ポストスクリプトを用いた

プロッターサブルーチン群

1995 年 9 月 1 日

佐藤工業（株）中央技術研究所

吉田　望

はじめに

ポストスクリプト言語は Adobe 社によって開発された、印刷用言語である。この言語はテキスト形式で書かれているので、自分で作成することもできる。また、プロッター用の標準的なサブルーチン群である、CALCOMP 仕様とのインターフェイスプログラムも開発されている。

ポストスクリプト言語は、そのままプリンターに送れば作図してくれるが、例えば Windows の様なシステム中のプログラムのイメージとして用いることはできないという欠点がある。この欠点を除くためには、EPS（Encapsulated PostScript）という形式のファイルを作る必要がある。EPS 形式であれば、図としてプログラムで取り込むことができる。しかし、単なる EPS ファイルでは次のような不便な点もある。

①EPS ファイルは、図としてプログラムに取り込むことができるが、通常のプログラムではその内容を変更することはできない。（たとえば Corel Draw のように例外的にできるプログラムもあるが、完全ではない）

②EPS ファイルは、画面イメージを持たないときはテキストファイルであるが、画面イメージを持つと、テキストファイルではなくなる。したがって、テキストファイルのままでは、取り込んだプログラムでどのように見えているか確認できない。

この欠点を除くには、EPS 形式のファイルが、何かの作図プログラムのデータとなるようにテキストファイルを作ればよい。このサブルーチン群では、Illustrator 3.0J と同じ出力形式になるよう EPS 形式のファイルを作成する。したがって、Illustrator で読み込むことによって画面イメージを作ることもできる。また、Corel Draw 5.0J を用いても同じ事ができる。なお、Illustrator は 3.0J と 4.0J は同じ出力形式であるが、5.0J では出力形式が異なっている。また、Corel Draw では、線種が指定したとおり出力されない、文字の形式が異なるなどの違いがあり、完全にはイメージが移らない。

このマニュアルでは、まず、Illustrator に用いられるポストスクリプト言語の特徴と使い方を示し、その後、サブルーチンの説明をする。

ポストスクリプトの詳細は、次の図書を参照のこと。ただし、Illustrator 関係の記事はない。

・PostScript チュートリアル＆クックブック　アスキー出版局
・PostScript レファレンス・マニュアル　アスキー出版局

目　　次

ポストスクリプト言語と Illustrator 言語……1
1　基本的手続き……1
1.1 ヘッダーとフッター……1
1.2 命令の方法……1
2　線の種類……2
3　線とストローク……3
4　グループ化……4
5　円 4
6　文字……4
6.1 基本セット……4
6.2 特殊な作業……5
6.3 行揃え（alignment）……6
6.4 文字関連の関数……6
7　漢字……8
8　色……9
9　日本語仕様……9

作図サブルーチンの構成と使用法……10
1　最低限の使用法……10
2　基本サブルーチン……10
2.1 PLOTS (X, Y, IPEN)……11
2.2 PLOT (X, Y, IPEN)……11
2.3 PLOTL (X, Y, IPEN)……11
2.4 PLOTCV (X1, Y1, X2, Y2, X3, Y3)……12
2.5 NEWPEN (IP)……12
2.6 PLAPHA (CHR)……12
2.7 PLOTAL (ITA, TS)……12
2.8 PLOTSY (IP, X, Y, SIZE, ANG, NCR, CHR)……12
2.9 PLOTKJ (IP, X, Y, SIZE, ANG, NCR, CHR)……13
2.10 ARC1 (X, Y, R, SANG, EANG)……13
2.11 PLOTMK (ITP, XX, YY, RLEN)……13
2.12 GRID (X0, Y0, XD, YD, MX, MY)……13
2.13 PATRN (XX, YY, NP, N0)……13
2.14 AROHD (XS, YS, XE, YE, S, W, KY)……14
2.15 CIRCLE (XX, YY, SANG, EANG, R)……14
2.16 PLOTGR (IP)……14
2.17 SYMBOL (XX, YY, SIZE, CHR, ANGLE, NCH)……14
2.18 FACTOR (FACT)……14
2.19 WHERE (XX, YY, FAC)……15
2.20 SPACE (SP)……15
2.21 PLOTWD (ITYPE, WLIN)……15
3　応用ルーチン……16
3.1 PSYM (XPAGE, YPAGE, SIZE, CHR, ANGLE, NCHAR, IPOSIT)……16
3.2 PNUM (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)……17

3.3 PNUMEX (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)····· 17
3.4 PNUMPW (XPAGE, YPAGE, SIZE, FPN, NDEC, PFPN, IPNUM, ANGLE, IPOSIT) ····· 17
3.5 PNUM10 (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)····· 17
3.6 PSYMSZ (SIZE, CHR, NCHAR, XLEN, YMIN, YMAX)····· 17
3.7 PNLN (SIZE, FPN, NDEC, XLEN)····· 17
3.8 PNLNEX (SIZE, FPN, NDEC, XLEN)····· 17
3.9 PNLNPW (SIZE, FPN, NDEC, PFPN, IPNUM, XLEN, YMAX) ····· 17
3.10 PNLN10 (SIZE, FPN, NDEC, XLEN, YMAX) ····· 17
3.11 PALPHA (CSET)····· 17
3.12 PMARK (XPAGE, YPAGE, SIZE, IMK, ANGLE)····· 18
3.13 SYMOPT (ICON, OPT)····· 18
3.14 KANJI (XX, YY, SIZE, ITEXT, ANG, NCH)····· 18
3.15 KSPACE (SPS)····· 18
3.16 PSMLEN (CHR, NCH, SIZE)····· 18
3.17 AXIS (JOB, XS, XE, ANGLE, X0, Y0, XBAS, NFG, SCALE, SCALES, NBR, NDEC, SIZE, SIZES, SMULT, SPS, NFGS, SLNS, SVAL, PEN) ····· 18
3.18 LGAXIS (JOB, XS, XE, ANGLE, X0, Y0, XBAS, NFG, NBR1, NBR2, SCALE, SIZE, SIZES, SMULT, SPS, NFGS, SLNS, SVAL, PEN)····· 19
3.19 点列を結ぶベジェ曲線····· 19
(1) CURVES (IOPEN, ILINE, IB, SIZE, FNP, IDEC, INBR, ALINE) ····· 19
(2) CURVEO (XMIN, XMAX, YMIN, YMAX)····· 19
(3) CURVE (X, Y) ····· 20
(4) CURVEE ····· 20
(5) 曲線の作図法····· 20
4 プログラムの移行 ·····21
4.1 文字の作図 ·····21
4.2 数字の作図 ·····21
4.3 記号の作図 ·····21
4.4 その他 ·····21

プログラムの引数関係 ·····23

ポストスクリプト言語と Illustrator 言語

１　基本的手続き

1.1 ヘッダーとフッター

Illustrator の EPS 形式では、ヘッダーとして多くの関数を定義し、その後本体があり、最後に少量のフッターが付属するという形式を取る。ヘッダーには更に、いくつかの情報もつけ加えられている。これらは、イラストレータの元の出力から引用してきている。ユーザーにとって影響があるとすれば、次の部分である。

