocentre $H$, the escenter $E$, the nine points circle center and the Lemoinie point (or symmedian point) of the triangle $ABC$.

\begin{verbatim}
\pstTriangleGC[\texttt{Options}] {A}{B}{C}{G}[M_1][M_2]
\pstTriangleHC[\texttt{Options}] {A}{B}{C}{H}[H_1][H_2]
\pstTriangleEC[\texttt{Options}] {A}{B}{C}{E}[T_1]
\pstTriangleNC[\texttt{Options}] {A}{B}{C}{N}[M_1][M_2][M_3]
\pstTriangleLC[\texttt{Options}] {A}{B}{C}{L}[S_1][S_2][S_3]
\end{verbatim}

You can use the options of node like as \texttt{PointName=...}, \texttt{PosAngle=...}, \texttt{PointSymbol=...} to control the output nodes $G, H, E$. But if you give the optional output parameters $M_1, M_2,$ or $H_1, H_2$ or $T_1$, then you should pass the option value in list like as \texttt{PointName=...}, \texttt{PosAngle=...}, \texttt{PointSymbol=...}. For example,
2.5. 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}

The valid optional arguments controlling this command, excepting the ones which controlled the line, are RightAngleType, RightAngleSize, RightAngleRadius, and RightAngleDotDistance.

The symbol is controlled by the parameter RightAngleType default. Its possible values are:
- *: standard symbol;
- german: german symbol (given by U. Durr);
- suisseromand: swiss romand symbol (given P. Schnewlin).

The optional argument RightAngleSize defines the size of the symbol (by default 0.28 unit).

For a right angle style german or swissromand the distance of the dot is preset to 0.5 (german) or 0.45 (swissromand), relative to the radius. However, it can be controlled by the optional argument RightAngleDotDistance which is preset to 1. A greater value moves the dot away from the reference point.

For other angles, there is the command:

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

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 0). 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.
2. Basic Objects

\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){A'}(2;15){B'}(2;195){C'}
\psRightAngle[linecolor=red,fillstyle=solid,fillcolor=blue]{C}{B}{A}
\psRightAngle[linecolor=blue,RightAngleType=suisseromand]{A}{B'}{C}
\psRightAngle[linecolor=magenta,RightAngleType=german]{A}{B''}{C}
\psset{arcsep=pslinewidth}
\pstMarkAngle[linecolor=cyan,Mark=MarkHash]{C}{B}{A}{$\theta$}
\psMarkAngle[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}
\psGeonode(1,2){A}(1,1){B}(2,1){C}(2,2){D}
\psRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\psRightAngle{B}{A}{D} \pspolygon(A)(B)(C)(D)
\psset{RightAngleType=suisseromand}
\psGeonode(3,2){A}(3,1){B}(4,1){C}(4,2){D}
\psRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\psRightAngle{D}{C}{B} \psRightAngle(A){D}{C}
\psGeonode(5,2){A}(5,1){B}(6,1){C}(6,2){D}
\psRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\psRightAngle{D}{C}{B} \psRightAngle(A){D}{C}
\psset{RightAngleType=german}
\psGeonode(5,1){A}(5,0){B}(6,0){C}(6,1){D}
\psRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\psRightAngle{D}{C}{B} \psRightAngle(A){D}{C}
\psGeonode(5,0){A}(5,-1){B}(6,-1){C}(6,0){D}
\psRightAngle[fillstyle=solid,fillcolor=blue!40]{C}{B}{A}
\psRightAngle{D}{C}{B} \psRightAngle(A){D}{C}
\end{pspicture}
2. Basic Objects

2.6. Lines, half-lines and segments

The classical line \(AB\)!

\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}{$\theta$}
\end{pspicture}

\begin{pspicture}[showgrid=false](-2,-2)(2,2)
\pstGeonode(1,1){A}(-1,-1){B}
\pstLineAB[Options][]{A}{B}
\pstLineAB[Options][]{B}{C}
\pstLineAB[Options][]{C}{A}
\pstLineAB[Options][]{C}{D}
\end{pspicture}

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

---

\[3 \text{ which is the comble for a line!}\]
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.

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

Here are some examples:

\begin{verbatim}
\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}
\end{verbatim}

The macro \pstLineCoef is used to draw a line $ax + by + c = 0$ with the given coefficients $a, b, c$, and create two new node $A, B$ on the line.

\begin{verbatim}
\pstLineCoef [Options] {a,b,c}{A}{B}
\end{verbatim}

Here are some examples:

\begin{verbatim}
\begin{pspicture}[showgrid=true](-2,-2)(2,2)
\pstLineCoef[linestyle=dashed, linecolor=red!60, PosAngle=(210,0)]{3,-2,1}{A}{B}
\pstLineCoef[linestyle=dashed, linecolor=blue!60, PosAngle=(180,0)]{4,3,2}{C}{D}
\pstLineCoef[linestyle=dashed, linecolor=green!60, PosAngle=(90,90)]{0,3,-3}{E}{F}
\pstLineCoef[linestyle=dashed, linecolor=purple!160, PosAngle=(180,180)]{4,0,4}{G}{H}
\end{pspicture}
\end{verbatim}

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 or \fpeval to get a purely numerical result, or use \pstAbscissa and \pstOrdinate to get the abscissa and ordinate of any other node.
For example,

The macro \texttt{\textbackslash{p}stProportionNode} creates the nodes $C$ and $C'$ on segment $AB$ which are satisfied $|AC| : |BC| = \lambda$. ($\lambda > 0$). The node $C$ is inside the segment $AB$ and the node $C'$ is outside the segment $AB$, we have

\begin{align*}
x_C &= \frac{x_A + \lambda x_B}{1 + \lambda} \\
y_C &= \frac{y_A + \lambda y_B}{1 + \lambda}
\end{align*}

and

\begin{align*}
x_{C'} &= \frac{x_A - \lambda x_B}{1 - \lambda} \\
y_{C'} &= \frac{y_A - \lambda y_B}{1 - \lambda}
\end{align*}

One application of \texttt{\textbackslash{p}stProportionNode} is used to find the bisector and out bisector of a given angle. So we define the macro \texttt{\textbackslash{p}stBisectorAOB} to do this work, it is more friendly than the macros \texttt{\textbackslash{p}stBissectBAC} and \texttt{\textbackslash{p}stOutBissectBAC}, as it put the new node $T_1$ and $T_2$ on line $AB$, not arc $AB$. 

\begin{verbatim}
\pstProportionNode[Options] {A}{B} {\lambda} {C}{C'}
\end{verbatim}

\begin{verbatim}
\pstDistDiv[Options] {A}{B}{x_1}{C}
\end{verbatim}
The four collinear points $A, B, C, D$ are called Harmonic Conjugation Points if their cross ratio is $-1$, that is

$$(AB, CD) = \frac{AC}{BC} \cdot \frac{AD}{BD} = -1$$

If given three collinear points $A, B, C$, how can we get the fourth harmonic point? The following macro $\texttt{pstFourthHarmonicNode}$ is used to get the fourth harmonic point. It creates a new node $X$ on the same line, but when $A, B, C$ are not collinear, we put it at origin.

\begin{verbatim}
\pstFourthHarmonicNode[Options] {A}{B}{C}{X}
\end{verbatim}

If you want to draw a node like ‘Given $EF$, please find node $C$ on $AB$ such that $AC = EF$’, you can use the macro $\texttt{pstLocateAB}$ to do this, it can seek the node $C$ from $A$ to $B$ with the specified length $L$, which can be got from $\texttt{pstDist, pstDistConst, pstDistAdd, pstDistSub}$, etc.

\begin{verbatim}
\pstLocateAB[Options] {A}{B}{L}{C}
\end{verbatim}

Note that seek from $B$ will get the node $C$ in the reverse order, for example,
If you want to draw a node like ‘Given $EF$, please extend $AB$ to $C$ such that $BC = EF’$, you can use the macro $\texttt{pstExtendAB}$ to do this; it can extend $AB$ from $B$ to one node with the specified length $L$, which can be got from $\texttt{pstDist}$, $\texttt{pstDistConst}$, $\texttt{pstDistAdd}$, $\texttt{pstDistSub}$, etc.

\begin{verbatim}
\pstExtendAB[Options] {A}{B}{L}{C}
\end{verbatim}

Note that extend $BA$ to $C$ will get the node $C$ in the reverse order; for example,

\begin{verbatim}
\pstExtendAB[Options] {B}{A}{L}{C}
\end{verbatim}

You can find the node $C$ on segment $AB$ satisfied $|AC|:|AB|=\text{DistCoef}$ using $\texttt{pstTranslation}$, but it can’t do the same thing like $\texttt{pstLocateAB}$ and $\texttt{pstExtendAB}$ when the given segment $EF$ is not parallel with $AB$, it will be introduced in the later sections.

If you want to find the inversion point $C’$ of $C$ to the inversion center $O$ with inversion radius $R$,
that is, the point \(C'\) is satisfied the inversion transform equation

\[
|OC| \times |OC'| = R^2
\]
you can use the macro \texttt{pstInversion} to do this work. In fact, we use the macro \texttt{pstLocateAB} to implement this macro by passing the value \(\frac{R^2}{|OC|}\) to parameter length.

\[
\texttt{pstInversion [Options]} \{O\} \{A\} \{C\} \{C'\}
\]

It is possible to omit the parameter \(A\) and then to specify the inversion radius or the inversion diameter using the parameters \(\text{Radius}\) and \(\text{Diameter}\), which will be introduced in the next section.

It is clear that the inversion mapping of a line is a circle, and the inversion mapping of a point on the inversion circle is itself.

\[
\begin{align*}
\begin{pspicture}(-1,-2)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.5}
\pstGeonode[PosAngle=180,\text{\texttt{PosAngle}=0}]\{O\}{1,1}{0}
\pstCircleOA[\text{\texttt{linecolor}=red!50,\text{\texttt{Radius}=\texttt{pstDistVal}\{\ra\}}}]\{O\}{0}
\pstCircleOA[\text{\texttt{PosAngle}=180,\text{\texttt{RotAngle}=180,\text{\texttt{Radius}=\texttt{pstDistVal}\{\ra\}}}}] \{O\}{0}
\pstInversion[\text{\texttt{PosAngle}=0,\text{\texttt{Radius}=\texttt{pstDistVal}\{\ra\}}}]\{O\}\{A\}\{A'\}
\pstInversion[\text{\texttt{PosAngle}=180,\text{\texttt{Radius}=\texttt{pstDistVal}\{\ra\}}}] \{O\}\{C\}\{C'\}
\pstInversion[\text{\texttt{PosAngle}=180,\text{\texttt{RotAngle}=180,\text{\texttt{Radius}=\texttt{pstDistVal}\{\ra\}}}}] \{O\}\{D\}\{D'\}
\pstInversion[\text{\texttt{PosAngle}=90,\text{\texttt{Radius}=\texttt{pstDistVal}\{\ra\}}}] \{O\}\{E\}\{E'\}
\pstInversion[\text{\texttt{PosAngle}=180,\text{\texttt{Radius}=\texttt{pstDistVal}\{\ra\}}}] \{O\}\{F\}\{F'\}
\pstLineAB[\text{\texttt{linecolor}=blue!40,\text{\texttt{PosAngle}=0}]}\{O\}\{C'\}
\psdots[\text{\texttt{dotsize}=5}](3,3)
\psline[\text{\texttt{linecolor}=black!50}](3,3)(\ra){\text{\texttt{D}}}(\ra){\text{\texttt{E}}}
\end{pspicture}
\end{align*}
\]

If you want to find the node \(C\) from \(A\) to \(B\), such that \(C\) is the golden section of the given segments \(AB\), that is,

\[
|AC|^2 = |AB| \times |BC| \quad \text{or} \quad AC : AB = BC : AC \quad \text{or} \quad AC = \frac{\sqrt{5}-1}{2}AB
\]
you can use the macro \texttt{pstGoldenMean} to do this work.

\[
\texttt{pstGoldenMean [Options]} \{A\} \{B\} \{C\}
\]

In fact, we use the macro \texttt{pstLocateAB} to implement this macro by passing the value \(\frac{\sqrt{5}-1}{2}|AB|\) to parameter length.
If you want to find the node $C$ from $A$ to $B$, such that $AC$ is the geometric mean of two given segments $DE$ of $FG$, that is,

$$|AC|^2 = |DE| \times |FG|$$

you can use the macro \texttt{\pstGeometricMean} to do this work. It also can be used to draw a circle when given two points on the circle, and a line tangents to the circle.

\begin{verbatim}
\pstGeometricMean [Options] \{A\}\{B\}\{L_1\}\{L_2\}\{C\}
\end{verbatim}

In fact, we use the macro \texttt{\pstLocateAB} to implement this macro by passing the value $\sqrt{L_1 \times L_2}$ to parameter length. The length $L_1$ and $L_2$ can be got from \texttt{\pstDist}, \texttt{\pstDistConst}, \texttt{\pstDistAdd}, \texttt{\pstDistSub}, etc.

\begin{verbatim}
\pstLocateAB [Options] \{A\}\{B\}\{O\}\{L_1\}[\{L_2\}]
\end{verbatim}

If you want to find the node $C$ from $A$ to $B$, such that $AC$ is the harmonic mean of two given segments $DE$ of $FG$, that is,

$$\frac{1}{|AC|} = \frac{1}{2} \left( \frac{1}{|DE|} + \frac{1}{|FG|} \right)$$

you can use the macro \texttt{\pstHarmonicMean} to do this work.

\begin{verbatim}
\pstHarmonicMean [Options] \{A\}\{B\}\{L_1\}\{L_2\}\{C\}
\end{verbatim}

In fact, we use the macro \texttt{\pstLocateAB} to implement this macro by passing the value $\frac{2L_1L_2}{L_1 + L_2}$ to parameter length. The length $L_1$ and $L_2$ can be got from \texttt{\pstDist}, \texttt{\pstDistConst}, \texttt{\pstDistAdd}, \texttt{\pstDistSub}, etc.
2.7. Distance

Like as coordinates, the distance works at the PostScript level, that is, it should be used where the code is interpreted by PostScript engine, but not \TeX engine. There were three macros to operate the distance before v1.66:

\begin{verbatim}
\pstDistAB{A}{B}
\pstDistVal{l}
\pstDistCalc{expr}
\end{verbatim}

The first specifies a distance between two points. The second macro can be used to specify an explicit numerical value \(l\), which is in User coordinate. The third one uses the \pscalculate to calculate the result of the input expression, which is in User coordinate too. The parameter DistCoef can be used to specify a coefficient to reduce or enlarge the result distance. This parameter will come into effect if it is specified before these macros.

After v1.66, We provide three macros which disable the effect of parameter DistCoef one to one as following:

\begin{verbatim}
\pstDist{A}{B}
\pstDistConst{l}
\pstDistExpr{expr}
\end{verbatim}

We provide the macro \pstDistCoef to reduce or enlarge a given distance explicitly, for example: \\pstDistCoef{\pstDist{A}{B}}, or use macro \pstDistMul to multiply the input coefficient.

**Note:** The series of macros \pstDist* get the length result in the Screen coordinate, so you need to convert the length to the User coordinate by macro \pstUserDist, when use them where need the user coordinate numbers, e.g,

\begin{verbatim}
\pnodetwo{1}{\pstUserDist{\pstDistAdd{A}{B}{C}{D}}}{A}
\pstMoveNode{0}{\pstUserDist{\pstDistAdd{A}{B}{C}{D}}}{A}{E}
\end{verbatim}

You can convert the distance in User coordinate to Screen coordinate by macro \pstScreenDist, it is just another name of \pstDistConst. As we said before, macros \pstAbscissa and \pstOrdinate give the coordinate of one node in User coordinate, so if you want to draw a circle using them, you should type:

\begin{verbatim}
\pstCircleOA[Radius=\pstDistConst{\pstAbscissa{A}}]{A}
\end{verbatim}
It is possible to use the raw PostScript command to make more complex arithmetic operations. In order to hide the lower level PostScript language, we add more macros for distance addition and subtraction, such as \pstDistAdd[Val/Coef] and \pstDistSub[Val/Coef], etc. These macros can be used to calculate the Radius or Diameter to define a circle.

