

# pst-euclide

A PSTricks package for drawing geometric pictures; v.1.74

Dominique Rodriguez

Herbert Voß

July 15, 2020

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

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

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

Thanks to: Manuel Luque; Thomas Söll; Liao Xiongfei.

**Contents**

<b>I. The package</b>	<b>5</b>
<b>1. Special specifications</b>	<b>5</b>
1.1. PSTricks Options . . . . .	5
1.2. Conventions . . . . .	5
<b>2. Basic Objects</b>	<b>5</b>
2.1. Points . . . . .	5
2.2. Segment mark . . . . .	7
2.3. Segment labels . . . . .	8
2.4. Triangles . . . . .	9
2.5. Angles . . . . .	12
2.6. Lines, half-lines and segments . . . . .	14
2.7. Distance . . . . .	21
2.8. Circles . . . . .	23
2.9. Circle arcs . . . . .	25
2.10 Circle nodes . . . . .	25
2.11 Circle tangent . . . . .	27
2.12 Circle radical axis . . . . .	29
2.13 Regular polygons . . . . .	30
2.14 Generic curve . . . . .	32
<b>3. Conics</b>	<b>33</b>
3.1. Standard Ellipse . . . . .	33
3.2. General Ellipse . . . . .	38
3.3. Standard Parabola . . . . .	43
3.4. Standard Inversion Parabola . . . . .	46
3.5. General Parabola . . . . .	49
3.6. General Inversion Parabola . . . . .	55
3.7. Standard Hyperbola . . . . .	59
3.8. Standard Inversion Hyperbola . . . . .	62
3.9. General Hyperbola . . . . .	65
3.10 General Inversion Hyperbola . . . . .	71
<b>4. Geometric Transformations</b>	<b>75</b>
4.1. Central symmetry . . . . .	75
4.2. Orthogonal (or axial) symmetry . . . . .	75
4.3. Rotation . . . . .	76
4.4. Translation . . . . .	76
4.5. Homothetic . . . . .	77
4.6. Orthogonal projection . . . . .	77
<b>5. Special object</b>	<b>78</b>
5.1. Midpoint . . . . .	78
5.2. Triangle center of gravity . . . . .	78
5.3. Centre of the circumcircle of a triangle . . . . .	78
5.4. Perpendicular bisector of a segment . . . . .	79

---

5.5. Bisectors of angles . . . . .	79
<b>6. Intersections</b>	<b>80</b>
6.1. Line-Line . . . . .	80
6.2. Circle-Line . . . . .	81
6.3. Circle-Circle . . . . .	81
6.4. Function-function . . . . .	82
6.5. Function-line . . . . .	83
6.6. Function-Circle . . . . .	83
<b>7. Helper Macros</b>	<b>84</b>

---

<b>II. Examples gallery</b>	<b>86</b>
<b>A. Basic geometry</b>	<b>86</b>
A.1. Drawing of the bissector . . . . .	86
A.2. Transformation de polygones et courbes . . . . .	87
A.3. Triangle lines . . . . .	88
A.4. Euler circle . . . . .	89
A.5. Orthocenter and hyperbola . . . . .	90
A.6. 17 sides regular polygon . . . . .	91
A.7. Circles & tangents . . . . .	93
A.8. Fermat's point . . . . .	94
A.9. Escribed and inscribed circles of a triangle . . . . .	95
<b>B. Some locus points</b>	<b>97</b>
B.1. Parabola . . . . .	97
B.2. Hyperbola . . . . .	98
B.3. Cycloid . . . . .	98
B.4. Hypocycloids (Astroid and Deltoid) . . . . .	100
<b>C. Lines and circles envelope</b>	<b>102</b>
C.1. Conics . . . . .	102
C.2. Cardioid . . . . .	103
<b>D. Homotethy and fractals</b>	<b>104</b>
<b>E. hyperbolic geometry: a triangle and its altitudes</b>	<b>105</b>
<b>F. List of all optional arguments for pst-eucl</b>	<b>106</b>
<b>References</b>	<b>107</b>

# Part I.

## The package

### 1. Special specifications

#### 1.1. PSTricks Options

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

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

#### 1.2. Conventions

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

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

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

### 2. Basic Objects

#### 2.1. Points

```
\pstGeonode [Options] (x1,y1){A1}(x2,y2){A1}... (x,y){An}
```

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:

- |   |   |  |
|---|---|--|
| <ul style="list-style-type: none"> <li>• <code>*</code>: ●</li> <li>• <code>o</code>: ○</li> <li>• <code>+</code>: +</li> <li>• <code>x</code>: ·</li> <li>• <code>asterisk</code>: ✱</li> <li>• <code>oplus</code>: ⊕</li> </ul> | <ul style="list-style-type: none"> <li>• <code>otimes</code>: ⊗</li> <li>• <code>triangle</code>: △</li> <li>• <code>triangle*</code>: ▲</li> <li>• <code>square</code>: □</li> <li>• <code>square*</code>: ■</li> <li>• <code>diamond</code>: ◇</li> </ul> | <ul style="list-style-type: none"> <li>• <code>diamond*</code>: ◆</li> <li>• <code>pentagon</code>: ⬠</li> <li>• <code>pentagon*</code>: ⬡</li> <li>• <code> </code>:  </li> </ul> |
|---|---|--|

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

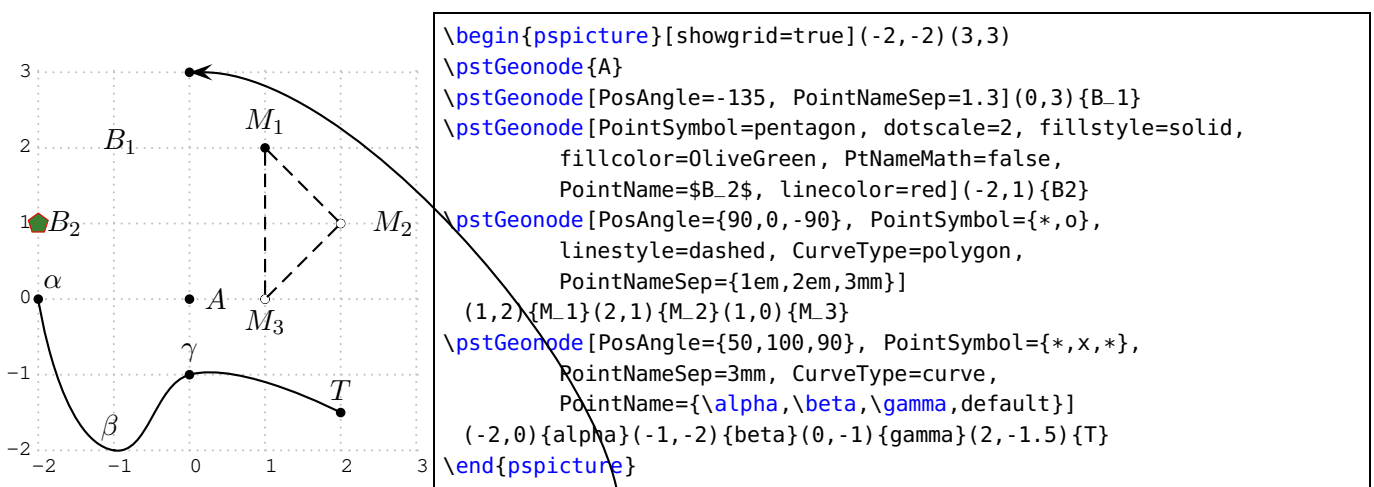
- their scale with `dotscale`, the value of whom is either two numbers defining the horizontal and vertical scale factor, or one single value being the same for both,
- their angle with parameter `dotangle`.

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.

```

\pstAbscissa{A}
\pstOrdinate{A}

```

Note that the value of abscissa or ordinate are transformed to the User coordinate, and then put into the stack of PostScript, so they can be used to do some compound arithmetic without concerned the xunit and yunit in the PSTricks `SpecialCoord` function. You need the other third package to do float arithmetic operation, like `\pscalculate`<sup>1</sup> to generate the numerical values, or the expandable command `\fpeval`<sup>2</sup> to get a purely numerical result.

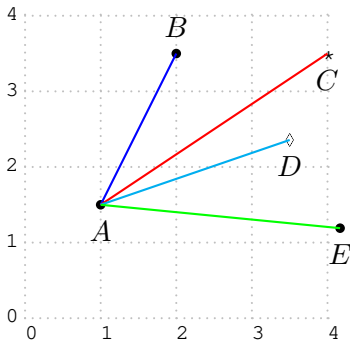
The macro `\pstMoveNode` use them to move node *A* by abscissa increment *dx* and ordinate increment *dy* to get the target node *B*.

<sup>1</sup> 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.

<sup>2</sup> Provided by package `xfp`, it can truncate the fraction part digits using the `trunc` function perfectly, e.g. `\fpeval{trunc(18/7,3)}`.

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

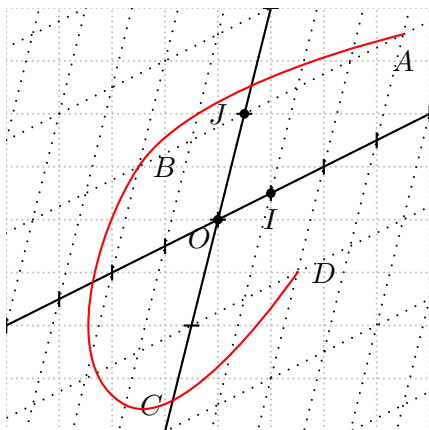
for example:



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

`\pst0IJGeonode` creates a list of points in the landmark ( $O; I; J$ ). Possible parameters are `PointName`, `PointNameSep`, `PosAngle`, `PointSymbol`, and `PtNameMath`.

```
\pst0IJGeonode [Options] (x1,y1){A1}{O}{I}{J} (x2,y2){A2}... (x,y){An}
```



```
\psset{unit=.7}
\begin{pspicture*}[showgrid=true](-4,-4)(4,4)
\pstGeonode[PosAngle={-135,-90,180}]{O}(1,0.5){I}(0.5,2){J}
\pstLineAB[nodesep=10]{O}{I}
\pstLineAB[nodesep=10]{O}{J}
\multisps(-5,-2.5)(1,0.5){11}{\psline(0,-.15)(0,.15)}
\multisps(-2,-8)(0.5,2){9}{\psline(-.15,0)(.15,0)}
\psset{linestyle=dotted}%
\multisps(-5,-2.5)(1,0.5){11}{\psline(-10,-40)(10,40)}
\multisps(-2,-8)(0.5,2){9}{\psline(-10,-5)(10,5)}
\psset{PointSymbol=x,linestyle=solid}
\pst0IJGeonode[PosAngle={-90,0},CurveType=curve,
linecolor=red](3,1){A}{O}{I}{J}(-2,1){B}(-1,-1.5){C}(2,-1){D}
\end{pspicture*}
```

## 2.2. Segment mark

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

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

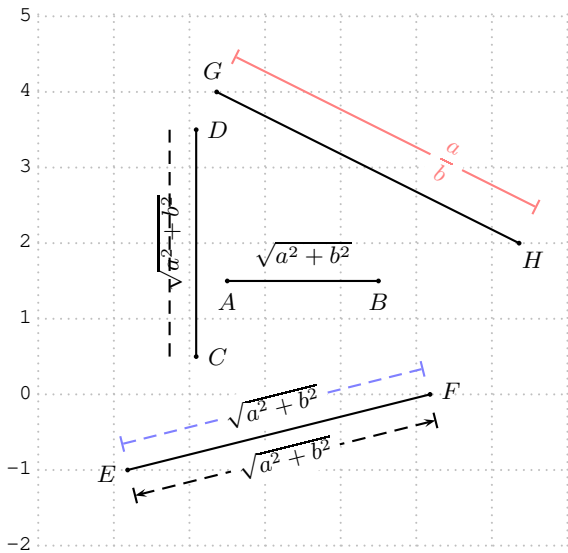
The symbol drawn on the segment is given by the parameter `SegmentSymbol`. Its value can be any valid command which can be used in math mode. Its default value is `MarkHashh`, which produced two slashes on the segment. The segment is drawn.

Several commands are predefined for marking the segment:

- `pstslash`
- `pstslashhh`
- `MarkHashh`
- `pstslaslh`
- `MarkHash`
- `MarkHashhh`







```

\begin{pspicture}[showgrid=true](-2,-2)(5,5)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90](0.5,1.5){A}
\pstGeonode[PosAngle=-90](2.5,1.5){B}\pstLineAB{A}{B}
\pstLabelAB{A}{B}{\sqrt{a^2+b^2}}
\pstGeonode[PosAngle=0](0,0.5){C}
\pstGeonode[PosAngle=0](0,3.5){D}\pstLineAB{C}{D}
\pstLabelAB[linestyle=dashed]{C}{D}{\sqrt{a^2+b^2}}
\pstGeonode[PosAngle=190](-1,-1){E}
\pstGeonode[PosAngle=10](3,0){F}\pstLineAB{E}{F}
\pstLabelAB*[linestyle=dashed,arrows=|-,offset=10pt,linicolor=blue!50]{E}{F}{\sqrt{a^2+b^2}}
\pstLabelAB*[linestyle=dashed,arrows=|<->,offset=10pt,nrot=D]{F}{E}{\sqrt{a^2+b^2}}
\pstGeonode[PosAngle=100](0,4){G}
\pstGeonode[PosAngle=-50](4,2){H}\pstLineAB{G}{H}
\pstLabelAB*[linestyle=solid,linicolor=red!50,arrows=|-,offset=15pt,nrot=U,npos=0.7]{G}{H}{\dfrac{a}{b}}
\end{pspicture}

```

## 2.4. Triangles

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

```
\pstTriangle [Options] (x1, y1){A}(x2, y2){B}(x3, y3){C}
```

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

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

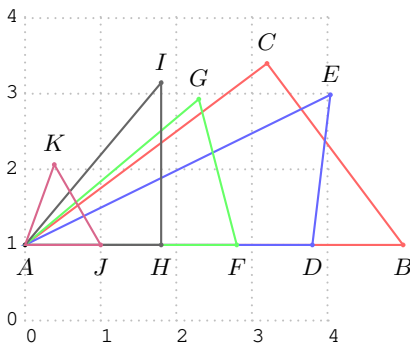
The macros `\pstTriangleSSS`, `\pstTriangleSAS`, `\pstTriangleAAS` and `\pstTriangleASA` are used to draw the triangle according the specified sides or angles.

```

\pstTriangleSSS [Options] (pos){A}(a,b,c){B}{C}
\pstTriangleSAS [Options] (pos){A}(b,∠A,c){B}{C}
\pstTriangleAAS [Options] (pos){A}(∠C,∠A,c){B}{C}
\pstTriangleASA [Options] (pos){A}(∠A,c,∠B){B}{C}

```

- Macro `\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  $C(x_3, y_3)$ .
- Macro `\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  $C(x_3, y_3)$ .
- Macro `\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  $C(x_3, y_3)$ .
- Macro `\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  $C(x_3, y_3)$ .

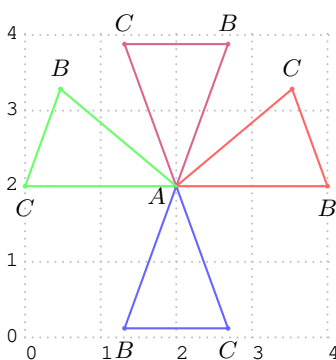


```

\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90](0,1){A}
\pstTriangleSSS[linecolor=red!60,PosAngle={-90,90}]{A}(3,4,5){B}{C}
\pstTriangleSSS[linecolor=blue!60,PosAngle={-90,90}]{A}
(2,4.5,3.8){D}{E}
\pstTriangleSAS[linecolor=green!60,PosAngle={-90,90}]{A}
(3,4,2.8){F}{G}
\pstTriangleAAS[linecolor=black!60,PosAngle={-90,90}]{A}
(4,5,1.8){H}{I}
\pstTriangleASA[linecolor=purple!60,PosAngle={-90,90}]{A}
(7,1.0,60){J}{K}
\end{pspicture}

```

The optional parameter `pos` setup the position of the first node  $A$ , it should be 'L' for left, 'R' for right, 'U' for up and 'D' for down. If you don't input this parameter, the default value is 'L'. The following example explains how to draw an isoceses triangle with the given isoceses sides and the vertex angle.



```

\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=205](2,2){A}
\pstTriangleSAS[linecolor=red!60,PosAngle={-90,90}]{A}(2,4,2){B}{C}
\pstTriangleSAS[linecolor=blue!60,PosAngle={-90,-90}]{A}
(2,4,2){B}{C}
\pstTriangleSAS[linecolor=purple!60,PosAngle={90,90}]{A}
(2,4,2){B}{C}
\pstTriangleSAS[linecolor=green!60,PosAngle={90,-90}]{A}
(2,4,2){B}{C}
\end{pspicture}

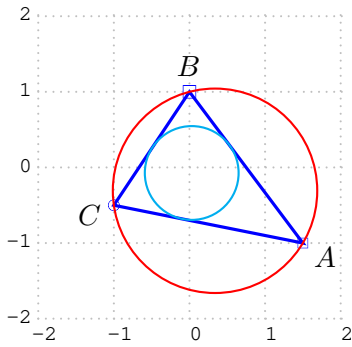
```

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

```

\pstTriangleIC [Options] {A}{B}{C} [I] [H]
\pstTriangleOC [Options] {A}{B}{C} [O]

```



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

The center of the inner circle is called  $IC_0$  as default and the outer circle  $OC_0$  as default, but you can change the node names by the optional parameters `[I]`, `[H]` and `[O]`. 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.

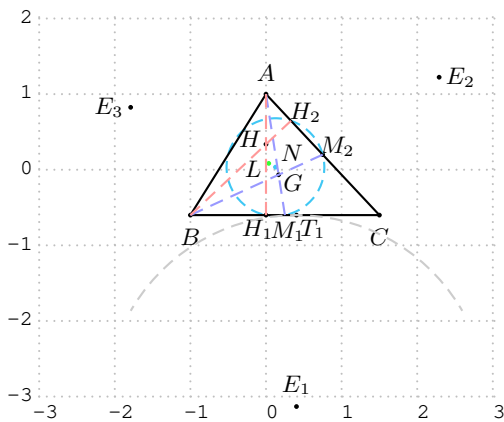
The inner center  $I$ , node  $H$  and outer circle center  $O$  are not printed out as default, but you can setup `PointSymbol` and `PointName` to display them. For example:

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

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

```
\pstTriangleGC [Options] {A}{B}{C}{G} [M1] [M2]
\pstTriangleHC [Options] {A}{B}{C}{H} [H1] [H2]
\pstTriangleEC [Options] {A}{B}{C}{E} [T1]
\pstTriangleNC [Options] {A}{B}{C}{N} [M1] [M2] [M3]
\pstTriangleLC [Options] {A}{B}{C}{L} [S1] [S2] [S3]
```

You can use the options of node like as `PointName=...`, `PosAngle=...`, `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 `PointName={...}`, `PosAngle={...}`, `PointSymbol={...}`. For example,



```

\begin{pspicture}[showgrid=true](-3,-3)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=90](0,1){A}
\pstGeonode[PosAngle=-90](-1,-0.6){B}
\pstGeonode[PosAngle=-90](1.5,-0.6){C}
\pstTriangleGC[PointSymbol={*,none,*},PosAngle={-30,-80,30},
PointNameSep=0.22cm]{A}{B}{C}{G}[M_1][M_2]
\pstTriangleHC[PointSymbol={*,*,none},PosAngle={160,-120,30},
PointNameSep=0.22cm]{A}{B}{C}{H}[H_1][H_2]
\pstTriangleEC[PointSymbol={*,*},PosAngle={90,-40}]{A}{B}{C}{E_1}[T_1]
\pstTriangleEC[PointSymbol=*,PosAngle=0]{B}{C}{A}{E_2}
\pstTriangleEC[PointSymbol=*,PosAngle=180]{C}{A}{B}{E_3}
\pstTriangleNC[PointSymbol=*,PosAngle=40,linestyle=dashed,
linecolor=cyan!60]{A}{B}{C}{N}
\pstTriangleLC[PointSymbol=*,PosAngle=200,linecolor=green!80,
PointNameSep=0.22cm]{A}{B}{C}{L}
\pstLineAB{A}{B}\pstLineAB{B}{C}\pstLineAB{C}{A}
\pstCircleOA[linestyle=dashed,linecolor=gray!40]{E_1}{T_1}[30][150]
\pstLineAB[linestyle=dashed,linecolor=blue!40]{A}{M_1}
\pstLineAB[linestyle=dashed,linecolor=blue!40]{B}{M_2}
\pstLineAB[linestyle=dashed,linecolor=red!40]{A}{H_1}
\pstLineAB[linestyle=dashed,linecolor=red!40]{B}{H_2}
\end{pspicture}

```

## 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`, `RightAngleSize`, and `RightAngleDotDistance`.

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

- `*` : standard symbol ;
- `german` : german symbol (given by U. Dirr) ;
- `suissromand` : swiss romand symbol (given P. Schneulin).

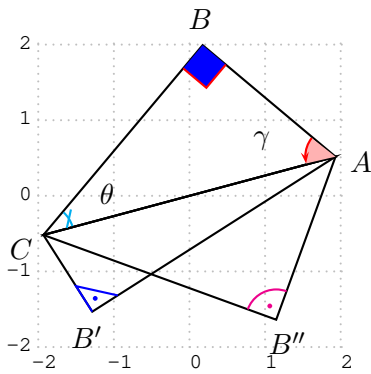
The optional argument `RightAngleSize` defines the size of the symbol (by default 0.28 unit).

For a right angle style `german` or `suissromand` the distance of the dot is preset to 0.5 (`german`) or 0.45 (`suissromand`), 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}
```

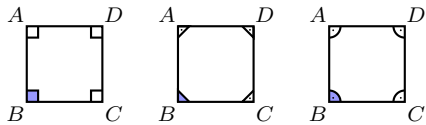
Valid optional arguments are `MarkAngleRadius`, `LabelAngleOffset`, `MarkAngleType` and `Mark`. The `Label` can be any valid  $\text{T}_{\text{E}}\text{X}$  box, it is put at `LabelSep` (by default 1 unit) of the node in the direction of the bisector of the angle modified by `LabelAngleOffset`(by default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



```

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

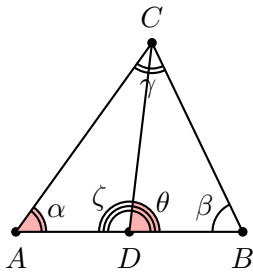
```



```

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

```



```

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

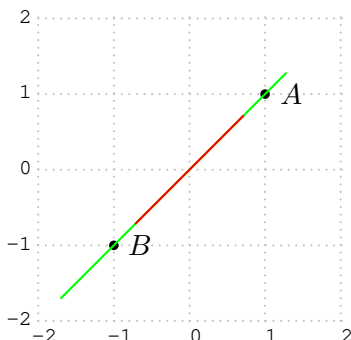
```

## 2.6. Lines, half-lines and segments

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

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

In order to control its length<sup>3</sup>, the two parameters `nodesepA` et `nodesepB` specify the abscissa of the extremity of the drawing part of the line. A negative abscissa specify an outside point, while a positive abscissa specify an internal point. If these parameters have to be equal, `nodesep` can be used instead. The default value of these parameters is equal to 0.



```

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

```

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

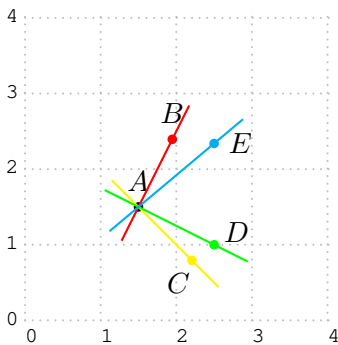
<sup>3</sup> which is the comble for a line!

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

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

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

Here are some examples:

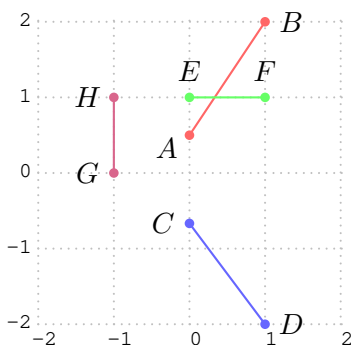


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\pstGeonode[PosAngle=90](1.5,1.5){A}
% draw a line with angle atan(2/1), about 63.43 degree.
\pstLineAA[linecolor=red,nodesep=-0.5,PosAngle=90]{A}{2 1 atan}{B}
}
\pstLineAA[linecolor=yellow,nodesep=-0.5,PosAngle=-120]{A}{-45}{C}
}
\pstLineAS[linecolor=green,nodesep=-0.5,PosAngle=30]{A}{-0.5}{D}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

The macros `\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.

```
\pstLineCoef [Options] {a,b,c}{A}{B}
```

Here are some examples:

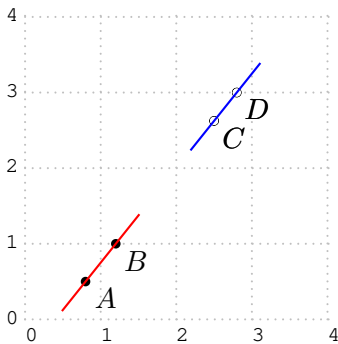


```
\begin{pspicture}[showgrid=true](-2,-2)(2,2)
\pstLineCoef[linecolor=red!60, PosAngle={210,0}]{3,-2,1}{A}{B}
\pstLineCoef[linecolor=blue!60, PosAngle={180,0}]{4,3,2}{C}{D}
\pstLineCoef[linecolor=green!60, PosAngle={90,90}]{0,3,-3}{E}{F}
\pstLineCoef[linecolor=purple!60, PosAngle={180,180}]{4,0,4}{G}{H}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

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.

```
\pstLineAbsNode [Options] {A}{B}{x_1}{C}
\pstLineOrdNode [Options] {A}{B}{y_1}{C}
```

For example,



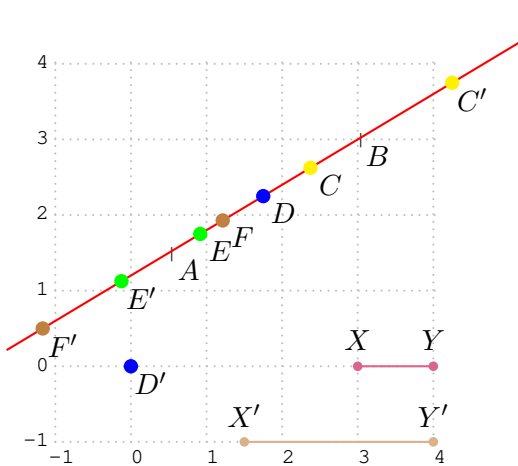
```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\pstGeonode[PosAngle=-40](0.8,0.5){A}
\pstGeonode[PosAngle=-40](1.2,1.0){B}
\pstLineAB[linecolor=red,nodesep=-0.5]{A}{B}
\pstLineAbsNode[PosAngle=-40,PointSymbol=o]{A}{B}{2.5}{C}
\pstLineOrdNode[PosAngle=-40,PointSymbol=o]{A}{B}{3.0}{D}
\pstLineAB[linecolor=blue,nodesep=-0.5]{C}{D}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

The macro `\pstProportionNode` 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

$$\begin{cases} x_C = \frac{x_A + \lambda x_B}{1 + \lambda} \\ y_C = \frac{y_A + \lambda y_B}{1 + \lambda} \end{cases} \quad \text{and} \quad \begin{cases} x_{C'} = \frac{x_A - \lambda x_B}{1 - \lambda} \\ y_{C'} = \frac{y_A - \lambda y_B}{1 - \lambda} \end{cases}$$

```
\pstProportionNode [Options] {A}{B}{\lambda}{C}{C'}
```

You can use `\pstDistDiv` to get the ratio of two segments to  $\lambda$ , we will introduce `\pstDistDiv` later.

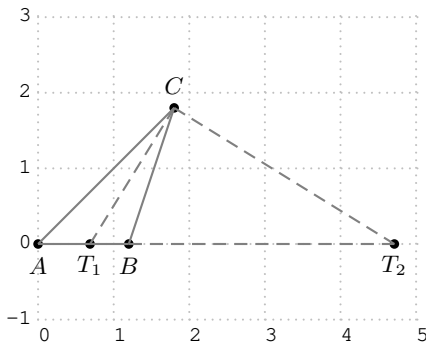


```
\begin{pspicture}[showgrid=true](-1,-1)(4,4)
\pstGeonode[PosAngle=-40,PointSymbol=|](0.5,1.5){A}
\pstGeonode[PosAngle=-40,PointSymbol=|](3.0,3.0){B}
\pstGeonode[PosAngle=90,linecolor=purple!60,CurveType=polyline]
](3,0){X}(4,0){Y}
\pstGeonode[PosAngle=90,linecolor=brown!60,CurveType=polyline]
](1.5,-1){X'}(4,-1){Y'}
\pstLineAB[linecolor=red,nodesep=-2.5]{A}{B}
\psset{PosAngle=-40,PointSymbol=*,dotscale=1.5}
\pstProportionNode[linecolor=yellow]{A}{B}{3.0}{C}{C'}
\pstProportionNode[linecolor=blue]{A}{B}{1.0}{D}{D'}
\pstProportionNode[linecolor=green]{A}{B}{0.2}{E}{E'}
\pstProportionNode[linecolor=brown]{A}{B}{\pstDistDiv X}{Y}{X'}{Y'}
}{F}{F'}
\end{pspicture}
```

One application of `\pstProportionNode` is used to find the bisector and out bisector of a given angle. So we define the macro `\pstBisectorAOB` to do this work, it is more friendly than the macros `\pstBisectBAC` and `\pstOutBisectBAC`, as it put the new node  $T_1$  and  $T_2$  on line  $AB$ , not arc  $AB$ .

```
\pstBisectorAOB [Options] {A}{O}{B}{T_1}{T_2}
```





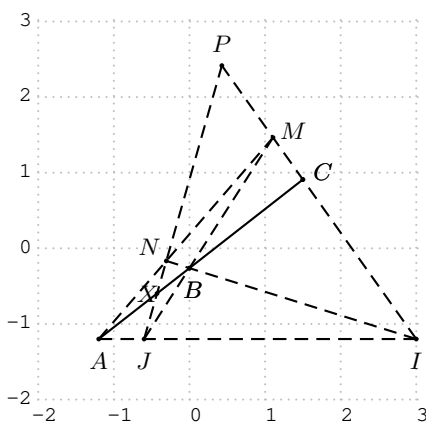
```
\begin{pspicture}[showgrid=true](0,-1)(5,3)
\psset{unit=0.6cm}\footnotesize\psset{PointSymbol=*}
\pstGeonode[PosAngle=-90](0,0){A}
\pstGeonode[PosAngle=90](3,3){C}
\pstGeonode[PosAngle=-90](2,0){B}
\pstBisectorAOB[PosAngle={-90,-90}]{A}{C}{B}{T_1}{T_2}
\pstLineAB[linecolor=black!50]{A}{C}
\pstLineAB[linecolor=black!50]{B}{C}
\pstLineAB[linecolor=black!50]{A}{B}
\pstLineAB[linestyle=dashed,linecolor=gray]{C}{T_1}
\pstLineAB[linestyle=dashed,linecolor=gray]{C}{T_2}
\pstLineAB[linestyle=dashed,linecolor=gray]{B}{T_2}
\end{pspicture}
```

The four collinear points  $A, B, C, D$  are called Harmonic Conjugation Points if their cross ratio is  $-1$ , that is

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

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

```
\pstFourthHarmonicNode [Options] {A}{B}{C}{X}
```



```
\begin{pspicture}[showgrid=true](-2,-2)(3,3)\footnotesize
\psset{unit=0.6cm}\psset{dotscale=0.5}\psset{PointSymbol=*}
\pstGeonode[PosAngle=-90](-2,-2){A}(5,-2){I}(-1,-2){J}
\pstLineAA[linestyle=none,PointName=none,PointSymbol=none]{A}
  {38}{A'}
\pstLineAbsNode[PosAngle=-80]{A}{A'}{0}{B}
\pstLineAbsNode[PosAngle=20]{A}{A'}{2.5}{C}
\pstFourthHarmonicNode[PosAngle=180,PointNameSep=0.2]{A}{B}{C}{X}
% check if A,N,M are also collinear.
\pstInterLL[PosAngle=90]{J}{X}{I}{C}{P}
\pstLineAB[linestyle=dashed]{J}{P}
\pstLineAB[linestyle=dashed]{I}{P}
\pstInterLL[PosAngle=20]{J}{B}{I}{P}{M}
\pstInterLL[PosAngle=140]{I}{B}{J}{P}{N}
\pstLineAB[linestyle=dashed]{J}{M}
\pstLineAB[linestyle=dashed]{I}{N}
\pstLineAB[linestyle=dashed]{A}{M}
\pstLineAB[linestyle=dashed]{A}{I}
\pstLineAB{A}{C}
\end{pspicture}
```

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 `\pstLocateAB` to do this, it can seek the node  $C$  from  $A$  to  $B$  with the specified length  $L$ , which can be got from `\pstDist`, `\pstDistConst`, `\pstDistAdd`, `\pstDistSub`, etc.

```
\pstLocateAB [Options] {A}{B}{L}{C}
```

Note that seek from  $B$  will get the node  $C$  in the reverse order, for example,



that is, the point  $C'$  is satisfied the inversion transform equation

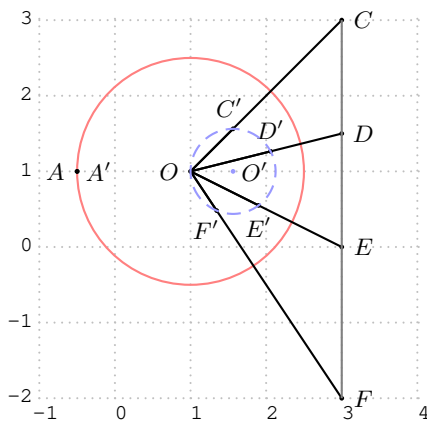
$$|OC| \times |OC'| = R^2$$

you can use the macro `\pstInversion` to do this work. In fact, we use the macro `\pstLocateAB` to implement this macro by passing the value  $\frac{R^2}{|OC|}$  to parameter length.

```
\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 `Radius` and `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{pspicture}[showgrid=true](-1,-2)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.5}
\pstGeonode[PosAngle=180](1,1){O}
\pstCircleOA[linecolor=red!50,Radius=\pstDistVal{\ra}]{O}{}
\pstCircleRotNode[PosAngle=180,RotAngle=180,Radius=\pstDistVal{\ra}]{O}{}{A}
\pstInversion[PosAngle=0,Radius=\pstDistVal{\ra}]{O}{}{A}{A'}
\pstGeonode[PosAngle=0](3,3){C}
\pstInversion[PosAngle=100,Radius=\pstDistVal{\ra}]{O}{}{C}{C'}
\pstLineAB{O}{C}\pstLineAB{O}{C'}
\pstGeonode[PosAngle=0](3,1.5){D}
\pstInversion[PosAngle=90,Radius=\pstDistVal{\ra}]{O}{}{D}{D'}
\pstLineAB{O}{D}\pstLineAB{O}{D'}
\pstGeonode[PosAngle=0](3,0){E}
\pstInversion[PosAngle=-90,Radius=\pstDistVal{\ra}]{O}{}{E}{E'}
\pstLineAB{O}{E}\pstLineAB{O}{E'}
\pstGeonode[PosAngle=0](3,-2){F}
\pstInversion[PosAngle=-120]{O}{}{F}{F'}
\pstLineAB{O}{F}\pstLineAB{O}{F'}
\pstLineAB[linecolor=black!50]{C}{F}
\pstCircleABC[linestyle=dashed,linecolor=blue!40,PosAngle=0]{C'}{D'}{E'}{O'}
\end{pspicture}
```

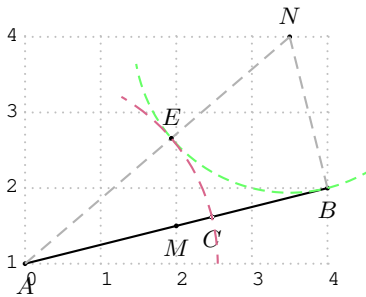
If you want to find the node  $C$  from  $A$  to  $B$ , such that  $C$  is the golden section of the given segments  $AB$ , that is,

$$|AC|^2 = |AB| \times |BC| \quad \text{or} \quad AC : AB = BC : AC \quad \text{or} \quad AC = \frac{\sqrt{5}-1}{2} AB$$

you can use the macro `\pstGoldenMean` to do this work.

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

In fact, we use the macro `\pstLocateAB` to implement this macro by passing the value  $\frac{\sqrt{5}-1}{2}|AB|$  to parameter length.



