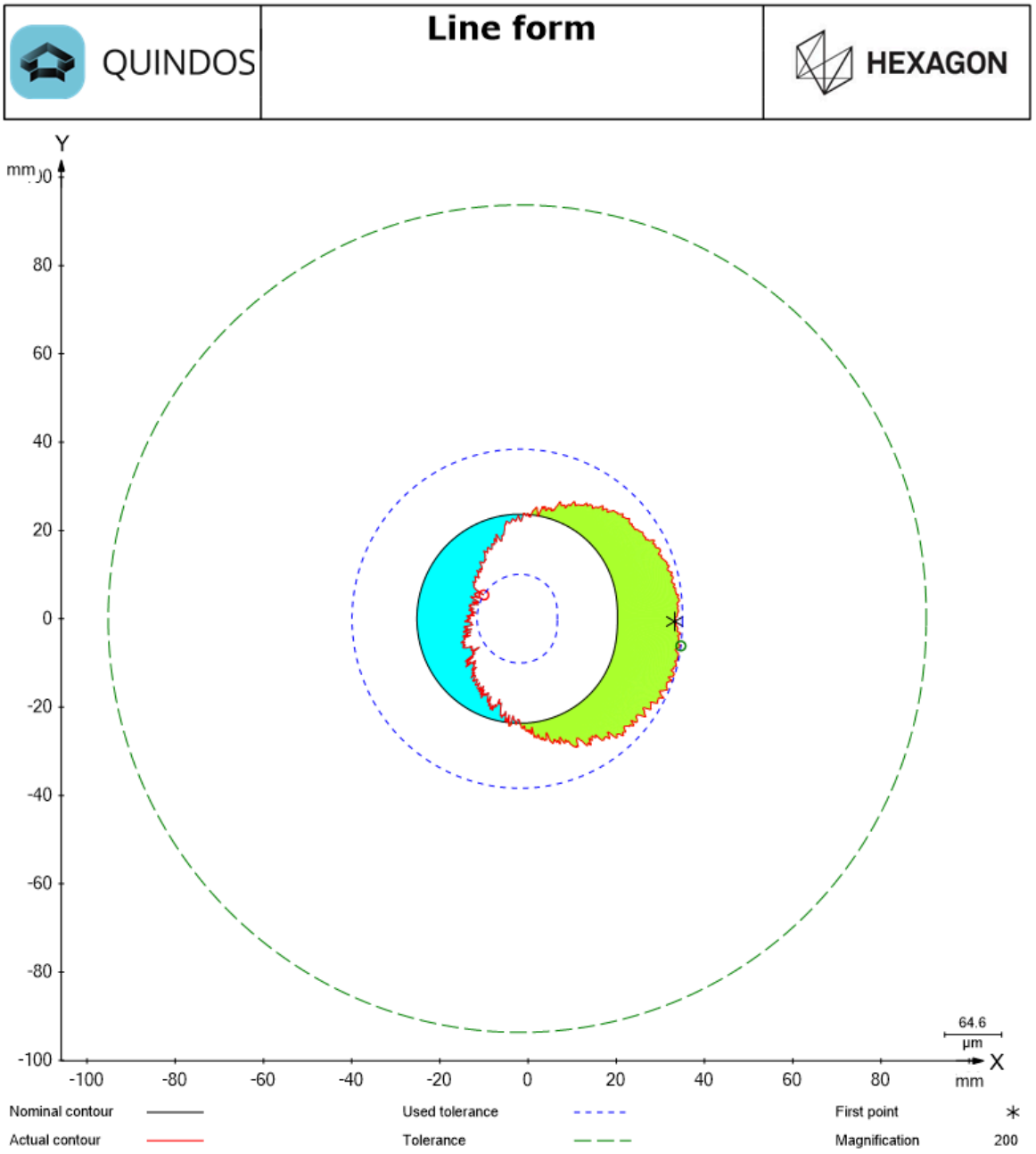
Quindos作图指南

Quindos内置了两套作图指令集，分别以**DRWPLY**和**PLS_ADDCRVL**为核心，展开一系列相关的指令。使用下来，PLS系列的指令与中文字符的配合更好，并且有类似【图层】的设计，可以更方便地更改、组合调整整体图形。目前更建议全套使用PLS系列指令进行作图，因为两者所需要定义的作图区域，在指令和逻辑上并不相同。两者可以混用但是可能会造成一些不便。

指南将以轮廓度作图为案例，讲解两套指令的使用方式。



以官方轮廓度图形报告（部分截选）为例子分析得出，进行自定义作图时，我们要了解的几块核心知识：

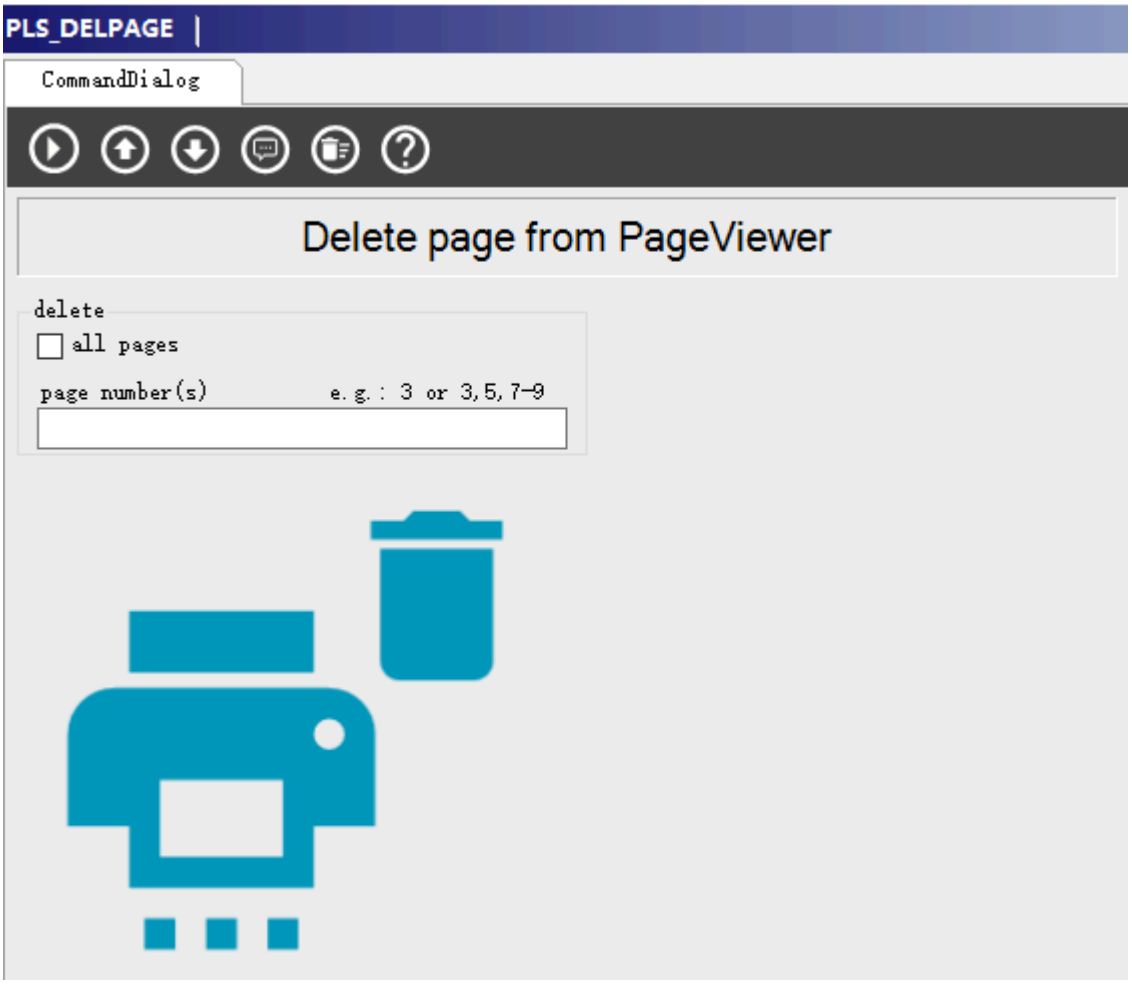
1. 将原始测量**图形轮廓、偏差轮廓**画出。以**不同颜色**填充正负偏差。
2. 找到偏差里**最大、最小以及测量起始点**，在**相应位置**画出标记。
3. 以原始轮廓为基础，**画出公差带**。再根据最大最小点的位置，**画出公差带已使用的范围**。（以轮廓中心为基础，同心放大的两个轮廓，将所有偏差包络）
4. 画出**坐标系与比例尺**，以及**图例**。
5. 制作**表头**，插入**文字与图片**。

在上述几个知识点中，还含有一个隐藏要求：***正确掌握作图的缩放比例***。放大比例不受控制，从第二步开始就会变得难以完成。

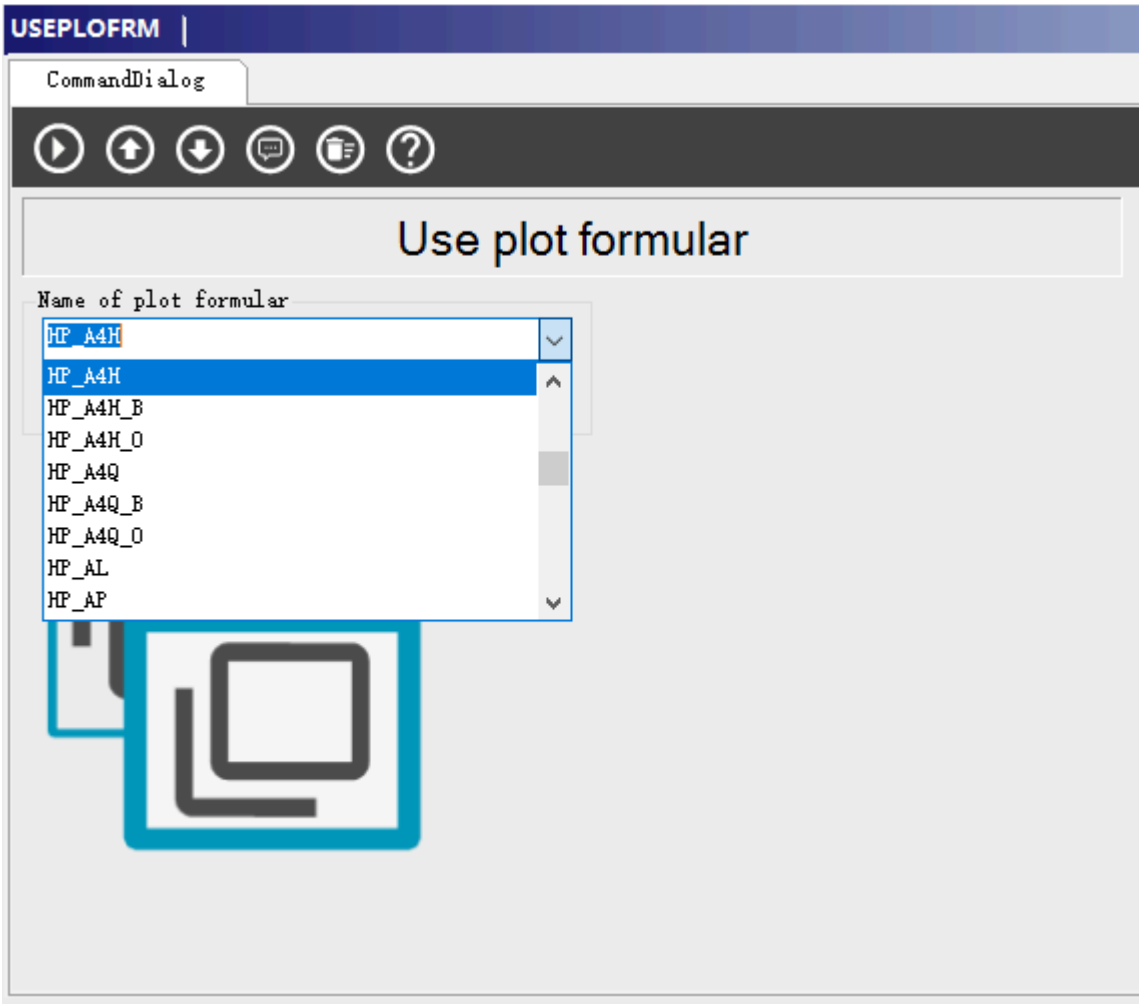
以下部分为作图的核心步骤与指令介绍。两个指令集不通用的地方，将分开讲解；没有明确标出两种方案的即为通用的部分。

创建空白页：

PLS_DELPAGE：该指令可以删除目前已有的PDF页。勾选【all pages】将删除所有现有的报告页，下方【page number】可以指定删除指定页。如果不进行任何设置而执行命令，系统将会删除最后的一页。



USEPLOFRM：该指令可以根据用户选择，创建大小与方向不同的空白页面。常用的页面可以在HP_A4H和HP_A4Q中选择，前者是A4纸纵向，后者是A4纸横向。



定义绘图区域：

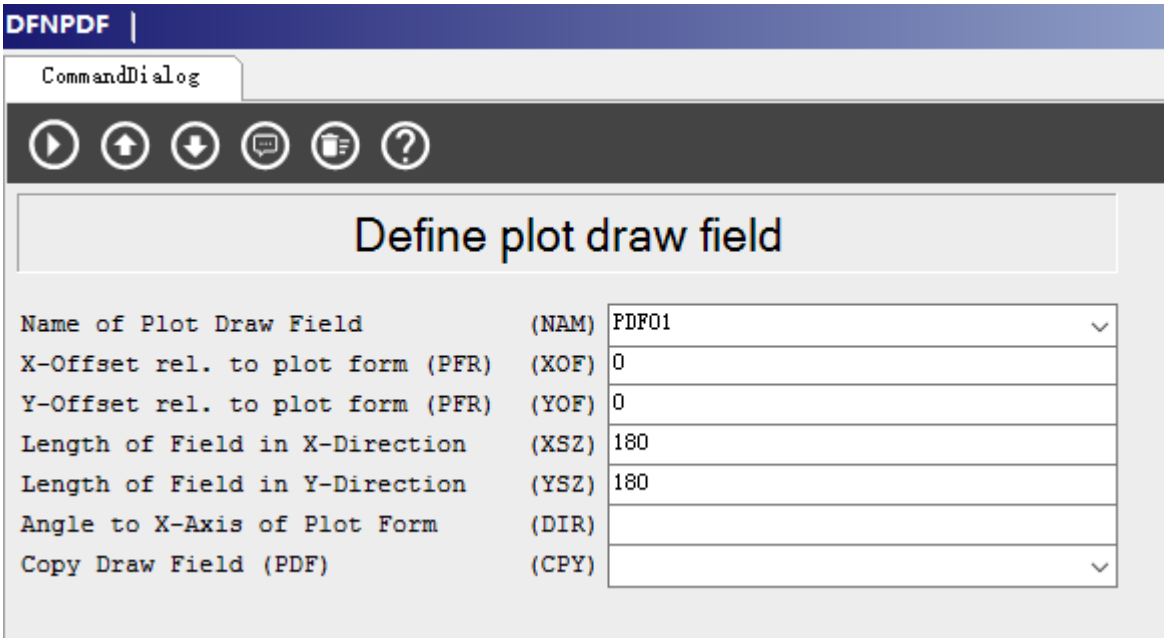
DRWPLY系列：

DFNPDF：通过参数确定作图框的范围和角度。

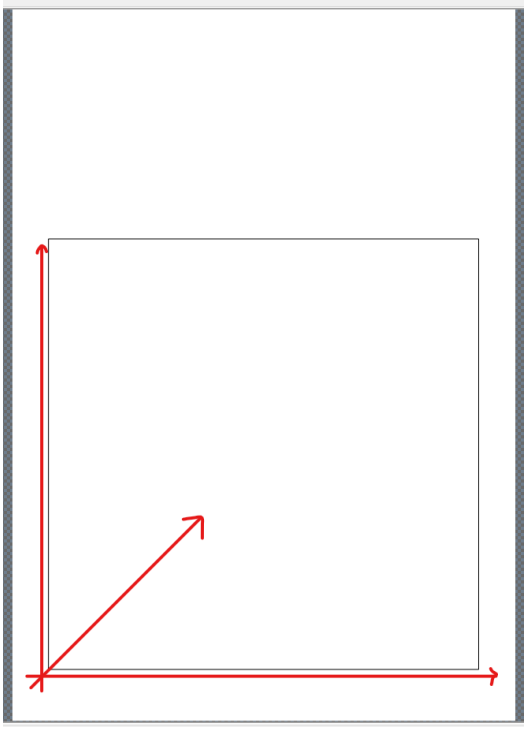
定义作图区域 复制代码

```
DFNPDF          (NAM=PDF01, XOF=0, YOF=0, XSZ=180, YSZ=180)
```

- NAM：定义作图框的名称，以便后续激活调用
- XOF：起始点X坐标
- YOF：起始点Y坐标
- XSZ：X方向宽度
- YSZ：Y方向宽度
- DIR：与X轴夹角
- CPY：复制已定义区域的参数



从下图可以看出，方框即为上个指令中180*180的区域。**DFNPDF**的起始点为**页面的左下角**。（与PLS系列不同）



补充：

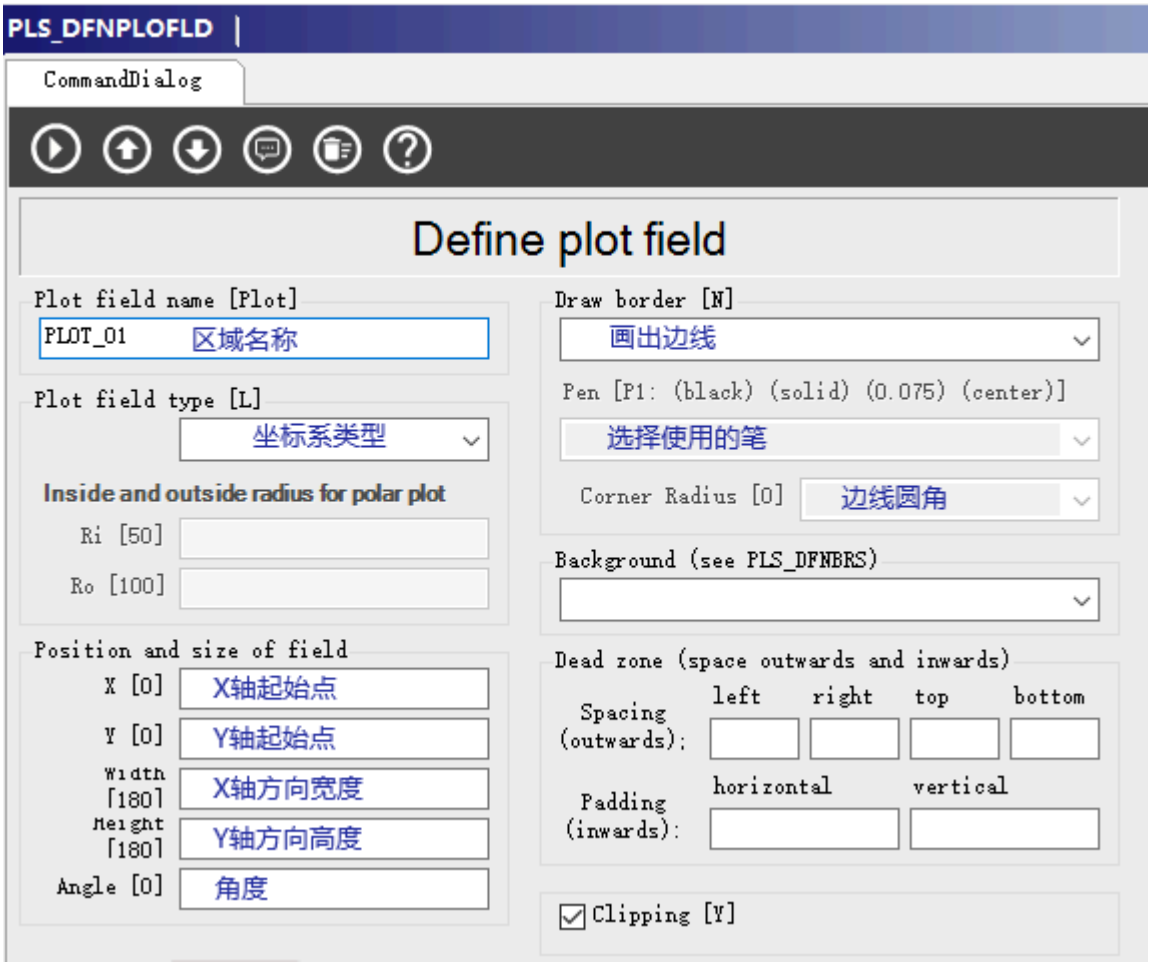
一次定义了多个PDF后，可以使用**USEPDF**来激活任一区域，系统默认的激活区域为最后一个定义的区域。

PLS_ADDCRVL系列：

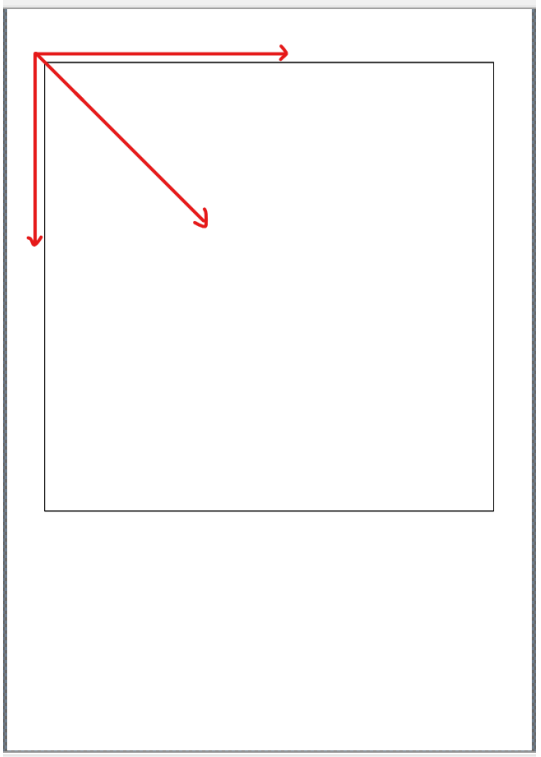
PLS_DFNPLOFLD：如图设置作图区域参数。

定义作图区域 复制代码

```
PLS_DFNPLOFLD   (NAM=PLOT_01)
```



从下图可以看出方框即为上个指令中180*180的区域。**PLS_DFNPLOFLD**的起始点为**页面的左上角**。（与DRWPLY系列不同）

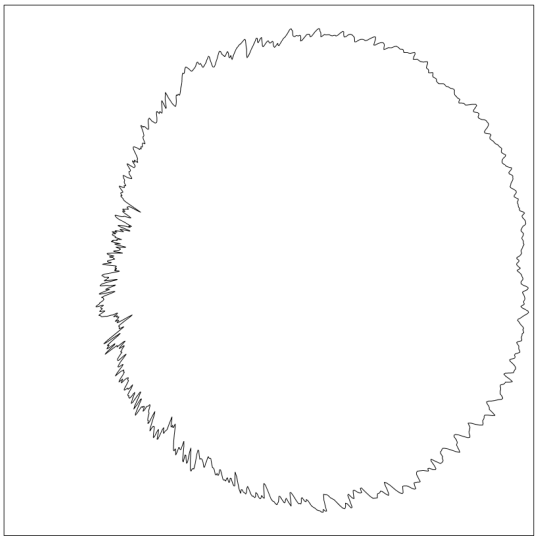


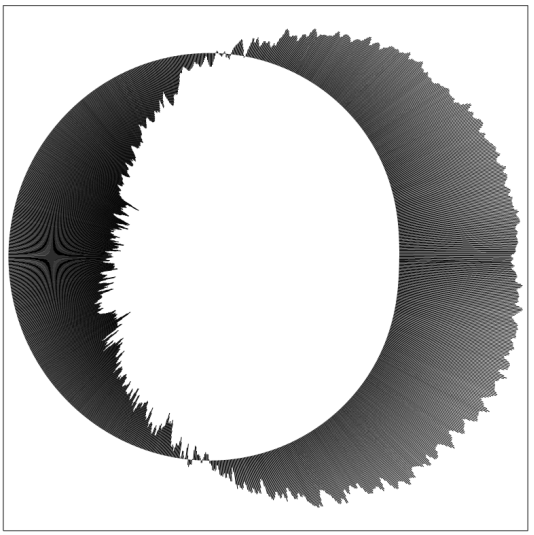
补充：

PLS_DFNFRAME也可以设定作图区域。设置与**PLS_DFNPLOFLD**相同，但是其定义的FRAME并不支持作出轮廓图形，更适合用于制作表头，图例等元素。详细介绍请参照后续**【画出报告的标题】**章节。

画出轮廓-基础：

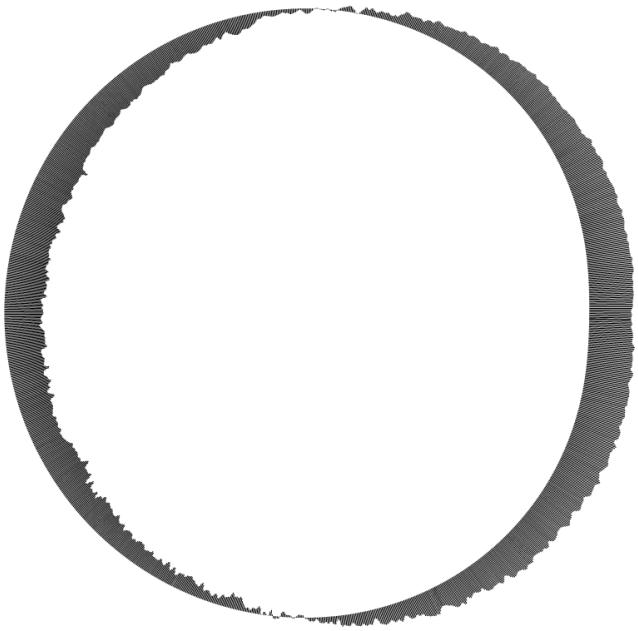
作图时，需要按照一定的**偏差放大倍率**和**图形缩放比例**作出元素APT点的轮廓和偏差。



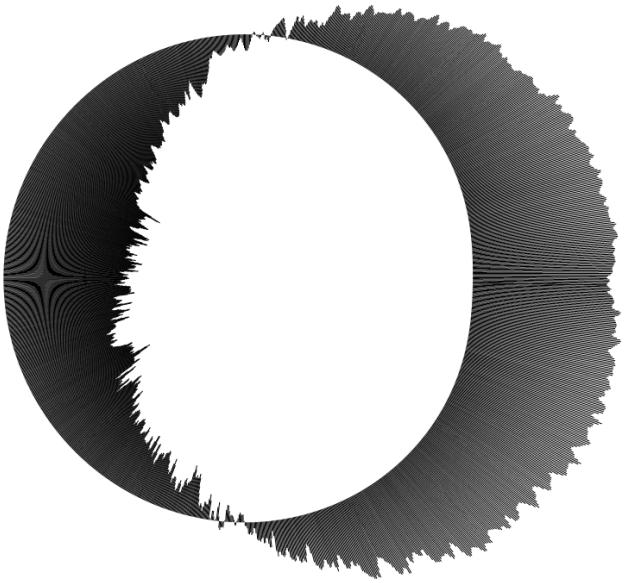


重点：上述两个放大并非同一个概念。

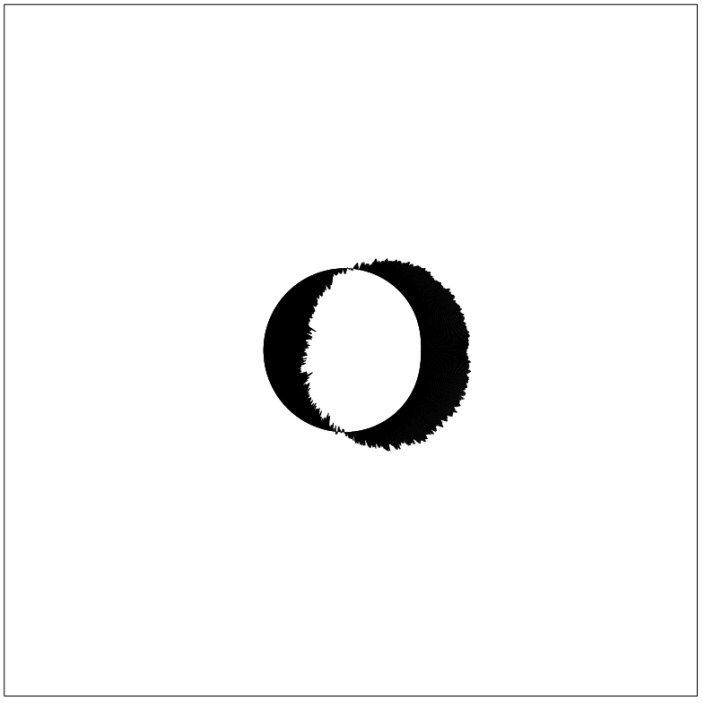
- **偏差放大倍率**：将偏离理论值的距离通过几何倍率放大，使偏差的形状更易表现，而图形本身的轮廓不会发生变化。
    - 低倍率偏差放大：



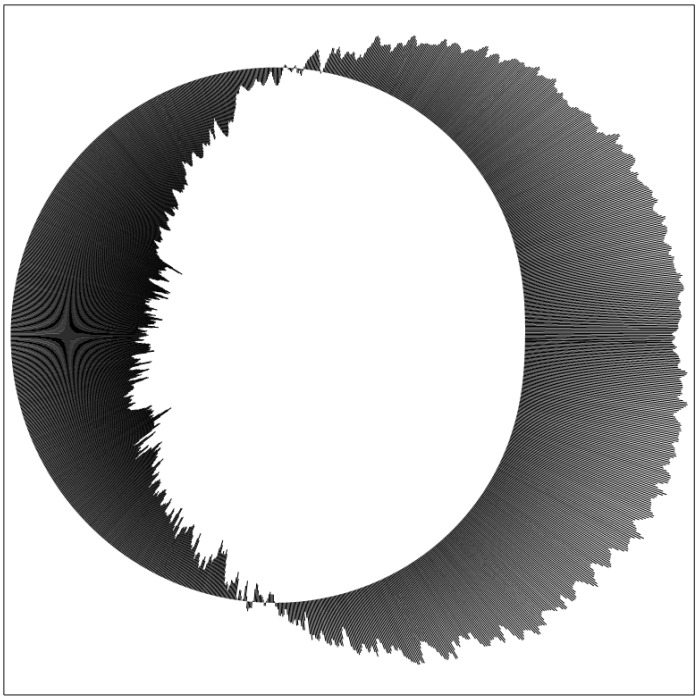
    - 高倍率偏差放大：



- **图形缩放比率**：图形在作图区域的位置和尺寸大小
  - 居中低倍缩放



  - 居中高倍缩放

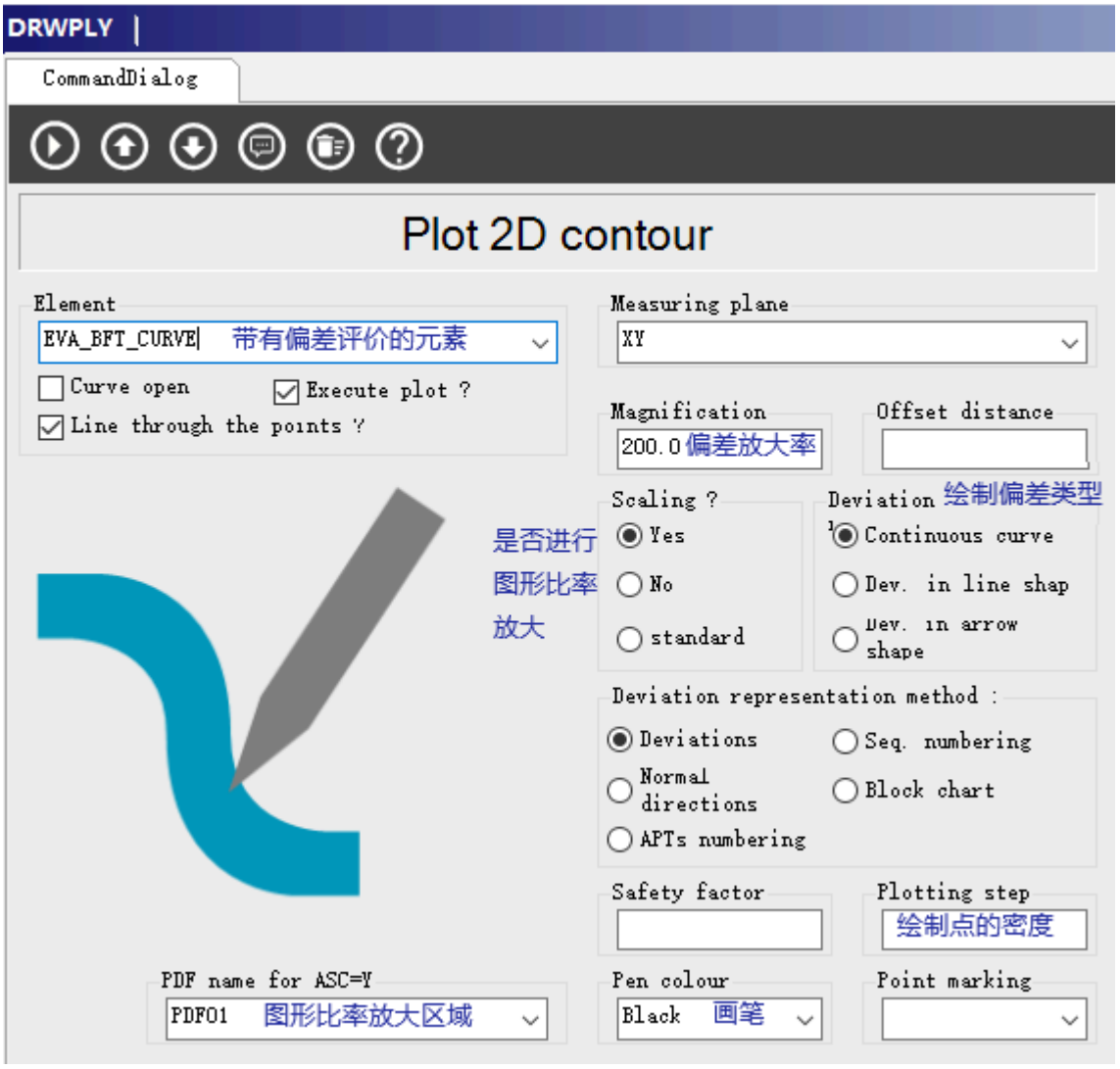


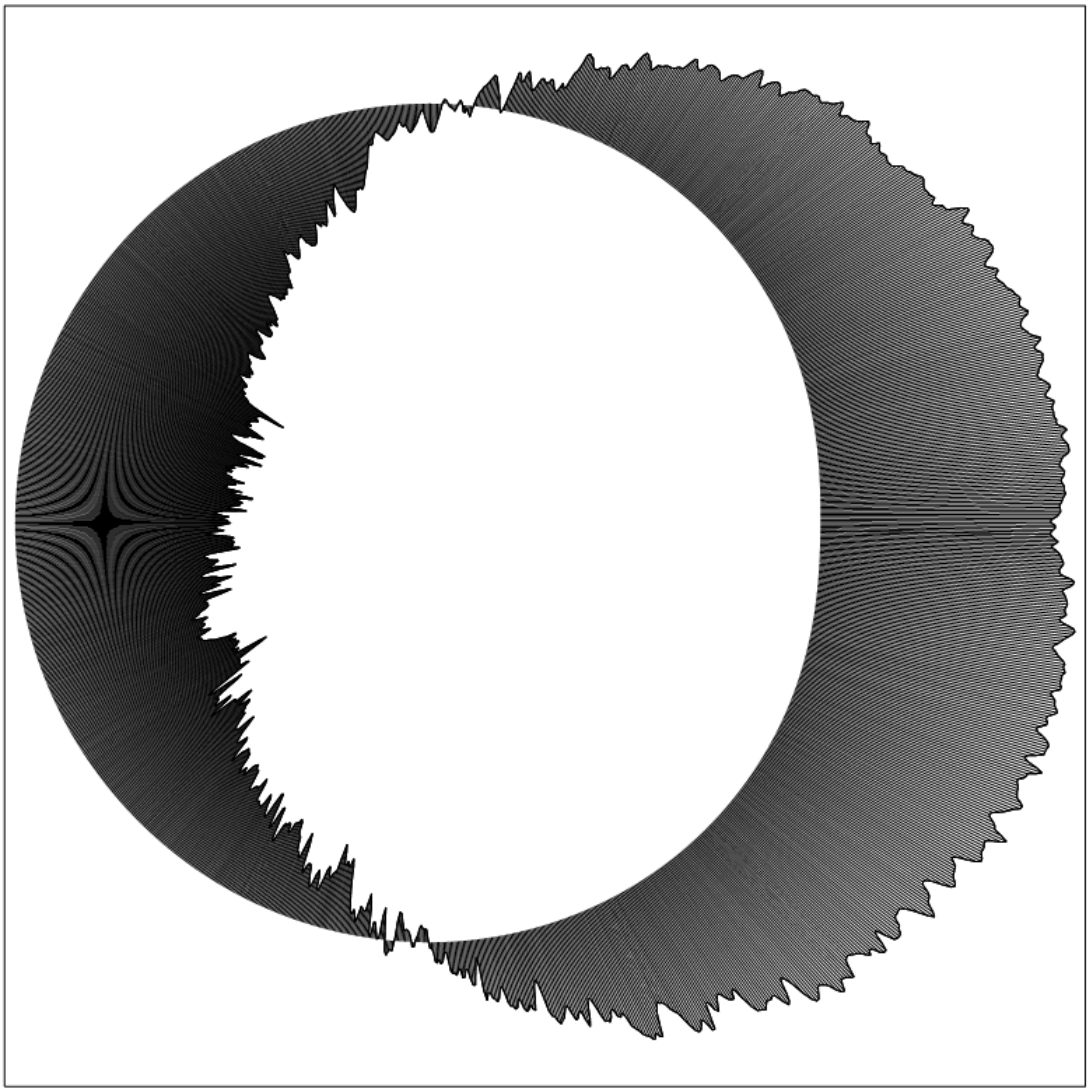
DRWPLY系列：

DRWPLY：使用方式如下。当**Scaling**选择为**Yes**（缩放到最大）或**Standard**（按预设缩放）时，系统会综合APT点坐标、最大最小偏差以及作图区域大小，进行**自动缩放**。