The macros \pstDistAdd and \pstDistSub are used to get the addition and subtraction of the given segments $AB$ and $CD$. The macro \pstDistDiv is used to get the length ratio of the given segments $AB$ and $CD$, you can pass the ratio to macro \pstProportionNode, or setup the ratio to parameter DistCoef in macro \pstTranslation, or pass the ratio to any \pstDist* macros which need a $\lambda$ parameter.

\begin{verbatim}
\pstDistMul{A}{B}{\lambda}
\pstDistAdd{A}{B}{C}{D}
\pstDistAddVal{A}{B}{\lambda}{L}
\pstDistAddCoef{A}{B}{\lambda_1}{C}{D}{\lambda_2}
\pstDistSub{A}{B}{C}{D}
\pstDistSubVal{A}{B}{\lambda}{L}
\pstDistSubCoef{A}{B}{\lambda_1}{C}{D}{\lambda_2}
\pstDistDiv{A}{B}{C}{D}
\end{verbatim}

In these macros, the length $L$ is a numerical value in the Screen Coordinate, so it is possible to pass the result of any macros like \pstDist to it. $\lambda$ is a numerical value to multiply, and most important is that the parameter DistCoef doesn’t take effect any more. It is better to describe in formula:

- macro \pstDistAB get the screen length of DistCoef * $|AB|$
- macro \pstDistVal get the screen length of DistCoef * $l$
- macro \pstDistCalc get the screen length of DistCoef * expr
- macro \pstDistCoef get the screen length of DistCoef * <arg>
- macro \pstDist get the screen length of $|AB|$ 
- macro \pstDistConst get the screen length of $l$
- macro \pstDistExpr get the screen length of expr
- macro \pstDistMul get the screen length of $\lambda|AB|$
- macro \pstDistAdd get the screen length of $|AB| + |CD|$
- macro \pstDistAddVal get the screen length of $\lambda|AB| + L$
- macro \pstDistAddCoef get the screen length of $\lambda_1|AB| + \lambda_2|CD|$
- macro \pstDistSub get the screen length of $\text{abs}(|AB| - |CD|)$
- macro \pstDistSubVal get the screen length of $\text{abs}(\lambda|AB| - L)$
- macro \pstDistSubCoef get the screen length of $\text{abs}(\lambda_1|AB| - \lambda_2|CD|)$
- macro \pstDistDiv get the the ratio of length $|AB| : |CD|$

For example, the following one draw a circle with radius length $2|AB| + 3|CD| + 4|EF|$, it shows how to operate more than two distances.

\begin{verbatim}
\pstCircleOA[Radius=\pstDistAddVal{A}{B}{2.0}{\pstDistAddCoef{C}{D}{3.0}{E}{F}{4.0}}]{A}
\end{verbatim}

Another example is for \pstDistMul, the old code like as

\begin{verbatim}
\pstCircleOA[DistCoef=1 3 \text{ div},Radius=\pstDistAB{A}{B}]{0}
\pstCircleOA[DistCoef=1 3 \text{ div},Radius=\pstDistAB{A}{B}]{A}{B}{0}{1}{J}
\end{verbatim}

\pstInterC[DistCoef=1 3 \text{ div},RadiusA=\pstDistAB{A}{B},DistCoef=mone,RadiusB=\pstDistAB{C}{D}]{01}{02}{1}{J}

could be simplified to

\begin{verbatim}
\pstCircleOA[Radius=\pstDistMul{A}{B}{1 3 \text{ div}}]{0}
\end{verbatim}
2. Basic Objects

```latex
\pstInterLC\[Radius=\pstDistMul\{A\}{B}\{1\ 3\ div\}\{A\}{B}\{0\}\{I\}{J}\]
\pstInterCC\[RadiusA=\pstDistMul\{A\}{B}\{1\ 3\ div\},RadiusB=\pstDistAB\{C\}{D}\{01\}\{02\}\{I\}{J}\]
```

Important! We recommend that you should use the distance macros which disable the parameter DistCoef instead of `\pstDistAB`, `\pstDistVal` or `\pstDistCalc`, when you need to pass their result into `\pstDistAddVal` or `\pstDistSubVal`, as it will give you the error result sometimes. For example, the following code

```latex
\pstDistAddVal\{A\}{B}\{2.0\}\{\pstDistAB\{C\}{D}\}
```

is expected to get the length of $2 |AB| + |CD|$. If current DistCoef is $\lambda$, then it will give the error result as $2 |AB| + \lambda |CD|$. The right way is

```latex
\pstDistAddVal\{A\}{B}\{2.0\}\{\pstDist\{C\}{D}\}
```

At last, we provide a macro named `\pstDistABC` to get the distance from $C$ to line $AB$.

```latex
\pstDistABC\{A\}{B}\{C\}
```

2.8. Circles

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

```latex
\pstCircleOA\[Options\] \{O\}\{A\}\{angleA\}\{angleB\]
\pstCircleAB\[Options\] \{A\}\{B\}\{angleA\}\{angleB\]
```

`\pstCircleOA` draws the circle of center $O$ crossing $A$ from $angleA$ to $angleB$, going counter clockwise. Possible options are Radius and Diameter.

`\pstCircleAB` draws the circle of diameter $AB$ with the same options.

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 $^4$ and Diameter. The values of these parameters can be specified with one of the `\pstDist*` series macros.

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 $BC$**;

$^4$ The package `pst-fractal` also defines an optional key named Radius, if you need to use this package with `pst-eucl`, you need to setup the key Radius as following: `\psset[pst-eucl]{Radius=\pstDistVal{3}}`. 

```latex
\begin{pspicture}[showgrid=true](-1,-1)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode\[PosAngle=-90\](0,0){O}
\pstGeonode\[PosAngle=-90\](2,0){A}
\pstGeonode\[PosAngle=90\](1,1.5){B}
\pstCircleOA\[linecolor=red,Radius=\pstDistABC\{O\}{A}\{B\}\{B\}\{A\}\{O\}\{I\}\{J\}]
\pstCircleOA\[linecolor=blue,Radius=\pstDistABC\{B\}{O}\{A\}\{A\}\{B\}\{O\}\{I\}\{J\}]
\pstCircleOA\[linecolor=green,Radius=\pstDistABC\{A\}{B}\{O\}\{O\}\{A\}\{B\}\{I\}\{J\}]
\pstLineAB\[linecolor=red\]\{O\}{A}
\pstLineAB\[linecolor=blue\]\{B\}{O}
\pstLineAB\[linecolor=green\]\{A\}\{B\}
\end{pspicture}
```


- the circle of center \( B \) whose radius is \( AC \);
- the circle of center \( B \) of diameter \( AC \);
- the circle whose diameter is \( BC \).

The macro \texttt{\textbackslash pstCircleABR} draws the circle of given radius length \( R \), through two given nodes \( A \) and \( B \), then outputs the circle center \( O \).

\begin{verbatim}
\texttt{\textbackslash pstCircleABR [Options] \{A\}\{B\}\{R\}\{O\}}
\end{verbatim}

Note that through from \( A \) to \( B \) and through from \( B \) to \( A \) will get the figure symmetric to \( AB \). For example,

\begin{verbatim}
\begin{pspicture}[showgrid](-3,-3)(3,3) \footnotesize
\psset{unit=0.5cm}\psset{dotscale=0.5}\psset{PointSymbol=*}
\pstGeonode[PosAngle={30,210}]{1,0}(A)(-2,-1)(B)
\psCircleABR[linecolor=red!80]{A}{B}{\pstDistConst{2.5}}{O_1}
\psCircleABR[linecolor=blue!80]{B}{A}{\pstDistConst{2.5}}{O_2}
\end{pspicture}
\end{verbatim}

The following example show how to use the more complex distance macros, and the parameter to fill the circle.

\begin{verbatim}
\begin{pspicture}[showgrid](-3,-3)(3,3) \footnotesize
\psset{unit=0.5cm}\psset{dotscale=0.5}\psset{PointSymbol=*}
\pstGeonode[PosAngle={0,-135,90},PointSymbol=*,*,square]{1,0}(A)(-2,-1)(B)
\psCircleOA[linecolor=red]{A}{B}
\psCircleOA[linecolor=green,Radius=\pstDistMul{A}{C}{2 3 div}]{A}{C}
\psCircleOA[linecolor=blue,Radius=\pstDistAB{B}{C}]{A}{C}
\psCircleOA[linecolor=Sepia,Radius=\pstDistAB{A}{C}]{B}{C}
\psCircleOA[linecolor=Aquamarine,Diameter=\pstDistAB{A}{C}]{B}{C}
\end{pspicture}
\end{verbatim}
The last row set the absolute value of the abscissa of node $D$ to $\text{Radius}$, and then draw a circle at center $D$. Note that it does not work before v1.67, as the \pstCircleOA and \pstCircleAB were implemented with a \rput command, which will set the center $D$'s coordinate to origin, it causes that the Radius was set to zero and none circle will be draw out, so we remove the \rput code in v1.67, and everything works well now.

2.9. 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 $O$.

2.10. Circle nodes

Do you want to draw a point on the circle? A point can be positioned on a circle using its rotation angle by macro \pstCircleNode or \pstCircleRotNode. The first \pstCircleNode requires an explicit parameter angle $\theta$ to calculate the point; but the second \pstCircleRotNode requires an implicit parameter Ro\texttt{tAn}gle to calculate the point. If you not set Ro\texttt{tAn}gle, the default value is 60°.

The circle is defined by center $O$ and point $A$ on the circle or Radius or Diameter in parameter.
Sometimes we need to draw a chord with the given length from the start node, it is not possible to get the end node via the already defined macros, so we provide the macro \texttt{\textbackslash pstCircleChordNode} to do this work. This macro find the node \textit{X} on the circle such that the length of chord \textit{AX} is the given value \textit{L}, which can be got from \texttt{\textbackslash dist}, \texttt{\textbackslash distConst}, \texttt{\textbackslash distAdd}, \texttt{\textbackslash distSub}, etc.

\begin{verbatim}
\pstCircleChordNode[Options]{O}{A}{L}{X}
\end{verbatim}

The circle is just defined by center \textit{O} and point \textit{A} in this macro, so you can’t omit the parameter \textit{A}.

The direction to find node \textit{X} is anti-clockwise by default. The parameter \texttt{CurvAbsNeg}(by default false) can change this behavior.

At last, the chord length \textit{L} shouldn’t large than the diameter of the circle, else we will put the node \textit{X} at origin.

\begin{verbatim}
\pstCircleAbsNode[Options]{O}{A}{x}{C}{D}  \\
pstCircleOrdNode[Options]{O}{A}{y}{C}{D}
\end{verbatim}

for example,
A point can be positioned on a circle using its curved abscissa, that is, the arc length from a given node.

```
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 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 order to be aligned with the circle center and the point.

```
\begin{pspicture}[showgrid=true](-2.5,-2.5)(2.5,2.5)
\pstGeonode(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}
```

### 2.11. Circle tangent

The macro `\pstCircleTangentLine` is used to draw a tangent line \( AT \) from a point \( A \) on the circle, and the macro `\pstCircleTangentNode` is used to draw the tangent points \( T_1 \) and \( T_2 \) from a point \( P \) out of the circle.

```
\pstCircleTangentLine [Options] {O}{A}{T}
\pstCircleTangentNode [Options] {O}{A}{P}{T1}{T2}
```
2. Basic Objects

The macro \texttt{pstCircleExternalCommonTangent} is used to find the external common tangent lines of two circle $A(O_1)$ and $B(O_2)$, and the macro \texttt{pstCircleInternalCommonTangent} is used to find the internal common tangent lines of two circle $A(O_1)$ and $B(O_2)$. They both create four tangent point nodes $T_1, T_2, T_3, T_4$, where $T_1, T_2$ lie on circle $A(O_1)$, and $T_3, T_4$ lie on circle $B(O_2)$.

\begin{verbatim}
\pstCircleExternalCommonTangent[Options] {O_1}{A}{O_2}{B}{T_1}{T_2}{T_3}{T_4}
\pstCircleInternalCommonTangent[Options] {O_1}{A}{O_2}{B}{T_1}{T_2}{T_3}{T_4}
\end{verbatim}

You can use \texttt{RadiusA} and \texttt{RadiusB} to define the two circles like as following:

\begin{verbatim}
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=**}\footnotesize
\pstGeonode[PosAngle=90,120,-30]{1,1}{0,0}{T}{(3,0)}{S}
\pstCircleOA[Radius=\pstDistVal{1.5},linecolor=red]{T}{A}
\pstCircleRotNode[Radius=\pstDistVal{1.5},PosAngle=90,RotAngle=-30]{0}{A}{}
\pstCircleTangentLine[PosAngle=-10,PointName=A_1]{0}{A}{A1}
\pstCircleInternalCommonTangent[Radius=\pstDistVal{1.5},PosAngle=150,90,PointName=(T_1,T_2)]{0}{}
\pstCircleTangentNode[PosAngle=80,200]{0}{S_1}{}
\end{pspicture}
\end{verbatim}

You also can use \texttt{DiameterA} and \texttt{DiameterB} to define the two circles like as following:
2.12. Circle radical axis

If you want to draw the Radical Axis of two given circles, read the following sentences. For given \( O_1 \) with radius \( r_1 \) and \( O_2 \) with radius \( r_2 \), and the center \( O_1(x_1, y_1), O_2(x_2, y_2) \), then any point \( P(x, y) \) on the Radical Axis is satisfied:

\[
(x - x_1)^2 + (y - y_1)^2 - r_1^2 = (x - x_2)^2 + (y - y_2)^2 - r_2^2
\]

It can be simplified to a equation of a line:

\[
2(x_2 - x_1)x + 2(y_2 - y_1)y = (x_2^2 + y_2^2 - r_2^2) - (x_1^2 + y_1^2 - r_1^2)
\]

It is clear that the circles with same center have no radical axis, and the radical axis is perpendicular to the line of centers.

We provide the macro \texttt{pstCircleRadicalAxis} to draw the Radical Axis of two given circles. It can handler every position relations of circles such as separation, intersection and inclusion.

\[
\texttt{\textbackslash pstCircleRadicalAxis[Options]} \{O_1\}{A}\{O_2\}{B}\{C\}{D}
\]

Both parameter \( A \) and \( B \) can be omitted and then to specify the each radius or diameter using the parameters \( \text{Radius}A \), \( \text{Diameter}A \), and \( \text{Radius}B \), \( \text{Diameter}B \). This macro create two new nodes \( C \) and \( D \) on the radical axis, it can find them in following examples.

When they are intersected, we can see the radical axis is the intersected chord line.

\[
\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\end{pspicture}
\]

When they are tangent, we can see the radical axis is the common tangent line.
2. Basic Objects

2.13. Regular polygons

For the 3-side and 4-side regular polygon, we provide the macro \texttt{pstETriangleAB} and \texttt{pstSquareAB} to draw them. In general, you can use the macro \texttt{pstRegularPolygonAB} and \texttt{pstRegularPolygonOA} to get a n-side regular polygon.
\pstETriangleAB{Options}{A}{B}{C}
\pstSquareAB{Options}{A}{B}{C}{D}
\pstRegularPolygonAB{Options}{A_0}{A_1}{\cdots}{A_n} {n}
\pstRegularPolygonOA{Options}{O}{A_0}{\cdots}{A_n} {n}

The macro \texttt{\pstETriangleAB} draw a equilateral triangle on a given side \textit{AB}, and output the third node \textit{C}; The macro \texttt{\pstSquareAB} draw a square on a given side \textit{AB}, and output the other two nodes \textit{C, D}; The macro \texttt{\pstRegularPolygonAB} draw a \textit{n}-side regular polygon on a given side \textit{A_0A_1}, and output the other nodes \textit{A_2, A_3, \cdots, A_{n-1}}; The macro \texttt{\pstRegularPolygonOA} draw a \textit{n}-side regular polygon with center \textit{O} and base point \textit{A_0}, and output the other nodes \textit{A_1, A_2, \cdots, A_{n-1}}.

You can use the parameters linestyle, linecolor, linewidth to control the line style; and use the parameters PointName, PosAngle, PointSymbol to control the point nodes; and use the parameters CurveType, fillstyle, fillcolor to control the polygon style.

For the last output point list in macro \texttt{\pstRegularPolygonAB} and \texttt{\pstRegularPolygonOA}, if you do not enter a complete point list, the remaining points will be automatically named.

Note that draw regular polygon with side from \textit{A} to \textit{B} and side from \textit{B} to \textit{A} will get the figure symmetric to \textit{AB}. Here are some examples.

\begin{pspicture}[showgrid](-1,-2)(3,2)
\psset{unit=0.40cm}\footnotesize\psset{PointSymbol=none, PointNameSep=0.22cm}
\pstGeonode[PosAngle=180,PointSymbol=*](0,0){A}(5,0){B}
\pstSquareAB[linecolor=red!60,PosAngle=90,PointSymbol=*]{A}{B}{C}{D}
\pstSquareAB[linecolor=blue!60,PosAngle=-90,PointSymbol=*, fillstyle=solid,fillcolor=blue!20,opacity=0.2]{B}{C}{D}
\pstETriangleAB[linecolor=red!60,PosAngle=90,PointSymbol=*]{A}{B}{C}
\pstETriangleAB[linecolor=blue!60,PosAngle=-90,PointSymbol=*, fillstyle=solid,fillcolor=blue!20]{B}{A}{C'}
\end{pspicture}

\begin{pspicture}[showgrid](-3,-3)(3,3)
\psset{unit=0.40cm}\footnotesize\psset{PointSymbol=none, PointNameSep=0.22cm}
\pstGeonode[PosAngle=180,PointSymbol=*](0,0){A}(65,0){A}
\pstRegularPolygonOA[CurveType=polygon, linecolor=red!60, PointSymbol=*,PosAngle=120,PointSymbol=]{A}{B}{C}{D}
\pstRegularPolygonOA[CurveType=polygon, linecolor=green!60, PointSymbol=*,PosAngle=-90,PointSymbol=]{A}{B}{C}{D}
\pstRegularPolygonOA[CurveType=polygon, linecolor=blue!60, PointSymbol=*,PosAngle=65,PointSymbol=]{A}{B}{C}{D}
\pstRegularPolygonOA[CurveType=polygon, linecolor=green!60, PointSymbol=*,PosAngle=-90,PointSymbol=]{A}{B}{C}{D}
\pstRegularPolygonOA[CurveType=polygon, linecolor=blue!60, PointSymbol=*,PosAngle=65,PointSymbol=]{A}{B}{C}{D}
\end{pspicture}
2.14. 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 an interpolated curve crossing all such kind of points.

\texttt{\textbf{\textbackslash{pstranslate}}{\textbf{Options}} \{\textbf{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 \(n_{1}\) to \(n_{2}\). 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).
3. Conics

3.1. Standard Ellipse

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

$$\frac{(x-x_0)^2}{a^2} + \frac{(y-y_0)^2}{b^2} = 1$$

(1)

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

$$\begin{align*}
&x = a \cos \alpha + x_0 \\
y = b \sin \alpha + y_0
\end{align*}$$

(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.

\begin{verbatim}
\pstEllipse[Options](O)(a,b)[angleA][angleB]
\end{verbatim}

Like as the coordinates, the parameters $a, b$ can be got by the raw PostScript commands too, where you can use the macros \pstDist*, for example,
Now you can draw some points on this Ellipse using macro $\texttt{pstEllipseNode}$ or $\texttt{pstEllipseRotNode}$. The macro $\texttt{pstEllipseNode}$ requires an explicit parameter $t$ as $\alpha$ in equation (2) to calculate the point; but the macro $\texttt{pstEllipseRotNode}$ requires an implicit parameter $\text{RotAngle}$ as $\alpha$ in equation (2) to calculate the point.

\begin{verbatim}
\texttt{pstEllipseNode [Options]} \(O)(a,b)\{t\}\{P\}
\texttt{pstEllipseRotNode [Options]} \(O)(a,b)\{P\}
\end{verbatim}

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

![Diagram](image)

The macros $\texttt{pstEllipseAbsNode}$ and $\texttt{pstEllipseOrdNode}$ are used to get the two nodes $A$ and $B$ whose abscissas or ordinate 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.

\begin{verbatim}
\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}
\pstEllipse[linestyle=\textcolor{red}{-}]{\ra}{\rb}\(O)\{\rb\,\ra\}\{P\}
\pstEllipse[linestyle=\textcolor{red}{-}](\ra,\rb){O}\{\rb,\ra\}\{P\}
\pstLineAB[linestyle=dashed]{A}{H}
\pstLineAB[linestyle=dashed]{A}{H}
\pstLineAB[linestyle=dashed]{B}{X}
\pstLineAB[linestyle=dashed]{B}{X}
\pstMarkAngle[LabelSep=.6,MarkAngleRadius=.3,MarkAngleType=double]{O}{A}{$\alpha$}{\textcolor{red}{\circ}$}
\end{pspicture}
\end{verbatim}

![Diagram](image)

Here we find the focus node of Standard Ellipse! Please use macro $\texttt{pstEllipseFocusNode}$ 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}{\rb}{\ra}{D}
\pstEllipseFocusNode{0}{\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},PosAngle={210,210,-30,-30},nodesep=-1,linecolor=red!40]{O}{\ra}{\rb}{Lx}{Ly}{Rx}{Ry}
\pstEllipseDirectrixLine[PointName={D_x,D_y,U_x,U_y},PosAngle={-30,-30,30,30},nodesep=-1,linecolor=blue!40]{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]{Dy}{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:
The macro \texttt{\texttt{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.

\begin{verbatim}
\texttt{\texttt{pstEllipsePolarNode}} \[\text{Options}] (O)(a,b){A}{B}{T}
\end{verbatim}

We use the following theorem to find the node \( T \):

\textbf{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 \( B \).
The macro \texttt{pstEllipseTangentNode} is used to draw the tangent line of a point $T$ out of the Standard Ellipse $E$. It draws two tangent lines through the point $T$ to the Standard Ellipse $E$ and get the node $A$ and $B$ on the Ellipse.

\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}
\def\ra{2.8}\def\rb{1.2}
\pstGeode{PosAngle=-50,PointNameSep=0.2}{2,2}{0}
\pstEllipse[linecolor=red!40](0)(\ra,\rb)
\pstLine[nodesep=0.8,linecolor=black!30,linestyle=dashed](1,2)(2.5,3.5)
\pstEllipseLineInter{PosAngle={-100,90}}{0}{\ra,\rb}
\pstEllipsePolNode{PosAngle=120}{0}{\ra,\rb}{A}{B}
\pstEllipses{PosAngle=-90}{0}{\ra,\rb}{A}{B}
\pstEllipses{PosAngle=-10}{0}{\ra,\rb}{P}{Q}
\pstEllipses{PosAngle=-100,PointNameSep=0.2}{2,2}{0}
\pstEllipses{PosAngle=0,PointNameSep=0.2}{2,2}{0}
\pstInterLL{PosAngle=-90}{0}{\ra,\rb}{A}{B}{X}
\pstInterLL{PosAngle=-10}{0}{\ra,\rb}{A}{B}{Y}
\pstInterLL{PosAngle=-90}{0}{\ra,\rb}{A}{B}{S}
\psset{linecolor=gray!40}
\pstLine(A){P}{B}{P}{linecolor=red!40}
\pstLine(A){R}{B}{S}
\pstLine(Q){X}{Y}
\pstLine(R){M}{N}
\psset{linestyle=dashed,linestyle=dashed}
\pstLine{M}{N}
\end{pspicture}

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

\begin{proof}
\textbf{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 $B$.
\end{proof}

\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}
\def\ra{2.8}\def\rb{1.2}
\pstGeode{PosAngle=-50,PointNameSep=0.2}{2,2}{0}
\pstEllipse[linecolor=red!40](0)(\ra,\rb)
\pstGeode{PosAngle=-50,PointNameSep=0.2}{2,2}{0}
\pstEllipseTangentNode{PosAngle=129}{0}{\ra,\rb}{A}{B}
\pstEllipses{PosAngle=71}{0}{\ra,\rb}{P}{Q}
\pstEllipses{PosAngle=31}{0}{\ra,\rb}{P}{Q}
\pstEllipses{PosAngle=0}{0}{\ra,\rb}{P}{Q}
\pstEllipses{PosAngle=0}{0}{\ra,\rb}{P}{Q}
\pstInterLL{PosAngle=0}{P}{S}{R}{I}
\pstInterLL{PosAngle=0}{P}{S}{R}{X}
\psset{linestyle=dashed,linestyle=dashed}
\pstLine{M}{N}
\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 $\max(\text{abs}(a), \text{abs}(b))$, the half of the minor axis $\min(\text{abs}(a), \text{abs}(b))$, and the rotation angle $\theta$ of the major axis.

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

\[
\begin{align*}
    x' &= x \cos \theta - y \sin \theta \\
    y' &= x \sin \theta + y \cos \theta
\end{align*}
\]

then we have

\[
\begin{align*}
    x' &= (a \cos \alpha + x_0) \cos \theta - (b \sin \alpha + y_0) \sin \theta = a \cos \alpha \cos \theta - b \sin \alpha \sin \theta + x_0' \\
    y' &= (a \cos \alpha + x_0) \sin \theta + (b \sin \alpha + y_0) \cos \theta = a \cos \alpha \sin \theta + b \sin \alpha \cos \theta + y_0'
\end{align*}
\]

where the $x'_0$ and $y'_0$ 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:

\[
\begin{align*}
    x &= a \cos \alpha \cos \theta - b \sin \alpha \sin \theta + x_0 \\
    y &= a \cos \alpha \sin \theta + b \sin \alpha \cos \theta + y_0
\end{align*}
\]

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 rotation angle $\theta$, the default value is $0^\circ$, at this time, the result of this macro is same as \pstEllipse. That is, \pstGeneralEllipse is more complex than \pstEllipse!