```

\begin{pspicture}[showgrid=true](0,1)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90,CurveType=polyline](0,1){A}(4,2){B}
\pstGoldenMean[PosAngle=-90,PointSymbol=o]{A}{B}{C}
% geometrical method to draw the golden point
\pstMiddleAB[PosAngle=-90]{A}{B}{M}
\pstRotation[RotAngle=-90,PosAngle=90]{B}{M}[N]
\pstLineAB[linestyle=dashed,linecolor=gray!60]{A}{N}
\pstLineAB[linestyle=dashed,linecolor=gray!60]{B}{N}
\pstInterLC[PointNameA=,PosAngleB=90]{N}{A}{N}{B}{B0}{E}
\pstCircleOA[linecolor=green!60,linestyle=dashed]{N}{B}[190][300]
\pstCircleOA[linecolor=purple!60,linestyle=dashed]{A}{E}[0][60]
\end{pspicture}

```

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,

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

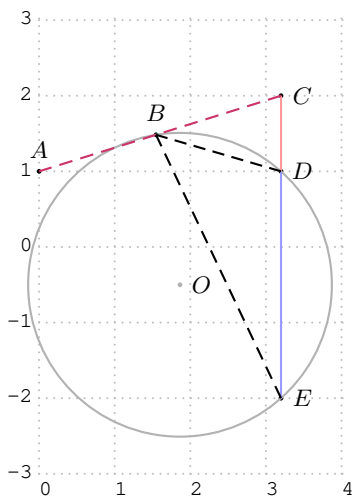
you can use the macro `\pstGeometricMean` to do this work. It also can be used to draw a circle when given two points on the circle, and a line tangents to the circle.

```

\pstGeometricMean [Options] {A}{B}{L1}{L2}{C}

```

In fact, we use the macro `\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 `\pstDist`, `\pstDistConst`, `\pstDistAdd`, `\pstDistSub`, etc.



```

\begin{pspicture}[showgrid=true](0,-3)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=90](0,1){A}
\pstGeonode[PosAngle=0](3.2,2){C}(3.2,1){D}(3.2,-2){E}
\pstGeometricMean[PosAngle=90]{C}{A}{\pstDistAB{C}{D}}{\pstDistAB{D}{E}}{B}
\pstCircleABC[linecolor=gray!60]{B}{D}{E}{O}
\pstLineAB[linecolor=red!40]{C}{D}
\pstLineAB[linecolor=blue!40]{D}{E}
\psset{linestyle=dashed}
\pstLineAB[linecolor=purple!80]{C}{A}
\pstLineAB{D}{B}\pstLineAB{E}{B}
\end{pspicture}

```

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,

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

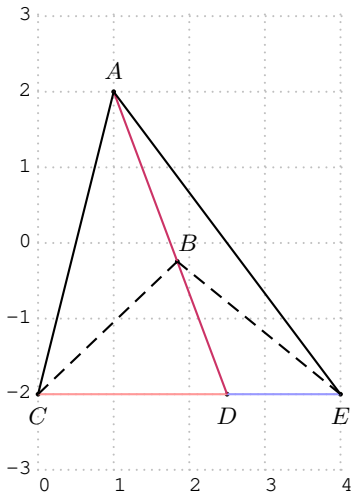
you can use the macro `\pstHarmonicMean` to do this work.

```

\pstHarmonicMean [Options] {A}{B}{L1}{L2}{C}

```

In fact, we use the macro `\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 `\pstDist`, `\pstDistConst`, `\pstDistAdd`, `\pstDistSub`, etc.



```

\begin{pspicture}[showgrid=true](0,-3)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=90](1,2){A}
\pstGeonode[PosAngle=-90](0,-2){C}(2.5,-2){D}(4,-2){E}
\pstHarmonicMean[PosAngle=60]{D}{A}{\pstDistAB{C}{D}}{\pstDistAB{
D}{E}}{B}
\pstLineAB[linecolor=red!40]{C}{D}
\pstLineAB[linecolor=blue!40]{D}{E}
\pstLineAB[linecolor=purple!80]{A}{D}
\pstLineAB[linestyle=dashed]{C}{B}
\pstLineAB[linestyle=dashed]{E}{B}
\pstLineAB{C}{A}\pstLineAB{E}{A}
\end{pspicture}

```

## 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:

```

\pstDistAB{A}{B}
\pstDistVal{l}
\pstDistCalc{expr}

```

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:

```

\pstDist{A}{B}
\pstDistConst{l}
\pstDistExpr{expr}

```

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,

```

\pnode(! 1 \pstUserDist{\pstDistAdd{A}{B}{C}{D}}){A}
\pstMoveNode(0,\pstUserDist{\pstDistAdd{A}{B}{C}{D}}){A}{E}

```

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:

```

\pstCircleOA[Radius=\pstDistConst{\pstAbscissa{A}}]{A}{}

```

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.

```
\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}
```

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  $\text{DistCoef} * |AB|$
- macro `\pstDistVal` get the screen length of  $\text{DistCoef} * l$
- macro `\pstDistCalc` get the screen length of  $\text{DistCoef} * \text{expr}$
- macro `\pstDistCoef` get the screen length of  $\text{DistCoef} * \langle \text{arg} \rangle$
- macro `\pstDist` get the screen length of  $|AB|$
- macro `\pstDistConst` get the screen length of  $l$
- macro `\pstDistExpr` get the screen length of  $\text{expr}$
- 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.

```
\pstCircle0A[Radius=\pstDistAddVal{A}{B}{2.0}{\pstDistAddCoef{C}{D}{3.0}{E}{F}{4.0}}]{A}{}
```

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

```
\pstCircle0A[DistCoef=1 3 div,Radius=\pstDistAB{A}{B}]{0}{}
\pstCircle0A[DistCoef=1 3 div,Radius=\pstDistAB{A}{B}]{A}{B}{0}{I}{J}
\pstInterCC[DistCoef=1 3 div,RadiusA=\pstDistAB{A}{B},DistCoef=none,RadiusB=\pstDistAB{C}{D}]{01}{}{02}{I}{J}
```

could be simplified to

```
\pstCircleOA[Radius=\pstDistMul{A}{B}{1 3 div}]{}{}
\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

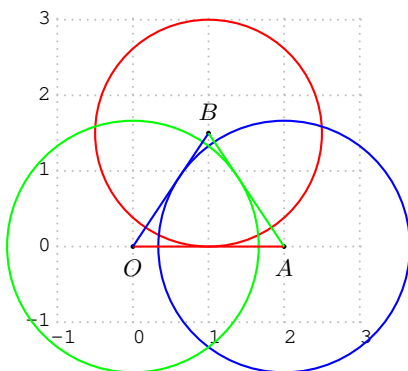
```
\pstDistAddVal{A}{B}{2.0}{\pstDistAB{C}{D}}
```

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

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

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

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



```
\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}]{}{}
\pstCircleOA[linecolor=blue,Radius=\pstDistABC{B}{O}{A}]{}{}
\pstCircleOA[linecolor=green,Radius=\pstDistABC{A}{B}{O}]{}{}
\pstLineAB[linecolor=red]{}{}
\pstLineAB[linecolor=blue]{}{}
\pstLineAB[linecolor=green]{}{}
\end{pspicture}
```

## 2.8. Circles

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

```
\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.

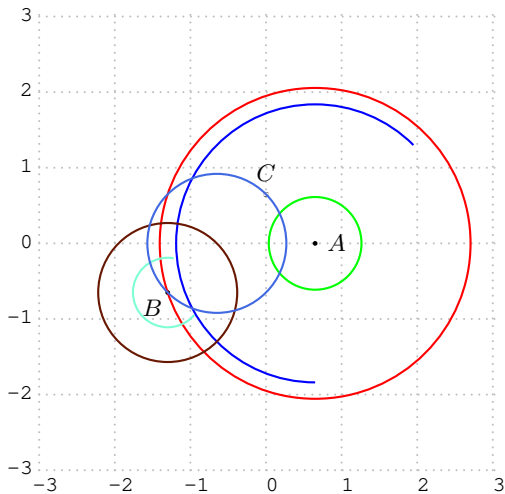
For the first macro, it is possible to omit the second point and then to specify a radius or a diameter using the parameters `Radius`<sup>4</sup> and `Diameter`. The values of these parameters can be specified with one of the `\pstDist*` series macros.

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  $B$ ;

<sup>4</sup> 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}}`.

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



```

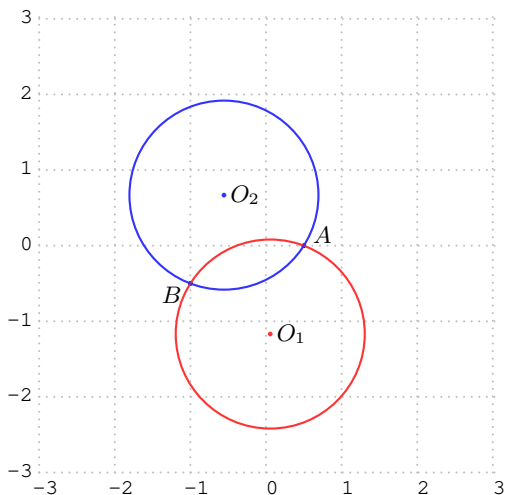
\begin{pspicture}[showgrid](-3,-3)(3,3)\footnotesize
\psset{unit=0.65cm}\psset{dotscale=0.5}\psset{PointSymbol=*}
\pstGeonode[PosAngle={0,-135,90},PointSymbol={*,*,square}](1,0){A}
(-2,-1){B}(0,1){C}
\pstCircleOA[linecolor=red]{A}{B}
\pstCircleOA[linecolor=green,Radius=\pstDistMul{A}{C}{2 3 div}]{A}{C}
\pstCircleOA[linecolor=blue,Radius=\pstDistAB{B}{C}]{A}{B}
\pstCircleOA[linecolor=Sepia,Radius=\pstDistAB{A}{C}]{B}{C}
\pstCircleOA[linecolor=Aquamarine,Diameter=\pstDistAB{A}{C}]{B}{C}
\pstCircleAB[linecolor=RoyalBlue]{B}{C}
\end{pspicture}

```

The macro `\pstCircleABR` draws the circle of given radius length  $R$ , through two given nodes  $A$  and  $B$ , then outputs the circle center  $O$ .

```
\pstCircleABR [Options] {A}{B}{R}{O}
```

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



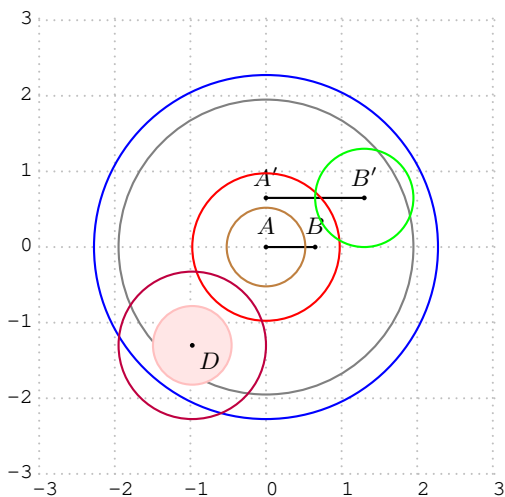
```

\begin{pspicture}[showgrid](-3,-3)(3,3)\footnotesize
\psset{unit=0.50cm}\psset{dotscale=0.5}\psset{PointSymbol=*}
\pstGeonode[PosAngle={30,210}](1,0){A}(-2,-1){B}
\pstCircleABR[linecolor=red!80]{A}{B}{\pstDistConst{2.5}}{O_1}
\pstCircleABR[linecolor=blue!80]{B}{A}{\pstDistConst{2.5}}{O_2}
\end{pspicture}

```

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





```

\begin{pspicture}[showgrid=true](-3,-3)(3,3)\footnotesize
\psset{unit=0.65cm}\psset{dotscale=0.5}\psset{PointSymbol=*}
\pstGeonode[PosAngle=90,CurveType=polyline](0,0){A}(1,0){B}
\pstGeonode[PosAngle=90,CurveType=polyline](0,1){A'}(2,1){B'}
\pstCircleOA[linecolor=gray,Radius=\pstDistAdd{A}{B}{A'}{B'}]{A}
{} % R=|AB|+|A'B'|
\pstCircleOA[linecolor=red,Radius=\pstDistAddVal{A}{B}{1.0}{\
pstDistConst{0.5}}]{A}{B} % R=|AB|+0.5
\pstCircleOA[linecolor=blue,Radius=\pstDistAddCoef{A}{B}{0.5}{A
'}{B'}{1.5}]{A}{B} % R=0.5|AB|+1.5|A'B'|
\pstCircleOA[linecolor=green,Radius=\pstDistSub{A}{B}{A'}{B'}]{B
'} % R=|AB|-|A'B'|
\pstCircleOA[linecolor=brown,Radius=\pstDistSubCoef{A}{B}{1.8}{A
'}{B'}{0.5}]{A}{B} % R=1.8|AB|-0.5|A'B'|
\node(-1.5,-2){D}
\pstCircleOA[linecolor=pink,fillstyle=solid,fillcolor=pink!40,
Radius=\pstDistMul{A}{B}{0.8}]{D} % R=0.8|AB|
\psdot(D)\uput{0.2}[-45](D){D}
\pstCircleOA[linecolor=purple,Radius=\pstDistConst{\pstAbscissa{D}
}} abs]{D} % R=|D.x|
\end{pspicture}

```

The last row set the absolute value of the abscissa of node  $D$  to 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.

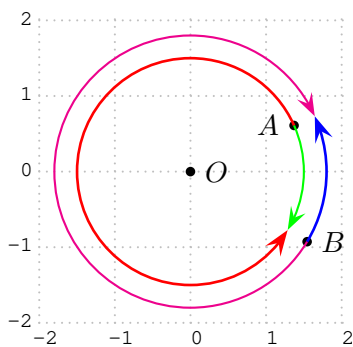
## 2.9. Circle arcs

```

\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  $O$ .



```

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

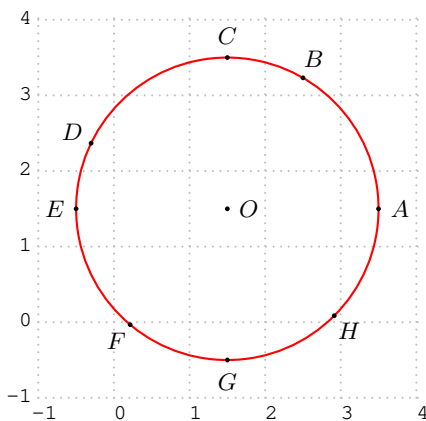
```

## 2.10. 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 `RotAngle` to calculate the point, If you not set `RotAngle`, the default value is  $60^\circ$ .

The circle is defined by center  $O$  and point  $A$  on the circle or Radius or Diameter in parameter.

```
\pstCircleNode [Options] {O}{A}{\theta}{X}
\pstCircleRotNode [Options] {O}{A}{X}
```



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

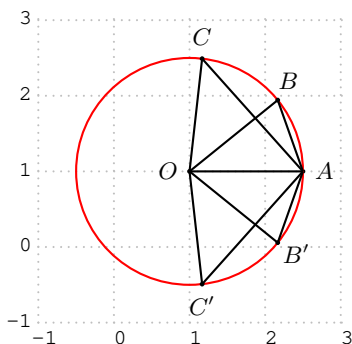
Sometimes we need to draw a chord with the given length from the start node, it is not possible to get the end node via the already defined macros, so we provide the macro `\pstCircleChordNode` to do this work. This macro find the node  $X$  on the circle such that the length of chord  $AX$  is the given value  $L$ , which can be got from `\pstDist`, `\pstDistConst`, `\pstDistAdd`, `\pstDistSub`, etc.

```
\pstCircleChordNode [Options] {O}{A}{L}{X}
```

The circle is just defined by center  $O$  and point  $A$  in this macro, so you can't omit the parameter  $A$ .

The direction to find node  $X$  is anti-clockwise by default. The parameter `CurvAbsNeg`(by default false) can change this behavior.

At last, the chord length  $L$  shouldn't large than the diameter of the circle, else we will put the node  $X$  at origin.

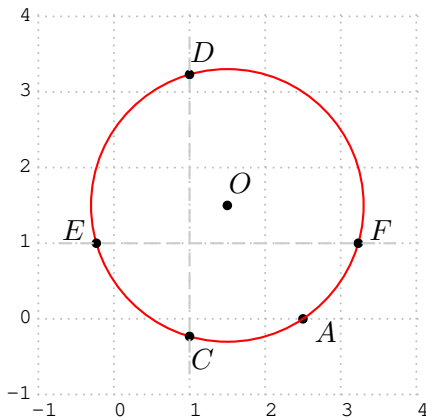


```
\begin{pspicture}[showgrid=true](-1,-1)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle={180,0}](1,1){O}(2.5,1){A}
\pstCircleOA[linecolor=red]{O}{A}
\pstCircleChordNode[PosAngle=60]{O}{A}{\pstDistConst{1}}{B}
\pstCircleChordNode[PosAngle=90]{O}{A}{\pstDistConst{2}}{C}
\pstCircleChordNode[PosAngle=-30,CurvAbsNeg=true]{O}{A}{\pstDistConst{1}}{B'}
\pstCircleChordNode[PosAngle=-90,CurvAbsNeg=true]{O}{A}{\pstDistConst{2}}{C'}
\pstLineAB{O}{A}\pstLineAB{O}{B}\pstLineAB{O}{C}
\pstLineAB{O}{B'}\pstLineAB{O}{C'}
\pstLineAB{A}{B}\pstLineAB{A}{C}
\pstLineAB{A}{B'}\pstLineAB{A}{C'}
\end{pspicture}
```

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

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

for example,



```

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

```

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

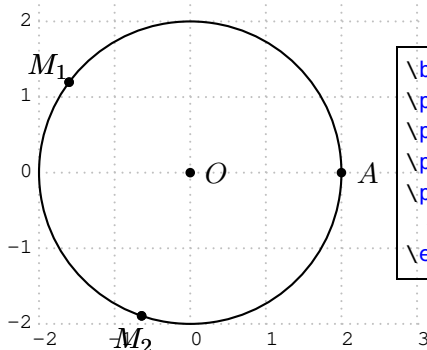
```

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

```

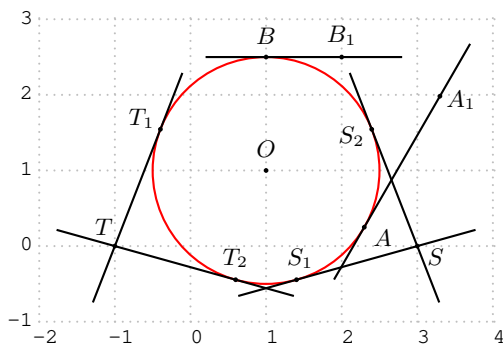
## 2.11. Circle tangent

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}

```



```

\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\psset{nodesep=-0.8}
\pstGeonode[PosAngle={90,120,-30}](1,1){O}(-1,0){T}(3,0){S}
\pstCircleOA[Radius=\pstDistVal{1.5},linecolor=red]{O}{}
\pstCircleRotNode[Radius=\pstDistVal{1.5},PosAngle=-30,RotAngle
=-30]{O}{}{A}
\pstCircleTangentLine[PosAngle=-10,PointName=A_1]{O}{A}{A1}
\pstCircleRotNode[Radius=\pstDistVal{1.5},PosAngle=90,RotAngle
=90]{O}{}{B}
\pstCircleTangentLine[PosAngle=90,PointName=B_1]{O}{B}{B1}
\pstCircleTangentNode[Radius=\pstDistVal{1.5},PosAngle={150,90},
PointName={T_1,T_2}]{O}{}{T}{T1}{T2}
\pstCircleTangentNode[PosAngle={80,200},PointName={S_1,S_2}]{O}{A}
}{S}{S1}{S2}
\end{pspicture}

```

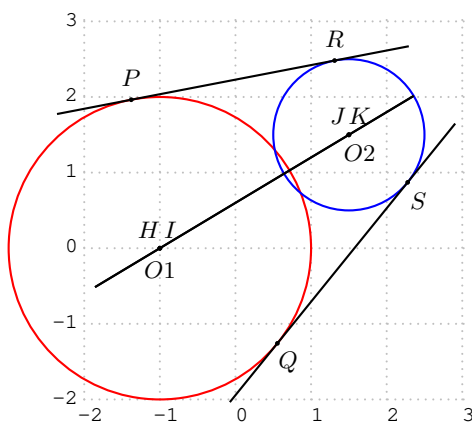
The macro `\pstCircleExternalCommonTangent` is used to find the external common tangent lines of two circles  $A(O_1)$  and  $B(O_2)$ , and the macro `\pstCircleInternalCommonTangent` is used to find the internal common tangent lines of two circles  $A(O_1)$  and  $B(O_2)$ . They both create four tangent 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  $B(O_2)$ .

```

\pstCircleExternalCommonTangent [Options] {O1}{A}{O2}{B}{T1}{T2}{T3}{T4}
\pstCircleInternalCommonTangent [Options] {O1}{A}{O2}{B}{T1}{T2}{T3}{T4}

```

You can use `RadiusA` and `RadiusB` to define the two circles like as following:

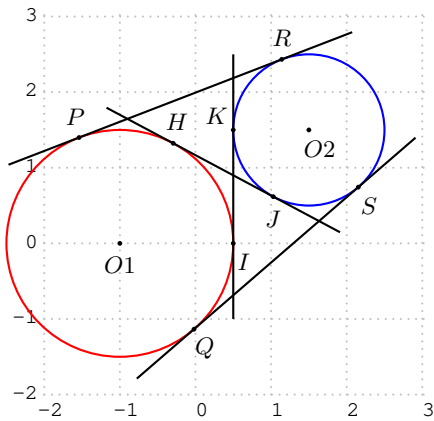


```

\begin{pspicture}[showgrid=true](-2,-2)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90](-1,0){O1}
\pstGeonode[PosAngle=-60](1.5,1.5){O2}
\pstCircleOA[Radius=\pstDistVal{2},linecolor=red]{O1}{}
\pstCircleOA[Radius=\pstDistVal{1},linecolor=blue]{O2}{}
\pstCircleExternalCommonTangent[RadiusA=\pstDistVal{2},RadiusB=\
pstDistVal{1},PosAngle={90,-60,90,-60}]{O1}{O2}{P}{Q}{R}
}{S}
\pstLine[nodesep=-1]{P}{R}
\pstLine[nodesep=-1]{Q}{S}
\pstCircleInternalCommonTangent[RadiusA=\pstDistVal{2},RadiusB=\
pstDistVal{1},PosAngle={120,60,120,60}]{O1}{O2}{H}{I}{J}
}{K}
\pstLine[nodesep=-1]{H}{J}
\pstLine[nodesep=-1]{I}{K}
\end{pspicture}

```

You also can use `DiameterA` and `DiameterB` to define the two circles like as following:



```

\begin{pspicture}[showgrid=true](-2,-2)(3,3)
\psset{dotsscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90](-1,0){O1}
\pstGeonode[PosAngle=-60](1.5,1.5){O2}
\pstCircleOA[Diameter=\pstDistVal{3},linecolor=red]{O1}{}
\pstCircleOA[Diameter=\pstDistVal{2},linecolor=blue]{O2}{}
\pstCircleExternalCommonTangent[DiameterA=\pstDistVal{3},
DiameterB=\pstDistVal{2},PosAngle={100,-60,90,-60}]{O1}{}{O
2}{}{P}{Q}{R}{S}
\pstLine[nodesep=-1]{P}{R}
\pstLine[nodesep=-1]{Q}{S}
\pstCircleInternalCommonTangent[DiameterA=\pstDistVal{3},
DiameterB=\pstDistVal{2},PosAngle={80,-60,-90,140}]{O1}{}{O
2}{}{H}{I}{J}{K}
\pstLine[nodesep=-1]{H}{J}
\pstLine[nodesep=-1]{I}{K}
\end{pspicture}

```

### 2.12. Circle radical axis

If you want to draw the Radical Axis of two given circles, read the following sentences. For given  $\odot O_1$  with radius  $r_1$  and  $\odot O_2$  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 `\pstCircleRadicalAxis` to draw the Radical Axis of two given circles. It can handler every position relations of circles such as separation, intersection and inclusion.

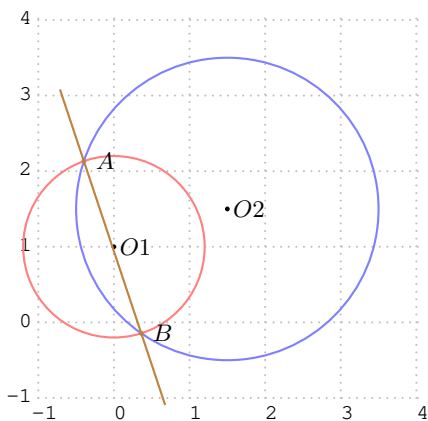
```

\pstCircleRadicalAxis [Options] {O1}{A}{O2}{B}{C}{D}

```

Both parameter  $A$  and  $B$  can be omitted and then to specify the each radius or diameter using the parameters `RadiusA`, `DiameterA`, and `RadiusB`, `DiameterB`. This macro create two new nodes  $C$  and  $D$  on the radical axis, you can find them in following examples.

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

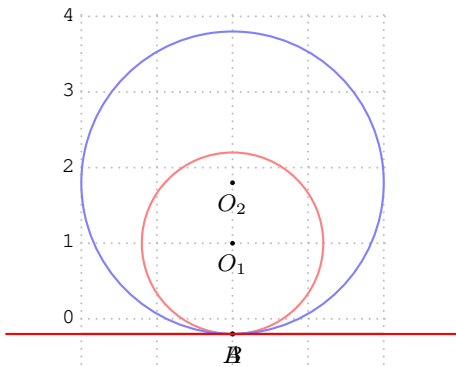


```

\begin{pspicture}[showgrid=true](-1,-1)(4,4)
\psset{dotsscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.2}\def\rb{2.0}
\pstGeonode[PosAngle=0](0,1){O1}(1.5,1.5){O2}
\pstCircleOA[linecolor=red!50,Radius=\pstDistVal{\ra}]{O1}{}
\pstCircleOA[linecolor=blue!50,Radius=\pstDistVal{\rb}]{O2}{}
\pstCircleRadicalAxis[PosAngle={0,0},RadiusA=\pstDistVal{\ra},
RadiusB=\pstDistVal{\rb},nodesep=-1,linecolor=brown]{O1}{}{O
2}{}{A}{B}
\end{pspicture}

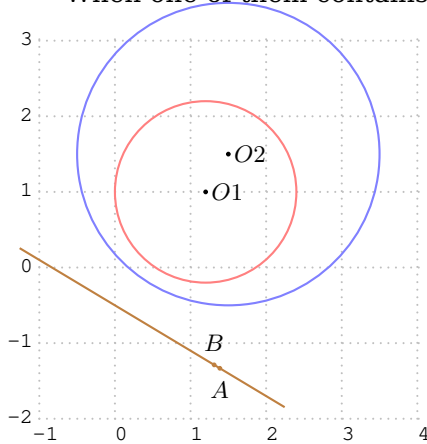
```

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



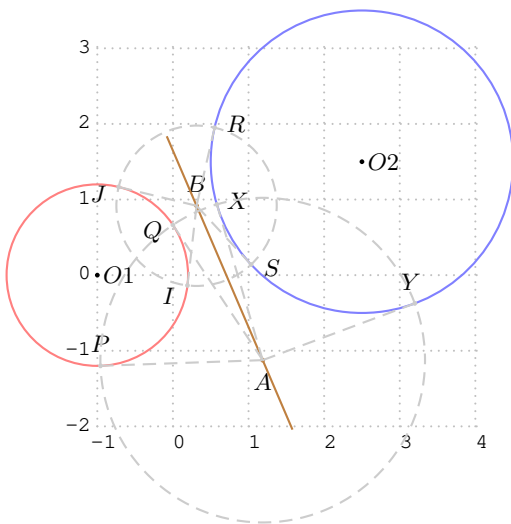
```
\begin{pspicture}[showgrid=true](-1,-1)(3,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.2}\def\rb{2.0}
\pstGeonode[PosAngle=-90,PointName={O_1,O_2}](1,1){O1}(1,1.8){O2}
\pstCircleOA[linecolor=red!50,Radius=\pstDistVal{\ra}]{O1}{}
\pstCircleOA[linecolor=blue!50,Radius=\pstDistVal{\rb}]{O2}{}
\pstCircleRadicalAxis[nodesep=-2,PosAngle={-90,-90},RadiusA=\pstDistVal{\ra},RadiusB=\pstDistVal{\rb}]{O1}{}{O2}{}{A}{B}
\pstLineAB[linecolor=red,nodesep=-3]{A}{B}
\end{pspicture}
```

When one of them contains the other, the radical axis is out of the circles.



```
\begin{pspicture}[showgrid=true](-1,-2)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.2}\def\rb{2.0}
\pstGeonode[PosAngle=0](1.2,1){O1}(1.5,1.5){O2}
\pstCircleOA[linecolor=red!50,Radius=\pstDistVal{\ra}]{O1}{}
\pstCircleOA[linecolor=blue!50,Radius=\pstDistVal{\rb}]{O2}{}
\pstCircleRadicalAxis[PosAngle={-90,90},RadiusA=\pstDistVal{\ra},RadiusB=\pstDistVal{\rb},nodesepA=-1,nodesepB=-3,linecolor=brown]{O1}{}{O2}{}{A}{B}
\end{pspicture}
```

When they are separated, the radical axis is between of the circles.



```
\begin{pspicture}[showgrid=true](-1,-2)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{1.2}\def\rb{2.0}
\pstGeonode[PosAngle=0](-1,0){O1}(2.5,1.5){O2}
\pstCircleOA[linecolor=red!50,Radius=\pstDistVal{\ra}]{O1}{}
\pstCircleOA[linecolor=blue!50,Radius=\pstDistVal{\rb}]{O2}{}
\pstCircleRadicalAxis[PosAngle={-90,90},RadiusA=\pstDistVal{\ra},RadiusB=\pstDistVal{\rb},nodesep=-1,linecolor=brown]{O1}{}{O2}{}{A}{B}
\psset{linestyle=dashed,linecolor=gray!40}
\pstCircleTangentNode[Radius=\pstDistVal{\ra},PosAngle={90,200}]{O1}{}{A}{P}{Q}
\pstCircleTangentNode[Radius=\pstDistVal{\rb},PosAngle={10,100}]{O2}{}{A}{X}{Y}
\pstCircleOA{A}{P}
\pstCircleTangentNode[Radius=\pstDistVal{\ra},PosAngle={210,200}]{O1}{}{B}{I}{J}
\pstCircleTangentNode[Radius=\pstDistVal{\rb},PosAngle={10,-10}]{O2}{}{B}{R}{S}
\pstCircleOA{B}{I}
\end{pspicture}
```

### 2.13. Regular polygons

For the 3-side and 4-side regular polygon, we provide the macro `\pstETriangleAB` and `\pstSquareAB` to draw them. In general, you can use the macro `\pstRegularPolygonAB` and `\pstRegularPolygonOA` to get a n-side regular polygon.

```

\pstETriangleAB [Options] {A}{B}{C}
\pstSquareAB [Options] {A}{B}{C}{D}
\pstRegularPolygonAB [Options] {A0}{A1}{n}{A2, A3, ..., An-1}
\pstRegularPolygonOA [Options] {O}{A0}{n}{A1, A2, ..., An-1}

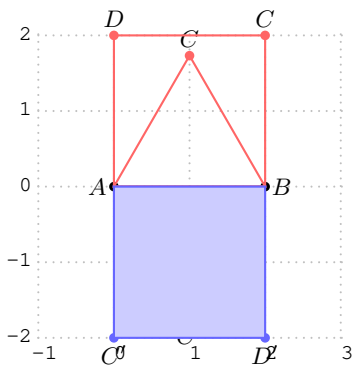
```

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

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 `\pstRegularPolygonAB` and `\pstRegularPolygonOA`, if you do not enter a complete point list, the remaining points will be automatically named.

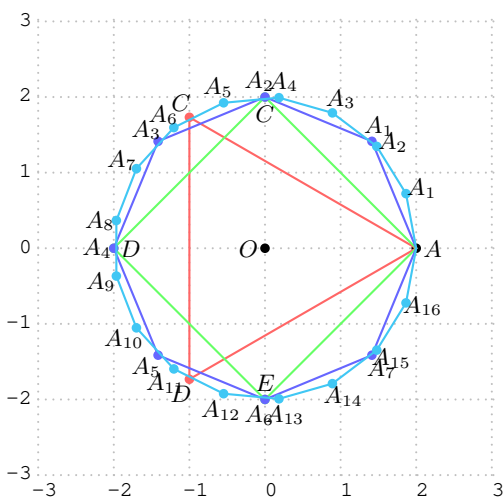
Note that draw regular polygon with side from  $A$  to  $B$  and side from  $B$  to  $A$  will get the figure symmetric to  $AB$ . Here are some examples.