画出轮廓 复制代码

```
DRWPLY          (NAM=EVA_BFT_CURVE, ASC=Y, DEV=Y, FAC=200.0, DRP=1,
PEN=1, OPN=N, A_O=XY, PDF=PDF01)
```


DRWPLY
CommandDialog
Plot 2D contour
Element
EVA_BFT_CURVE
带有偏差评价的元素
Curve open
Execute plot ?
Line through the points ?
Measuring plane
XY
Magnification
200.0
偏差放大率
Offset distance
Scaling ?
Yes
No
standard
是否进行
图形比率
放大
Deviation
绘制偏差类型
Continuous curve
Dev. in line shap
Dev. in arrow shape
Deviation representation method :
Deviations
Seq. numbering
Normal directions
Block chart
APTs numbering
Safety factor
Plotting step
绘制点的密度
PDF name for ASC=Y
PDF01
图形比率放大区域
Pen colour
Black
画笔
Point marking

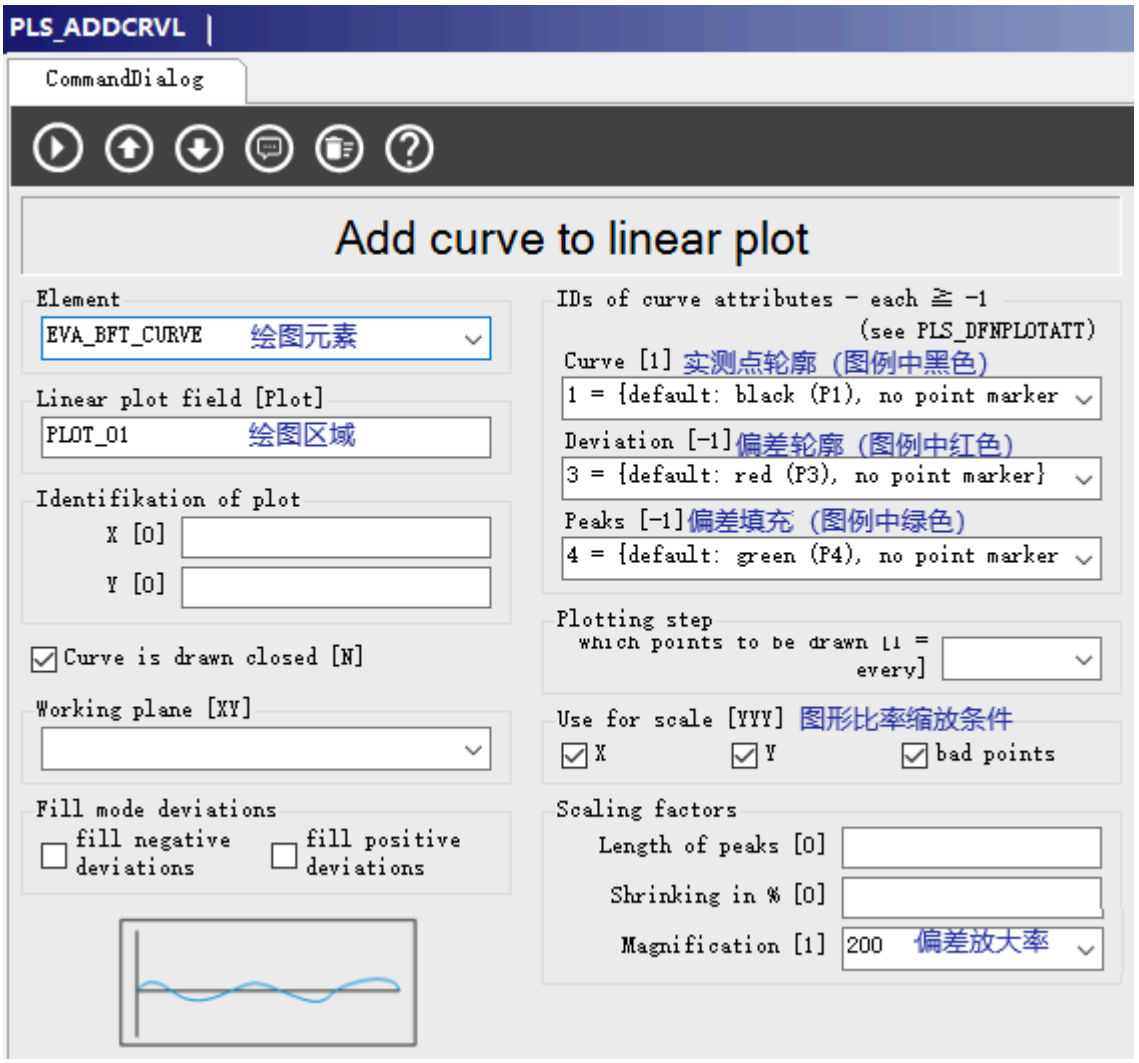


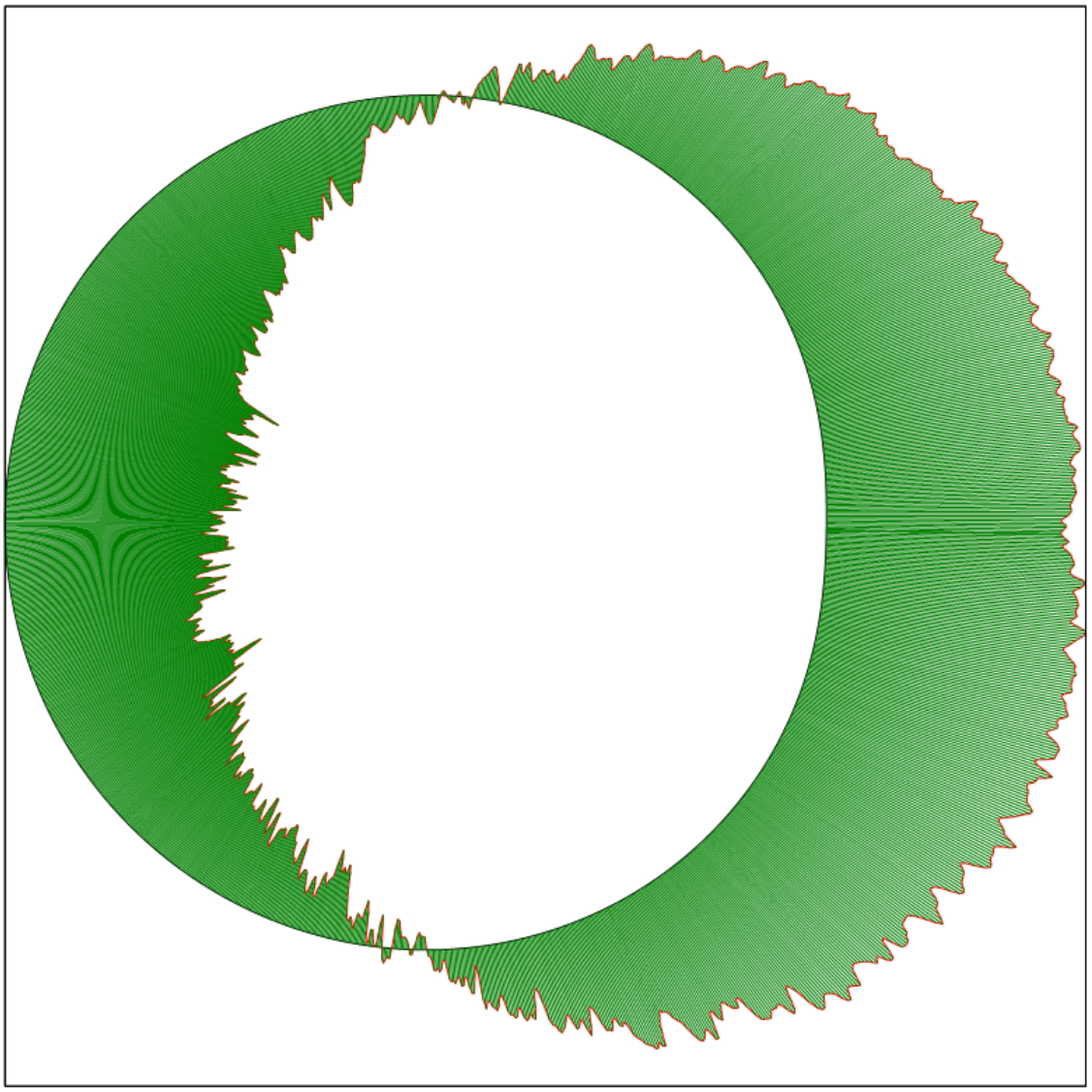
PLS_ADDCRVL系列：

PLS_ADDCRVL：如图进行参数设置。相比DRWPLY指令，该指令的优势在于便捷的画笔选择，支持一次性将多种元素以不同的颜色画出。不需要的元素，也可在画笔选项中选择-1={Do not draw}消除。

画出轮廓 复制代码

```
PLS_ADDCRVL     (ELE=EVA_BFT_CURVE, PFL=PLOT_01, CLS=Y, CVA=(1,3,4),
SCF=200, FLL= )
```

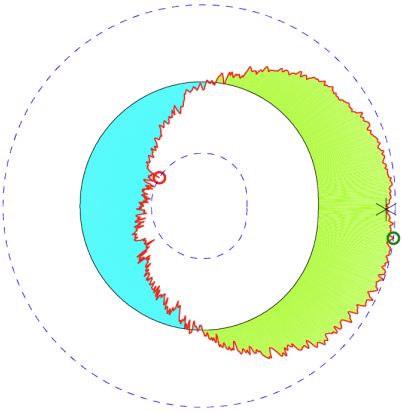

PLS_ADDCRVL
CommandDialog
Add curve to linear plot
Element
EVA_BFT_CURVE
绘图元素
Linear plot field [Plot]
PLOT_01
绘图区域
Identifikation of plot
X [0]
Y [0]
Curve is drawn closed [N]
Working plane [XY]
Fill mode deviations
fill negative deviations
fill positive deviations
IDs of curve attributes - each ≧ -1
(see PLS_DFNPLOTATT)
Curve [1] 实测点轮廓（图例中黑色）
1 = {default: black (P1), no point marker
Deviation [-1] 偏差轮廓（图例中红色）
3 = {default: red (P3), no point marker}
Peaks [-1] 偏差填充（图例中绿色）
4 = {default: green (P4), no point marker
Plotting step
which points to be drawn [1 = every]
Use for scale [YYY] 图形比率缩放条件
X
Y
bad points
Scaling factors
Length of peaks [0]
Shrinking in % [0]
Magnification [1]
200
偏差放大率

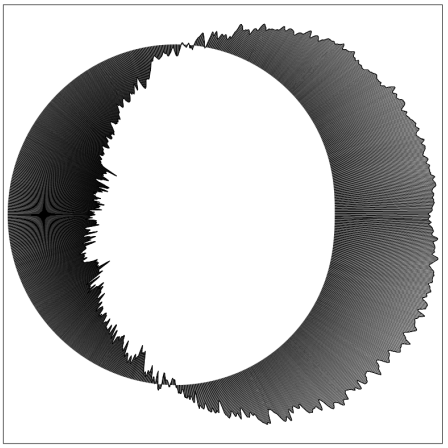


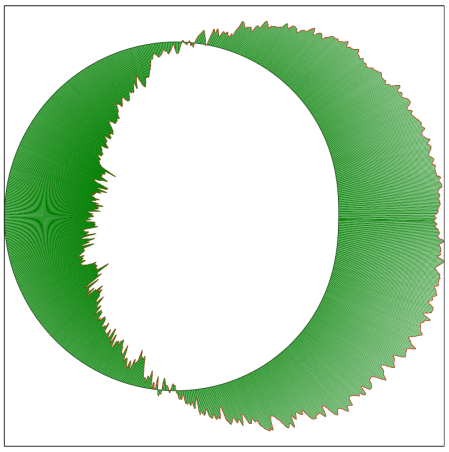
至此，基础的图形已经可以绘制出来了，但是细节仍然需要很多的调整。

画出轮廓-进阶01：数据的处理与分步作图

分析官方报告可以发现，**正负偏差**是以**不同的颜色标出**的。在**DRWPLY**指令中所有的颜色都是一样；而**PLS_ADDCRVL**指令虽然可以用颜色区分填充和轮廓，但由于填充颜色还是一样的，依然不便于观察。





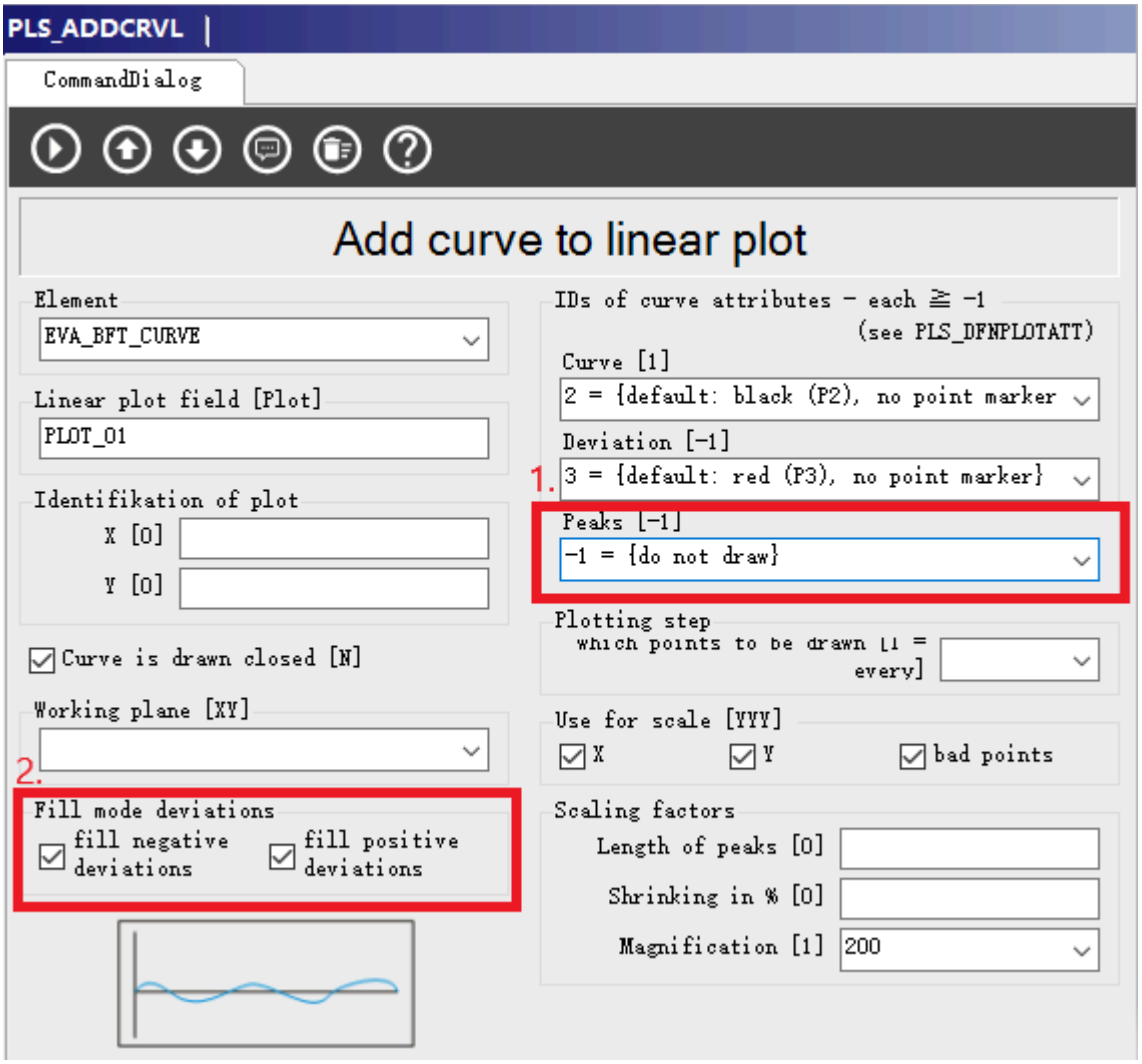


如果需要区分颜色，最为便捷的方案可以在PLS_ADDCRVL指令中找到：

1. 将偏差填充Peak选择为-1，抑制填充线的生成。
2. 勾选Fill mode deviations中填充正、负偏差区域的选项。

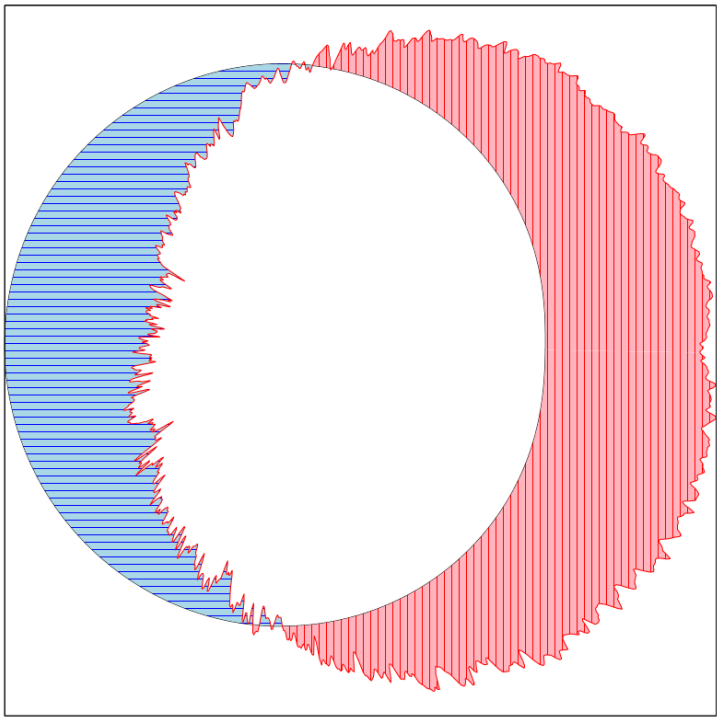
双色填充偏差区域 复制代码

```
PLS_ADDCRVL     (ELE=EVA_BFT_CURVE, PFL=PLOT_01, CLS=Y, CVA=(1,3,-1),
SCF=200, FLL= YY)
```



系统会自动以红蓝双色填充偏差。该图形样式与颜色为默认，不支持修改。

如果对自定义颜色有进一步的要求，则需要**对数据进行处理后分次画出**。



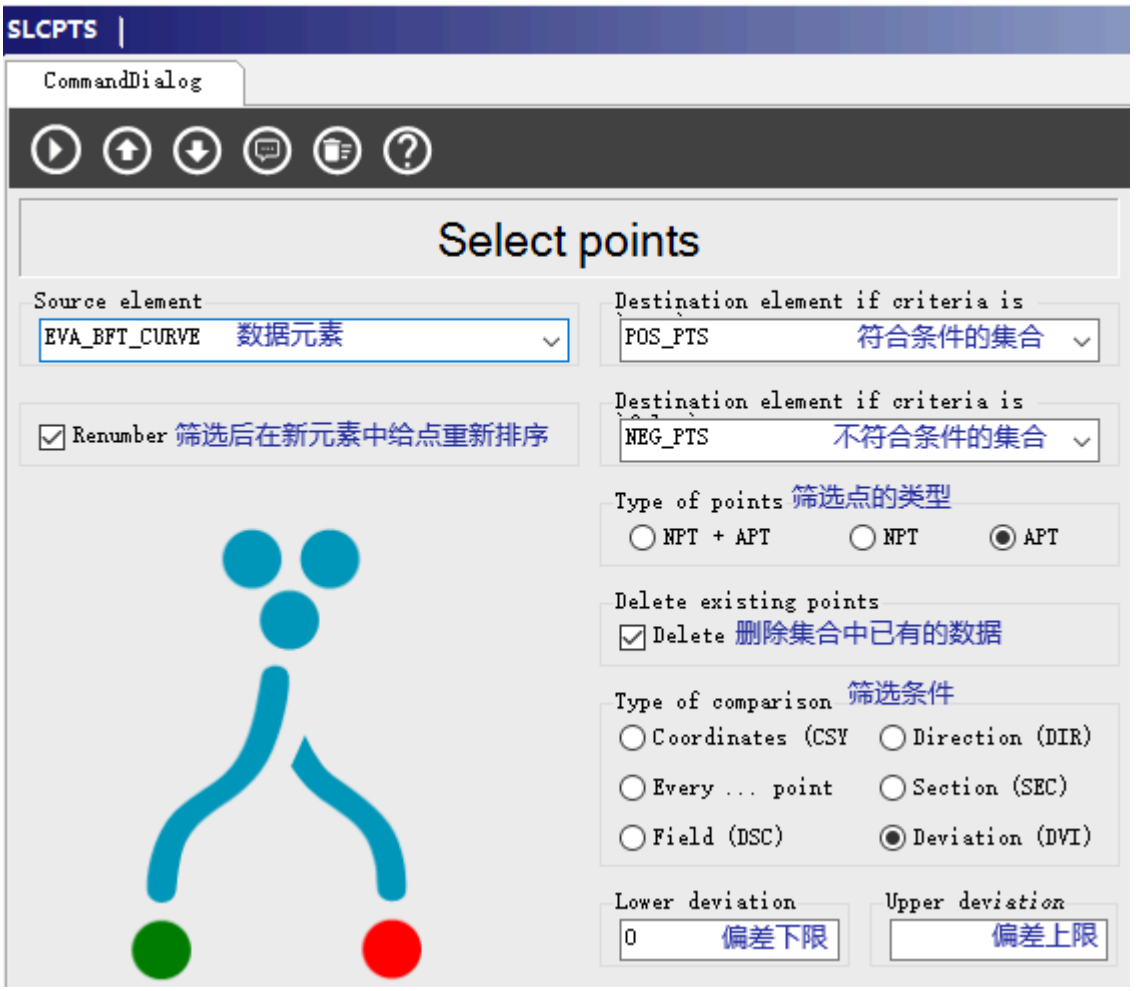
数据处理：

如果希望以不同颜色填充偏差，基于一个图形只能选择一种颜色填充的限制，我们可以考虑将图形拆分。将偏差为正、偏差为负的点分别收集到不同元素中，然后用不同的颜色分别绘制，重叠后即可构成完整图形。

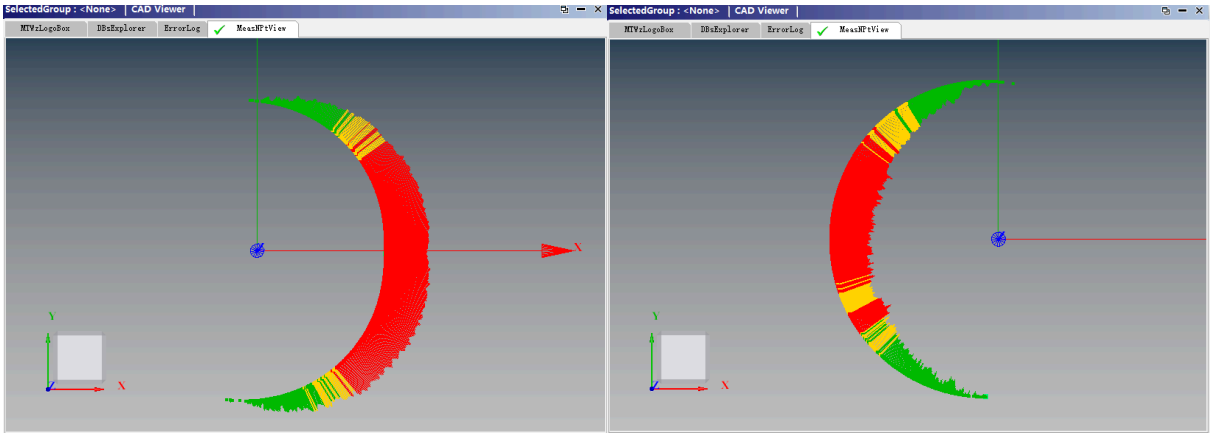
使用**SLCPTS**指令对数据点进行筛选：筛选条件设定为偏差Deviation，下限值取0。所有偏差大于等于0的数据点将被收集到新元素POS_PTS中，剩下小于0的点则进入新元素NEG_PTS中。

筛选数据点 复制代码

```
SLCPTS          (SRC=EVA_BFT_CURVE, TRU=POS_PTS, FLS=NEG_PTS, STY=APT,
DEL=Y, TYP=DVI, MIX=0)
```



执行命令后，就可以得到两个新元素。查看APT图形界面，确认点已经被正确划分。



（左侧为POS_PTS，右侧为NEG_PTS）

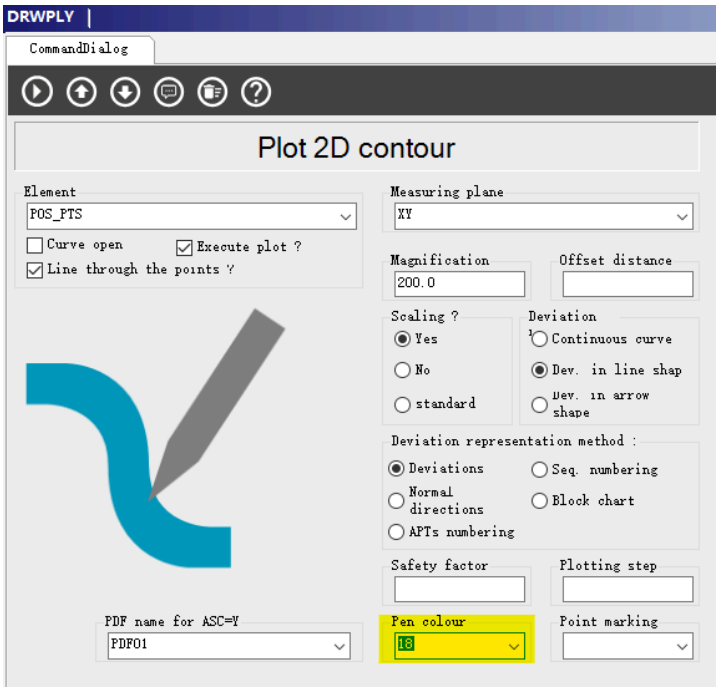
DRWPLY系列：

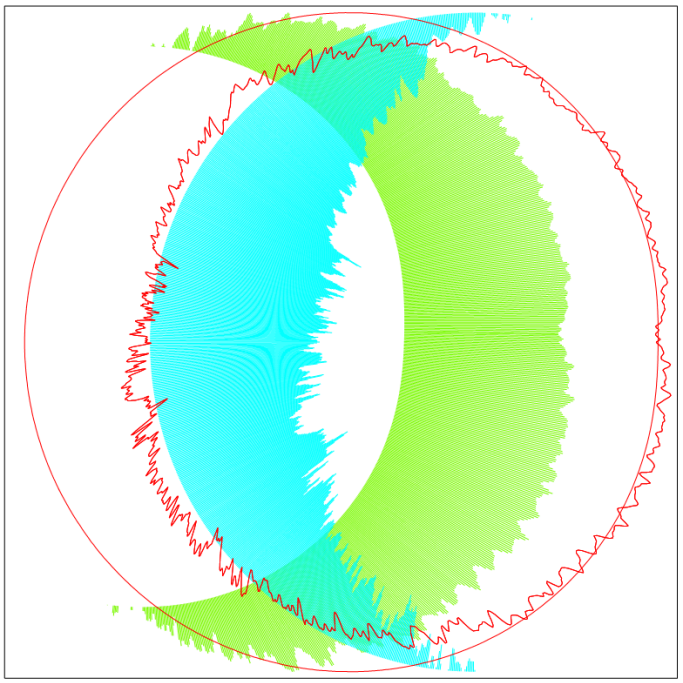
保持其他选项不变，仅改变**DRWPLY**中Pen color的设置，将两个元素分步绘出。同时再加上本来的实测轮廓和偏差轮廓，一同作出。

分步绘图 复制代码

```
DRWPLY          (NAM=POS_PTS, ASC=Y, DEV=Y, FAC=200.0, DRP=1, PEN=18,
OPN=N, A_O=XY)
DRWPLY          (NAM=NEG_PTS, ASC=Y, DEV=Y, FAC=200.0, DRP=1, PEN=11,
OPN=N, A_O=XY)
DRWPLY          (NAM=EVA_BFT_CURVE, ASC=Y, DEV=Y, FAC=200.0, DRP=0,
PEN=3, OPN=N, A_O=XY)
DRWPLY          (NAM=EVA_BFT_CURVE, ASC=Y, DEV=Y, FAC=1.0, DRP=0,
PEN=3, OPN=N, A_O=XY)
```

得到如下图形。可以看到，两个偏差的区域已经用不同的颜色画出了，实测轮廓和偏差轮廓也已经出现。但是位置出现了问题，它们并没有组合成一个完整的图形，而是交叠在一起。这是因为我们在**Scaling**中选择了【自动缩放】。系统会尝试**将单个图形缩放到最大**。由于分步绘图，系统对四个图形分别进行了最大缩放的计算，所以它们的位置不对了。为了解决这个问题，我们需要对作图区域的坐标系和缩放比例进行定义，让**缩放的大小和绘图的坐标位置统一**。（下一章节内容）



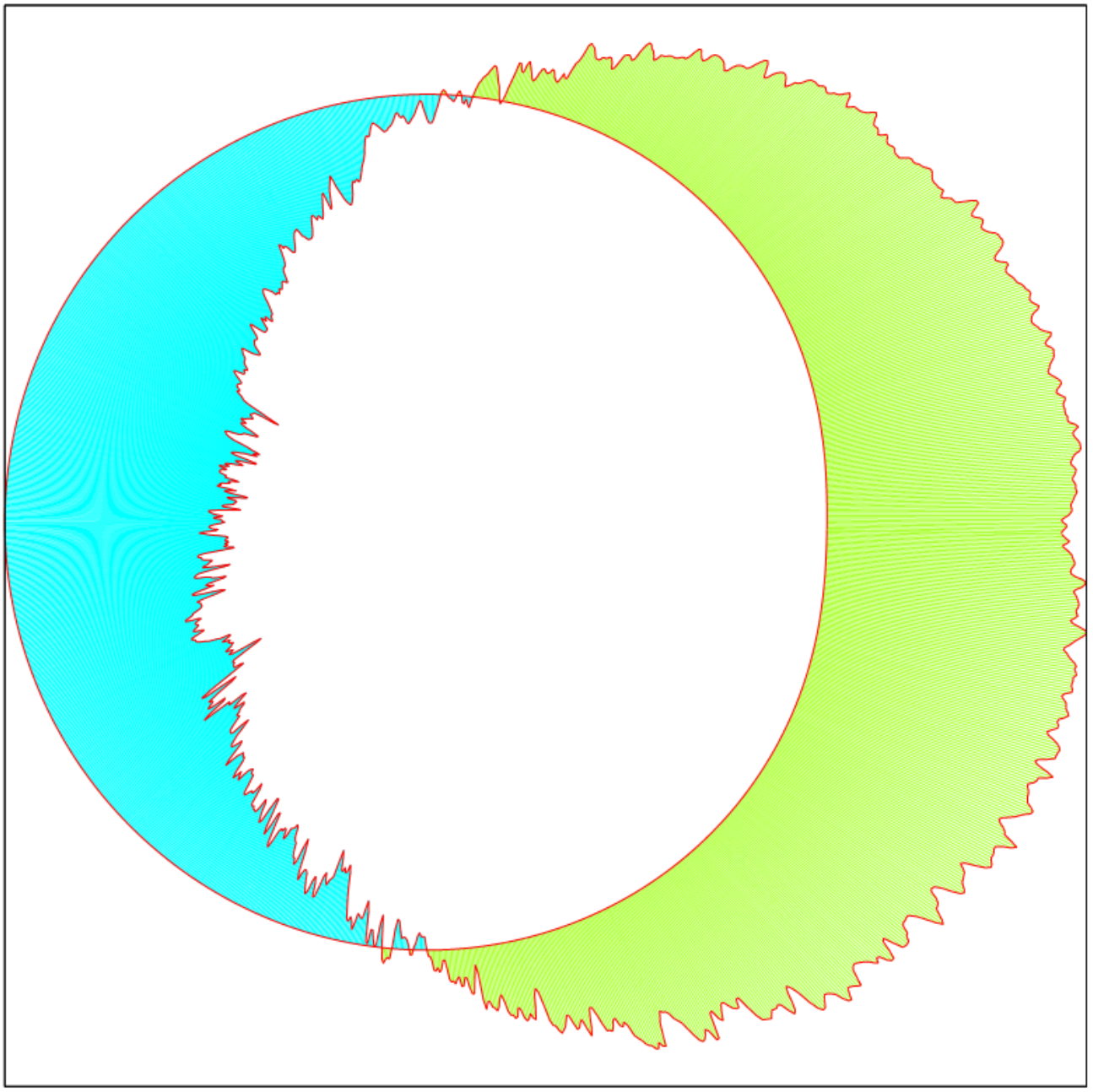


PLS_ADDCRVL系列:

使用PLS_ADDCRVL将两个元素连同本来的实测轮廓和偏差轮廓一并作出。

分步绘图 复制代码

```
PLS_ADDCRVL     (ELE=POS_PTS, PFL=PLOT_01, CVA=(-1,-1,1002), SCA=
, SCF=200, FLL=, A_O=XY)
PLS_ADDCRVL     (ELE=NEG_PTS, PFL=PLOT_01, CVA=(-1,-1,9), SCA=      ,
SCF=200, FLL=, A_O=XY)
```