```latex
\pstGeneralEllipse [Options] \(O)(a, b) [\theta] [\text{angleA}] [\text{angleB}]
```

The Macro \pstGeneralEllipse is used to define a General Ellipse with Focus $F$, directrix line $l$, and the eccentricity $e$, where $0 \leq e < 1$. It just calculate the center $O$, major radius $a$, minor radius $b$ and the rotation angle $\theta$ of the major axis, then you can pass them into macro \pstGeneralEllipse to draw this ellipse.

```latex
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5, 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](0)(\ra,\rb)[10]
\pstGeneralEllipse[linecolor=gray!20](0)(\ra,\rb)[20]
\pstGeneralEllipse[linecolor=gray!30](0)(\ra,\rb)[30]
\pstGeneralEllipse[linecolor=gray!40](0)(\ra,\rb)[40]
\pstGeneralEllipse[linecolor=magenta!40](0)(\ra,\rb)[50]
\end{pspicture}
```

The output parameter $O$ is a node name to store the center point, its label and symbol can be controlled by the options for PSTricks node, such as PosAngle. The output parameter Rab is a PostScript key to store the pair of major radius and minor radius, it just use PSTricks node coordinate to store a pair of value, but not a geometrical point. The output parameter $\theta$ is also a PostScript key to store the rotation angle of major axis, when you pass it to \pstGeneralEllipse, PostScript will lookup the value of this key in current dictionary.
The Macro \texttt{\textbackslash pstGeneralEllips\textbackslash eFFN} is used to define a General Ellipse by the given focus nodes $F_1$, $F_2$, and one node $N$ on it. It just calculates the center $O$, major radius $a$, minor radius $b$ and the rotation angle $\theta$ of the major axis, then you can pass them into macro \texttt{\textbackslash pstGeneralEllipse} to draw this ellipse.

\begin{verbatim}
\texttt{\textbackslash pstGeneralEllips\textbackslash eFFN [Options] \{F_1\}{F_2}\{O\}{\text{Rab}}\{\theta\}}
\end{verbatim}

The output parameter 0, the output parameter Rab and the output parameter $\theta$ are same with \texttt{\textbackslash pstGeneralEllips\textbackslash eF}. 

\begin{verbatim}
\texttt{\textbackslash pstGeneralEllips\textbackslash eCoef [Options] \{a,b,c,d,e,f\}\{O\}{\text{Rab}}\{\theta\}}
\end{verbatim}

The output parameter 0, the output parameter Rab and the output parameter $\theta$ are same with \texttt{\textbackslash pstGeneralEllips\textbackslash eF}. They are set to zero if the coefficients are invalid to construct an ellipse.
The Macro \texttt{\textbackslash pstGeneralEllipseABCDE} is used to define a General Ellipse by the given five points \(A, B, C, D, E\), it just calculate the center \(O\), major radius \(a\), minor radius \(b\) and the rotation angle \(\theta\) of the major axis, then you can pass them into macro \texttt{\textbackslash pstGeneralEllipse} to draw this ellipse.

\begin{verbatim}
\texttt{\textbackslash pstGeneralEllipseABCDE \{Options\} \{A\}\{B\}\{C\}\{D\}\{E\}\{O\}\{Rab\}\{\theta\}}
\end{verbatim}

The output parameter \(O\), the output parameter \(Rab\) and the output parameter \(\theta\) are same with \texttt{\textbackslash pstGeneralEllipseFle}. They are set to zero if the points are invalid to construct an ellipse.
3. Conics

We can location the points on the General Ellipse using the macros \pstGeneralEllipseNode, \pstGeneralEllipseRotNode, \pstGeneralEllipseAbsNode and \pstGeneralEllipseOrdNode as following:

\begin{pspicture}[showgrid=true](-1,-2)(5,4)
\pstGeonode[PosAngle=180]{A}
\pstGeonode[PosAngle=-90]{B}
\pstGeonode[PosAngle=90]{C}
\pstGeonode[PosAngle=-90]{D}
\pstGeonode[PosAngle=0]{E}
\pstGeneralEllipseABCDE[PosAngle=0]{A}{B}{C}{D}{E}{O}{R}{MajorRotAngle}
\end{pspicture}

Some examples all together:

\begin{pspicture}[showgrid=true](-1,-2)(5,4)
\pstGeonode[PosAngle=-90,PointNameSep=0.2]{A}{(2,2)}{B}
\pstGeneralEllipse[Linecolor=magenta]{0}{A}{B}{C}{D}{E}{O}{R}{MajorRotAngle}
\end{pspicture}

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

\begin{pspicture}[showgrid=true](-1,-2)(5,4)
\pstGeneralEllipseFocusNode[Options]{O}{(a, b)}{t}{F_1}{F_2} \pstGeneralEllipseDirectrixLine[Options]{O}{(a, b)}{L_x}{L_y}{R_x}{R_y}
\end{pspicture}

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

\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){C}
\pstGeneralEllipse[linecolor=magenta](0)(\ra,\rb){50}
\pstGeneralEllipseFocusNode[PosAngle=-40,\ra,\rb][50]{L}
\pstGeneralEllipseDirectrixLine[PosName={Lx,L_y,Rx,R_y},nodesep=-1,\ra,\rb][50]{Lx}{Ly}{Rx}{Ry}
\pstLine[nodesep=-1,\ra,\rb][50]{L}{R}
\pstLine[nodesep=-1,\ra,\rb][50]{L}{R}
\end{pspicture}

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

\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){C}
\pstGeneralEllipse[linecolor=magenta](0)(\ra,\rb){50}
\pstLine[nodesep=-1,\ra,\rb][50]{Lx}{Ly}{Rx}{Ry}
\end{pspicture}

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

\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){C}
\pstGeneralEllipse[linecolor=magenta](0)(\ra,\rb){50}
\pstLine[nodesep=-1,\ra,\rb][50]{Lx}{Ly}{Rx}{Ry}
\end{pspicture}
3. Conics

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 \(\text{abs}(p)\). Note that the sign of \(p\) indicates the direction of the parabola.

The equation can be written as:

\[
(x - x_0)^2 = 2p(y - y_0)
\]

and the parametric function can be written as:

\[
\begin{align*}
  x &= t + x_0 \\
  y &= \frac{t^2}{2p} + y_0
\end{align*}
\]

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

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

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

\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}
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.
The macro \texttt{\lstinline!\pstParabolaTangentNode!} is used to find the two nodes $A$ and $B$ on the Parabola through the point $T$:

\begin{verbatim}
\pstParabolaTangentNode [Options] (O){p}{T}{A}{B}
\end{verbatim}

We use the following theorem 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 \(T P Q\) and \(T RS\), \(P S\) and \(Q R\) 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 \(B\).

3.4. Standard Conjugate Parabola

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

\[
(y - y_0)^2 = 2p(x - x_0)
\]

and the parametric function can be written as:

\[
\begin{align*}
x &= \frac{t^2}{2p} + x_0 \\
y &= t + y_0
\end{align*}
\]

The macro \texttt{\textbackslash pstIParabola} is used to draw a Standard Conjugate Parabola from \(y_1\) to \(y_2\) with Vertex \(O\), the half of the focus chord axis \(\text{abs}(p)\).

\[
\texttt{\textbackslash pstIParabola[Options]}(O\{p\}\{y_1\}\{y_2\})
\]

The macro \texttt{\textbackslash pstIParabolaNode} is used to draw a node whose parameter is the given value \(t\) on parabola, please refer to equation (9). The macro \texttt{\textbackslash pstIParabolaAbsNode} is used to draw a node whose abscissa is the given value \(x_1\) on parabola. The macro \texttt{\textbackslash pstIParabolaOrdNode} is used to draw a node whose ordinate is the given value \(y_1\) on parabola. Note that \texttt{\textbackslash pstIParabolaAbsNode} will create two nodes \(A\) and \(B\) at most time.

\[
\begin{align*}
\texttt{\textbackslash pstIParabolaNode[Options]}(O\{p\}\{t\}\{A\}) \\
\texttt{\textbackslash pstIParabolaAbsNode[Options]}(O\{p\}\{x_1\}\{A\}\{B\}) \\
\texttt{\textbackslash pstIParabolaOrdNode[Options]}(O\{p\}\{y_1\}\{A\})
\end{align*}
\]
3. Conics

The macro \texttt{\texttt{pstIParabolaFocusNode}} is used to find the focus of the parabola, and the macro \texttt{\texttt{pstIParabolaDirectrixLine}} is used to find the directrix line of the parabola.

\begin{verbatim}
\begin{pspicture}(-2,-2)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*
\footnotesize
\def\p{0.4}
\pstGeonode{PosAngle=0,PointNameSep=0.2}(2,0){O}
\pstIParabola[linecolor=blue!40](0){\p}(-1.5)(1.5)
\pstIParabola[BorderColor=black](0){(A)
\end{pspicture}
\end{verbatim}

The macro \texttt{\texttt{pstIParabolaLineInter}} is used to find the intersections \(C\) and \(D\) of the parabola and the given line \(AB\).

\begin{verbatim}
\begin{pspicture}(-2,-2)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*
\footnotesize
\def\p{0.4}
\pstGeonode{PosAngle=30,PointNameSep=0.2}(2,0){O}
\pstIParabola[linecolor=blue!40](0){\p}(-1.5)(1.5)
\pstIParabola[BorderColor=black](0){(A)
\end{pspicture}
\end{verbatim}

The macro \texttt{\texttt{pstIParabolaPolarNode}} is used to find the polar point \(T\) of chord \(AB\) on Parabola \(P\).

\begin{verbatim}
\begin{pspicture}(-2,-2)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*
\footnotesize
\def\p{0.4}
\pstGeonode{PosAngle=0,PointNameSep=0.2}(2,0){O}
\pstIParabola[linecolor=blue!40](0){\p}(-1.5)(1.5)
\pstIParabola[BorderColor=black](0){(A)
\end{pspicture}
\end{verbatim}

We also use the theorem 3 to find the polar point \(T\) 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.
We also use the theorem 4 to find the tangent node $A$ and $B$ of outside point $T$. 

\begin{pspicture}[showgrid=true](1,-2)(5,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](0){p}(-1.5){1.5}
\pstLine[linecolor=gray!40,odeseppA=-0.5](2,1){4,-2}
\pstParabolaLineInter[PosAngle=(80,-100)](0){p}{2,1}{4,-2}{Q}

% if you don't know focus F or directrix line
\pstParabolaPolarNode[linecolor=purple!40,PosAngle=-90](0){p}{O}{Q}{T}
\end{pspicture}
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 rotation angle $\theta$ of the symmetrical axis. The symmetrical axis is perpendicular to $x$-axis when $\theta = 0^\circ$, and perpendicular to $y$-axis when $\theta = 90^\circ$.

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

\[
\begin{cases}
  x' = (t + x_0) \cos \theta - (t^2 + y_0) \sin \theta = x'_0 + t \cos \theta - t^2 \frac{\sin \theta}{2p} \\
y' = (t + x_0) \sin \theta + (t^2 + y_0) \cos \theta = y'_0 + t \sin \theta + t^2 \frac{\cos \theta}{2p}
\end{cases}
\]  
\[(10)\]

where the $x'_0$ and $y'_0$ 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:

\[
\begin{cases}
x = x_0 + t \cos \theta - t^2 \frac{\sin \theta}{2p} \\
y = y_0 + t \sin \theta + t^2 \frac{\cos \theta}{2p}
\end{cases}
\]  
\[(11)\]

The macro $\texttt{\textbackslash pstGeneralParabola}$ is used to draw a General Parabola from $x_1$ to $x_2$ with Vertex $O$, the half of the focus chord axis $abs(p)$.

\[
\texttt{\textbackslash pstGeneralParabola [Options]} (O){p} [\theta] \{x_1\}{x_2}
\]

The macro $\texttt{\textbackslash pstGeneralParabolaNode}$ is used to draw a node whose parameter is the given value $t$ on parabola, please refer to equation (11). The macro $\texttt{\textbackslash pstGeneralParabolaAbsNode}$ is used to draw a node whose abscissa is the given value $x_1$ on parabola. The macro $\texttt{\textbackslash pstGeneralParabolaOrdNode}$ is used to draw a node whose ordinate is the given value $y_1$ on parabola.

Note that $\texttt{\textbackslash pstGeneralParabolaAbsNode}$ and $\texttt{\textbackslash pstGeneralParabolaOrdNode}$ both create two nodes $A$ and $B$ at most time.

\[
\texttt{\textbackslash pstGeneralParabolaNode [Options]} (O){p} [\theta] \{t\}\{A\} \\
\texttt{\textbackslash pstGeneralParabolaAbsNode [Options]} (O){p} [\theta] \{x_1\}\{A\}\{B\} \\
\texttt{\textbackslash pstGeneralParabolaOrdNode [Options]} (O){p} [\theta] \{y_1\}\{A\}\{B\}
\]
The Macro \texttt{\textbackslash pstGeneralParabolaFl} is used to define a General Parabola with Focus $F$, and the directrix line $l$. It just calculate the vertex $O$, half focal chord $p$, and the rotation angle $\theta$ of the symmetrical axis, then you can pass them into macro \texttt{\textbackslash pstGeneralParabola} to draw this parabola.

\begin{verbatim}
\pstGeneralParabolaFl[Options] {F}{A}{B}{O}{p}{\theta}
\end{verbatim}

The output parameter $O$ is a node name to store the vertex point, its label and symbol can be controlled by the options for PSTricks node, such as \texttt{PosAngle}. The output parameter $p$ is a PostScript key to store the value of half focal chord. The output parameter $\theta$ is also a PostScript key to store the rotation angle of symmetrical axis, when you pass it to \texttt{\textbackslash pstGeneralParabola}, PostScript will lookup the value of this key in current dictionary.
3. Conics

The Macro \texttt{\textbackslash{}pstGeneralParabolaCoef} is used to define a General Parabola by the quadratic curve equation \( ax^2 + bxy + cy^2 + dx + ey + f = 0 \), it just calculate the vertex \( O \), half focal chord \( p \) and the rotation angle \( \theta \) of the symmetrical axis, then you can pass them into macro \texttt{\textbackslash{}pstGeneralParabola} to draw this parabola. The package \texttt{pst-func} provides macro \texttt{\textbackslash{}psplotImp} to draw an implicit defined functions too, but it can’t tell you the geometrical elements like as center or radii, and it will take more time to calculate the function value point by point.

\begin{verbatim}
\texttt{\textbackslash{}pstGeneralParabolaCoef [Options] \{a,b,c,d,e,f\}\{O\}\{p\}\{\theta\}}
\end{verbatim}

The output parameter \( O \), \( p \) and \( \theta \) are same with \texttt{\textbackslash{}pstGeneralParabolaFl}. They are set to zero if the coefficients are invalid to construct a parabola. If you pass the zero \( p \) into macro \texttt{\textbackslash{}pstGeneralParabola}, it will abort with the exception of dividing by zero.

In the following example, we use \texttt{\textbackslash{}psplotImp} to draw the same parabolas, just to check the results given by macros \texttt{\textbackslash{}pstGeneralParabolaCoef} are correct.
The Macro \texttt{pstGeneralParabolaABCDE} is used to define a General Parabola by the given five points $A, B, C, D, E$, it just calculate the vertex $O$, half focal chord $p$ and the rotation angle $\theta$ of the symmetrical axis, then you can pass them into macro \texttt{pstGeneralParabola} to draw this parabola.

\begin{verbatim}
\pstGeneralParabolaABCDE [Options] {A}{B}{C}{D}{E}{O}{p}{\theta}
\end{verbatim}

The output parameter 0, $p$ and $\theta$ are same with \texttt{pstGeneralParabolaFl}. They are set to zero if the points are invalid to construct a parabola. If you pass the zero $p$ into macro \texttt{pstGeneralParabola}, it will abort with the exception of dividing by zero.

Note the algorithm may fit a hyperbola quadratic curve from the given five points, in order to get the right parabola curve, you must input the point coordinates very precisely. In the following example, if you input point $A$ as $(3,1.732)$, it will fail as no such parabola can fit these five points.

\begin{verbatim}
\end{verbatim}

The macro \texttt{pstGeneralParabolaFocusNode} is used to find the focus of the parabola, and the macro \texttt{pstGeneralParabolaDirectrixLine} is used to find the directrix line of the parabola.

\begin{verbatim}
\pstGeneralParabolaFocusNode [Options] (O){p}{\theta} {F}
\pstGeneralParabolaDirectrixLine [Options] (O){p}{\theta} {L_x}{L_y}
\end{verbatim}
The macro \texttt{\textbackslash pstGeneralParabolaLineInter} is used to find the intersections $C$ and $D$ of the parabola and the given line $AB$.

\begin{verbatim}
\texttt{\textbackslash pstGeneralParabolaLineInter[Options]} \textbackslash (O){p}\ [\theta] \{A\} \{B\} \{C\} \{D\}
\end{verbatim}

When General Parabola becomes a Standard Parabola, the intersections with any kind of lines:

\begin{verbatim}
\texttt{\textbackslash begin\{pspicture\}[showgrid=true](0,-1)(4,3)\\
\texttt{\textbackslash pset\{dotscale=0.5\}} \texttt{\textbackslash pset\{PointSymbol=\*\}} \texttt{\footnotesize}\\
\texttt{\textbackslash def\{p\}(0.4)\\
\texttt{\textbackslash pstGeonode[PosAngle=-90,PointNameSep=0.2] \textbackslash (2,0)\{O\}\\
\texttt{\textbackslash pstGeneralParabola[linestyle=dashed,linestyle=black!25,nodesepA=-0.2,nodesepB=1.2] \textbackslash (O){p}\[\theta]\{A\} \{B\} \{C\} \{D\}\\
\texttt{\textbackslash end\{pspicture\}}}
\end{verbatim}

Here is the intersections of a real General Parabola with any kind of lines:
When General Parabola becomes a Standard Conjugate Parabola, the intersections with any kind of lines:

The macro \texttt{\lstinline|\pstGeneralParabolaPolarNode|} is used to find the polar point $T$ of chord $AB$ on Parabola $P$.

We also use the theorem 3 to find the polar point $T$ 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.
The macro \pstGeneralParabolaTangentNode is used to find the two nodes $A$ and $B$ on the Parabola through the point $T$.

We also use the theorem 4 to find the tangent node $A$ and $B$ of outside point $T$.

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

The equation can be got from the parametric function of the conjugate parabola (9), using the rotation transform formula (3), then we have

\[
\begin{align*}
x' &= \left(\frac{t^2}{2p} + x_0\right) \cos \theta - \left(\frac{t^2}{2p} + y_0\right) \sin \theta = x_0' - t \sin \theta + t^2 \cos \theta \\
y' &= \left(\frac{t^2}{2p} + x_0\right) \sin \theta + \left(\frac{t^2}{2p} + y_0\right) \cos \theta = y_0' + t \cos \theta + t^2 \sin \theta
\end{align*}
\] (12)
where the $x'_0$ and $y'_0$ are the coordinate of the given vertex $O$ after rotation. So we get the parametric function of the General Conjugate Parabola with coordinate translation and rotation as following:

\[
\begin{align*}
  x &= x_0 - t \sin \theta + t^2 \cos \theta \\
  y &= y_0 + t \cos \theta + t^2 \sin \theta \\
\end{align*}
\]  

(13)

The macro `\pstGeneralIParabola` is used to draw a Standard Conjugate Parabola from $y_1$ to $y_2$ with Vertex $O$, the half of the focus chord axis $abs(p)$.