%%For:	この文はユーザー名を表している。内容については、日本語の表現方法参照。
%%Title: (PLT007.ai)	この文は、文書ファイル名を表している。
%%CreationDate: (9/1/95) (0:00 AM)	この文は作成年月日
%%BoundingBox: 0 0 596 843	グラフの作図領域（ポイント数）の x_{min}、y_{min}、x_{max}、y_{max} が示されている。この数字は A4 縦の場合を表している。

これらの文は epsf サブルーチンでは変更されない。Illustrator でデータとして読み込むときには Illustrator の方で適当に処理してくれるので、気にする必要はない。また、多くのプログラムはこの部分は無視しているようである。場合によっては修正する必要があるかもしれない。

1.2 命令の方法

Illustrator では、ヘッダー部に多くの関数を定義している。したがって、本文はほとんどがその関数を使って書かれている。これらは全部が解読できているわけではないが、通常の作図に必要なものはだいたい分かっている。次項以降でその内容と、プロッターサブルーチンでの取り扱いを示す。

基本的には、次のような書き方がされている。

（引数）　（命令）　（引数）　　（最後の引数が使われる関数はまれ）

命令は関数形の定義であり、通常一文字または二文字のアルファベットである。引数は、具体的な形を指定する数字や文字からなる。それぞれの間は一つ以上のブランクで区別する。

例　　5 10 m　　(5,10)の点に移動。

　　3 6 10 9 1 100 c　　ベジェ曲線を書く

２　線の種類

下表に主要なものを示す。なお、引数の説明で#1 というのは一番目の、#n は n 番目の引数を表している。長さ等の単位は原則としてポイント数。epsf 言語ではこれを変更することも可能であるが、epsf サブルーチンではそのまま用いている。また、数値は例には整数が挙げられていても、実際には実数で構わない。この場合、単精度の数字とする必要があるように思われる？。FORTRAN の E 形式の数字は読めない。

作業	関数形	使用例	引数の意味
線の太さ	w	1 w	#1 は線の幅。
線の種類	[]0 d	[5 5]0 d	#1, #2, ...は実線部と空白部の長さで、偶数ペアの数字列となる。これを[]で括っている。その次の数字は前の括弧で指定させたサイクルのどこから書き始めるかを指定するもので、通常 0（すなわち、左端の実線部から書き始める）である。 実線は []0 d となる。
線の濃さ	G	0 G	#1 は 0（黒 100%）～1（白 100%）までの数字。この変更が行われる時には、 0 R という文が直前に現れることが多い。フラグのチェックをしているようで、明らかに不要な時もあるが、常につけておけば間違いない。
塗り潰しの濃さ	g	0 g	#1 は 0（クロ 100%）～1（白 100%）までの数字。この変更が行われる時には、 0 O という文が直前に現れることが多い。フラグのチェックをしているようで、明らかに不要な時もあるが、常につけておけば間違いない。
線端の形状	J	0 J	#1 は形状を表す数値で、0、1、2 のいずれか 
角の形状	j	0 j	#1 は形状を表す数値で、0、1、2 のいずれか。0 のとき角はとがっている（マイター接続）。1（ラウンド接続：線幅と同じ直径の円で接続）、2（ベベル接続：先端は段切り型で終端）と選べる。マイターリミット参照
曲線平坦率	i	10 i	#1 は平坦率を表す。印刷では、通常曲線を直線の集合として出力する。この際の許容値。
角の比率	M	10 M	#1 はマイターリミット。マイター接続では平行に近い線の時見え方が不自然なので、マイターリミット以下の時

			は自動的にベベル接続に変更する。角の内側と外側の長さ/線幅を指定し、初期は 10。これは約 11 度に相当する。

３　線とストローク

線を書くためには、まずペンを始点に移動する必要がある。この命令は

x　y　m　（例：10　300　m）

である。ここで、x、y は座標値。

次に、線を書く命令を行う。Illustrator で使えるのは、直線とベジェ曲線である。

①直線

x　y　l　（例：20　100 l）

ここで、ここで、x、y は終点の座標値。

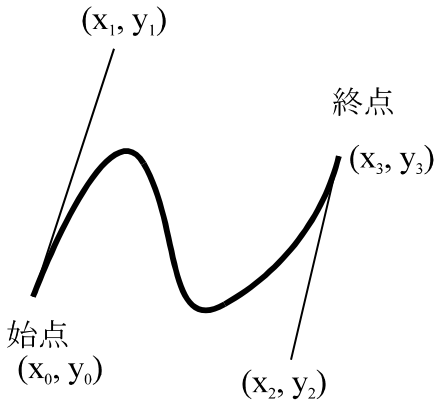
②ベジェ曲線

x1　y1　x2　y2　x3　y3　c　（例：196. 305. 290. 291. 351.45 334.52 c）

ベジェ曲線は、x、y それぞれの方向に 3 次式で表される曲線である。すなわち、

$$x = a_1 s^3 + a_2 s^2 + a_3 s + a_4$$
$$y = b_1 s^3 + b_2 s^2 + b_3 s + b_4$$

今、ペンは始点にあることが条件であるので、ここで、s の始点を現在のペン位置とすれば、a4=b4=0。したがって、6 つの独立な変数を与えれば曲線が決まる。この独立な変数が x1～y3 である。これらは二つの腕の位置および終点の位置に対応している。



今、s=s_1～s_2 までの区間でという、より一般的なケースを考える。x 方向についての係数は次のように表される。ここで、s は距離なので、

$$s_2 - s_1 = \sqrt{(x_3 - x_0)^2 + (y_3 - y_0)^2}$$

これを用いると、

$$A = a_1 (s_2 - s_1)^3$$
$$B = (3a_1 s_1 + a_2)(s_1 - s_2)^2$$
$$C = (3a_1 s_1^2 + a_2 s_1 + a_3)(s_1 - s_2)$$
$$D = a_1 s_1^3 + a_2 s_1^2 + a_3 s_1 + d_1$$
$$x_1 = D + C/3$$
$$x_2 = x_1 + (C + B)/3$$
$$x_3 = D + C + B + A$$

y 方向についても同様である。

③ストローク

①、②の命令は、線を書きなさいという準備をするだけ（バッファにためるだけ）。実際には、この節の命令により図を書く。余り多くをバッファにためるとエラーの原因となるので、適当にこの節の命令を挟む必要がある。

S　または　s　線だけ

F　または　f　塗りつぶし

B　または　b　線と塗りつぶしの両方

大文字と小文字の使い分けは、一般に大文字を使う。小文字を使うのは、始点と終点の座標が同じ（閉曲面）を閉曲面と指定するときのみ。この場合、大文字にすると先端は閉じないで、二つの直線が同じ点で集まっているだけというようになる。したがって、小文字を使うときは最初と最後の座標が同じになっている必要がある。それ以外の場合はエラーで落ちる可能性がある。