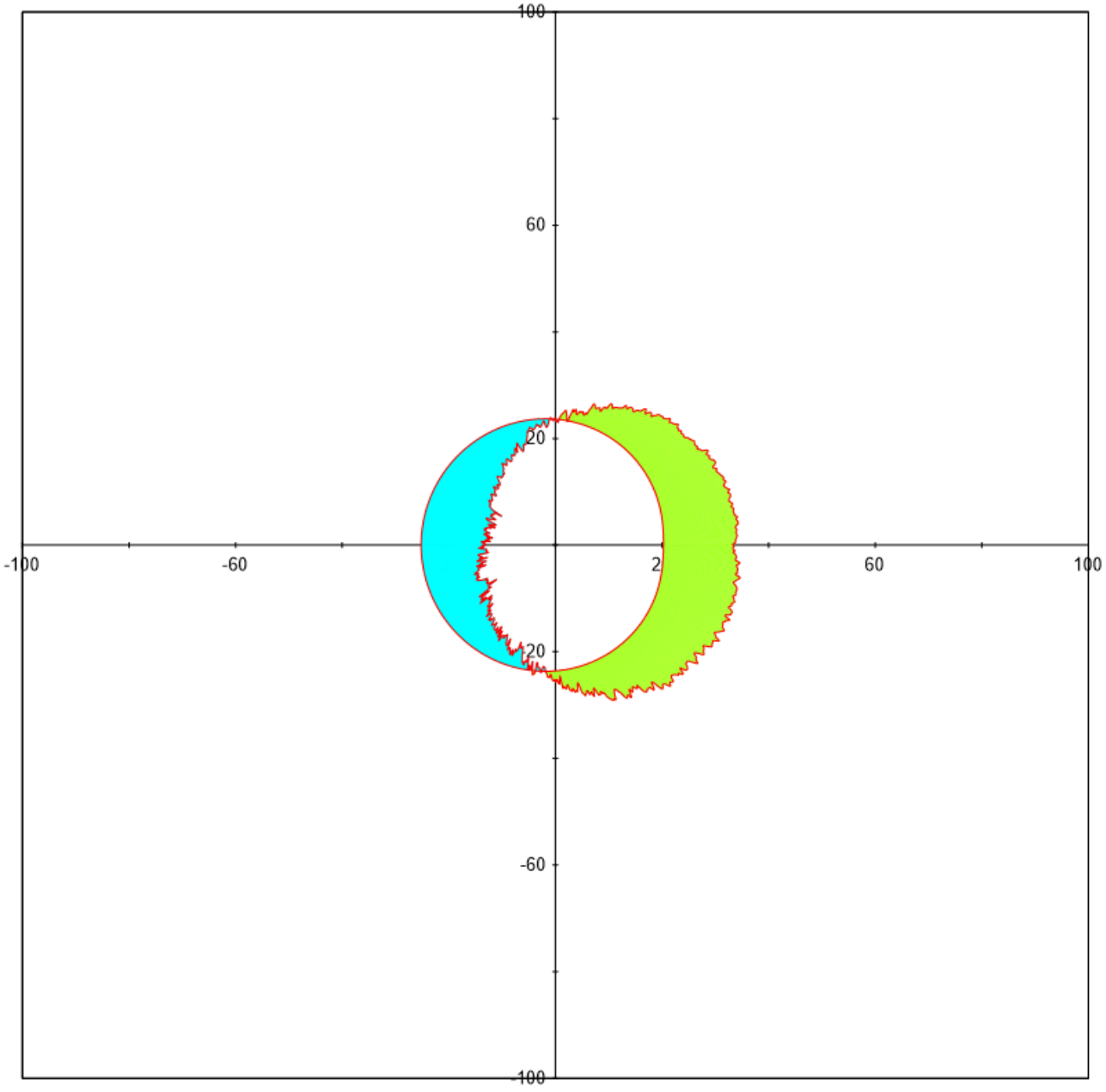
效果比DRWPLY要好很多，相比DRWPLY的独立计算绘制，PLS_ADDCRVL看起来对所有区域内的图形进行了整体的调整。由此可以推断，PLS_ADDCRVL的指令具有多元素关联绑定的逻辑。在指定作图区域内，根据坐标系位置、偏差大小，联动计算出组合元素的最佳缩放比例。所以在同一坐标系下的元素，位置关系正确且缩放合理。

但是，这个图形依然存在一个潜在的问题：**坐标系原点位置不明确**。这个问题会导致在标记最大最小点以及画出坐标轴时造成麻烦。（下一章节内容）

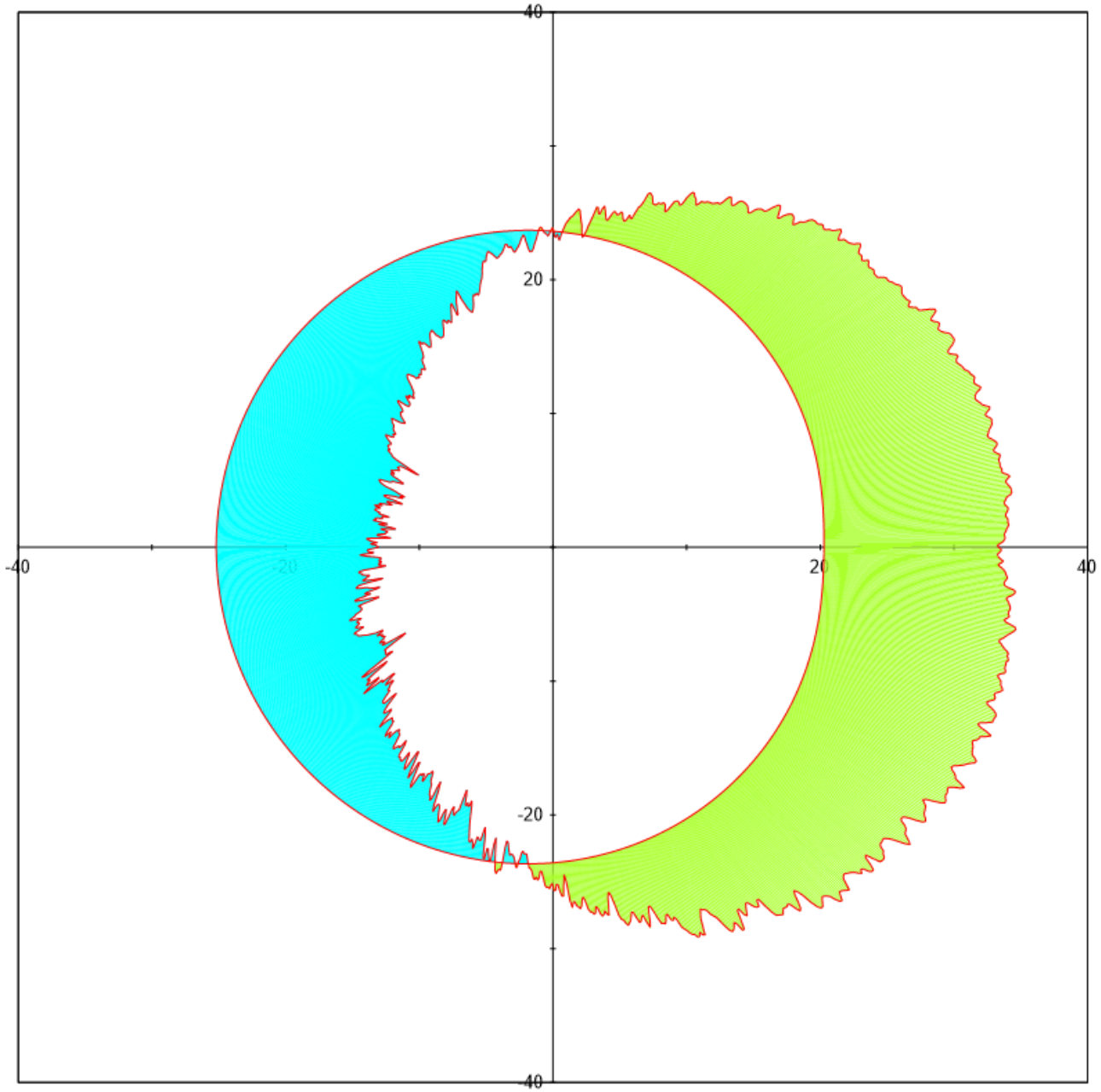
画出轮廓-进阶02：控制缩放坐标系

控制图形缩放比率的原理是：在指定作图区域生成一个坐标系。当坐标系在区域内显示的XY轴长度越长，图形在实测点本身的XY坐标信息不变的条件下，它们在图形上的显示就会越小。

XY轴都从**-100到+100**时：



XY轴都从**-40到+40**时：



所以控制缩放的核心是控制坐标系的原点位置和轴的长短。

- **原点位置**：改变原点位置，就会改变图形在定义作图区域内的中心位置。
- **轴**：改变轴的长短，就会改变图形的放大比率。**如果两个轴的比例不一样，则可以对图形进行非线性缩放。**

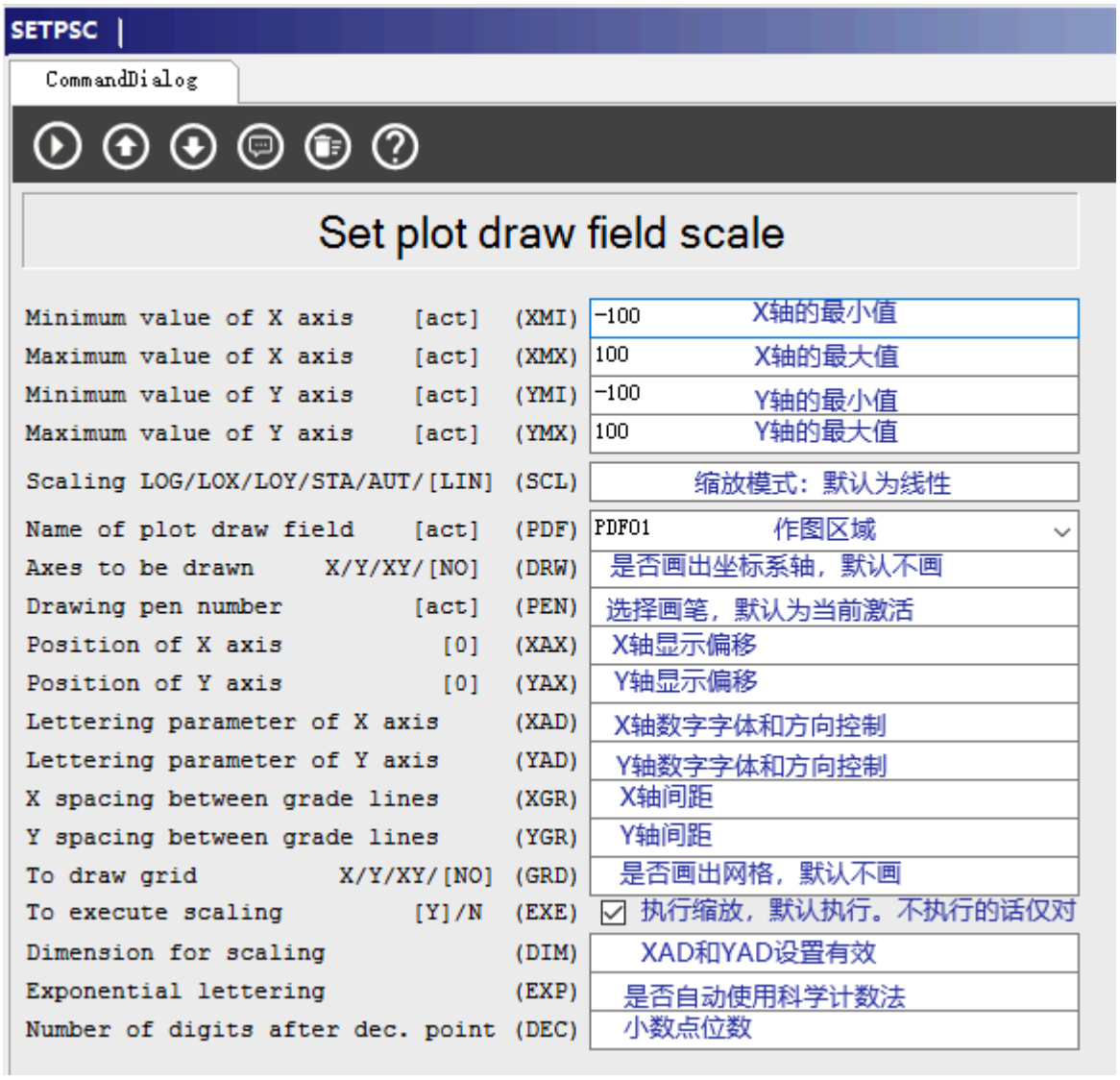
以下部分将对两个系列的比例缩放控制进行说明：

DRWPLY系列：

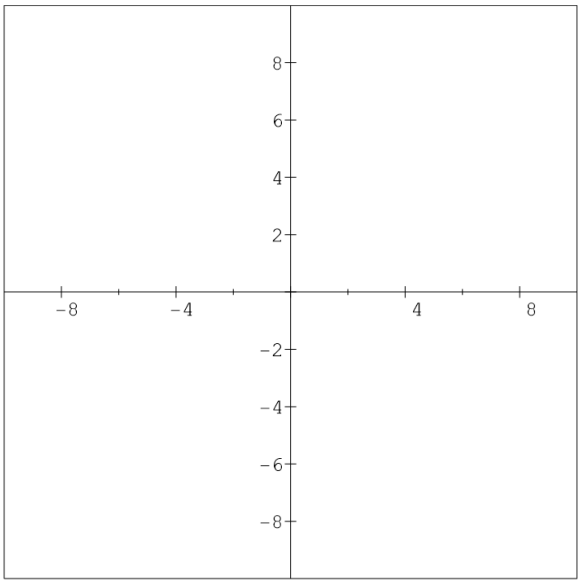
使用指令**SETPSC**进行设置：

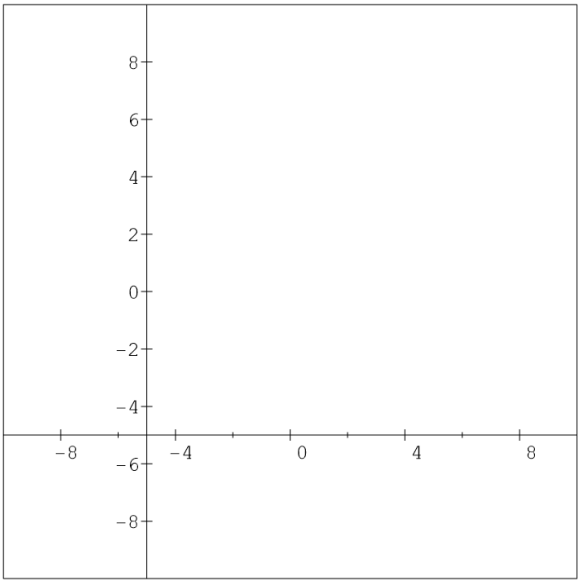
设置作图区域缩放 复制代码

```
SETPSC          (PDF=PDF01, XMI=-100, XMX=100, YMI=-100, YMX=100)
```

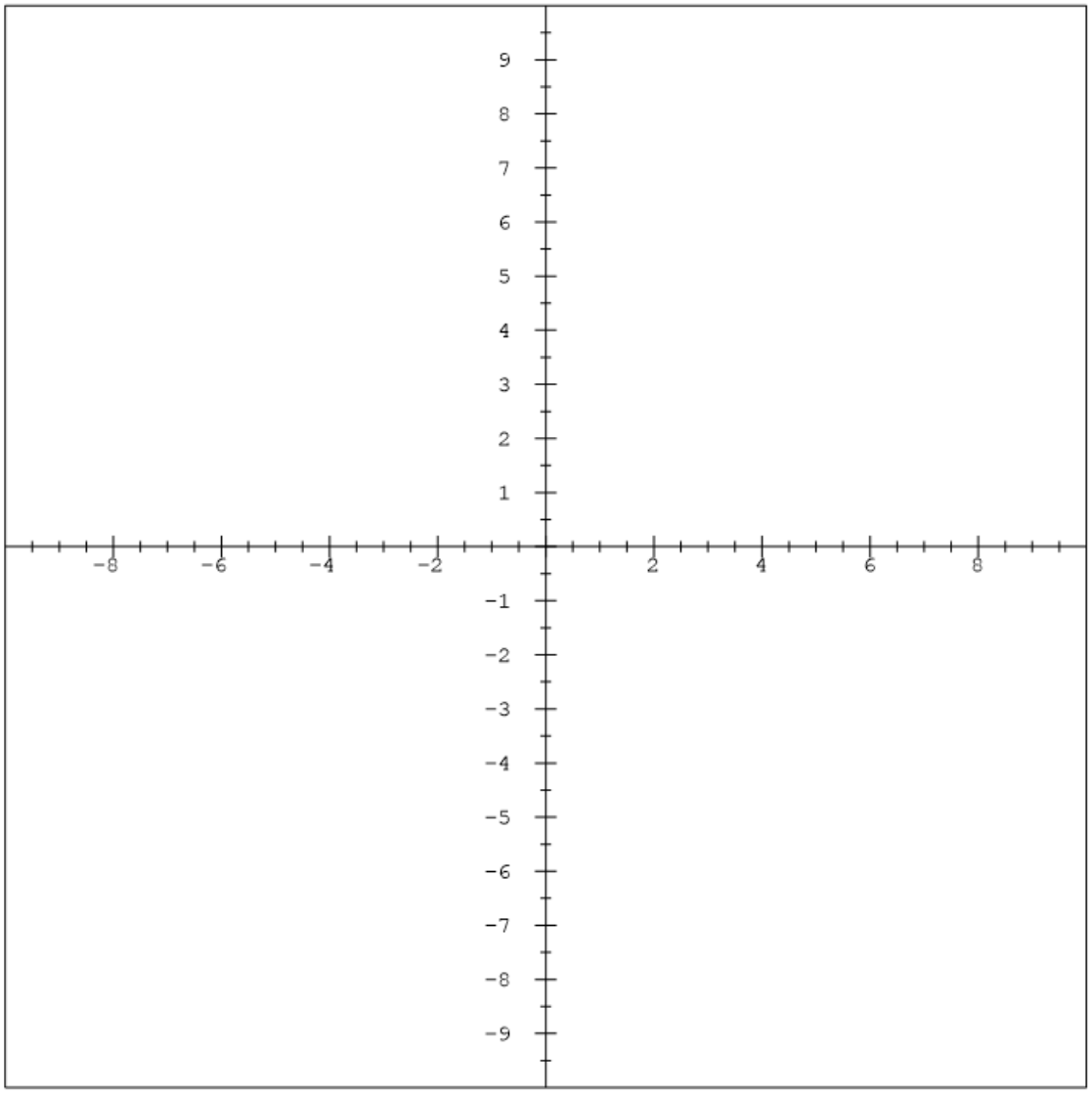


- **指令详情**
  - **XMI,XMX**决定了X轴在正负两个方向的长度。如果**两者绝对值相等**，那**坐标系原点就在该作图区域水平方向的正中**（适合观察测量形状和函数图形等）；如果**不相等**，则根据需要**偏置显示**（适合绘制折线图、柱状图等）。
  - **YMI,YMX**决定了Y轴在正负两个方向的长度。如果**两者绝对值相等**，那**坐标系原点就在该作图区域垂直方向的正中**；如果**不相等**，则根据需要**偏置显示**。
  - **PDF**需要输入相对应的作图区域。
  - **DRW**可以快速画出图形区域的XY坐标轴以及轴刻度。可以在只作X轴、只作Y轴、作XY轴和不作图四个模式中选择。
  - **PEN**输入画笔的数字，不输入默认使用当前激活画笔。
  - **XAX,YAX**可以让坐标轴显示偏移。从原始位置移动一定的距离到新位置作图。**需要注意的是，这个偏移只是偏移坐标轴显示位置，并非偏移坐标系原点。偏移后坐标系两个轴的交点并非原点（0，0）**。请仔细观察下面两张图。

- 

- 

- **XAD,YAD**的输入格式为（**数字1，数字2，数字3**）。括弧也需要输入。这三个数字控制了在坐标轴刻度上显示的数字字体大小、频率和角度。
  - 数字1：输入大于0的数字，设定字体大小；等于0时保留当前默认设置；小于0时不显示数字。
  - 数字2：当坐标轴上被划分刻度时，如果刻度有空缺且显示区域足够时，只划分和显示【数字2】n倍相关的数值。一般设为1。（注意：此设置并不影响刻度的单位长度，只是控制刻度数字的显示频率）
  - 数字3：控制数字显示的角度。根据需要可以在5个数字中选择。
    - 0：自动根据大小和重叠程度调整与轴的夹角
    - 1：夹角0度
    - 2：夹角90度
    - 3：夹角180度
    - 4：夹角270度
- **XGR,YGR**控制坐标轴上刻度的间距密度。数字越小刻度越密，反之越疏。



- **GRD**控制坐标系是否需要根据刻度作出网格线。可以在只作X轴、只作Y轴、作XY轴和不作图四个模式中选择。
- **EXE**控制是否执行缩放。默认为执行，取消勾选为不执行。如果不勾选，此时坐标系位置大小相关的设置不生效，只能调整PEN、XAD、YAD相关的字符显示设置。此状态在需要调整绘制坐标系刻度数字时很方便。
- **DIM**设定英尺模式下的参数。作用不详，常规不用设置。
- **EXP**设定了是否需要自动使用科学计数法。默认为Y激活，但如果绘制折线图、条状图等，希望微米以0.00X格式显示，需要设置为**N**。
- **DEC**可以用于控制小数点后位数，配合上面**EXP**一起使用。

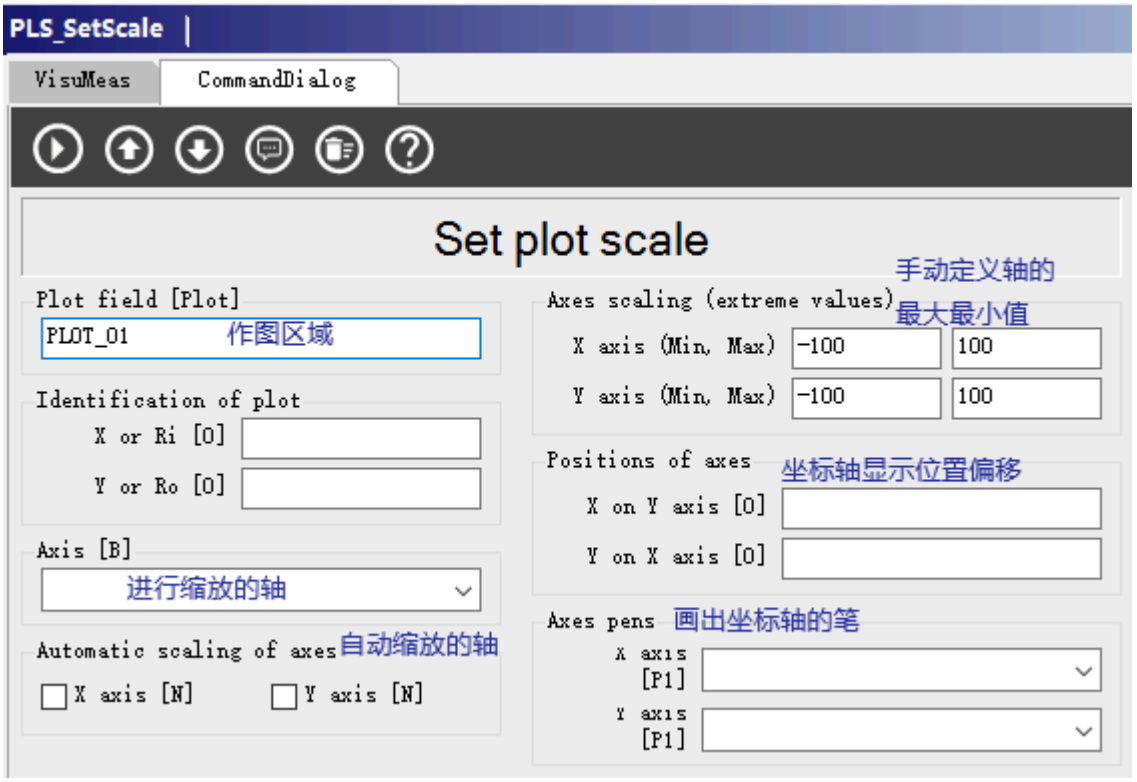
PLS_ADDCRVL系列:

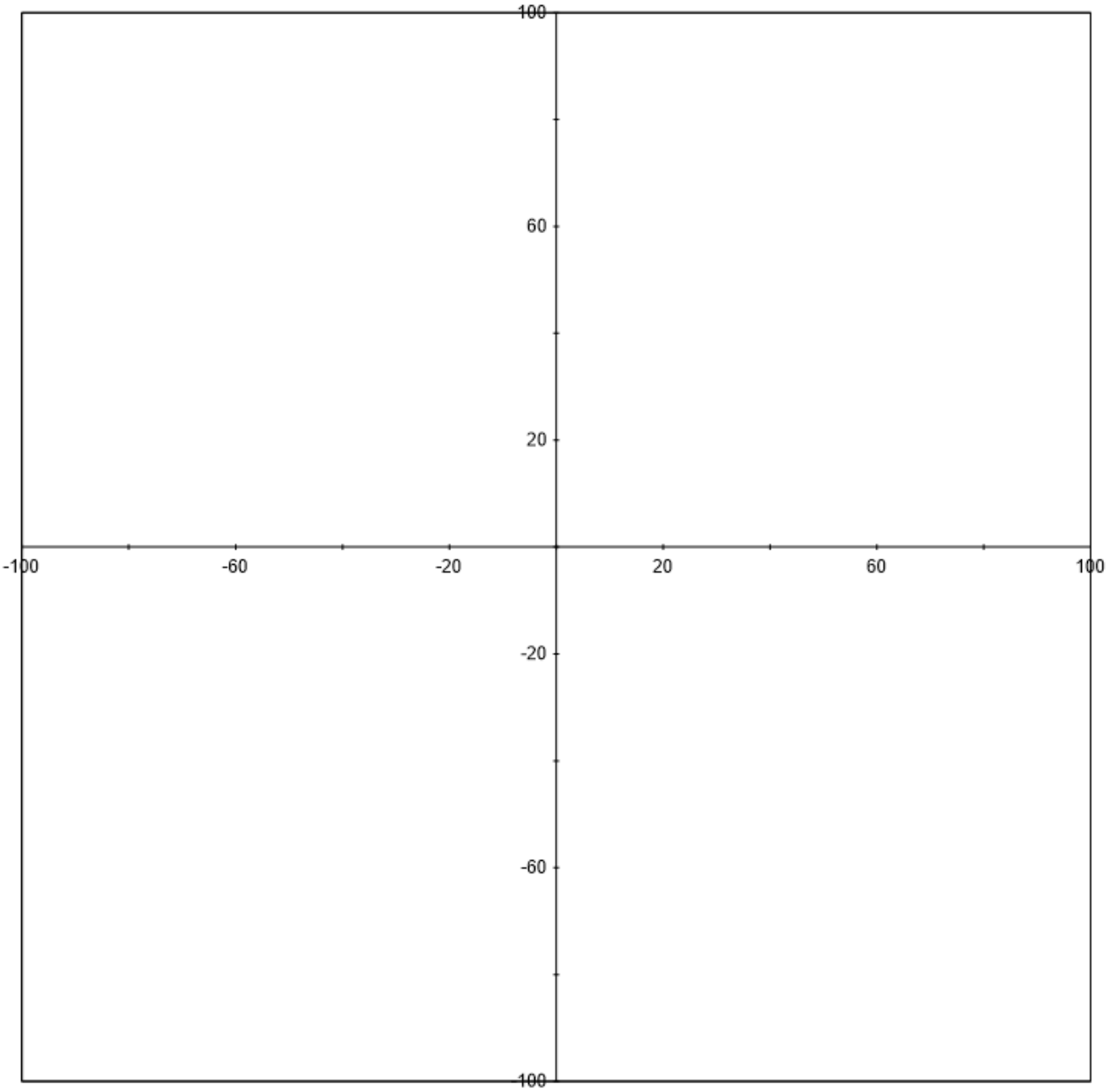
使用指令**PLS_SetScale**进行设置：

设置作图区域缩放 复制代码

```
PLS_SetScale    (PFL=PLOT_01, ASC=NN, VAL=(-100,100,-100,100), PEN=
(Empty,Empty))
```

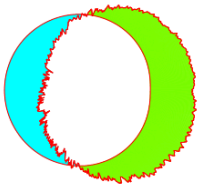
该指令和SETPSC的关键含义相同，都是通过定义轴的最大和最小值，来确定放大的比例和中心位置。此处不再重复说明。



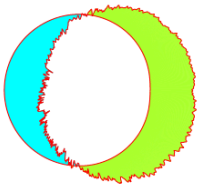


通过上述设置缩放后，再次执行轮廓绘图命令，将带有颜色填充的偏差以我们规定的缩放比率和位置作出。如下所示：

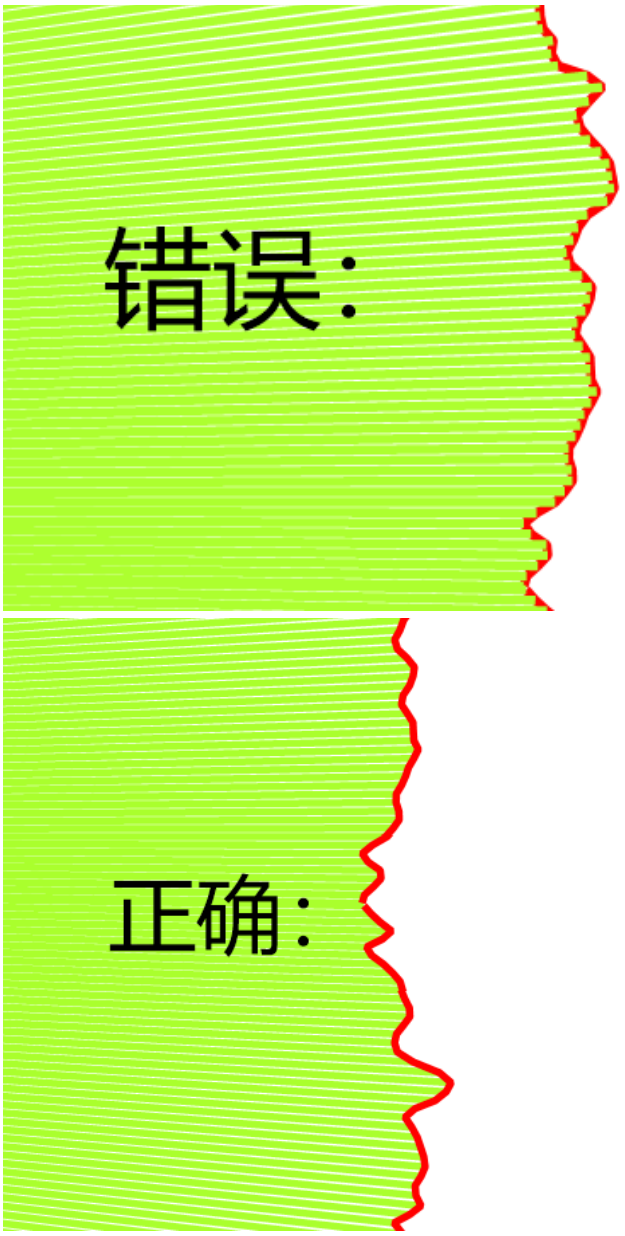
DRWPLY：



PLS_ADDCRVL:



补充：目前的作图已经涉及多个元素，需要注意的一点是**图层覆盖顺序。DRWPLY**的逻辑很简单，按照指令执行顺序，后执行的覆盖在前执行的图形上面；**PLS_ADDCRVL**的逻辑为**按元素名称的排序放置图层**。必要时需要将绘出元素**重命名进行排序**，否侧会出现如下的意外状况：Peak元素绘制在Curve和Deviation之上，导致轮廓线破碎不清晰。修复这个问题，需要将包含Peak信息的元素重命名或创建一个新名字，使其在ELE列表里排序靠后。



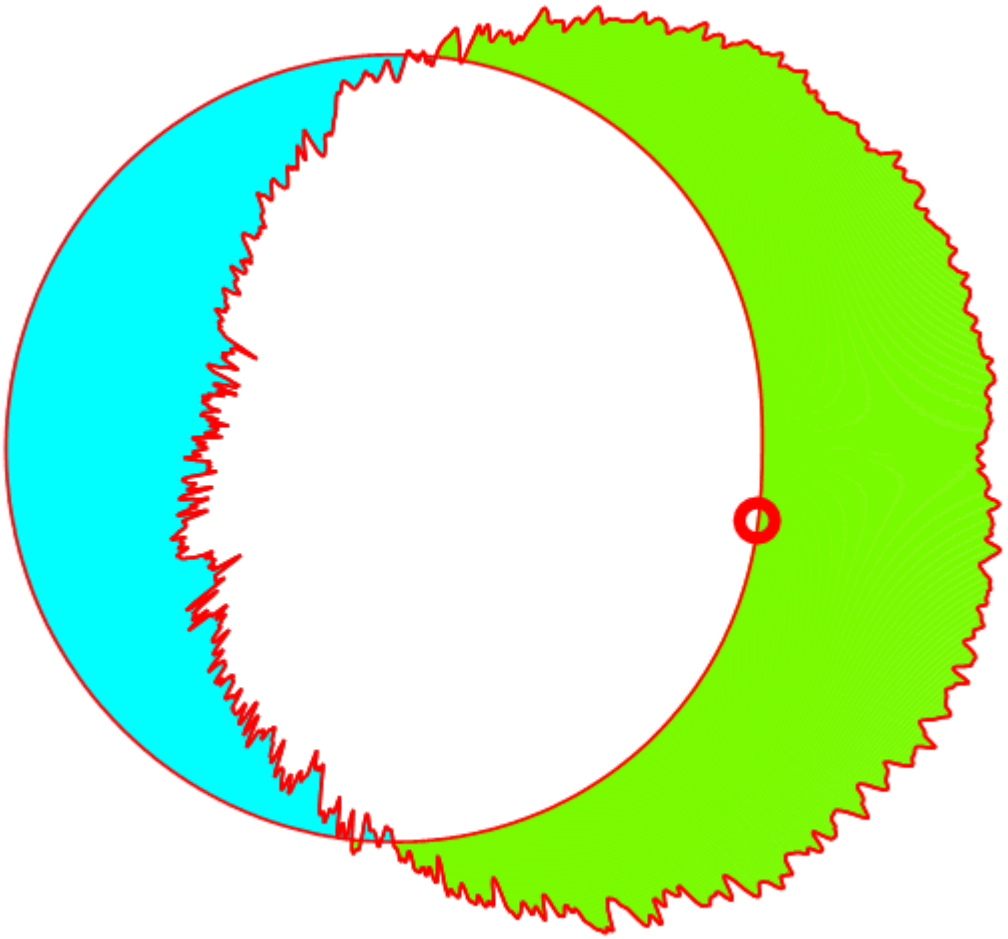
画出最大偏差点、最小偏差点和测量起始点的位置：

通过上一节的说明，我们已经明确了目前作图区域坐标系的情况。这一节的内容是结合坐标系，计算偏差点放大后的坐标系位置，然后在该位置画一个标记。

首先需要了解的是，为什么要计算偏差放大后的点坐标?