```

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

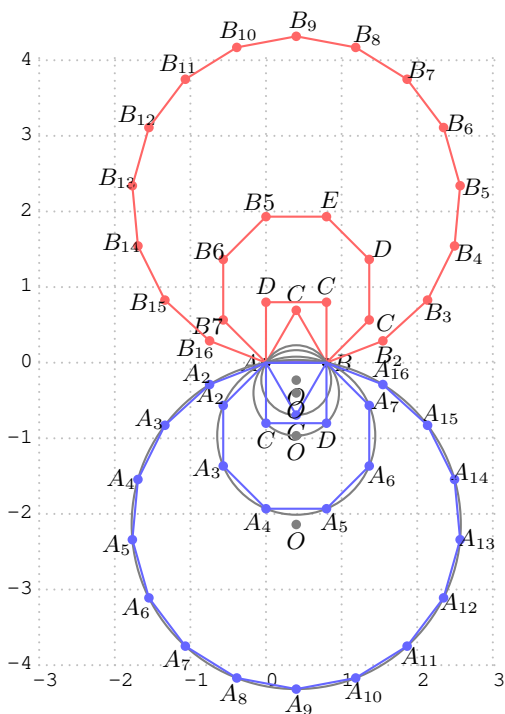
```



```

\begin{pspicture}[showgrid](-3,-3)(3,3)
\psset{unit=0.40cm}\footnotesize\psset{PointSymbol=none,
PointNameSep=0.22cm}
\pstGeonode[PosAngle={180,0},PointSymbol=*](0,0){O}(5,0){A}
\pstRegularPolygonOA[CurveType=polygon,linecolor=red!60,
PointSymbol=*,PosAngle={120,240}]{O}{A}{3}{C,D}
\pstRegularPolygonOA[CurveType=polygon,linecolor=green!60,
PointSymbol=*,PosAngle={-90,0,90}]{O}{A}{4}{C,D,E}
\pstRegularPolygonOA[CurveType=polygon,linecolor=blue!60,
PointSymbol=*,PosAngle={65,110,135,180,225,250,305},
PointName={A_1,A_2,A_3,A_4,A_5,A_6,A_7}]{O}{A}{8}{A1,A2,A3,A
4,A5,A6,A7}
\pstRegularPolygonOA[CurveType=polygon,linecolor=cyan!60,
PointSymbol=*,PosAngle
={20,15,60,75,110,130,150,180,200,220,240,260,290,310,330,350},
PointName={A_1,A_2,A_3,A_4,A_5,A_6,A_7,A_8,A_9,A_{10},A
_{11},A_{12},A_{13},A_{14},A_{15},A_{16}}]{O}{A}{17}{A1,A2,A
3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16}
\end{pspicture}

```



```

\begin{pspicture}[showgrid](-3,-4)(3,4)
\psset{unit=0.40cm}\footnotesize\psset{PointSymbol=none,
PointNameSep=0.22cm}
\pstGeonode[PosAngle={180,0},PointSymbol=*](0,0){A}(2,0){B}
\pstRegularPolygonAB[CurveType=polygon,linecolor=red!60,
PointSymbol=*,PosAngle={90,90,90}]{A}{B}{3}{C,D,E}
\pstRegularPolygonAB[CurveType=polygon,linecolor=red!60,
PointSymbol=*,PosAngle={90,90}]{A}{B}{4}{C,D,E}
\pstRegularPolygonAB[CurveType=polygon,linecolor=red!60,
PointSymbol=*,PosAngle={-10,40,80,115,150,200}]{A}{B}{8}{C,D
,E}
\pstRegularPolygonAB[CurveType=polygon,linecolor=red!60,
PointSymbol=*,PosAngle
={290,310,330,350,10,30,45,65,85,115,135,155,170,190,210},
PointName={B_2,B_3,B_4,B_5,B_6,B_7,B_8,B_9,B_10,B_11,B_12,
B_13,B_14,B_15,B_16}]{A}{B}{17}{B2}
\psset{CodeFig=true,CodeFigColor=gray,CodeFigStyle=solid}
\pstRegularPolygonAB[CurveType=polygon,linecolor=blue!60,
PointSymbol=*,PosAngle={-90,-90}]{B}{A}{3}{C,D,E}
\pstRegularPolygonAB[CurveType=polygon,linecolor=blue!60,
PointSymbol=*,PosAngle={-90,-90}]{B}{A}{4}{C,D,E}
\pstRegularPolygonAB[CurveType=polygon,linecolor=blue!60,
PointSymbol=*,PosAngle={140,190,240,-60,-40,10},PointName={A
_2,A_3,A_4,A_5,A_6,A_7}]{B}{A}{8}{A2,A3,A4,A5,A6,A7}
\pstRegularPolygonAB[CurveType=polygon,linecolor=blue!60,
PointSymbol=*,PosAngle
={140,160,180,200,220,240,260,280,300,320,340,360,20,40,60},
PointName={A_2,A_3,A_4,A_5,A_6,A_7,A_8,A_9,A_10,A_11,A
_12,A_13,A_14,A_15,A_16}]{B}{A}{17}{A2,A3,A4,A5,A
6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16}
\end{pspicture}

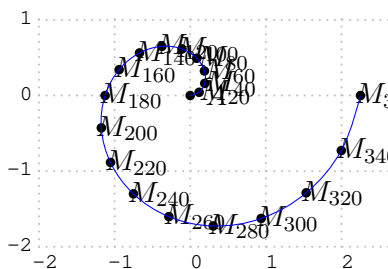
```

## 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 a interpolated curve crossing all such kind of points.

```
\pstGenericCurve [Options] {Radical}{n1}{n2}
```

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



```

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

```



### 3. Conics

#### 3.1. Standard Ellipse

The Standard Ellipse  $E$  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:

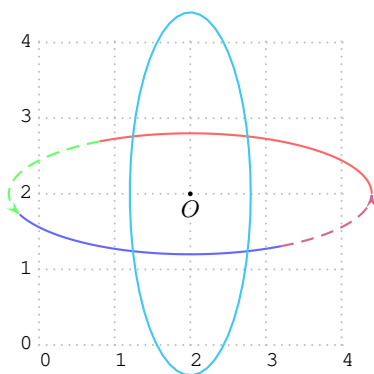
$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1 \quad (1)$$

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

$$\begin{cases} x = a \cos \alpha + x_0 \\ y = b \sin \alpha + y_0 \end{cases} \quad (2)$$

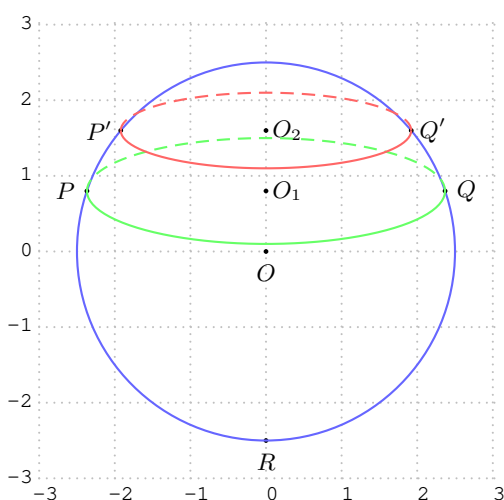
The Macro `\pstEllipse` is used to draw a Standard Ellipse with center  $O$  from `angleA` to `angleB`, going counter clockwise. It combines the function like `\psellipse` and `\psellipticarc` in PSTricks. If `angleA` and `angleB` are not specified, the macro will draw the whole ellipse.

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



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

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,

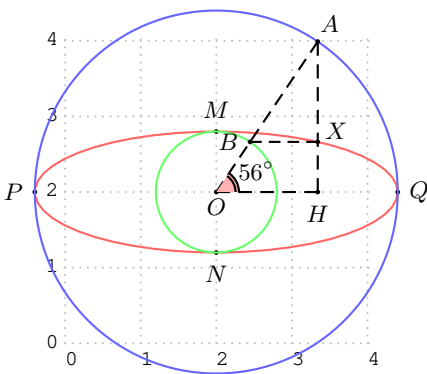


```
\begin{pspicture}[showgrid=true](-3,-3)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90,PointSymbol=*](0,0){O}(0,-2.5){R}
\pstCircleOA[linecolor=blue!60]{O}{R}
\pstGeonode[PosAngle=0,PointName=O_1,PointSymbol=*](0,0.8){O1}
\pstGeonode[PosAngle=0,PointName=O_2,PointSymbol=*](0,1.6){O2}
\pstCircleOrdNode[PointName={P,Q},PosAngle={180,0}]{O}{R}{\
pstOrdinate{O1}}{P}{Q}
\pstCircleOrdNode[PointName={P',Q'},PosAngle={180,0}]{O}{R}{\
pstOrdinate{O2}}{P'}{Q'}
\pstEllipse[linecolor=green!60,linestyle=dashed](O1)(! \
pstUserDist{\pstDist{O1}{P}} 0.7)[0][180]
\pstEllipse[linecolor=green!60](O1)(! \pstUserDist{\pstDist{O1}{P}
}} 0.7)[180][360]
\pstEllipse[linecolor=red!60,linestyle=dashed](O2)(! \pstUserDist
{\pstDist{O2}{P'}}} 0.5)[0][180]
\pstEllipse[linecolor=red!60](O2)(! \pstUserDist{\pstDist{O2}{P
'}}} 0.5)[180][360]
\end{pspicture}
```

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

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

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

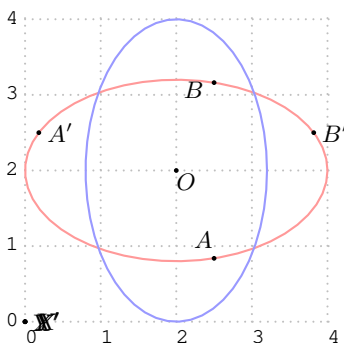


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{0.8}\def\rot{56}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
%\psellipse[linecolor=red!60](O)(\ra,\rb)
\pstEllipse[linecolor=red!60](O)(\ra,\rb)
\pstEllipseNode[PosAngle=180](O)(\ra,\rb){Q}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

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

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

```
\pstEllipseAbsNode [Options] (O)(a,b){x1}{A}{B}
\pstEllipseOrdNode [Options] (O)(a,b){y1}{A}{B}
```

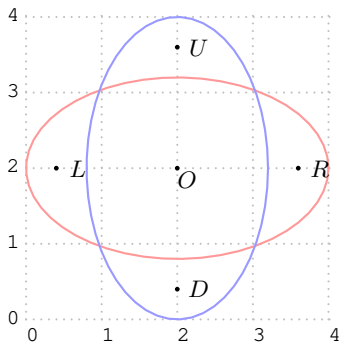


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](O)(\ra,\rb)
\pstEllipse[linecolor=blue!40](O)(\rb,\ra)
\pstEllipseAbsNode[PosAngle={120,200}](O)(\ra,\rb){2.5}{A}{B}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

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

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

For example:

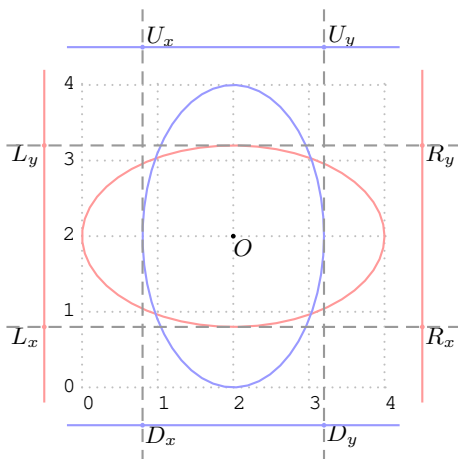


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](0)(\ra,\rb)
\pstEllipse[linecolor=blue!40](0)(\rb,\ra)
\pstEllipseFocusNode(0)(\ra,\rb){L}{R}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

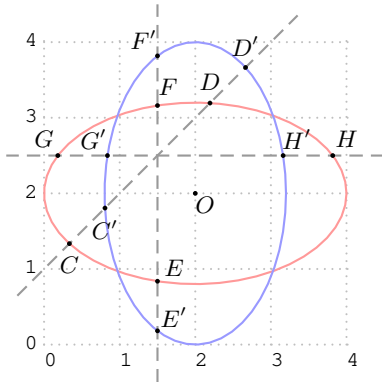


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](0)(\ra,\rb)
\pstEllipse[linecolor=blue!40](0)(\rb,\ra)
\pstEllipseDirectrixLine[PointName={L_x,L_y,R_x,R_y},PosAngle
={210,210,-30,-30},nodesep=-1,linecolor=red!40](0)(\ra,\rb){
L_x}{L_y}{R_x}{R_y}
\pstEllipseDirectrixLine[PointName={D_x,D_y,U_x,U_y},PosAngle
={-30,-30,30,30},nodesep=-1,linecolor=blue!40](0)(\rb,\ra){
D_x}{D_y}{U_x}{U_y}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{L_x}{R_x}
}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{L_y}{R_y}
}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{D_x}{U_x}
}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]{D_y}{U_y}
}
\end{pspicture}
```

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

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

Here is examples:



```

\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotsscale=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](0)(\ra,\rb)
\pstEllipse[linecolor=blue!40](0)(\rb,\ra)
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]
]{0,1}{3,4}
\pstEllipseLineInter[PosAngle={-90,90}](0)(\ra,\rb){0,1}{3,4}{C}{
D}
\pstEllipseLineInter[PosAngle={-90,90}](0)(\rb,\ra){0,1}{3,4}{C
'}{D'}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]
]{1.5,0}{1.5,4}
\pstEllipseLineInter[PosAngle={40,60}](0)(\ra,\rb){1.5,0}{1.5,4}{
E}{F}
\pstEllipseLineInter[PosAngle={40,130}](0)(\rb,\ra)
]{1.5,1}{1.5,4}{E'}{F'}
\pstLine[nodesep=-0.5,linecolor=black!40,linestyle=dashed]
]{4,2.5}{0,2.5}
\pstEllipseLineInter[PosAngle={130,50}](0)(\ra,\rb)
]{4,2.5}{0,2.5}{G}{H}
\pstEllipseLineInter[PosAngle={130,50}](0)(\rb,\ra)
]{4,2.5}{0,2.5}{G'}{H'}
\end{pspicture}

```

The macro `\pstEllipsePolarNode` is used 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 intersection node  $T$  of the two tangent lines. We call  $T$  as the polar point of chord  $AB$  as normal.

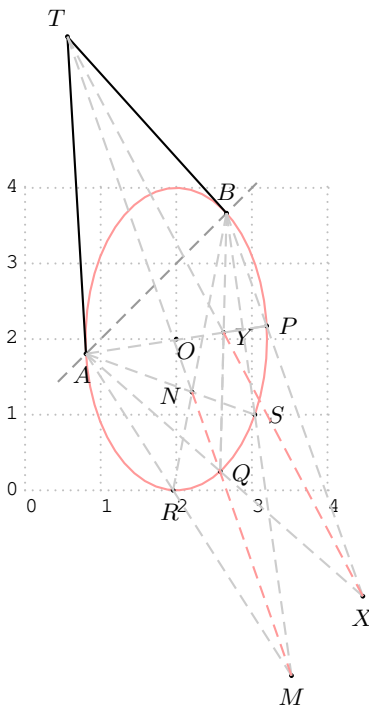
```

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

```

We use the following theorem to find the node  $T$ :

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



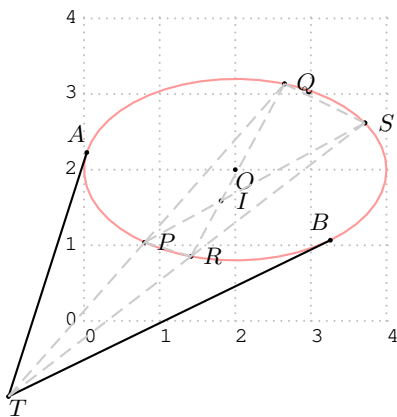
```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.0}\def\rb{-1.2}
\pstGeonode[PosAngle=-50,PointNameSep=0.2](2,2){O}
\pstEllipse[linecolor=red!40](0)(\ra,\rb)
\pstLine[nodesep=-0.8,linecolor=black!40,linestyle=dashed]
{1,2}{2.5,3.5}
\pstEllipseLineInter[PosAngle={-100,90}](0)(\ra,\rb)
{1,2}{2.5,3.5}{A}{B}
\pstEllipsePolarNode[PosAngle=120](0)(\ra,\rb){A}{B}{T}
% Here are the auxiliary lines to explain Theorem 1.
\pstEllipseRotNode[PosAngle=0,RotAngle=5](0)(\ra,\rb){P}
\pstEllipseRotNode[PosAngle=-10,RotAngle=-61](0)(\ra,\rb){Q}
\pstEllipseRotNode[PosAngle=-100,RotAngle=-92](0)(\ra,\rb){R}
\pstEllipseRotNode[PosAngle=0,RotAngle=-30](0)(\ra,\rb){S}
\pstInterLL[PosAngle=-90]{A}{Q}{B}{P}{X}
\pstInterLL[PosAngle=-10]{A}{P}{B}{Q}{Y}
\pstInterLL[PosAngle=-90]{A}{R}{B}{S}{M}
\pstInterLL[PosAngle=190]{A}{S}{B}{R}{N}
\psset{linestyle=dashed,linecolor=gray!40}
\pstLine{A}{Q}\pstLine{B}{P}\pstLine{A}{P}\pstLine{B}{Q}
\pstLine{A}{R}\pstLine{B}{S}\pstLine{A}{S}\pstLine{B}{R}
\pstLine{Q}{X}\pstLine{Q}{Y}\pstLine{P}{X}\pstLine{P}{Y}
\pstLine{R}{M}\pstLine{S}{M}\pstLine{T}{Y}\pstLine{T}{N}
\pstLine[linestyle=dashed,linecolor=red!40]{X}{Y}
\pstLine[linestyle=dashed,linecolor=red!40]{M}{N}
\end{pspicture}
```

The macro `\pstEllipseTangentNode` is used to draw the tangent line of a point  $T$  out of the Standard Ellipse  $E$ . It draws the two tangent lines through the point  $T$  to the Standard Ellipse  $E$  and gets the nodes  $A$  and  $B$  on the Ellipse.

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

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

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



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

### 3.2. General Ellipse

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

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

then we have

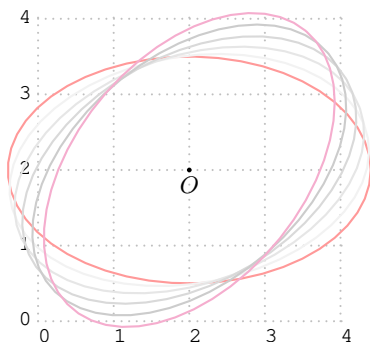
$$\begin{cases} x' = (a \cos \alpha + x_0) \cos \theta - (b \sin \alpha + y_0) \sin \theta = a \cos \alpha \cos \theta - b \sin \alpha \sin \theta + x'_0 \\ y' = (a \cos \alpha + x_0) \sin \theta + (b \sin \alpha + y_0) \cos \theta = a \cos \alpha \sin \theta + b \sin \alpha \cos \theta + y'_0 \end{cases} \quad (4)$$

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:

$$\begin{cases} x = a \cos \alpha \cos \theta - b \sin \alpha \sin \theta + x_0 \\ y = a \cos \alpha \sin \theta + b \sin \alpha \cos \theta + y_0 \end{cases} \quad (5)$$

The Macro `\pstGeneralEllipse` is used to draw a General Ellipse with center  $O$  from angleA to angleB, going counter clockwise. If angleA and angleB are not specified, the macro will draw the whole ellipse. If you not input 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`!

`\pstGeneralEllipse` [Options] (O) (a, b) [ $\theta$ ] [angleA] [angleB]

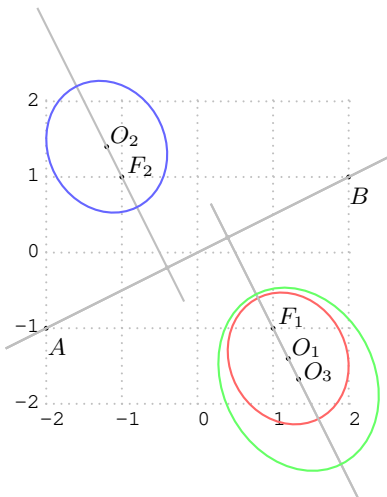


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{-1.5}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=red!40](O)(\ra,\rb)[0]
\pstGeneralEllipse[linecolor=gray!10](O)(\ra,\rb)[10]
\pstGeneralEllipse[linecolor=gray!20](O)(\ra,\rb)[20]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

The Macro `\pstGeneralEllipseFle` 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.

`\pstGeneralEllipseFle` [Options] {F}{A}{B}{e}{O}{Rab}{ $\theta$ }

The output parameter 0 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.



```

\begin{pspicture}[showgrid=true](-2,-2)(2,2)
\psset{unit=1.0cm}\psset{dotscale=0.5}\footnotesize
\psset{CodeFig=true,CodeFigColor=gray!50}\psset{PointSymbol=*}
\pstGeonode[PosAngle=30](1,-1){F_1}
\pstGeonode[PosAngle=30](-1,1){F_2}
\pstGeonode[PosAngle=-60](-2,-1){A}
\pstGeonode[PosAngle=-60](2,1){B}
\pstGeneralEllipseFle[PosAngle=30]{F_1}{A}{B}{0.5}{0_1}{R_1}{
MajorRotAngle1}
\pstGeneralEllipse[linecolor=red!60](0_1)(R_1)[MajorRotAngle1]
\pstGeneralEllipseFle[PosAngle=30]{F_2}{A}{B}{0.5}{0_2}{R_2}{
MajorRotAngle2}
\pstGeneralEllipse[linecolor=blue!60](0_2)(R_2)[MajorRotAngle2]
\pstGeneralEllipseFle[PosAngle=20]{F_1}{A}{B}{0.6}{0_3}{R_3}{
MajorRotAngle3}
\pstGeneralEllipse[linecolor=green!60](0_3)(R_3)[MajorRotAngle3]
\end{pspicture}

```

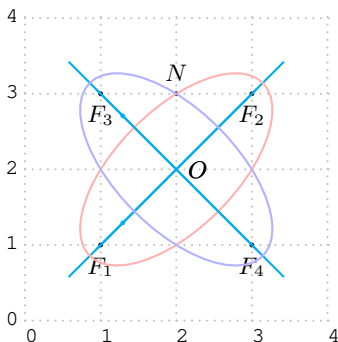
The Macro `\pstGeneralEllipseFFN` is used to define a General Ellipse by the given focus nodes  $F_1$ ,  $F_2$ , and one node  $N$  on it. It just calculate the center  $O$ , major radius  $a$ , minor radius  $b$  and the rotation angle  $\theta$  of the major axis, then you can pass them into macro `\pstGeneralEllipse` to draw this ellipse.

```

\pstGeneralEllipseFFN [Options] {F_1}{F_2}{O}{Rab}{\theta}

```

The output parameter  $O$ , the output parameter  $Rab$  and the output parameter  $\theta$  are same with `\pstGeneralEllipseFle`.



```

\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90](1,1){F_1}
\pstGeonode[PosAngle=-90](3,3){F_2}
\pstGeonode[PosAngle=-90](1,3){F_3}
\pstGeonode[PosAngle=-90](3,1){F_4}
\pstGeonode[PosAngle=90](2,3){N}
\pstGeneralEllipseFFN[linecolor=red!30,CodeFig=true]{F_1}{F_2}{N}
{O}{R1}{angle1}
\pstGeneralEllipse[linecolor=red!30](O)(R1)[angle1]
\pstGeneralEllipseFFN[linecolor=blue!30,CodeFig=true]{F_3}{F_4}{N}
{O}{R2}{angle2}
\pstGeneralEllipse[linecolor=blue!30](O)(R2)[angle2]
\end{pspicture}

```

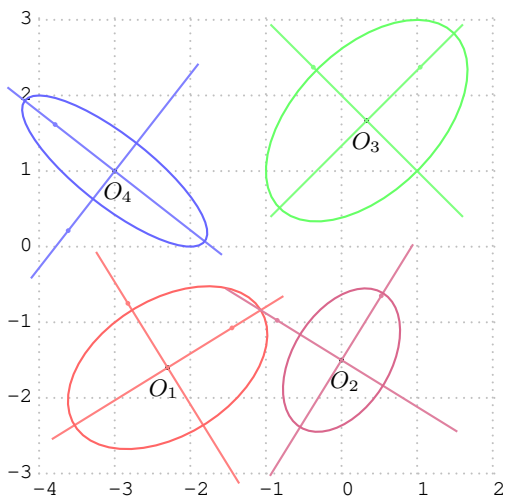
The Macro `\pstGeneralEllipseCoef` is used to define a General Ellipse by the quadratic curve equation  $ax^2 + bxy + cy^2 + dx + ey + f = 0$ , it just calculate the center  $O$ , major radius  $a$ , minor radius  $b$  and the rotation angle  $\theta$  of the major axis, then you can pass them into macro `\pstGeneralEllipse` to draw this ellipse. 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.

```

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

```

The output parameter  $O$ , the output parameter  $Rab$  and the output parameter  $\theta$  are same with `\pstGeneralEllipseFle`. They are set to zero if the coefficients are invalid to construct an ellipse.

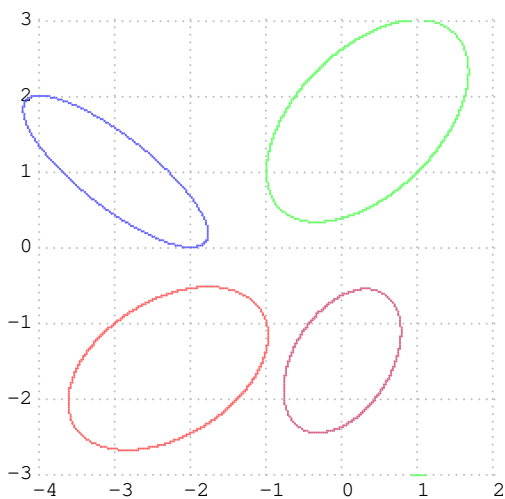


```

\begin{pspicture}[showgrid=true](-4,-3)(2,3)
\psset{unit=1.0cm}\footnotesize\psset{dotscale=0.5}
\psset{CodeFig=true}\psset{PointSymbol=*}
%2x^2-2xy+3y^2+6x+5y+8=0
\pstGeneralEllipseCoeF[PosAngle=-100,CodeFigColor=red
!50]{2,-2,3,6,5,8}{O_1}{R_1}{MajorRotAngle1}
\pstGeneralEllipse[linecolor=red!60](O_1)(R_1)[MajorRotAngle1]
%3x^2-2xy+2y^2-3x+6y+3=0
\pstGeneralEllipseCoeF[PosAngle=-80,CodeFigColor=purple
!50]{3,-2,2,-3,6,3}{O_2}{R_2}{MajorRotAngle2}
\pstGeneralEllipse[linecolor=purple!60](O_2)(R_2)[MajorRotAngle2]
%x^2-xy+y^2+x-3y+1=0
\pstGeneralEllipseCoeF[PosAngle=-90,CodeFigColor=green
!50]{1,-1,1,1,-3,1}{O_3}{R_3}{MajorRotAngle3}
\pstGeneralEllipse[linecolor=green!60](O_3)(R_3)[MajorRotAngle3]
%2x^2+4xy+3y^2+8x+6y+8=0
\pstGeneralEllipseCoeF[PosAngle=-80,CodeFigColor=blue
!50]{2,4,3,8,6,8}{O_4}{R_4}{MajorRotAngle4}
\pstGeneralEllipse[linecolor=blue!60](O_4)(R_4)[MajorRotAngle4]
\end{pspicture}

```

You can verify the output figures with `\psplotImp` as following:



```

\begin{pspicture}[showgrid=true](-4,-3)(2,3)
\psset{unit=1.0cm}\footnotesize\psset{dotscale=0.5}
%2x^2-2xy+3y^2+6x+5y+8=0
\psplotImp[linecolor=red!60](-4,-3)(2,3){ 2 x dup mul mul -2 x
mul y mul add 3 y dup mul mul add 6 x mul add 5 y mul add 8
add }
%3x^2-2xy+2y^2-3x+6y+3=0
\psplotImp[linecolor=purple!60](-4,-3)(2,3){ 3 x dup mul mul -2 x
mul y mul add 2 y dup mul mul add -3 x mul add 6 y mul add
3 add }
%x^2-xy+y^2+x-3y+1=0
\psplotImp[linecolor=green!60](-4,-3)(2,3){ 1 x dup mul mul -1 x
mul y mul add 1 y dup mul mul add 1 x mul add -3 y mul add 1
add }
%2x^2+4xy+3y^2+8x+6y+8=0
\psplotImp[linecolor=blue!60](-5,-3)(2,3){ 2 x dup mul mul 4 x
mul y mul add 3 y dup mul mul add 8 x mul add 6 y mul add 8
add }
\end{pspicture}

```

The Macro `\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 `\pstGeneralEllipse` to draw this ellipse.

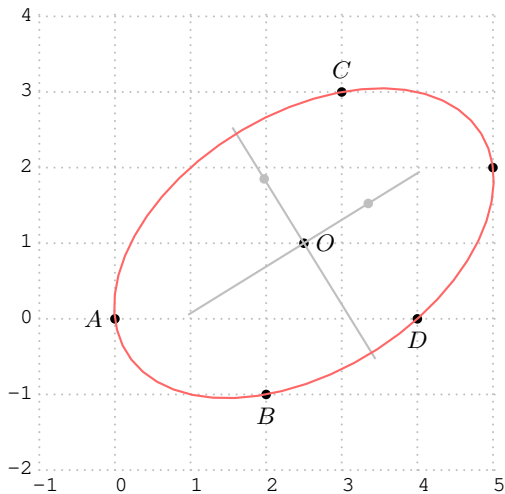
```

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

```

The output parameter  $O$ , the output parameter  $Rab$  and the output parameter  $\theta$  are same with `\pstGeneralEllipseFle`. They are set to zero if the points are invalid to construct an ellipse.



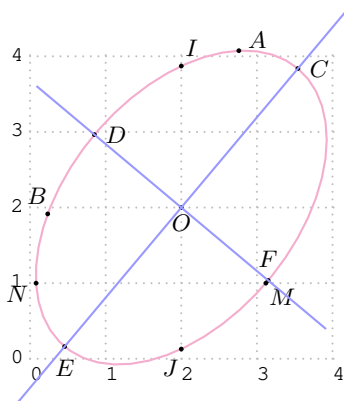


```
\begin{pspicture}[showgrid=true](-1,-2)(5,4)
\psset{unit=1.0cm}\footnotesize\psset{PointSymbol=*}
\psset{CodeFig=true,CodeFigColor=gray!50}
\pstGeonode[PosAngle=180](0,0){A}
\pstGeonode[PosAngle=-90](2,-1){B}
\pstGeonode[PosAngle=90](3,3){C}
\pstGeonode[PosAngle=-90](4,0){D}
\pstGeonode[PosAngle=0](5,2){E}
\pstGeneralEllipseABCDE[PosAngle=0]{A}{B}{C}{D}{E}{0}{R}{
MajorRotAngle}
\pstGeneralEllipse[linecolor=red!60](0)(R)[MajorRotAngle]
\end{pspicture}
```

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

```
\pstGeneralEllipseNode [Options] (O)(a,b)[ $\theta$ ]{t}{A}
\pstGeneralEllipseRotNode [Options] (O)(a,b)[ $\theta$ ]{A}
\pstGeneralEllipseAbsNode [Options] (O)(a,b)[ $\theta$ ]{x1}{A}{B}
\pstGeneralEllipseOrdNode [Options] (O)(a,b)[ $\theta$ ]{y1}{A}{B}
```

Some examples all together:

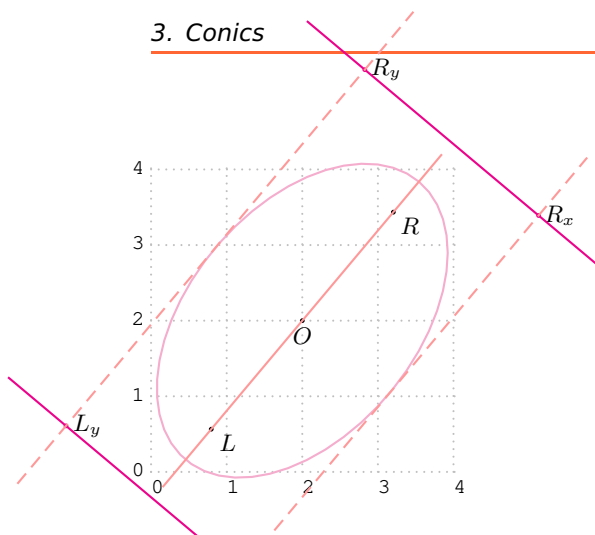


```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ra{2.4}\def\rb{-1.5}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,2){O}
\pstGeneralEllipse[linecolor=magenta!40](0)(\ra,\rb)[50]
\pstGeneralEllipseNode[PosAngle=30](0)(\ra,\rb)[50]{30}{A}
\pstGeneralEllipseRotNode[PosAngle=120,RotAngle=120](0)(\ra,\rb)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

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

```
\pstGeneralEllipseFocusNode [Options] (O)(a,b)[ $\theta$ ]{t}{F1}{F2}
\pstGeneralEllipseDirectrixLine [Options] (O)(a,b)[ $\theta$ ]{Lx}{Ly}{Rx}{Ry}
```

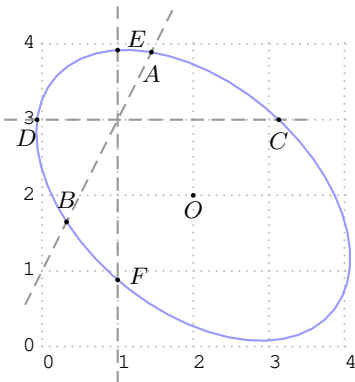
for example,



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

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

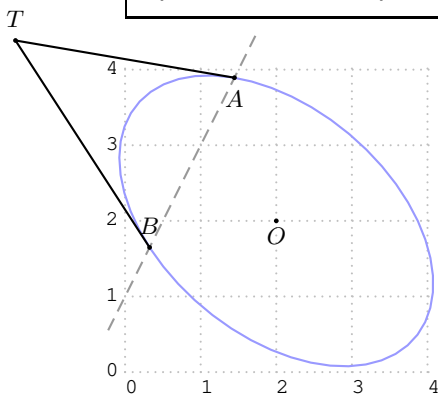
```
\pstGeneralEllipseLineInter [Options] (O)(a,b)[\theta]{A}{B}{C}{D}
```



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

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

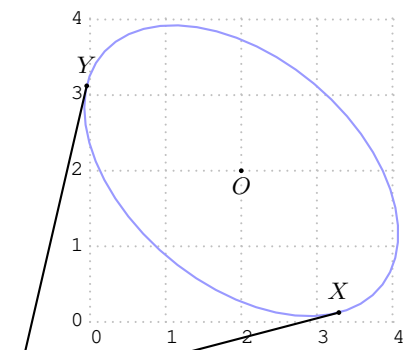
```
\pstGeneralEllipsePolarNode [Options] (O)(a,b)[\theta]{A}{B}{T}
```



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

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

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



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

### 3.3. Standard Parabola

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

The equation can be written as:

$$(x - x_0)^2 = 2p(y - y_0) \quad (6)$$

and the parametric function can be written as:

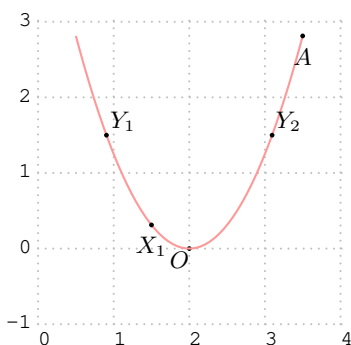
$$\begin{cases} x = t + x_0 \\ y = \frac{t^2}{2p} + y_0 \end{cases} \quad (7)$$

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

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

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

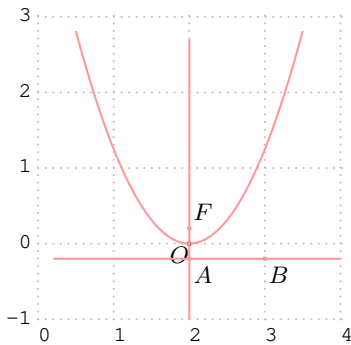
```
\pstParabolaNode [Options] (O){p}{t}{A}
\pstParabolaAbsNode [Options] (O){p}{x1}{A}
\pstParabolaOrdNode [Options] (O){p}{y1}{A}{B}
```



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

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

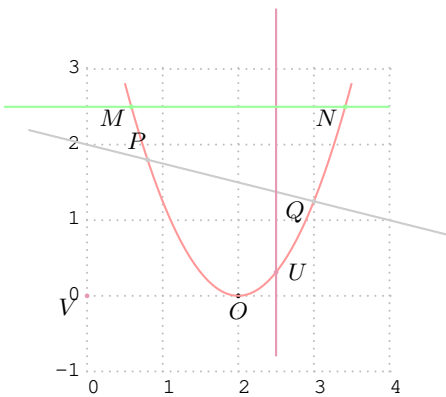
```
\pstParabolaFocusNode [Options] (O){p}{F}
\pstParabolaDirectrixLine [Options] (O){p}{Lx}{Ly}
```



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

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

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



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

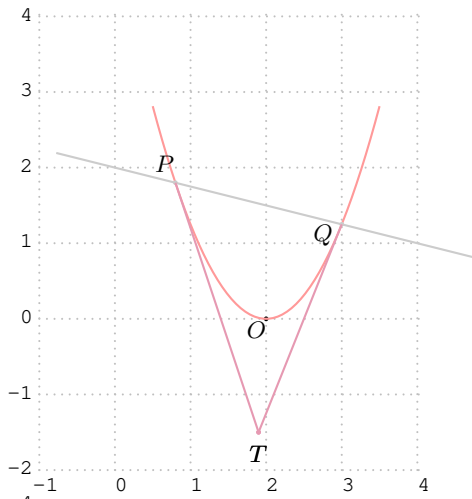
The macro `\pstParabolaPolarNode` is used to find the polar point  $T$  of chord  $AB$  on Parabola  $P$ .