４　グループ化

「u」で始まり、「U」で終わる行の間の部分がグループ化される。幾重に重なっていてもよい。

５　円

Illustrator ではベジェ曲線を用いているので、完全な円は書けない（ポストスクリプト言語には円を書く命令があるが、Illustrator ではこれを用いていない）。三次式で近似している。元の Illustrator では完全な円しかないが、同じ手法を用いて円弧も書けるようにしているのが epsf サブルーチン ARC1 である。

illustrator では 4 つの 1/4 円弧により円を書いている。各 1/4 円弧のベジェ曲線のハンドルの腕の長さは、半径の 0.55228 倍である。

円弧の場合には、円弧の角度が小さくなった分だけ腕の長さも小さくする。また、腕の方向は接線方向とする。

６　文字

6.1 基本セット

基本的には、次の例題に示すようなセットが必要である。このうち、状態変数を変

えるような設定は、一度行えば次に変更するまで指定する必要はない。
例題：

0 To	*1
1 0 0 1 240.9834 373.083 0 Tp	*2
0 Tv	*3
TP	*4
0 g	前述（塗りつぶしが黒）
0 Tr	*5
(This is a) Tx	*6
(¥r) Tx	*7
TO	*8

①「0 To」と「TO」で囲まれた範囲が作図範囲
②2 行目は、次のような内容である。

cc ss -ss cc x y 0 Tp
cc, ss は文字を書く方向の方向余弦
x, y は座標値　（単位系は標準はポイントである）

「0 Tp」はこのまま必要。
③「0 Tv」「TP」「0 Tr」は無条件に必要。
TV は line orientation、TP はテキスト指定の終わり、Tr は render（表現）の始まりを意味する。
0 TV は横書き、1 TV は縦書きである。
4)「(書くべきテキスト) Tx
5)「(¥r) Tx」で実際に文字を書く作業を行う。この行は、線の場合のストロークと同じ意味である。ここで、「¥r」は改行を表している。

6.2 特殊な作業

添字、位置の変更等特殊な作業をしたいときは、*5 と*6 の間に必要な行を入れればよい。

①文字の位置
*6 の行の前に、「5 Ts」のように、数字と Ts を組み合わせた行を入れる。ここで、5 はベースラインから 5 ポイントあがった位置であることを示している。なお、一旦設定すると、ずっと有効である。従って、次に影響を与えないようにするためには、例えば TO の前の行に「0 Ts」を入れるようにする。
②いくつかの文字列を書きたいとき
*6 に相当する行を必要なだけつけ加える。例えば、

(This is a pen) Tx

(. I am young.) Tx

とすれば、印字は　「This is a pen. I am young」となる。

③フォントを変えた文字を書きたいとき

*6 の行も前にフォントを制御する行を挿入する。

例：/_Times-Roman 7 6.1421 -1.9927 Tf

最初はフォントを指定している。ただしフォントを指定するには、これだけでは不十分で、一番最初の部分に用いるフォントを定義する必要がある。現在のサブルーチンで用意しているのは、「Arial」「Times-Roman」「Symbol」である。また、漢字フォントとして「MS-Mincho」を用意している。

2 つ目はフォントのサイズで上の例では 7 ポイントである。

次の二つは、フォントごとの拡大率を表している。フォントごとに値が異なる。

6.3 行揃え（alignment）

*6 の前に次の行を入れる。

例：0 Ta

最初の文字は位置を表しており、0=左、1=中央、2=右を表している。

6.4 文字関連の関数

関数	使用例	意味と引数	
Text Objects			
To	0 To	begin text	bindType To テキストを表すブロックの始まり
TO	-	end text	テキストを表すブロックの終わり
Tp	1 0 0 -1 0 0 0 Tp	begin text path	a b c d tx ty startPt Tp テキストを書く方向と位置の指定
TP		end text path	TP テキストの方向指定終わり
Render and Matrix Operators			
Tr	0 Tr	begin render	render Tr
iTm		internal set text matrix	iTm - (uses _tm as implicit argument)
Tm		set text matrix	a b c d tx ty Tm
Td		translate text matrix	tx ty Td
Te		end render	Te
Ta	0 Ta	set alignment	alignment Ta 0 は左寄せ、1 は中央、2 は右寄せ
Tf		set font name and metrics	fontname size Tf
Tl	0 0 Tl	set leading	leading

			行間と段落前の行間（ポイント）。0 はプログラムが自動的に決める。
Tt	0 Tt	set user tracking	userTracking Tt トラッキング
TW	100 100 200 TW	set word spacing	minSpace optSpace maxSpace TW 単語間の空白
Tw	0 Tw	set computed word spacing	wordSpace Tw
TC	0 0 5 Tc	set character spacing	minSpace optSpace maxSpace TC 文字間の空白
Tc	0 Tc	set computed char spacing	charSpace Tc
Ts		set super/subscripting	offset Ts
Ti		set indentation	firstStartIndent otherStartIndent stopIndent Ti
Tz	100 100 Tz	set line scaling	withStreamScalePercent crossStreamScalePercent Tz 水平、鉛直の文字の縮尺率(%)
TA	0 TA	set pairwise kerning	autoKern TA カーニング（0 の時 off、1 の時 on）
Tq	0 Tq	set hanging quotes	hangingQuotes Tq 0 と 1 のみ可能
Tg		set binding punctuation	bindingPunctuation Tg 禁則処理あり（1）となし（0）
TG	25 TG	set binding space	bindingSpace TG 禁則処理のスペース（%）
Tv	0 Tv	set line orientation	orientation Tv 0 は水平方向（左→右）、1 は鉛直方向（上→下）に字を書く。
TV		set character orientation	orientation TV
Tk		kern	autoKern kernValue Tk
TK		non-printing kern	autoKern kernValue TK
T*		carriage return and line feed	T*
T*-		carriage return and negative line feed	T*-
T-		discretionary hyphen (printed)	T-
T+		discretionary hyphen (not printed)	T+
TR		reset pattern matrix	a b c d tx ty TR
TS		special chars	textString justified TS

7　漢字

漢字も英文字と本質的に同じだが、字の表現方法が異なる。基本的には、一つの文字を次のように表す。

¥nnn¥nnn　　（この二組で文字一つ）

ここで、¥の次の nnn で示したのは数字（と記号）でコード番号を表している。最初のブロックは 210 から始まり、区を表している。次は各区内の番号を表している。

例： 亜唖娃阿

¥210¥237¥210¥240¥210¥241¥210¥242

院陰隠韻吋右宇烏

¥211@¥211A¥211B¥211C¥211D¥211E¥211F¥211G

この例で、上の段は¥の間に数字が 3 文字なので二つで一文字を表している。一方、下の方は¥の間に 4 つの英数字が含まれており、一つで一文字を表している。しかし、これを二つにしても構わない。つまり、「¥211¥100¥211¥101¥211¥102¥211¥103.....」の様にしても構わない。