```
\pstGeneralIParabola[Options] (O){p}[\theta] {y_1}{y_2}
```

The macro `\pstGeneralIParabolaNode` is used to draw a node whose parameter is the given value $t$ on parabola, please refer to equation (13). The macro `\pstGeneralIParabolaAbsNode` is used to draw a node whose abscissa is the given value $x_1$ on parabola. The macro `\pstGeneralIParabolaOrdNode` is used to draw a node whose ordinate is the given value $y_1$ on parabola.

Note that `\pstGeneralIParabolaAbsNode` and `\pstGeneralIParabolaOrdNode` will create two nodes $A$ and $B$ at most time.

```
\pstGeneralIParabolaNode[Options] (O){p}[\theta] {t}\{A}
\pstGeneralIParabolaAbsNode[Options] (O){p}[\theta] {x_1}\{A\}{B}
\pstGeneralIParabolaOrdNode[Options] (O){p}[\theta] {y_1}\{A\}{B}
```

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

```
\pstGeneralIParabolaFocusNode[Options] (O){p}[\theta] \{F\}
\pstGeneralIParabolaDirectrixLine[Options] (O){p}[\theta] \{L_x\}{L_y}\n```
When $\theta = 0$, the intersections with any kind of lines:

\begin{pspicture}[showgrid=true](-2,0)(2,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize\def\p{0.4}
\pstGeonode{0,0}{O}
\psset{PointNameSep=0.2}
\psset{linestyle=dashed, linecolor=gray!40}$E$\psline[linecolor=red!40](-1.5,1.5)(1.5,-1.5)
\psset{linestyle=dashed, linecolor=gray!40}$O$\psline[linecolor=red!40](-2,0)(2,4)
\psset{linestyle=dashed, linecolor=gray!40}$A$\psline[linecolor=red!40](0,-2)(0,2)
\psset{linestyle=dashed, linecolor=gray!40}$B$\psline[linecolor=red!40](2,-2)(2,4)
\end{pspicture}
When $\theta = 50$, the intersections with any kind of lines:

\begin{pspicture}[showgrid=true](1,-1)(5,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}{footnot.esize}
\def\p{0.4}
\pstGeonode[PosAngle=-70,PointNameSep=0.2](2,0){O}
\pstGeneralIParabola[linecolor=red!40](0){\p}[50]{1.5}{-1.5}{1.5}
\pstGeneralIParabolaFocusNode[linecolor=red!40,PosAngle=80](0){\p}[50]{F}
\pstLineAB[linestyle=dashed,linecolor=black!25,nodesepB=-0.2,
nodesepA=-2.5](0){F}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep =-0.8](3,-1){3,3}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle =(-60,40)](0){\p}[50]{3,-1}{3,3}{A}{B}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep =0.0](2,3){4,0}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle =(-10,170)](0){\p}[50]{2,3}{4,0}{C}{D}
\begin{footnotesize}
% a line with gradient $k=\tan50$ parallel to OF
\pstLineAS[linestyle=dashed,linecolor=gray!40,nodesep=-0.8,
PointName=none,PointSymbol=none](2,1)(50 \tan){X}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle =(180,-90)](0){\p}[50]{2,1}{X}{E}{G}
\end{pspicture}

When $\theta = 90$, the intersections with any kind of lines:

\begin{pspicture}[showgrid=true](0,-1)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}{footnot.esize}
\def\p{0.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,0){O}
\pstGeneralIParabola[linecolor=red!40](0){\p}[90]{1.5}{-1.5}{1.5}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep =-0.5](1,0){1,2}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle =(-180,-90)](0){\p}[90]{1,0}{1,2}{A}{B}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep =-0.5](1,0){3,1}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle =(-60,-90)](0){\p}[90]{1,0}{3,1}{C}{D}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep =-0.5](0.8,2){3,2}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle =(120,60)](0){\p}[90]{0.8,2}{3,2}{E}{G}
\end{pspicture}

The macro $\texttt{pstGeneralIParabolaPolarNode}$ is used to find the polar point $T$ of chord $AB$ on Parabola $P$.

$\texttt{pstGeneralIParabolaPolarNode[Options]} \ (O){p} \ [\theta] \ \{A\}{B}\{T}$

$\texttt{pstGeneralIParabolaPolarNode[Options]} \ (O){p} \ [\theta] \ (F)\{A\}{B}\{T}$

$\texttt{pstGeneralIParabolaPolarNode[Options]} \ (O){p} \ [\theta] \ (F) \ [L_x] \ [L_y] \ \{A\}{B}\{T}$
We also use the theorem 3 to find the polar point \( T \) 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.

The macro `\texttt{pstGeneralIParabolaTangentNode}` is used to find the two nodes on the Parabola through the point \( T \).

\begin{verbatim}
\pstGeneralIParabolaTangentNode [Options] \( O \) \([p]\) \([\theta]\) \{\(T\}\{A\}\{B\}
\end{verbatim}

### 3.7. Standard Hyperbola

The Standard Hyperbola \( H \) with coordinate translation is defined by center \( O \), the half of the real axis \( a \), the half of the imaginary axis \( b \). The equation can be written as:

\[
\frac{(x - x_0)^2}{a^2} - \frac{(y - y_0)^2}{b^2} = 1
\]

and the parametric function can be written as:

\[
\begin{cases}
  x = a \sec \alpha + x_0 \\
  y = b \tan \alpha + y_0
\end{cases}
\]

The macro `\texttt{pstHyperbola [Options]} \( O \)(\(a, b\)) [\(\alpha_{\text{angleX}}\)]` is used to draw a Standard Hyperbola with Center \( O \), the half of the real axis \( a \), the half of the imaginary axis \( b \). The parameter \( \alpha_{\text{angleX}} \) is used to truncate the width of the figure, it should be setup from 0 to 90.

\begin{verbatim}
\pstHyperbola [Options] \( O \)(\(a, b\)) [\(\alpha_{\text{angleX}}\)]
\end{verbatim}
nodes whose abscissa are the given value \( x_1 \) on Hyperbola. The macro \( \texttt{pstHyperbolaAbsNode} \) is used to draw the nodes whose ordinate are the given value \( y_1 \) on Hyperbola.

Note that \( \texttt{pstHyperbolaAbsNode} \) and \( \texttt{pstHyperbolaOrdNode} \) will create two nodes \( A \) and \( B \) at most time.

\[
\begin{array}{ll}
\texttt{pstHyperbolaNode} \left[ \text{Options} \right] (O)(a, b\{t}\{A} \\
\texttt{pstHyperbolaAbsNode} \left[ \text{Options} \right] (O)(a, b\{x_1}\{A}\{B} \\
\texttt{pstHyperbolaOrdNode} \left[ \text{Options} \right] (O)(a, b\{y_1}\{A}\{B}
\end{array}
\]

The macro \( \texttt{pstHyperbolaFocusNode} \) is used to find the focus nodes of the Hyperbola, and the macro \( \texttt{pstHyperbolaDirectrixLine} \) is used to find the directrix lines of the Hyperbola.

\[
\begin{array}{ll}
\texttt{pstHyperbolaFocusNode} \left[ \text{Options} \right] (O)(a, b\{F_1}\{F_2} \\
\texttt{pstHyperbolaDirectrixLine} \left[ \text{Options} \right] (O)(a, b\{L_x\{L_y}\{R_x\{R_y}
\end{array}
\]

Note that you can use \( \texttt{pstLineAS} \) to draw the asymptote line of the hyperbola by passing the slope gradient \( k = \pm \frac{b}{a} \); or you can use the macro \( \texttt{pstHyperbolaAsymptoteLine} \) to get them, this macro only create one node on each asymptote line, as the other one is the center of the hyperbola.

\[
\begin{array}{ll}
\texttt{pstHyperbolaAsymptoteLine} \left[ \text{Options} \right] (O)(a, b\{L_1}\{L_2}
\end{array}
\]
The macro \texttt{\pstHyperbolaLineInter} is used to find the intersections \( C \) and \( D \) of the hyperbola and the given line \( AB \).

\begin{verbatim}
\pstHyperbolaLineInter[Options](O)(a, b){A}{B}{C}{D}
\end{verbatim}

In the following example, the Line \( CX \) and \( CY \) are parallel to the asymptote of the hyperbola.

\begin{verbatim}
\begin{pspicture}【showgrid=true】(-2,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\a{0.5}\def\b{0.3}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](-90,90)(1,1.5){C}
\pstHyperbola[Linecolor=blue!40]{O}(a, b){80}
\pstHyperbolaNode[Linecolor=blue!40]{O}(a, b){80}{A}
\pstHyperbolaLineInter[Linecolor=blue!40]{O}(a, b){80}{A}
\pstLineAS[PointName=\textbullet](1,1.5){C}
\pstHyperbola[Linecolor=blue!40]{O}(a, b){80}
\pstHyperbolaFocusNode[Linecolor=blue!40,PointName=F_1,F_2]{O}(180,0){F_1,F_2}
\pstHyperbolaDirectrixLine[Linecolor=blue!40,PointNameSep=2]{O}(180,0){F_1,F_2}
\end{pspicture}
\end{verbatim}

The macro \texttt{\pstHyperbolaPolarNode} is used to find the polar point \( T \) of chord \( AB \) on the hyperbola.

\begin{verbatim}
\pstHyperbolaPolarNode[Options](O)(a, b){A}{B}{T}
\end{verbatim}

We use the following theorem to find the polar point \( T \) of chord \( AB \):
**Theorem 5** Let \( P, Q \) are vertex points of the hyperbola, for any chord \( AB \) of hyperbola, suppose \( PA \) and \( BQ \) intersect at \( E \), \( PB \) and \( AQ \) intersect at \( F \), then the middle point \( T \) of \( EF \) is the polar point of chord \( AB \).

The macro \texttt{psthypbolaTangentNode} is used to find the tangent point \( A \) and \( B \) of point \( T \) outside of the hyperbola.

\begin{lstlisting}[language=TeX]
psthypbolaTangentNode[Options] (O)(a, b){T}{A}{B}
\end{lstlisting}

We use the following theorem to find the tangent points \( A \) and \( B \) of \( T \):

**Theorem 6** Let \( T \) is a point out of the hyperbola, for any two chords \( TPQ \) and \( TRS \) of the hyperbola, suppose \( PR \) and \( QS \) intersect at \( X \), \( RQ \) and \( PS \) intersect at \( Y \), then the intersection points \( A \) and \( B \) of \( XY \) and the hyperbola are the tangent points from \( T \).

\begin{lstlisting}[language=TeX]
psthypbolaTangentNode[Options] (O)(a, b){T}{A}{B}
\end{lstlisting}

**3.8. Standard Conjugate Hyperbola**

The Standard Conjugate Hyperbola \( H \) with coordinate translation is defined by center \( O \), the half of the real axis \( a \), the half of the imaginary axis \( b \). The equation can be written as:

\[
\frac{(y - y_0)^2}{a^2} - \frac{(x - x_0)^2}{b^2} = 1
\]  

and the parametric function can be written as:

\[
\begin{align*}
x &= b \tan \alpha + x_0 \\
y &= a \sec \alpha + y_0
\end{align*}
\]  

The macro \texttt{psthypbola} is used to draw a Standard Conjugate Hyperbola with Center \( O \), the half of the real axis \( a \), the half of the imaginary axis \( b \). The parameter \texttt{angleY} is used to truncate the height of the figure, it should be setup from 0 to 90.
3. Conics

The macro \texttt{\pstricksname{HyperbolaNode}} is used to draw a node whose parameter is the given value $t$ on Conjugate Hyperbola, please refer to equation (17). The macro \texttt{\pstricksname{HyperbolaAbsNode}} is used to draw the nodes whose abscissa are the given value $x_1$ on Conjugate Hyperbola. The macro \texttt{\pstricksname{HyperbolaOrdNode}} is used to draw the nodes whose ordinate are the given value $y_1$ on Conjugate Hyperbola.