```
\pstParabolaPolarNode [Options] (O){p}{A}{B}{T}
\pstParabolaPolarNode [Options] (O){p}{F}{A}{B}{T}
\pstParabolaPolarNode [Options] (O){p}{F}[Lx][Ly]{A}{B}{T}
```

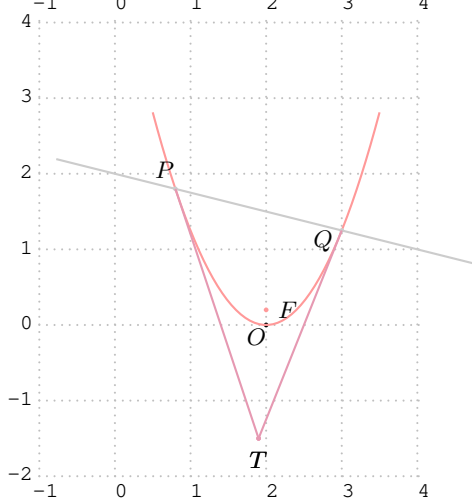
We use the following theorem to find the polar point  $T$  of chord  $AB$ :

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

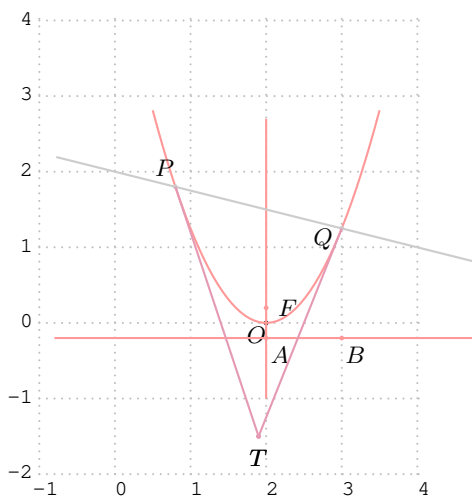
If you don't know the focus  $F$ , or the directrix line, we will find them automated, otherwise you can pass them to this macro.



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



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



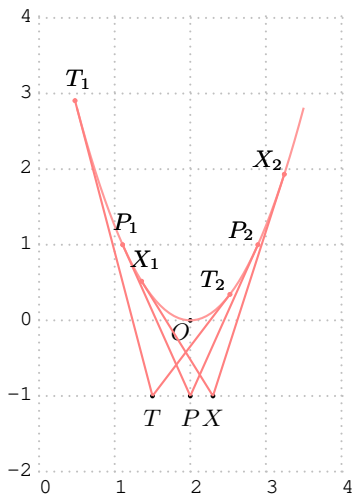
```
\begin{pspicture}[showgrid=true](-1,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstParabola[linecolor=red!40](O){\p}{-1.5}{1.5}
\pstParabolaFocusNode[linecolor=red!40](O){\p}{F}
\pstParabolaDirectrixLine[linecolor=red!40,nodesepA=-2.8,nodesepB=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

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

```
\pstParabolaTangentNode [Options] (O){p}{T}{A}{B}
```

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  $TPQ$  and  $TRS$ ,  $PS$  and  $QR$  intersect at  $I$ ,  $PR$  and  $QS$  intersect at  $X$ ,  $XI$  and Parabola intersect at  $A$  and  $B$ , then  $TA$  is the tangent line through  $A$  and  $TB$  is the tangent line through  $B$ .



```
\begin{pspicture}[showgrid=true](0,-2)(4,4)
\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}
\pstGeonode[PosAngle=-90](1.5,-1){T}
\pstParabolaTangentNode[linecolor=red!50,PosAngle={80,140},
PointName={T_1,T_2}](0){\p}{T}{T1}{T2}
\pstGeonode[PosAngle=-90](2,-1){P}
\pstParabolaTangentNode[linecolor=red!50,PosAngle={80,140},
PointName={P_1,P_2}](0){\p}{P}{P1}{P2}
\pstGeonode[PosAngle=-90](2.3,-1){X}
\pstParabolaTangentNode[linecolor=red!50,PosAngle={80,140},
PointName={X_1,X_2}](0){\p}{X}{X1}{X2}
\end{pspicture}
```

### 3.4. Standard Inversion Parabola

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

$$(y - y_0)^2 = 2p(x - x_0) \quad (8)$$

and the parametric function can be written as:

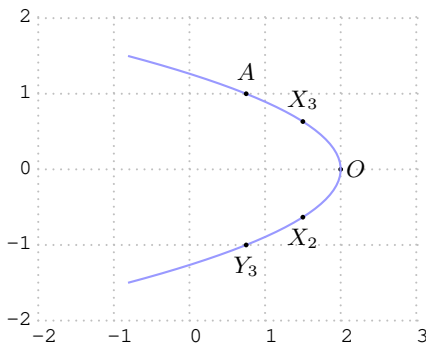
$$\begin{cases} x = \frac{t^2}{2p} + x_0 \\ y = t + y_0 \end{cases} \quad (9)$$

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

```
\pstIParabola [Options] (O){p}{y1}{y2}
```

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

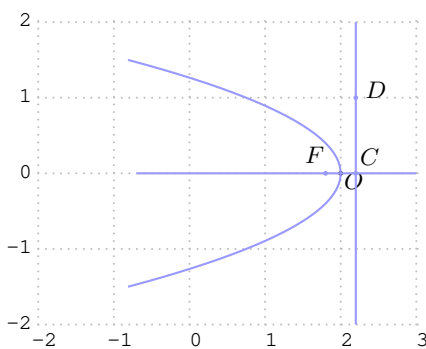
```
\pstIParabolaNode [Options] (O){p}{t}{A}
\pstIParabolaAbsNode [Options] (O){p}{x1}{A}{B}
\pstIParabolaOrdNode [Options] (O){p}{y1}{A}
```



```
\begin{pspicture}[showgrid=true](-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}
\pstIParabolaNode[PosAngle=90](0){-\p}{1}{A}
\pstIParabolaAbsNode[PosAngle=90,PointName={X_2,X_3},PosAngle
=-90,90](0){-\p}{1.5}{X2}{X3}
\pstIParabolaOrdNode[PosAngle=-90,PointName=Y_3](0){-\p}{-1}{Y3}
\end{pspicture}
```

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

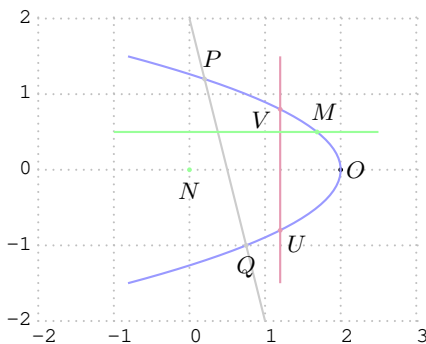
```
\pstIParabolaFocusNode [Options] (O){p}{F}
\pstIParabolaDirectrixLine [Options] (O){p}{L_x}{L_y}
```



```
\begin{pspicture}[showgrid=true](-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}
\pstIParabolaFocusNode[linecolor=blue!40,PosAngle=120](0){-\p}{F}
\pstIParabolaDirectrixLine[linecolor=blue!40,nodesepA=-2,nodesepB
=-1,PosAngle={50,20}](0){-\p}{C}{D}
\pstLine[linecolor=blue!40,nodesepA=-0.8,nodesepB=-2.5]{C}{F}
\end{pspicture}
```

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

```
\pstIParabolaLineInter [Options] (O){p}{A}{B}{C}{D}
```

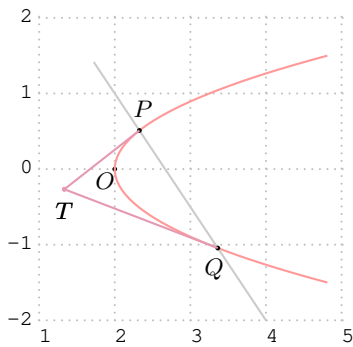


```
\begin{pspicture}[showgrid=true](-2,-2)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=0,PointNameSep=0.2](2,0){O}
\pstIParabola[linecolor=blue!40](0){-\p}{-1.5}{1.5}
\pstLine[linecolor=gray!40]{0,2}{1,-2}
\pstIParabolaLineInter[linecolor=gray!40,PosAngle={70,-90}](0){-\p}{1,-2}{0,2}{P}{Q}
\pstLine[linecolor=purple!40]{1.2,-1.5}{1.2,1.5}
\pstIParabolaLineInter[linecolor=purple!40,PosAngle={-40,210}](0){-\p}{1.2,-1.5}{1.2,1.5}{U}{V}
\pstLine[linecolor=green!40]{-1,0.5}{2.5,0.5}
\pstIParabolaLineInter[linecolor=green!40,PosAngle={70,-90}](0){-\p}{-1,0.5}{2.5,0.5}{M}{N}
\end{pspicture}
```

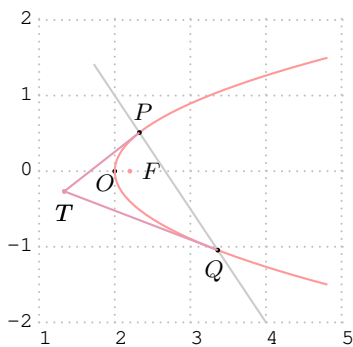
The macro `\pstIParabolaPolarNode` is used to find the polar point  $T$  of chord  $AB$  on Parabola  $P$ .

```
\pstIParabolaPolarNode [Options] (O){p}{A}{B}{T}
\pstIParabolaPolarNode [Options] (O){p}(F){A}{B}{T}
\pstIParabolaPolarNode [Options] (O){p}(F) [L_x] [L_y] {A}{B}{T}
```

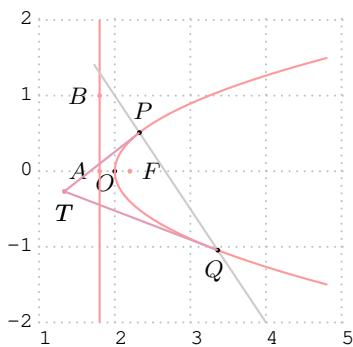
We also use the theorem 3 to find the polar point  $T$  of chord  $AB$ . If you don't know the focus  $F$ , or the directrix line, we will find them automated, otherwise you can pass them to this macro.



```
\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}
\pstIParabola[linecolor=red!40](0){\p}{-1.5}{1.5}
\pstLine[linecolor=gray!40,nodesepA=-0.5]{2,1}{4,-2}
\pstIParabolaLineInter[PosAngle={80,-100}](0){\p}{2,1}{4,-2}{P}{Q}
% if you don't know focus F or directrix line
\pstIParabolaPolarNode[linecolor=purple!40,PosAngle=-90](0){\p}{P}{Q}{T}
\end{pspicture}
```



```
\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}
\pstIParabola[linecolor=red!40](0){\p}{-1.5}{1.5}
\pstIParabolaFocusNode[linecolor=red!40](0){\p}{F}
\pstLine[linecolor=gray!40,nodesepA=-0.5]{2,1}{4,-2}
\pstIParabolaLineInter[PosAngle={80,-100}](0){\p}{2,1}{4,-2}{P}{Q}
% if you know focus F, but don't know directrix line
\pstIParabolaPolarNode[linecolor=purple!40,PosAngle=-90](0){\p}{F}{P}{Q}{T}
\end{pspicture}
```



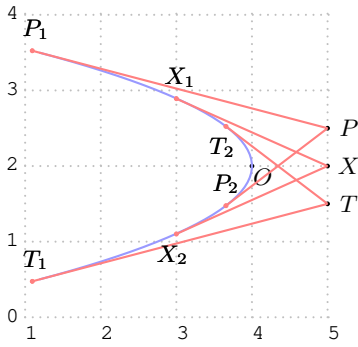
```
\begin{pspicture}[showgrid=true](1,-2)(5,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-130,PointNameSep=0.2](2,0){O}
\pstIParabola[linecolor=red!40](0){\p}{-1.5}{1.5}
\pstIParabolaFocusNode[linecolor=red!40](0){\p}{F}
\pstIParabolaDirectrixLine[linecolor=red!40,nodesepA=-2,nodesepB=-1,PosAngle={180,180}](0){\p}{A}{B}
\pstLine[linecolor=gray!40,nodesepA=-0.5]{2,1}{4,-2}
\pstIParabolaLineInter[PosAngle={80,-100}](0){\p}{2,1}{4,-2}{P}{Q}
% if you know focus F and also directrix line
\pstIParabolaPolarNode[linecolor=purple!40,PosAngle=-90](0){\p}{F}{A}{B}{P}{Q}{T}
\end{pspicture}
```

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

```
\pstIParabolaTangentNode [Options] (O){p}{T}{A}{B}
```

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





```

\begin{pspicture}[showgrid=true](1,0)(5,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-45,PointNameSep=0.2](4,2){O}
\pstIParabola[linecolor=blue!40](O){-\p}{-1.5}{1.5}
\pstGeonode[PosAngle=0](5,1.5){T}
\pstIParabolaTangentNode[linecolor=red!50,PosAngle={80,-100},
PointName={T_1,T_2}](O){-\p}{T}{T1}{T2}
\pstGeonode[PosAngle=0](5,2.5){P}
\pstIParabolaTangentNode[linecolor=red!50,PosAngle={80,90},
PointName={P_1,P_2}](O){-\p}{P}{P1}{P2}
\pstGeonode[PosAngle=0](5,2){X}
\pstIParabolaTangentNode[linecolor=red!50,PosAngle={80,-100},
PointName={X_1,X_2}](O){-\p}{X}{X1}{X2}
\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

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

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:

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

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

```

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

```

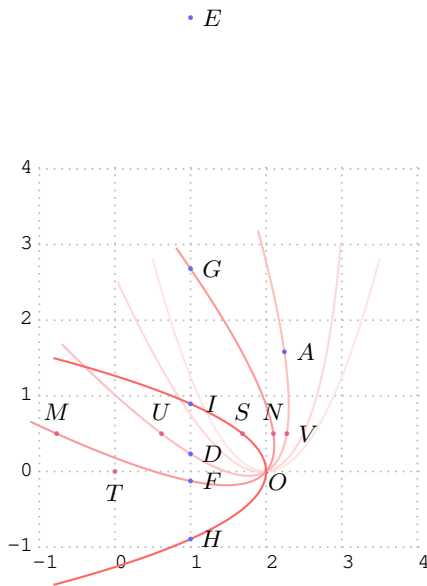
The macro `\pstGeneralParabolaNode` is used to draw a node whose parameter is the given value  $t$  on parabola, please refer to equation (11). The macro `\pstGeneralParabolaAbsNode` is used to draw a node whose abscissa is the given value  $x_1$  on parabola. The macro `\pstGeneralParabolaOrdNode` is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

Note that `\pstGeneralParabolaAbsNode` and `\pstGeneralParabolaOrdNode` both create two nodes  $A$  and  $B$  at most time.

```

\pstGeneralParabolaNode [Options] (O){p} [\theta] {t}{A}
\pstGeneralParabolaAbsNode [Options] (O){p} [\theta] {x_1}{A}{B}
\pstGeneralParabolaOrdNode [Options] (O){p} [\theta] {y_1}{A}{B}

```



```

\begin{pspicture}[showgrid=true](-1,-1)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-40,PointNameSep=0.2](2,0){O}
\pstGeneralParabola[linecolor=red!10](O){\p}[0]{-1.5}{1.5}
\pstGeneralParabola[linecolor=red!15](O){\p}[10]{-1.5}{1.5}
\pstGeneralParabola[linecolor=red!25](O){\p}[30]{-1.5}{1.5}
\pstGeneralParabola[linecolor=red!40](O){\p}[50]{-1.5}{1.5}
\pstGeneralParabola[linecolor=red!60](O){\p}[90]{-1.5}{1.5}
\pstGeneralParabolaNode[PosAngle=0,linecolor=blue!60](O){\p}
}{30}{1.0}{A}
\pstGeneralParabolaAbsNode[PosAngle={0,0},linecolor=blue!60](O){\p}
}[30]{1.0}{D}{E}
\pstGeneralParabolaAbsNode[PosAngle={0,0},linecolor=blue!60](O){\p}
}[50]{1.0}{F}{G}
\pstGeneralParabolaAbsNode[PosAngle={0,0},linecolor=blue!60](O){\p}
}[90]{1.0}{H}{I}
\pstGeneralParabolaOrdNode[PosAngle={90,0},linecolor=purple!60](O)
){\p}[30]{0.5}{U}{V}
\pstGeneralParabolaOrdNode[PosAngle={90,90},linecolor=purple!60](O)
){\p}[50]{0.5}{M}{N}
\pstGeneralParabolaOrdNode[PosAngle={90,-90},linecolor=purple!60](O)
){\p}[90]{0.5}{S}{T}
\end{pspicture}

```

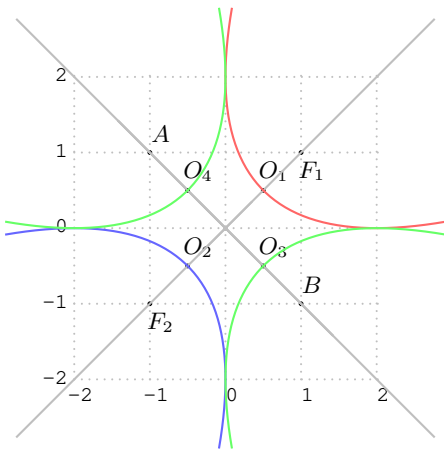
The Macro `\pstGeneralParabolaFl` is used to define a General Parabola with Focus  $F$ , and the directrix line  $l$ . It just calculate the vertex  $O$ , half focal chord  $p$ , and the rotation angle  $\theta$  of the symmetrical axis, then you can pass them into macro `\pstGeneralParabola` to draw this parabola.

```

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

```

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 `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 `\pstGeneralParabola`, PostScript will lookup the value of this key in current dictionary.



```

\begin{pspicture}[showgrid=true](-2,-2)(2,2)
\psset{unit=1.0cm}\psset{dotscale=0.5}\footnotesize
\psset{CodeFig=true,CodeFigColor=gray!50}\psset{PointSymbol=*}
\pstGeonode[PosAngle=-60](1,1){F_1}
\pstGeonode[PosAngle=-60](-1,-1){F_2}
\pstGeonode[PosAngle=60](1,-1){B}
\pstGeonode[PosAngle=60](-1,1){A}
\pstGeneralParabolaFl[PosAngle=60]{F_1}{A}{B}{0_1}{
semifocalchordp1}{SymAxisRotAngle1}
\pstGeneralParabola[linecolor=red!60](0_1){semifocalchordp1}[
SymAxisRotAngle1]{-2}{2}
\pstGeneralParabolaFl[PosAngle=60]{F_2}{A}{B}{0_2}{
semifocalchordp2}{SymAxisRotAngle2}
\pstGeneralParabola[linecolor=blue!60](0_2){semifocalchordp2}[
SymAxisRotAngle2]{-2}{2}
\pstGeneralParabolaFl[PosAngle=60]{B}{F_1}{F_2}{0_3}{
semifocalchordp3}{SymAxisRotAngle3}
\pstGeneralParabola[linecolor=green!60](0_3){semifocalchordp3}[
SymAxisRotAngle3]{-2}{2}
\pstGeneralParabolaFl[PosAngle=60]{A}{F_1}{F_2}{0_4}{
semifocalchordp4}{SymAxisRotAngle4}
\pstGeneralParabola[linecolor=green!60](0_4){semifocalchordp4}[
SymAxisRotAngle4]{-2}{2}
\end{pspicture}

```

The Macro `\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 `\pstGeneralParabola` to draw this parabola. 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.

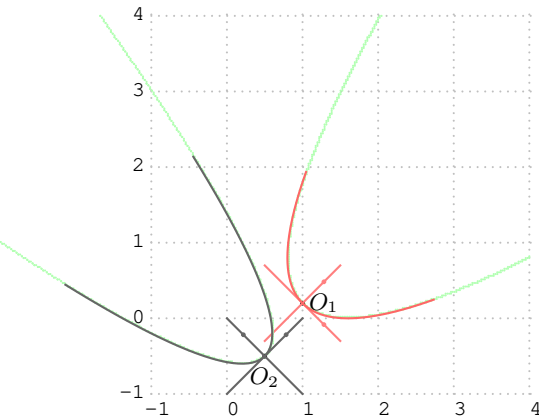
```

\pstGeneralParabolaCoef [Options] {a,b,c,d,e,f}{O}{p}{\theta}

```

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

In the following example, we use `\psplotImp` to draw the same parabolas, just to check the results given by macros `\pstGeneralParabolaCoef` are correct.



```

\begin{pspicture}[showgrid=true](-1,-1)(4,4)
\psset{unit=0.40cm}\footnotesize\psset{dotscale=0.5}
\psset{CodeFig=true}\psset{PointSymbol=*}
% $x^2-2xy+y^2-8x+16=0$ 
\psplotImp[linecolor=green!30](-10,-10)(10,10){ 1 x dup mul mul
-2 x mul y mul add 1 y dup mul mul add -8 x mul add 0 y mul
add 16 add }
\pstGeneralParabolaCoef[PosAngle=0,CodeFigColor=red
!50]{1,-2,1,-8,0,16}{O_1}{P1}{SymAxisRotAngle1}
\pstGeneralParabola[linecolor=red!60](O_1){P1}[SymAxisRotAngle
1]{-3}{3}
% $x^2+2xy+y^2+2x-2y-5=0$ 
\psplotImp[linecolor=green!30](-10,-10)(10,10){ 1 x dup mul mul 2
x mul y mul add 1 y dup mul mul add 2 x mul add -2 y mul
add -5 add }
\pstGeneralParabolaCoef[PosAngle=-90,CodeFigColor=black
!60]{1,2,1,2,-2,-5}{O_2}{P2}{SymAxisRotAngle2}
\pstGeneralParabola[linecolor=black!60](O_2){P2}[SymAxisRotAngle
2]{-3}{3}
\end{pspicture}

```

The Macro `\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 `\pstGeneralParabola` to draw this parabola.

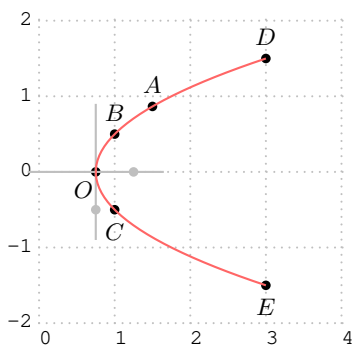
```

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

```

The output parameter  $0$ ,  $p$  and  $\theta$  are same with `\pstGeneralParabolaFl`. They are set to zero if the points are invalid to construct a parabola. If you pass the zero  $p$  into macro `\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{pspicture}[showgrid=true](0,-2)(4,2)
\psset{unit=0.5cm}\footnotesize\psset{PointSymbol=*}
\psset{CodeFig=true,CodeFigColor=gray!50}
% five points from  $y^2-2x+3=0$ 
\pstGeonode[PosAngle=90](3,1.73205){A}
\pstGeonode[PosAngle=90](2,1){B}
\pstGeonode[PosAngle=-90](2,-1){C}
\pstGeonode[PosAngle=90](6,3){D}
\pstGeonode[PosAngle=-90](6,-3){E}
\pstGeneralParabolaABCDE[PosAngle=235]{A}{B}{C}{D}{E}{O}{P}{
SymAxisRotAngle}
\pstGeneralParabola[linecolor=red!60](O){P}[SymAxisRotAngle
]{-3}{3}
\end{pspicture}

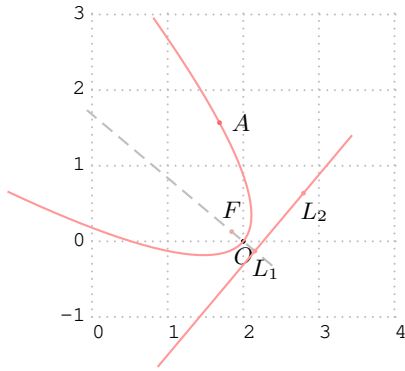
```

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

```

\pstGeneralParabolaFocusNode [Options] (O){p}{\theta}{F}
\pstGeneralParabolaDirectrixLine [Options] (O){p}{\theta}{L_x}{L_y}

```



```

\begin{pspicture}[showgrid=true](0,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,0){O}
\pstGeneralParabola[linecolor=red!40](0){\p}[50]{-1.5}{1.5}
\pstGeneralParabolaFocusNode[linecolor=red!40,PosAngle=90](0){\p}
}[50]{F}
\pstLineAB[linestyle=dashed,linecolor=black!25,nodesepA=-0.5,
nodesepB=-2.5]{O}{F}
\pstGeneralParabolaDirectrixLine[linecolor=red!40,nodesepA=-2,
nodesepB=-1,PosAngle={-60,-60},PointName={L_1,L_2}](0){\p}
}[50]{L1}{L2}
\pstGeneralParabolaNode[linecolor=red!60](0){\p}[50]{1.0}{A}
\end{pspicture}

```

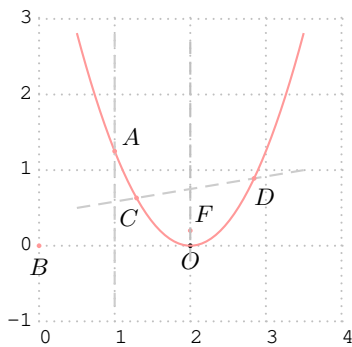
The macro `\pstGeneralParabolaLineInter` is used to find the intersections  $C$  and  $D$  of the parabola and the given line  $AB$ .

```

\pstGeneralParabolaLineInter [Options] (O){p} [θ] {A}{B}{C}{D}

```

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

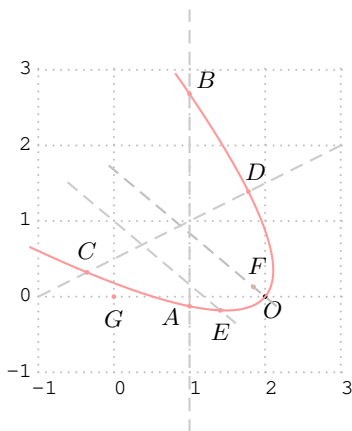


```

\begin{pspicture}[showgrid=true](0,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,0){O}
\pstGeneralParabola[linecolor=red!40](0){\p}[0]{-1.5}{1.5}
\pstGeneralParabolaFocusNode[linecolor=red!40,PosAngle=50](0){\p}
}[0]{F}
\pstLineAB[linestyle=dashed,linecolor=black!25,nodesepA=-0.2,
nodesepB=-2.5]{O}{F}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep
=-0.8]{1,0}{1,2}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle={40,-90}](
0){\p}[0]{1,0}{1,2}{A}{B}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep
=0]{0.5,0.5}{3.5,1}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle
=-110,-60](0){\p}[0]{0.5,0.5}{3.5,1}{C}{D}
\end{pspicture}

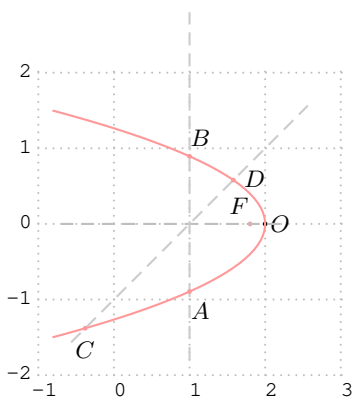
```

Here is the intersections of a real General Parabola with any kind of lines:



```
\begin{pspicture}[showgrid=true](-1,-1)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-60,PointNameSep=0.2](2,0){O}
\pstGeneralParabola[linecolor=red!40](0){\p}[50]{-1.5}{1.5}
\pstGeneralParabolaFocusNode[linecolor=red!40,PosAngle=80](0){\p}
[50]{F}
\pstLineAB[linestyle=dashed,linecolor=black!25,nodesepA=-0.2,
nodesepB=-2.5]{O}{F}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep
=-0.8]{1,-1}{1,3}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle
={-150,40}](0){\p}[50]{1,-1}{1,3}{A}{B}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep
=0.0]{-1,0}{3,2}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle={90,70}](0)
{\p}[50]{-1,0}{3,2}{C}{D}
% a line with gradient k=-\cos50/\sin50 parallel to OF
\pstLineAS[linestyle=dashed,linecolor=gray!40,nodesep=-0.8,
PointName=none,PointSymbol=none](0,1){50 cos 50 sin div neg}
{X}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle
={-90,-90}](0){\p}[50]{0,1}{X}{E}{G}
\end{pspicture}
```

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

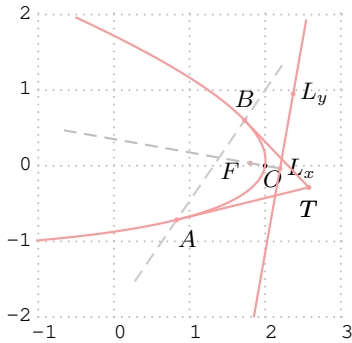


```
\begin{pspicture}[showgrid=true](-1,-2)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=0,PointNameSep=0.2](2,0){O}
\pstGeneralParabola[linecolor=red!40](0){\p}[90]{-1.5}{1.5}
\pstGeneralParabolaFocusNode[linecolor=red!40,PosAngle=120](0){\p}
[90]{F}
\pstLineAB[linestyle=dashed,linecolor=black!25,nodesepA=-0.2,
nodesepB=-2.5]{O}{F}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep
=-0.8]{1,-1}{1,2}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle={-60,60}](0)
{\p}[90]{1,-1}{1,2}{A}{B}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep
=-0.8]{0,-1}{2,1}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle={-90,5}](0)
{\p}[90]{0,-1}{2,1}{C}{D}
\end{pspicture}
```

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

```
\pstGeneralParabolaPolarNode [Options] (O){p} [\theta] {A}{B}{T}
\pstGeneralParabolaPolarNode [Options] (O){p} [\theta] (F){A}{B}{T}
\pstGeneralParabolaPolarNode [Options] (O){p} [\theta] (F) [Lx] [Ly] {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.

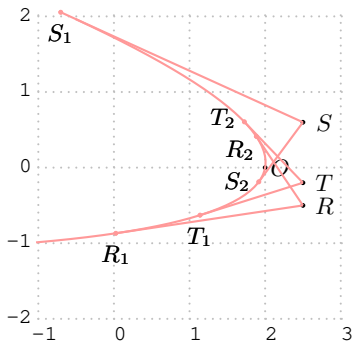


```
\begin{pspicture}[showgrid=true](-1,-2)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-60,PointNameSep=0.2](2,0){O}
\pstGeneralParabola[linecolor=red!40](0){\p}[80]{-1.5}{1.5}
\pstGeneralParabolaFocusNode[linecolor=red!40,PosAngle=200](0){\p}
}{80}{F}
\pstGeneralParabolaDirectrixLine[linecolor=red!40,nodesepA=-2,
nodesepB=-1,PosAngle={0,0},PointName={L_x,L_y}](0){\p}[80]{
L_x}{L_y}
\pstLine[linestyle=dashed,linecolor=black!25,nodesepA=-0.2,
nodesepB=-2.5]{O}{F}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep
=-0.4]{O.5,-1.2}{2,1}
\pstGeneralParabolaLineInter[linecolor=red!40,PosAngle={-60,90}](
0){\p}[80]{0.5,-1.2}{2,1}{A}{B}
%\pstGeneralParabolaPolarNode[linecolor=red!40,PosAngle=-90](0){\
p}[80]{A}{B}{T}
%\pstGeneralParabolaPolarNode[linecolor=red!40,PosAngle=-90](0){\
p}[80]{F}{A}{B}{T}
\pstGeneralParabolaPolarNode[linecolor=red!40,PosAngle=-90](0){\p}
}[80]{F}[L_x][L_y]{A}{B}{T}
\end{pspicture}
```

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

```
\pstGeneralParabolaTangentNode [Options] (O){p} [\theta] {T}{A}{B}
```

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



```
\begin{pspicture}[showgrid=true](-1,-2)(3,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=0,PointNameSep=0.2](2,0){O}
\pstGeneralParabola[linecolor=red!40](0){\p}[80]{-1.5}{1.5}
\pstGeonode[PosAngle=0](2.5,-0.5){R}(2.5,-0.2){T}(2.5,0.6){S}
\pstGeneralParabolaTangentNode[linecolor=red!40,PosAngle
={-90,220},PointName={R_1,R_2}](0){\p}[80]{R}{R1}{R2}
\pstGeneralParabolaTangentNode[linecolor=red!40,PosAngle
={-90,170},PointName={T_1,T_2}](0){\p}[80]{T}{T1}{T2}
\pstGeneralParabolaTangentNode[linecolor=red!40,PosAngle
={-90,180},PointName={S_1,S_2}](0){\p}[80]{S}{S1}{S2}
\end{pspicture}
```

### 3.6. General Inversion Parabola

The General Inversion 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 inversion parabola (9), using the rotation transform formula (3), then we have

$$\begin{cases} x' = (\frac{t^2}{2p} + x_0) \cos \theta - (t + y_0) \sin \theta = x'_0 - t \sin \theta + t^2 \frac{\cos \theta}{2p} \\ y' = (\frac{t^2}{2p} + x_0) \sin \theta + (t + y_0) \cos \theta = y'_0 + t \cos \theta + t^2 \frac{\sin \theta}{2p} \end{cases} \quad (12)$$

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 Inversion Parabola with coordinate translation and rotation as following:

$$\begin{cases} x = x_0 - t \sin \theta + t^2 \frac{\cos \theta}{2p} \\ y = y_0 + t \cos \theta + t^2 \frac{\sin \theta}{2p} \end{cases} \quad (13)$$

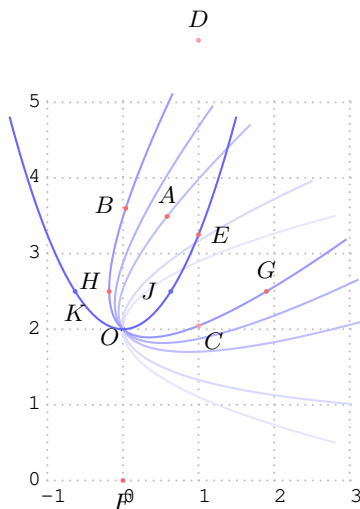
The macro `\pstGeneralIParabola` is used to draw a Standard Inversion Parabola from  $y_1$  to  $y_2$  with Vertex  $O$ , the half of the focus chord axis  $abs(p)$ .