具体的な書き方は epsf.f を参照すること。

```
      IC = 0
      DO 9200 I = 1, NCH
        JTXT = ITEXT(I)
        IF ( JTXT .LE. 100 ) JTXT = 101
        IC = IC + 1
        CNUM(IC:IC) = '¥'
        IC = IC + 1
        CNUM(IC:IC) = '2'
        I1 = JTXT / 100 + 1
        IF (I1 .GE. 64) CNUM(IC:IC) = '3'
        I2 = MOD(JTXT,100) - 1
        IF ( IFIX(I1/2.+0.1) * 2 .NE. I1) THEN
          I2 = I2 + 95
        ELSE
          IF (I2 .GE. 63) I2 = I2 + 1
        ENDIF
        I1 = I1 / 2
        IC = IC + 1
        CNUM(IC:IC) = CHARCT(17 + I1 / 8)
        IC = IC + 1
        CNUM(IC:IC) = CHARCT(17+MOD(I1,8))
C
        IC = IC + 1
        CNUM(IC:IC) = '¥'
        IC = IC + 1
        CNUM(IC:IC) = CHARCT(18 + I2 / 64)
        IC = IC + 1
        CNUM(IC:IC) = CHARCT(17 + MOD(I2,64) / 8)
        IC = IC + 1
        CNUM(IC:IC) = CHARCT(17 + MOD(I2,8))
 9200 CONTINUE
```

最初の¥

100 位以上が 63 間では最初のコードが 2、それ以上は 3

下 2 桁は 1～94 が入ってくるので、0～93 に修正

偶数の区点には I2 に 95 を加える。

奇数の区点では、63 を境に番号が飛ぶので、ここで補正する。

最初の¥2 または¥3 に続く 2 桁はこのようにして与える。

次の¥に続く 3 文字はこのようにして与える。

ここで、 CHARCT はアスキーコードの 16 以上に対応した記号で、次のように与えられている。

```
    1     ' ', '|', '"', '#', '$', '%', '&', 1H', '(', ')',
    2     '*', '+', ',', '-', '.', '/', '0', '1', '2', '3',
    3     '4', '5', '6', '7', '8', '9', ':', ';', '<', '=',
    4     '>', '?', '@', 'A', 'B', 'C', 'D', 'E', 'F', 'G',
    5     'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q',
    6     'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', ']',
    7     '¥', '!', '^', '_', ' ', 'a', 'b', 'c', 'd', 'e',
    8     'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o',
    9     'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y',
   10     'z', '{', '[', '}', ' ', ' ' /
```

８　色

変数 k が色を指定している。

シアン、マゼンダ、黄色、黒　の比率（0 はなし、1 は 100%）を指定

例　0 0 0 1 k　　黒 100%

変数 K も同じ。

９　日本語仕様

Tg、TG、Tv、TV は Illustrator 3.0J 用の仕様であり、3.0 では使われていない。また、3.0J では文字を指定する際、

　フォント名　数字１　数字２　数字３　Tf

であるが、3.0 では

　フォント名　数字１　Tf

である。ここで、数字１は文字の大きさである。

作図サブルーチンの構成と使用法

作図サブルーチンは、CALCOMP 仕様に近い使い方で、FORTRAN により呼ぶことで、Illustrator 3J のデータを作成する。出力は、epsf ファイルとなっているので、他のプログラムからも読むことができる。ただし、画面イメージは含まれないので、Illustrator 以外のプログラムで読み込んだときには画面を見ることはできない。

Illustrator で読み込んだときには、編集作業が可能である。また、Corel Draw 5J で Illustrator のフィルターを介して読み込んでも編集が可能である。ただし、後者の場合には、破線などの情報が失われることがある。

FORTRAN のコンパイラーによっては「¥」は「¥¥」と書かないと判定しないものがある。本プログラムでもそのようなコンパイラーを想定して書いている。そうでないコンパイラー（例えば MS-Fortran）では、全ての¥¥を「¥」に変換する。

１　最低限の使用法

サブルーチンを使う前に、

```
CALL PLOTS (X, Y, IPEN)
```

を一度だけ呼ぶ必要がある。引数は何でも構わない。

サブルーチンの使用が終わった最後に

```
CALL PLOT(X, Y, 999)
```

を呼ぶ。X、Y は何でも構わない。

FORTRAN からの出力は、ユニット番号 7 に行われる。ファイル名は「plt007.ai」が割り当てられている。

なお、座標値(0, 0)に点が一つ書かれる（実際にはペンを動かしただけなので点ではないが、作図した領域に影響する）。これば、プログラムの都合で書いただけのものであり、通常は不要であるので、作図ソフトを使ったとき削除しておくと便利である。

線の太さは標準的には 1 ポイントである。

特に指定のない限り、長さの単位は cm である。

内部で、　PLOTCM　というブロックコモンを使用しているので、同じ名前をほかで使用してはいけない。

２　基本サブルーチン

基本サブルーチンでは、CALCOMP 使用のサブルーチン群とその他の基本的なサブルーチン群である。ユーザーが直接 CALL するものもあるし、ユーザーが直接 CALL する必要のないものもある。

2.1 PLOTS (X, Y, IPEN)

サブルーチン使用の宣言。使用前に一度だけ CALL する。引数に意味はない。

2.2 PLOT (X, Y, IPEN)

線を描く。

X	x 座標（cm）
Y	y 座標（cm）
IPEN	ペンの動き方。2 はペンダウン（すなわち現在の位置から線を書きながら）、3 はペンアップで(x, y)まで移動する。IPEN が負の場合にはペンの動いた位置が新しい原点となる。IPEN=999 をすべてのプロッターサブルーチン群の使用が終わった最後に一度 CALL する必要がある。

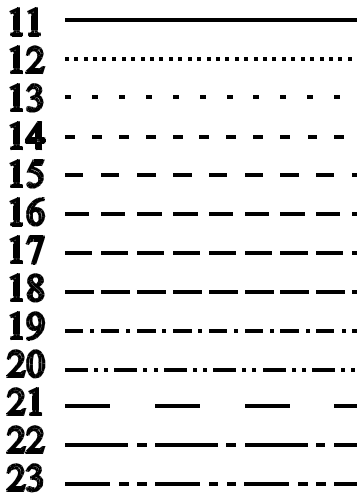
2.3 PLOTL (X, Y, IPEN)

PLOT と同じだが、線の種類などが選べる。

IPEN の絶対値が 2 または 3 のときの意味は PLOT と同じであるので、説明は省略する。

IPEN>10 は線の種類を指定するオプションとなっている。この指定は次々に PLOTL を読んでいる間有効である。11～23 は線の種類を指定している。

IPEN	線種	ピッチ
11	実線	
12	点線	0.1
13	破線	0.3
14	破線	0.3
15	破線	0.4
16	破線	0.4
17	破線	0.4
18	破線	0.4
19	一点鎖線	0.4
20	二点鎖線	0.5
21	破線	1.0
22	一点鎖線	1.0
23	二点鎖線	1.0



一般に、X はピッチ(cm)を表す。0 が入力されると、上に示した標準値が用いられる。

IPEN=12、21、22、23 についてはピッチ以外の部分の指定も可能である。IPEN=12 のとき、Y に 0 以外の値が入力されていれば、点線の実線部の長さ（cm：初期値