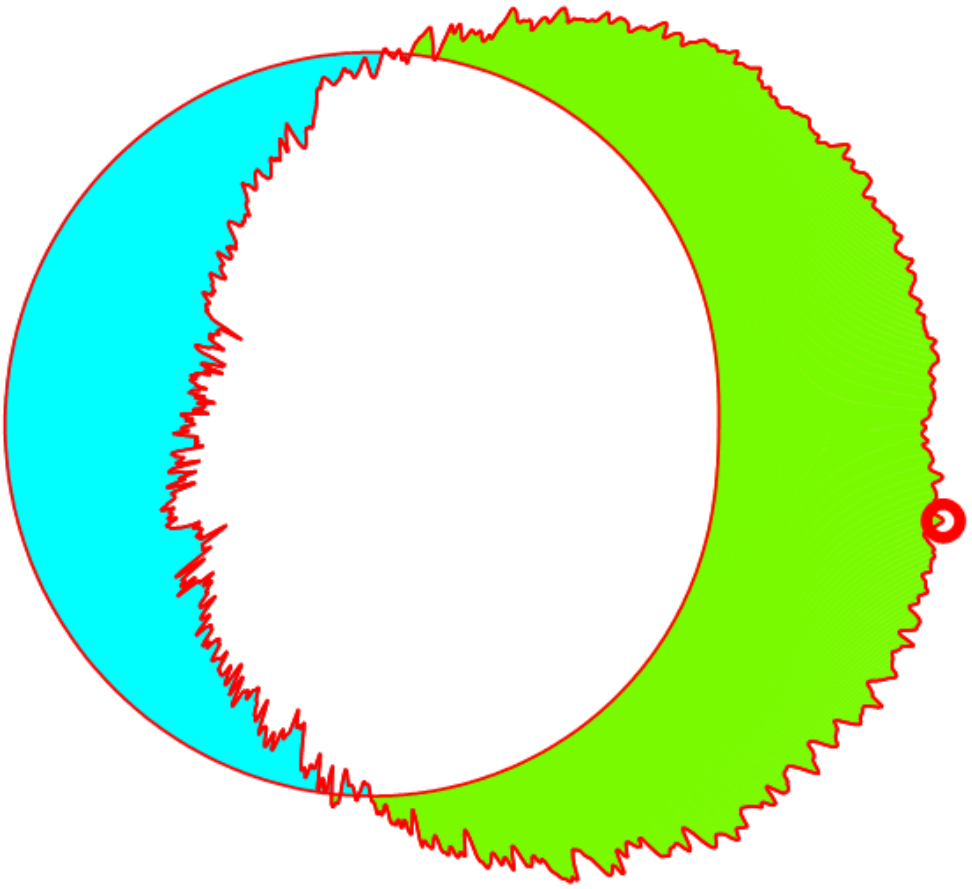
根据**【画出轮廓-基础】**中提到的**偏差放大倍率**，可以知道的是，偏差放大倍率如果是1，那偏差点就和原始轮廓重合。此时它们的坐标点完全一致，但是无法有效观察误差的形状；随着偏差放大倍率的提升，偏差点成倍率远离原始实测点，形状开始可见。但此时它的坐标我们不知道。如果依然通过获取原始数据点的XY坐标并作图，就会出现如下情况：

错误：



此时被标记的点在原始轮廓（实测点位置构成的形状）上，而非偏差轮廓（实测点加上偏差放大构成的形状），效果并不直观。并且，**错误的坐标位置还会影响公差带作图**（下一章节讲解）。理想的标记位置应该如下所示：

正确：



所以我们必须得到偏差放大后的点坐标。以测量平面在XY平面为例，计算公式如下：

$$X_2 = X_1 + \text{偏差放大率} * U * \text{偏差值}$$

$$Y_2 = Y_1 + \text{偏差放大率} * V * \text{偏差值}$$

（X2，Y2）为放大后偏差点坐标，坐标点（X1,Y1）为原始数据点坐标。U，V为方向余弦，等同于I，J。偏差放大率在**DRWPLY**和**PLS_ADDCRVL**中定义，此案例中设置为200。偏差值即为每一个点的**Deviation**数值。

Name : LDBELE:EVA_BFT_CURVE - Type : SRF - Csy : LDBCSY:CSY_A - IsNotBoundToCad | BoundTo : NB30581:/Root/Srf_auto(2) |

EditAPt

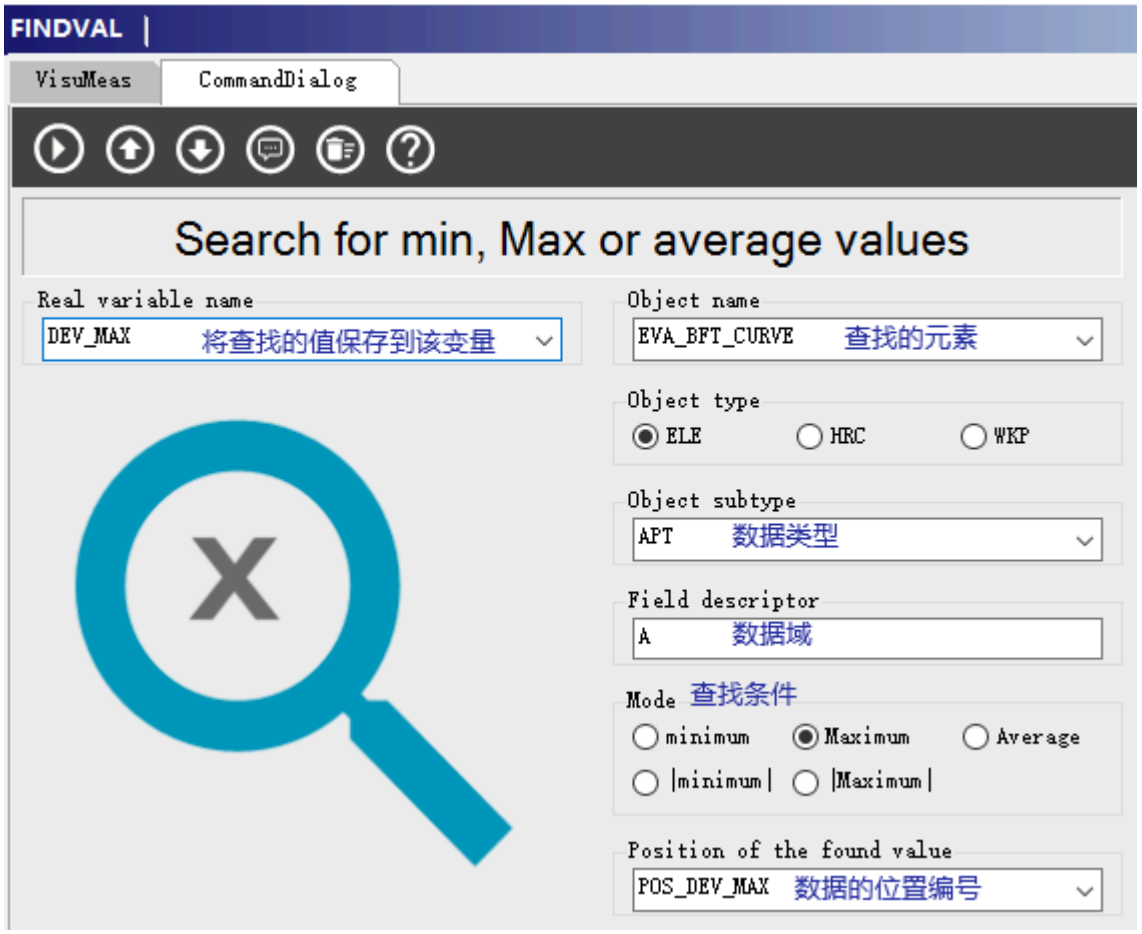
X-Coord	Y-Coord	Z-Coord	RotTbl	U-Dir	V-Dir	W-Dir	PointOkFlags	Deviation
20.2367	-1.6338	-30.4005	0.0000	0.9992	-0.040	0.0000	-1	0.0688
20.2406	-1.5339	-30.4005	0.0000	0.9993	-0.038	0.0000	-1	0.0684
20.2443	-1.4339	-30.4005	0.0000	0.9994	-0.036	0.0000	-1	0.0681
20.2478	-1.3340	-30.4005	0.0000	0.9994	-0.034	0.0000	-1	0.0679
20.2510	-1.2340	-30.4005	0.0000	0.9995	-0.032	0.0000	-1	0.0673
20.2541	-1.1341	-30.4005	0.0000	0.9996	-0.030	0.0000	-1	0.0666
20.2569	-1.0341	-30.4005	0.0000	0.9996	-0.028	0.0000	-1	0.0657
20.2596	-0.9342	-30.4005	0.0000	0.9997	-0.026	0.0000	-1	0.0653
20.2621	-0.8342	-30.4005	0.0000	0.9997	-0.024	0.0000	-1	0.0665
20.2644	-0.7342	-30.4005	0.0000	0.9998	-0.022	0.0000	-1	0.0672
20.2665	-0.6343	-30.4005	0.0000	0.9998	-0.020	0.0000	-1	0.0663
20.2684	-0.5343	-30.4005	0.0000	0.9998	-0.018	0.0000	-1	0.0656
20.2702	-0.4343	-30.4005	0.0000	0.9999	-0.017	0.0000	-1	0.0650

理论部分已经解释完毕，接下来进入Quindos程序环境，将我们需要的这些信息全部收集并加工。

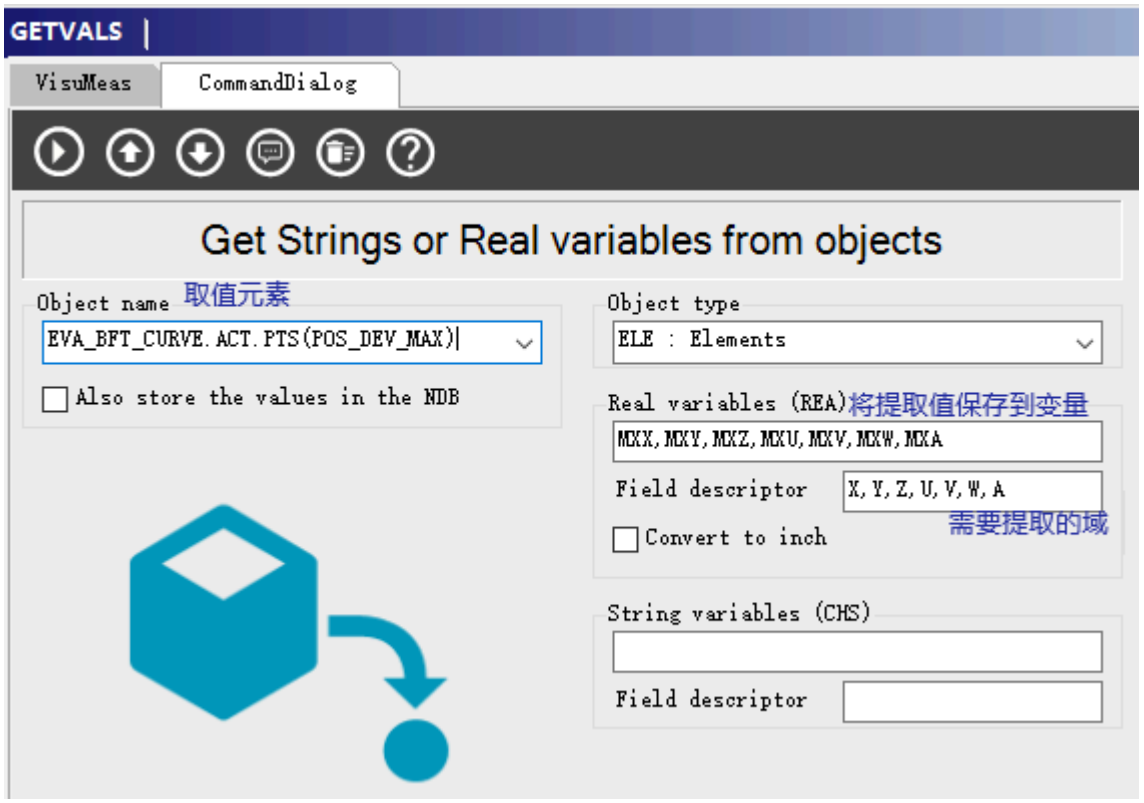
获取原始数据点信息 复制代码

```
!查找轮廓中偏差值最大和最小的点
FINDVAL          (NAM=DEV_MAX, OBJ=EVA_BFT_CURVE, TYP=ELE, STY=APT,
DSC=A, MOD=MAX, POS=POS_DEV_MAX)
FINDVAL          (NAM=DEV_MIN, OBJ=EVA_BFT_CURVE, TYP=ELE, STY=APT,
DSC=A, MOD=MIN, POS=POS_DEV_MIN)
!获取偏差最大、最小点和轮廓起始点的各项数据
GETVALS          (OBJ=EVA_BFT_CURVE.ACT.PTS(POS_DEV_MAX), TYP=ELE, RDS=
(X,Y,Z,U,V,W,A), REA=(MXX,MXY,MXZ,MXU,MXV,MXW,MXA))
GETVALS          (OBJ=EVA_BFT_CURVE.ACT.PTS(POS_DEV_MIN), TYP=ELE, RDS=
(X,Y,Z,U,V,W,A), REA=(MIX,MIY,MIZ,MIU,MIV,MIW,MIA))
GETVALS          (OBJ=EVA_BFT_CURVE.ACT.PTS(1), TYP=ELE, RDS=
(X,Y,Z,U,V,W,A), REA=(STX,STY,STZ,STU,STV,STW,STA))
```

- **补充**：以下是**FINDVAL**和**GETVALS**的用法解释。



- 数据域可以在Edit APT/NPT菜单中，通过鼠标**右键点击**列表抬头名字，如Deviation，在出现的菜单中点击Edit Properties查看。StringDsc行显示的**$A**即为A域。**$**符号可以输入或不输入。



- 取值元素需要一定的语法表达。详细规则可以参考Quindos在线帮助文档中关于**GETVAL**指令（最后不带S）的内容。此处表达公式为**轮廓元素.ACT.PTS(点编号)**，

> 其含义为提取目标是该轮廓元素下APT实测点中第（POS_DEV_MAX）个点。（POS_DEV_MAX）即为之前在**FINDVAL**中获得的最大偏差点在所有APT数据点中的位置编号。

通过上述命令，我们已经记录了最大、最小偏差发生在轮廓数据点上的位置编号，并通过编号取值了最大点、最小点以及轮廓起始点的XY坐标、方向余弦值和偏差数据。接下来进入作图部分。

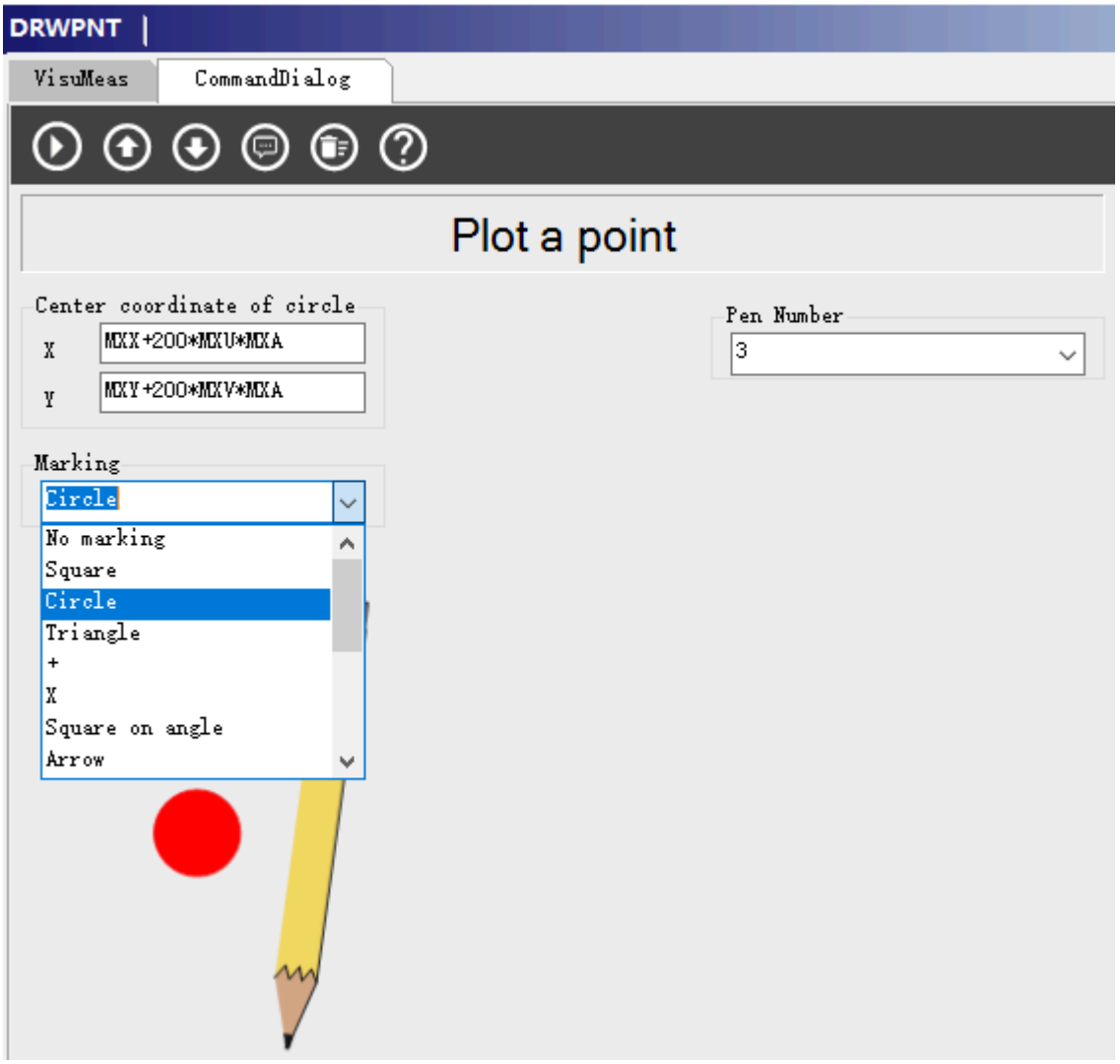
DRWPLY系列:

此处通过**DRWPNT**指令实现。

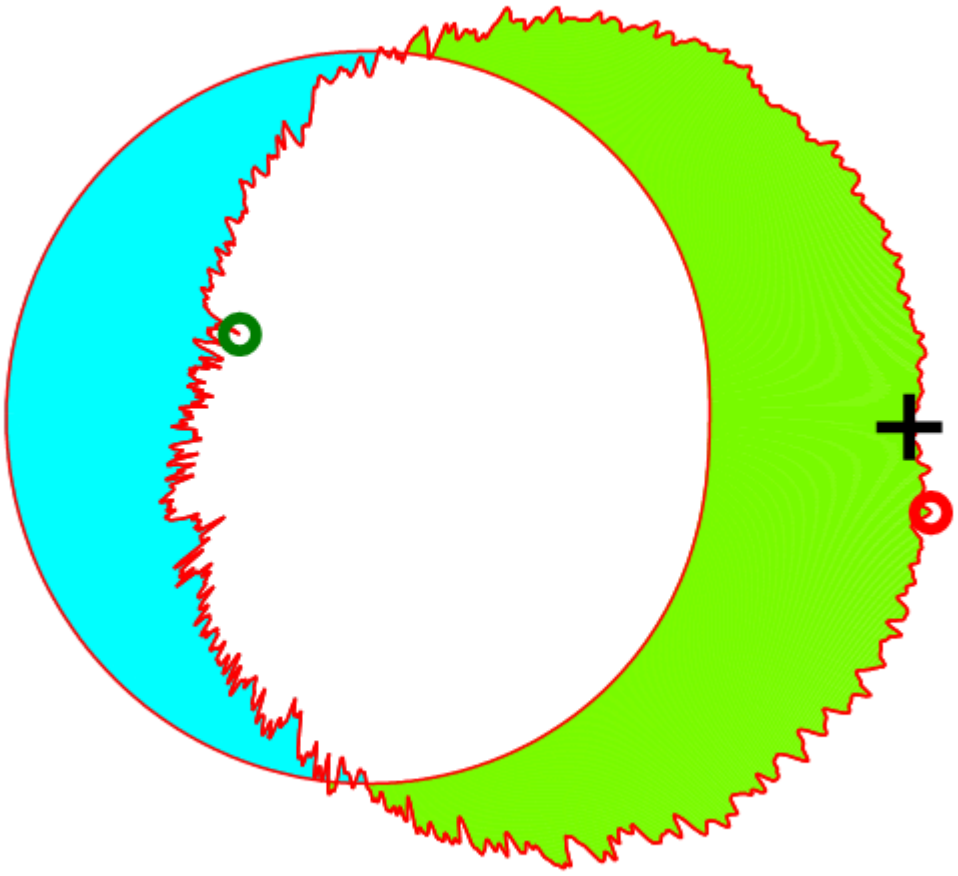
标记最大点，最小点和起始点 复制代码

```
DRWPNT          (X  =MXX+200*MXU*MXA, Y  =MXY+200*MXV*MXA, MRK=1, PEN=3)
DRWPNT          (X  =MIX+200*MIU*MIA, Y  =MIY+200*MIV*MIA, MRK=1, PEN=4)
DRWPNT          (X  =STX+200*STU*STA, Y  =STY+200*STV*STA, MRK=3, PEN=1)
```

XY坐标直接输入公式让Quindos自动完成计算。**Marking**可以选择多种标记样式，此处选择了圆形标记。**Pen Number**可以选择画笔颜色，此处3为红色。



将三个标记全部画出后，画面如下所示：

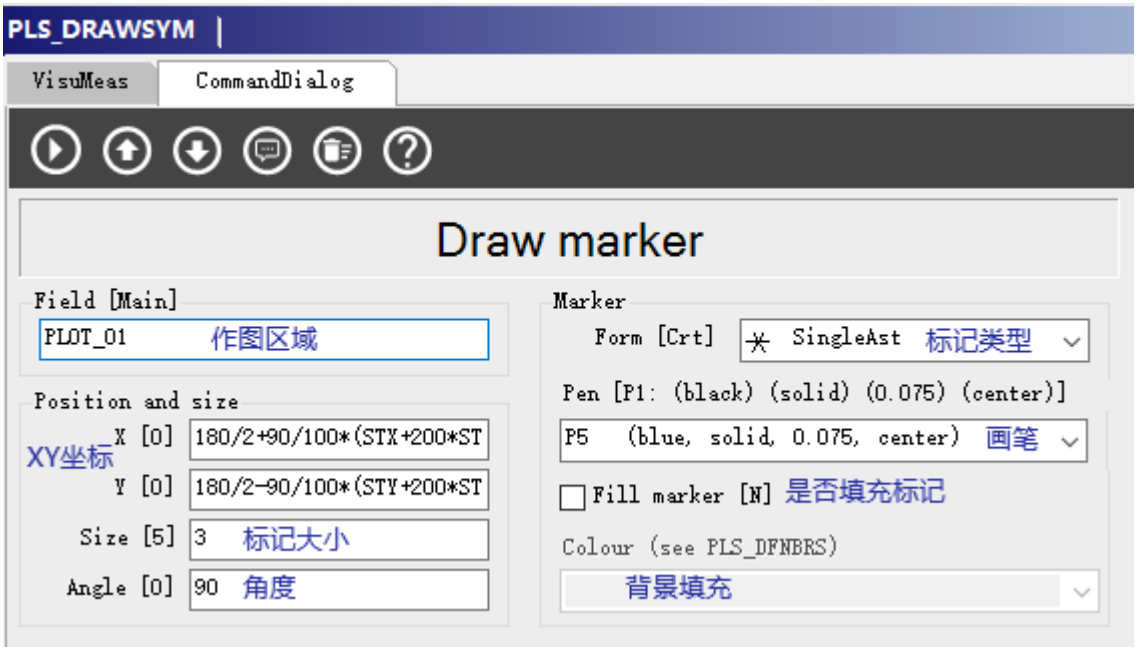


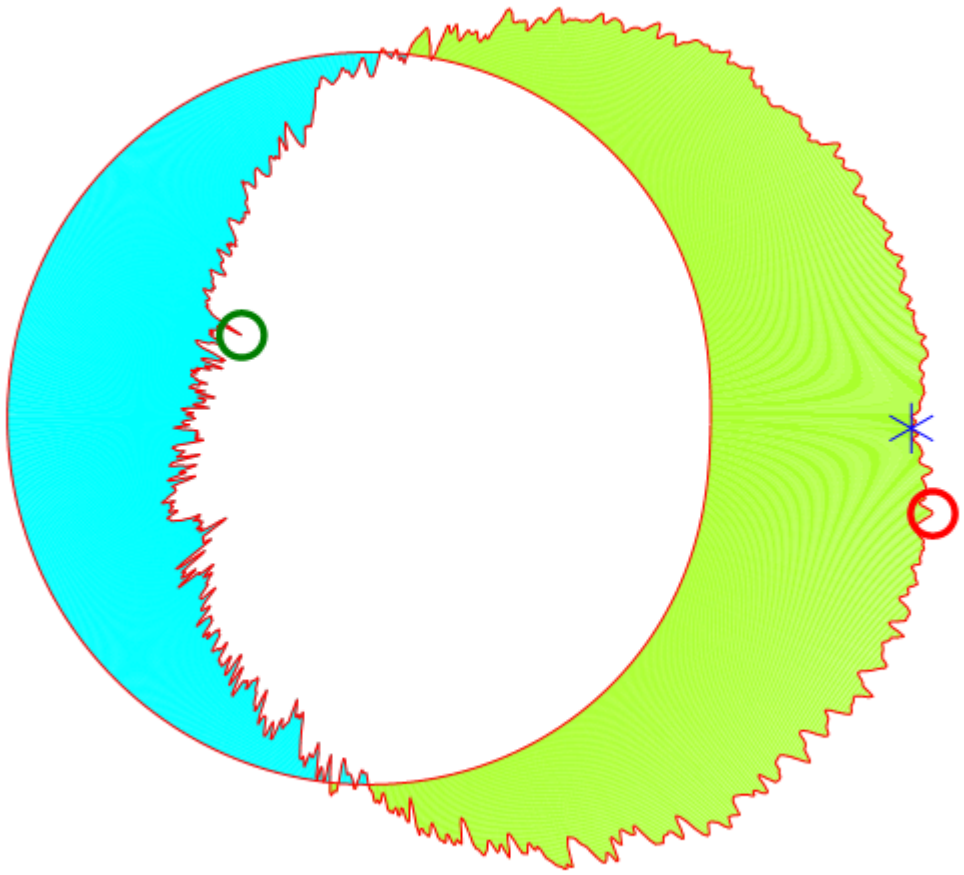
PLS_ADDCRVL系列：

此处通过**PLS_DRAWSYM**命令实现。

标记最大点，最小点和起始点 复制代码

```
PLS_DRAWSYM     (FRA=PLOT_01, X  =90+90/100*(MXX+200*MXU*MXA), Y  =90-
90/100*(MXY+200*MXV*MXA), PEN=~PEN3, SCO=Cir, SIZ=3)
PLS_DRAWSYM     (FRA=PLOT_01, X  =90+90/100*(MIX+200*MIU*MIA), Y  =90-
90/100*(MIY+200*MIV*MIA), PEN=~PEN4, SCO=Cir, SIZ=3)
PLS_DRAWSYM     (FRA=PLOT_01, X  =180/2+90/100*(STX+200*STU*STA), Y
=180/2-90/100*(STY+200*STV*STA), PEN=P5, SCO=SingleAst, SIZ=3, PHI=90)
```





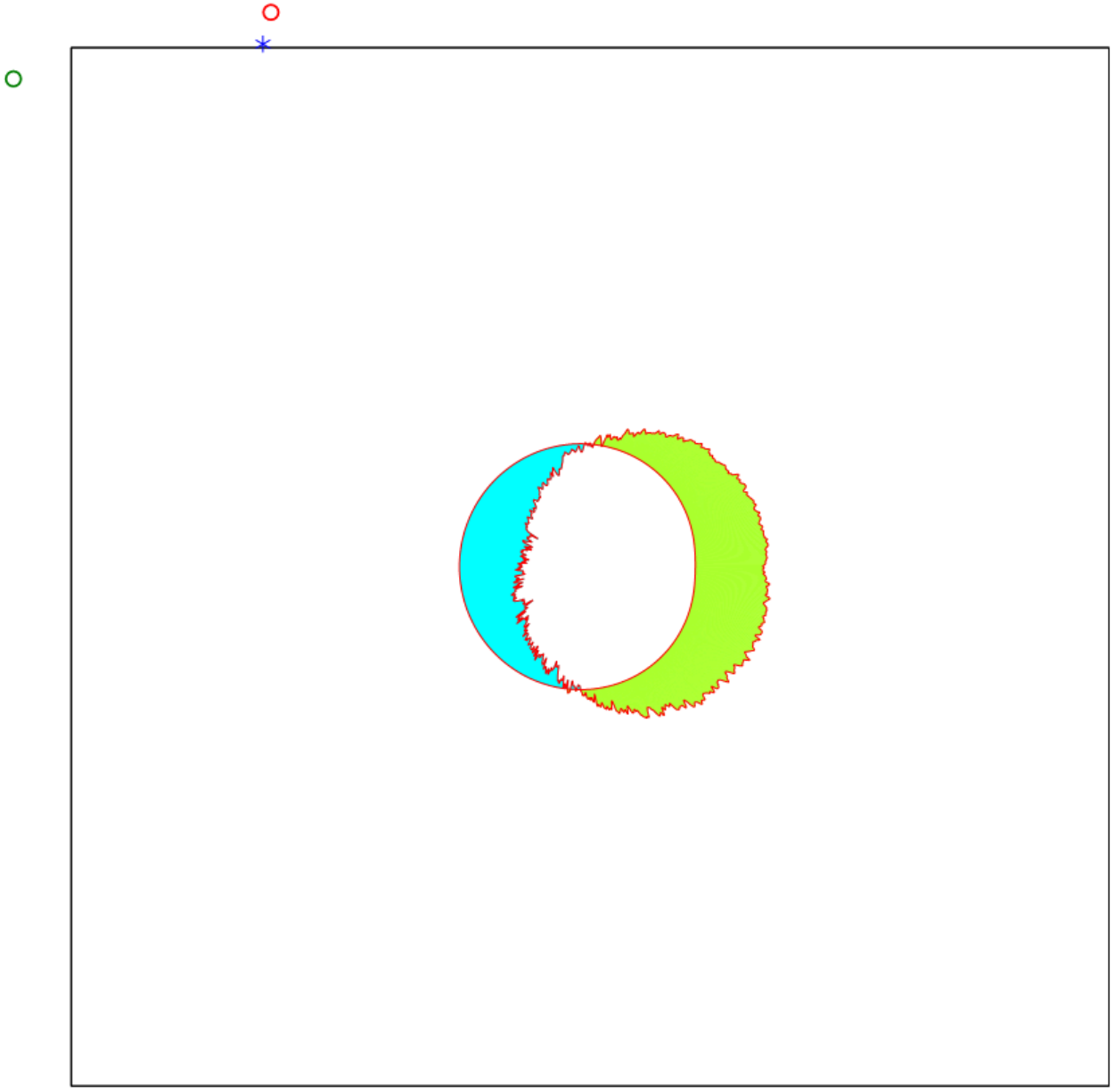
设置如上图所示，给定坐标系，选择标记种类和颜色即可画出图形。但是**细看XY轴坐标部分**，明显与**DRWPNT**的设置不一样，公式变长后截图已经不能完全显示。

完整公式表达如下：

$$X = 180/2 + 90/100 * (STX + 200 * STU * STA)$$

$$Y = 180/2 - 90/100 * (STY + 200 * STV * STA)$$

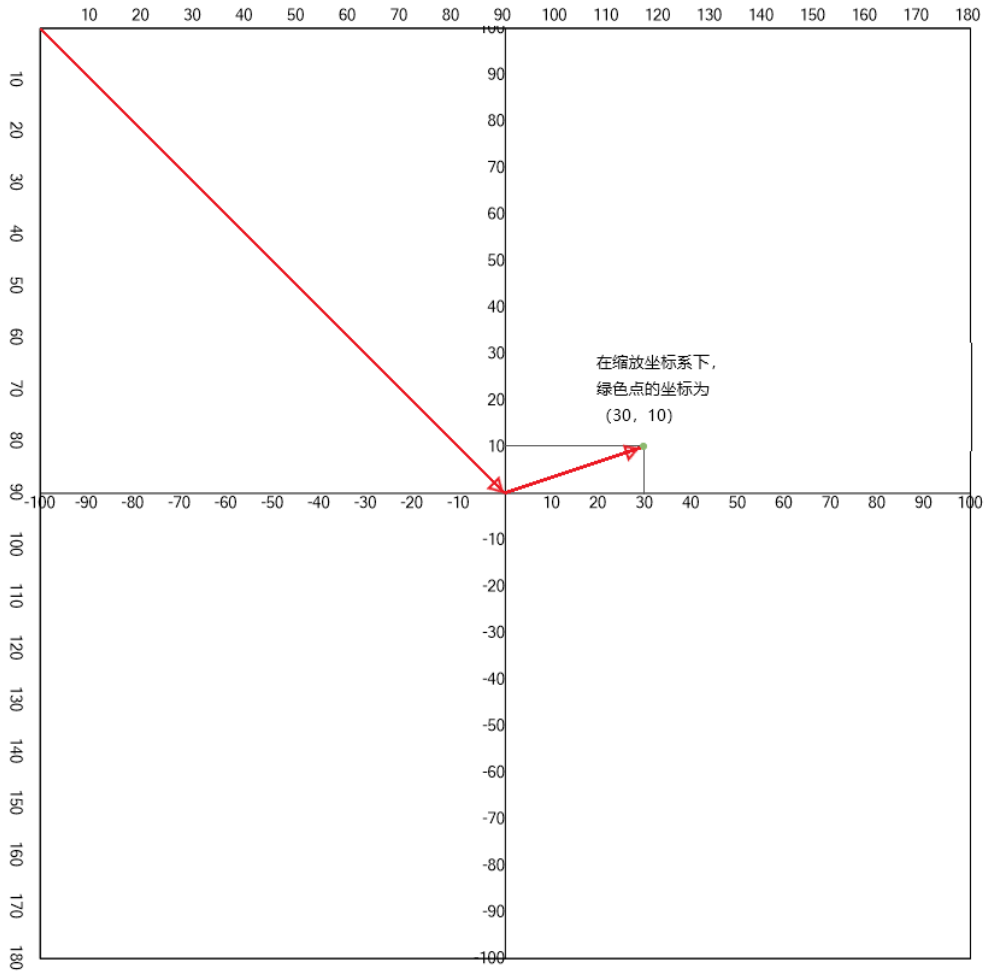
可以看出，后半部分**括号里的公式**和上文提及的是一致的，但多了一些前面的修正部分。需要修正的原因是：**PLS_DRWSYM**作图时使用的坐标是最初用**PLS_DFNPLOFLD**定义作图区域时的坐标，而非**PLS_SetScale**所定义的缩放坐标系。如果直接使用原公式，将会得到下图。**三处标记出现在了左上角**，即最初在【定义绘图区域】章节中提到的左上角起始的坐标系。



接下来我们需要将坐标系进行转换。注意：后续计算中的参数，是仅以本示例中的情况作为条件，数据不具有通用性。但原理具有通用性，后期可以使用变量进行自动的替代计算。此处为了清楚解释其中原理，所以使用了具体数字。

➔ ***转换坐标系需要考虑的要点***

- 原点的转换
  - 下图中可以看出，缩放坐标系的原点在作图区域坐标系（90，90），即原点转换到（X/2,Y/2）的位置，也是公式中180/2的部分的含义。



  -
- ***坐标轴缩放比例的转换***
  - 仔细观察上图，虽然已经将坐标系原点进行了转换，但是并不能简单地再加上缩放坐标系的XY值得到数据。缩放坐标系从原点到绿色点在X轴上的长度为30，而在作图区域坐标系上，长度仅为27左右。这是因为作图区域坐标系当初定义的宽度是180，两侧各90。而缩放坐标系是宽度200，两个各100。转换比例根据以下公式可以推算出：
    - ➔ 作图区域坐标系X半轴长（90）/缩放坐标系X半轴长（100）=作图区域坐标系偏移距离/缩放坐标系偏移距离（X+偏差放大率*U*偏差值）
  - ***转换一下可得：***
    - ➔ 作图区域坐标系偏移距离=作图区域坐标系X半轴长（90）/缩放坐标系X半轴长（100）*缩放坐标系偏移距离（X+偏差放大率*U*偏差值）
  - 放大偏移距离=90/100*(STX+200*STU*STA)
  - 最后将两个部分结合，就是命令中的公式

$$X = 180/2 + 90/100 * (STX + 200 * STU * STA)$$

- **需要注意的是：Y轴的情况有所不同**

  由于作图区域坐标系的Y轴方向是从上向下**递增**，而缩放坐标系的Y轴方向是从上向下**递减**。所以在缩放坐标系中向+Y的偏移，转换到作图区域坐标系，实际是向-Y偏移。所以公式在转换了原点坐标后再计算偏移距离时，需要变成**减法**。 $Y = 180/2 - 90/100 * (STY + 200 * STV * STA)$