Note that \texttt{\pstricksname{HyperbolaAbsNode}} and \texttt{\pstricksname{HyperbolaOrdNode}} will create two nodes $A$ and $B$ at most time.

\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\end{pspicture}

The macro \texttt{\pstricksname{HyperbolaFocusNode}} is used to find the focus nodes of the Conjugate Hyperbola, and the macro \texttt{\pstricksname{HyperbolaDirectrixLine}} is used to find the directrix lines of the Conjugate Hyperbola.

\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\end{pspicture}

Note that you can use \texttt{\pstricksname{LineAS}} to draw the asymptote line of the hyperbola by passing the slope gradient $k = \pm \frac{a}{b}$; or you can use the macro \texttt{\pstricksname{HyperbolaAsymptoteLine}} to get them, this macro only create one node on each asymptote line, as the other one is the center of the hyperbola.

\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\end{pspicture}
The macro \texttt{\textbackslash pstIHyperbolaLineInter} is used to find the intersections \( C \) and \( D \) of the hyperbola and the given line \( AB \).

\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\psset{dotsize=0.5}\psset{PointSymbol=*}\footnotesize
\def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.35}
\pstGeode [PosAngle=180,PointName=0.2](1,1){0}
\pstIHyperbola [linecolor=blue!40,PointSymbol=\textbullet](0,0){(a,b)}{80}
\pstIHyperbolaFocusNode [linecolor=blue!40,PointSymbol=\textbullet](F,-2,3,4){0.1}{(a,b)}{80}
\pstIHyperbolaDirectrixLine [linecolor=blue!40,nodesepA=2,nodesepB=1,PointNameSep=0.2](0,0){(a,b)}{80}
\pstLineAS [PointName=5,6,7,8]{0.8}{0}(\a,\b){80}
\pstIHyperbolaLineInter [linecolor=red!40,PointNameSep=0.2](0,0){(b,\a)}{78}
\end{pspicture}

In the following example, the Line \( CX \) and \( CY \) are parallel to the asymptote of the hyperbola.

\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\psset{dotsize=0.5}\psset{PointSymbol=*}\footnotesize
\def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.35}
\pstGeode [PosAngle=0,180,PointName=0,1]{0,1}{C}
\pstIHyperbola [linecolor=blue!40,PointNameSep=\textbullet](0,0){(a,b)}{82}
\pstLine [linestyle=dashed,PointName=0,1,2,3]{0.8}{0}(2,2,2,4)
\pstIHyperbolaLineInter [linecolor=blue!40,PointName=0,1,2,3,4](0,0){(a,b)}{82}
\end{pspicture}

The macro \texttt{\textbackslash pstIHyperbolaPolarNode} is used to find the polar point \( T \) of chord \( AB \) on the hyperbola.

\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\psset{dotsize=0.5}\psset{PointSymbol=*}\footnotesize
\def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.35}
\pstGeode [PosAngle=0,180]{0,1}{X}
\pstIHyperbola [linecolor=blue!40,PointNameSep=\textbullet](0,0){(a,b)}{82}
\pstIHyperbolaLineInter [linecolor=blue!40,PointName=0,1,2,3,4](0,0){(a,b)}{82}
\end{pspicture}

We also use the theorem 5 to find the polar point \( T \) of chord \( AB \):
The macro \texttt{pstIHyperbolaTangentNode} is used to find the tangent point \(A\) and \(B\) of point \(T\) outside of the hyperbola.

\begin{verbatim}
\pstIHyperbolaTangentNode [Options] (O) (a, b) \{T\} \{A\} \{B\}
\end{verbatim}

We also use the theorem 6 to find the tangent points \(A\) and \(B\) of \(T\).

\begin{verbatim}
\pstIHyperbolaTangentNode [Options] (O) (a, b) \{T\} \{A\} \{B\}
\end{verbatim}

### 3.9. General Hyperbola

The General Hyperbola \(H\) with coordinate translation and rotation is defined by center \(O\), the half of the real axis \(a\), the half of the imaginary axis \(b\), and the rotation angle \(\theta\) of the principal axis. The equation can be got from the parametric function of the Standard Hyperbola equation (15), using the rotation transform formula (3), then we have

\[
\begin{align*}
x' &= (a \sec \alpha + x_0) \cos \theta - (b \tan \alpha + y_0) \sin \theta = x'_0 + a \sec \alpha \cos \theta - b \tan \alpha \sin \theta \\
y' &= (a \sec \alpha + x_0) \sin \theta + (b \tan \alpha + y_0) \cos \theta = y'_0 + a \sec \alpha \sin \theta + b \tan \alpha \cos \theta
\end{align*}
\]

(18)

where the \(x'_0\) and \(y'_0\) are the coordinate of the given center \(O\) after rotation. So we get the parametric function of the General Hyperbola with coordinate translation and rotation as following:

\[
\begin{align*}
x &= x_0 + a \sec \alpha \cos \theta - b \tan \alpha \sin \theta \\
y &= y_0 + a \sec \alpha \sin \theta + b \tan \alpha \cos \theta
\end{align*}
\]

(19)

The macro \texttt{pstGeneralHyperbola} is used to draw a General Hyperbola with Center \(O\), the half of the real axis \(a\), the half of the imaginary axis \(b\), and the rotation angle \(\theta\) of the symmetrical axis. The parameter \texttt{angleX} is used to truncate the width of the figure, it should be setup from 0 to 90.
The Macro \texttt{pstGeneralHyperbolaFle} is used to define a General Hyperbola with Focus \textit{F}, directrix line \textit{l}, and the eccentricity \textit{e}, where \textit{e} > 1. It just calculate the center \textit{O}, real radius \textit{a}, imaginary radius \textit{b} and the rotation angle \textit{θ} of the real axis, then you can pass them into macro \texttt{pstGeneralHyperbola} to draw this hyperbola.

\begin{verbatim}
\pstGeneralHyperbolaFle                     \{F\}{A}{B}{e}{O}{Rab}{θ}
\end{verbatim}

The output parameter \texttt{O} is a node name to store the center point, its label and symbol can be controlled by the options for PSTricks node, such as \texttt{PosAngle}. The output parameter \texttt{Rab} is a PostScript key to store the pair of real radius and imaginary radius, it just use PSTricks node coordinate to store a pair of value, but not a geometrical point. The output parameter \textit{θ} is also a PostScript key to store the rotation angle of real axis, when you pass it to \texttt{pstGeneralHyperbola}, PostScript will lookup the value of this key in current dictionary.

\begin{verbatim}
\end{verbatim}

The Macro \texttt{pstGeneralHyperbolaFFN} is used to define a General Hyperbola by the given focus nodes \textit{F}_1, \textit{F}_2, and one node \textit{N} on it. It just calculate the center \textit{O}, major radius \textit{a}, minor radius \textit{b} and the rotation angle \textit{θ} of the major axis, then you can pass them into macro \texttt{pstGeneralHyperbola} to draw this hyperbola.

\begin{verbatim}
\end{verbatim}
\pstGeneralHyperbolaFFN [Options] \{F_1\}{F_2}\{O\}\{Rab\}\{\theta\}

The output parameter 0, the output parameter Rab and the output parameter $\theta$ are same with \pstGeneralHyperbolaFle.

The Macro \pstGeneralHyperbolaCoef is used to define a General Hyperbola by the quadratic curve equation $ax^2 + bxy + cy^2 + dx + ey + f = 0$, it just calculate the center $O$, real radius $a$ and imaginary radius $b$ and the rotation angle $\theta$ of the real axis, then you can pass them into macro \pstGeneralHyperbola to draw this hyperbola. The package pst-func provides macro \psplotImp to draw an implicit defined functions too, but it can’t tell you the geometrical elements like as center or radii, and it will take more time to calculate the function value point by point.

\pstGeneralHyperbolaCoef [Options] \{a,b,c,d,e,f\}\{O\}\{Rab\}\{\theta\}

The output parameter 0, the output parameter Rab and the output parameter $\theta$ are same with \pstGeneralHyperbolaFle. They are set to zero if the coeffients are invalid to construct a hyperbola.

In the following example, we use \psplotImp to draw the same hyperoblas, just to check the results given by macros \pstGeneralHyperbolaCoef are correct.
The Macro `\pstGeneralHyperbolaABCDE` is used to define a General Hyperbola by the given five points $A, B, C, D, E$, it just calculate the center $O$, real radius $a$ and imaginary radius $b$ and the rotation angle $\theta$ of the real axis, then you can pass them into macro `\pstGeneralHyperbola` to draw this hyperbola.

```
\pstGeneralHyperbolaABCDE [Options] {A}{B}{C}{D}{E}{O}{Rab}{\theta}
```

The output parameter 0, the output parameter $Rab$ and the output parameter $\theta$ are same with `\pstGeneralHyperbola`. They are set to zero if the points are invalid to construct a hyperbola.

```
\begin{pspicture}[showgrid=true](-1,-3)(4,2)
\psset{unit=0.5cm,\footnoterule,\psset{PointSymbol=*}}
\psset{CodeFig=true,CodeFigColor=gray!50}
\pstGeonode [PosAngle=180](0,0){A}
\pstGeonode [PosAngle=90](2,-1){B}
\pstGeonode [PosAngle=-90](3,-3){C}
\pstGeonode [PosAngle=90](4,0){D}
\pstGeonode [PosAngle=0](5,2){E}
\pstGeneralHyperbolaABCDE [PosAngle=0]{A}{B}{C}{D}{E}{O}{R}{\theta}
\pspicture[0.5]
\end{pspicture}
```

The macro `\pstGeneralHyperbolaNode` is used to draw a node whose parameter is the given value $t$ on General Hyperbola, please refer to equation (19). The macro `\pstGeneralHyperbolaAbsNode` is used to draw the nodes whose abscissa are the given value $x_1$ on General Hyperbola. The macro `\pstGeneralHyperbolaOrdNode` is used to draw the nodes whose ordinate are the given value $y_1$ on General Hyperbola.

Note that `\pstGeneralHyperbolaAbsNode` and `\pstGeneralHyperbolaOrdNode` will create two nodes $A$ and $B$ at most time.

```
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*,\footnoterule,\psset{PointNameSep=0.3}}
\def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.3}
\pstGeonode [PosAngle=-90](1,1){O}
\pstGeneralHyperbola [linecolor=purple!80](0){(\a,\b)[150][80]}
\pstGeneralHyperbolaAbsNode [linecolor=purple!80,PosAngle=(200,90)](0){(\a,\b)[150][2]}(P)
\pstGeneralHyperbolaAbsNode [linecolor=purple!80,PosAngle=(-90,280)](0){(\a,\b)[150][80]}(X)(Y)
\pstGeneralHyperbolaAbsNode [linecolor=purple!80,PosAngle=(-40,-40)](0){(\a,\b)[150][80]-0.59378}(M)(N)
\pstLine [linestyle=dashed,\linecolor=gray!40]{0.59378,-1}{0.59378,3}
\pstGeneralHyperbolaOrdNode [linecolor=purple!80,PosAngle=(200,90)](0){(\a,\b)[150][2]}(G)(H)
\pstGeneralHyperbolaOrdNode [linecolor=purple!80,PosAngle=(-90,280)](0){(\a,\b)[150][80]}(I)(J)
\pstGeneralHyperbolaOrdNode [linecolor=purple!80,PosAngle=(90,90)](0){(\a,\b)[150][1]}(K)(L)
\pstLine [linestyle=dashed,\linecolor=gray!80,\nodesep=-1.5] (K)(L)
\end{pspicture}
```
The macro \texttt{pstGeneralHyperbolaFocusNode} is used to find the focus nodes of the General Hyperbola, the macro \texttt{pstGeneralHyperbolaVertexNode} is used to find the vertex nodes of the General Hyperbola, and the macro \texttt{pstGeneralHyperbolaDirectrixLine} is used to find the directrix lines of the General Hyperbola.

\begin{verbatim}
\pstGeneralHyperbolaFocusNode [Options] (O)(a, b) [\theta] \{F_1\}{F_2}
\pstGeneralHyperbolaVertexNode [Options] (O)(a, b) [\theta] \{V_1\}{V_2}
\pstGeneralHyperbolaDirectrixLine [Options] (O)(a, b) [\theta] \{L_x\}{L_y}\{R_x\}{R_y}
\end{verbatim}

Note that you can use the macro \texttt{pstGeneralHyperbolaAsymptoteLine} to get the asymptote lines, this macro only create one node on each asymptote line, as the other one is the center of the hyperbola.

\begin{verbatim}
\begin{pspicture}(-2,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=\*}\footnotesize\def\a{0.5}\def\b{0.3}
\pstGeonode[PosAngle=180,PointNameSep=0.2]((1,1)\}(O)
\pstGeneralHyperbola[linecolor=red!40](0)(\a,\b)[80]
\pstGeneralHyperbolaFocusNode[linecolor=red!40,PointName={X_1,X_2},PosAngle=(180,0)](0)(\a,\b)[80](X1)(X2)
\pstGeneralHyperbolaDirectrixLine[linecolor=red!40,PointNameSep=2,PointName={L_x,L_y,R_x,R_y}](O)(\a,\b)[80](L1)(L2)
\pstGeneralHyperbolaAsymptoteLine[linecolor=red!40,PointNameSep=2,PointName={L_1,L_2}](O)(\a,\b)[80](L1)(L2)
\pstGeneralHyperbolaAsymptoteLine[linecolor=blue!40,PointName={L_1,L_2}](O)(\a,\b)[40][80]
\pspicture[showgrid=true](-2,-2)(4,4)
\end{pspicture}
\end{verbatim}

The macro \texttt{pstGeneralHyperbolaLineInter} is used to find the intersections $C$ and $D$ of the general hyperbola and the given line $AB$.

\begin{verbatim}
\pstGeneralHyperbolaLineInter [Options] (O)(a, b) [\theta] \{A\}{B}\{C\}{D}
\end{verbatim}

In the following example, the lines $YY'$ and $ZZ'$ are parallel to the asymptote of the hyperbola, so there are only one intersection $M$ and $P$ for each line, and the second node $N$ and $Q$ are put at the origin.
The macro `\pstGeneralHyperbolaPolarNode` is used to find the polar point $T$ of chord $AB$ on the general hyperbola.

\begin{verbatim}
\pstGeneralHyperbolaPolarNode [Options] (O)(a, b) [\theta] {A}{B}{T}
\end{verbatim}

We also use the theorem 5 to find the polar point $T$ of chord $AB$:

\begin{verbatim}
\pstGeneralHyperbolaTangentNode [Options] (O)(a, b) [\theta] {T}{A}{B}
\end{verbatim}

The macro `\pstGeneralHyperbolaTangentNode` is used to find the tangent point $A$ and $B$ of point $T$ outside of the general hyperbola.
3.10. General Conjugate Hyperbola

The General Conjugate Hyperbola \( H \) with coordinate translation and rotation is defined by center \( O \), the half of the real axis \( a \), the half of the imaginary axis \( b \), and the rotation angle \( \theta \) of the principal axis. The equation can be got from the parametric function of the Standard Conjugate Hyperbola equation (17), using the rotation transform formula (3), then we have

\[
\begin{align*}
  x' &= (b \tan \alpha + x_0) \cos \theta - (a \sec \alpha + y_0) \sin \theta = x_0' + b \tan \alpha \cos \theta - a \sec \alpha \sin \theta \\
y' &= (b \tan \alpha + x_0) \sin \theta + (a \sec \alpha + y_0) \cos \theta = y_0' + b \tan \alpha \sin \theta + a \sec \alpha \cos \theta
\end{align*}
\]

where the \( x_0' \) and \( y_0' \) are the coordinate of the given center \( O \) after rotation. So we get the parametric function of the General Conjugate Hyperbola with coordinate translation and rotation as following:

\[
\begin{align*}
x &= x_0 + b \tan \alpha \cos \theta - a \sec \alpha \sin \theta \\
y &= y_0 + b \tan \alpha \sin \theta + a \sec \alpha \cos \theta
\end{align*}
\]

The macro \texttt{pstGeneralHyperbola} is used to draw a General Conjugate Hyperbola with Center \( O \), the half of the real axis \( a \), the half of the imaginary axis \( b \), and the rotation angle \( \theta \) of the symmetrical axis. The parameter \texttt{angleY} is used to truncate the height of the figure, it should be setup from 0 to 90.

\texttt{\begin{ppicture} [showgrid=true] (-2,-1)(4,3) \psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize \def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.3} \pstGeoNode [PosAngle=120](1,1){O} \pstGeneralHyperbola [linecolor=blue!40](0)(\a,\b)[40][80] \pstGeneralHyperbolaNode [linecolor=red!40,PosAngle=120](0)(\a,\b)\{T\}{A}{B} \pstGeneralHyperbola [linecolor=red!40,PosAngle=-90](1,2,0.8){T} \pstGeneralHyperbolaTangentNode [linecolor=red!40,PosAngle=(140,-90),nodesep=-0.5](0)(\a,\b)\{T\}{A}{B} \end{ppicture}}

\texttt{\begin{ppicture} [showgrid=true] (-2,-1)(4,3) \psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize \def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.3} \pstGeoNode [PosAngle=90](1,1){O} \pstGeneralHyperbola [linecolor=blue!40](0)(\a,\b)[40][80] \pstGeneralHyperbolaNode [linecolor=blue!40,PosAngle=190](0)(\a,\b)\{O\}{A}{B} \pstGeneralHyperbola [linecolor=red!40,PosAngle=-90](1,2,0.8){T} \pstGeneralHyperbolaTangentNode [linecolor=red!40,PosAngle=(140,-90),nodesep=-0.5](0)(\a,\b)\{T\}{A}{B} \pstGeneralHyperbola [linecolor=green!60](0)(\a,\b)[90][80] \pstGeneralHyperbolaNode [linecolor=green!60,PosAngle=90](0)(\a,\b)\{O\}{A}{B} \pstGeneralHyperbola [linecolor=green!60,PosAngle=0](0)(\a,\b)\{O\}{A}{B} \end{ppicture}}

The macro \texttt{pstGeneralHyperbolaNode} is used to draw a node whose parameter is the given value \( t \) on General Conjugate Hyperbola, please refer to equation (21).
The macro \texttt{\texttt{pstGeneralIHyperbolaAbsNode}} is used to draw the nodes whose abscissa are the given value $x_1$ on General Conjugate Hyperbola. The macro \texttt{\texttt{pstGeneralIHyperbolaOrdNode}} is used to draw the nodes whose ordinate are the given value $y_1$ on General Conjugate Hyperbola.