0.03cm）を指定したことになる。IPEN=21～23 については、Y に 0 以外の値が入力されていれば、最初の長い線分以外の線分と空白部（両方とも同じ長さ）の距離（cm）を指定したことになる。なお、これらの設定は保存されるので、一度だけ指定すればよい。

IPEN が 100 以上の場合には、100 以下の数字は上記と同じ線の種類を表し、100 以上の部分は 0.1mm 単位の線の太さを表す。すなわち、100 が 0.1mm、200 が 0.02mm. . . という調子である。

2.4 PLOTCV (X1, Y1, X2, Y2, X3, Y3)

ベジェ曲線を描く。現在のペン位置から線を描く。引数の説明は、ベジェ曲線参照のこと。必要な点の数だけ PLOTCV を呼ぶ。

2.5 NEWPEN (IP)

CALCOMP ではペンの種類を変える指令であるが、ここではペン幅を変える指定である。IP=1 のときは 0.5 ポイントの線幅。IP>1 のとき、(IP-1)/100 cm の線幅の線となる。なお、PLOTL を呼ぶとこの指定は失われ、ペン幅 0.5 ポイントとなる。

2.6 PLAPHA (CHR)

英語文字のフォントを変える。初期は「HELVETICA」である。他に、「ROMAN」「SYMBOL」が可能である。CHR に大文字で必要なものを入力する。スペルが間違っていると HELVETICA となる。

Windows に対応するように、HELVETICA を指定すると Arial、ROMAN を指定すると TimesNewRoman、SYMBOL を指定すると Symbol が選ばれるようになっている。

2.7 PLOTAL (ITA, TS)

行揃えと上下のシフトを指定する。先にこれで場所を指定し、次に、PLOTSY や PLOTKJ で文字を書く。一旦指定すれば、次に変更するまで有効。

ITA	行揃え　=0：左、=1：中央、=2：右揃え
TS	文字のベースラインを TS ポイント上に上げる。

2.8 PLOTSY (IP, X, Y, SIZE, ANG, NCR, CHR)

内部サブルーチンで、ユーザーが使うことはない。文字列の一部を描く。

IP=1：通常の文字
IP=2：文字列の最初の部分
IP=3：最後の部分
IP=4：上添字
IP=5：下添字

IP=6：通常

2.9 PLOTKJ (IP, X, Y, SIZE, ANG, NCR, CHR)

日本語を書くためのルーチン。PLOTSY と同じく内部ルーチンである。ユーザーはこれを使わず、KANJI を使う。

2.10 ARC1 (X, Y, R, SANG, EANG)

円または円弧を描く。

X	X 座標
Y	Y 座標
R	半径。R<0 の時、内部を塗りつぶす。
SANG	始点の x 軸に対する角度（度）
ENAG	終点の x 軸に対する角度。円は左回りに描かれる。

2.11 PLOTMK (ITP, XX, YY, RLEN)

シンボルマークを作図する。内部サブルーチンで、ユーザーは PMARK の方を用いるべきである。

ITP	種類 =1：正方形 =2：三角形 =3：逆三角形
XX, YY	中心の座標
RLEN	サイズ。RLEN が負の時は中を塗りつぶす。

2.12 GRID (X0, Y0, XD, YD, MX, MY)

格子を描く。

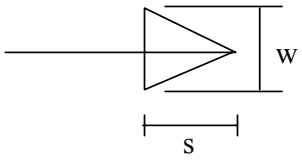
X0, Y0	始点の x, y 座標
XD	x 軸方向の増分
YD	y 軸方向の増分
MX	x 方向の数
MY	y 方向の数

2.13 PATRN (XX, YY, NP, N0)

多角形を描く。

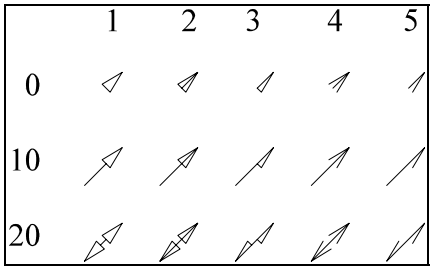
XX, YY	頂点の座標を入れた一次元配列
NP	点の数
N0	塗りつぶしの程度。0（黒 100%）～10（白 100%）までの 10 段階。

2.14 AROHD (XS, YS, XE, YE, S, W, KY)



矢印を描く。

XS, YS	始点の座標
XE, YE	終点の座標
S, W	矢印の長さ（右図参照）
KY	書き方の指示。10 位（0, 1, 2）と 1 位 (1～5) が別の意味を持っている。



2.15 CIRCLE (XX, YY, SANG, EANG, R)

円を描く。ARC1 と同じ機能だが、引数の順番が異なる。

2.16 PLOTGR (IP)

Illustrator 専用の処理を行うためのサブルーチンで、IP により、次のような作業をする。

=1	グループ化開始
=2	グループ化終了
=3	閉曲線のストローク。この場合、図形の始点となった位置までの作図を終わっている必要がある。
=4	開曲線のストローク
=5,6	IP=3,4 と同じ意味だが、内部も塗りつぶす。

2.17 SYMBOL (XX, YY, SIZE, CHR, ANGLE, NCH)

CALCOMP ライブラリーの SYMBOL である。ただし、CALCOMP では NCH<0 の時記号を描くが、このルーチンではこの機能はサポートされていない。この原因は、昔の FORTRAN では CHR のタイプを判定しなかったので、呼ぶ側で適当に文字または数字を入れておけばよかったのが、最近の Fortran では CHR に文字宣言が必要なので、曖昧な形で両方を併存させることができないためである。この機能に関しては別に PMARK を用意しているので、そちらを使えばよい。

XX, YY	始点の座標値
SIZE	文字の高さ。この高さをポイント数にして文字を書くので、実際の高さは少し小さくなるのが普通である。
CHR	文字列
ANGLE	角度（度）
NCH	文字数。CHR の最初の NCH 文字を書く。

2.18 FACTOR (FACT)

CALCOMP ルーチンでは作図比率を変えることになっているが、ここでは何もしな

い。したがって、CALCOMP サブルーチンで FACTOR を使っているものがあれば結果が異なる可能性がある。

2.19 WHERE (XX, YY, FAC)

CALCOMP ルーチンでは現在のペン位置を出力するが、ここでは何もしない。

2.20 SPACE (SP)

CALCOMP ルーチンでは文字の間隔を変えるルーチンであるが、ここでは何もしない。

2.21 PLOTWD (ITYPE, WLIN)

作図時の線の太さを制御する。

ITYPE	PLOT を指定するとき 0 を，PLOTL を指定するとき 1 を入力。
WLIN	ポイント単位で指定した線の太さ。0 が入力されると前の値は変えない。初期値は 0.5 ポイント。