```
\pstGeneralIParabola [Options] (O){p} [\theta] {y1}{y2}
```

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] {x1}{A}{B}
\pstGeneralIParabolaOrdNode [Options] (O){p} [\theta] {y1}{A}{B}
```

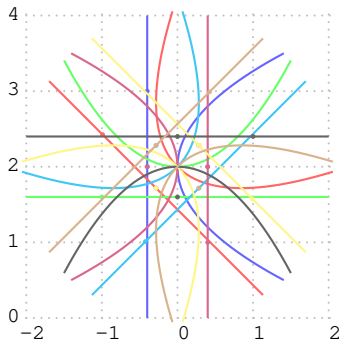


```
\begin{pspicture}[showgrid=true](-1,0)(3,5)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=210,PointNameSep=0.2](0,2){O}
\pstGeneralIParabola[linecolor=blue!10](0){\p}[0]{-1.5}{1.5}
\pstGeneralIParabola[linecolor=blue!15](0){\p}[10]{-1.5}{1.5}
\pstGeneralIParabola[linecolor=blue!25](0){\p}[30]{-1.5}{1.5}
\pstGeneralIParabola[linecolor=blue!30](0){\p}[40]{-1.5}{1.5}
\pstGeneralIParabola[linecolor=blue!40](0){\p}[50]{-1.5}{1.5}
\pstGeneralIParabola[linecolor=blue!60](0){\p}[90]{-1.5}{1.5}
\pstGeneralIParabolaNode[linecolor=red!60,PosAngle=90](0){\p}
{30}{1.0}{A}
\pstGeneralIParabolaNode[linecolor=red!60,PosAngle=170](0){\p}
{50}{1.0}{B}
\pstGeneralIParabolaAbsNode[linecolor=red!40,PosAngle={-45,90}](0)
{\p}[50]{1.0}{C}{D}
\pstGeneralIParabolaAbsNode[linecolor=red!60,PosAngle={0,-90}](0)
{\p}[90]{1.0}{E}{F}
\pstGeneralIParabolaOrdNode[linecolor=red!60,PosAngle={90,150}](0)
{\p}[50]{2.5}{G}{H}
\pstGeneralIParabolaOrdNode[linecolor=blue!60,PosAngle
={180,-90}](0){\p}[90]{2.5}{J}{K}
\end{pspicture}
```

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] {Lx}{Ly}
```





```

\begin{pspicture}[showgrid=true](-2,0)(2,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\psset{PointName=none,nodesepA=-2,nodesepB=-1}
\pstGeonode(0,2){O}\def\p{0.8}
\psset{linecolor=blue!60}
\pstGeneralIParabola(0){\p}[0]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[0]{A}
\pstGeneralIParabolaDirectrixLine(0){\p}[0]{A1}{A2}
\psset{linecolor=red!60}
\pstGeneralIParabola(0){\p}[45]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[45]{B}
\pstGeneralIParabolaDirectrixLine(0){\p}[45]{B1}{B2}
\psset{linecolor=green!60}
\pstGeneralIParabola(0){\p}[90]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[90]{C}
\pstGeneralIParabolaDirectrixLine(0){\p}[90]{C1}{C2}
\psset{linecolor=cyan!60}
\pstGeneralIParabola(0){\p}[135]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[135]{D}
\pstGeneralIParabolaDirectrixLine(0){\p}[135]{D1}{D2}
\psset{linecolor=purple!60}
\pstGeneralIParabola(0){\p}[180]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[180]{E}
\pstGeneralIParabolaDirectrixLine(0){\p}[180]{E1}{E2}
\psset{linecolor=yellow!60}
\pstGeneralIParabola(0){\p}[225]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[225]{F}
\pstGeneralIParabolaDirectrixLine(0){\p}[225]{F1}{F2}
\psset{linecolor=black!60}
\pstGeneralIParabola(0){\p}[270]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[270]{G}
\pstGeneralIParabolaDirectrixLine(0){\p}[270]{G1}{G2}
\psset{linecolor=brown!60}
\pstGeneralIParabola(0){\p}[315]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode(0){\p}[315]{H}
\pstGeneralIParabolaDirectrixLine(0){\p}[315]{H1}{H2}
\end{pspicture}

```

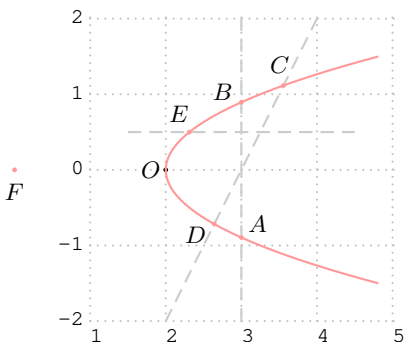
The macro `\pstGeneralIParabolaLineInter` is used to find the intersections  $C$  and  $D$  of the parabola and the given line  $AB$ .

```

\pstGeneralIParabolaLineInter [Options] (O){p} [θ] {A}{B}{C}{D}

```

When  $\theta = 0$ , the intersections with any kind of lines:

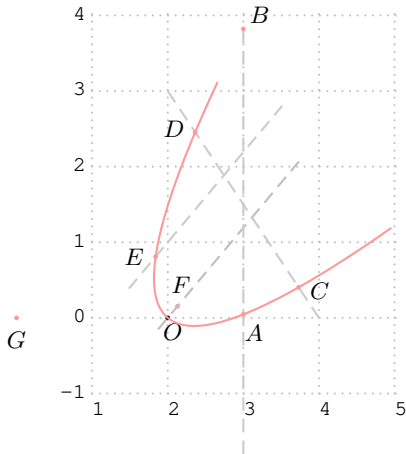


```

\begin{pspicture}[showgrid=true](1,-2)(5,2)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=180,PointNameSep=0.2](2,0){O}
\pstGeneralIParabola[linecolor=red!40](0){\p}[0]{-1.5}{1.5}
\pstLine[linecolor=gray!40,linestyle=dashed]{3,-2}{3,2}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle
={40,150}](0){\p}[0]{3,-2}{3,2}{A}{B}
\pstLine[linecolor=gray!40,linestyle=dashed]{2,-2}{4,2}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle
={100,210}](0){\p}[0]{2,-2}{4,2}{C}{D}
\pstLine[linecolor=gray!40,linestyle=dashed]{1.5,0.5}{4.5,0.5}
\pstGeneralIParabolaLineInter[linecolor=red!40,PosAngle
={120,-90}](0){\p}[0]{1.5,0.5}{4.5,0.5}{E}{F}
\end{pspicture}

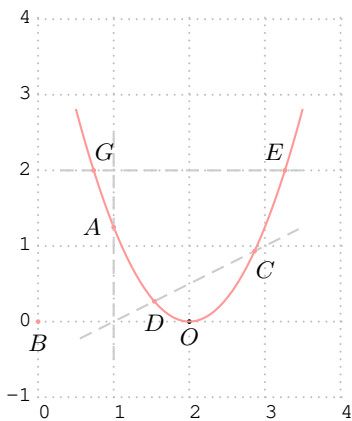
```

When  $\theta = 50$ , the intersections with any kind of lines:



```
\begin{pspicture}[showgrid=true](1,-1)(5,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-70,PointNameSep=0.2](2,0){O}
\pstGeneralIParabola[linecolor=red!40](0){\p}[50]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode[linecolor=red!40,PosAngle=80](0){\p}
{50}{F}
\pstLineAB[linestyle=dashed,linecolor=black!25,nodesepA=-0.2,
nodesepB=-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}
% 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}
```

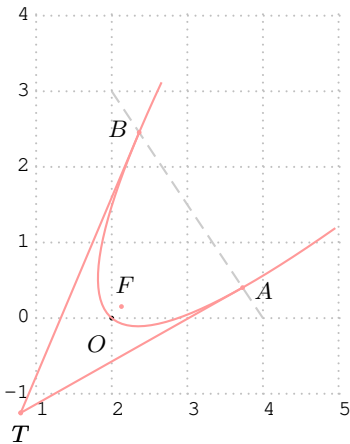
When  $\theta = 90$ , the intersections with any kind of lines:



```
\begin{pspicture}[showgrid=true](0,-1)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,0){O}
\pstGeneralIParabola[linecolor=red!40](0){\p}[90]{-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 `\pstGeneralIParabolaPolarNode` is used to find the polar point  $T$  of chord  $AB$  on Parabola  $P$ .

```
\pstGeneralIParabolaPolarNode [Options] (O){p} [\theta] {A}{B}{T}
\pstGeneralIParabolaPolarNode [Options] (O){p} [\theta] (F){A}{B}{T}
\pstGeneralIParabolaPolarNode [Options] (O){p} [\theta] (F) [L_x] [L_y] {A}{B}{T}
```

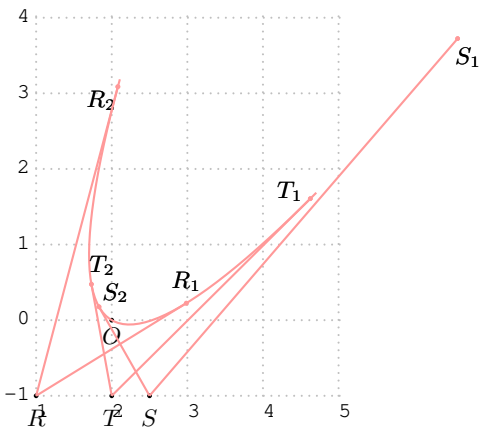


```
\begin{pspicture}[showgrid=true](1,-1)(5,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=240,PointNameSep=0.4](2,0){O}
\pstGeneralIParabola[linecolor=red!40](0){\p}[50]{-1.5}{1.5}
\pstGeneralIParabolaFocusNode[linecolor=red!40,PosAngle=80](0){\p}
[50]{F}
\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}{A}{B}
\pstGeneralIParabolaPolarNode[linecolor=red!40,PosAngle=-90](0){\p}[50]{F}{A}{B}{T}
\end{pspicture}
```

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 `\pstGeneralIParabolaTangentNode` is used to find the two nodes  $A$  and  $B$  on the Parabola through the point  $T$ .

```
\pstGeneralIParabolaTangentNode [Options] (O){p} [θ] {T}{A}{B}
```



```
\begin{pspicture}[showgrid=true](1,-1)(5,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\p{0.4}
\pstGeonode[PosAngle=-90,PointNameSep=0.2](2,0){O}
\pstGeneralIParabola[linecolor=red!40](0){\p}[60]{-1.5}{1.5}
\pstGeonode[PosAngle=-90](1,-1){R}(2,-1){T}(2.5,-1){S}
\pstGeneralIParabolaTangentNode[linecolor=red!40,PosAngle={90,220},PointName={R_1,R_2}](0){\p}[60]{R}{R1}{R2}
\pstGeneralIParabolaTangentNode[linecolor=red!40,PosAngle={160,60},PointName={T_1,T_2}](0){\p}[60]{T}{T1}{T2}
\pstGeneralIParabolaTangentNode[linecolor=red!40,PosAngle={-60,40},PointName={S_1,S_2}](0){\p}[60]{S}{S1}{S2}
\end{pspicture}
```

### 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:

$$\frac{(x - x_0)^2}{a^2} - \frac{(y - y_0)^2}{b^2} = 1 \tag{14}$$

and the parametric function can be written as:

$$\begin{cases} x = a \sec \alpha + x_0 \\ y = b \tan \alpha + y_0 \end{cases} \tag{15}$$

The macro `\pstHyperbola` 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 `angleX` is used to truncate the width of the figure, it should be setup from 0 to 90.

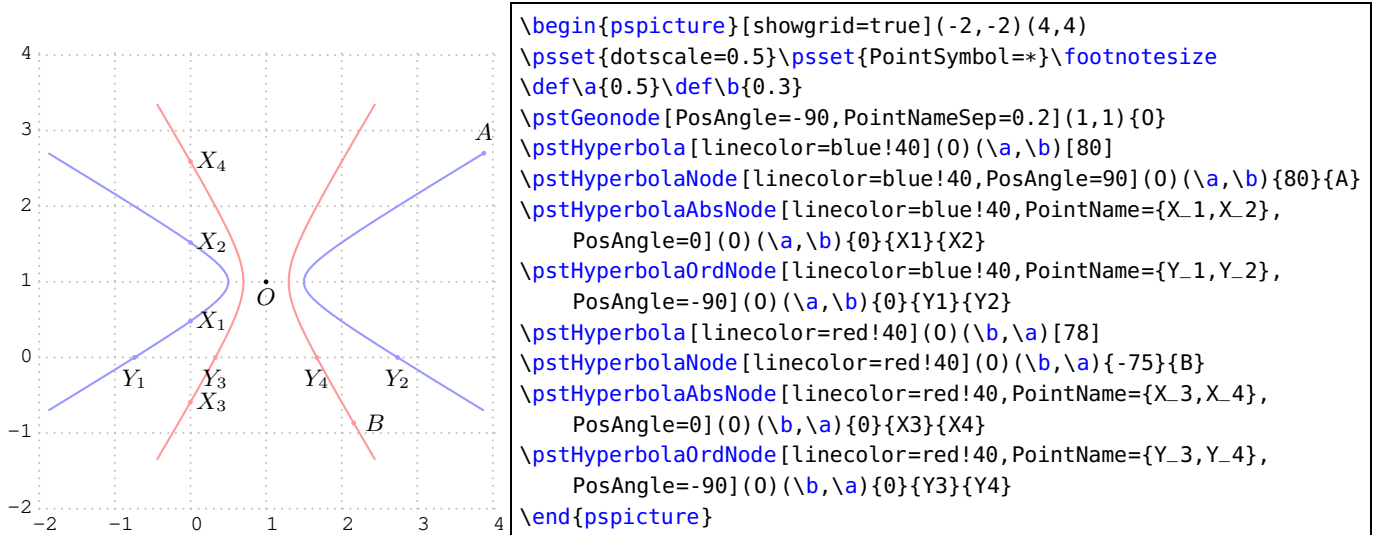
```
\pstHyperbola [Options] (O)(a,b) [angleX]
```

The macro `\pstHyperbolaNode` is used to draw a node whose parameter is the given value  $t$  on Hyperbola, please refer to equation (15). The macro `\pstHyperbolaAbsNode` is used to draw the

nodes whose abscissa are the given value  $x_1$  on Hyperbola. The macro `\pstHyperbolaOrdNode` is used to draw the nodes whose ordinate are the given value  $y_1$  on Hyperbola.

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

```
\pstHyperbolaNode [Options] (O)(a,b){t}{A}
\pstHyperbolaAbsNode [Options] (O)(a,b){x_1}{A}{B}
\pstHyperbolaOrdNode [Options] (O)(a,b){y_1}{A}{B}
```

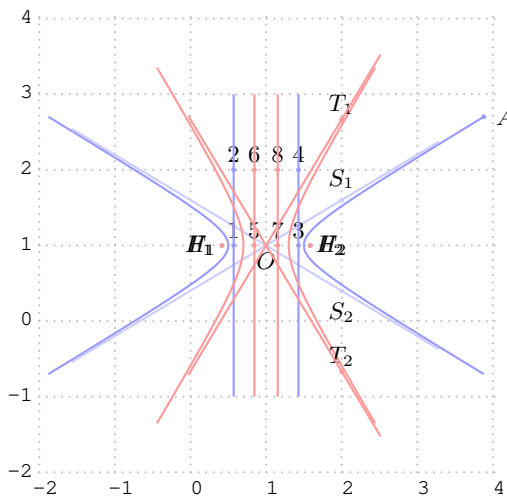


The macro `\pstHyperbolaFocusNode` is used to find the focus nodes of the Hyperbola, and the macro `\pstHyperbolaDirectrixLine` is used to find the directrix lines of the Hyperbola.

```
\pstHyperbolaFocusNode [Options] (O)(a,b){F_1}{F_2}
\pstHyperbolaDirectrixLine [Options] (O)(a,b){L_x}{L_y}{R_x}{R_y}
```

Note that you can use `\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 `\pstHyperbolaAsymptoteLine` to get them, this macro only create one node on each asymptote line, as the other one is the center of the hyperbola.

```
\pstHyperbolaAsymptoteLine [Options] (O)(a,b){L_1}{L_2}
```



```

\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](1,1){O}
\pstHyperbola[linecolor=blue!40](0)(\A,\B)[80]
\pstHyperbolaNode[linecolor=blue!40](0)(\A,\B){80}{A}
\pstLineAS[PointName=S_1,PosAngle=90,nodesepA=-3,nodesepB=-1.5,
linecolor=blue!20]{0}{\B\space \A\space div}{S1}
\pstLineAS[PointName=S_2,PosAngle=-90,nodesepA=-3,nodesepB=-1.5,
linecolor=blue!20]{0}{\B\space \A\space div neg}{S2}
\pstHyperbolaFocusNode[linecolor=blue!40,PointName={F_1,F_2},
PosAngle={180,0}](0)(\A,\B){F1}{F2}
\pstHyperbolaDirectrixLine[linecolor=blue!40,nodesepA=-2,nodesepB
=-1,PointName={1,2,3,4},PosAngle=90,PointNameSep=0.2](0)(\A
,\B){Lx}{Ly}{Rx}{Ry}
\pstHyperbola[linecolor=red!40](0)(\B,\A)[78]
\pstHyperbolaFocusNode[linecolor=red!40,PointName={H_1,H_2},
PosAngle={180,0}](0)(\B,\A){H1}{H2}
\pstHyperbolaDirectrixLine[linecolor=red!40,nodesepA=-2,nodesepB
=-1,PointName={5,6,7,8},PosAngle=90,PointNameSep=0.2](0)(\B
,\A){Mx}{My}{Nx}{Ny}
\pstHyperbolaAsymptoteLine[linecolor=red!40,nodesepA=-2,nodesepB
=-1,PointName={T_1,T_2},PosAngle=90,PointNameSep=0.2](0)(\B
,\A){T1}{T2}
\end{pspicture}

```

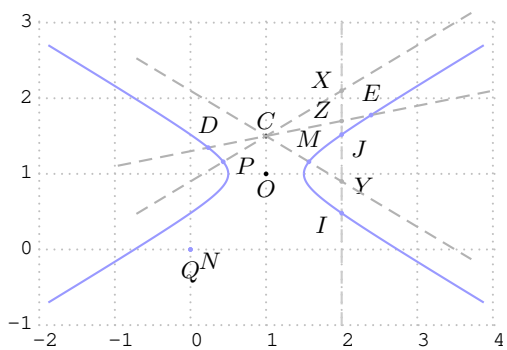
The macro `\pstHyperbolaLineInter` is used to find the intersections  $C$  and  $D$  of the hyperbola and the given line  $AB$ .

```

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

```

In the following example, the Line  $CX$  and  $CY$  are parallel to the asymptote of the 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}
\pstGeonode[PosAngle={-90,90},PointNameSep=0.2](1,1){O}(1,1.5){C}
\pstHyperbola[linecolor=blue!40](0)(\A,\B)[80]
\pstLine[linestyle=dashed,linecolor=gray!40]{2,-1}{2,3}
\pstHyperbolaLineInter[linecolor=blue!40,PosAngle={210,-40}](0)(\
A,\B){2,-1}{2,3}{I}{J}
\pstLineAS[linestyle=dashed,linecolor=gray!60,nodesep=-2,PosAngle
=150]{1,1.5}{\B\space \A\space div}{X}
\pstLineAS[linestyle=dashed,linecolor=gray!60,nodesep=-2,PosAngle
=-10]{1,1.5}{\B\space \A\space div neg}{Y}
\pstLineAS[linestyle=dashed,linecolor=gray!60,nodesep=-2,PosAngle
=150]{1,1.5}{0.2}{Z}
\pstHyperbolaLineInter[linecolor=blue!40,PosAngle={-10,-90}](0)(\
A,\B){1,1.5}{X}{P}{Q}
\pstHyperbolaLineInter[linecolor=blue!40,PosAngle={90,-30}](0)(\A
,\B){1,1.5}{Y}{M}{N}
\pstHyperbolaLineInter[linecolor=blue!40,PosAngle={90,90}](0)(\A
,\B){1,1.5}{Z}{D}{E}
\end{pspicture}

```

The macro `\pstHyperbolaPolarNode` is used to find the polar point  $T$  of chord  $AB$  on the hyperbola.

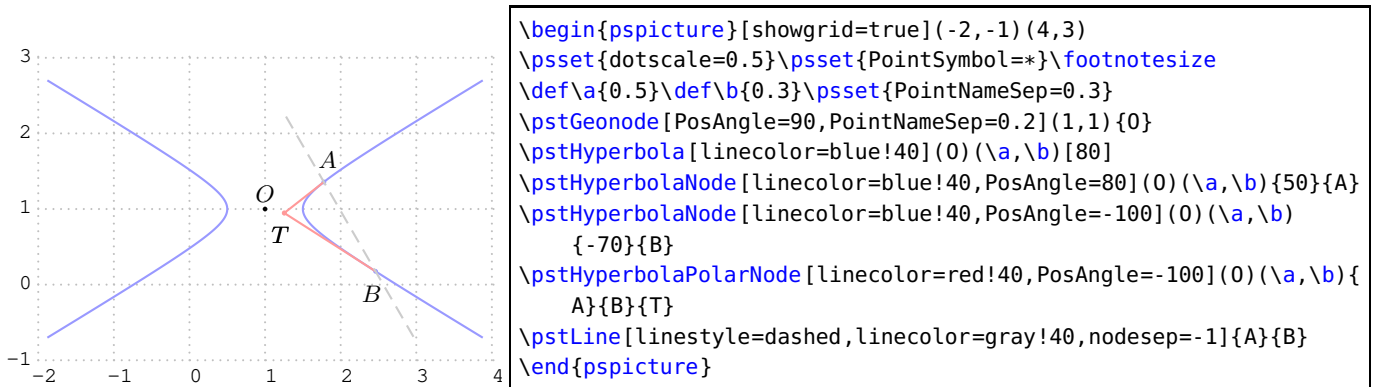
```

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

```

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  $AB$ .

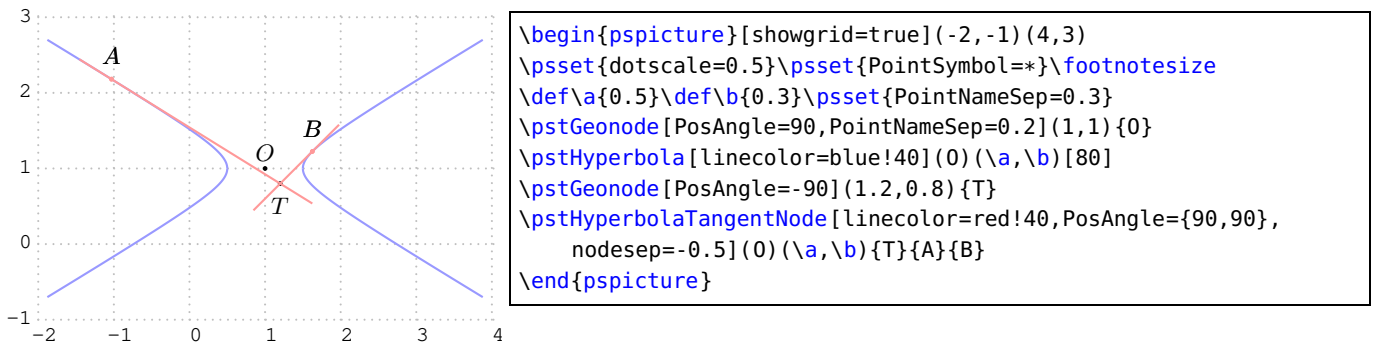


The macro `\pstHyperbolaTangentNode` is used to find the tangent point  $A$  and  $B$  of point  $T$  outside of the hyperbola.

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

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  $T$ .



### 3.8. Standard Inversion Hyperbola

The Standard Inversion 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:

$$\frac{(y - y_0)^2}{a^2} - \frac{(x - x_0)^2}{b^2} = 1 \quad (16)$$

and the parametric function can be written as:

$$\begin{cases} x = b \tan \alpha + x_0 \\ y = a \sec \alpha + y_0 \end{cases} \quad (17)$$

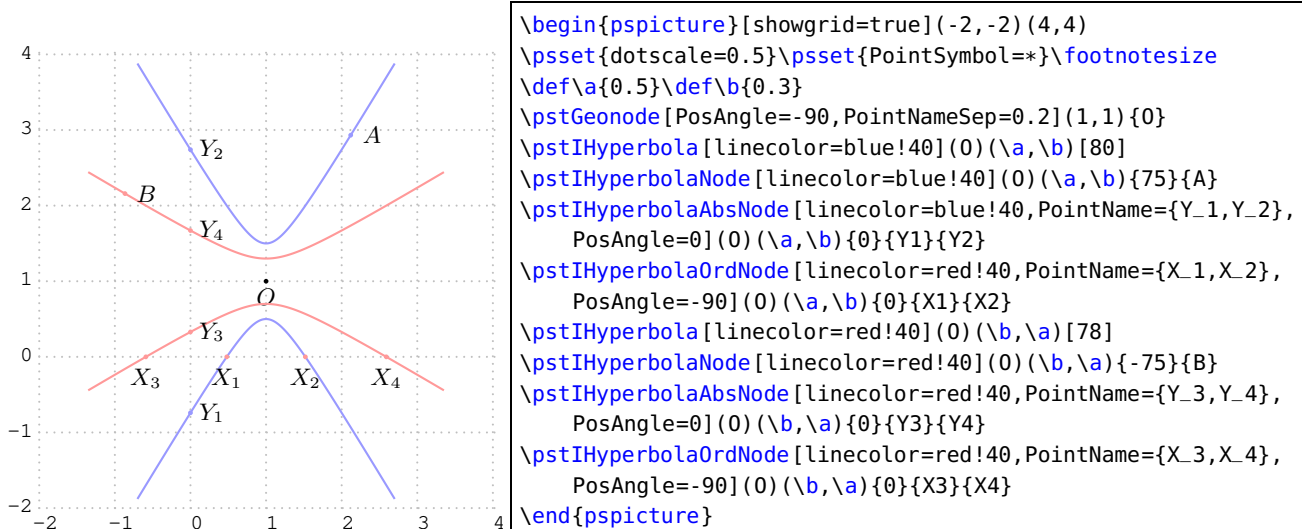
The macro `\pstIHyperbola` is used to draw a Standard Inversion Hyperbola with Center  $O$ , the half of the real axis  $a$ , the half of the imaginary axis  $b$ . The parameter `angleY` is used to truncate the height of the figure, it should be setup from 0 to 90.

```
\pstIHyperbola [Options] (O)(a, b) [angleY]
```

The macro `\pstIHyperbolaNode` is used to draw a node whose parameter is the given value  $t$  on Inversion Hyperbola, please refer to equation (17). The macro `\pstIHyperbolaAbsNode` is used to draw the nodes whose abscissa are the given value  $x_1$  on Inversion Hyperbola. The macro `\pstIHyperbolaOrdNode` is used to draw the nodes whose ordinate are the given value  $y_1$  on Inversion Hyperbola.

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

```
\pstIHyperbolaNode [Options] (O)(a, b){t}{A}
\pstIHyperbolaAbsNode [Options] (O)(a, b){x_1}{A}{B}
\pstIHyperbolaOrdNode [Options] (O)(a, b){y_1}{A}{B}
```

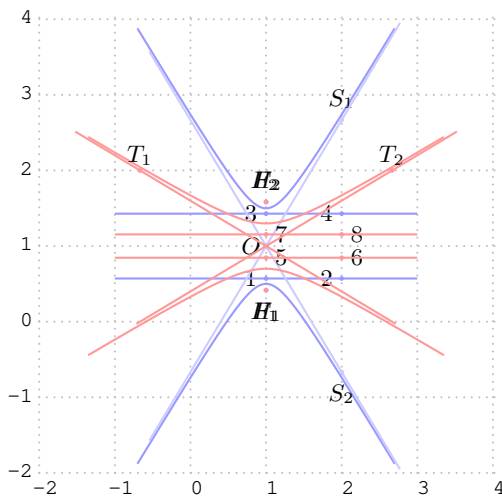


The macro `\pstIHyperbolaFocusNode` is used to find the focus nodes of the Inversion Hyperbola, and the macro `\pstIHyperbolaDirectrixLine` is used to find the directrix lines of the Inversion Hyperbola.

```
\pstIHyperbolaFocusNode [Options] (O)(a, b){F_1}{F_2}
\pstIHyperbolaDirectrixLine [Options] (O)(a, b){L_x}{L_y}{R_x}{R_y}
```

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

```
\pstHyperbolaAsymptoteLine [Options] (O)(a, b){L_1}{L_2}
```



```

\begin{pspicture}[showgrid=true](-2,-2)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}
\pstGeonode[PosAngle=180,PointNameSep=0.2](1,1){O}
\pstIHyperbola[linecolor=blue!40](0)(\ax,\bx)[80]
\pstIHyperbolaFocusNode[linecolor=blue!40,PointName={F_1,F_2},
  PosAngle={-90,90}](0)(\ax,\bx){F1}{F2}
\pstIHyperbolaDirectrixLine[linecolor=blue!40,nodesepA=-2,
  nodesepB=-1,PointName={1,2,3,4},PosAngle=180,PointNameSep
  =0.2](0)(\ax,\bx){Lx}{Ly}{Rx}{Ry}
\pstLineAS[PointName=S_1,PosAngle=90,nodesepA=-3,nodesepB=-1.5,
  linecolor=blue!20]{0}{\ax\space \bx\space div}{S1}
\pstLineAS[PointName=S_2,PosAngle=-90,nodesepA=-3,nodesepB=-1.5,
  linecolor=blue!20]{0}{\ax\space \bx\space div neg}{S2}
\pstIHyperbola[linecolor=red!40](0)(\bx,\ax)[78]
\pstIHyperbolaFocusNode[linecolor=red!40,PointName={H_1,H_2},
  PosAngle={-90,90}](0)(\bx,\ax){H1}{H2}
\pstIHyperbolaDirectrixLine[linecolor=red!40,nodesepA=-2,nodesepB
  =-1,PointName={5,6,7,8},PosAngle=0,PointNameSep=0.2](0)(\bx,\
  \ax){Mx}{My}{Nx}{Ny}
\pstIHyperbolaAsymptoteLine[linecolor=red!40,nodesepA=-2,nodesepB
  =-1,PointName={T_1,T_2},PosAngle=90,PointNameSep=0.2](0)(\bx
  ,\ax){T1}{T2}
\end{pspicture}

```

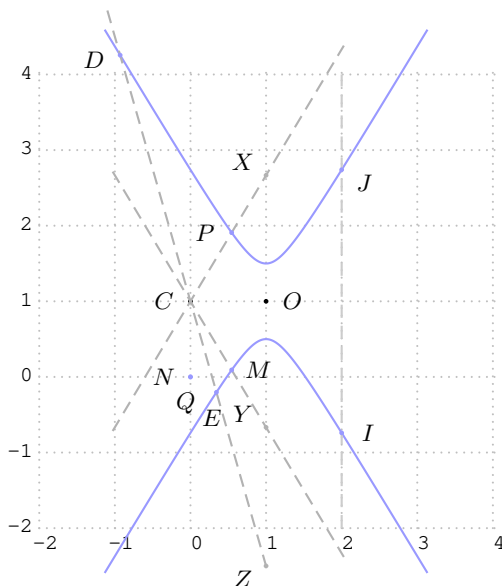
The macro `\pstIHyperbolaLineInter` is used to find the intersections  $C$  and  $D$  of the hyperbola and the given line  $AB$ .

```

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

```

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{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.35}
\pstGeonode[PosAngle={0,180}](1,1){O}(0,1){C}
\pstIHyperbola[linecolor=blue!40](0)(\ax,\bx)[82]
\pstLine[linecolor=gray!40,linestyle=dashed]{2,-2}{2,4}
\pstIHyperbolaLineInter[linecolor=blue!40,PosAngle={0,-30}](0)(\ax
,\bx){2,-2}{2,4}{I}{J}
\pstLineAS[linestyle=dashed,linecolor=gray!60,nodesep=-2,PosAngle
  =150]{0,1}{\ax\space \bx\space div}{X}
\pstLineAS[linestyle=dashed,linecolor=gray!60,nodesep=-2,PosAngle
  =150]{0,1}{\ax\space \bx\space div neg}{Y}
\pstLineAS[linestyle=dashed,linecolor=gray!60,nodesepA=-4,
  PosAngle=210]{0,1}{-3.5}{Z}
\pstIHyperbolaLineInter[linecolor=blue!40,PosAngle={180,-100}](0)
(\ax,\bx){0,1}{X}{P}{Q}
\pstIHyperbolaLineInter[linecolor=blue!40,PosAngle={0,180}](0)(\ax
,\bx){0,1}{Y}{M}{N}
\pstIHyperbolaLineInter[linecolor=blue!40,PosAngle={190,-100}](0)
(\ax,\bx){0,1}{Z}{D}{E}
\end{pspicture}

```

The macro `\pstIHyperbolaPolarNode` is used to find the polar point  $T$  of chord  $AB$  on the hyperbola.