Note that \texttt{\texttt{pstGeneralIHyperbolaAbsNode}} and \texttt{\texttt{pstGeneralIHyperbolaOrdNode}} will create two nodes $A$ and $B$ at most time.

\begin{verbatim}
\ptsetGeneralIHyperbolaAbsNode[Options] (O)(a, b) [\theta] \{\{A\}\{B\}
\ptsetGeneralIHyperbolaOrdNode[Options] (O)(a, b) [\theta] \{\{A\}\{B\}
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{nodesep=0.5}\psset{PointSymbol=*}
\def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.3}
\ptsetGeonode[PosAngle=180]{1,1}{O}
\ptsetGeneralIHyperbolaAbsNode[linecolor= purple!80]{0}{\a, \b}[150]{2}{P}(Q)
\ptsetGeneralIHyperbolaOrdNode[linecolor= purple!80,PosAngle =200,90]{0}{\a, \b}[150]{2}{P}(Q)
\ptsetGeneralIHyperbolaAbsNode[linecolor= purple!80,PosAngle =(90,200)]{0}{\a, \b}[150]{1}(X)(Y)
\ptsetGeneralIHyperbolaOrdNode[linecolor= purple!80,PosAngle =(90,200)]{0}{\a, \b}[150]{1}(X)(Y)
\ptsetGeneralIHyperbolaAbsNode[linecolor= purple!80,PosAngle =80]{0}{\a, \b}[150]{1}(M)(N)
\ptsetGeneralIHyperbolaOrdNode[linecolor= purple!80,PosAngle =80]{0}{\a, \b}[150]{1}(M)(N)
\ptsetLine[linestyle=dashed, linecolor=gray!40, nodesep=1.5]{M}{N}
\ptsetLine[linestyle=dashed, linecolor=gray!40, nodesep=1.5]{M}{N}
\ptsetGeneralIHyperbolaAbsNode[linecolor= purple!80,PosAngle =80]{0}{\a, \b}[150]{1}(I){J}
\ptsetGeneralIHyperbolaAbsNode[linecolor= purple!80,PosAngle =80]{0}{\a, \b}[150]{1}(I){J}
\ptsetGeneralIHyperbolaOrdNode[linecolor= purple!80,PosAngle =80,1.4063]{0}{\a, \b}[150]{1}(L){K}
\ptsetGeneralIHyperbolaOrdNode[linecolor= purple!80,PosAngle =80,1.4063]{0}{\a, \b}[150]{1}(L){K}
\ptsetLine[linestyle=dashed, linecolor=gray!80, nodesep=1]{I}{J}
\ptsetLine[linestyle=dashed, linecolor=gray!80, nodesep=1]{I}{J}
\ptsetLine[linestyle=dashed, linecolor=gray!80, nodesep=1]{I}{J}
\ptsetLine[linestyle=dashed, linecolor=gray!80, nodesep=1]{I}{J}
\end{pspicture}
\end{verbatim}

The macro \texttt{\texttt{pstGeneralIHyperbolaFocusNode}} is used to find the focus nodes of the General Conjugate Hyperbola, the macro \texttt{\texttt{pstGeneralIHyperbolaVertexNode}} is used to find the vertex nodes of the General Conjugate Hyperbola, and the macro \texttt{\texttt{pstGeneralIHyperbolaDirectrixLine}} is used to find the directrix lines of the General Conjugate Hyperbola.

\begin{verbatim}
\ptsetGeneralIHyperbolaFocusNode[Options] (O)(a, b) [\theta] \{F_1\}{F_2}
\ptsetGeneralIHyperbolaVertexNode[Options] (O)(a, b) [\theta] \{V_1\}{V_2}
\ptsetGeneralIHyperbolaDirectrixLine[Options] (O)(a, b) [\theta] \{L_x\}{L_y}\{R_x\}{R_y}
\end{verbatim}

Note that you can use the macro \texttt{\texttt{pstGeneralIHyperbolaAsymptoteLine}} to get the asymptote lines, this macro only create one node on each asymptote line, as the other one is the center of the hyperbola.

\begin{verbatim}
\ptsetGeneralIHyperbolaAsymptoteLine[Options] (O)(a, b) [\theta] \{L_1\}{L_2}
\end{verbatim}
The macro \texttt{pstGeneralIHyperbolaLineInter} is used to find the intersections $C$ and $D$ of the general conjugate hyperbola and the given line $AB$.

$$\texttt{pstGeneralIHyperbolaLineInter[Options]}(O)(a, b)\{\theta\}\{A\}{B}\{C\}{D}$$

In the following example, the lines $YY'$ and $ZZ'$ are parallel to the asymptote of the hyperbola, so there are only one intersection $M$ and $P$ for each line, and the second node $N$ and $Q$ are put at the origin.
We also use the theorem 5 to find the polar point $T$ of chord $AB$ on the general conjugate hyperbola.

\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize\def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.3}
\pstGenode[PosAngle=225\angle](1,1){O}
\pstGeneralIHyperbolaPolarNode[Options]{(a, b)}[\theta]{A}{B}{T}
\end{pspicture}

We also use the theorem 6 to find the tangent points $A$ and $B$ of point $T$ outside of the general conjugate hyperbola.

\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize\def\a{0.5}\def\b{0.3}\psset{PointNameSep=0.3}
\pstGenode[PosAngle=40\angle](0,0){O}
\pstGeneralIHyperbolaTangentNode[Options]{(a, b)}[\theta]{A}{B}
\end{pspicture}
3. Conics

3.11. General Conics

A General Conic is defined by the quadratic curve equation

\[ ax^2 + bxy + cy^2 + dx + ey + f = 0 \]  \hspace{1cm} (22)

it can be reduced to an ellipse, hyperbola or parabola after translation and rotation.

In the previous sections, we use the macros \texttt{\textbackslash pstGeneralEllipseCoef}, \texttt{\textbackslash pstGeneralHyperbolaCoef}, \texttt{\textbackslash pstGeneralParabolaCoef} to define the general ellipse, general hyperbola and general parabola from the quadratic curve equation (22). Here we provide the macros \texttt{\textbackslash pstGeneralConicEquation}, \texttt{\textbackslash pstGeneralEllipseEquation}, \texttt{\textbackslash pstGeneralHyperbolaEquation} and \texttt{\textbackslash pstGeneralParabolaEquation} to get the coefficients \(a, b, c, d, e, f\) of their quadratic curve equation (22).

\begin{verbatim}
\pstGeneralConicEquation [Options] {A}{B}{C}{D}{E}{F}{a,b,c,d,e,f}
\pstGeneralEllipseEquation [Options] (O) (a,b) [\theta] {a,b,c,d,e,f}
\pstGeneralHyperbolaEquation [Options] (O) (a,b) [\theta] {a,b,c,d,e,f}
\pstGeneralParabolaEquation [Options] (O) {p} [\theta] {a,b,c,d,e,f}
\end{verbatim}

The macro \texttt{\textbackslash pstGeneralConicEquation} take five points \(A, B, C, D, E\) as input parameter; the macro \texttt{\textbackslash pstGeneralEllipseEquation} take the general ellipse parameter \(O, a, b, \theta\) as input parameter; the macro \texttt{\textbackslash pstGeneralHyperbolaEquation} take the general hyperbola parameter \(O, a, b, \theta\) as input parameter; the macro \texttt{\textbackslash pstGeneralParabolaEquation} take the general parabola parameter \(O, p, \theta\) as input parameter. They all output the six coefficients \(a, b, c, d, e, f\) of the quadratic curve equation.

Note that the output coefficients \(a, b, c, d, e, f\) are the variables in PostScript level, they will be dumped out in PostScript’s console when you set CodeFig to true. You can pass them into the macros where need these six coefficients. In order to prevent the name conflict in PostScript level, You should setup the name of the coefficients more than three letters, for example,

% calculate the coefficients CoefA,CoefB,...,CoefF and CoefA',CoefB',...,CoefF'
\pstGeneralConicEquation(A){B}{C}{D}{E}{CoefA,CoefB,CoefC,CoefD,CoefE,CoefF}
\pstGeneralConicEquation(A'){B'}{C'}{D'}{E'}{CoefA',CoefB',CoefC',CoefD',CoefE',CoefF'}
% then use them to find the intersections
\pstGeneralConicInter{CoefA,CoefB,CoefC,CoefD,CoefE,CoefF}{CoefA',CoefB',CoefC',CoefD',CoefE',CoefF'}{I}{J}{I'}{J'}
The following example output the coefficients \(a, b, c, d, e, f\) for the ellipse defined by five points, and then draw the ellipse using these coefficients to check the result.

\[
\begin{pspicture}(-1.2,-2)(4.3)
\psset{unit=0.6cm,\footnotemarksize,PointSymbol=**}
\psset{CodeFig=false,CodeFigColor=gray!50}
\pstGeonode\PosAngle=180,0\{\text{A}\}(0,0)\{\text{B}\}(2,1)\{\text{C}\}(3,3)\{\text{D}\}(4,0)
\psline(0,0)(2,1)(3,3)(4,0)(0,0)
\end{pspicture}
\]

\[
\text{The following example output the coefficients } a, b, c, d, e, f \text{ for the hyperbola } 2xy - x + 2y + 1 = 0, \text{ just for example, and then draw the hyperbola using these coefficients to check the result.}
\]

\[
\begin{pspicture}(-3,-2)(2,2)
\psset{unit=0.6cm,\footnotemarksize,PointSymbol=**}
\psset{CodeFig=false,CodeFigColor=gray!50}
\psset{CodeFig=\footnotemarksize,PointSymbol=**}
\psset{CodeFig=false,CodeFigColor=gray!50}
\pstGeonode\PosAngle=180,0\{\text{A}\}(0,0)\{\text{B}\}(2,1)\{\text{C}\}(3,3)\{\text{D}\}(4,0)
\psline(0,0)(2,1)(3,3)(4,0)(0,0)
\end{pspicture}
\]

\[
\text{The following example output the coefficients } a, b, c, d, e, f \text{ for the parabola defined by five points, and then draw the parabola using these coefficients to check the result.}
\]

\[
\begin{pspicture}(-3,-2)(2,2)
\psset{unit=0.6cm,\footnotemarksize,PointSymbol=**}
\psset{CodeFig=false,CodeFigColor=gray!50}
\psset{CodeFig=\footnotemarksize,PointSymbol=**}
\psset{CodeFig=false,CodeFigColor=gray!50}
\pstGeonode\PosAngle=180,0\{\text{A}\}(0,0)\{\text{B}\}(2,1)\{\text{C}\}(3,3)\{\text{D}\}(4,0)
\psline(0,0)(2,1)(3,3)(4,0)(0,0)
\end{pspicture}
\]
Here are some macros to find the intersections with the quadratic curve equation \((22)\).

- \texttt{\pstGeneralConicLineInter[Options] \{A\}\{B\}\{a,b,c,d,e,f\}\{C\}\{D\}}
- \texttt{\pstGeneralConicCircleInter[Options] \{O\}\{A\}\{a,b,c,d,e,f\}\{C\}\{D\}\{E\}\{F\}}
- \texttt{\pstGeneralConicEllipseInter[Options] \{O\}\{m,n\}\{a,b,c,d,e,f\}\{C\}\{D\}\{E\}\{F\}}
- \texttt{\pstGeneralConicHyperbolaInter[Options] \{O\}\{m,n\}\{a,b,c,d,e,f\}\{C\}\{D\}\{E\}\{F\}}
- \texttt{\pstGeneralConicParabolaInter[Options] \{O\}\{p\}\{a,b,c,d,e,f\}\{C\}\{D\}\{E\}\{F\}}
- \texttt{\pstGeneralConicIParabolaInter[Options] \{O\}\{p\}\{a,b,c,d,e,f\}\{C\}\{D\}\{E\}\{F\}}

The macro \texttt{\pstGeneralConicLineInter} is used to find the intersections \(C, D\) of the quadratic curve equation \((22)\) and the given line \(AB\).

The macro \texttt{\pstGeneralConicCircleInter} is used to find the intersections \(C, D, E, F\) of the quadratic curve equation \((22)\) and the given circle \((x - x_A)^2 + (y - y_A)^2 = |OA|^2\).

The macro \texttt{\pstGeneralConicEllipseInter} is used to find the intersections \(C, D, E, F\) of the quadratic curve equation \((22)\) and the given ellipse \(\frac{(x - x_O)^2}{m^2} + \frac{(y - y_O)^2}{n^2} = 1\).

The macro \texttt{\pstGeneralConicHyperbolaInter} is used to find the intersections \(C, D, E, F\) of the quadratic curve equation \((22)\) and the given hyperbola \(\frac{(x - x_O)^2}{m^2} - \frac{(y - y_O)^2}{n^2} = 1\).

The macro \texttt{\pstGeneralConicIParabolaInter} is used to find the intersections \(C, D, E, F\) of the quadratic curve equation \((22)\) and the given conjugate hyperbola \(\frac{(y - y_O)^2}{m^2} - \frac{(x - x_O)^2}{n^2} = 1\).

The macro \texttt{\pstGeneralConicParabolaInter} is used to find the intersections \(C, D, E, F\) of the quadratic curve equation \((22)\) and the given parabola \((x - x_O)^2 = 2p(y - y_O)\).

The macro \texttt{\pstGeneralConicIParabolaInter} is used to find the intersections \(C, D, E, F\) of the quadratic curve equation \((22)\) and the given conjugate parabola \((y - y_O)^2 = 2p(x - x_O)\).

The macro \texttt{\pstGeneralConicInter} is used to find the intersections \(C, D, E, F\) of the quadratic curve equation \((22)\) and the other one.
Here are some macros to find the tangent line or tangent chord of the quadratic curve equation \( (22). \)

\[
\text{\texttt{\textbf{pstGeneralConicTangentLine [Options] \{A\}\{b,c,d,e,f\}\{B\}}}\\
\text{\texttt{\textbf{pstGeneralConicTangentChord [Options] \{T\}\{a,b,c,d,e,f\}\{A\}\{B\}}}}\\
\]

The macro \texttt{\textbf{pstGeneralConicTangentLine}} find the Tangent Line of the quadratic curve through the point \( A \) which is lie on the curve, output the other node \( B \) on the tangent line. The macro \texttt{\textbf{pstGeneralConicTangentChord}} find the Tangent Chord of the quadratic curve through the point \( T \) which is not lie on the curve, output the tangent point \( A \) and \( B \) on the curve.
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 and 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.

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 \((AB)\).
4.3. Rotation

\[
\text{\texttt{\textbackslash pstRotation}} [\text{\textbackslash Options}] \{O\} \{M_1, M_2, \ldots, M_n\} [M'_1, M'_2, \ldots, M'_p]
\]

\[
\text{\texttt{\textbackslash pstAngleAOB}} \{A\} \{O\} \{B\}
\]

Possible optional arguments are PointSymbol, PosAngle, PointName, PointNameSep, PtNameMath, and RotAngle for \texttt{\textbackslash pstRotation} and AngleCoef, RotAngle for \texttt{\textbackslash 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.

4.4. Translation

\[
\text{\texttt{\textbackslash pstTranslation}} [\text{\textbackslash Options}] \{A\} \{B\} \{M_1, M_2, \ldots, M_n\} [M'_1, M'_2, \ldots, M'_p]
\]

Possible optional arguments are PointSymbol, PosAngle, PointName, PointNameSep, PtNameMath, and DistCoef Draws 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 multiplicative coefficient to modify the translation vector. The parameter CodeFig draws the translation vector \(\vec{AB}\) et 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).
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.

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, and CodeFigStyle. 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 example.

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.
5. Special object

\begin{pspicture}[showgrid](-3,-2)(2,2)
\psTriangle[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)
\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 segments 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. Special object

5.4. Perpendicular bisector of a segment

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

Possible optional arguments are \texttt{PointName}, \texttt{PointNameSep}, \texttt{PosAngle}, \texttt{PointSymbol}, \texttt{PtNameMath}, \texttt{CodeFig}, \texttt{CodeFigColor}, \texttt{CodeFigStyle}, and \texttt{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:

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

5.5. Bisectors of angles

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

Possible optional arguments are \texttt{PointSymbol}, \texttt{PosAngle}, \texttt{PointName}, \texttt{PointNameSep}, and \texttt{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 \(B\).
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

\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}

\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}
6. Intersections

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 specified by macros \pstDist, \pstDistMul, \pstDistAdd, \pstDistSub etc.

The position of the wo points is such that the vectors \(\vec{AB}\) and \(\vec{OC}\) are in the same direction. Thus, if the points defining 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.

\pstset{unit=0.8}
\begin{pspicture}(-3,-2)(4,4)
\pstGeonode[PosAngleA=-135,PosAngleB=80,PointSymbol=bullet](1,0){I}
\pstGeonode[PosAngleA=-80,PosAngleB=0,PointSymbol=bullet](1,0){J}
\pstInterLC[PointName=I,Diameter=60,RADIUS=\pstDistAB{O}{D}]{I}{C}{O}{}{F}{G}
\pstInterLC[PosAngleB=180,Diameter=\pstDistMul{O}{D}{1.3}]{I}{B}{O}{}{H}{J}
\pstCircleOA[linestyle=none,PointName=E]{O}{C}
\pstLineAB[linestyle=none,PointName=E]{C}
\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 CodeFigArcA and CodeFigBarc specified the direction of these optional arcs: trigonometric (by default) or clockwise. Here is a first example.
6. Intersections

And a more complete one, which includes the special circle specification using radius and diameter. For such specifications it exists the parameters \( \text{RadiusA} \), \( \text{RadiusB} \), \( \text{DiameterA} \) and \( \text{DiameterB} \).