３　応用ルーチン

以下の説明でほとんど共通する引数は次のようなものである。

XGAGE, YPAGE　始点や中心点の x、y 座標

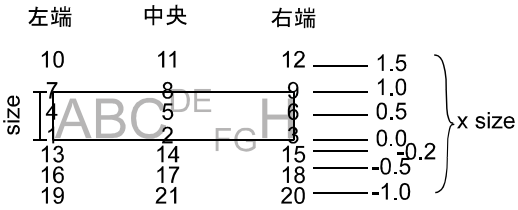
SIZE　文字の高さ

CHR　対象となる文字

FPN, PFPN　対象となる数字

ANGLE　角度（度）

NCHAR　CHR のうち、対象となる文字数。文字列の制御のための文字が入るときは、それも文字数に数える。



IPOSIT　指定した座標と実際に文字が書かれる位置との相対関係。通常は１。1 から 21 間での数字。

NDEC　小数以下の桁数。正であれば、小数以下 IDEC 項まで数字を書き、負であれば、-IDEC 桁目まで書き、その一つ横を四捨五入した正の数字。CALCOMP の NUMBER のケースと異なり、整数はちゃんと桁数だけ書く。

3.1 PSYM (XPAGE, YPAGE, SIZE, CHR, ANGLE, NCHAR, IPOSIT)

CALCOMP の SYMBOL に IPOSIT が付け加わっただけ。ただし、SYMBOL の NCHAR≤0 のケースは使えない。PMARK を利用すること。

CHR は単に文字を入れるだけでなく、若干の制御ができるようになっている。制御は文字$とその次の文字一つの二つでセットになっている。

$A　以降の文字セットを HELVETICA に変える。

$B　以降の文字セットを ROMAN に変える。

$C　以降の文字セットを SYMBOL に変える。

$$　$という文字。

$2　以降を上添字

$3　以降を下添字

$4　以降の添字をなくする。

$（その他）　無視

なお、NCHAR はこれらの制御文字の字数も考慮したものである。

例：AABCBDECFG212$334$445→ABCDEΦΓ$^{12}_{34}$45　このとき、NCHR=25

3.2 PNUM (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)

CALCOMP の NUMBER に IPOSIT が付け加わっただけ。ただし、NDEC<0 の時には、整数部は全部書く。

3.3 PNUMEX (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)

FORTRAN の E 変換で数字を書く。

3.4 PNUMPW (XPAGE, YPAGE, SIZE, FPN, NDEC, PFPN, IPNUM, ANGLE, IPOSIT)

FPN^{PFPN} というような指数を書く。

3.5 PNUM10 (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)

10 を底とするべきで、$A\times10^{B}$ のように書く（$1\leq A<10$、B は整数）。

3.6 PSYMSZ (SIZE, CHR, NCHAR, XLEN, YMIN, YMAX)

SIZE､CHR､NCHAR を入力とし､x 軸に沿うように文字を書いたときの横幅 XLEN、縦の座標 YMIN と YMAX を出力する。得られる数字は近似値で、正確なものではない。

3.7 PNLN (SIZE, FPN, NDEC, XLEN)

SIZE、FPN、NDEC を入力として数字を書いたときの長さ XLEN を出力する。得られる数字は近似値で、正確なものではない。

3.8 PNLNEX (SIZE, FPN, NDEC, XLEN)

E 変換形式で、SIZE、FPN、NDEC を入力として数字を書いたときの長さ XLEN を出力する。得られる数字は近似値で、正確なものではない。

3.9 PNLNPW (SIZE, FPN, NDEC, PFPN, IPNUM, XLEN, YMAX)

SIZE, FPN, NDEC, PFPN, IPNUM で定義されるべきを書いたときの長さ XLEN と高さ YMAX を出力する。得られる数字は近似値で、正確なものではない。

3.10 PNLN10 (SIZE, FPN, NDEC, XLEN, YMAX)

SIZE、FPN、NDEC を入力として 10 のべき乗の数字を書いたときの長さ XLEN と高さ YMAX を出力する。得られる数字は近似値で、正確なものではない。

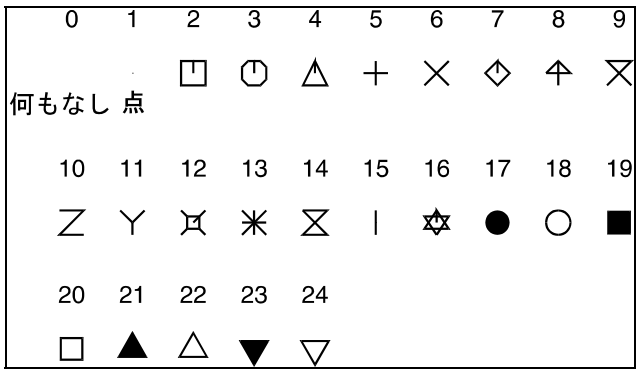
3.11 PALPHA (CSET)

使う文字をセットする。文字セットとして、「HELVETICA」「TIMES-ROMAN」「SYMBOL」を用意している。初期状態は HELVETICA である。CSET に三つのどれ

かを入力することで default の設定値が変更できる。変更は、次に変更されるまで有効である。

3.12 PMARK (XPAGE, YPAGE, SIZE, IMK, ANGLE)

与えられた座標を中心とするシンボルマークを一つ書く。0 はなし。1～16 は CALCOMP と同じ。17 以降は、見かけのサイズが同じになるように作図している。



3.13 SYMOPT (ICON, OPT)

何もしない。

3.14 KANJI (XX, YY, SIZE, ITEXT, ANG, NCH)

漢字を作図する。ITEXT は漢字に対応する数字を入れた一次元配列。漢字は区点コード番号で、ITEXT なる一次元配列に一文字が 4 桁の整数となるように入力する。4 桁の数字とは、（100×区＋点）である。9 区より小さい場合には 3 桁の数字である。

1～3 区：記号、数字、アルファベット

4～9 区：ひらがな、カタカナ、ロシア文字など

16 区～：漢字

なお、対応するコードを書式付自由フォーマットで書き出しているので、FORTRAN のコンパイラーによっては文字数が多いとき一行に入らない可能性があり、このとき文字化けする。I 文字に対して「8×I+5」文字の出力があるので、もし自由フォーマットで 1 行に 80 文字しか書かないコンパイラー（たとえば、マイクロソフトフォートラン）であれば 9 文字しか書けない。

3.15 KSPACE (SPS)

何もしない。

3.16 PSMLEN (CHR, NCH, SIZE)

これは関数である。CHR, NCH, SIZE で定義される文字を書いたときの幅を PSMLEN で出力する。

3.17 AXIS (JOB, XS, XE, ANGLE, X0, Y0, XBAS, NFG, SCALE, SCALES, NBR, NDEC, SIZE, SIZES, SMULT, SPS, NFGS, SLNS, SVAL, PEN)