至此，我们已经完成了一定程度的数据处理和作图。后续公差带的绘图所依赖的计算原理，目前已经全部说明了，仅需要用同样的方法作一些转换即可。

画出公差带：

系统指令已经自带了非常方便的公差带绘制功能。在**DRWPLY**指令中包含了Offset Distance功能，可以提供公差带作图功能；**PLS_ADDSHF**也提供了画出偏移公差带的功能。在本章的最后，我还会提供一个方案，可以从更底层的角度对公差带图形的生成进行理解和操控。

DRWPLY系列：

Offset Distance功能仍在研究中。有进展会更新到文档。

目前已知的情况：

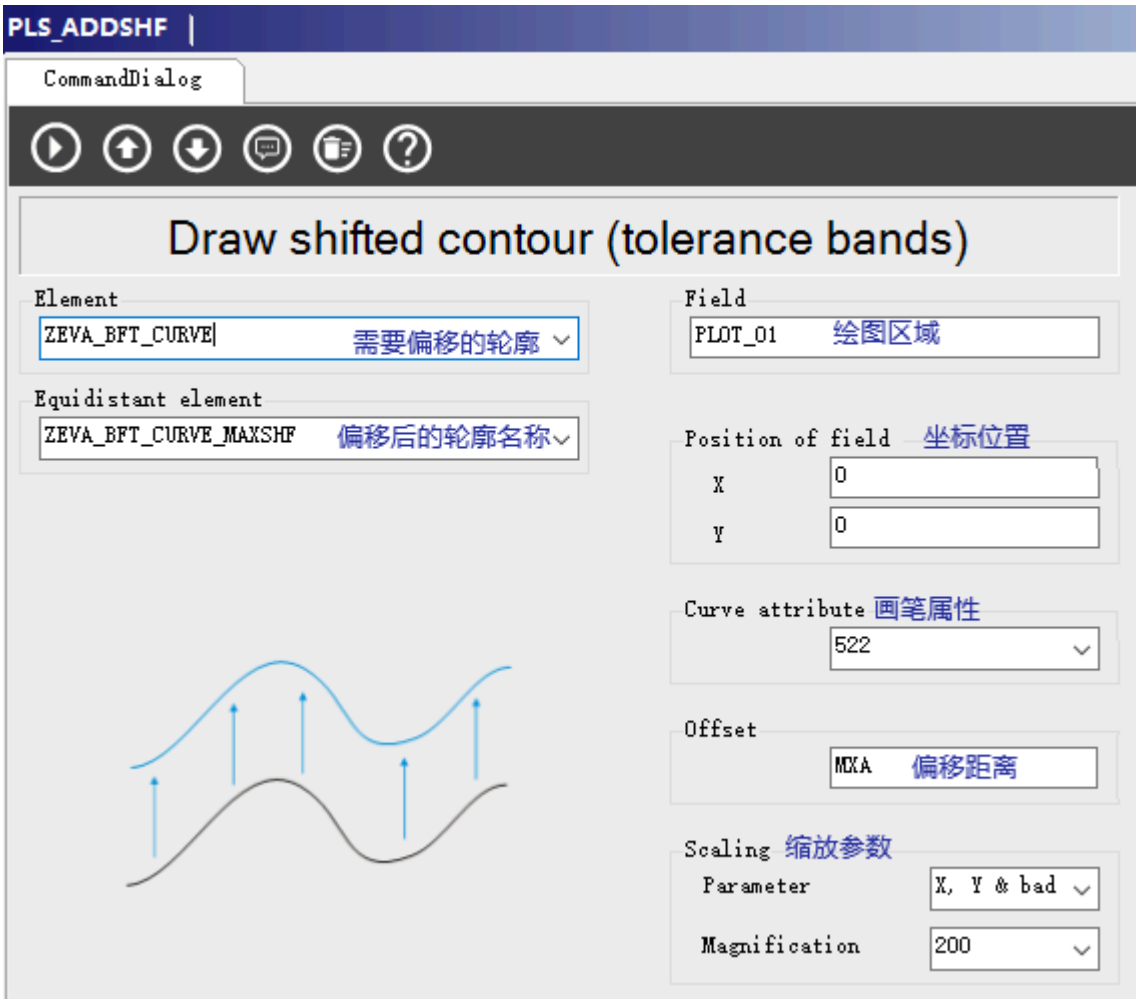
仅在Scaling选择Yes或standard时可以使用，并且只能单向向内收缩（但此时的矢量方向应该是向外的）。如果Scaling选择no，不论如何设置都不缩放偏移。这与之前需要手动控制缩放的做法不符。

PLS_ADDCRVL系列：

使用指令**PLS_ADDSHF**可以快速画出偏移后的公差带轮廓。

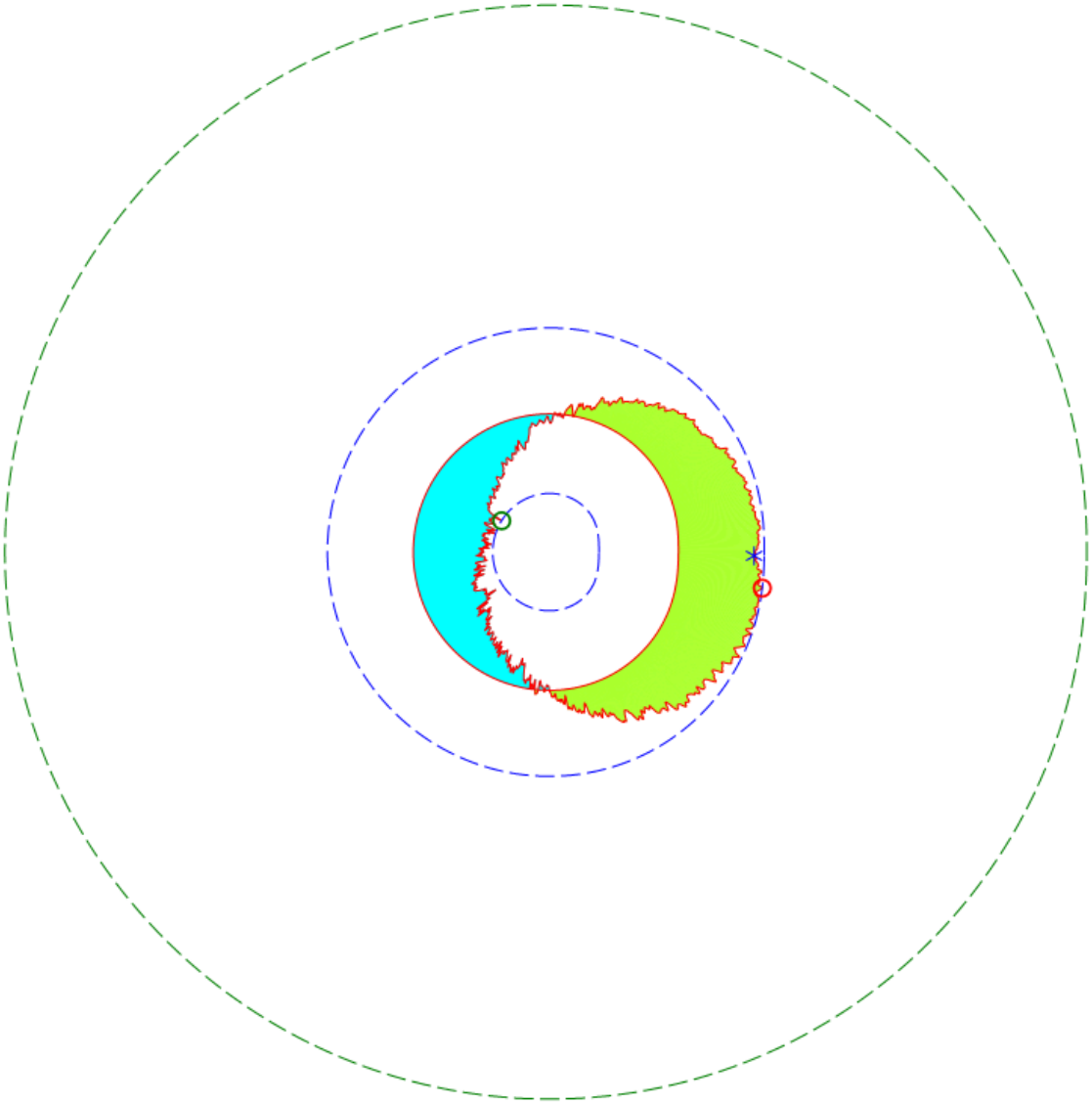
画出公差带：PLS_ADDSHF　复制代码

```
PLS_ADDSHF      (ELE=ZEVA_BFT_CURVE, PFL=PLOT_01, X  =0, Y  =0,
AEQ=ZEVA_BFT_CURVE_MAXSHF, CVA=522, SCF=200, SHF=MXA, SCA=YYY)
PLS_ADDSHF      (ELE=ZEVA_BFT_CURVE, PFL=PLOT_01, X  =0, Y  =0,
AEQ=ZEVA_BFT_CURVE_MIXSHF, CVA=522, SCF=200, SHF=MIA, SCA=YYY)
PLS_ADDSHF      (ELE=ZEVA_BFT_CURVE, PFL=PLOT_01, X  =0, Y  =0,
AEQ=ZEVA_BFT_CURVE_TOLSHF, CVA=422, SCF=200, SHF=0.35, SCA=YYY)
```



- 指令说明
  - **Element**是需要进行偏移的轮廓数据元素。
  - **Equidistant element**是偏移后的曲线名称，可以在原始轮廓的名称后加上一个后缀，如_MAXSHF（最大偏移）、_MIXSHF（最小偏移）、TOLSHF（公差带偏移）。
  - **Field**是定义的作图区域。
  - **Position of field**是坐标信息，一般填0或者空着都可以。
  - **Curve attribute**是画笔信息。打开下拉列表选择或自定义(定义画笔的方法请参考本指南最后的章节**【自定义画笔】**)。
  - **Offset**是偏移距离。在上一章节中，我们已经获得了最大偏差值并保存在变量***MXA***，最小偏差值并保存在变量***MIA***，并且根据图纸得到了轮廓度公差为***0.35***。所以此处我们只需要把对应的参数输入即可。
  - **Parameter**是缩放条件，与绘制轮廓元素保持一致即可。一般为XY与Bad Points。
  - **Magnification**为缩放倍率，与绘制轮廓元素保持一致即可，此案例为200。

用三条指令分别将轮廓画出，得到的效果如下：



为了理解如何实现绘制偏移的公差带，以及应对可能指令不能满足的特殊情况，在此我提供一个更为底层的解决方案——**将轮廓的坐标点进行偏移计算，将其转变为三个不同公差带的轮廓图形。**

补充：如果深入考究细节的话，此处不应该将原始轮廓进行偏移计算，而是应该将理论线进行偏移。但是由于我们根据图纸的基准，可能需要进行**最佳拟合**，此时理论线和拟合后的原始轮廓一定会出现平移、旋转甚至缩放的偏差。最严谨的做法需要将最佳拟合的转换保存一个TRA数据，再对理论线进行TRA变换，以保证此时理论线和轮廓线的中心位置重合，旋转角度也重合。但是由于此案例的图形报告中，这三条公差带线仅仅作为参考，并没有非常精准的要求。正常加工的产品，原始轮廓在形状上和理论线的差别肉眼无法区分，所以直接对原始轮廓的点进行了转换，省去将理论线进行TRA转换的步骤。但工件若有直观的毛刺或缺

口，此时仍然使用原始轮廓制作公差带图形的话，会导致公差带的理论图形也出现突起或缺口。这是不应该的。（在使用PLS_ADDSHF时也需要相同的考虑）

首先，我们需要完成数据的复制和偏移计算。这里的步骤为通用的前置准备，不区分系列。

```
GETVALS          (OBJ=EVA_BFT_CURVE, RDS=$j, REA=TOTAL_PTS)
COPY             (FRM=EVA_BFT_CURVE, TO =SL_TRB, DEL=Y, TYP=ELE,
STY=APT, FRS=1, LST=TOTAL_PTS)
COPY             (FRM=EVA_BFT_CURVE, TO =SU_TRB, DEL=Y, TYP=ELE,
STY=APT, FRS=1, LST=TOTAL_PTS)
COPY             (FRM=EVA_BFT_CURVE, TO =TO_TRB, DEL=Y, TYP=ELE,
STY=APT, FRS=1, LST=TOTAL_PTS)
DO               (NAM=LOOPNO, BGN=1, END=TOTAL_PTS)
    GETVALS          (OBJ=EVA_BFT_CURVE.ACT.PTS(LOOPNO), TYP=ELE, RDS=
(X,Y,Z,U,V,W,A), REA=(SUX,SUY,SUZ,SUU,SUV,SUW,SUA))
    PUTVALS          (OBJ=SU_TRB.ACT.PTS(LOOPNO), RDS=(X,Y), VAL=
(SUX+SUU*200*MXA,SUY+SUV*200*MXA))
    PUTVALS          (OBJ=SL_TRB.ACT.PTS(LOOPNO), RDS=(X,Y), VAL=
(SUX+SUU*200*MIA,SUY+SUV*200*MIA))
    PUTVALS          (OBJ=TO_TRB.ACT.PTS(LOOPNO), RDS=(X,Y), VAL=
(SUX+SUU*200*0.35,SUY+SUV*200*0.35))
ENDDO
```

- 命令解释
  - 获取轮廓数据的总点数
  - 将轮廓数据的实测点复制到新元素SL_TRB（最小点的位置）
  - 将轮廓数据的实测点复制到新元素SU_TRB（最大点的位置）
  - 将轮廓数据的实测点复制到新元素TO_TRB（允许的最大公差位置）
  - 执行循环（总次数为总点数）
    - 获取轮廓数据的第n个点的坐标位置（n为当前循环次数）
    - 将SU_TRB中第n个APT点的XY坐标数据通过偏移公式（X+放大*U***最大**偏差）（Y+放大*V***最大**偏差）计算得出并赋值
    - 将SL_TRB中第n个APT点的XY坐标数据通过偏移公式（X+放大*U***最小**偏差）（Y+放大*V***最小**偏差）计算得出并赋值
    - 将TO_TRB中第n个APT点的XY坐标数据通过偏移公式（X+放大*U***允许**公差）（Y+放大*V***允许**公差）计算得出并赋值（此案例中允许的最大公差为0.35）
  - 结束循环

通过上述命令，我们可以获得三条轮廓线，其中两条和偏差轮廓放大后的最大、最小点相切，还有一条为偏差放大后相应公差带的位置。接下来只需要把他们画出来即可。

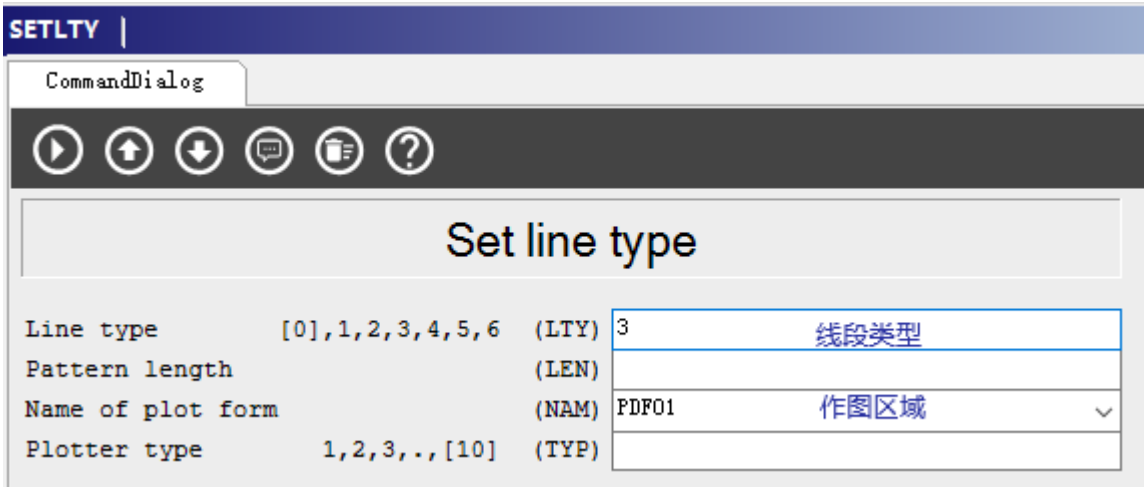
DRWPLY系列：

画出公差带 复制代码

```
SETLTY          (NAM=PDF01, LTY=3)
DRWPLY          (NAM=SU_TRB, ASC=N, DEV=Y, DRP=0, PEN=5, OPN=N, A_O=XY)
DRWPLY          (NAM=SL_TRB, ASC=N, DEV=Y, DRP=0, PEN=5, OPN=N, A_O=XY)
DRWPLY          (NAM=TO_TRB, ASC=N, DEV=Y, DRP=0, PEN=4, OPN=N, A_O=XY)
```

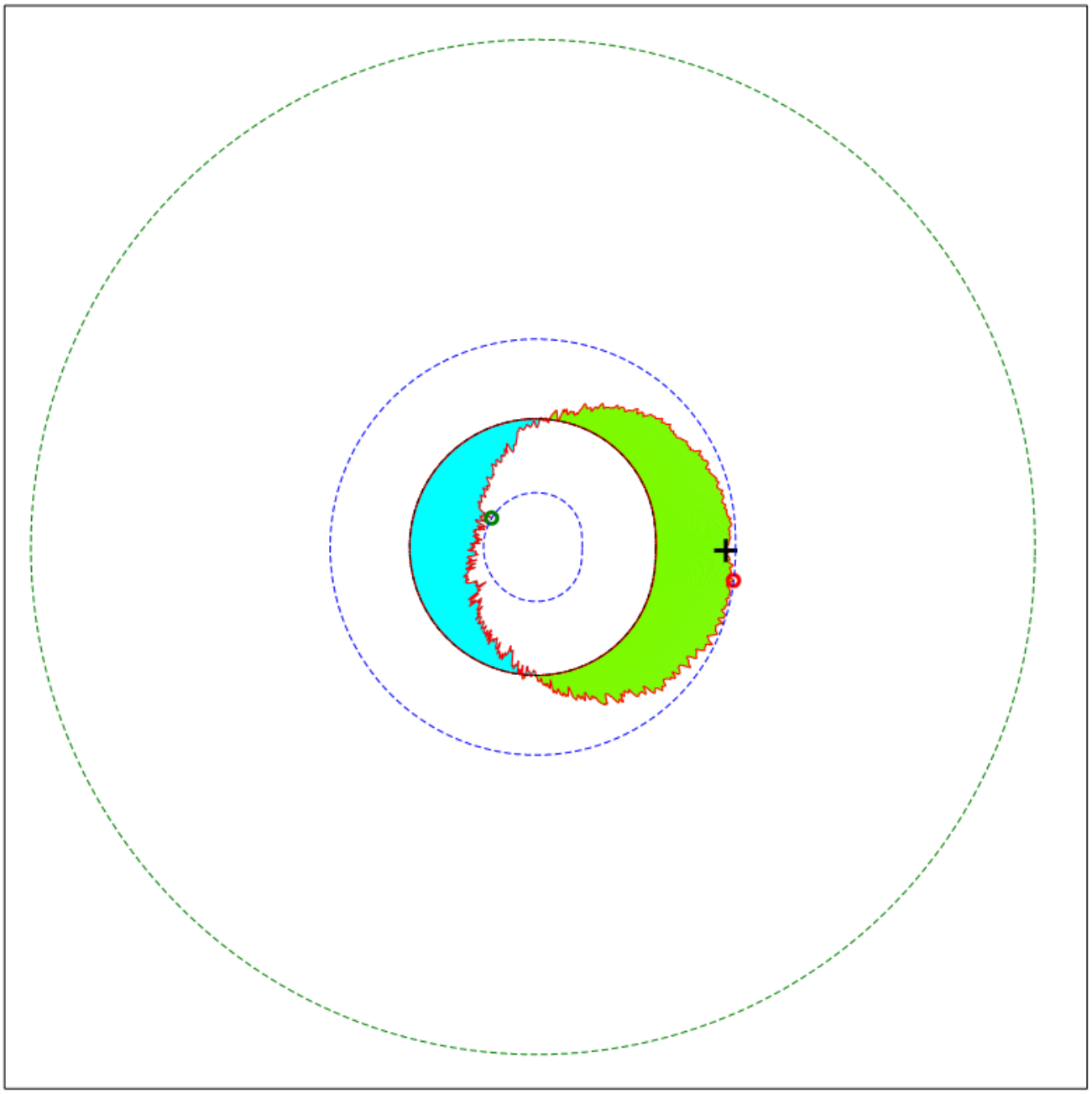
DRWPLY的使用和【画出轮廓】部分一样，唯一区别是**此处不要偏差放大。**

为了让公差带图形和实测数据有更好的区分，我们除了可以更改作线颜色，还可以为公差带图形设置不同的线条类型。使用**SETLTY**指令实现。



LTY为线段类型，共有7种类型，包含了实线、虚线、点划线等模式。具体格式可以参考该指令的Quindos在线帮助文档。此处的3为短虚线。

执行命令获得如下效果：

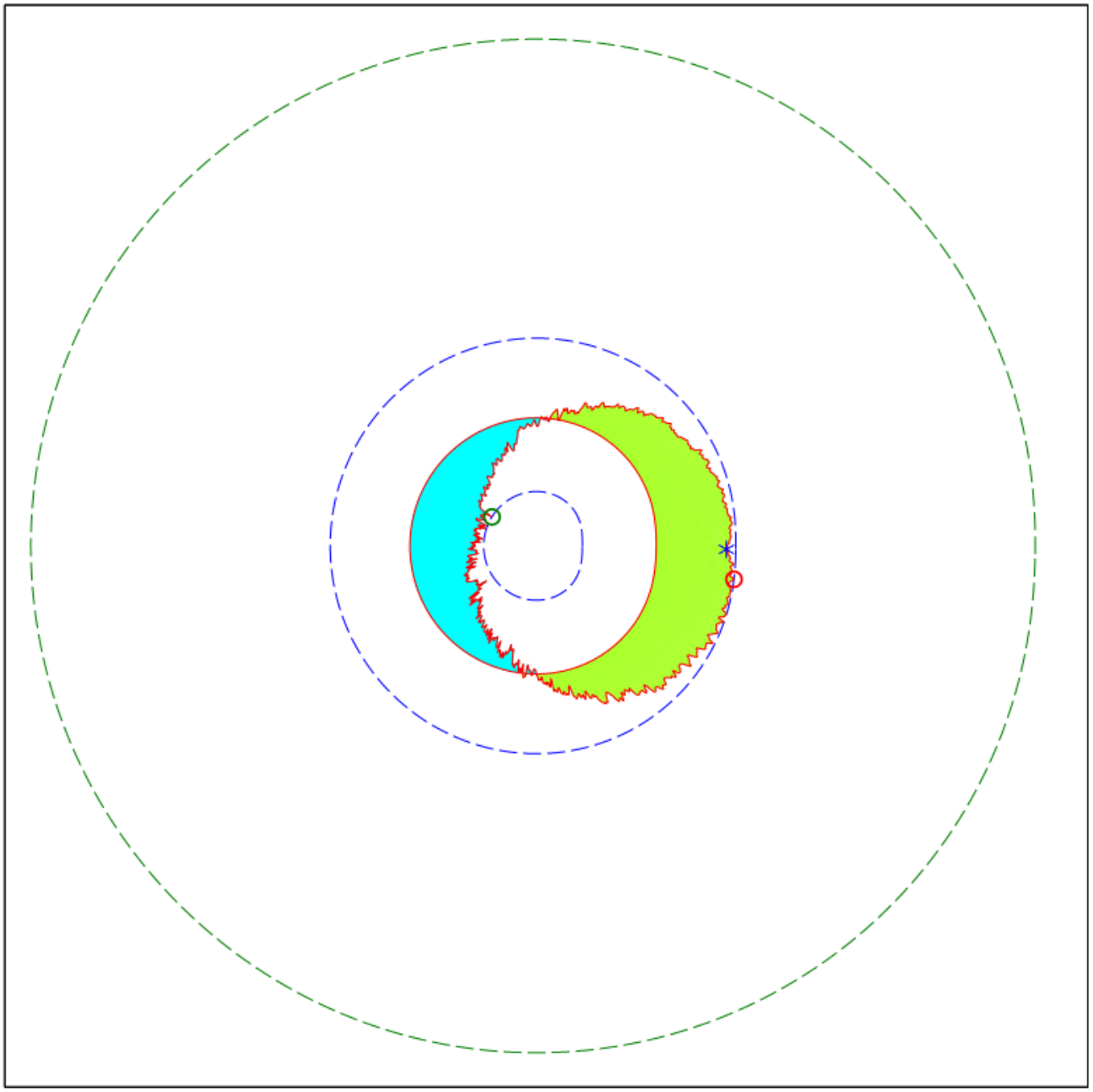


PLS_ADDCRVL系列：

画出公差带 复制代码

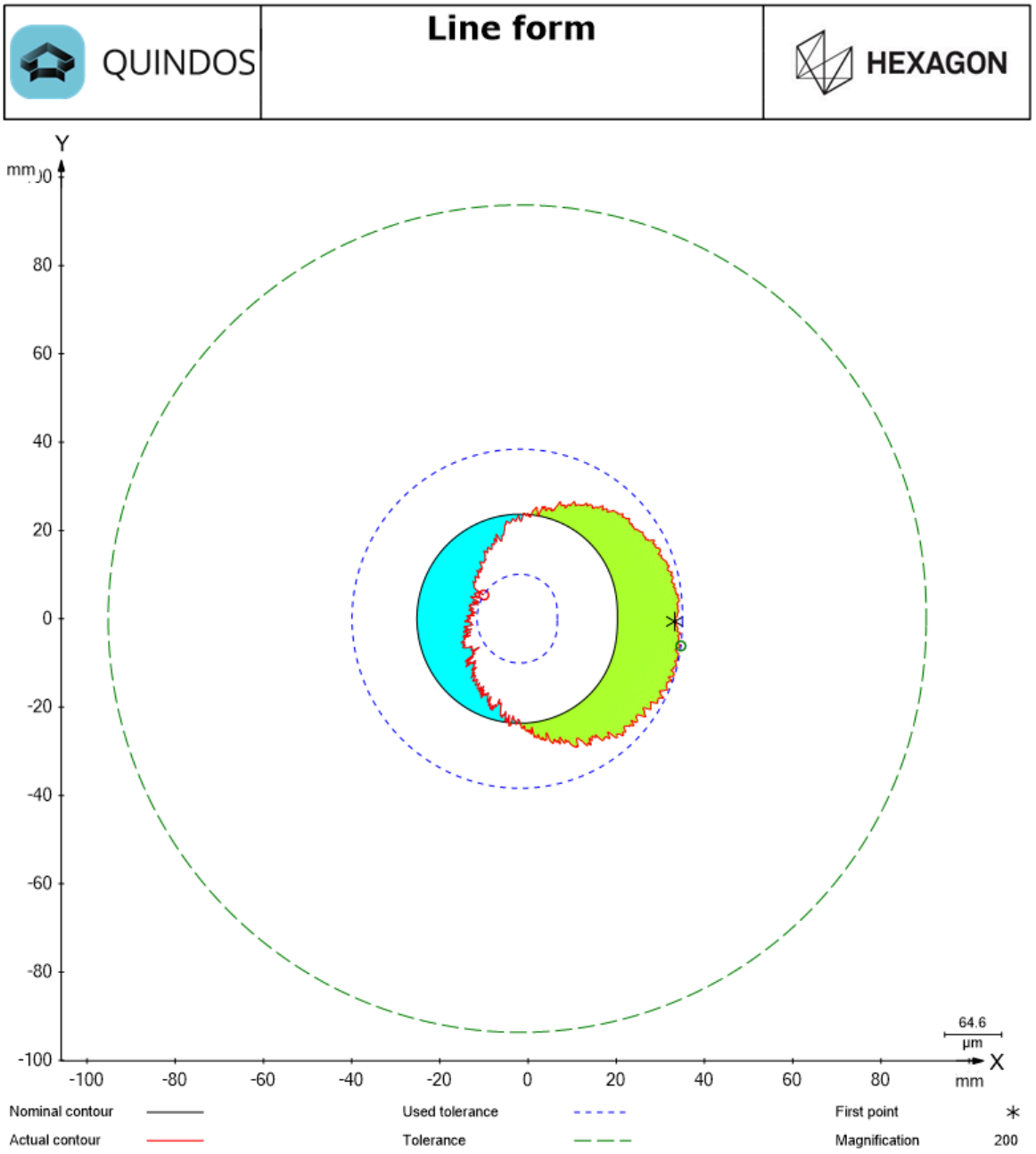
```
PLS_ADDCRVL     (ELE=SL_TRB, PFL=PLOT_01, CVA=(1001,-1,-1), SCA='  N',
SCF=200, FLL=, A_O=XY, STP=1)
PLS_ADDCRVL     (ELE=SU_TRB, PFL=PLOT_01, CVA=(1001,-1,-1), SCA='  N',
SCF=200, FLL=, A_O=XY, STP=1)
PLS_ADDCRVL     (ELE=TO_TRB, PFL=PLOT_01, CVA=(1005,-1,-1), SCA='  N',
SCF=200, FLL=, A_O=XY, STP=1)
```

继续使用**PLS_ADDCRVL**指令将三个图形画出，画笔颜色和线段类型调整为合适的设置即可。执行命令得到如下效果：



这一章节我们已经将公差带画出，与测量轮廓本身相关的部分已经全部讲解完毕，但与之相关的还有一个元素——坐标系。其绘制的过程和图形仍然具有一定关联。

画出坐标系：



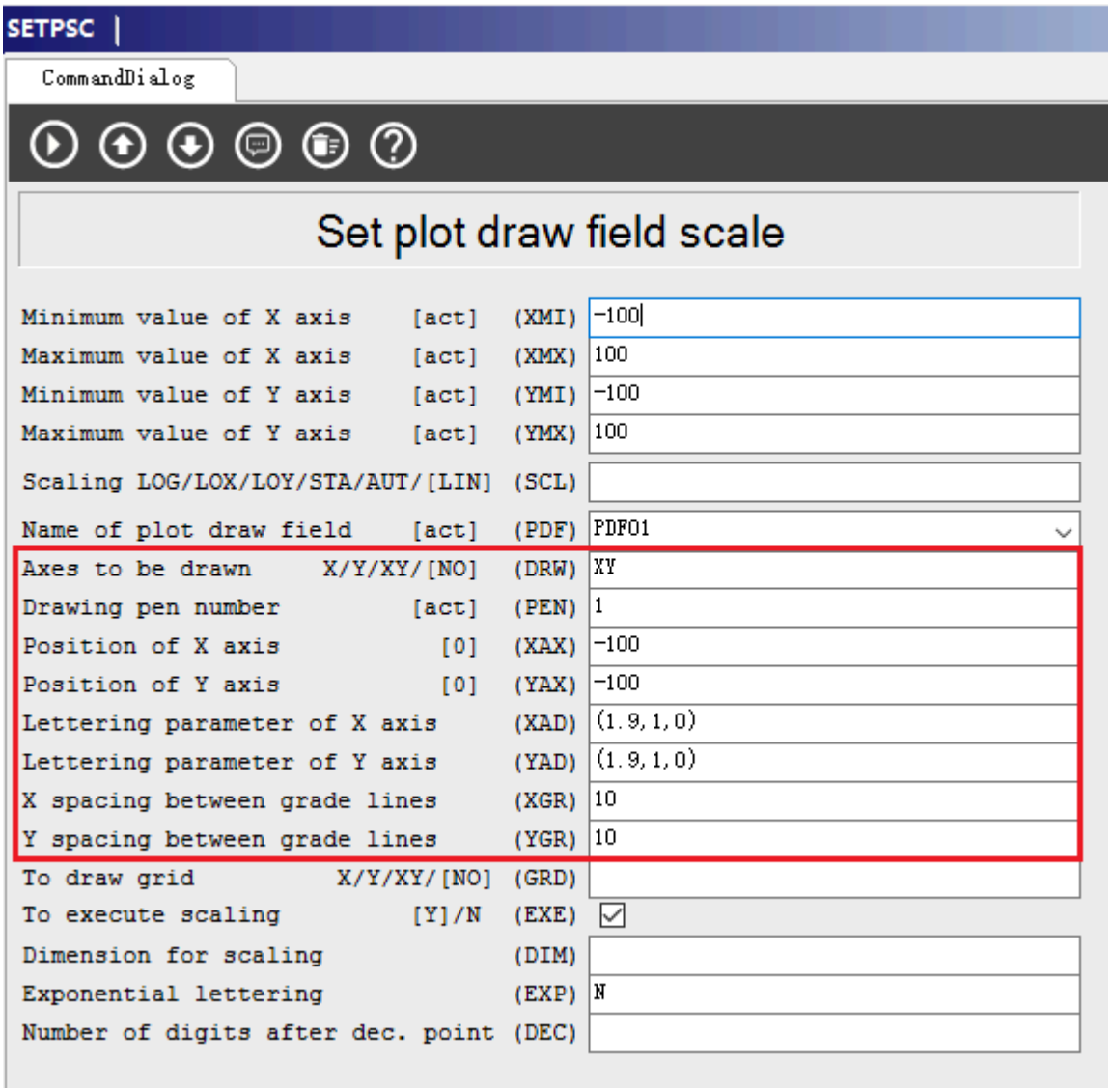
我们再回到最开始的这张官方图中。通过前面的所有了解，我们已经明白图中的坐标系是用于控制图形缩放比率的。严格意义上来说，其mm的刻度与被放大的轮廓度误差是完全不同的。此时的刻度仅对轮廓形状本身有意义，对于后面画的三个公差带圈，虽有定位作用，但不具备识图的作用。所以坐标轴是用于展现原始轮廓的度数，还是展现几何公差的读数，需要二者选其一。后续讲解采用和官方图一样的方案，使用对应轮廓图形的mm坐标轴。