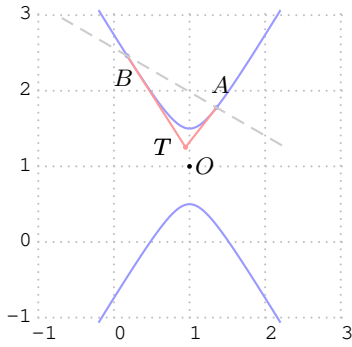
```

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

```

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





```

\begin{pspicture}[showgrid=true](-1,-1)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=0,PointNameSep=0.2](1,1){O}
\pstIHyperbola[linecolor=blue!40](O)(\ax,\bx)[76]
\pstIHyperbolaNode[linecolor=blue!40,PosAngle=80](O)(\ax,\bx){50}{A}
}
\pstIHyperbolaNode[linecolor=blue!40,PosAngle=-100](O)(\ax,\bx)
{-70}{B}
\pstIHyperbolaPolarNode[linecolor=red!40,PosAngle=180](O)(\ax,\bx){
A}{B}{T}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep=-1]{A}{B}
\end{pspicture}

```

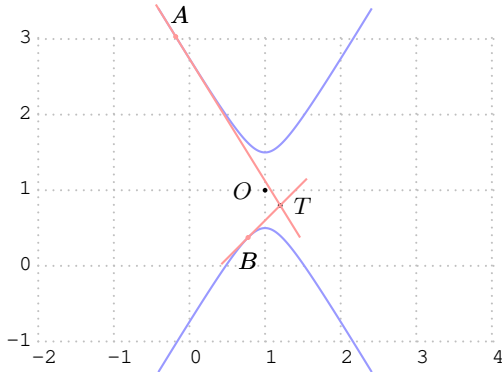
The macro `\pstIHyperbolaTangentNode` is used to find the tangent point  $A$  and  $B$  of point  $T$  outside of the hyperbola.

```

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

```

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



```

\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=180](1,1){O}
\pstIHyperbola[linecolor=blue!40](O)(\ax,\bx)[78]
\pstGeonode[PosAngle=0](1.2,0.8){T}
\pstIHyperbolaTangentNode[linecolor=red!40,PosAngle={80,-90},
nodesep=-0.5](O)(\ax,\bx){T}{A}{B}
\end{pspicture}

```

### 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

$$\begin{cases} x' = (a \sec \alpha + x_0) \cos \theta - (b \tan \alpha + y_0) \sin \theta = x'_0 + a \sec \alpha \cos \theta - b \tan \alpha \sin \theta \\ y' = (a \sec \alpha + x_0) \sin \theta + (b \tan \alpha + y_0) \cos \theta = y'_0 + a \sec \alpha \sin \theta + b \tan \alpha \cos \theta \end{cases} \quad (18)$$

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:

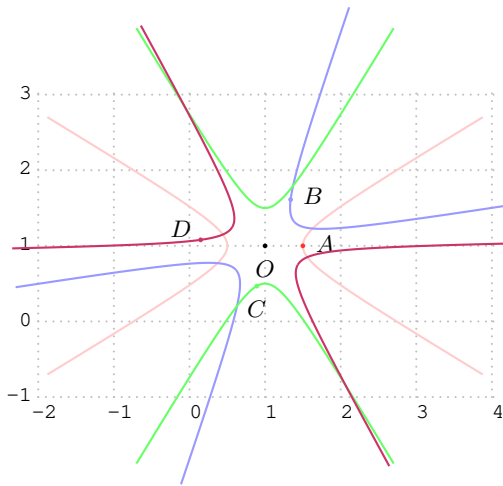
$$\begin{cases} x = x_0 + a \sec \alpha \cos \theta - b \tan \alpha \sin \theta \\ y = y_0 + a \sec \alpha \sin \theta + b \tan \alpha \cos \theta \end{cases} \quad (19)$$

The macro `\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 `angleX` is used to truncate the width of the figure, it should be setup from 0 to 90.

```

\pstGeneralHyperbola [Options] (O)(a,b)[\theta][angleX]

```



```

\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}
\pstGeonode[PosAngle=-90](1,1){O}
\pstGeneralHyperbola[linecolor=red!20](0)(\a,\b)[0][80]
\pstGeneralHyperbolaNode[linecolor=red!80,PosAngle=5](0)(\a,\b)
[0]{0}{A}
\pstGeneralHyperbola[linecolor=blue!40](0)(\a,\b)[40][80]
\pstGeneralHyperbolaNode[linecolor=blue!40,PosAngle=10](0)(\a,\b)
[40]{40}{B}
\pstGeneralHyperbola[linecolor=green!60](0)(\a,\b)[90][80]
\pstGeneralHyperbolaNode[linecolor=green!60,PosAngle=-90](0)(\a,\b)
[90]{200}{C}
\pstGeneralHyperbola[linecolor=purple!80](0)(\a,\b)[150][80]
\pstGeneralHyperbolaNode[linecolor=purple!80,PosAngle=150](0)(\a,\b)
[150]{50}{D}
\end{pspicture}

```

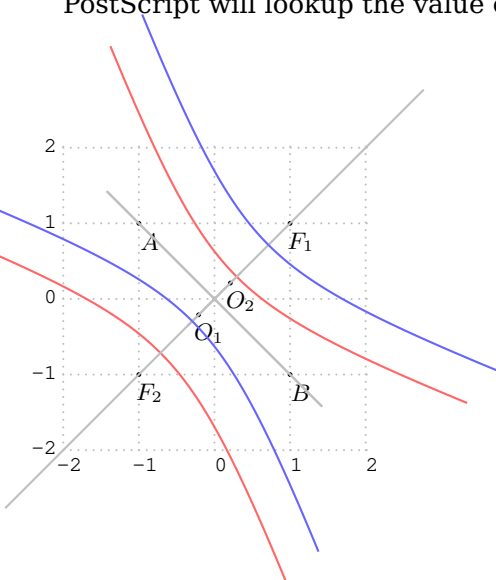
The Macro `\pstGeneralHyperbolaFle` is used to define a General Hyperbola with Focus  $F$ , directrix line  $l$ , and the eccentricity  $e$ , where  $e > 1$ . It just calculate the center  $O$ , real radius  $a$ , imaginary radius  $b$  and the rotation angle  $\theta$  of the real axis, then you can pass them into macro `\pstGeneralHyperbola` to draw this hyperbola.

```

\pstGeneralHyperbolaFle [Options] {F}{A}{B}{e}{O}{Rab}{\theta}

```

The output parameter 0 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 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  $\theta$  is also a PostScript key to store the rotation angle of real axis, when you pass it to `\pstGeneralHyperbola`, PostScript will lookup the value of this key in current dictionary.



```

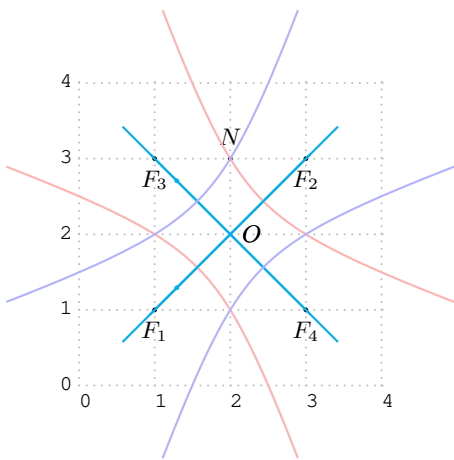
\begin{pspicture}[showgrid=true](-2,-2)(2,2)
\psset{unit=1.0cm}\psset{dotscale=0.5}\footnotesize
\psset{CodeFig=true,CodeFigColor=gray!50}\psset{PointSymbol=*}
\pstGeonode[PosAngle=-60](1,1){F_1}
\pstGeonode[PosAngle=-60](-1,-1){F_2}
\pstGeonode[PosAngle=-60](1,-1){B}
\pstGeonode[PosAngle=-60](-1,1){A}
\pstGeneralHyperbolaFle[PosAngle=-60]{F_1}{A}{B}{2.4}{0_1}{R_1}{
RealAxisRotAngle1}
\pstGeneralHyperbola[linecolor=red!60](0_1)(R_1)[RealAxisRotAngle
1][65]
\pstGeneralHyperbolaFle[PosAngle=-60]{F_2}{A}{B}{2.4}{0_2}{R_2}{
RealAxisRotAngle2}
\pstGeneralHyperbola[linecolor=blue!60](0_2)(R_2)[
RealAxisRotAngle2][65]
\end{pspicture}

```

The Macro `\pstGeneralHyperbolaFFN` is used to define a General Hyperbola by the given focus nodes  $F_1$ ,  $F_2$ , and one node  $N$  on it. 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 `\pstGeneralHyperbola` to draw this hyperbola.

```
\pstGeneralHyperbolaFFN [Options] {F_1}{F_2}{O}{Rab}{\theta}
```

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



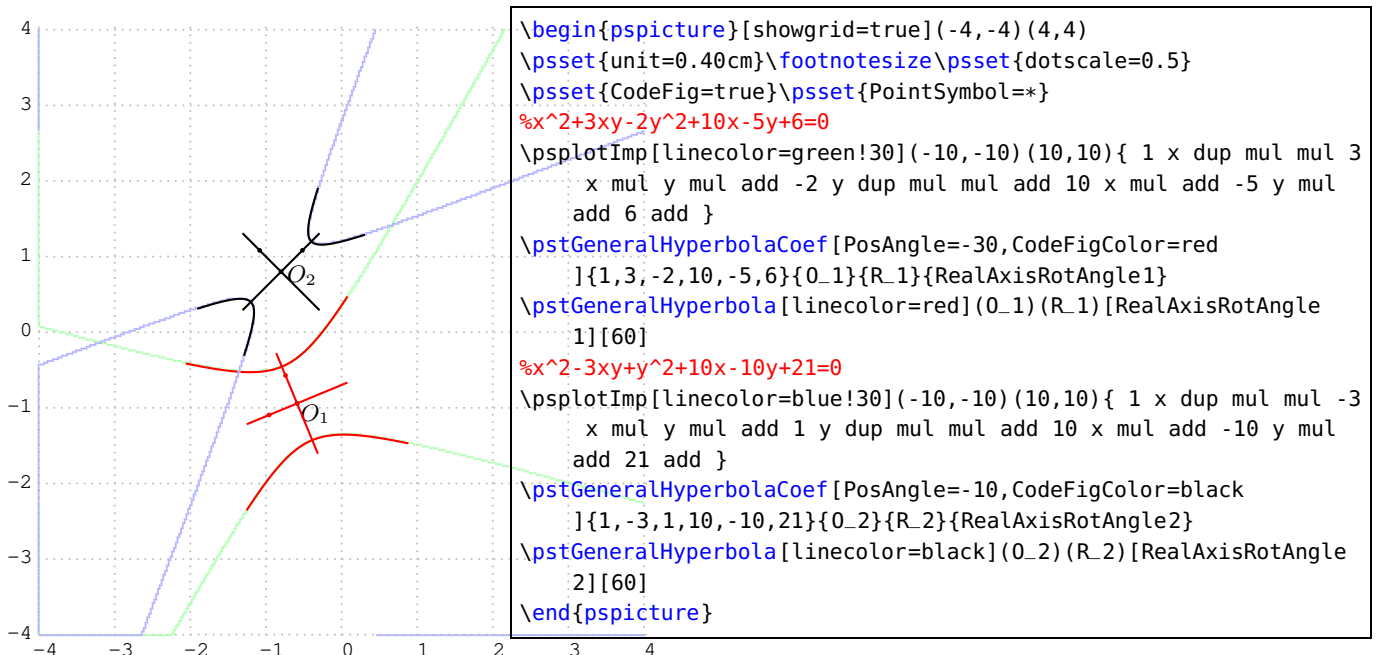
```
\begin{pspicture}[showgrid=true](0,0)(4,4)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\pstGeonode[PosAngle=-90](1,1){F_1}
\pstGeonode[PosAngle=-90](3,3){F_2}
\pstGeonode[PosAngle=-90](1,3){F_3}
\pstGeonode[PosAngle=-90](3,1){F_4}
\pstGeonode[PosAngle=90](2,3){N}
\pstGeneralHyperbolaFFN[linecolor=red!30,CodeFig=true]{F_1}{F_2}{
N}{O}{R1}{angle1}
\pstGeneralHyperbola[linecolor=red!30](O)(R1)[angle1][65]
\pstGeneralHyperbolaFFN[linecolor=blue!30,CodeFig=true]{F_3}{F
_4}{N}{O}{R2}{angle2}
\pstGeneralHyperbola[linecolor=blue!30](O)(R2)[angle2][65]
\end{pspicture}
```

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  $O$ , the output parameter  $Rab$  and the output parameter  $\theta$  are same with `\pstGeneralHyperbolaFle`. They are set to zero if the coefficients are invalid to construct a hyperbola.

In the following example, we use `\psplotImp` to draw the same hyperbolas, just to check the results given by macros `\pstGeneralHyperbolaCoef` are correct.

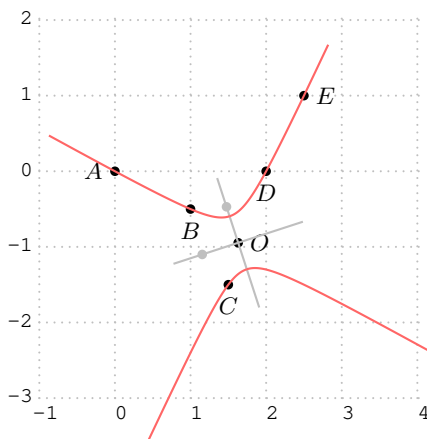


```
\begin{pspicture}[showgrid=true](-4,-4)(4,4)
\psset{unit=0.40cm}\footnotesize\psset{dotscale=0.5}
\psset{CodeFig=true}\psset{PointSymbol=*}
%x^2+3xy-2y^2+10x-5y+6=0
\psplotImp[linecolor=green!30](-10,-10)(10,10){ 1 x dup mul mul 3
x mul y mul add -2 y dup mul mul add 10 x mul add -5 y mul
add 6 add }
\pstGeneralHyperbolaCoef[PosAngle=-30,CodeFigColor=red
]{1,3,-2,10,-5,6}{O_1}{R_1}{RealAxisRotAngle1}
\pstGeneralHyperbola[linecolor=red](O_1)(R_1)[RealAxisRotAngle
1][60]
%x^2-3xy+y^2+10x-10y+21=0
\psplotImp[linecolor=blue!30](-10,-10)(10,10){ 1 x dup mul mul -3
x mul y mul add 1 y dup mul mul add 10 x mul add -10 y mul
add 21 add }
\pstGeneralHyperbolaCoef[PosAngle=-10,CodeFigColor=black
]{1,-3,1,10,-10,21}{O_2}{R_2}{RealAxisRotAngle2}
\pstGeneralHyperbola[linecolor=black](O_2)(R_2)[RealAxisRotAngle
2][60]
\end{pspicture}
```

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

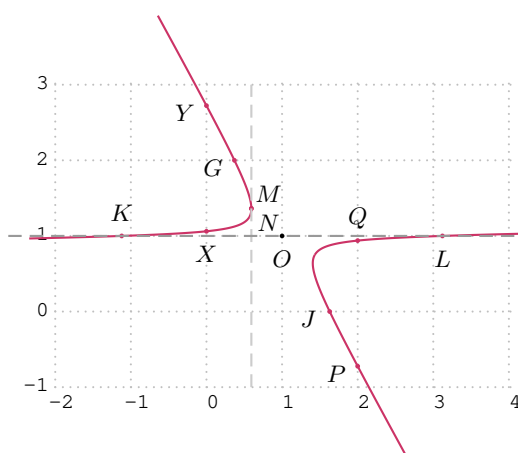


```
\begin{pspicture}[showgrid=true](-1,-3)(4,2)
\psset{unit=0.5cm}\footnotesize\psset{PointSymbol=*}
\psset{CodeFig=true,CodeFigColor=gray!50}
\pstGeonode[PosAngle=180](0,0){A}
\pstGeonode[PosAngle=-90](2,-1){B}
\pstGeonode[PosAngle=-90](3,-3){C}
\pstGeonode[PosAngle=-90](4,0){D}
\pstGeonode[PosAngle=0](5,2){E}
\pstGeneralHyperbolaABCDE [PosAngle=0]{A}{B}{C}{D}{E}{O}{R}{
RealAxisRotAngle}
\pstGeneralHyperbola [linecolor=red!60](O)(R)[RealAxisRotAngle
][80]
\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.

```
\pstGeneralHyperbolaNode [Options] (O)(a,b)[\theta]{t}{A}
\pstGeneralHyperbolaAbsNode [Options] (O)(a,b)[\theta]{x_1}{A}{B}
\pstGeneralHyperbolaOrdNode [Options] (O)(a,b)[\theta]{y_1}{A}{B}
```



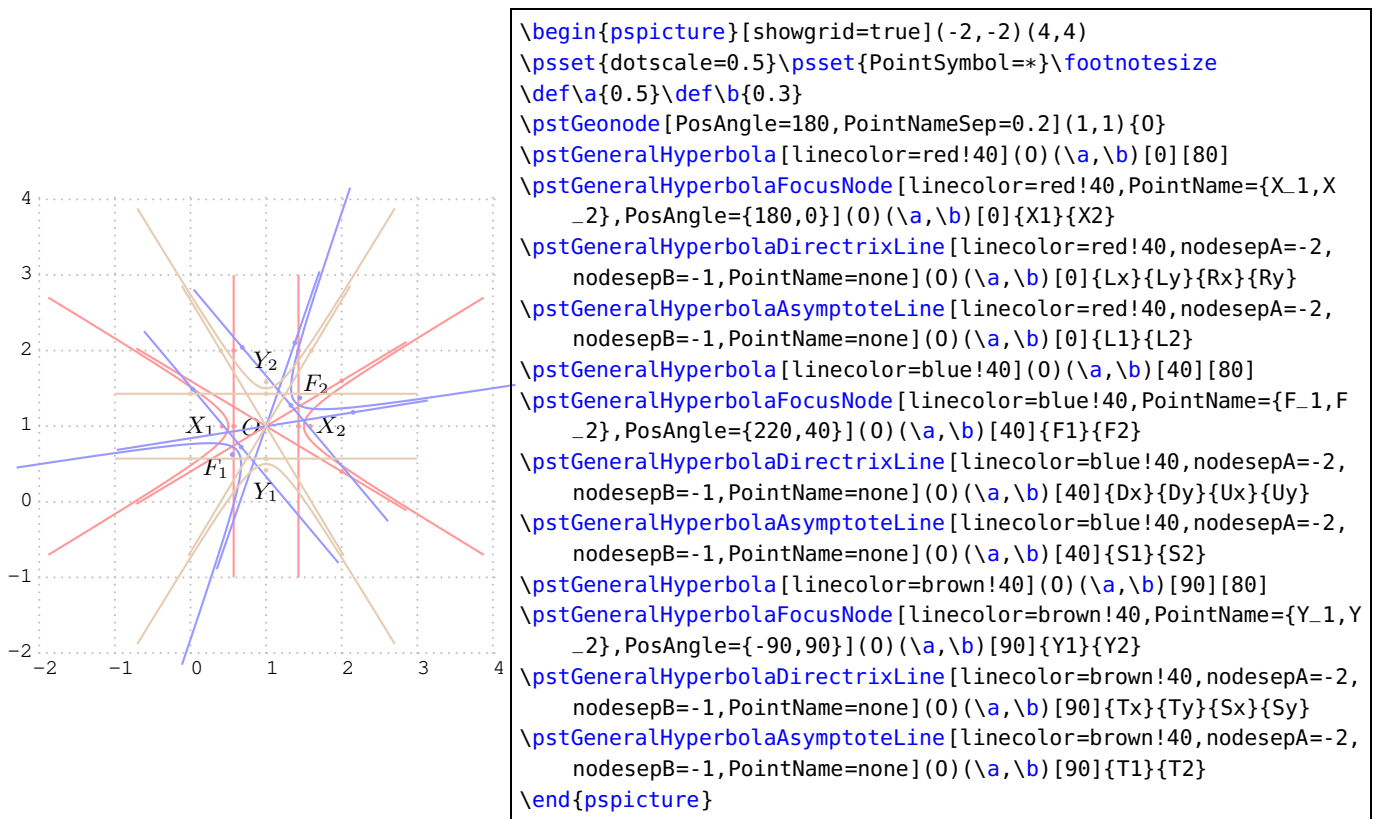
```
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\aa{0.5}\def\bb{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=-90](1,1){O}
\pstGeneralHyperbola [linecolor=purple!80](O)(\aa,\bb)[150][80]
\pstGeneralHyperbolaAbsNode [linecolor=purple!80,PosAngle
={200,90}](O)(\aa,\bb)[150]{2}{P}{Q}
\pstGeneralHyperbolaAbsNode [linecolor=purple!80,PosAngle
={-90,200}](O)(\aa,\bb)[150]{0}{X}{Y}
\pstGeneralHyperbolaAbsNode [linecolor=purple!80,PosAngle
={40,-40}](O)(\aa,\bb)[150]{0.59378}{M}{N}
\pstLine [linestyle=dashed,linecolor=gray
!40]{0.59378,-1}{0.59378,3}
\pstGeneralHyperbolaOrdNode [linecolor=purple!80,PosAngle
={200,90}](O)(\aa,\bb)[150]{2}{G}{H}
\pstGeneralHyperbolaOrdNode [linecolor=purple!80,PosAngle
={-90,200}](O)(\aa,\bb)[150]{0}{I}{J}
\pstGeneralHyperbolaOrdNode [linecolor=purple!80,PosAngle
={90,-90}](O)(\aa,\bb)[150]{1}{K}{L}
\pstLine [linestyle=dashed,linecolor=gray!80,nodesep=-1.5]{K}{L}
\end{pspicture}
```

The macro `\pstGeneralHyperbolaFocusNode` is used to find the focus nodes of the General Hyperbola, the macro `\pstGeneralHyperbolaVertexNode` is used to find the vertex nodes of the General Hyperbola, and the macro `\pstGeneralHyperbolaDirectrixLine` is used to find the directrix lines of the General Hyperbola.

```
\pstGeneralHyperbolaFocusNode [Options] (O)(a, b) [\theta] {F1}{F2}
\pstGeneralHyperbolaVertexNode [Options] (O)(a, b) [\theta] {V1}{V2}
\pstGeneralHyperbolaDirectrixLine [Options] (O)(a, b) [\theta] {Lx}{Ly}{Rx}{Ry}
```

Note that you can use the macro `\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.

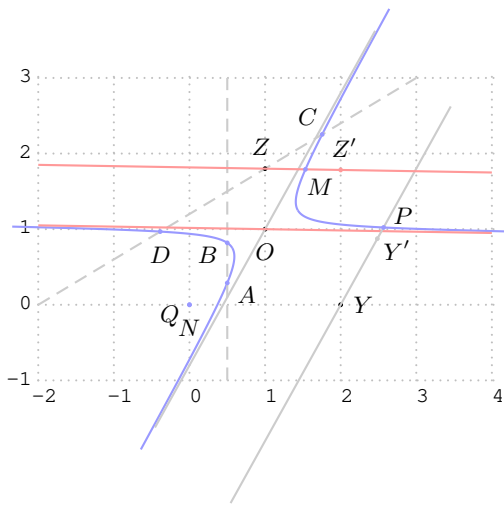
```
\pstGeneralHyperbolaAsymptoteLine [Options] (O)(a, b) [\theta] {L1}{L2}
```



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

```
\pstGeneralHyperbolaLineInter [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.



```

\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=-90](1,1){O}
\pstGeneralHyperbola[linecolor=blue!40](0)(\ax,\b)[30][80]
\pstLine[linestyle=dashed,linecolor=gray!40]{0.5,-1}{0.5,3}{A}{B}
\pstGeneralHyperbolaLineInter[linecolor=blue!40,PosAngle
=-{30,210}](0)(\ax,\b)[30]{0.5,-1}{0.5,3}{A}{B}
\pstLine[linestyle=dashed,linecolor=gray!40]{-2,0}{3,3}
\pstGeneralHyperbolaLineInter[linecolor=blue!40,PosAngle
={130,-90}](0)(\ax,\b)[30]{-2,0}{3,3}{C}{D}
\pstGeonode[PosAngle={0,100}](2,0){Y}(1,1.8){Z}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=gray!40,PointName=
none,PointSymbol=none]{0}{\b\space \a\space div 1 atan 30
add}{U}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=red!40,PointName=
none,PointSymbol=none]{0}{\b\space \a\space div neg 1 atan
30 add}{V}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=gray!40,PosAngle
=-30]{Y}{\b\space \a\space div 1 atan 30 add}{Y'}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=red!40,PosAngle=80]{
Z}{\b\space \a\space div neg 1 atan 30 add}{Z'}
\pstGeneralHyperbolaLineInter[linecolor=blue!40,PosAngle
=-{50,-90}](0)(\ax,\b)[30]{Z}{Z'}{M}{N}
\pstGeneralHyperbolaLineInter[linecolor=blue!40,PosAngle
={30,210}](0)(\ax,\b)[30]{Y}{Y'}{P}{Q}
\end{pspicture}

```

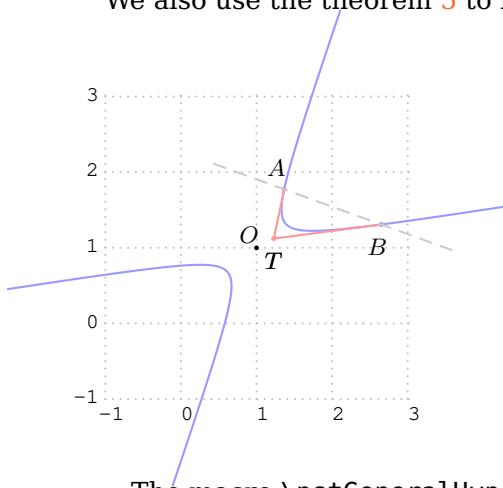
The macro `\pstGeneralHyperbolaPolarNode` is used to find the polar point  $T$  of chord  $AB$  on the general hyperbola.

```

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

```

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



```

\begin{pspicture}[showgrid=true](-1,-1)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=120,PointNameSep=0.2](1,1){O}
\pstGeneralHyperbola[linecolor=blue!40](0)(\ax,\b)[40][80]
\pstGeneralHyperbolaNode[linecolor=blue!40,PosAngle=110](0)(\ax,\b)
[40]{50}{A}
\pstGeneralHyperbolaNode[linecolor=blue!40,PosAngle=-100](0)(\ax,\b)
[40]{-70}{B}
\pstGeneralHyperbolaPolarNode[linecolor=red!40,PosAngle=-90](0)(\ax,\b)
[40]{A}{B}{T}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep=-1]{A}{B}
\end{pspicture}

```

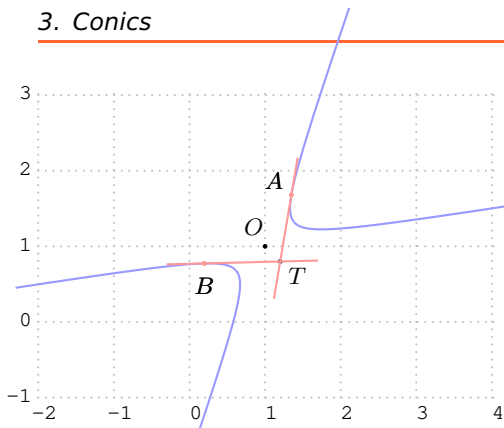
The macro `\pstGeneralHyperbolaTangentNode` is used to find the tangent point  $A$  and  $B$  of point  $T$  outside of the general hyperbola.

```

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

```

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



```
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=120](1,1){O}
\pstGeneralHyperbola[linecolor=blue!40](0)(\ax,\bx)[40][80]
\pstGeonode[PosAngle=-40](1.2,0.8){T}
\pstGeneralHyperbolaTangentNode[linecolor=red!40,PosAngle
={140,-90},nodesep=-0.5](0)(\ax,\bx)[40]{T}{A}{B}
\end{pspicture}
```

### 3.10. General Inversion Hyperbola

The General Inversion 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 Inversion Hyperbola equation (17), using the rotation transform formula (3), then we have

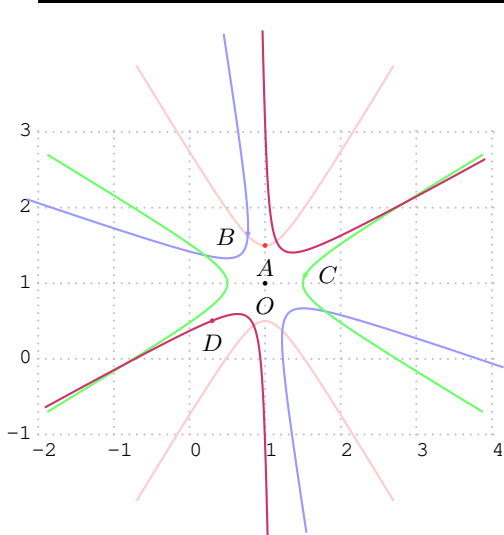
$$\begin{cases} x' = (b \tan \alpha + x_0) \cos \theta - (a \sec \alpha + y_0) \sin \theta = x'_0 + b \tan \alpha \cos \theta - a \sec \alpha \sin \theta \\ y' = (b \tan \alpha + x_0) \sin \theta + (a \sec \alpha + y_0) \cos \theta = y'_0 + b \tan \alpha \sin \theta + a \sec \alpha \cos \theta \end{cases} \quad (20)$$

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 Inversion Hyperbola with coordinate translation and rotation as following:

$$\begin{cases} x = x_0 + b \tan \alpha \cos \theta - a \sec \alpha \sin \theta \\ y = y_0 + b \tan \alpha \sin \theta + a \sec \alpha \cos \theta \end{cases} \quad (21)$$

The macro `\pstGeneralIHyperbola` is used to draw a General Inversion 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 `angleY` is used to truncate the height of the figure, it should be setup from 0 to 90.

```
\pstGeneralIHyperbola [Options] (O)(a, b) [\theta] [angleY]
```



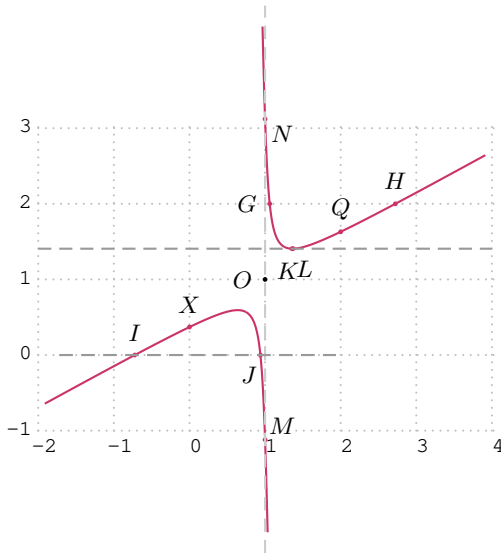
```
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=-90](1,1){O}
\pstGeneralIHyperbola[linecolor=red!20](0)(\ax,\bx)[0][80]
\pstGeneralIHyperbolaNode[linecolor=red!80,PosAngle=-90](0)(\ax,\bx)
[0]{0}{A}
\pstGeneralIHyperbola[linecolor=blue!40](0)(\ax,\bx)[40][80]
\pstGeneralIHyperbolaNode[linecolor=blue!40,PosAngle=190](0)(\ax,\bx)
[40]{40}{B}
\pstGeneralIHyperbola[linecolor=green!60](0)(\ax,\bx)[90][80]
\pstGeneralIHyperbolaNode[linecolor=green!60,PosAngle=0](0)(\ax,\bx)
[90]{200}{C}
\pstGeneralIHyperbola[linecolor=purple!80](0)(\ax,\bx)[150][80]
\pstGeneralIHyperbolaNode[linecolor=purple!80,PosAngle=-90](0)(\ax,\bx)
[150]{50}{D}
\end{pspicture}
```

The macro `\pstGeneralIHyperbolaNode` is used to draw a node whose parameter is the given value  $t$  on General Inversion Hyperbola, please refer to equation (21).

The macro `\pstGeneralIHyperbolaAbsNode` is used to draw the nodes whose abscissa are the given value  $x_1$  on General Inversion Hyperbola. The macro `\pstGeneralIHyperbolaOrdNode` is used to draw the nodes whose ordinate are the given value  $y_1$  on General Inversion Hyperbola.

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

```
\pstGeneralIHyperbolaNode [Options] (O)(a, b) [\theta] {t}{A}
\pstGeneralIHyperbolaAbsNode [Options] (O)(a, b) [\theta] {x_1}{A}{B}
\pstGeneralIHyperbolaOrdNode [Options] (O)(a, b) [\theta] {y_1}{A}{B}
```



```
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=180](1,1){O}
\pstGeneralIHyperbola[linecolor=purple!80](0)(\ax,\bx)[150][80]
\pstGeneralIHyperbolaAbsNode[linecolor=purple!80,PosAngle
={200,90}](0)(\ax,\bx)[150]{2}{P}{Q}
\pstGeneralIHyperbolaAbsNode[linecolor=purple!80,PosAngle
={90,200}](0)(\ax,\bx)[150]{0}{X}{Y}
\pstGeneralIHyperbolaAbsNode[linecolor=purple!80,PosAngle
={40,-40}](0)(\ax,\bx)[150]{1}{M}{N}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep=-1.5]{M}{N}
\pstGeneralIHyperbolaOrdNode[linecolor=purple!80,PosAngle
={180,90}](0)(\ax,\bx)[150]{2}{G}{H}
\pstGeneralIHyperbolaOrdNode[linecolor=purple!80,PosAngle
={90,240}](0)(\ax,\bx)[150]{0}{I}{J}
\pstGeneralIHyperbolaOrdNode[linecolor=purple!80,PosAngle
={-100,-60}](0)(\ax,\bx)[150]{1.4063}{K}{L}
\pstLine[linestyle=dashed,linecolor=gray!80,nodesep=-1]{I}{J}
\pstLine[linestyle=dashed,linecolor=gray!80,nodesep
=-1]{-1}{-1,1.4063}{3,1.4063}
\end{pspicture}
```

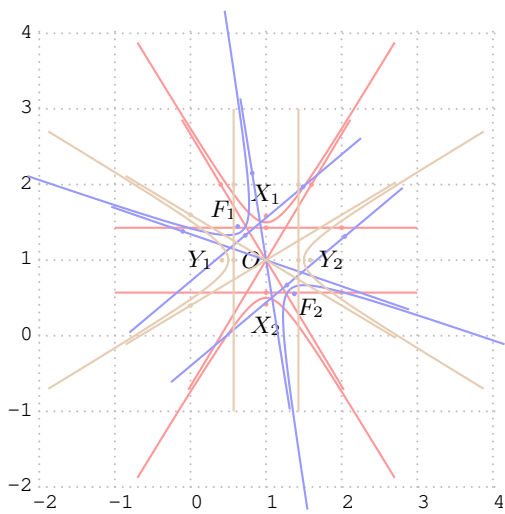
The macro `\pstGeneralIHyperbolaFocusNode` is used to find the focus nodes of the General Inversion Hyperbola, the macro `\pstGeneralIHyperbolaVertexNode` is used to find the vertex nodes of the General Inversion Hyperbola, and the macro `\pstGeneralIHyperbolaDirectrixLine` is used to find the directrix lines of the General Inversion Hyperbola.

```
\pstGeneralIHyperbolaFocusNode [Options] (O)(a, b) [\theta] {F_1}{F_2}
\pstGeneralIHyperbolaVertexNode [Options] (O)(a, b) [\theta] {V_1}{V_2}
\pstGeneralIHyperbolaDirectrixLine [Options] (O)(a, b) [\theta] {L_x}{L_y}{R_x}{R_y}
```

Note that you can use the macro `\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.

```
\pstGeneralHyperbolaAsymptoteLine [Options] (O)(a, b) [\theta] {L_1}{L_2}
```





```

\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=180,PointNameSep=0.2](1,1){O}
\pstGeneralIHyperbola[linecolor=red!40](0)(\A,\B)[0][80]
\pstGeneralIHyperbolaFocusNode[linecolor=red!40,PointName={X_1,X_2},PosAngle={90,-90}](0)(\A,\B)[0]{X1}{X2}
\pstGeneralIHyperbolaDirectrixLine[linecolor=red!40,nodesepA=-2,nodesepB=-1,PointName=none](0)(\A,\B)[0]{Lx}{Ly}{Rx}{Ry}
\pstGeneralIHyperbolaAsymptoteLine[linecolor=red!40,nodesepA=-2,nodesepB=-1,PointName=none](0)(\A,\B)[0]{T1}{T2}
\pstGeneralIHyperbola[linecolor=blue!40](0)(\A,\B)[40][80]
\pstGeneralIHyperbolaFocusNode[linecolor=blue!40,PointName={F_1,F_2},PosAngle={130,-40}](0)(\A,\B)[40]{F1}{F2}
\pstGeneralIHyperbolaDirectrixLine[linecolor=blue!40,nodesepA=-2,nodesepB=-1,PointName=none](0)(\A,\B)[40]{Dx}{Dy}{Ux}{Uy}
\pstGeneralIHyperbolaAsymptoteLine[linecolor=blue!40,nodesepA=-2,nodesepB=-1,PointName=none](0)(\A,\B)[40]{S1}{S2}
\pstGeneralIHyperbola[linecolor=brown!40](0)(\A,\B)[90][80]
\pstGeneralIHyperbolaFocusNode[linecolor=brown!40,PointName={Y_1,Y_2},PosAngle={180,0}](0)(\A,\B)[90]{Y1}{Y2}
\pstGeneralIHyperbolaDirectrixLine[linecolor=brown!40,nodesepA=-2,nodesepB=-1,PointName=none](0)(\A,\B)[90]{Tx}{Ty}{Sx}{Sy}
\pstGeneralIHyperbolaAsymptoteLine[linecolor=brown!40,nodesepA=-2,nodesepB=-1,PointName=none](0)(\A,\B)[90]{R1}{R2}
\end{pspicture}

```

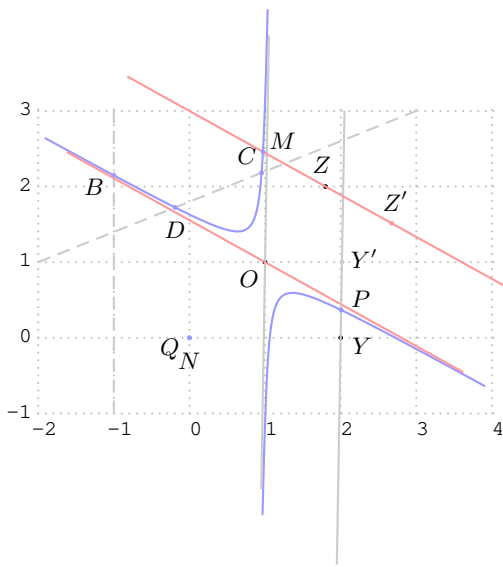
The macro `\pstGeneralIHyperbolaLineInter` is used to find the intersections  $C$  and  $D$  of the general inversion hyperbola and the given line  $AB$ .

```

\pstGeneralIHyperbolaLineInter [Options] (O)(a,b)[θ]{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.

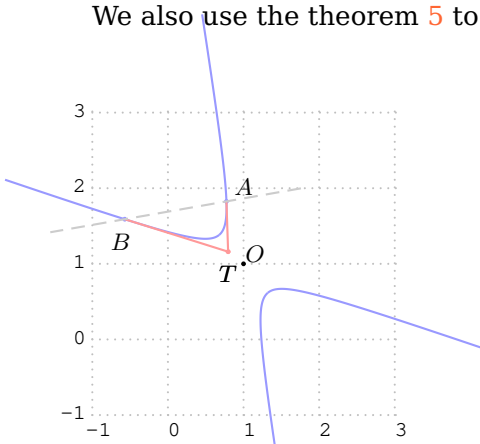