座標軸を作図する。CALCOMP とは全く異なっている。

現在は別のサブルーチンになっている。

3.18 LGAXIS (JOB, XS, XE, ANGLE, X0, Y0, XBAS, NFG, NBR1, NBR2, SCALE, SIZE, SIZES, SMULT, SPS, NFGS, SLNS, SVAL, PEN)

対数軸を作図する。

現在は別のサブルーチンになっている。

3.19 点列を結ぶベジェ曲線

与えられた点を直線や、連続するスプライン曲線（ベジェ曲線）で結んでいく。途中に指定した数字を書くことも可能である。4 つのサブルーチンから構成されている。CURVES は点列の始まりを指示、CURVE で点列を一つづつ指定していく。最後に CURVEE で点列が終わることを指示する。必要に応じ CURVEO で長方形領域を指定すれば、その中しか書かないようにすることができる。

出力された曲線は、Illustrator では一つの線として認識される。しかし、等高線のように途中に数字を書いた場合には、その前後で分断された線として認識される。分断されているといっても、曲線そのものは連続している。なお、CURVES～CURVEE の一固まりのブロックはグループ化されている。

(1) CURVES (IOPEN, ILINE, IB, SIZE, FNP, IDEC, INBR, ALINE)

IOPEN	=0：開曲線
	=1：閉曲線
ILINE	=0：ベジェ曲線（最低 3 つの点が必要）
	=1：部分線形曲線（点列を直線で結ぶ）
IB	=0：何もしない。
	=1：CURVEO で入力した領域の外に出た点は作図しない。（厳密には、内部と外部を結ぶ線までは作図し、それより外部が続くと作図しない。
SIZE	等高線のように線の上に書く数字の高さ
FPN	等高線のように線の上に書く数字
IDEC	等高線のように線の上に書く数字の小数以下桁数。
INBR	端部に書く数字に関する flag
	=0：書かない。
	=1：両端
	=2：始点
	=4：終点
ALINE	ALINE≠0 の時、ALINE の絶対値おきに数字を書く。ALINE>0 であれば線の中央に、ALINE<0 であれば線の横に数字を書く。

(2) CURVEO (XMIN, XMAX, YMIN, YMAX)

指定された長方形領域。IB=1 であれば、この領域をはみ出る点は作図しない。

(3) CURVE (X, Y)

点の座標。必要な点の数だけ CURVE を CALL する必要がある。

(4) CURVEE

点列の終わり。

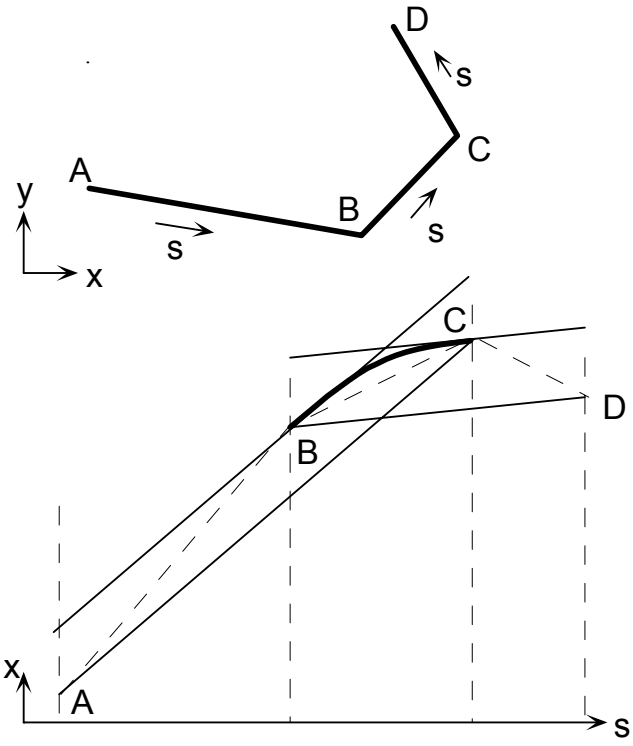
(5) 曲線の作図法

下図のように、4 点 A～D が与えられたとき、その中央部の区間 BC を通る曲線を次のように決める。まず、座標軸 s を各点を通る距離とする。すなわち、s は単調増加関数である。そして x、y をそれぞれ s を助変数とする三次式で表す。三次式の係数は 4 つあるので次の 4 つの条件より決める。

①二点、B、C を通る。

②B 点における勾配は AC に平行、点 C における勾配は BD に平行である。

このようにすれば、与えられた線分に対し次々なめらかな線分を決めることができる。なお、線端が開曲線の場合には、始点の次の点に対し、始点を軸とする軸対象な仮想点を考える。また、閉曲線の場合には最初の二区間については最後の条件が決まってから作図する。



４　プログラムの移行

CALCOMP に基づくサブルーチンは、次のようにして移行する。

4.1 文字の作図

SYMBOL は通常そのまま使える。しかし、PSYM を使う方がよい。この場合、最後に引数を一つ加える。

```
CALL SYMBOL (X, Y, SIZE, CHR, ANGLE, NC)
CALL PSYM (X, Y, SIZE, CHR, ANGLE, NC, 1)
```

ただし、CHR は文字変数である。CALCOMP の SYMBOL では NC<0 とする事でシンボルマークを書くことができるが、それには、PMARK を使うようにする。

4.2 数字の作図

NUMBER の代わりに PNUM を用いる。この場合、最後に引数を一つ加える。

```
CALL NUMBER (X, Y, SIZE, FPN, ANGLE, NC)
CALL PNUM (X, Y, SIZE, FPN, ANGLE, NC, 1)
```

なお、NUMBER では NC<0 の場合、整数部 NC -1 桁目を四捨五入し、それより上の桁の数字のみを書くのに対し、PNUM では全整数を書くので、NC<0 の場合、FPN を FPN*10**(NC+1)に置き換える。

4.3 記号の作図

CALCOMP では SYMBOL で行っていた。これを PMARK を使うようにする。

```
CALL SYMBOL (X, Y, SIZE, IMK, ANGLE, NCH)
```

①NCH=0

```
SS = SIZE / 2
THETA = ANGLE * 3.14159 / 180.
XX = X - SS * COS(THETA) + SS * SIN(THETA)
YY = Y + SS * SIN(THETA) + SS * COS(THETA)
CALL PMARK (XX, YY, SIZE, IMK+1, ANGLE)
```

②NCH=-1

```
CALL PMARK (X, Y, SIZE, IMK+1, ANGLE)
```

③NCH=-2

```
CALL PLOT(X, Y, 2)
CALL PMARK (X, Y, SIZE, IMK+1, ANGLE)
```

4.4 その他

WHERE, FACT などは使えない。したがって、これらを使っている場合にはシステムの方を変える必要がある。epsf ファイルを修正することもそれほど困難ではないの

で、必要であれば、プログラマーに相談されたい。

プログラムの引数関係

Subroutine	Entry	Call
PLOTS	PLOTL	PLOTSD
		PLOTSD
	PLOT	PLOTSD
	PLOTCV	
	NEWPEN	PLOTSD
	PLOTWD	
	PALPHA	
	PLOTAL	
	PLOTSY	
	PLOTKJ	
	ARC1	PLOTSL
		PLOTGR
	PLOTMK	PLOTGR
		PLOTSL
	PLOTGT	
PLOTSD		
PLOTSL		
GRID		PLOTGR
PATRN		PLOT
		PLOTGR
AROHD		PLOTGR
		PLOT
CIRCLE		ARC1
PLOTGR		PLOTSD
		PLOTSL
SYMBOL		PLOTSY
		PMARK1
PSYM		PLOTAL
		PLOTSY
		PALPHA
	PNUM	
	PNUMEX	
	PNUMPW	
	PNUM10	
	PSYMSZ	
	PNLN	
	PNLNEX	
	PNLNPW	
	PNLN10	
	PMARK1	
	PMARK	PLOT
		PLOTGR
		ARC1
		PLOTMK
	SYMOPT	
	KANJI	PLOTKJ
	KSPACE	
PSMLEN		
AXIS		
LGAXIS		
CURVE	CURVEO	
	CURVES	PLOTGR
	CURVEE	PLOTL

		PNUM
		PLOTCV
		PLOTGR

Plotter Subroutine by EPS Files

September 1, 1995

Nozomu YOSHIDA

Sato Kogyo Co., Ltd.