The macro \( \text{\textbackslash pstInterCC} \) will not display the intersections as default, if you want to display the label or symbol of the intersections, you must setup the parameters \( \text{PosAngleA} \) and \( \text{PosAngleB} \) to change the default behavior.

\begin{verbatim}
\begin{pspicture}[showgrid](0,-1)(4,3)
\psset{dash=2mm 2mm}
\put(10){%
\pstGeonode[PosAngle=\{0,-90,90,90\}]
(1,-1){A}(2,0.1){B}(2.5,1){C}
\pstCircleOA[linewidth=1pt,linestyle=none,\psset{linecolor=magenta}](0){A}(B)(C)(D)(E)
\pstInterCC[PosAngleA=130, CodeFigA=true, CodeFigAarc=false,\psset{CodeFigColor=magenta}](0){A}(B)(C)(D)(E)
\end{pspicture}
\end{verbatim}

\begin{verbatim}
\begin{pspicture}[showgrid](-3,-4)(7,3)
\pstGeonode[PointName=\{\Omega,0\}](3,-1){Omega}
\pstGeonode[PointSymbol=square, PosAngle={90,90}](0,3){A}(2,2){B}
\psset{PointSymbol=o}
\pstCircleOA[linewidth=1pt,linestyle=none](0){A}(B)(C)(D)(E)
\pstInterCC[PosAngleA=130, CodeFigA=true, CodeFigAarc=false,\psset{CodeFigColor=magenta}](0){A}(B)(C)(D)(E)
\end{pspicture}
\end{verbatim}

6.4. Function–function

\begin{verbatim}
\psInterFF[Options]{f}{g}{x_0}{M}
\end{verbatim}

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 use this function several time if more than one intersection is present. Each function is described in PostScript in the same
6. Intersections

way as the description used by the \psplot macro of PSTricks. A constant function can be specified, and then searching 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)
\psset{linewidth=1.5pt,algebraic}
\psplot[linecolor=gray]{-2}{2}{x^2}
\psplot{2-x/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}

\section*{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(-3,-1)(3,4)
\psplot[linecolor=gray]{-2.5}{2.5}{F}
\psset{PointSymbol=*}
pstGeonode[PosAngle={-45,0}](0,-.2){N}(2.5,1){M}
pstLineAB[nodesepA=-3cm]{N}{M}
pset{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}

\section*{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 \( OA \).
7. Helper Macros

\begin{pspicture}[-3,-4](3,4)
\def\F{2*cos(x)}
\psset{algebraic}
\pstGeonode(0,3)\(O\){0}(2,0.5)\(M\)
\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}

\textbf{7. Helper Macros}

\texttt{\textbackslash psGetDistanceAB[Options]} \((x_1,y_1)(x_2,y_2)\{<name>\}
\texttt{\textbackslash psGetAngleABC[Options]} \((x_1,y_1)(x_2,y_2)(x_3,y_3)\{<symbol>\}

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

\begin{pspicture}[-1,0](11,8)
\psgrid[gridlabels=0pt, subgriddiv=2, width=0.4pt, gridcolor=black!60, subgridcolor=black!40]
\def\sideC{6} \def\sideA{7} \def\sideB{8}
\psset{PointSymbol=none, linejoin=1, linewidth=0.4pt, PointNameMath=false, labelsep=0.07, MarkAngleRadius=1.1, decimals=1, comma}
\pstGeonode[PosAngle={-90,-90}]{\sideC}{A}{\sideB}{B}
\pstInterCC[RadiusA=\psDistVal{\sideB}, RadiusB=\psDistVal{\sideA}, PosAngle=90, PointNameA=C]{A}{B}{C}{D}{E}
\pstInterLC[Radius=\psDistAB{A}{C}]{C}{D}{C}{A'}{A'}
\end{pspicture}
\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\}\{$\beta$}\{E\}
\psGetAngleABC[LabelSep=0.8,AngleArc=false,WedgeColor=green,arrows=->,xShift=-15,yShift=0]\{C\}{E}\{B\}\{$\gamma$}
% \psGetDistanceAB[xShift=-8,yShift=4]\{B\}{E}\{M\}
\psGetDistanceAB[fontscale=15,xShift=4,decimals=0]\{A\}{C}\{M\}
\psGetDistanceAB[xShift=-17,decimals=2]\{E\}{C}\{M\}
\end{pspicture}
Part II.
Examples gallery

A. Basic geometry

A.1. Drawing of the bissector

\begin{pspicture}(0,-1)(4.4,5)
\psset{PointSymbol=none,PointName=none}
\pstGeonode[PosAngle={180,130,-90},PointSymbol=+,PointName=default](2,0){B}(0,1){O}(1,4){A}
\pstLineAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{O}{B}
\pstLineAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{O}{A}
\pstLineAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{B}{C}
\pstArcOAB[linecolor=green,linestyle=dashed,nodesepB=-1]{O}{A}{C}
\pstArcOAB[linecolor=blue,linestyle=dashed,nodesepB=-1]{A}{O}{C}
\pstArcOAB[linecolor=blue,linestyle=dashed,nodesepB=-1]{B}{C}{A}
\psset{arcsepA=-1,arcsepB=-1}
\pstArcOAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{A}{O}{B}
\pstArcOAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{B}{A}{C}
\psset{arcsepA=-1,arcsepB=-1}
\pstArcOAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{C}{O}{A}
\psset{arcsep=1pt,linewidth=1pt,linestyle=dashed,linewidth=1pt,linestyle=dashed}
\pstArcOAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{A}{O}{B}
\psset{arcsepA=-1,arcsepB=-1}
\pstArcOAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{B}{C}{A}
\psset{arcsepA=-1,arcsepB=-1}
\pstArcOAB[nodesepB=-1,linestyle=dashed,linewidth=1pt,linestyle=dashed]{C}{O}{A}
\pstMarkAngle[MarkAngleRadius=.5]{O}{C}{O'}{}{O}{A}
\pstMarkAngle{O}{A}{O'}{}{O'}{O'}
\end{pspicture}
A.2. Transformation de polygones et courbes

Here is an example of the use of CurveType with transformation.
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[C]{A}{J}{JP}
\pstMediatorAB[B]{C}{I}{IP}
\psset{PointSymbol=none}
\pstInterLL[PointSymbol=square, PosAngle=-170]{I}{IP}{J}{JP}{O}
\psset{nodesep=-.8, linecolor=green}
\pstLineAB(O){I}\pstLineAB(O){J}\pstLineAB(O){K}
\psset{PointSymbol=square}
\pstInterLL{A}{A'}{B}{B'}{H}
\psset{linecolor=blue}
\ncline{A}{A'}\ncline{C}{C'}\ncline{B}{B'}
\psset{linecolor=red}
\pstCGravABC{A}{B}{C}{G}
\end{pspicture}
A. Basic geometry

A.4. Euler circle

\begin{pspicture}(-3,-1.5)(3,2.5)
\psset{PointSymbol=none}
\pstTriangle(-2,-1){A}(1,2){B}(2,-1){C}
\pstMediatorAB(A){B}{K}{KP}
\pstMediatorAB(C){A}{J}{JP}
\pstMediatorAB(B){C}{I}{IP}
\pstInterLL[I]{I}{IP}{J}{JP}{O}
\psdot[dotstyle=square](O)
\pstProjection{B}{A}{C}
\pstProjection{B}{C}{A}
\pstProjection{A}{C}{B}
\psset{linestyle=none, PointSymbolB=none, PointNameB=none}
\pstMediatorAB(A){B}{K}{KP}
\pstMediatorAB(C){A}{J}{JP}
\pstMediatorAB(B){C}{I}{IP}
\pstInterLL[I]{I}{IP}{J}{JP}{O}
\psdot[dotstyle=square](O)
\pstProjection{B}{A}{C}
\pstProjection{B}{C}{A}
\pstProjection{A}{C}{B}
\psset{nodesep=-.8, linecolor=green}
\pstLineAB{O}{I}\pstLineAB{O}{J}\pstLineAB{O}{K}
\psset{nodesep=-.8, linecolor=blue}
\ncline{A}{A'}\ncline{C}{C'}\ncline{B}{B'}
\pstInterLL[I]{I}{IP}{J}{JP}{O}
\psset{PointSymbol=o, PointName=\omega}
\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 $H : y = a/x$ belong to this hyperbola.
A.6. 17 sides regular polygon

Striking picture created by K. F. Gauss. He also proved that it is possible to build the regular polygons which have $2^{2^n} + 1$ sides, the following one has 257 sides!
\pstRightAngle{N.4}{P.1}{P.6}
\pstRightAngle{N.4}{P.1}{P.5}
\pstInterCC{P.1}{P.4}{P.6}{H}{PosAngleA=90, PointSymbolA=none, PointNameA=none}
\pstInterCC{P.1}{P.3}{H}{PosAngleA=90, PointSymbolA=none, PointNameA=none}
\pstInterCC{P.1}{P.6}{H}{PosAngleA=100, PointSymbolA=none, PointNameA=none}
\pstInterCC{P.1}{P.7}{P.9}{H}{PosAngleA=135, PointSymbolA=none, PointNameA=none}
\pspolygon{linecolor=green, linestyle=solid, linewidth=2\pslinewidth}
A.7. Circles & tangents

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

A.8. Fermat’s point

Drawing of Manuel Luque.
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}
A. Basic geometry

\psset{linecolor=magenta, linestyle=dashed}
\pstProjection{(A)(B)(I)(C)}
\pstLineAB{(I)(C)}\pstRightAngle{linestyle=solid}{(A)(I)(C)}
\pstLineAB{(A)(C)(I)}\pstRightAngle{linestyle=solid}{(C)(I)(B)}
\pstLineAB{(I)(IA)}\pstRightAngle{linestyle=solid}{(B)(I.A)}\pstCircleOA{linestyle=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)(BOB)}\pstInterLL{PosAngle=90}{(C)(COB)(BOB)}
\psset{linecolor=magenta, linestyle=dashed}
\pstLineAB{(I)(I1C)}\pstRightAngle{linestyle=solid}{(I1C)(A)}
\pstLineAB{(I)(I1C)}\pstRightAngle{linestyle=solid}{(I1C)(I)(I1B)}
\pstLineAB{(I)(I2B)}\pstRightAngle{linestyle=solid}{(A)(I)(I1B)}
\pstLineAB{(I)(I1A)}\pstRightAngle{linestyle=solid}{(A)(I)(I1A)}
\pstLineAB{(I)(I2B)}\pstRightAngle{linestyle=solid}{(A)(I)(I2B)}
\pstLineAB{(I)(I2C)}\pstRightAngle{linestyle=solid}{(A)(I)(I2C)}
\pstLineAB{(I)(I2A)}\pstRightAngle{linestyle=solid}{(C)(I)(I2A)}
\pstLineAB{(I)(I3A)}\pstRightAngle{linestyle=solid}{(C)(I)(I3A)}
\pstLineAB{(I)(I3C)}\pstRightAngle{linestyle=solid}{(A)(I)(I3C)}
\pstLineAB{(I)(I3B)}\pstRightAngle{linestyle=solid}{(I)(I3B)}
\psset{linecolor=yellow, linestyle=solid}
\pstCircleOA{(I)}\pstCircleOA{(I2B)}\pstCircleOA{(I3A)}
\psset{linecolor=red, linestyle=solid, nodeSepA=-1, nodeSepB=-1}
\pstLineAB{(IIB)(I3B)}\pstLineAB{(IIA)(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]{(5,4)}{A}(9,1.5){B}
\newcommand{Parabole}{\textcolor{green}{\textbullet}}
\pstLineAB[nodesep=-.9, linestyle=dashed, linecolor=green]{A}{B}
\psset{RotAngle=90, PointSymbol=none, PointName =none}
\multido(n=1+1){\NbPt}{%}
\pstHomO[HomCoef=\textcolor{blue}{\textbullet}]{\textcolor{blue}{\textbullet}}{A}{B}{M\bullet}
\pstMediatorAB[linestyle=none]{M\bullet}{O}{M\bullet}{I\bullet}{M\bullet}
\pstRotation{M\bullet}{A}{M\bullet}{P\bullet}
\pstInterLL[PointSymbol=square, PointName=none]{M\bullet}{I\bullet}{M\bullet}{P\bullet}
\ifnum\n=1 \bgroup \fi}
\Parabole[2]\pstGenericCurve[linestyle=magenta]{P\bullet}{1}{\NbPt}
\pstGeonode[PointSymbol=*, PosAngle=-90]{(10,3.5)}{B}
\Parabole[\textcolor{magenta}{\textbullet}]{P\bullet}{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}(-4,-4)(4,4)
\newcommand\Sommet{\PSset{\Sommet=1.4142135623730951}}
\newcommand\PosFoyer{2}
\newcommand\HypAngle{0}
\newcommand\Inc{20}
\newcommand\Ri{2}
\newcommand\F{0}
\newcommand\CoefDiv{7}
\psLineAB*{\F}{\F'}{90}
\psRotation*[\Ri\Sommet]{90}{\PointSymbol=none}{\S}{(0)\PosFoyer}{\HypAngle}{F}
\psSymO*[\F]{\PosAngle=180}{0}{F}{F'}\psCircleOA*[0]{F}
\psGeonode*[\PosAngle=-135]{\PosAngle=45}{\S}{\S'}\psRotation*[\Ri]{90}{\PointSymbol=none}{\S}{(0)\PosFoyer}{\HypAngle}{F}
\psLineAB*[\NodesepA=-3,\NodesepB=-5]{A_1}{A_2}{A_1}{O}
\psMarkAngle*[\LabelSep=1.1\LabelSep=1.1]{F}{O}{\F}{A_1}{0}
\ncLine*[\LineColor=red]{A_1}{A_2}{\PointSymbol=none}{\RightAngleSize=15}{A_1}{S}{0}
\psset{\PointName=none}
\whiledo{\value{n}<8}{%}
\psSet*[\RadiusA=\RadiusB=]{\Ri}{\Ri}{\PointSymbol=none}{\F'}{\F'\arabic{n}\arabic{n}}{\arabic{n}}{\arabic{n}}{\arabic{n}}{\arabic{n}}{\arabic{n}}{\arabic{n}}{\arabic{n}}
\Psset*[\Inc]{\Inc}{\Inc}{\value{Inc}}{\value{Inc}}}{% fin de whiledo
\psSet*[\LineColor=blue]{\GenericCurve}[\GenericCurve⁎]{S}{M}{2}{7}
\psSet*[\GenericCurve]{\GenericCurve⁎}{S'}{M'}{2}{7}
\psSet*[\GenericCurve]{\GenericCurve⁎}{S'}{M'}{2}{7}
\psSet*[\GenericCurve]{\GenericCurve⁎}{S'}{M'}{2}{7}
\end{pspicture}

B.3. Cycloid

The wheel rolls from $M$ to $A$. The circle points are on a cycloid.
B. Some locus points

\begin{pspicture}[-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}{% \psset{linewidth=1.2\pslinewidth}
  \pstHomO[HomCoef=nA/\NbPt, PointSymbol=none, PointName=none]{O}{B}[nA]
  \pstProjection[PointSymbol=none, PointName=none, CurvAbsNeg=true]{O}{P}[nA]
  \psset{linecolor=magenta, linewidth=1.5\pslinewidth}
  \pstCircleOA{O}{M}[nA]
  \ifnum\nA=2 \psset{linecolor=blue, linewidth=1.5\pslinewidth}
  \pstGenericCurve[GenCurvFirst=M]{M}{1}{6}
  \pstGenericCurve[GenCurvLast=A]{M}{6}{\NbPt}
  \psset{linecolor=blue, linewidth=1.5\pslinewidth}
  \ncline{O}{A}[nA]
  \ncline{P}{M}[nA]
  \ncline{O}{P}[nA]
% fin du multido
\psset{linecolor=blue, linewidth=1.5\pslinewidth}
\pstGenericCurve[GenCurvFirst=M]{M}{1}{6} \psset{linecolor=blue, linewidth=1.5\pslinewidth}
\pstGenericCurve[GenCurvLast=A]{M}{6}{\NbPt}
\end{pspicture}
B. Some locus points

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.
Some locus points

\begin{pspicture}(-4.5,-4)(4.5,4.5)
\HypoCyclo[4][100]{% 
defR{#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=*,PointName={
default, none}, PosAngle=0](){O}{R}
\psset{PointSymbol=none,PointName=none}
\pscircleOA{O}{P}
\psHomO[HomCoef=\petitR{space}{R}{space}{div}]{P}{O}[M]
\multido(n=1+1){\NbPt}{% \pstRotation[RotAngle=\Anglen]{O}{M}[N]
\rput(M){\pstGeonode(\petitR;9){Q}}
\pstRotation[RotAngle=\Anglen]{M}[N][Q]
\pstRotation[RotAngle=n{space}{-360 \NbPt{space}{1 add div mul}{R}{space}{petitR}{space}{div mul}, PointSymbol
to,*}, PointName=none]{M}[N][N]
\ifnumn=#1
\pstcircleOA{M}[N][N]\ncircle(M)[N][N]%{psset}{linecolor=red, linewidth=2\pslinewidth}
\pstarcOAB{M}[N][N]\nstarcOAB{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: a hyperbola;
- outside the circle: an ellipse.