DRWPLY系列：

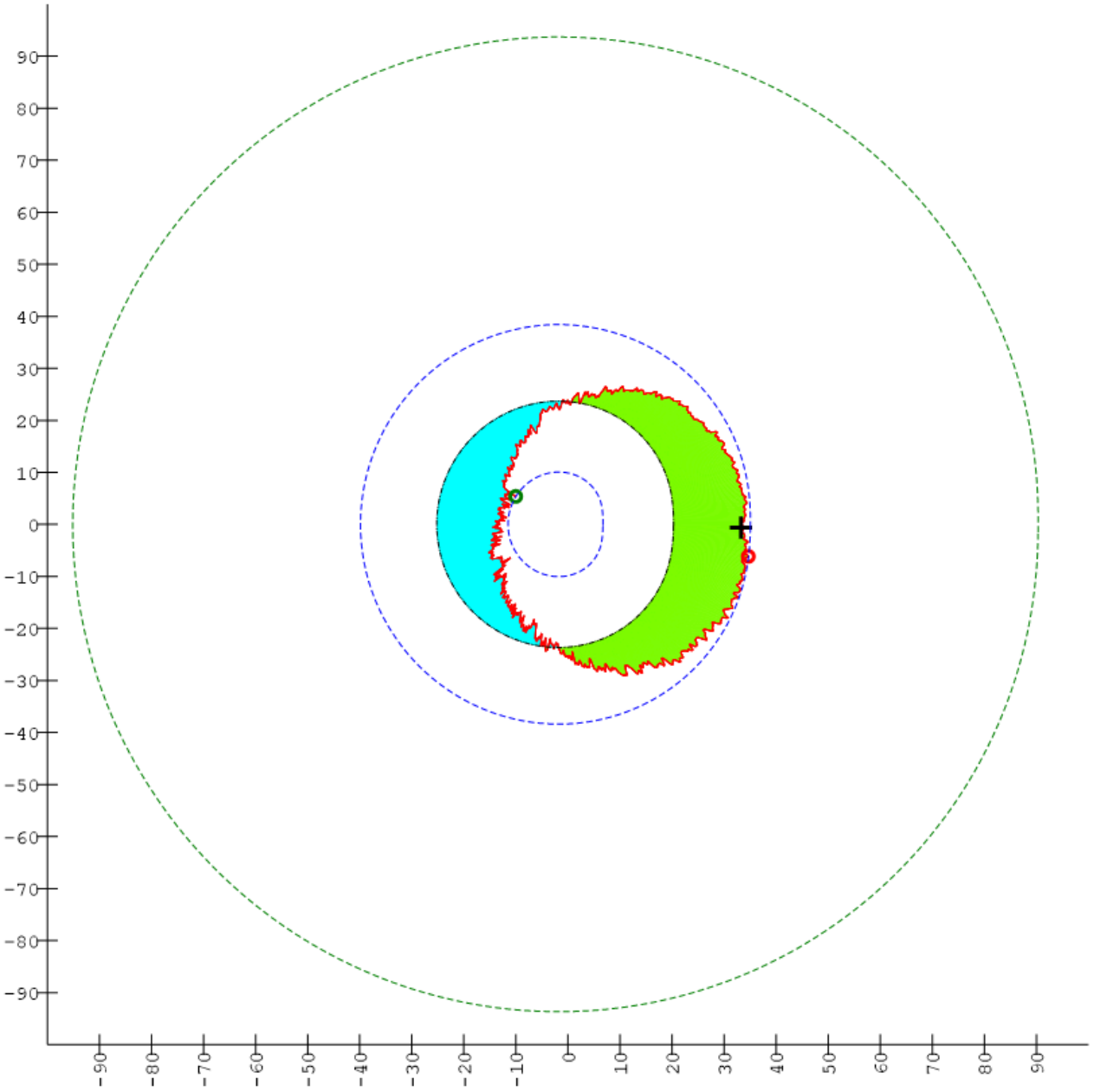
利用**SETPSC**指令中绘制坐标轴的功能，我们可以快速画出一个坐标系。设置如下，指令功能的详细介绍可以到**【画出轮廓-进阶02：控制缩放坐标系】**中回顾。

画出坐标轴 复制代码

```
SETPSC          (PDF=PDF01, XMI=-100, XMX=100, YMI=-100, YMX=100,
XGR=10, YGR=10, XAX=-100, XAD=(1.9,1,0), YAX=-100, YAD=(1.9,1,0),
DRW=XY, PEN=1, EXP=N)
```

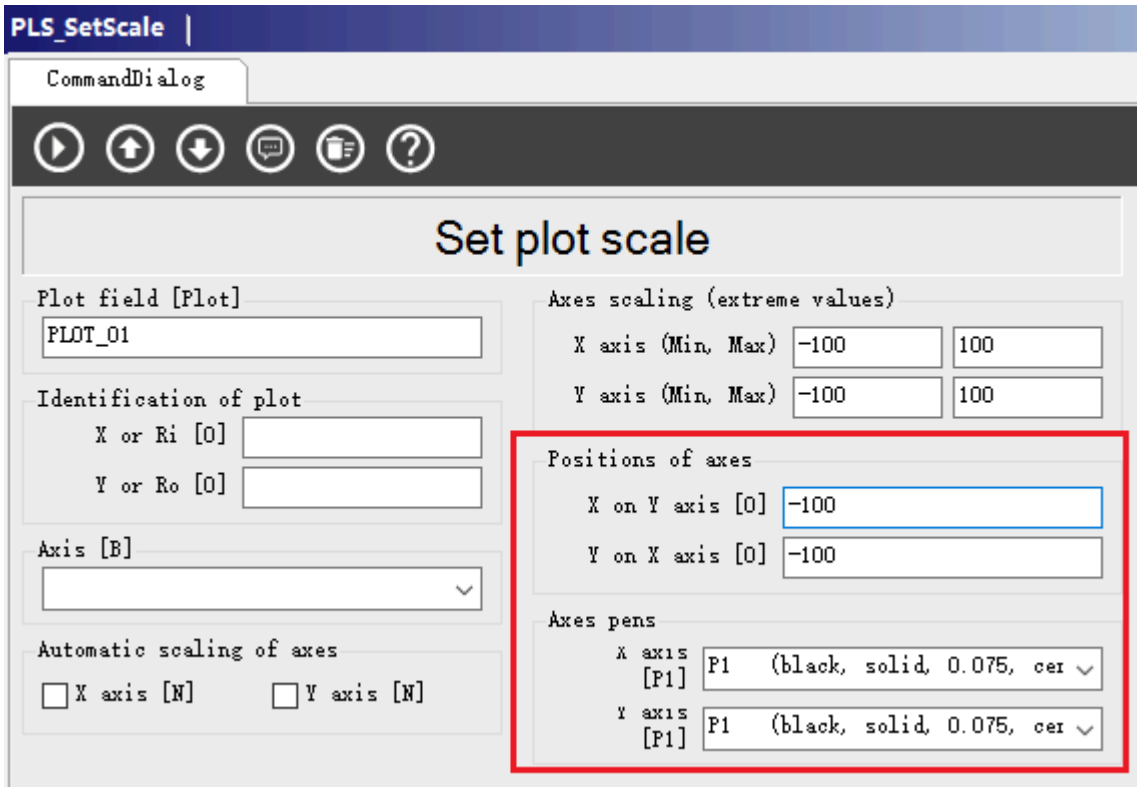


在之前已经画出图形的基础上，执行命令可以得到如下效果：

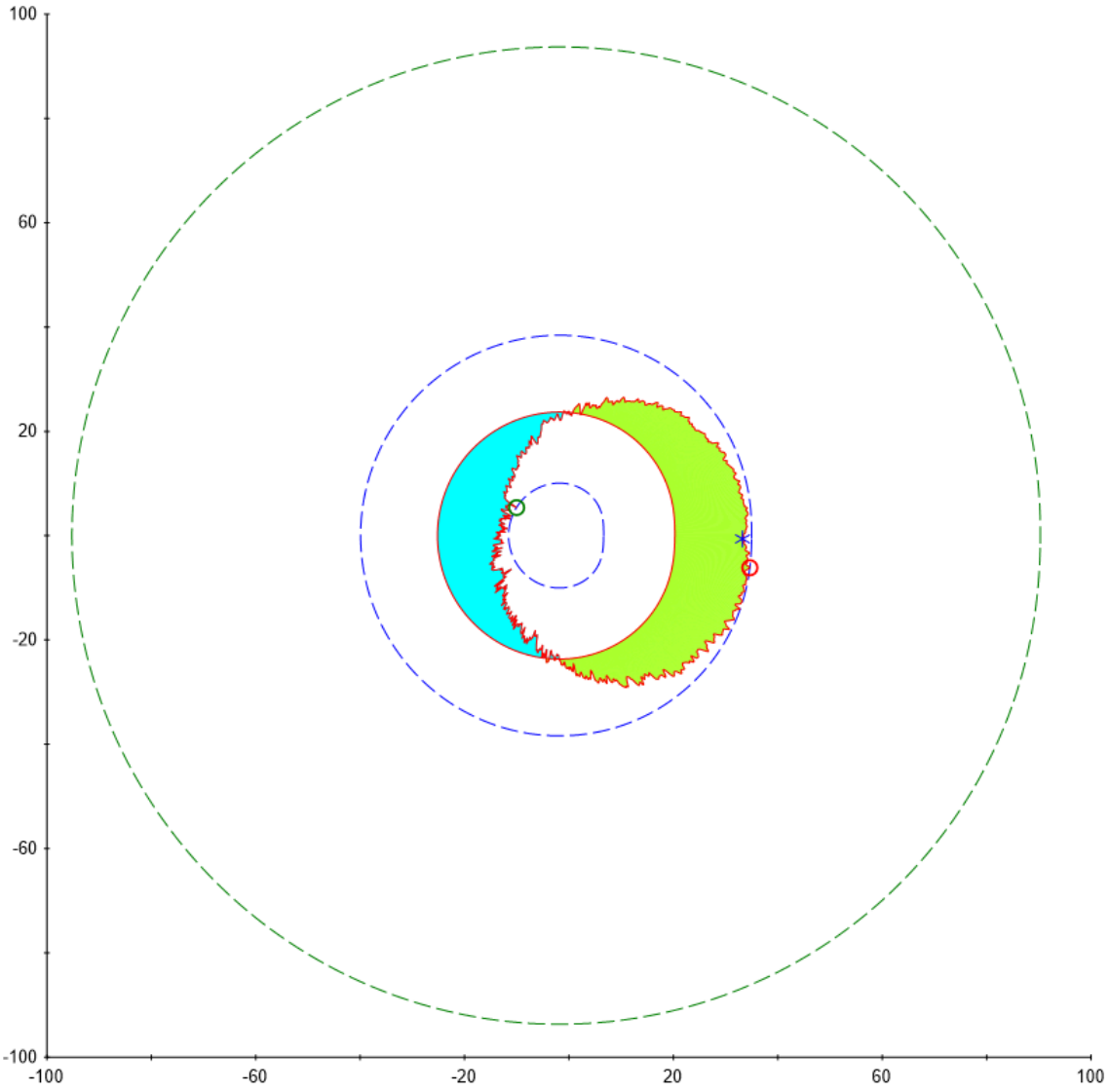


PLS_ADDCRVL系列：

- 方案一：在**【画出轮廓-进阶02：控制缩放坐标系】**中，用于设置缩放坐标系的指令**PLS_SetScale**可以进行快速绘制：

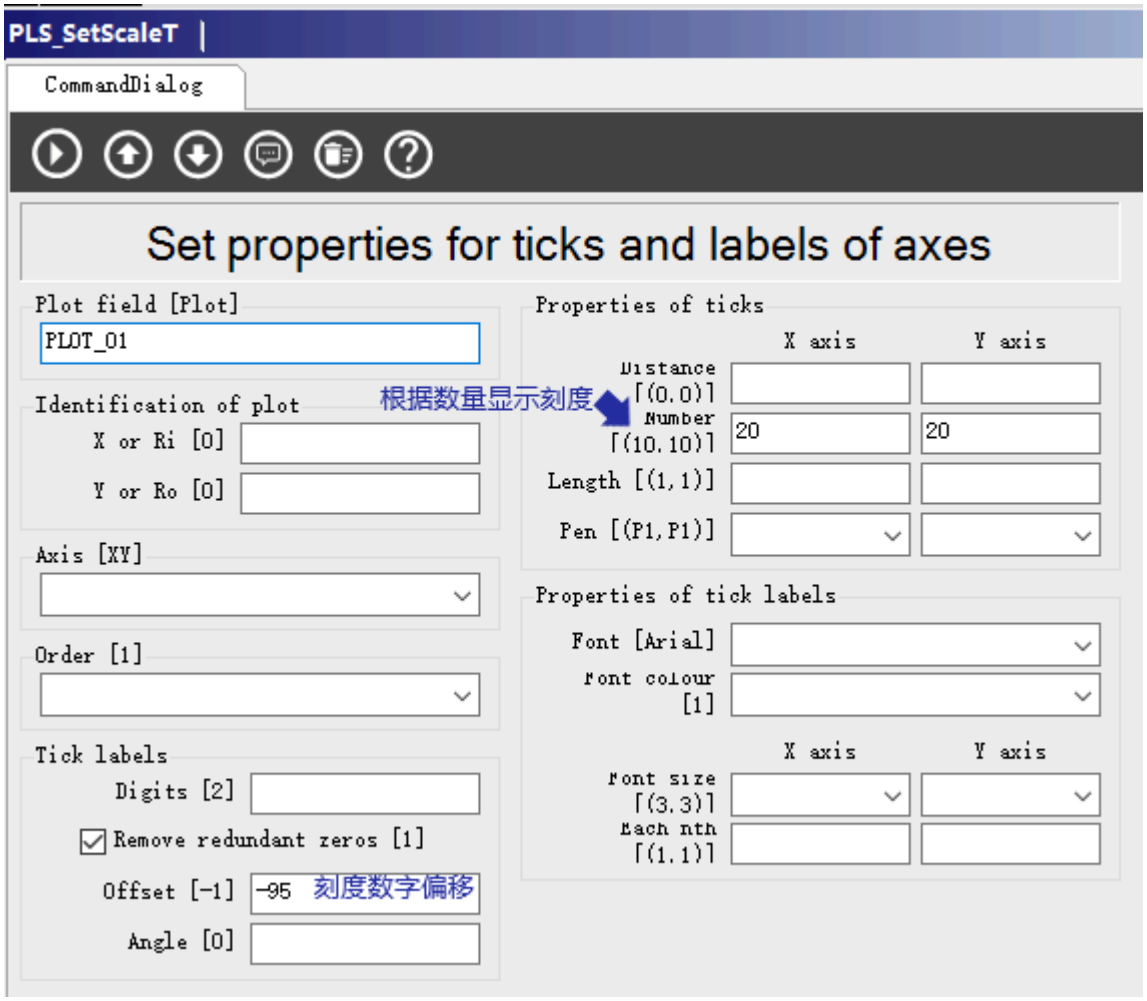


- 效果如下。可以看出坐标系已经出现，但是其刻度的设置在这个指令中没有更改的地方。如果需要进一步定制坐标轴的详细内容，可以采取方案二。

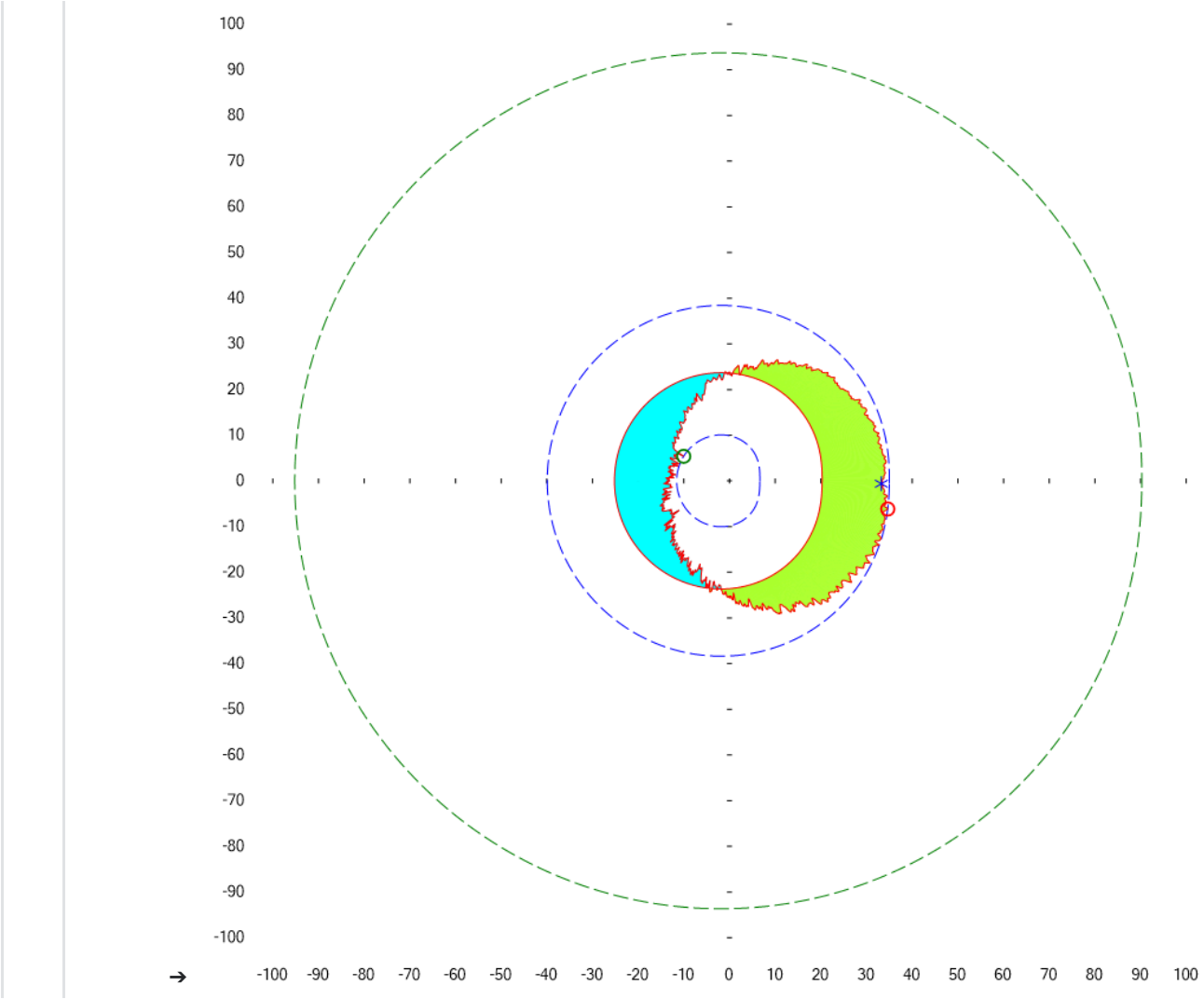


- 方案二：

- 先尝试使用**PLS_SetScaleT**指令进行绘图。



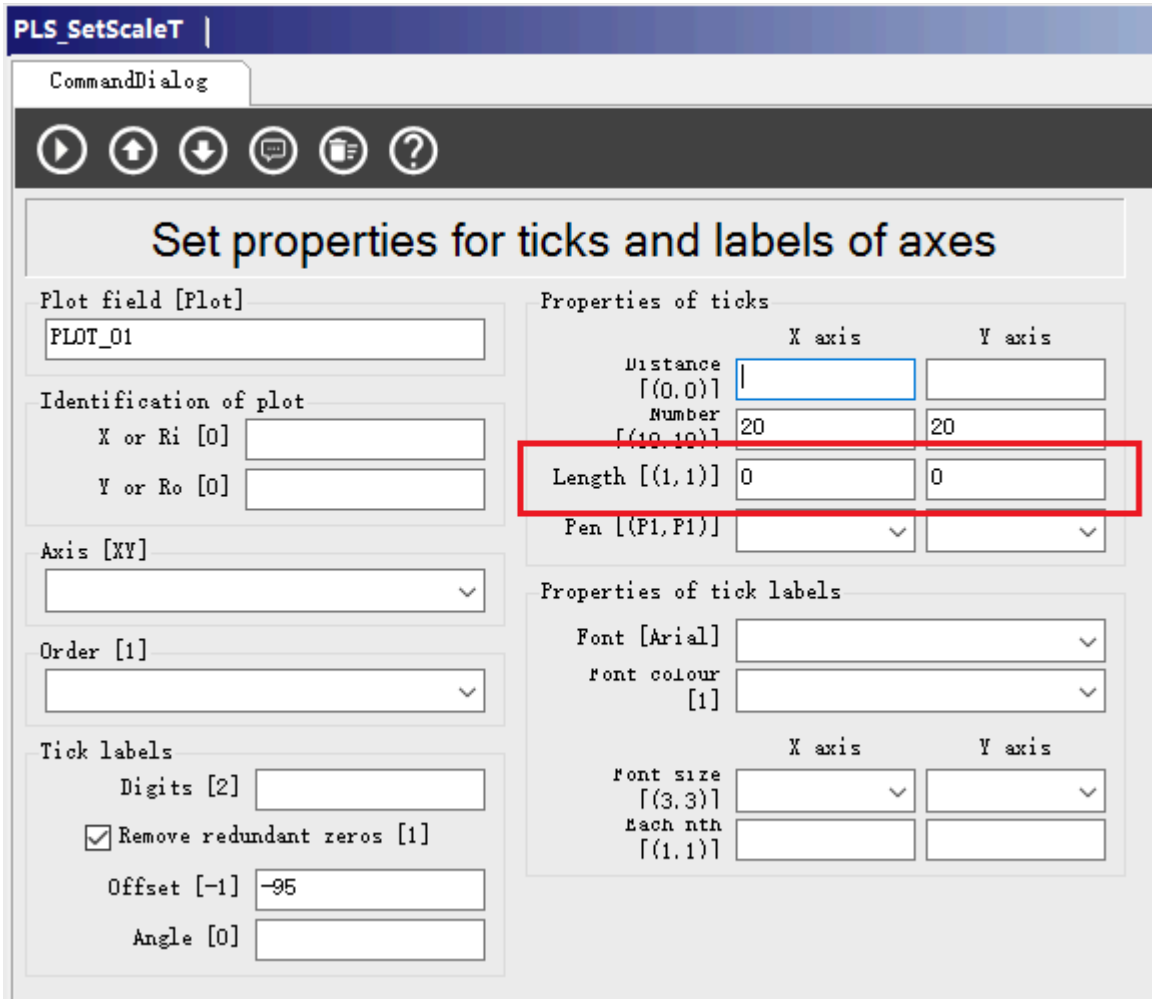
- 坐标系可以画出。但这种绘图方式我目前没有找到方法偏移刻度线的显示。如果对效果不满意，希望拥有完整的坐标系轴线、刻度线等标识。可以采用**PLS_SetScaleT**结合**PLS_DRAWLIN**结合的方式，或直接使用**PLS_DRAWLIN**和**PLS_DRAWREA**完全自定义坐标系的绘制。



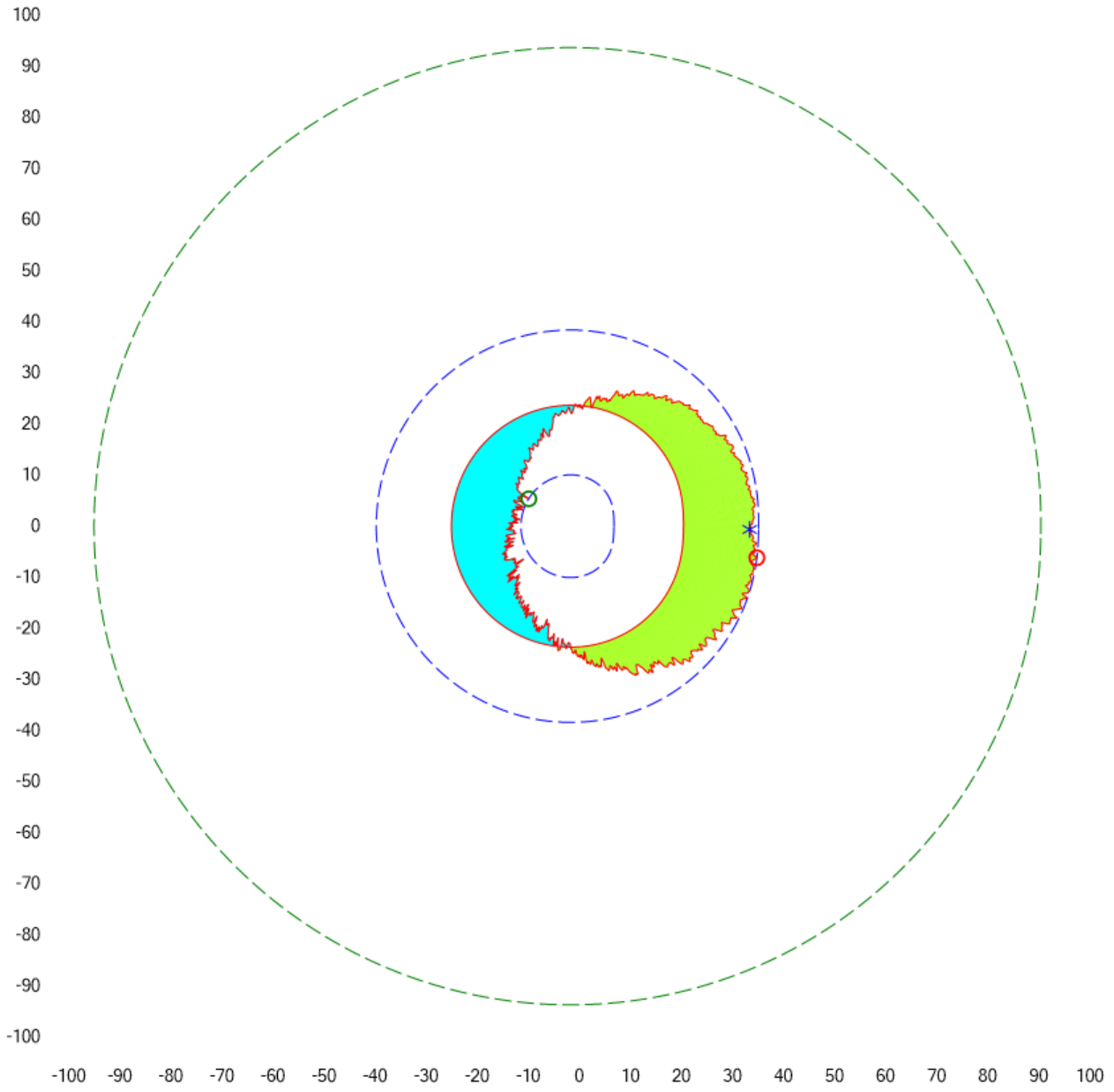
接下来介绍**PLS_SetScaleT**结合**PLS_DRAWLIN**的方式进行作图，不过其中的原理与其他方式是相同的。不管是使用**PLS_DRAWLIN**与**PLS_DRAWREA**或是使用**DRWLIN**与**DRWREA**，核心都是通过**DO循环**将线段和数字画到正确的位置上。

首先抑制**PLS_SetScaleT**中刻度的生成，将长度从默认的1设为0。

```
PLS_SetScaleT   (PFL=PLOT_01, ATR=(20,20), TLE=(0,0), TPA=(,1,-95))
```

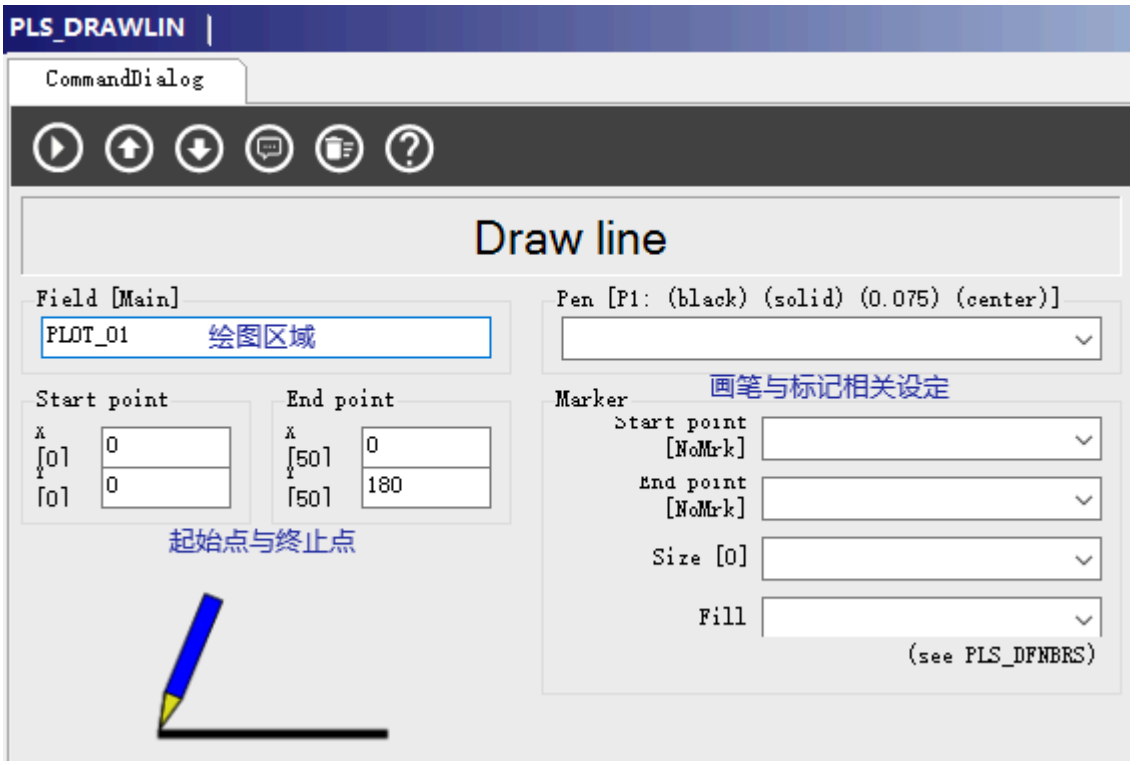


如此，我们就已经将数字部分绘制完毕。

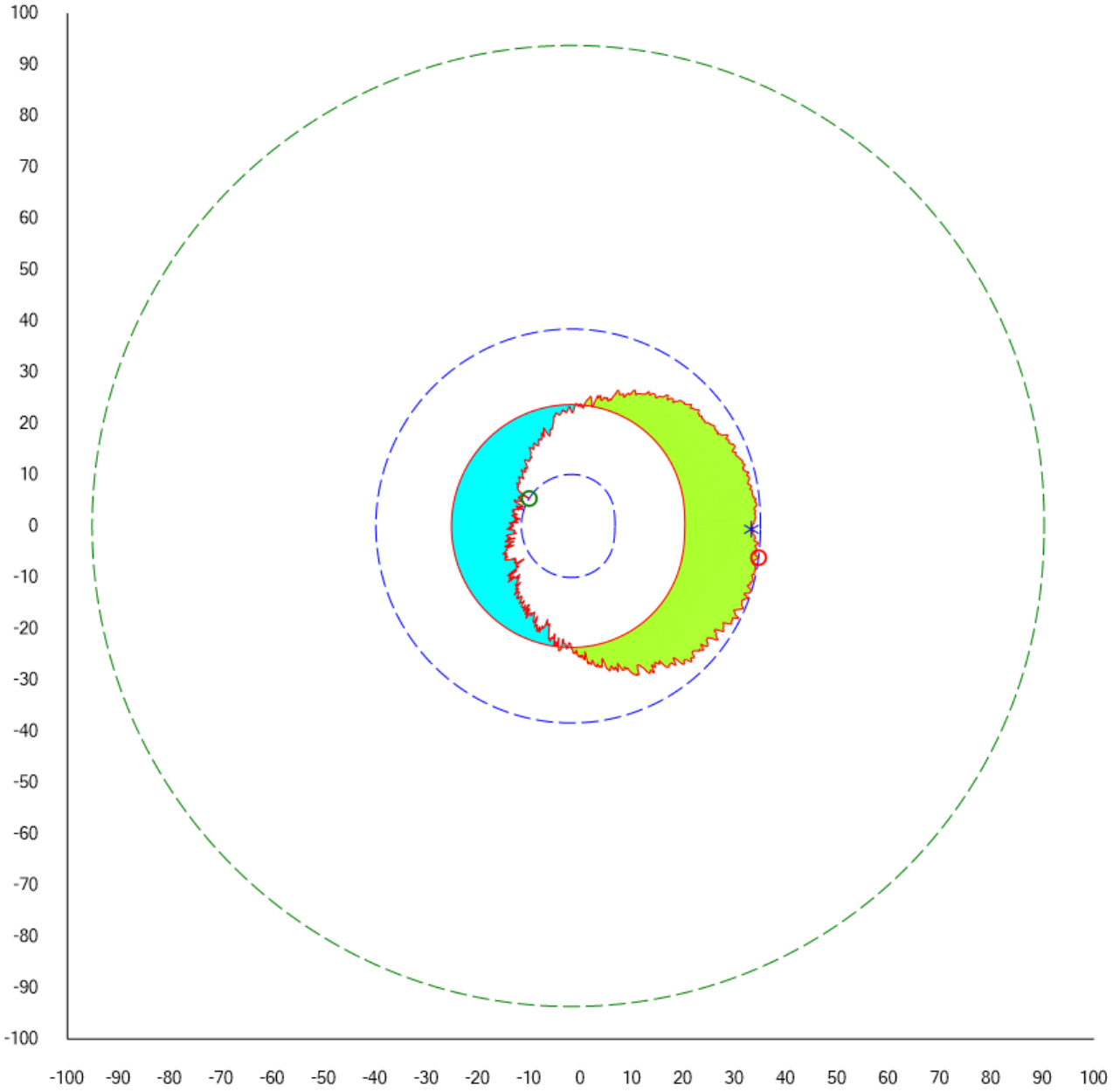


接下来，画出X和Y轴的轴线。使用**PLS_DRAWLIN**指令绘制。

```
PLS_DRAWLIN     (FRA=PLOT_01, X1 =0, Y1 =0, X2 =0, Y2 =180)
PLS_DRAWLIN     (FRA=PLOT_01, X1 =0, Y1 =180, X2 =180, Y2 =180)
```



沿着绘图区域的边画两个长度都是180，方向不同的直线。得到如下效果：



然后，根据坐标轴上的数字可以看出，我们在一条轴上需要画21个刻度，对应21个数字，所以循环的总次数是21。

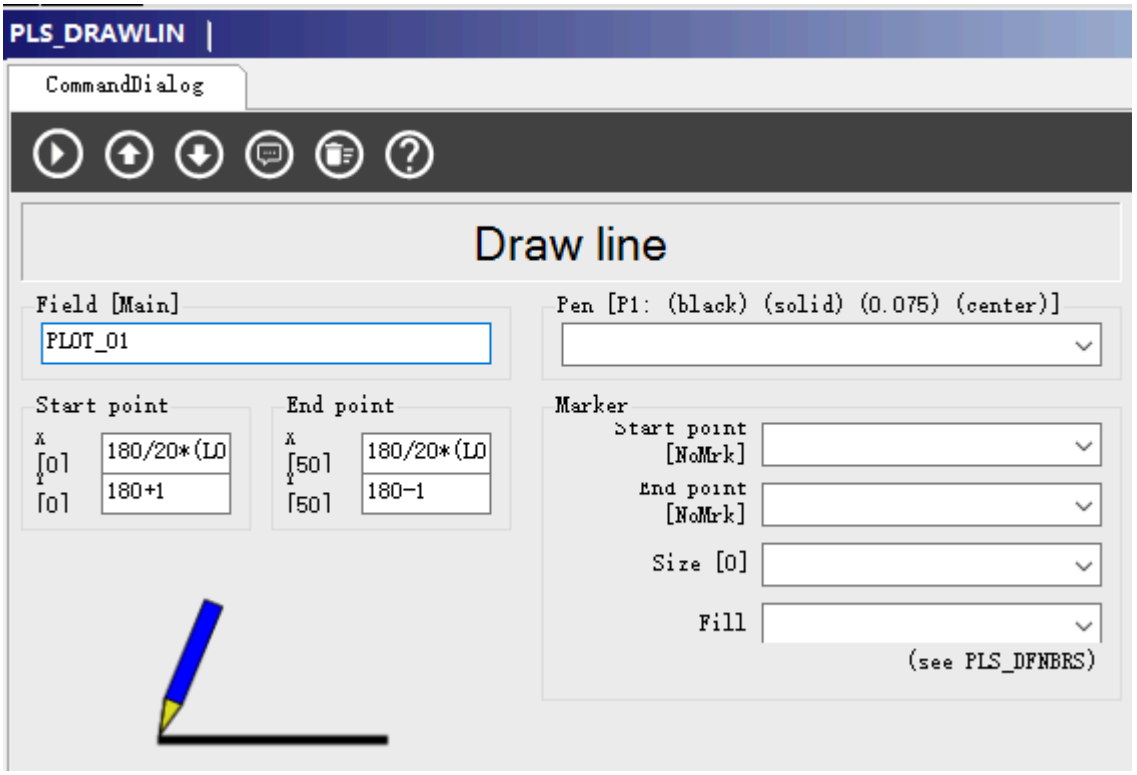
下方代码块中：

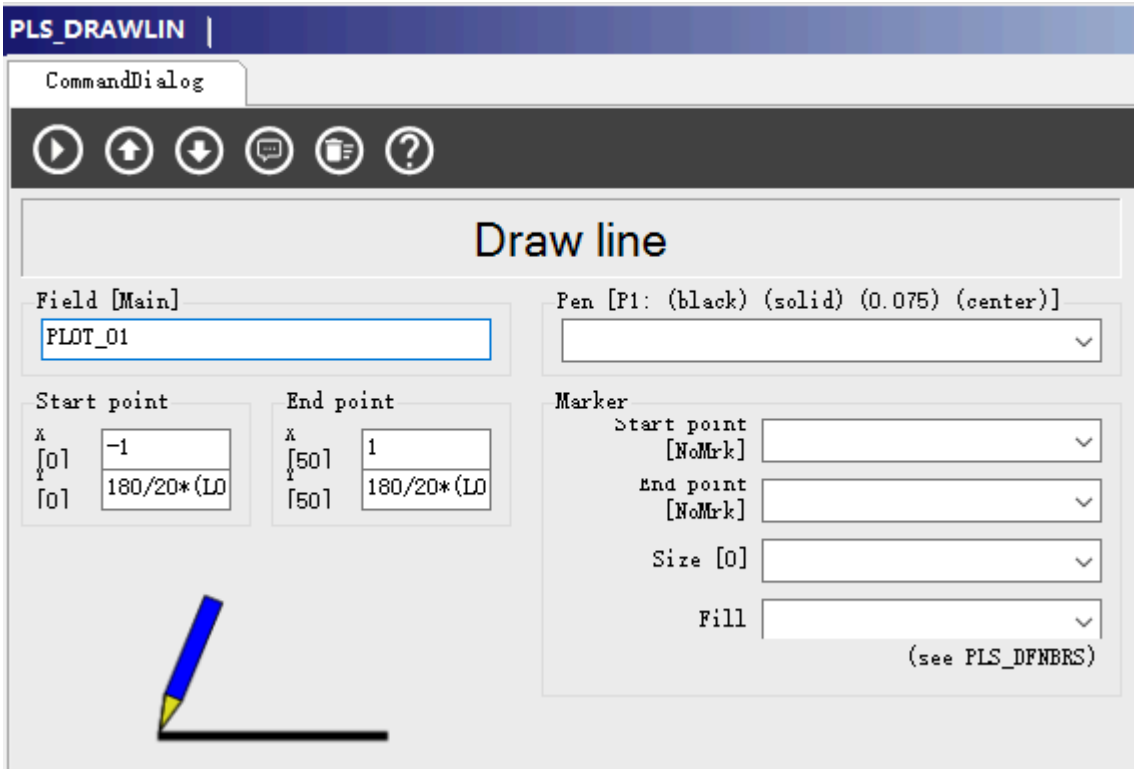
第一行的**PLS_DRAWLIN**将每次在X轴坐标上画出2个单位长的线段，间隔（180/20）的距离进行循环。**注意Y轴的坐标**并非-1到1，而是（180-1）到（180+1），原因如前面章节提过。此时是使用的**作图区域坐标系**。