```
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\aa{0.5}\def\bb{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=225](1,1){O}
\pstGeneralIHyperbola[linecolor=blue!40](0)(\aa,\bb)[30][80]
\pstLine[linestyle=dashed,linecolor=gray!40]{-1,-1}{-1,3}
\pstGeneralIHyperbolaLineInter[linecolor=blue!40,PosAngle
={-30,210}](0)(\aa,\bb)[30]{-1,-1}{-1,3}{A}{B}
\pstLine[linestyle=dashed,linecolor=gray!40]{-2,1}{3,3}
\pstGeneralIHyperbolaLineInter[linecolor=blue!40,PosAngle
={130,-90}](0)(\aa,\bb)[30]{-2,1}{3,3}{C}{D}
\pstGeonode[PosAngle={-20,100}](2,0){Y}(1.8,2){Z}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=gray!40,PointName=
none,PointSymbol=none]{0}{\aa\space \bb\space div 1 atan 30
add}{U}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=red!40,PointName=
none,PointSymbol=none]{0}{\aa\space \bb\space div neg 1 atan
30 add}{V}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=gray!40,PosAngle
=10]{Y}{\aa\space \bb\space div 1 atan 30 add}{Y'}
\pstLineAA[nodesepA=-3,nodesepB=-2,linecolor=red!40,PosAngle=80]{
Z}{\aa\space \bb\space div neg 1 atan 30 add}{Z'}
\pstGeneralIHyperbolaLineInter[linecolor=blue!40,PosAngle
={30,-90}](0)(\aa,\bb)[30]{Z}{Z'}{M}{N}
\pstGeneralIHyperbolaLineInter[linecolor=blue!40,PosAngle
={30,210}](0)(\aa,\bb)[30]{Y}{Y'}{P}{Q}
\end{pspicture}
```

The macro `\pstGeneralIHyperbolaPolarNode` is used to find the polar point  $T$  of chord  $AB$  on the general inversion hyperbola.

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

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

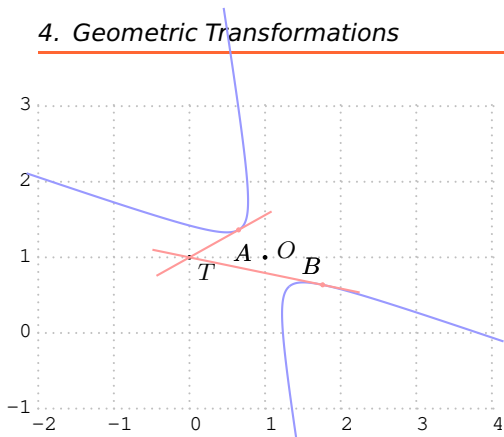


```
\begin{pspicture}[showgrid=true](-1,-1)(3,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\aa{0.5}\def\bb{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=40,PointNameSep=0.2](1,1){O}
\pstGeneralIHyperbola[linecolor=blue!40](0)(\aa,\bb)[40][80]
\pstGeneralIHyperbolaNode[linecolor=blue!40,PosAngle=40](0)(\aa,\bb)
[40]{50}{A}
\pstGeneralIHyperbolaNode[linecolor=blue!40,PosAngle=-100](0)(\aa,\bb)
[40]{-70}{B}
\pstGeneralIHyperbolaPolarNode[linecolor=red!40,PosAngle=-90](0)
(\aa,\bb)[40]{A}{B}{T}
\pstLine[linestyle=dashed,linecolor=gray!40,nodesep=-1]{A}{B}
\end{pspicture}
```

The macro `\pstGeneralIHyperbolaTangentNode` is used to find the tangent point  $A$  and  $B$  of point  $T$  outside of the general inversion hyperbola.

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

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



```
\begin{pspicture}[showgrid=true](-2,-1)(4,3)
\psset{dotscale=0.5}\psset{PointSymbol=*}\footnotesize
\def\ax{0.5}\def\bx{0.3}\psset{PointNameSep=0.3}
\pstGeonode[PosAngle=20](1,1){O}
\pstGeneralHyperbola[linecolor=blue!40](0)(\ax,\bx)[40][80]
\pstGeonode[PosAngle=-40](0,1){T}
\pstGeneralHyperbolaTangentNode[linecolor=red!40,PosAngle
=-80,120],nodesep=-0.5)(0)(\ax,\bx)[40]{T}{A}{B}
\end{pspicture}
```

## 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 syntactic 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.

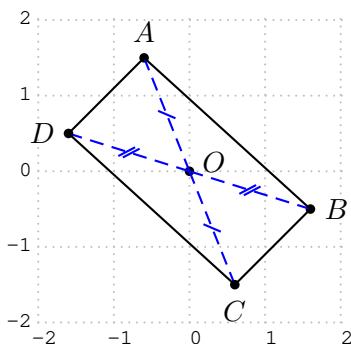
The first shared parameter is CodeFig which draws the specific constructions lines. Its default value is false, and a true value activates this optional drawing. The drawing is done using the line style CodeFigStyle (by default dashed), with the color CodeFigColor (by default cyan).

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

```
\pstSym0 [Options] {O}{M1, M2, ..., Mn} [M'1, M'2, ..., M'p]
```

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

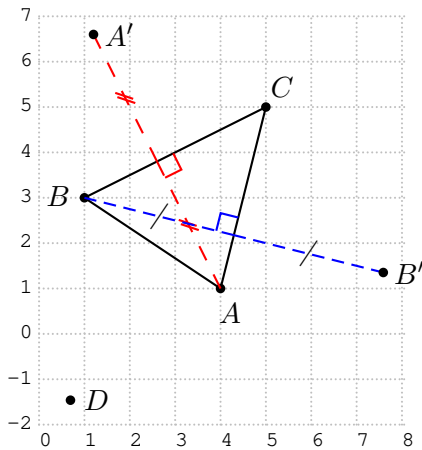


```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psset{CodeFig=true}
\pstGeonode[PosAngle={20,90,0}]{O}(-.6,1.5){A}(1.6,-.5){B}
\pstSym0[CodeFigColor=blue,
PosAngle={-90,180}]{O}{A, B}{C, D}
\pstLineAB{A}{B}\pstLineAB{C}{D}
\pstLineAB{A}{D}\pstLineAB{C}{B}
\end{pspicture}
```

### 4.2. Orthogonal (or axial) symmetry

```
\pst0rtSym [Options] {A}{B}{M1, M2, ..., Mn} [M'1, M'2, ..., M'p]
```

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



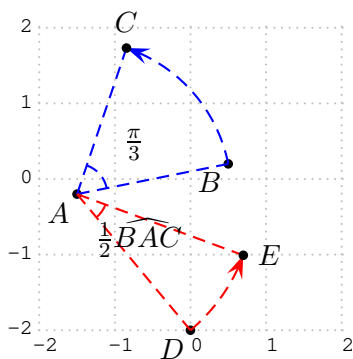
```
\psset{unit=0.6}
\begin{pspicture}[showgrid](0,-2)(8,7)
\pstTriangle(1,3){B}(5,5){C}(4,1){A}
\pstOrtSym{A}{B}{C}[D]
\psset{CodeFig=true}
\pstOrtSym[dash=2mm 2mm,CodeFigColor=red]%
{C}{B}{A}
\pstOrtSym[SegmentSymbol=pstslash,
linestyle=dotted,dotsep=3mm,CodeFigColor=blue]%
{C}{A}{B}
\end{pspicture}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

### 4.3. Rotation

```
\pstRotation [Options] {O}{M1, M2, ..., Mn} [M'1, M'2, ..., M'p]
\pstAngleAOB{A}{O}{B}
```

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

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



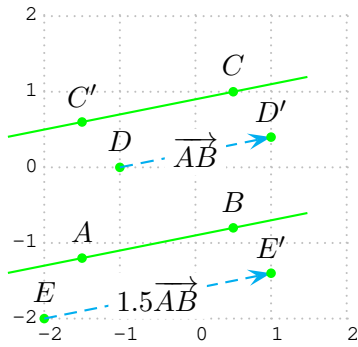
```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psset{arrowscale=2}
\pstGeonode[PosAngle=-135](-1.5,-.2){A}%
(.5,.2){B}(0,-2){D}
\pstRotation[PosAngle=90,RotAngle=60,
CodeFig,CodeFigColor=blue,
TransformLabel=\frac{\pi}{3}]{A}{B}[C]
\pstRotation[AngleCoef=.5,
RotAngle=\pstAngleAOB{B}{A}{C},
CodeFigColor=red,CodeFig,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

### 4.4. Translation

```
\pstTranslation [Options] {A}{B}{M1, M2, ..., Mn} [M'1, M'2, ..., M'p]
```

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

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

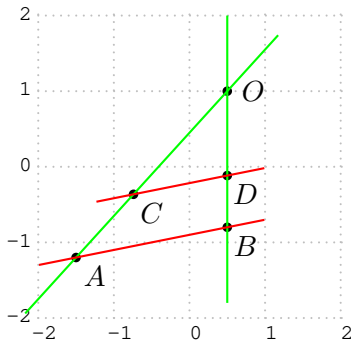


```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psset{linecolor=green,nodesep=-1,
PosAngle=90,arrowscale=2}
\pstGeonode(-1.5,-1.2){A}(.5,-.8){B}(.5,1){C}(-1,0){D}(-2,-2){E}
\pstTranslation{B}{A}{C}
\psset{CodeFig,TransformLabel=default}
\pstTranslation{A}{B}{D}
\pstTranslation[DistCoef=1.5]{A}{B}{E}
\pstLineAB{A}{B}\pstLineAB{C}{C'}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

#### 4.5. Homothetic

```
\pstHomO [Options] {O}{M1, M2, ..., Mn} [M'1, M'2, ..., M'p]
```

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

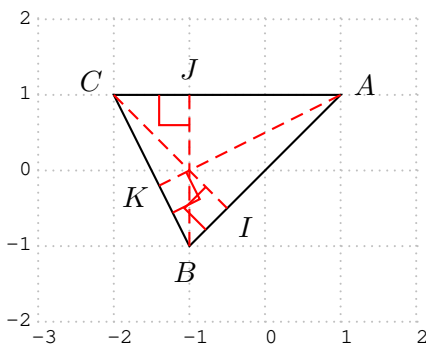


```
\begin{pspicture}[showgrid](-2,-2)(2,2)
\pstGeonode[PosAngle={0,-45}](.5,1){O}%
(-1.5,-1.2){A}(.5,-.8){B}
\pstHomO[HomCoef=.62,PosAngle=-45]{O}{A,B}[C,D]
\psset{linecolor=green,nodesep=-1}
\pstLineAB{A}{O}\pstLineAB{B}{O}
\psset{linecolor=red,nodesep=-.5}
\pstLineAB{A}{B}\pstLineAB{C}{D}
\end{pspicture}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

#### 4.6. Orthogonal projection

```
\pstProjection [Options] {A}{B}{M1, M2, ..., Mn} [M'1, M'2, ..., 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.



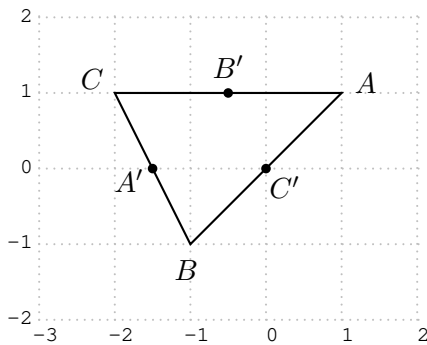
```
\begin{pspicture}[showgrid](-3,-2)(2,2)
\psset{PointSymbol=none,CodeFig,CodeFigColor=red}
\pstTriangle(1,1){A}(-2,1){C}(-1,-1){B}
\pstProjection{A}{B}{C}[I]
\pstProjection{A}{C}{B}[J]
\pstProjection{C}{B}{A}[K]
\end{pspicture}
```

## 5. Special object

### 5.1. Midpoint

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

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

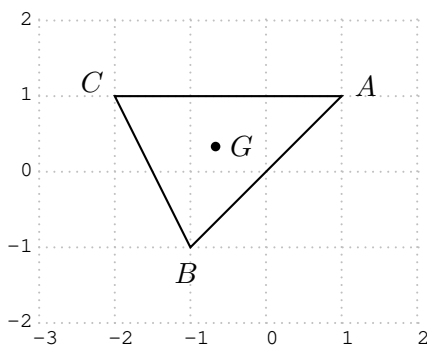


```
\begin{pspicture}[showgrid](-3,-2)(2,2)
\pstTriangle[PointSymbol=none]%
(1,1){A}(-1,-1){B}(-2,1){C}
\pstMiddleAB{A}{B}{C'}
\pstMiddleAB{C}{A}{B'}
\pstMiddleAB{B}{C}{A'}
\end{pspicture}
```

### 5.2. Triangle center of gravity

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

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

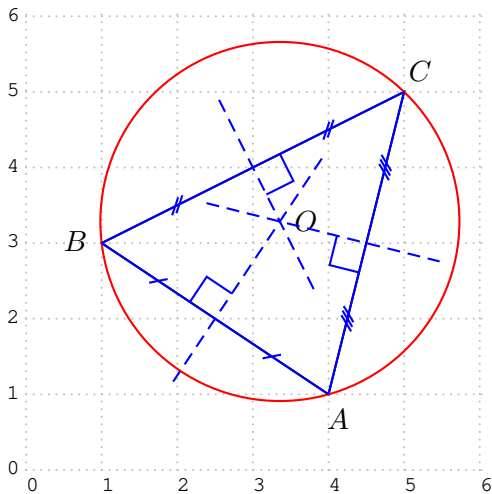


```
\begin{pspicture}[showgrid](-3,-2)(2,2)
\pstTriangle[PointSymbol=none]%
(1,1){A}(-1,-1){B}(-2,1){C}
\pstCGravABC{A}{B}{C}{G}
\end{pspicture}
```

### 5.3. Centre of the circumcircle of a triangle

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

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



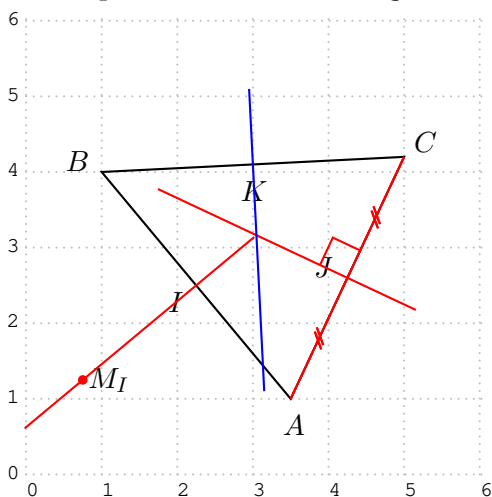
```
\begin{pspicture}[showgrid](6,6)
\pstTriangle[PointSymbol=none]%
(4,1){A}(1,3){B}(5,5){C}
\pstCircleABC[CodeFig,CodeFigColor=blue,
linecolor=red,PointSymbol=none]{A}{B}{C}{O}
\end{pspicture}
```

#### 5.4. Perpendicular bisector of a segment

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

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

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



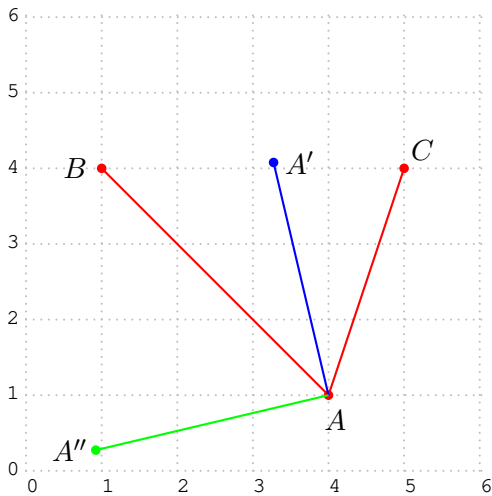
```
\begin{pspicture}[showgrid](6,6)
\pstTriangle[PointSymbol=none](3.5,1){A}(1,4){B}(5,4.2){C}
\psset{linecolor=red,CodeFigColor=red,nodesep=-1}
\pstMediatorAB[PointSymbolA=none]{A}{B}{I}{M_I}
\psset{PointSymbol=none,PointNameB=none}
\pstMediatorAB[CodeFig=true]{A}{C}{J}{M_J}
\pstMediatorAB[PosAngleA=45,linecolor=blue]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

#### 5.5. Bisectors of angles

```
\pstBisectBAC [Options] {B}{A}{C}{N}
\pstOutBisectBAC [Options] {B}{A}{C}{N}
```

Possible optional arguments are `PointSymbol`, `PosAngle`, `PointName`, `PointNameSep`, and `PtNameMath`. There are two bisectors for a given geometric angle: the inside one and the outside one; this is why there is two commands. The angle is specified by three points specified in the

trigonometric direction (anti-clockwise). The result of the commands is the specific line and a point belonging to this line. This point is built by a rotation of point  $B$ .



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

## 6. Intersections

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

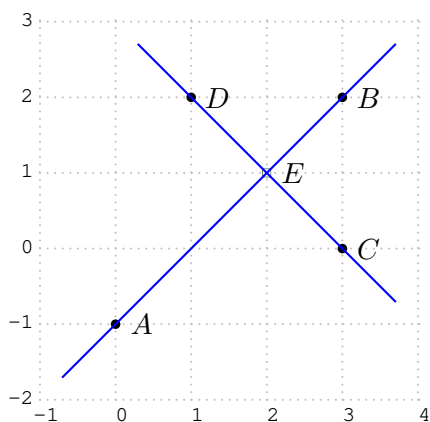
- line-line;
- line-circle;
- circle-circle;
- function-function;
- function-line;
- function-circle.

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

### 6.1. Line-Line

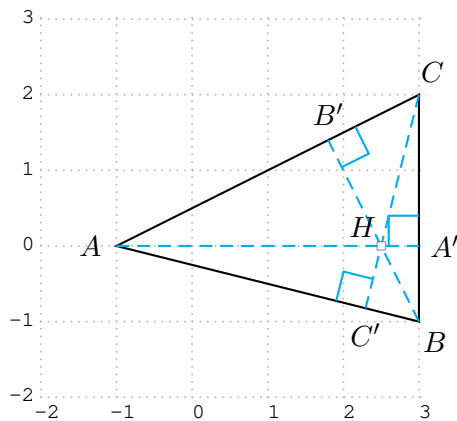
```
\pstInterLL [Options] {A}{B}{C}{D}{M}
```

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



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





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

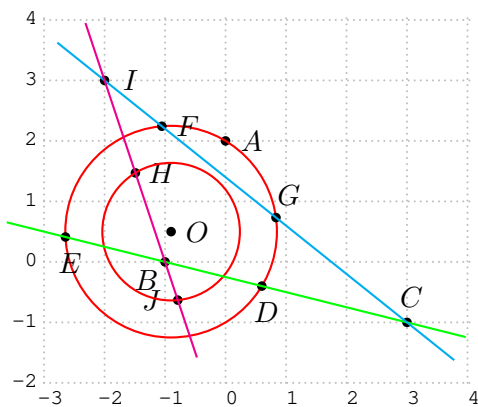
## 6.2. Circle-Line

```
\pstInterLC [Options] {A}{B}{O}{C}{M1}{M2}
```

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 two points is such that the vectors  $\vec{AB}$  and  $\vec{M_1M_2}$  are in the same direction. Thus, if the points defining the line are switched, then the resulting points will be also switched. If the intersection is void, then the points are positioned at the center of the circle.

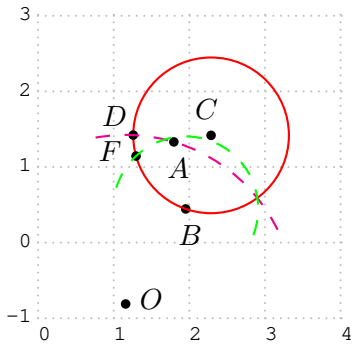


```
\psset{unit=0.8}
\begin{pspicture}[showgrid](-3,-2)(4,4)
\pstGeonode[PosAngle={-135,80,0}](-1,0){B}(3,-1){C}{-.9,.5}{O}
(0,2){A}
\pstGeonode(-2,3){I}
\pstCircleOA[linecolor=red]{O}{A}
\pstInterLC[PosAngle=-80]{C}{B}{O}{A}{D}{E}
\pstInterLC[PosAngleB=60,Radius=\pstDistAB{O}{D}]{I}{C}{O}{F}{G}
\pstInterLC[PosAngleB=180,Diameter=\pstDistMul{O}{D}{1.3}]{I}{B}{O}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

## 6.3. Circle-Circle

```
\pstInterCC [Options] {O1}{B}{O2}{C}{M1}{M2}
```

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 `CodeFigArc` and `CodeFigBarc` specified the direction of these optional arcs: trigonometric (by default) or clockwise. Here is a first example.



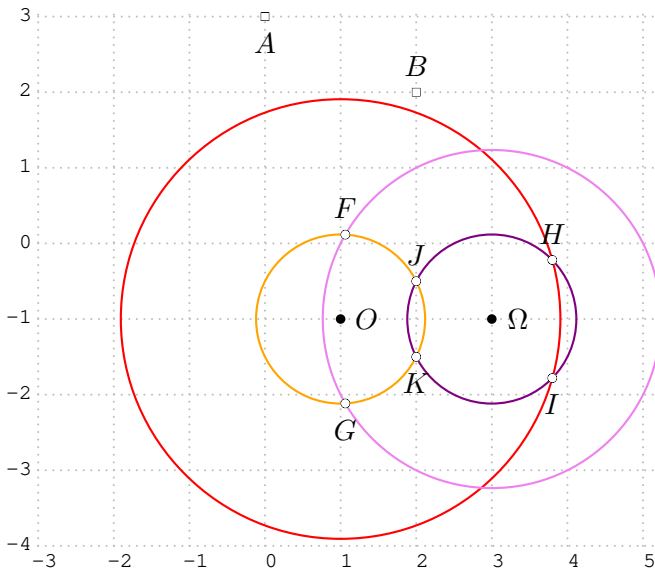
```

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

```

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

The macro `\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 `PosAngleA` and `PosAngleB` to change the default behavior.



```

\begin{pspicture}[showgrid](-3,-4)(7,3)
\pstGeonode[PointName={\Omega,0}](3,-1){\Omega}
(1,-1){O}
\pstGeonode[PointSymbol=square, PosAngle
={-90,90}](0,3){A}(2,2){B}
\psset{PointSymbol=o}
\pstCircleOA[linecolor=red, Radius=\pstDistMul{A}
{B}{1 3 10 div add}]{O}{}
\pstCircleOA[linecolor=Orange, Diameter=\
pstDistAB{A}{B}]{O}{}
\pstCircleOA[linecolor=Violet, Radius=\pstDistAB{
A}{B}]{\Omega}{}
\pstCircleOA[linecolor=Purple, Diameter=\
pstDistAB{A}{B}]{\Omega}{}
\pstInterCC[RadiusA=\pstDistMul{A}{B}{1 3 10 div
add},
RadiusB=\pstDistAB{A}{B}]{O}{\Omega}{D}{E}
\pstInterCC[DiameterA=\pstDistAB{A}{B}, RadiusB=\
pstDistAB{A}{B},
PosAngleA=90, PosAngleB=-90]{O}{\Omega}
}{F}{G}
\pstInterCC[RadiusA=\pstDistMul{A}{B}{1 3 10 div
add}, DiameterB=\pstDistAB{A}{B},
PosAngleA=90, PosAngleB=-90]{O}{\Omega}
}{H}{I}
\pstInterCC[DiameterA=\pstDistAB{A}{B}, DiameterB=
\pstDistAB{A}{B},
PosAngleA=90, PosAngleB=-90]{O}{\Omega}
}{J}{K}
\end{pspicture}

```

#### 6.4. Function-function

```

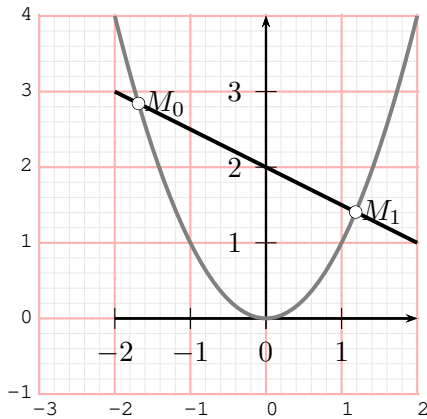
\pstInterFF [Options] {f}{g}{x_0}{M}

```

This function put a point at the intersection between two curves defined by a function.  $x_0$  is an intersection approximated value of the abscissa. It is obviously possible to use this function several time if more than one intersection is present. Each function is described in PostScript in the same

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 positioned at the origin. On the other hand, the research can be trapped (in a local extremum near zero).

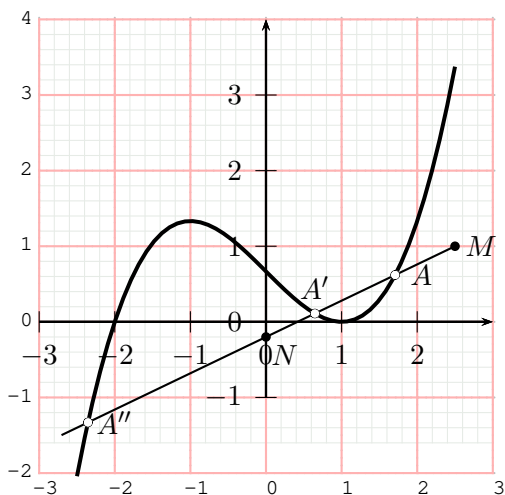


```
\begin{pspicture}[showgrid](-3,-1)(2,4)
\psaxes{->}(0,0)(-2,0)(2,4)
\psset{linewidth=1.5pt,algebraic}
\psplot[linecolor=gray]{-2}{2}{x^2}
\psplot{-2}{2}{2-x/2}
\psset{PointSymbol=o}
\pstInterFF{2-x/2}{x^2}{1}{M_1}
\pstInterFF{2-x/2}{x^2}{-2}{M_0}
\end{pspicture}
```

### 6.5. Function-line

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

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

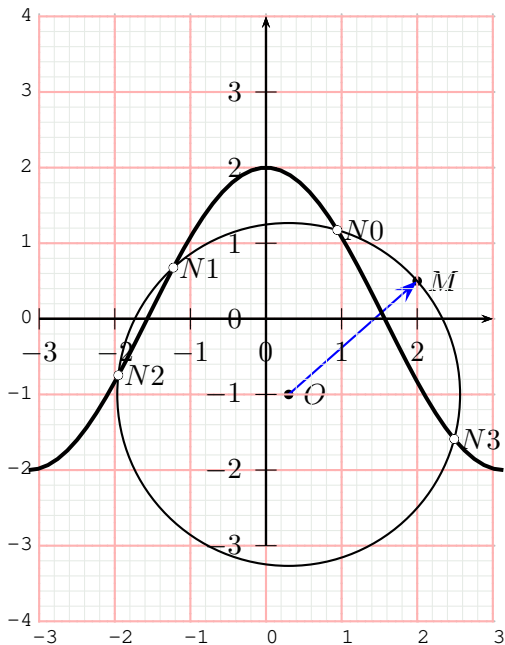


```
\begin{pspicture}[showgrid](-3,-1.5)(3,4)
\def\F{x^3/3 - x + 2/3 }
\psaxes{->}(0,0)(-3,-1)(3,4)
\psplot[linewidth=1.5pt,algebraic]{-2.5}{2.5}{\F}
\psset{PointSymbol=*}
\pstGeonode[PosAngle={-45,0}](0,-.2){N}(2.5,1){M}
\pstLineAB[nodesepA=-3cm]{N}{M}
\psset{PointSymbol=o,algebraic}
\pstInterFL{\F}{N}{M}{2}{A}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

### 6.6. Function-Circle

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

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



```

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

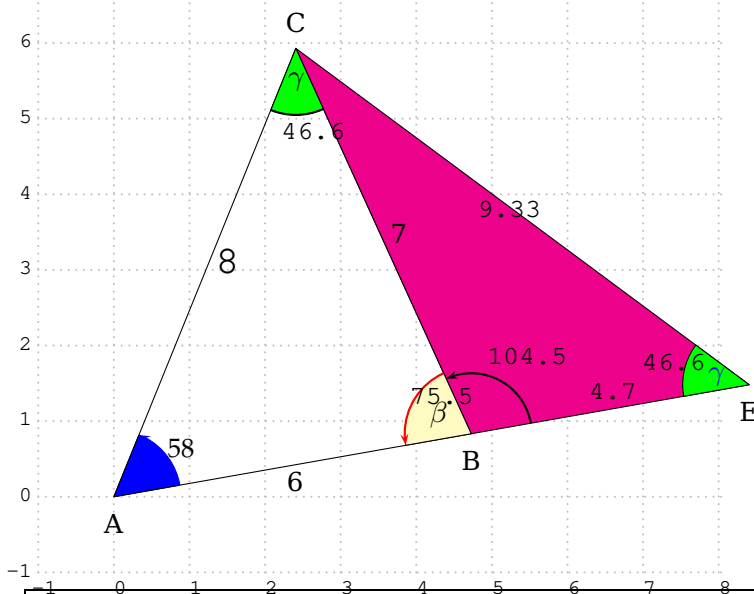
```

## 7. Helper Macros

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

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

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



```

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

```

```

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

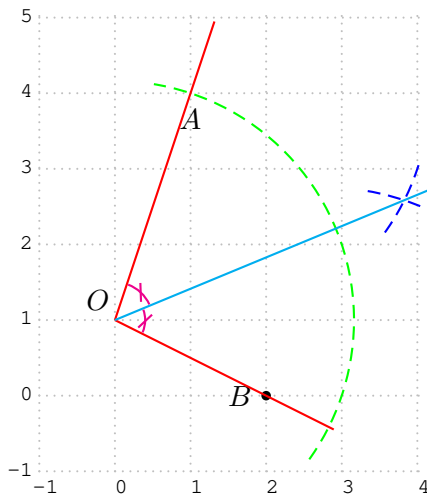
```

## Part II.

### Examples gallery

#### A. Basic geometry

##### A.1. Drawing of the bissector



```

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

```

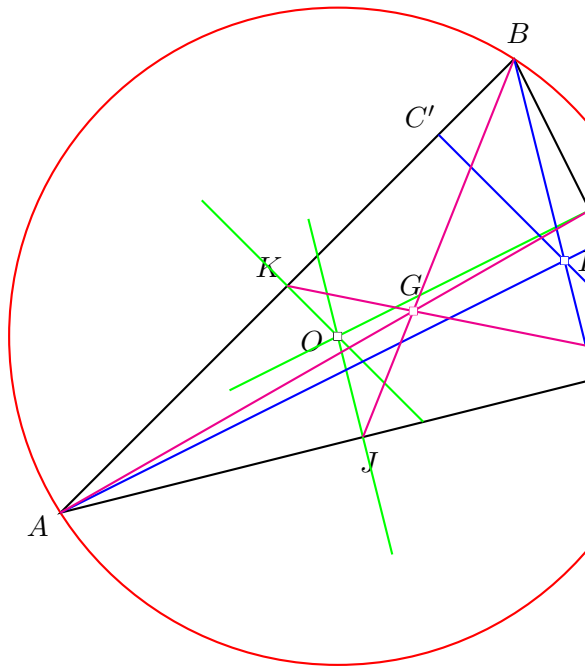
### A.2. Transformation de polygones et courbes

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

```

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

## A.3. Triangle lines



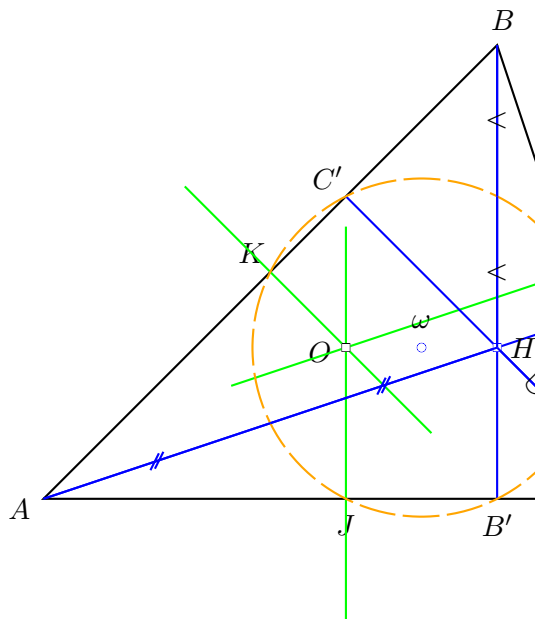
```

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

```



## A.4. Euler circle



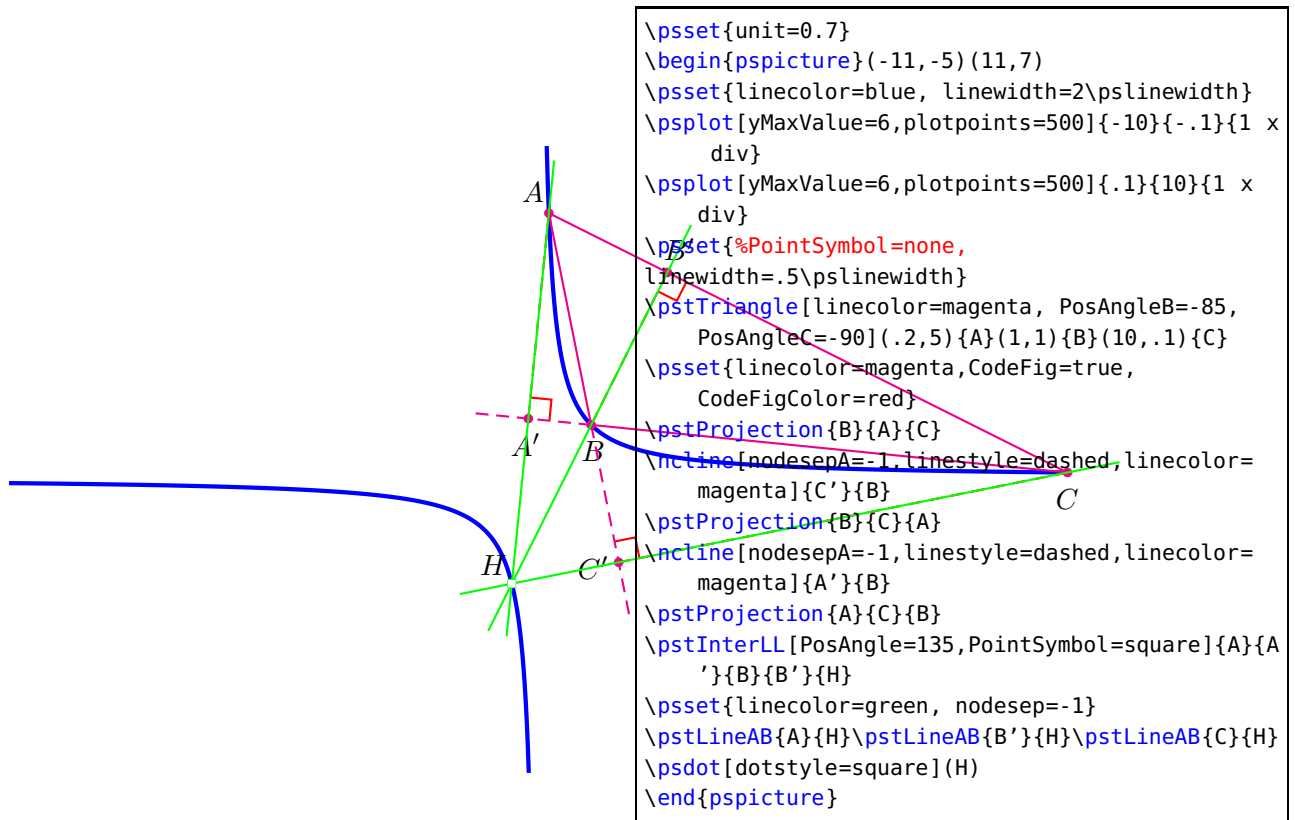
```

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