(figure of O. Reboux).
C. Lines and circles envelope

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,PointSymbol=x, nodesep=0, linecolor=magenta}
\pstGeonode[PointName=none]{O}(2,0){O'}
\pstCircleOA[linecolor=black]{O}{O'}
\multido\n=5+5\do {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)}
\end{pspicture}
### F. List of all optional arguments for \texttt{pst-eucl}

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

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CodeFigAarc</td>
<td>ordinary</td>
<td>true</td>
</tr>
<tr>
<td>CodeFigBarc</td>
<td>ordinary</td>
<td>true</td>
</tr>
<tr>
<td>Radius</td>
<td>ordinary</td>
<td>none</td>
</tr>
<tr>
<td>RadiusA</td>
<td>ordinary</td>
<td>undef</td>
</tr>
<tr>
<td>RadiusB</td>
<td>ordinary</td>
<td>undef</td>
</tr>
<tr>
<td>Diameter</td>
<td>ordinary</td>
<td>none</td>
</tr>
<tr>
<td>DiameterA</td>
<td>ordinary</td>
<td>undef</td>
</tr>
<tr>
<td>DiameterB</td>
<td>ordinary</td>
<td>undef</td>
</tr>
<tr>
<td>DistCoef</td>
<td>ordinary</td>
<td>none</td>
</tr>
<tr>
<td>AngleCoef</td>
<td>ordinary</td>
<td>none</td>
</tr>
<tr>
<td>CurvAbsNeg</td>
<td>ordinary</td>
<td>false</td>
</tr>
<tr>
<td>GenCurvFirst</td>
<td>ordinary</td>
<td>none</td>
</tr>
<tr>
<td>GenCurvLast</td>
<td>ordinary</td>
<td>none</td>
</tr>
<tr>
<td>GenCurvInc</td>
<td>ordinary</td>
<td>1</td>
</tr>
<tr>
<td>AngleValue</td>
<td>boolean</td>
<td>false</td>
</tr>
<tr>
<td>AngleArc</td>
<td>boolean</td>
<td>true</td>
</tr>
<tr>
<td>ShowWedge</td>
<td>boolean</td>
<td>true</td>
</tr>
<tr>
<td>ArcColor</td>
<td>ordinary</td>
<td>[none]</td>
</tr>
<tr>
<td>ArcLinestyle</td>
<td>ordinary</td>
<td>[none]</td>
</tr>
<tr>
<td>ArcLinewidth</td>
<td>ordinary</td>
<td>[none]</td>
</tr>
<tr>
<td>WedgeColor</td>
<td>ordinary</td>
<td>[none]</td>
</tr>
<tr>
<td>WedgeFillstyle</td>
<td>ordinary</td>
<td>[none]</td>
</tr>
<tr>
<td>WedgeOpacity</td>
<td>ordinary</td>
<td>[none]</td>
</tr>
</tbody>
</table>

References


Index

Symbols
*, 5, 11
+ , 5
...A, 82
...B, 82

A
angleA, 22, 32, 37
angleB, 22, 32, 37
AngleCoef, 79
arrows, 8, 11
asterisk, 5

C
CodeFig, 74, 78–82, 84
CodeFigA, 84
CodeFigAarc, 84
CodeFigB, 84
CodeFigBarc, 84
CodeFigColor, 78, 80–82
CodeFigStyle, 78, 80–82
CurvAbsNeg, 25, 26
CurveType, 30, 78, 90
CurveType=none, 6

D
default, 5
Diameter, 18, 22, 24, 84
DiameterA, 27, 28, 85
DiameterB, 27, 28, 85
diamond, 5
diamond*, 5
dimen, 5
DistCoef, 17, 20–22, 79
dotangle, 6
dotscale, 6
DrawCirABC, 81

F
false, 78
fillcolor, 30
fillstyle, 30\fpeval, 6, 14, 25

G
GenCurvFirst, 31
GenCurvInc, 31
GenCurvLast, 31
german, 11

H
HomCoef, 80

K
Keyvalue
- *, 11
- german, 11
- MarkArrow, 8
- MarkArroww, 8
- MarkArrowww, 8
- MarkCross, 8
- MarkCross, 8
- MarkHash, 7, 8
- MarkHashh, 7
- MarkHashhh, 7
- none, 5
- pstslash, 7
- pstslashh, 7
- pstslashhh, 7
- suisseromand, 11
- swissromand, 11

Keyword
- *, 5
- + , 5
- ...A, 82
- ...B, 82
- angleA, 22, 32, 37
- angleB, 22, 32, 37
- AngleCoef, 79
- arrows, 8, 11
- asterisk, 5
- CodeFig, 74, 78–82, 84
- CodeFigA, 84
- CodeFigAarc, 84
- CodeFigB, 84
- CodeFigBarc, 84
- CodeFigColor, 78, 80–82
- CodeFigStyle, 78, 80–82
- CurvAbsNeg, 25, 26
- CurveType, 30, 78, 90
- CurveType=none, 6
- Diameter, 18, 22, 24, 84
- DiameterA, 27, 28, 85
- DiameterB, 27, 28, 85
- diamond, 5
- diamond*, 5
- dimen, 5
- DistCoef, 17, 20–22, 79
- dotangle, 6
- dotscale, 6
- DrawCirABC, 81
- false, 78
- fillcolor, 30
- fillstyle, 30
- GenCurvFirst, 31
- GenCurvInc, 31
- GenCurvLast, 31
- HomCoef, 80
- label, 11
- LabelAngleOffset, 11
- LabelRefPt, 11
- LabelSep, 11
- linecolor, 8, 30
- linestyle, 8, 30
- linewidth, 8, 30
- Mark, 11
- MarkAngle, 8
- MarkAngleRadius, 11
- MarkAngleType, 11
- MarkHashLength, 8
- MarkHashSep, 8
- nodesep, 8, 13
- nodesepA, 13
- nodesepB, 13
- npos, 8
- nrot, 8
- o, 5
- offset, 8
- oplus, 5
- otimes, 5
- pentagon, 5
- pentagon*, 5
- PointName, 5–7, 9, 10, 26, 30, 78–84
- PointNameA, 9, 84
- PointNameB, 9, 84
- PointNameC, 9
- PointNameSep, 5–7, 9, 26, 78–84
- PointSymbol, 5–7, 9, 10, 26, 30, 78–84
- PointSymbolA, 9, 84
- PointSymbolB, 9, 84
- PointSymbolC, 9
- PosAngle, 5–7, 9, 26, 30, 37, 49, 65, 78–84
- PosAngleA, 9, 84, 85
- PosAngleB, 9, 84, 85
- PosAngleC, 9
- PtNameMath, 5, 7, 26, 78–84
- Radius, 18, 22, 24, 84
- radius, 84
- RadiusA, 27, 28, 85
- RadiusB, 27, 28, 85
- RightAngleDotDistance, 11
- RightAngleSize, 11
- RightAngleType, 11
- RotAngle, 24, 33, 79
- SegmentSymbol, 7, 80, 82
- SegmentSymbolA, 81
- SegmentSymbolB, 81
- SegmentSymbolC, 81
- square, 5
- square*, 5
- TransformLabel, 79
- triangle, 5
- triangle*, 5
- true, 74, 78
- x, 5

L
label, 11
LabelAngleOffset, 11
LabelRefPt, 11
LabelSep, 11
linecolor, 8, 30
linestyle, 8, 30
linewidth, 8, 30

M
Macro
- \fpeval, 6, 14, 25
- \naput, 8
- \nbput, 8
- \ncline, 7, 8
- \ncput, 8
- \pscalculate, 6, 14, 20, 25
- \psdot, 5
- \psellipticarc, 32
- \psellipticarc, 32
- \psGetAngleABC, 87
- \psGetDistanceAB, 87
- \psplot, 86
- \psplotImp, 38, 39, 50, 66
\pstAbscissa, 6, 14, 20, 25
\pstAngleABC, 79
\pstAngleAOB, 79
\pstArcnOAB, 24
\pstArcOAB, 24
\pstBisectorAOB, 15
\pstBisectBAC, 15, 82
\pstGravABC, 81
\pstCircleAB, 22, 24
\pstCircleABC, 81
\pstCircleABR, 23
\pstCircleAbsNode, 25
\pstCircleChordNode, 25
\pstCircleExternalCommonTangent, 27
\pstCircleInternalCommonTangent, 27
\pstCircleNode, 24, 25
\pstCircleOA, 22, 24
\pstCircleOrdNode, 25
\pstCircleRadicalAxis, 28
\pstCircleRotNode, 24, 25
\pstCircleTangentNode, 26
\pstCircleTangentLine, 26
\pstCurvAbsNode, 26
\pstDist, 16, 17, 19-21, 25, 32, 84
\pstDistAB, 20, 21
\pstDistABC, 22
\pstDistAdd, 16, 17, 19, 21, 25, 84
\pstDistAddCoef, 21
\pstDistAddVal, 21, 22
\pstDistCalc, 20, 21
\pstDistCoef, 20, 21
\pstDistConst, 16, 17, 19-21, 25
\pstDistDiv, 15, 21
\pstDistExpr, 20, 21
\pstDistMul, 20, 21, 84
\pstDistSub, 16, 17, 19, 21, 25, 84
\pstDistSubCoef, 21
\pstDistSubVal, 21, 22
\pstDistVal, 20, 21
\pstEllipse, 32, 37
\pstEllipseAbsNode, 33
\pstEllipseDirectrixLine, 34
\pstEllipseFocusNode, 33, 34
\pstEllipseLineInter, 34
\pstEllipseNode, 33
\pstEllipseOrdNode, 33
\pstEllipsePolarNode, 35
\pstEllipseRotNode, 33
\pstEllipseTangentNode, 36
\pstETriangleAB, 29, 30
\pstExtendAB, 17
\pstFourthHarmonicNode, 16
\pstGeneralConicCircleInter, 76
\pstGeneralConicEllipseInter, 76
\pstGeneralConicEquation, 74
\pstGeneralConicHyperbolaInter, 76
\pstGeneralConicIHyperbolaInter, 76
\pstGeneralConicInter, 76
\pstGeneralConicIParabolaInter, 76
\pstGeneralConicLineInter, 76
\pstGeneralConicParabolaInter, 76
\pstGeneralConicTangentChord, 77
\pstGeneralConicTangentLine, 77
\pstGeneralEllipse, 37-39
\pstGeneralEllipseABCD, 39
\pstGeneralEllipseAbsNode, 40
\pstGeneralEllipseCoef, 38, 74
\pstGeneralEllipseDirectrixLine, 40
\pstGeneralEllipseEquation, 74
\pstGeneralEllipseFFN, 38
\pstGeneralEllipseFle, 37-39
\pstGeneralEllipseFocusNode, 40
\pstGeneralEllipseLineInter, 41
\pstGeneralEllipseNode, 40
\pstGeneralEllipseOrdNode, 40
\pstGeneralEllipsePolarNode, 41
\pstGeneralEllipseRotNode, 40
\pstGeneralEllipseTangentNode, 41, 42
\pstGeneralHyperbola, 64-67
\pstGeneralHyperbolaABCD, 67
\pstGeneralHyperbolaAbsNode, 67
\pstGeneralHyperbolaAsymptoteLine, 68, 71
\pstGeneralHyperbolaCoef, 66, 74
\pstGeneralHyperbolaDirectrixLine, 68
\pstGeneralHyperbolaEquation, 74
\pstGeneralHyperbolaFFN, 65, 66
\pstGeneralHyperbolaFle, 65-67
\pstGeneralHyperbolaFocusNode, 68
\pstGeneralHyperbolaLineInter, 68
\pstGeneralHyperbolaNode, 67
\pstGeneralHyperbolaOrdNode, 67
\pstGeneralHyperbolaPolarNode, 69
\pstGeneralHyperbolaTangentNode, 69
\pstGeneralHyperbolaVertexNode, 68
\pstGeneralIHyperbola, 70
\pstGeneralIHyperbolaAbsNode, 71
\pstParabolaOrdNode, 42
\pstParabolaPolarNode, 43
\pstParabolaTangentNode, 44
\pstProjection, 80
\pstProportionNode, 15, 21
\pstRegularPolygonAB, 29, 30
\pstRegularPolygonOA, 29, 30
\pstRightAngle, 11
\pstRotation, 79
\pstScreenDist, 20
\pstSegmentMark, 7
\pstSquareAB, 29, 30
\pstSymO, 78
\pstTranslation, 17, 21, 79
\pstTriangle, 9
\pstTriangleAAS, 9
\pstTriangleASA, 9
\pstTriangleEC, 10
\pstTriangleGC, 10
\pstTriangleHC, 10
\pstTriangleIC, 10
\pstTriangleLC, 10
\pstTriangleNC, 10
\pstTriangleSAS, 9
\pstTriangleSSS, 9
\pstUserDist, 20
\rput, 24
\SpecialCoor, 5
Mark, 11
MarkAngle, 8
MarkAngleRadius, 11
MarkAngleType, 11
MarkArrow, 8
MarkArroww, 8
MarkArrowww, 8
MarkCross, 8
MarkCross, 8
MarkHash, 7, 8
MarkHashh, 7
MarkHashhh, 7
MarkHashLength, 8
MarkHashSep, 8
middle, 5

N
\naput, 8
\nbput, 8
\ncline, 7, 8
\ncput, 8
nodesep, 8, 13
nodesepA, 13
nodesepB, 13
none, 5
npos, 8
nrot, 8
O
o, 5
offset, 8
oplus, 5
otimes, 5
P
Package
- pst-eucl, 1
pentagon, 5
pentagon*, 5
PointName, 5–7, 9, 10, 26, 30, 78–84
PointNameA, 9, 84
PointNameB, 9, 84
PointNameC, 9
PointNameSep, 5–7, 9, 26, 78–84
PointSymbol, 5–7, 9, 10, 26, 30, 78–84
PointSymbolA, 9, 84
PointSymbolB, 9, 84
PointSymbolC, 9
PosAngle, 5–7, 9, 26, 30, 37, 49, 65, 78–84
PosAngleA, 9, 84, 85
PosAngleB, 9, 84, 85
PosAngleC, 9
\pscalculate, 6, 14, 20, 25
\psdot, 5
\psellipse, 32
\psellipticarc, 32
\psGetAngleABC, 87
\psGetDistanceABC, 87
\psplot, 86
\psplotImp, 38, 39, 50, 66
pst-eucl, 1
\pstAbscissa, 6, 14, 20, 25
\pstAngleABC, 79
\pstAngleAOB, 79
\pstArcnOAB, 24
\pstArcOAB, 24
\pstBisectorAOB, 15
\pstBisectBAC, 15, 82
\pstGeneralIHyperbolaTangentNode, 73
\pstGeneralIHyperbolaVertexNode, 71
\pstGeneralIParabola, 55
\pstGeneralIParabolaAbsNode, 55
\pstGeneralIParabolaAbsNode, 55
\pstGeneralIParabolaDirectrixLine, 55
\pstGeneralIParabolaFocusNode, 55
\pstGeneralIParabolaLineInter, 56
\pstGeneralIParabolaNode, 55
\pstGeneralIParabolaOrdNode, 55
\pstGeneralIParabolaPolarNode, 57
\pstGeneralIParabolaTangentNode, 58
\pstGeneralIParabolaTangentNode, 58
\pstGeneralParabolaABCDE, 51
\pstGeneralParabolaAbsNode, 48
\pstGeneralParabolaAbsNode, 48
\pstGeneralParabolaDirectrixLine, 51
\pstGeneralParabolaEquation, 74
\pstGeneralParabolaFL, 49–51
\pstGeneralParabolaFocusNode, 51
\pstGeneralParabolaLineInter, 52
\pstGeneralParabolaNode, 48
\pstGeneralParabolaOrdNode, 48
\pstGeneralParabolaPolarNode, 53
\pstGeneralParabolaTangentNode, 54
\pstGenericCurve, 31
\pstGeometricMean, 19
\pstGeonode, 1, 5
\pstHarmonicMean, 19
\pstHomO, 80
\pstHyperbola, 58
\pstHyperbolaAbsNode, 58, 59
\pstHyperbolaAbsNode, 58, 59
\pstHyperbolaAsymptoteLine, 59, 62
\pstHyperbolaDirectrixLine, 59
\pstHyperbolaFocusNode, 59
\pstHyperbolaLineInter, 60
\pstHyperbolaNode, 58, 59
\pstHyperbolaOrdNode, 59
\pstHyperbolaPolarNode, 60
\pstHyperbolaTangentNode, 61
\pstHyperbola, 61, 62
\pstHyperbolaAbsNode, 62
\pstHyperbolaAbsNode, 62
\pstHyperbolaAsymptoteLine, 62
\pstHyperbolaDirectrixLine, 62
\pstHyperbolaFocusNode, 62
\pstHyperbolaLineInter, 63
\pstHyperbolaNode, 62
\pstHyperbolaOrdNode, 62
\pstIHyperbolaPolarNode, 63
\pstIHyperbolaTangentNode, 64
\pstInterCC, 84, 85
\pstInterFC, 86
\pstInterFF, 85
\pstInterFL, 86
\pstInterLC, 84
\pstInversion, 18
\pstIParabola, 45
\pstIParabolaAbsNode, 45
\pstIParabolaDirectrixLine, 46
\pstIParabolaFocusNode, 46
\pstIParabolaLineInter, 46
\pstIParabolaNode, 45
\pstIParabolaOrdNode, 45
\pstIParabolaPolarNode, 46
\pstIParabolaTangentNode, 47
\pstLabelAB, 8
\pstLine, 13, 14
\pstLineAA, 14
\pstLineAB, 13
\pstLineAbsNode, 14, 15
\pstLineAS, 14, 59, 62
\pstLineCoef, 14
\pstLineOrdNode, 14, 15
\pstLocateAB, 16–19
\pstMarkAngle, 11
\pstMederatorAB, 82
\pstMiddleAB, 80
\pstMoveNode, 6, 7
\pstOIJGeonode, 7
\pstOrdinate, 6, 14, 20, 25
\pstOrtSym, 78
\pstOutBissectBAC, 15, 82
\pstParabola, 42
\pstParabolaAbsNode, 42
\pstParabolaDirectrixLine, 42, 43
\pstParabolaFocusNode, 42
\pstParabolaLineInter, 42, 43
\pstParabolaNode, 42
\pstParabolaOrdNode, 42
\pstParabolaPolarNode, 43
\pstParabolaTangentNode, 44
\pstProjection, 80
\pstProportionNode, 15, 21
\pstRegularPolygonAB, 29, 30
\pstRegularPolygonOA, 29, 30
\pstRightAngle, 11
\pstRotation, 79
\pstScreenDist, 20
\pstSegmentMark, 7
pstslash, 7
pstslashh, 7
\pstSquareAB, 29, 30
\pstSymO, 78
\pstTranslation, 17, 21, 79
\pstTriangle, 9
\pstTriangleAAS, 9
\pstTriangleASA, 9
\pstTriangleEC, 10
\pstTriangleGC, 10
\pstTriangleHC, 10
\pstTriangleIC, 10
\pstTriangleLC, 10
\pstTriangleNC, 10
\pstTriangleOC, 10
\pstTriangleSAS, 9
\pstTriangleSSS, 9
\pstUserDist, 20
PtNameMath, 5, 7, 26, 78–84

\textbf{R}
\textit{Radius}, 18, 22, 24, 84
\textit{radius}, 84
\textit{RadiusA}, 27, 28, 85
\textit{RadiusB}, 27, 28, 85
\textit{RightAngleDotDistance}, 11
\textit{RightAngleSize}, 11
\textit{RightAngleType}, 11
\textit{RotAngle}, 24, 33, 79
\texttt{rput}, 24

\textbf{S}
\textit{SegmentSymbol}, 7, 80, 82
\textit{SegmentSymbolA}, 81
\textit{SegmentSymbolB}, 81
\textit{SegmentSymbolC}, 81
\texttt{\SpecialCoor}, 5
\textit{square}, 5
\textit{square*}, 5
\textit{suisseromand}, 11
\textit{swissromand}, 11

\textbf{T}
\textit{TransformLabel}, 79