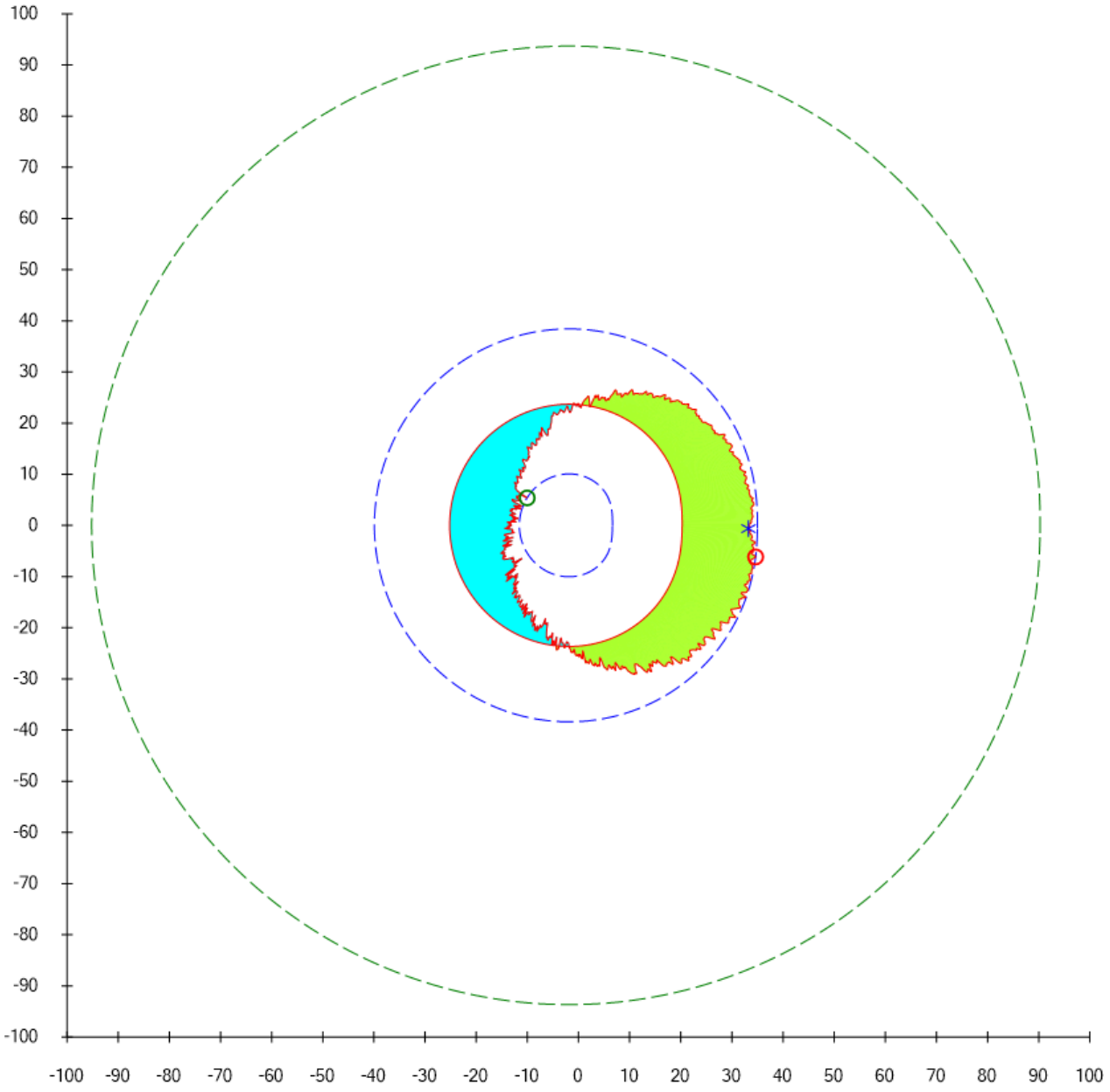
第二行的**PLS_DRAWLIN**将每次在Y轴坐标上画出2个单位长的线段，间隔（180/20）的距离进行循环。

```
DO              (NAM=LOOPNO, BGN=1, END=21)
```

```
    PLS_DRAWLIN     (FRA=PLOT_01, X1 =180/20*(LOOPNO-1), Y1 =180+1, X2
=180/20*(LOOPNO-1), Y2 =180-1)
    PLS_DRAWLIN     (FRA=PLOT_01, X1 =-1, Y1 =180/20*(LOOPNO-1), X2 =1,
Y2 =180/20*(LOOPNO-1))
ENDDO
```



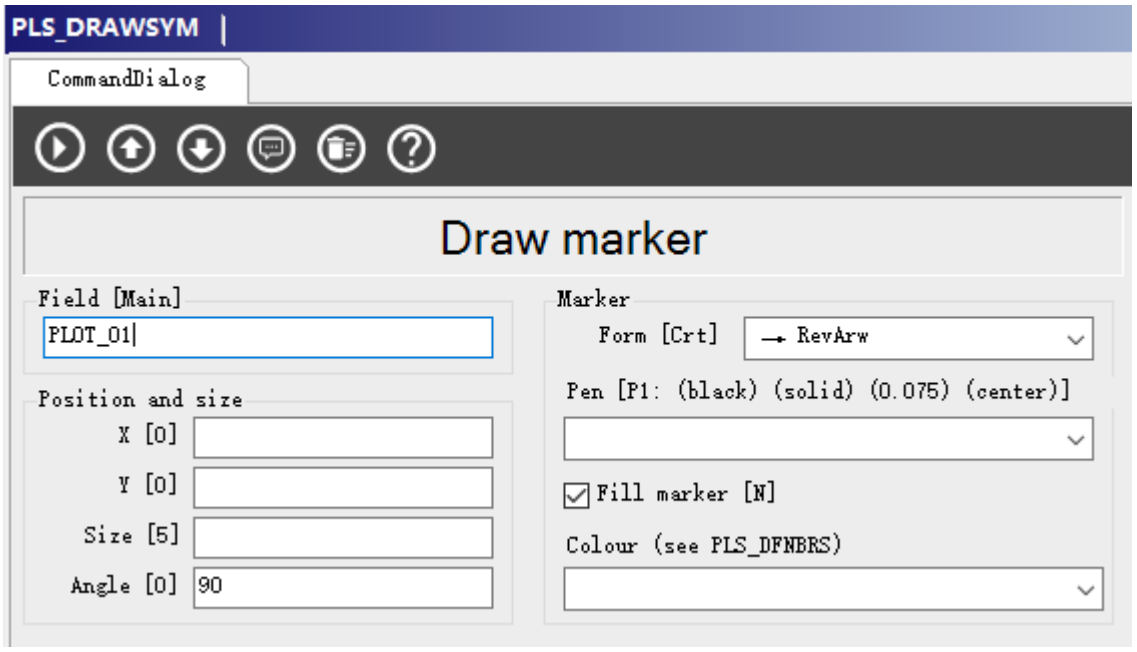


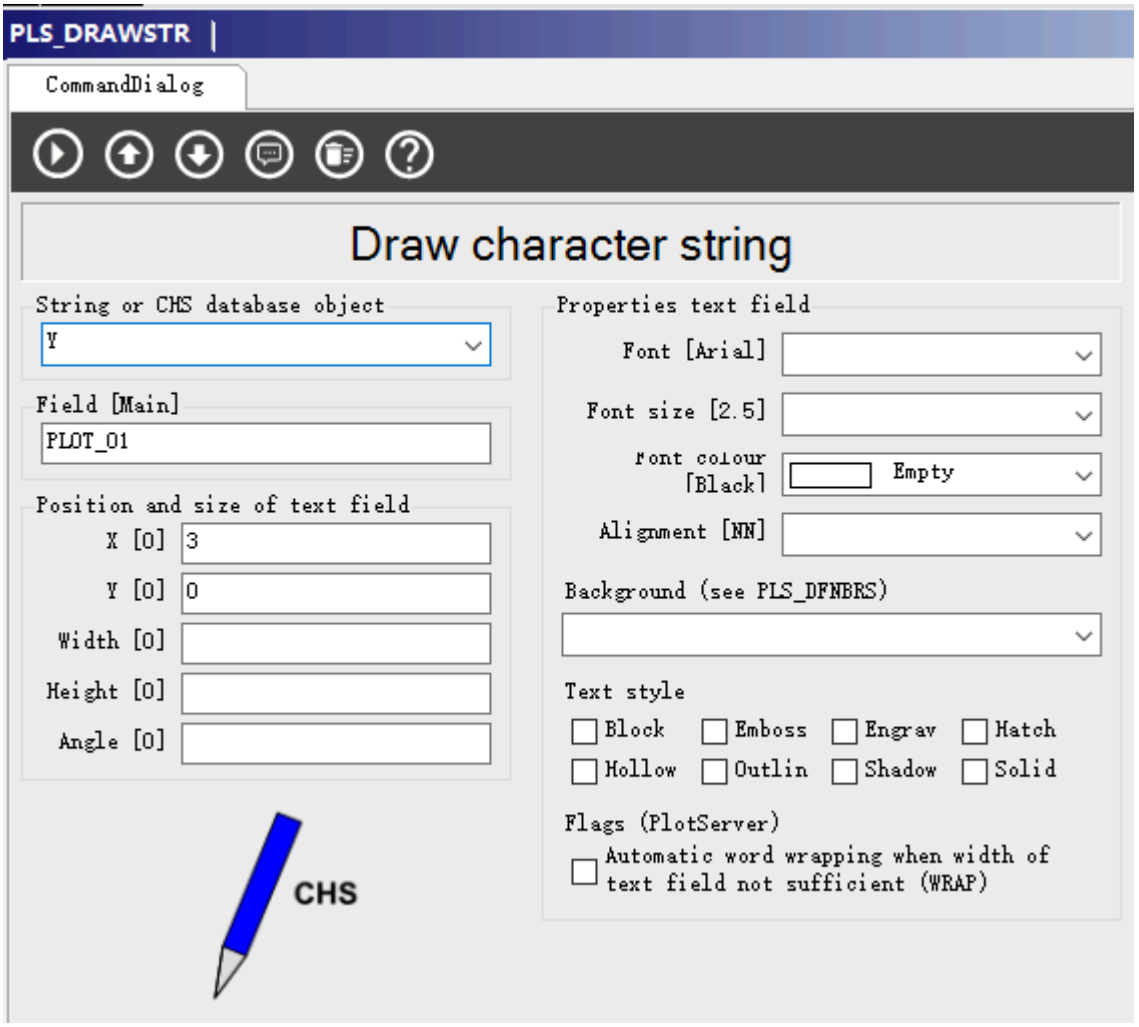


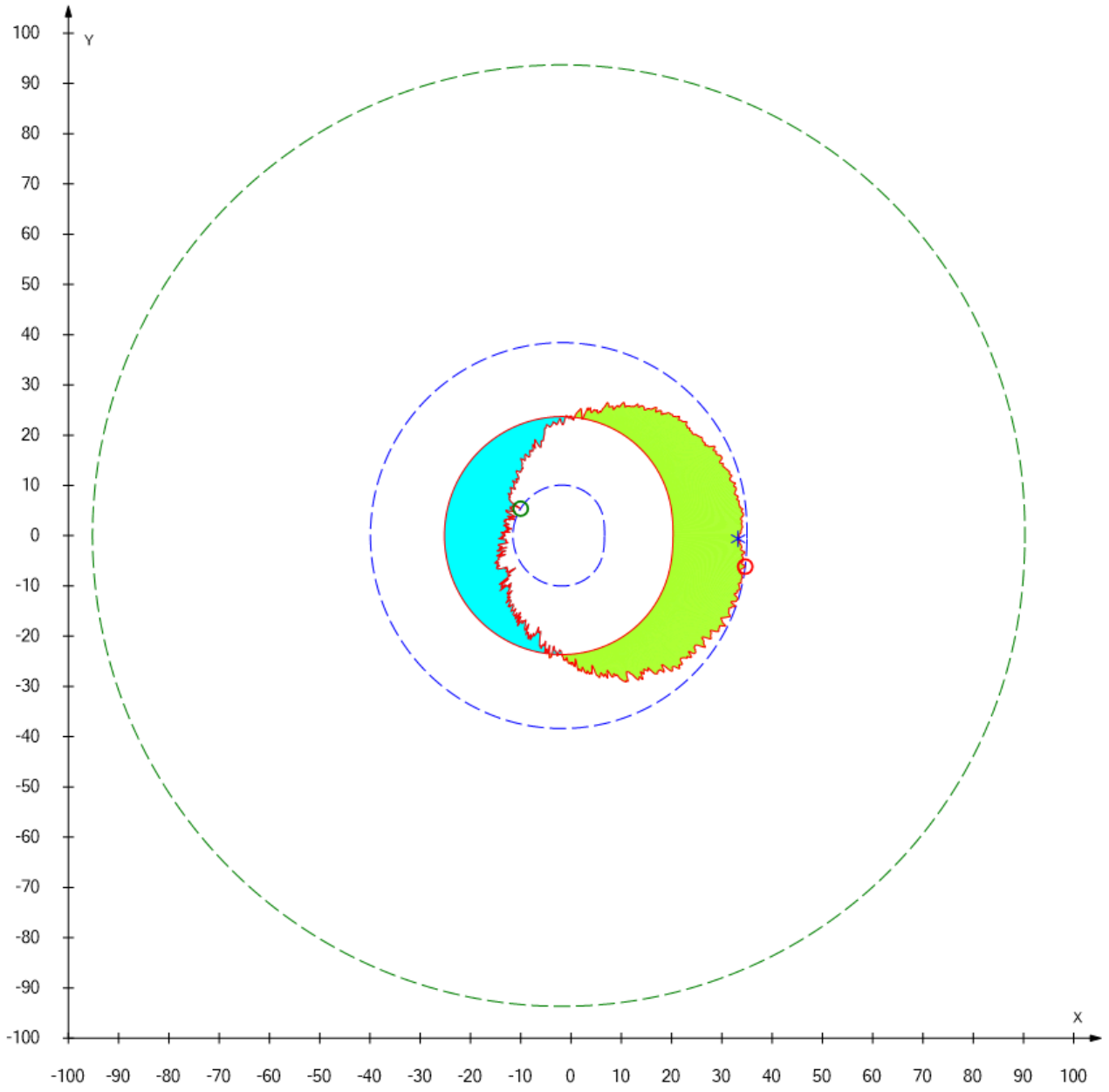
最后可以使用**PLS_DRAWSYM**和**PLS_DRAWSTR**补上坐标系箭头符号与文字标识。符号的种类、颜色，文字的字体、大小、颜色都可以进行在指令中进行选择。此案例中我全部使用了默认设置，没有进行调整，所以显示为空白。

画出箭头和文字 复制代码

```
PLS_DRAWSYM     (FRA=PLOT_01, SCO=RevArw, PHI=90, FIL=Y)
PLS_DRAWSYM     (FRA=PLOT_01, X  =180, Y  =180, SCO=RevArw, FIL=Y)
PLS_DRAWSTR     (FRA=PLOT_01, STR=Y, X  =3, Y  =0)
PLS_DRAWSTR     (FRA=PLOT_01, STR=X, X  =180, Y  =175)
```


PLS_DRAWSYM
CommandDialog
Draw marker
Field [Main]
PLOT_01
Position and size
X [0]
Y [0]
Size [5]
Angle [0]
90
Marker
Form [Crt]
RevArw
Pen [P1: (black) (solid) (0.075) (center)]
Fill marker [N]
Colour (see PLS_DFNBRS)


PLS_DRAWSTR
CommandDialog
Draw character string
String or CHS database object
Y
Field [Main]
PLOT_01
Position and size of text field
X [0]
3
Y [0]
0
Width [0]
Height [0]
Angle [0]
CHS
Properties text field
Font [Arial]
Font size [2.5]
Font colour [Black]
Empty
Alignment [NN]
Background (see PLS_DFNBRS)
Text style
Block
Emboss
Engrav
Hatch
Hollow
Outlin
Shadow
Solid
Flags (PlotServer)
Automatic word wrapping when width of text field not sufficient (WRAP)



与数据轮廓相关的作图内容讲解到这边已经全部完毕了。这意味着最复杂的部分已经结束了。剩下的只剩一些报告标题文字的绘制以及自定义画笔的设置。

画出报告的标题

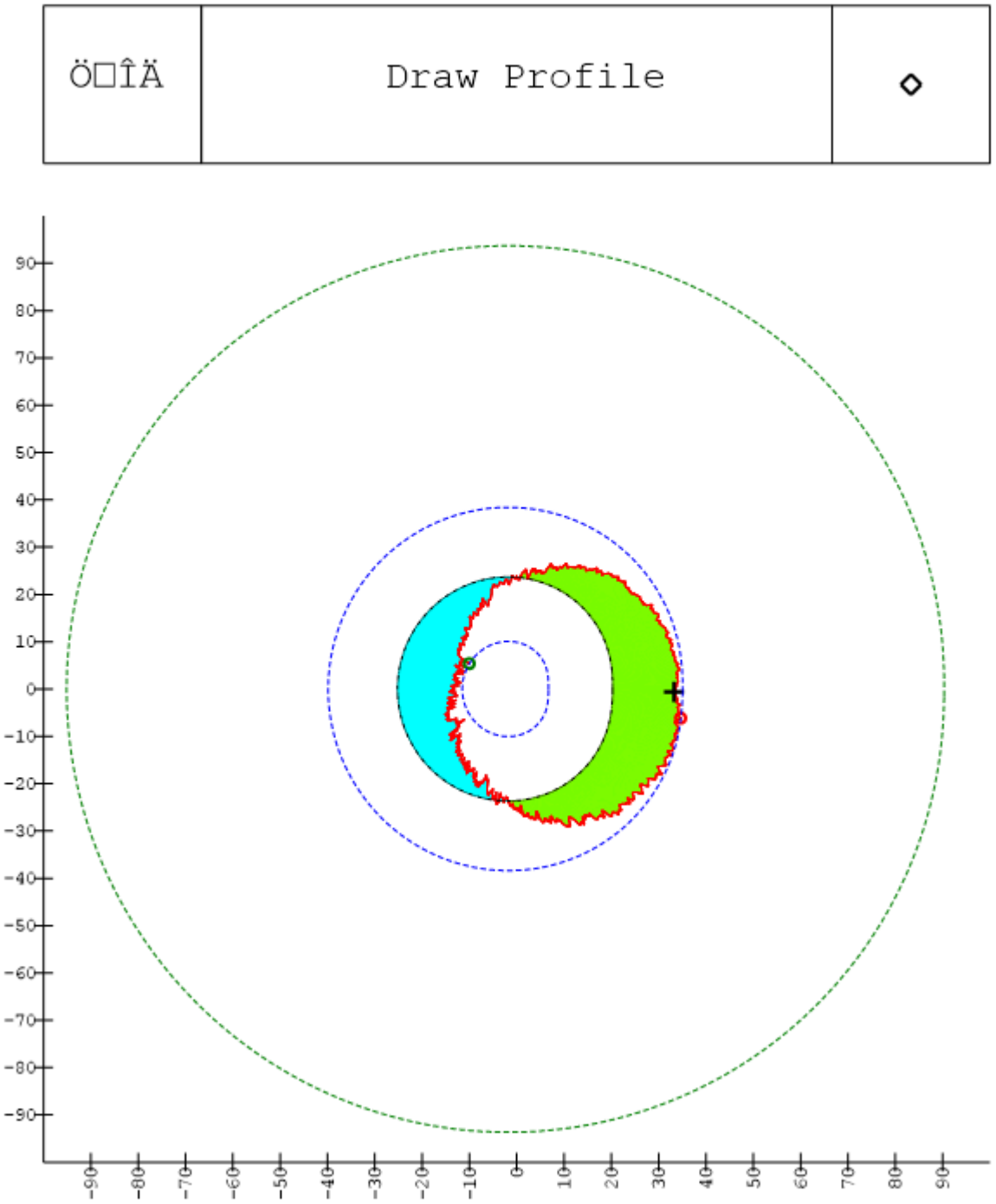
DRWPLY系列：

这一部分是DRWPLY系列作图中最薄弱的功能。我在下方案例中定义了新的三个PDF区域，并分别让其输出中文字符、英文字符和符号。从结果来看，第一个方框中的中文直接以乱码

形式出现，并且此系列不支持输出img图片。所以不推荐使用此方法制作标题、图例说明等元素。

```
DFNPDF          (NAM=PDF02, XOF=0, YOF=190, XSZ=30, YSZ=30)
DFNPDF          (NAM=PDF03, XOF=30, YOF=190, XSZ=120, YSZ=30)
DFNPDF          (NAM=PDF04, XOF=150, YOF=190, XSZ=30, YSZ=30)

USEPDF          (NAM=PDF02)
DRWSTR          (X  =5, Y  =15, STR=中文)
USEPDF          (NAM=PDF03)
DRWSTR          (X  =35, Y  =15, STR=Draw Profile)
USEPDF          (NAM=PDF04)
DRWPNT          (X  =15, Y  =15, MRK=5)
```

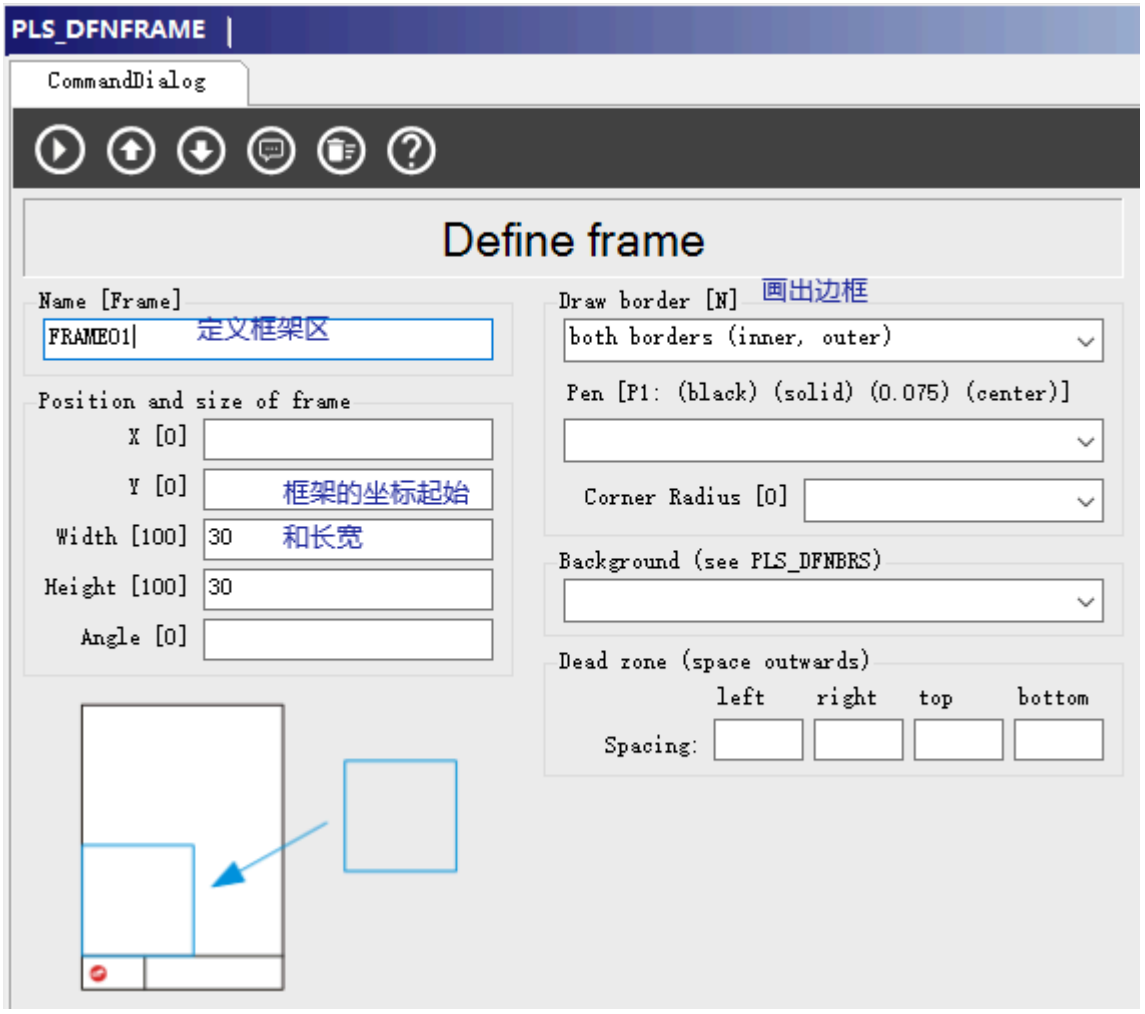


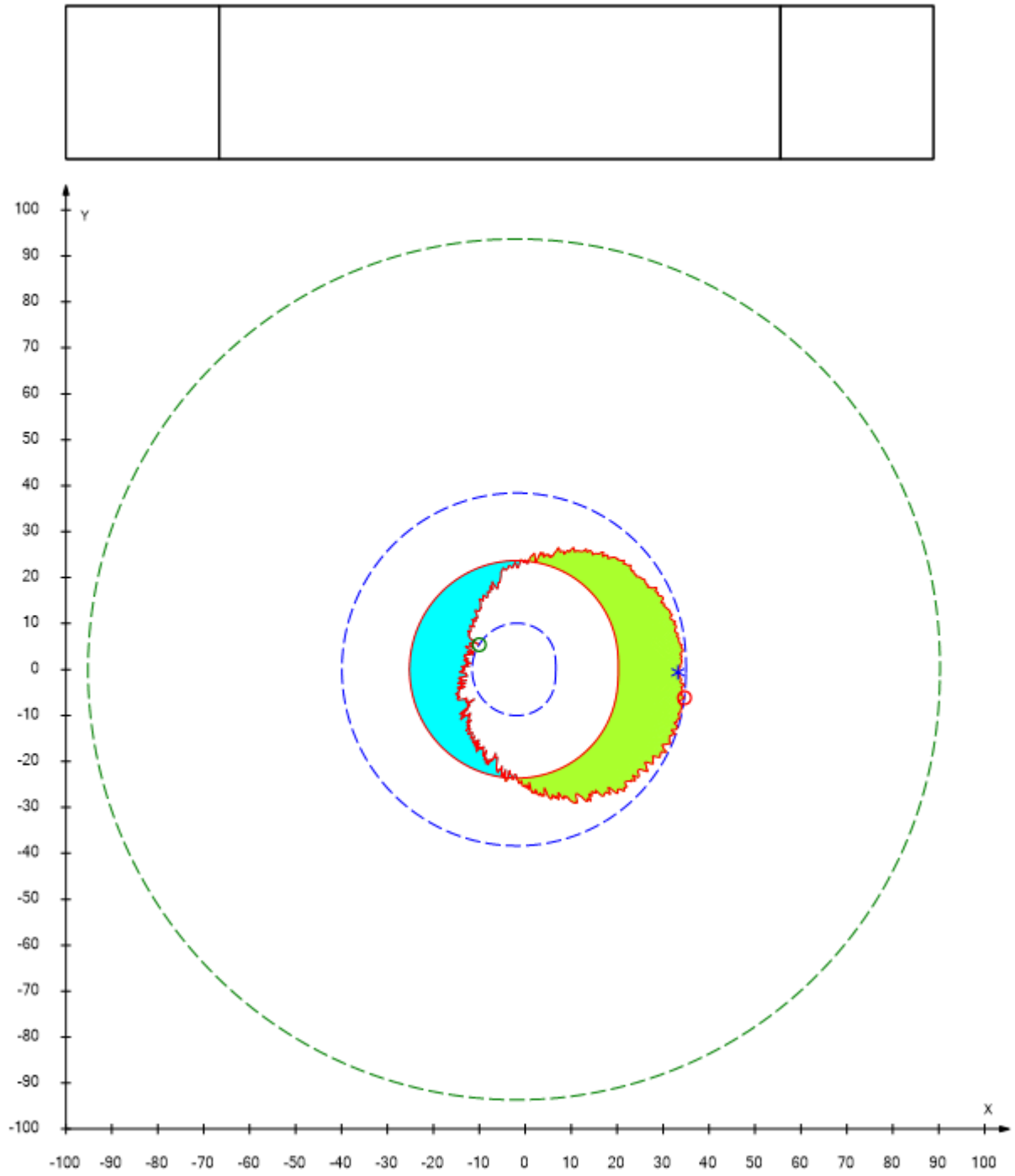
PLS_ADDCRVL系列：

可以使用指令**PLS_DFNFRAME**定义一个框架，使用界面的逻辑和**PLS_DFNPLOFLD**完全一致。

定义标题框架 复制代码

```
PLS_DFNFRAME     (NAM=FRAME01, W  =30, H  =30, DRW=A)
PLS_DFNFRAME     (NAM=FRAME02, X  =30, Y  =0, W  =110, H  =30, DRW=A)
PLS_DFNFRAME     (NAM=FRAME03, X  =140, Y  =0, W  =30, H  =30, DRW=A)
```


PLS_DFNFRAME
CommandDialog
Define frame
Name [Frame]
FRAME01
定义框架区
Position and size of frame
X [0]
Y [0]
框架的坐标起始
Width [100]
30
和长宽
Height [100]
30
Angle [0]
Draw border [N]
画出边框
both borders (inner, outer)
Pen [P1: (black) (solid) (0.075) (center)]
Corner Radius [0]
Background (see PLS_DFNBRS)
Dead zone (space outwards)
left
right
top
bottom
Spacing:



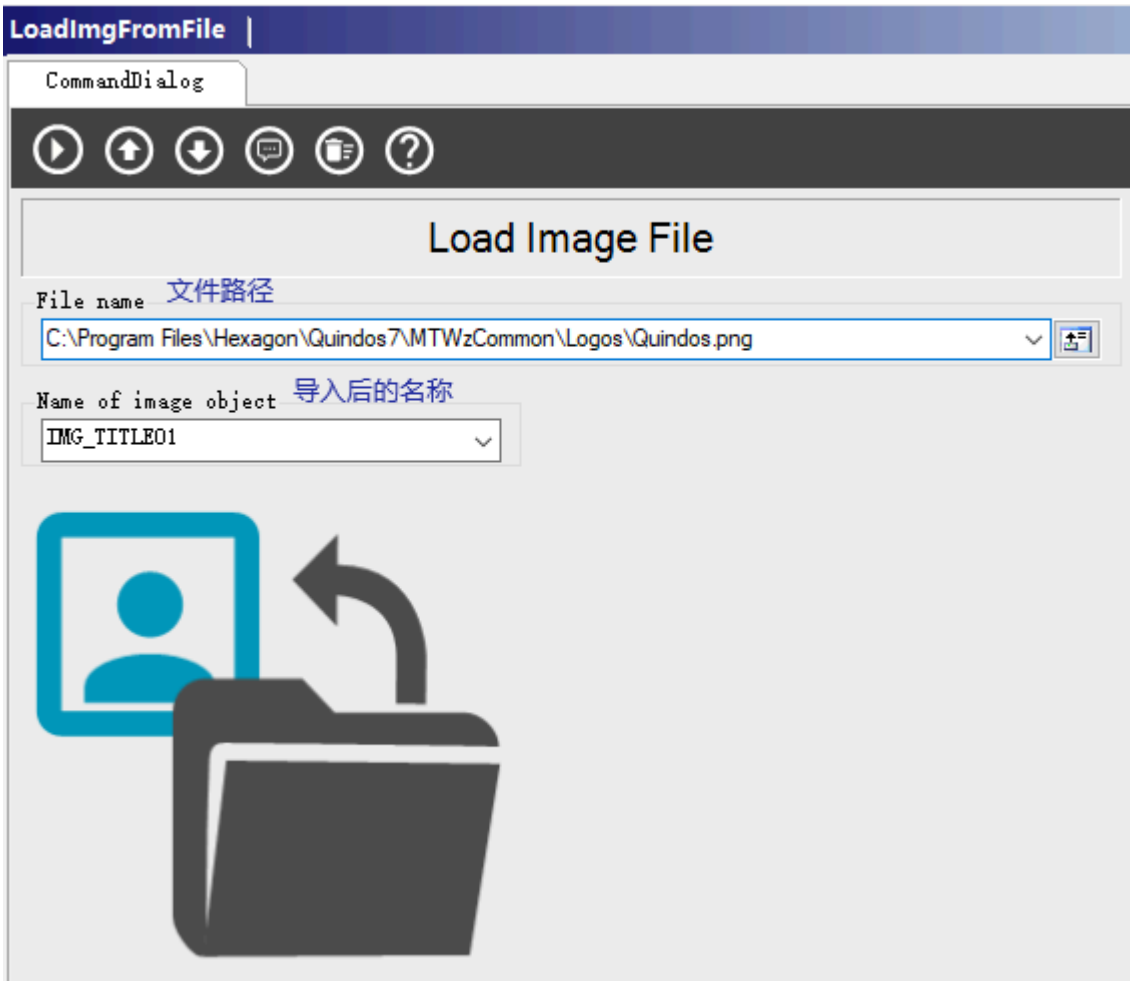
在框架中填充图片：

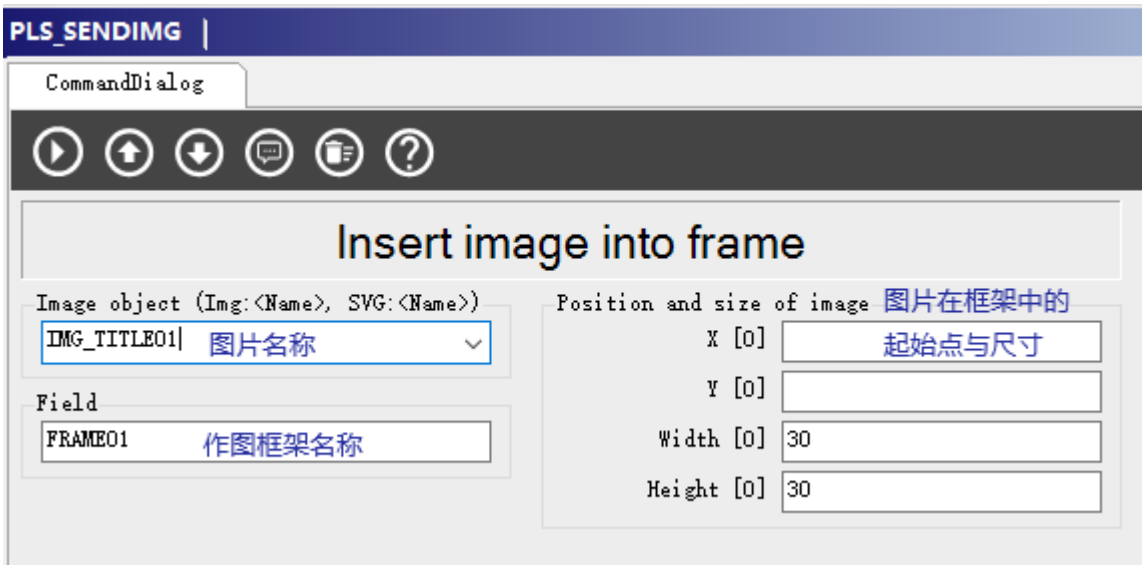
使用指令**LoadImgFromFile**可以将IMG图片导入程序；使用指令**PLS_SENDIMG**可以将图片输出到报告里。