```

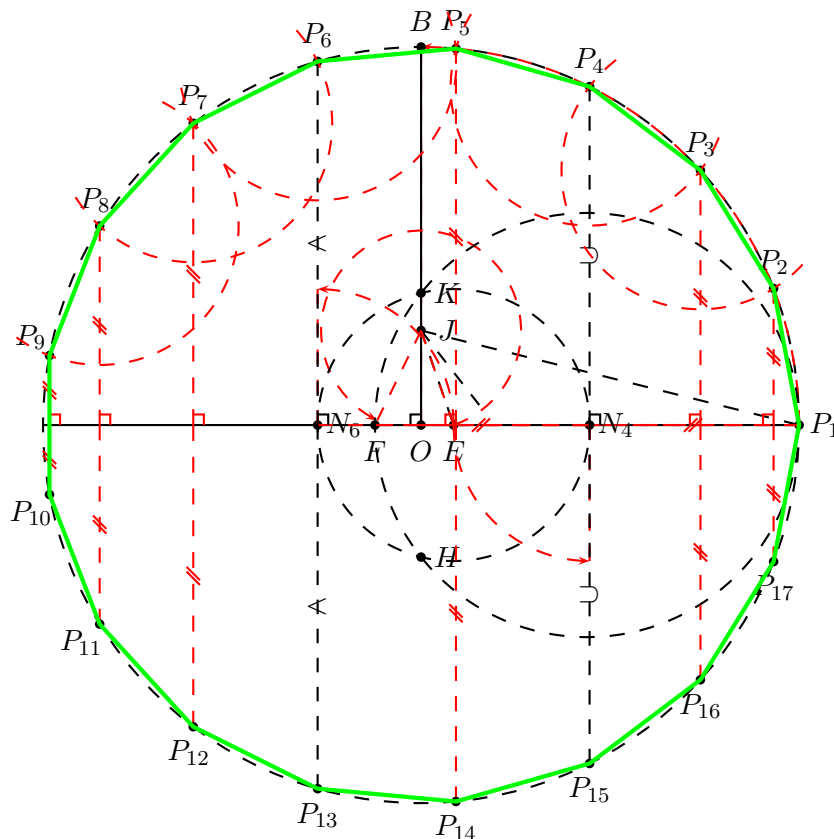
### A.5. Orthocenter and hyperbola

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



### 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  $2^{2^p} + 1$  sides, the following one has 257 sides!



```

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

```

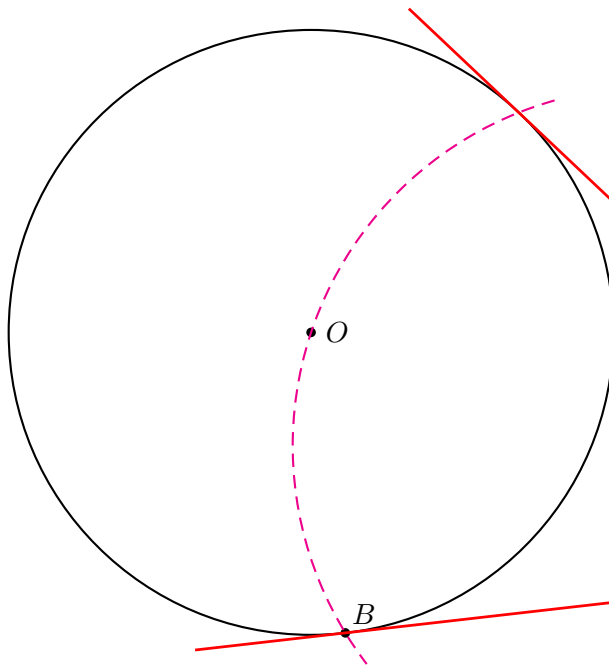
```

\pstRightAngle[linestyle=solid]{P_1}{N_6}{P_6}
\pstRightAngle[linestyle=solid]{P_1}{N_4}{P_4}
\pstBissectBAC[PosAngle=90, linestyle=none]{P_4}{O}{P_6}{P_5}
\pstInterCC[PosAngleB=90, PointSymbolA=none, PointNameA=none]{O}{P_1}{P_4}{P_5}{H}{P_3}
\pstInterCC[PosAngleB=90, PointSymbolA=none, PointNameA=none]{O}{P_1}{P_3}{P_4}{H}{P_2}
\pstInterCC[PosAngleA=90, PointSymbolB=none, PointNameB=none]{O}{P_1}{P_6}{P_5}{P_7}{H}
\pstInterCC[PosAngleA=100, PointSymbolB=none, PointNameB=none]{O}{P_1}{P_7}{P_6}{P_8}{H}
\pstInterCC[PosAngleA=135, PointSymbolB=none, PointNameB=none]{O}{P_1}{P_8}{P_7}{P_9}{H}
\pstOrtSym[PosAngle={-90,-90,-90,-100,-135}, PointName={P_{17},P_{16},P_{14},P_{12},P_{11},P_{10}}]
{O}{P_1}{P_2,P_3,P_5,P_7,P_8,P_9}[P_{17},P_{16},P_{14},P_{12},P_{11},P_{10}]
\pspolygon[linecolor=green, linestyle=solid, linewidth=2\pslinewidth]
(P_1)(P_2)(P_3)(P_4)(P_5)(P_6)(P_7)(P_8)(P_9)(P_{10})(P_{11})(P_{12})(P_{13})(P_{14})(P_{15})(P_{16})(P_{17})
\end{pspicture}

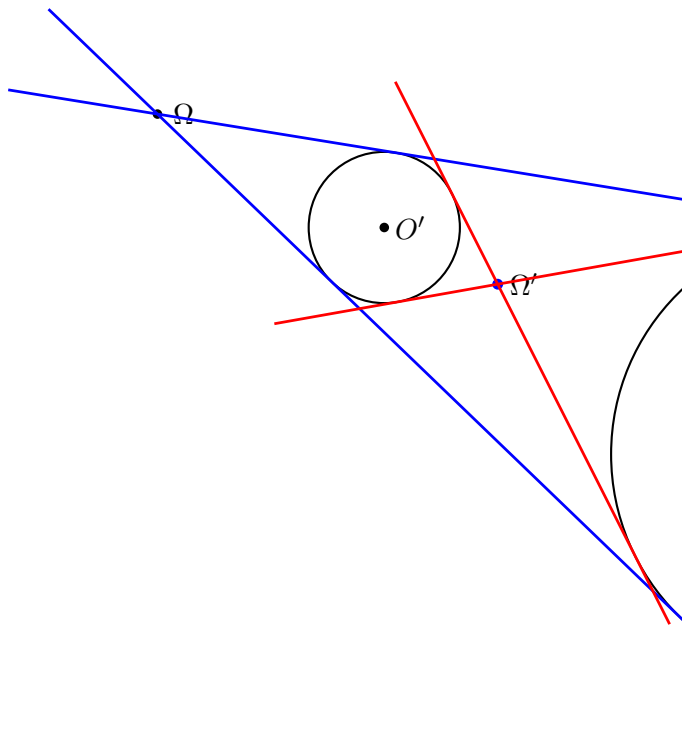
```

## A.7. Circles & tangents

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



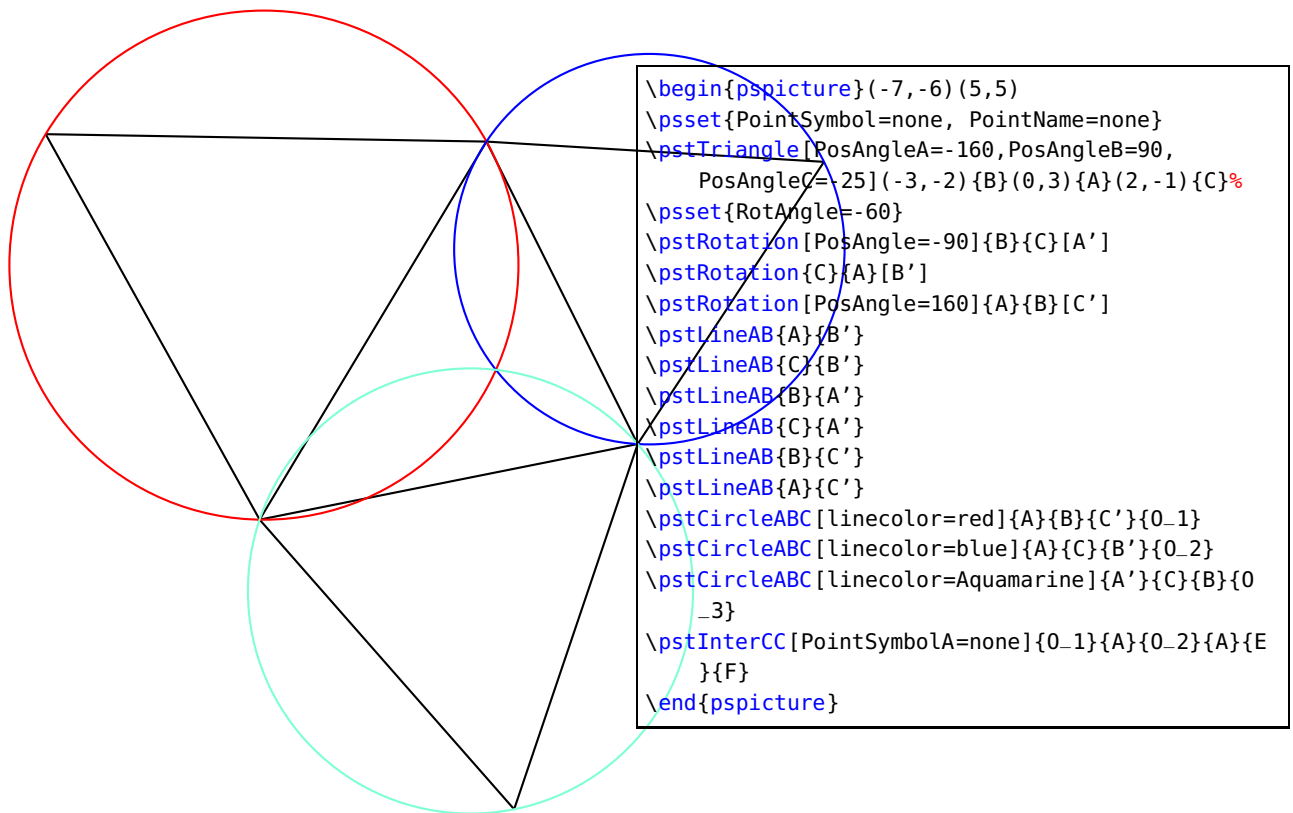
```
\begin{pspicture}(15,10)
\pstGeonode(5, 5){O}(14,2){M}
\pstCircleOA[Radius=\pstDistVal{4}]{O}{}
\pstMiddleAB[PointSymbol=none, PointName=none]{O}{}{M}{O'}
\pstInterCC[RadiusA=\pstDistVal{4}, DiameterB=\pstDistAB{O}{M},
CodeFigB=true, CodeFigColor=magenta,
PosAngleB=45]{O}{}{O'}{}{A}{B}
\psset{linecolor=red,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



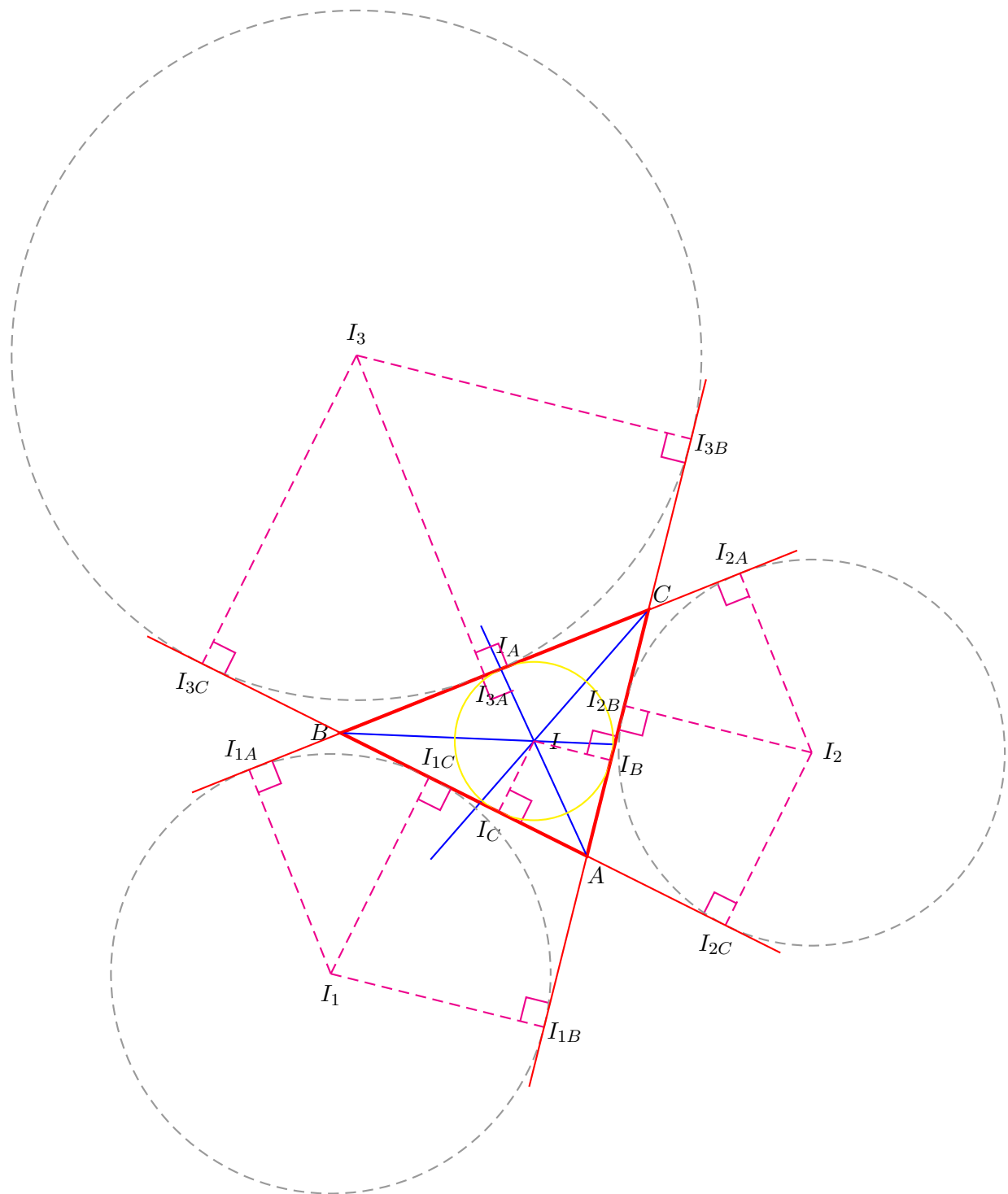
```
\begin{pspicture}(-2,0)(13,9)
\pstGeonode(9,3){O}(3,6){O'}\psset{PointSymbol=
none, PointName=none}
\pstCircleOA[Radius=\pstDistVal{3}]{O}{}
\pstCircleOA[Radius=\pstDistVal{1}]{O'}{}
\pstInterLC[Radius=\pstDistVal{3}]{O}{}{O'}{}{M}{}{toto}
\pstInterLC[Radius=\pstDistVal{1}]{O}{}{O'}{}{M}{}{toto}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

## A.8. Fermat's point

Drawing of Manuel Luque.



## A.9. Escribed and inscribed circles of a triangle



```

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

```

```

\psset{linecolor=magenta, linestyle=dashed}
\pstProjection{A}{B}{I}[I_C]
\pstLineAB{I}{I_C}\pstRightAngle[linestyle=solid]{A}{I_C}{I}
\pstProjection{A}{C}{I}[I_B]
\pstLineAB{I}{I_B}\pstRightAngle[linestyle=solid]{C}{I_B}{I}
\pstProjection[PosAngle=80]{C}{B}{I}[I_A]
\pstLineAB{I}{I_A}\pstRightAngle[linestyle=solid]{B}{I_A}{I}
\pstCircleOA[linecolor=yellow, linestyle=solid]{I}{I_A}
\psset{linecolor=magenta, linestyle=none}
\pstOutBissectBAC[PointSymbol=none,PointName=none]{C}{A}{B}{AOB}
\pstOutBissectBAC[PointSymbol=none,PointName=none]{A}{B}{C}{BOB}
\pstOutBissectBAC[PointSymbol=none,PointName=none]{B}{C}{A}{COB}
\pstInterLL[PosAngle=-90]{A}{AOB}{B}{BOB}{I_1}
\pstInterLL{A}{AOB}{C}{COB}{I_2}
\pstInterLL[PosAngle=90]{C}{COB}{B}{BOB}{I_3}
\psset{linecolor=magenta, linestyle=dashed}
\pstProjection[PointName=I_1C]{A}{B}{I_1}[I1C]
\pstLineAB{I_1}{I1C}\pstRightAngle[linestyle=solid]{I_1}{I1C}{A}
\pstProjection[PointName=I_1B]{A}{C}{I_1}[I1B]
\pstLineAB{I_1}{I1B}\pstRightAngle[linestyle=solid]{A}{I1B}{I_1}
\pstProjection[PointName=I_1A]{C}{B}{I_1}[I1A]
\pstLineAB{I_1}{I1A}\pstRightAngle[linestyle=solid]{I_1}{I1A}{C}
\pstProjection[PointName=I_2B]{A}{C}{I_2}[I2B]
\pstLineAB{I_2}{I2B}\pstRightAngle[linestyle=solid]{A}{I2B}{I_2}
\pstProjection[PointName=I_2C]{A}{B}{I_2}[I2C]
\pstLineAB{I_2}{I2C}\pstRightAngle[linestyle=solid]{I_2}{I2C}{A}
\pstProjection[PointName=I_2A]{B}{C}{I_2}[I2A]
\pstLineAB{I_2}{I2A}\pstRightAngle[linestyle=solid]{C}{I2A}{I_2}
\pstProjection[PointName=I_3A]{C}{B}{I_3}[I3A]
\pstLineAB{I_3}{I3A}\pstRightAngle[linestyle=solid]{C}{I3A}{I_3}
\pstProjection[PointName=I_3C]{A}{B}{I_3}[I3C]
\pstLineAB{I_3}{I3C}\pstRightAngle[linestyle=solid]{A}{I3C}{I_3}
\pstProjection[PointName=I_3B]{C}{A}{I_3}[I3B]
\pstLineAB{I_3}{I3B}\pstRightAngle[linestyle=solid]{I_3}{I3B}{A}
\psset{linecolor=yellow, linestyle=solid}
\pstCircleOA{I_1}{I1C} \pstCircleOA{I_2}{I2B} \pstCircleOA{I_3}{I3A}
\psset{linecolor=red, linestyle=solid, nodesepA=-1, nodesepB=-1}
\pstLineAB{I1B}{I3B}\pstLineAB{I1A}{I2A}\pstLineAB{I2C}{I3C}
\end{pspicture}

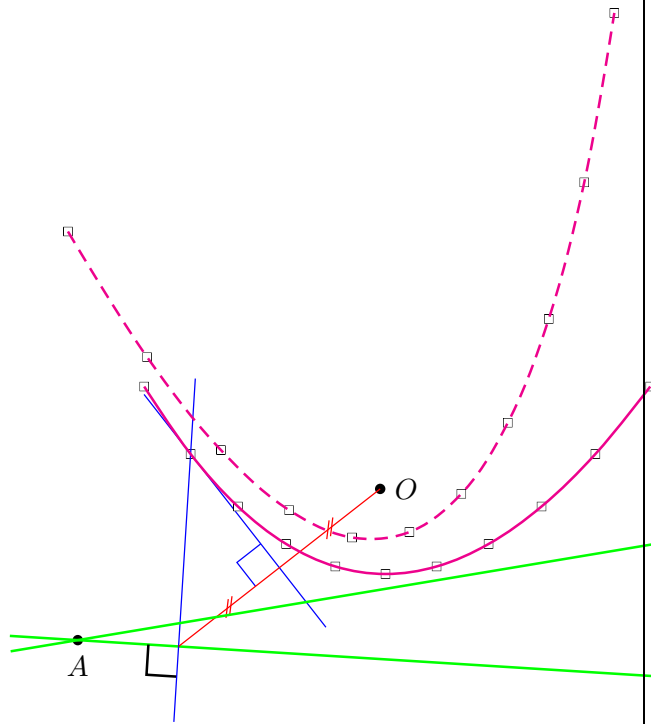
```



## B. Some locus points

### B.1. Parabola

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



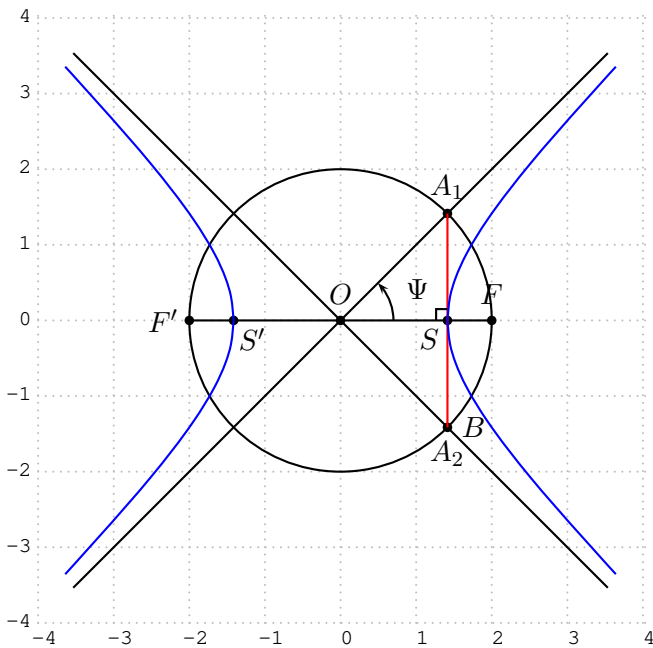
```

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

```

### B.2. Hyperbola

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



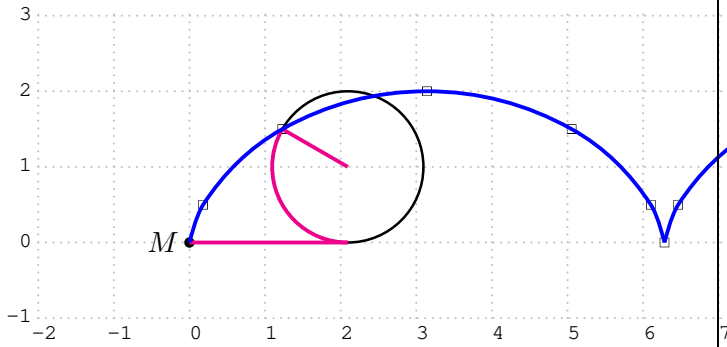
```

\begin{pspicture}[showgrid](-4,-4)(4,4)
\newcommand\Sommet{1.4142135623730951 } \
  \newcounter{i} \setcounter{i}{1}
\newcommand\PosFoyer{2 } \newcommand\HypAngle{0}
\newcounter{CoefDiv}\setcounter{CoefDiv}{20}
\newcounter{Inc}\setcounter{Inc}{2} \newcounter{n}
  \setcounter{n}{2}
\newcommand\Ri{ \PosFoyer \Sommet sub \arabic{i}\
  space\arabic{CoefDiv}\space div add }
\newcommand\Rii{\Ri \Sommet 2 mul add }
\pstGeonode[PosAngle=90]{0}(\PosFoyer;\HypAngle){
  F}
\pstSym0[PosAngle=180]{0}{F}\pstLineAB{F}{F'} \
  \pstCircleOA{0}{F}
\pstGeonode[PosAngle=-135](\Sommet;\HypAngle){S}
\pstGeonode[PosAngle=-45](-\Sommet;\HypAngle){S'}
\pstRotation[RotAngle=90, PointSymbol=none]{S}{0}
  {B}
\pstInterLC[PosAngleA=90, PosAngleB=-90]{S}{B}{0}
  {F}{A_1}{A_2}
\pstLineAB[nodesepA=-3,nodesepB=-5]{A_1}{0}\
  \pstLineAB[nodesepA=-3,nodesepB=-5]{A_2}{0}
\pstMarkAngle[LabelSep=.8,MarkAngleRadius=.7,
  arrows=->,LabelSep=1.1]{F}{0}{A_1}{\Psi}
\ncline[linicolor=red]{A_1}{A_2} \pstRightAngle[
  RightAngleSize=.15]{A_1}{S}{0}
\psset{PointName=none}
\whiledo{\value{n}<8}{%
  \psset{RadiusA=\pstDistVal{\Ri},RadiusB=\
    \pstDistVal{\Rii},PointSymbol=none}
  \pstInterCC{F}{F'}{M\arabic{n}}{P\arabic{n}
    }}
  \pstInterCC{F'}{F}{M'\arabic{n}}{P'\arabic{
    n}}
  \stepcounter{n}\addtocounter{i}{\value{Inc}}
  \addtocounter{Inc}{\value{Inc}}}%% fin de
  \whiledo
\psset{linicolor=blue}
\pstGenericCurve[GenCurvFirst=S]{M}{2}{7}
\pstGenericCurve[GenCurvFirst=S]{P}{2}{7}
\pstGenericCurve[GenCurvFirst=S']{M'}{2}{7}
\pstGenericCurve[GenCurvFirst=S']{P'}{2}{7}
\end{pspicture}

```

### B.3. Cycloid

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



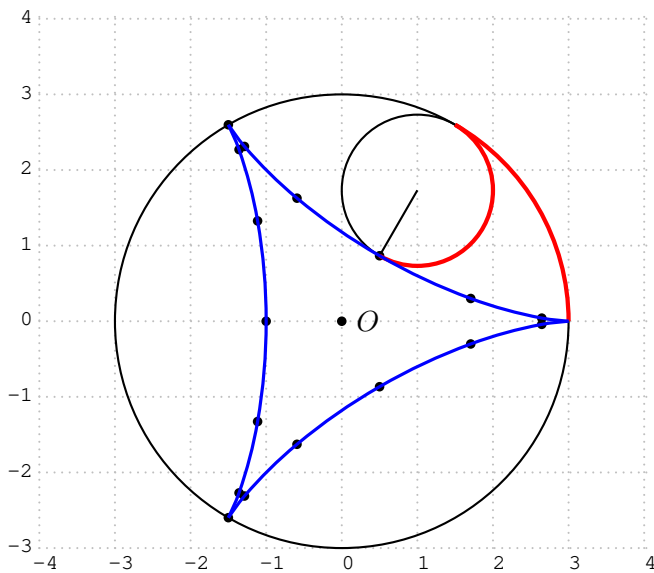
```

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

```

### B.4. Hypocycloids (Astroid and Deltoid)

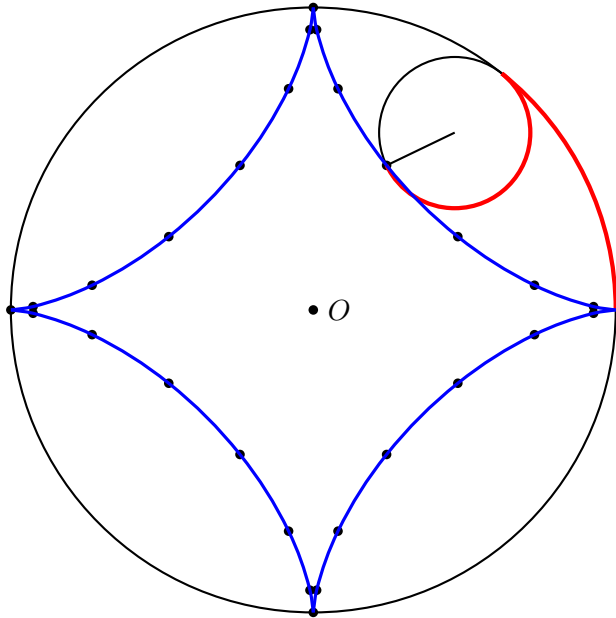
A wheel rolls inside a circle, and depending of the radius ratio, it is an astroid, a deltoid and in the general case hypo-cycloids.



```

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

```



```

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

```

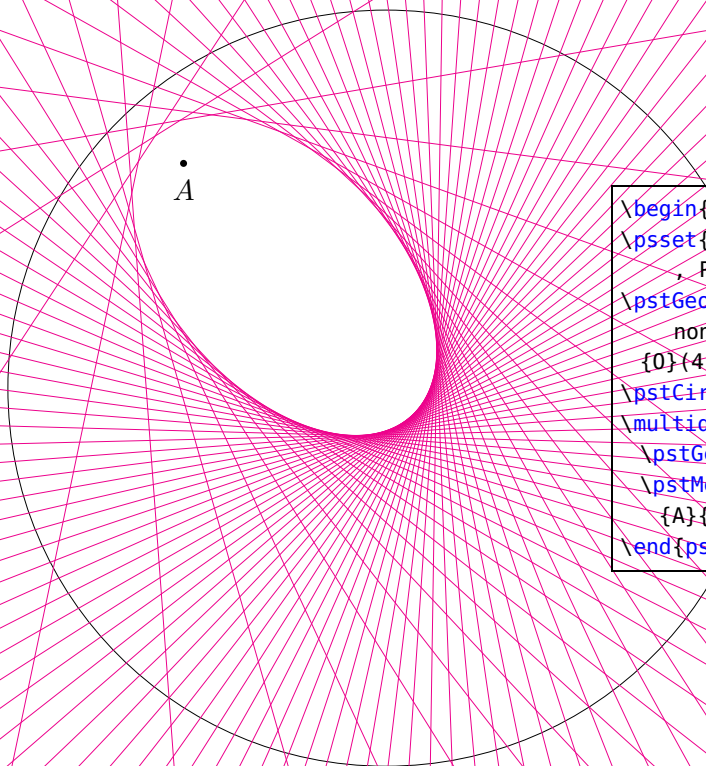
## C. Lines and circles envelope

### C.1. Conics

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

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

(figure of O. Reboux).



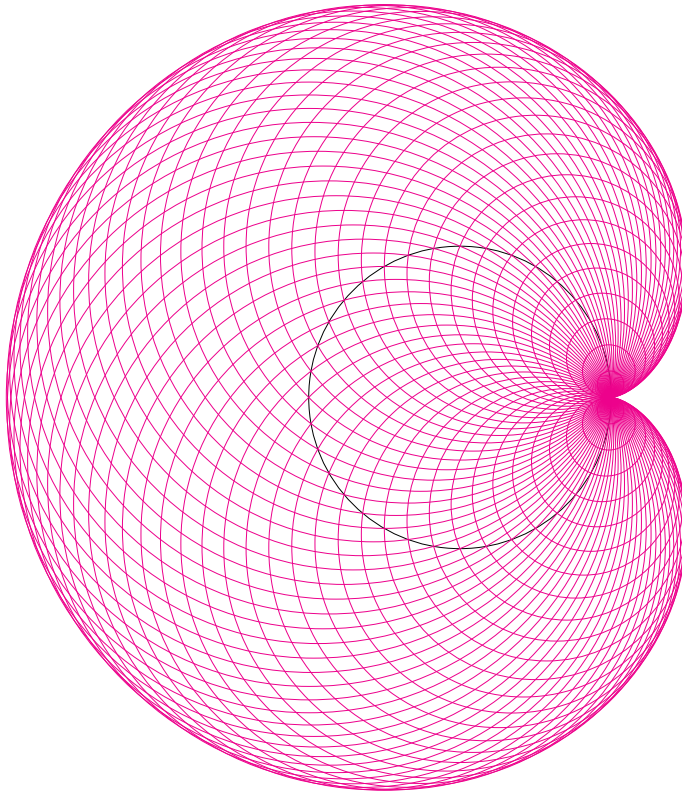
```

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

```

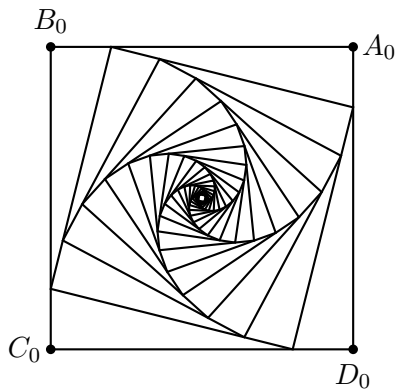
## C.2. Cardioid

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



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

## D. Homotethy and fractals



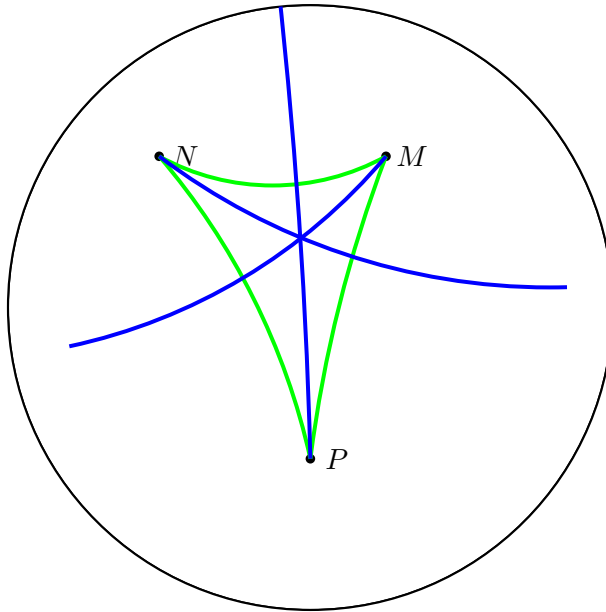
```

\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}{%
  \pstHom0[PosAngle=0]{B_\i}{A_\i}[A_\n]
  \pstHom0[PosAngle=90]{C_\i}{B_\i}[B_\n]
  \pstHom0[PosAngle=180]{D_\i}{C_\i}[C_\n]
  \pstHom0[PosAngle=270]{A_\i}{D_\i}[D_\n]
  \pspolygon(A_\n)(B_\n)(C_\n)(D_\n)}% fin multido
\end{pspicture}

```



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



```

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

```

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

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

*Continued on next page*

*Continued from previous page*

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

## References

- [1] Victor Eijkhout: *T<sub>E</sub>X by Topic – A T<sub>E</sub>Xnician Reference*, 1st ed., DANTE e.V and Lehmanns Media, Heidelberg and Berlin, 2014.
- [2] Denis Girou: “Présentation de PSTricks”, *Cahier GUTenberg*, 16 (1994), 21–70.
- [3] Michel Goossens et al.: *The L<sup>A</sup>T<sub>E</sub>X Graphics Companion*, 2nd ed., Addison-Wesley Publishing Company, Boston, Mass., 2007.
- [4] Nikolai G. Kollock: *PostScript richtig eingesetzt: vom Konzept zum praktischen Einsatz*, IWT, Vaterstetten, 1989.
- [5] Timothy Van Zandt, Herbert Voß, Rolf Niepraschk: *The Multido package, A loop facility for Generic T<sub>E</sub>X*, version 1.42, [CTAN:macros/latex/multido](https://ctan.org/ctan/packages/macros/latex/multido) (visited on 9/1/2018).
- [6] Herbert Voß: “Die mathematischen Funktionen von Postscript”, *Die T<sub>E</sub>Xnische Komödie*, 1/02 (2002), 40–47.
- [7] — *PSTricks – Graphics and PostScript for L<sup>A</sup>T<sub>E</sub>X*, 1st ed., UIT, Cambridge – UK, 2011.
- [8] — *L<sup>A</sup>T<sub>E</sub>X quick reference*, 1st ed., UIT, Cambridge – UK, 2012.
- [9] — *PSTricks – Grafik für T<sub>E</sub>X und L<sup>A</sup>T<sub>E</sub>X*, 7th ed., DANTE e.V and Lehmanns Media, Heidelberg and Hamburg, 2016.
- [10] — *Presentations with L<sup>A</sup>T<sub>E</sub>X*, 2nd ed., DANTE e.V and Lehmanns Media, Heidelberg and Berlin, 2019.

- [11] Timothy Van Zandt, Denis Girou: "Inside PStricks", *TUGboat*, 15 (1994), 239–246.