复制代码

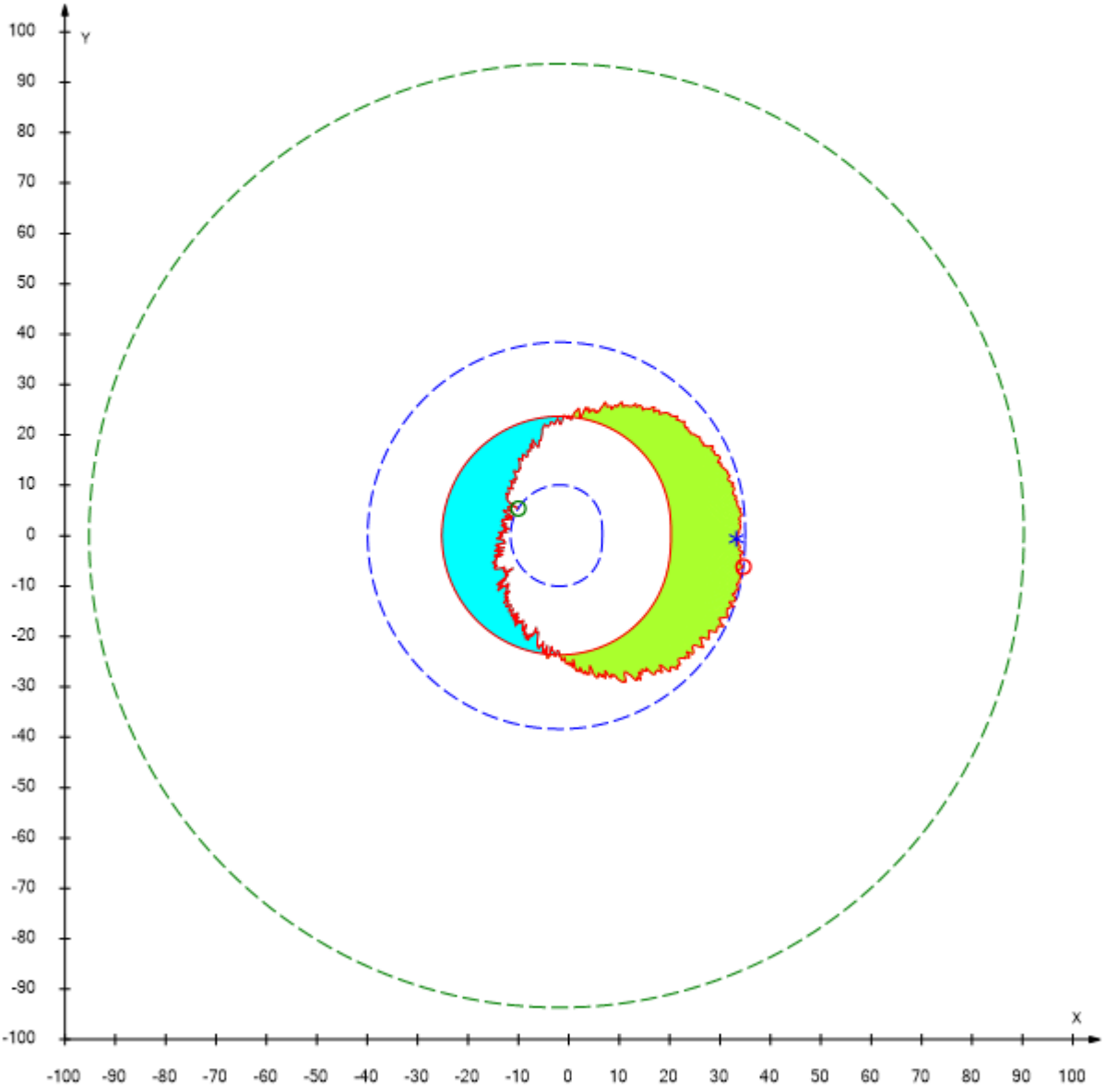
```
LoadImgFromFile   (NAM=IMG_TITLE01, FNA=C:\Program
Files\Hexagon\Quindos7\MTWzCommon\Logos\Quindos.png)
```

```
LoadImgFromFile   (NAM=IMG_TITLE02, FNA=C:\Program
Files\Hexagon\Quindos7\MTWzCommon\Logos\Leitz.jpg)
PLS_SENDIMG     (IMG=IMG_TITLE01, FRA=FRAME01, W  =30, H  =30)
PLS_SENDIMG     (IMG=IMG_TITLE02, FRA=FRAME03, X  =5, Y  =5, W  =20, H
```





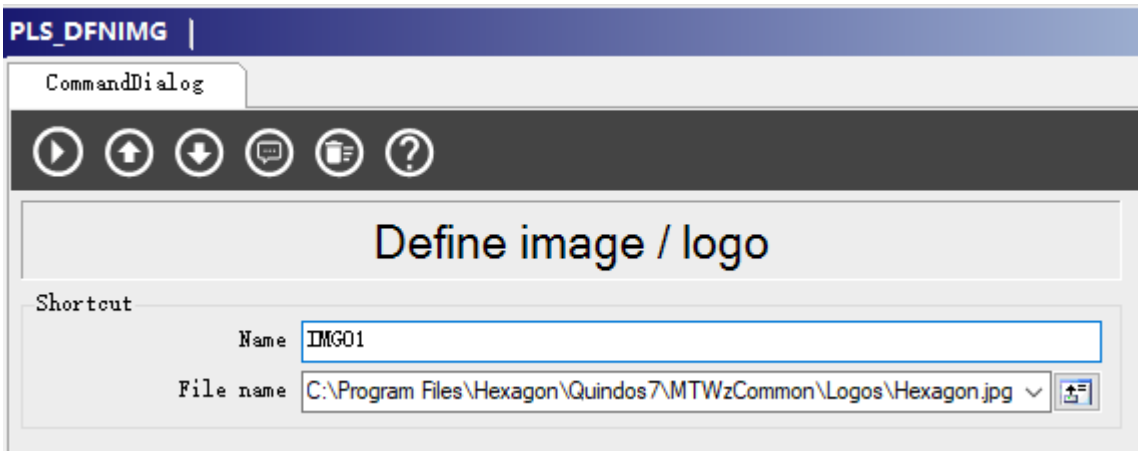
可以根据需要调整图片在框中的位置和大小。执行命令得到如下效果：

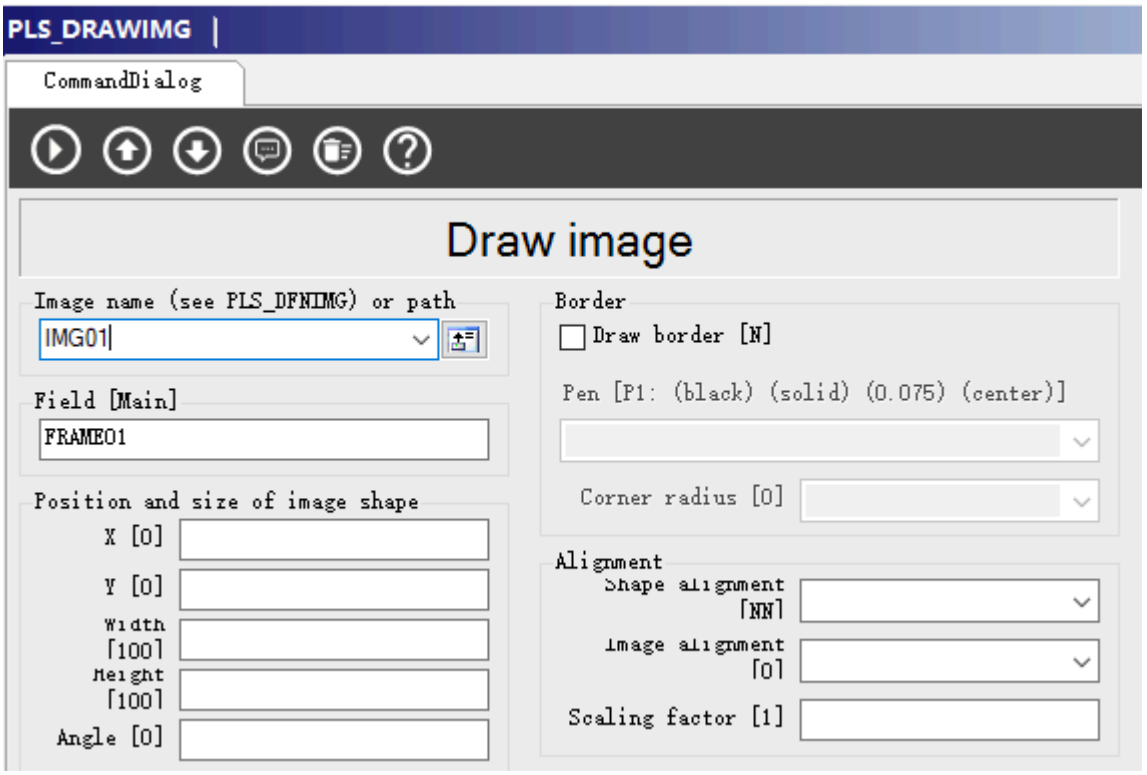


也可以直接使用**PLS_DRAWIMG**指令在不导入不定义的前提下，直接用路径连接图片。但是如果图片保存的位置发生变更，可能会失效。没有直接将图片导入保存在LDB的方式稳定。

需要注意的是，**PLS_DRAWING**如果需要事先定义图片到程序，不能使用**LoadImgFromFile**，而是需要使用新指令**PLS_DFNIMG**将图片保存在绘图服务器Plotserver。

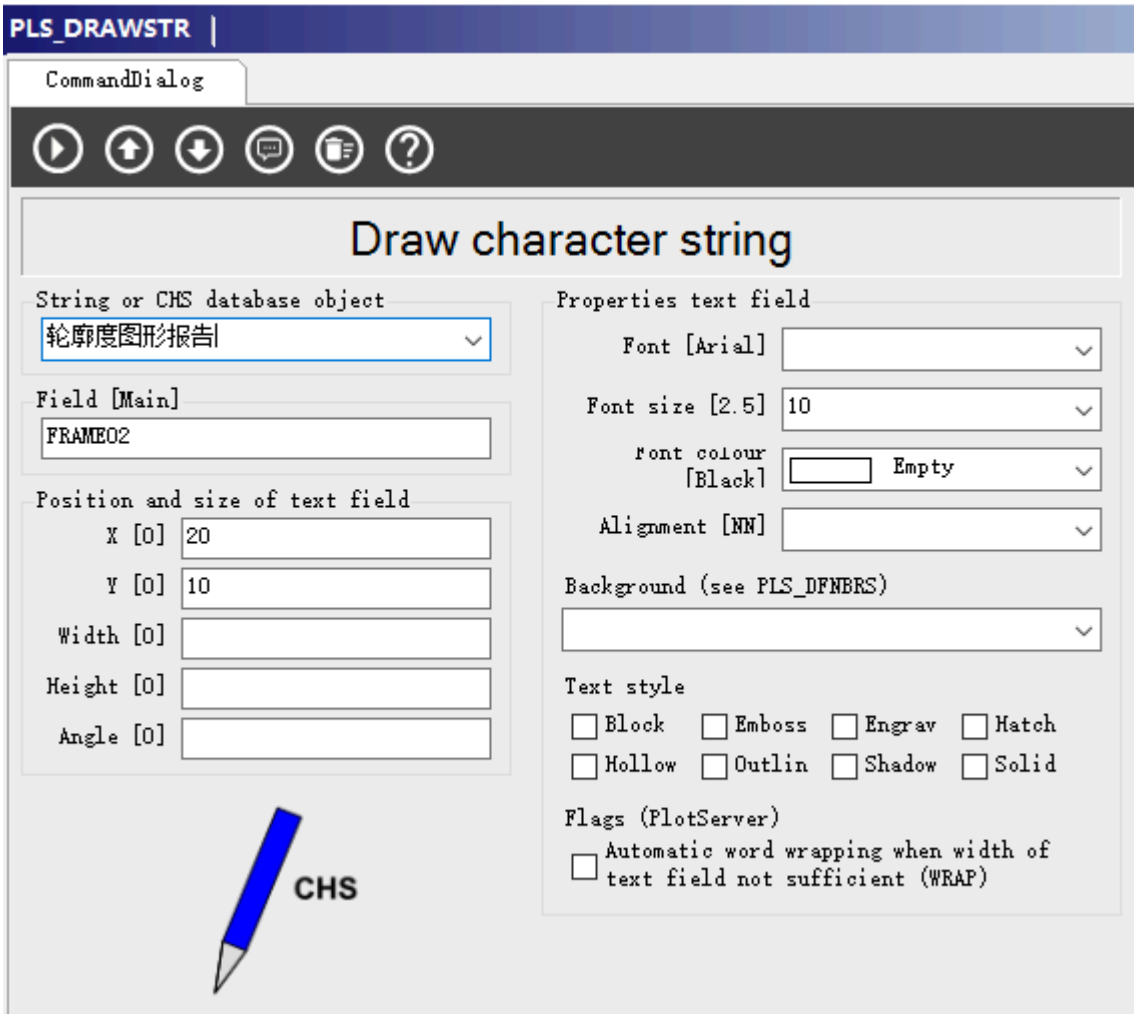
```
PLS_DFNIMG      (NAM=IMG01, FIL=C:\Program
Files\Hexagon\Quindos7\MTWzCommon\Logos\Hexagon.jpg)
PLS_DRAWIMG     (FRA=FRAME01, IMG=IMG01)
```

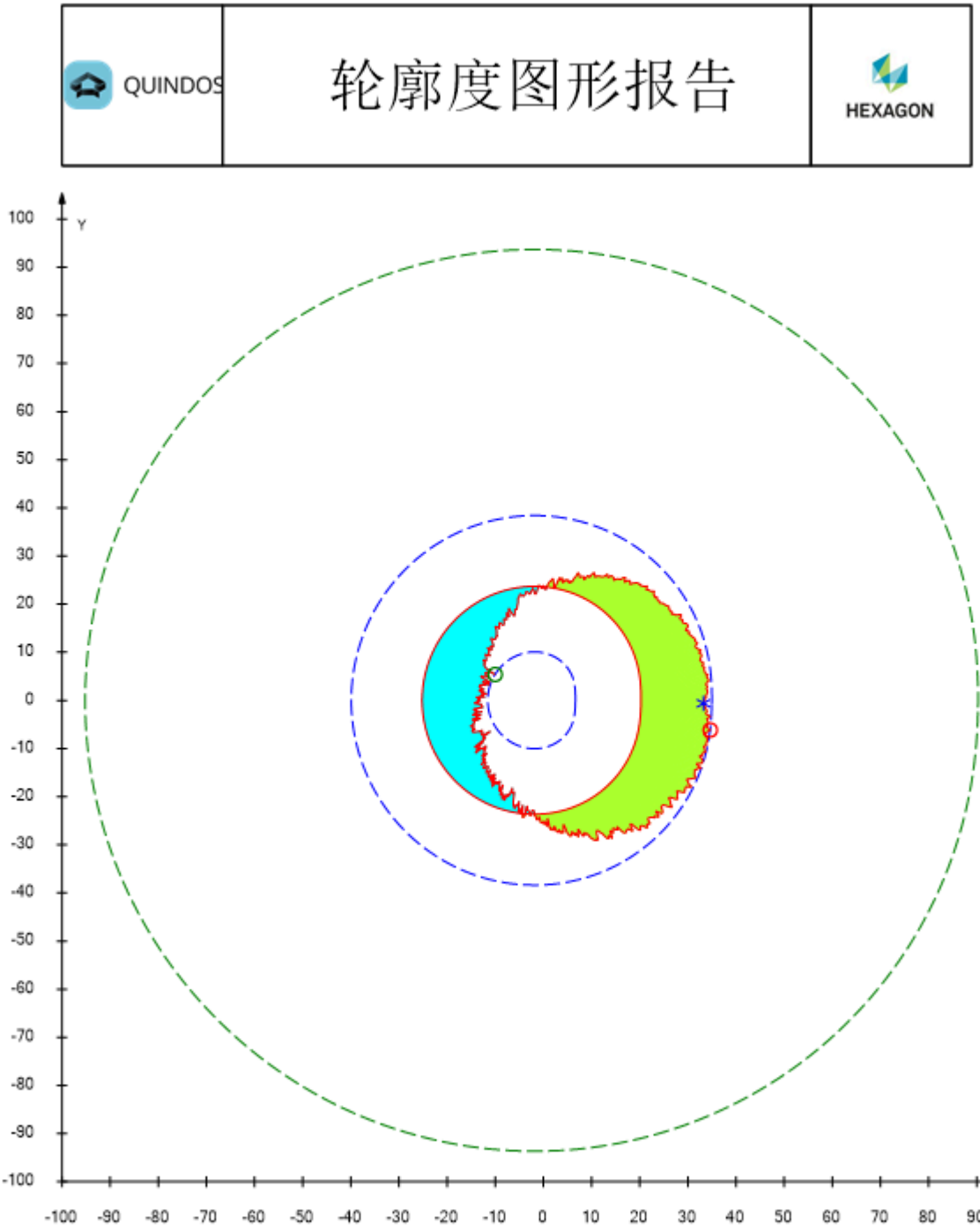




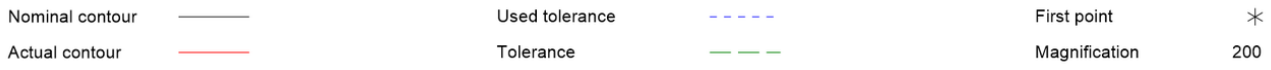
最后，在中间的框架中输入文字，使用**PLS_DRAWSTR**指令，在对应的框架中直接生成中文字符即可。根据需要位置可以调整。

```
PLS_DRAWSTR     (FRA=FRAME02, STR=轮廓度图形报告, X  =20, Y  =10, FSZ=10)
```


PLS_DRAWSTR
CommandDialog
Draw character string
String or CHS database object
轮廓度图形报告
Field [Main]
FRAME02
Position and size of text field
X [0]
20
Y [0]
10
Width [0]
Height [0]
Angle [0]
CHS
Properties text field
Font [Arial]
Font size [2.5]
10
Font colour [Black]
Empty
Alignment [NN]
Background (see PLS_DFNBRS)
Text style
Block
Emboss
Engrav
Hatch
Hollow
Outlin
Shadow
Solid
Flags (PlotServer)
Automatic word wrapping when width of text field not sufficient (WRAP)



至此，报告只剩下一个最底部图例的绘制了。它由一个框架，文字，不同颜色和类型的线段以及符号组成。用已经讲过的所有命令，试试把它画出来。



自定义画笔：

在作图的过程中，总有对预设画笔的颜色、粗细不满意的时候。此时可以自定义一个画笔，来满足作图需求。

DRWPLY系列：

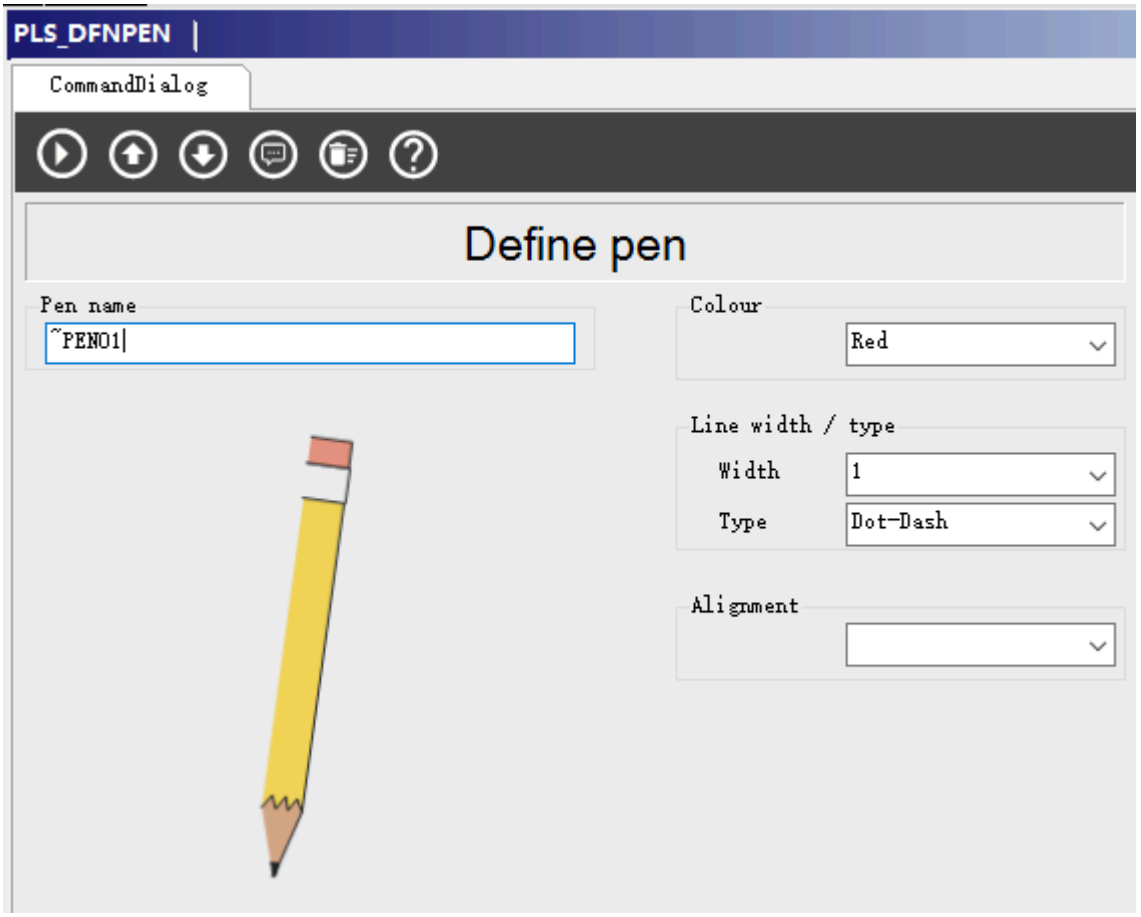
线段类型：使用**SETLTY**指令设定（前面的章节已经介绍过）。

颜色：在使用**DRWPLY**指令时，Pen color除了在下拉列表选择，还可以输入Quindos内置的总计36种颜色的数字代码。完整表格可以在Quindos帮助文档中搜索【Color table of Quindos pens】。

PLS_ADDCRVL系列：

使用指令PLS_DFNPEN，对颜色、粗细、线段种类进行任意的组合，并保存为一个名字。

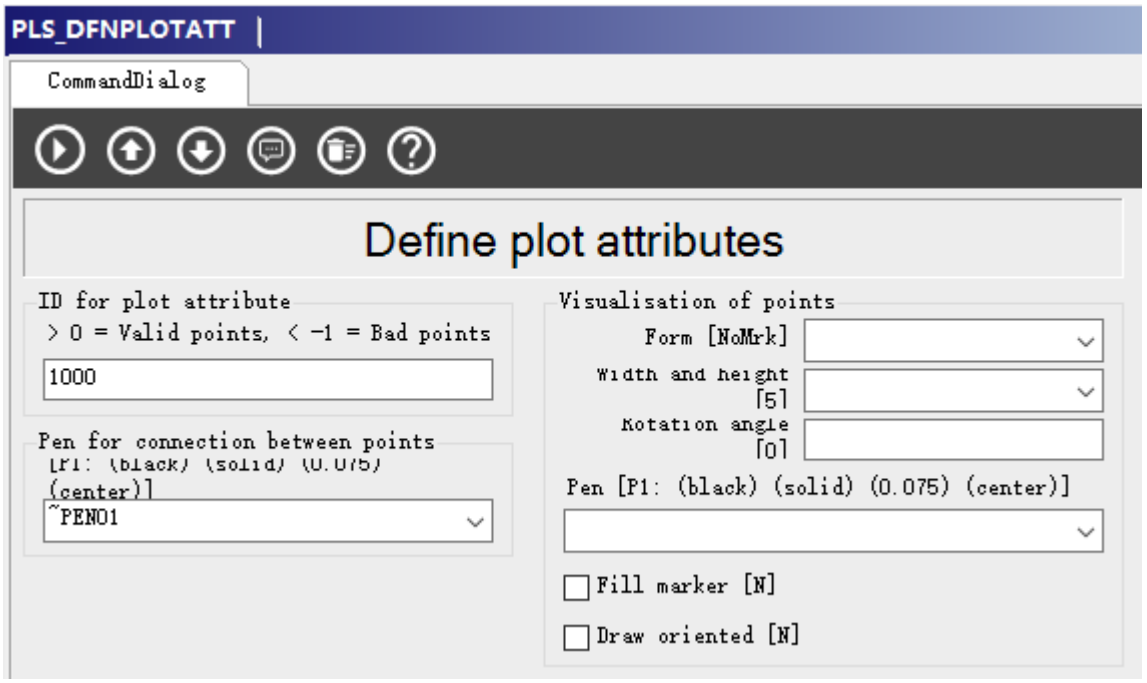
```
PLS_DFNPEN      (NAM=~PEN01, BRS=Red, WDT=1, DST=DAD)
```



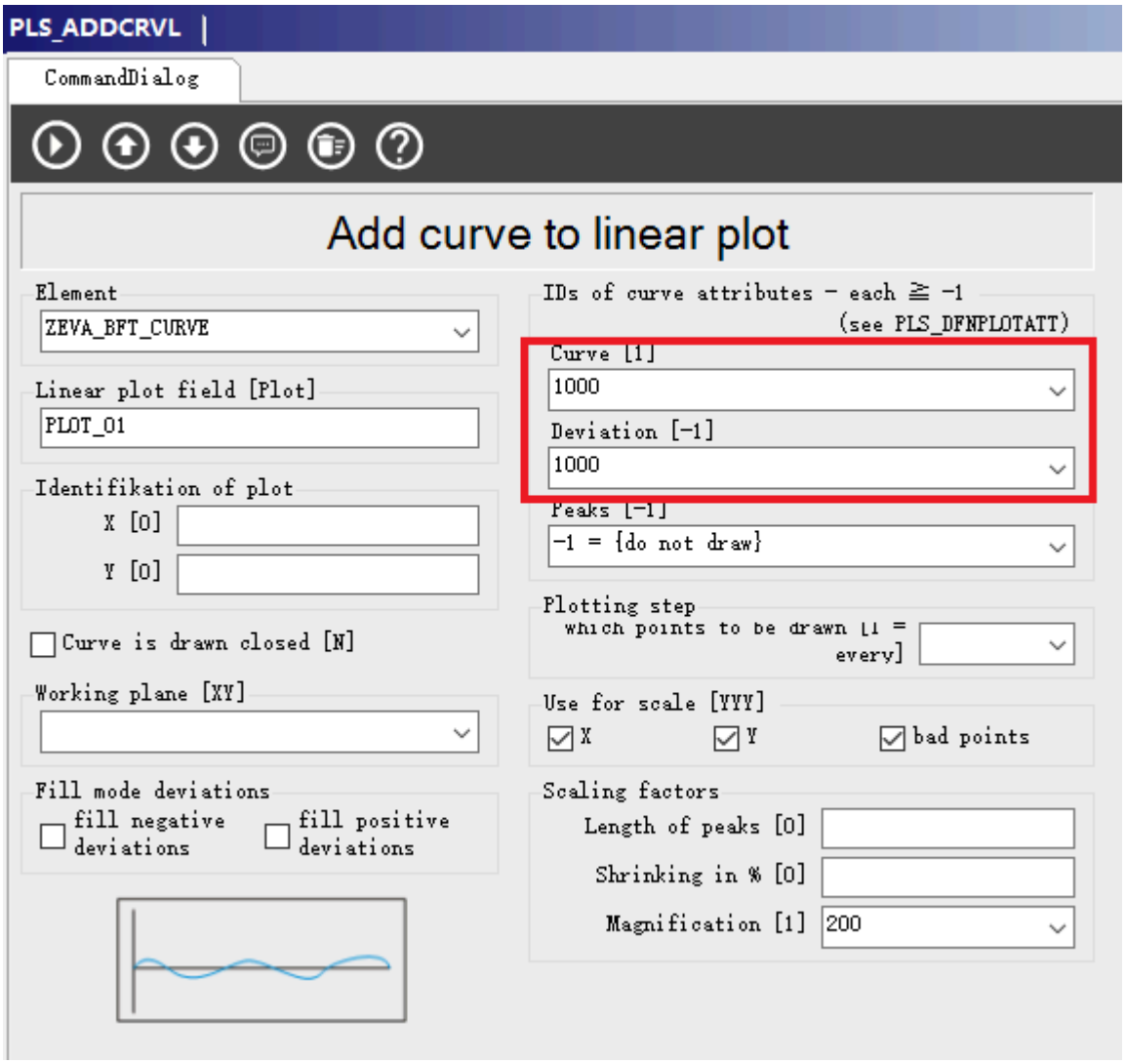
为了使这只笔在**PLS_ADDCRVL**中可以使用，我们还需要使用指令**PLS_DFNPLOTATT**将其定义到作图区域的属性中，并且**必须在PLS_DFNPLOFLD执行之前进行定义。**

定义作图区域属性 复制代码

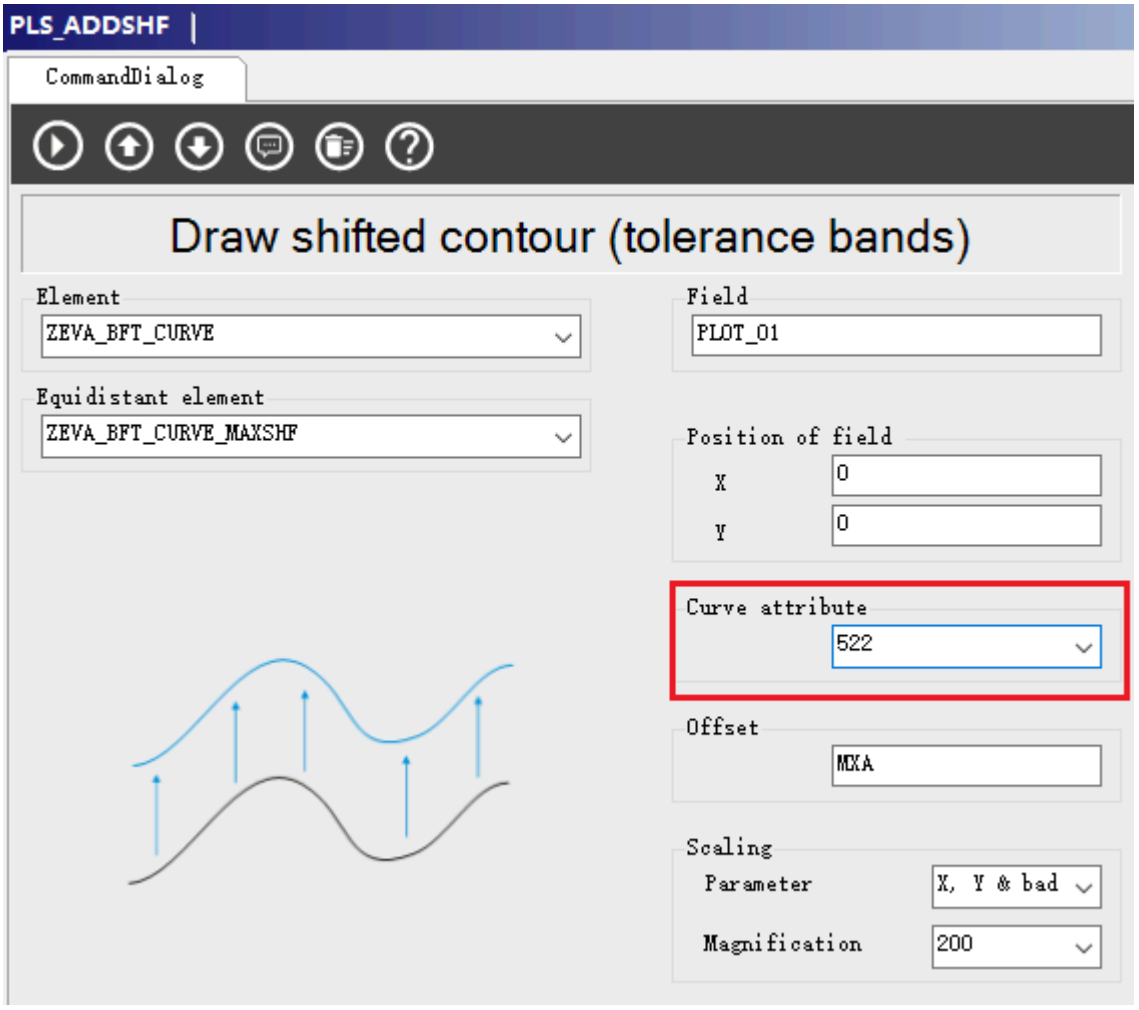
```
PLS_DFNPLOTATT  (PID=1000, PEN=~PEN01)
```



这样在使用**PLS_ADDCRVL**进行绘图时，就可以直接输入属性代码进行绘图。



还有一种快捷方式，可以通过输入组合代码直接在非定义的情况下调用。详细使用方式可以参见Quindos在线帮助文档【Pen - Shortcut】部分。由于需要对照表格查询4种属性【cswa】（颜色、线段类型、粗细、方向属性），在此不展开解释，只要查找表格内容并输入对应的数字即可。（开头不用加P，最后一项不输入为默认在中心）



最后——

通过上述所有的步骤，我们已经可以将一个轮廓度相关的报告画出。当然，还可以添加更多的详细信息，如轮廓度评价值是多少，最大偏差、最小偏差分别是多少，检测时间、人员等相关信息。这些内容都可以通过**【画出最大偏差点、最小偏差点和测量起始点的位置】**中获取到的数据以及**【画出报告的标题】**中提及的方式实现。

但是**目前的作图是不会自动调整放大倍率**。自动的放大倍率要么依赖作图指令中的自动缩放（但对坐标系失去控制），或者通过编写程序添加自动缩放的能力：将坐标系数据、偏差数据和公差带上限值进行收集分析，取其最大的值作为变量，加上逻辑判断选择最佳参数放入

定义缩放的指令中，提高程序的通用性。但此过程需要更多的程序编写能力，不是本指南的主要讨论内容。相信需要这份指南的读者，已经具备了一定程度的编写能力，或是相对应的研究能力，可以根据上述思路自行尝试。

同时还需要申明，所有内容都是个人探索后的理解，并不保证完全正确，仅供参考。

本指南到此结束。