INTRODUCTION

This subroutine package in an interface program from CALCOMP plotter to EPS files.

Post Script is a convenient printer language, but have shortage to use from Windows; they cannot bring the figure into files developed by, for example, word processor. To do so, we must use EPS (Encapsulated PostScript) File. By using it, we can put figures in the documents, but still has a problem. If it is written by text, we cannot see them. In addition, even if they have screen image, we cannot edit. These are inconvenient, because it is so frequent that we want to improve or modify the figure a little. To overcome it, it is better to develop EPS files so that it is equivalent to the output files of a drawing program.

This package develops output file equivalent to Adobe Illustrator V.3.0J. If one change the prolog part based on Illustrator v.3.0, is becomes available in English mode. Note that Illustrator V.5 uses different functions, so cannot be used. Of course, V.5 can read V.3 output.

Postscript and Illustrator Languages

1 Fundamental procedure

1.1 Header and Footer

In the Illustrator, many functions are defines at the head part, and main body is written using these functions, and after that there are small amount of footer. These part are dead copy from Illustrator V.3.0.

1.2 Main body

Main body is written using the defined functions. Generally, each command is in the form:

Arguments Command Arguments

Usually, command is one or two character alphabet. For example

5 10 m move pen to a coordinate (5,10)

3 6 10 9 1 100 c Draw Bezier curve

2 Lines

Followings are typical functions for lines. Here #1 indicates first arguments, etc. Unit system is points (1/72 inch). Note that E-format in FORTRAN cannot be read.

Work	Command	Ex.	meaning of arguments
Line width	w	1 w	#1 denotes line width
line type	[]0 d	[5 5]0 d	#1, #2, ...denotes length in solid and hollow part, therefore total numerals is to be even number. number after] denotes from which point of the previously defined couple, line start; generally 0. Solid line are []0 d
Gray scale of line	G	0 G	#1 is a number between 0（black 100%） to 1（ white 100%）. Before it, we need line 0 R The meaning is not clear, so for safe, we will put it always.
Gray scale of fill	g	0 g	#1 is a number between 0（black 100%） to 1（ white 100%）. Before it, we need line 0 O The meaning is not clear, so for safe, we will put it always.
Shape at the end of line	J	0 J	#1 specify shape at the end of line, 0, 1, or 2, corresponding 
Shape at the corner	j	0 j	#1 specify shape of corner. 0: miter connection, 1: Round connection ad 2: Bebel connection. See miter limit
Flatness	i	10 i	#1 is flatness. Curve is always drawn by using piecewise linear line. This command specify errors between them. If small, good looking, but large memory
Miter limit	M	10 M	#1 is miter limit. Miter connection gives bad looking when angle if intersection is small, in which case Bebel connection is used instead.

3 LINE and STROKE

To draw line, first we should move pen at the start point, which is

x y m （Example: 10 300 m）

where x and y are coordinates. Then we draw line by moving pen. We can use either straight line or Bezier curve.

1) Straight line

x y l （Example: 20 100 l）

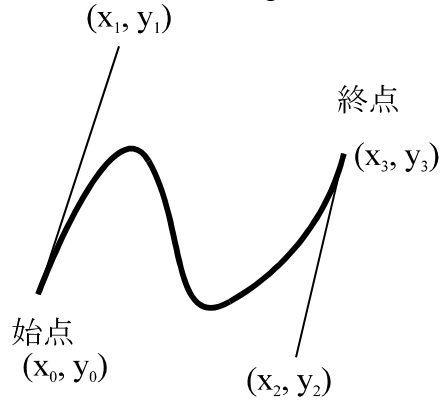
2) Bezier curve

x1 y1 x2 y2 x3 y3 c （Eg: 196. 305. 290. 291. 351.45 334.52 c）

Bezier curve is cubic curve in both x and y directions, i.e.,

$$x = a_1 s^3 + a_2 s^2 + a_3 s + a_4$$
$$y = b_1 s^3 + b_2 s^2 + b_3 s + b_4$$

where s denotes distance. Before use, we should move the pen at the start point. Then number of unknown is 6, which is specified in the command.



Let consider more general case where s start from s_1 to s_2. In the x-direction, coefficient can be computed as follows:

$$s_2 - s_1 = \sqrt{(x_3 - x_0)^2 + (y_3 - y_0)^2}$$
$$A = a_1 (s_2 - s_1)^3$$
$$B = (3a_1 s_1 + a_2)(s_1 - s_2)^2$$
$$C = (3a_1 s_1^2 + a_2 s_1 + a_3)(s_1 - s_2)$$
$$D = a_1 s_1^3 + a_2 s_1^2 + a_3 s_1 + d_1$$
$$x_1 = D + C/3$$
$$x_2 = x_1 + (C + B)/3$$
$$x_3 = D + C + B + A$$

The same discussion can be done in y-direction.

3) Stroke

Commands above just tells computer to prepare to draw line (stored in the buffer). So we need command to stroke them, which are

S or s : draw line

F or f: do not draw line, but shade in the enclosed area

B or b: both

Here upper case letter is ordinary case. Lower case letter is used when start and end point are the same to each other (closed loop)/

4 Group

A block starting from “u” to “U” are treated as group in the Illustrator. Nest is OK.

5 Circle

Postscript language have command to draw circle and semi-circle, but Illustrator does not. Instead, they use approximate four arcs defined by Bezier curve. Illustrator cannot draw semi-circle, but by using the same technique, subroutine ARC1 draw semi-circle.

6 Text

6.1 Fundamental commands

Fundamentally, we need following set of command when writing a text. Among them, state variable keeps its value until next command.

Example:

0 To	*1
1 0 0 1 240.9834 373.083 0 Tp	*2
0 Tv	*3
TP	*4
0 g	Previously shown
0 Tr	*5
(This is a) Tx	*6
(¥r) Tx	*7
TO	*8

1) Text block start from “0 To” and ends “TO”.

2) Second line shows:

cc ss -ss cc x y 0 Tp

cc, ss are direction cosine of orientation of line

x, y are coordinates

“0 Tp” is always necessary as it is.

3) “0 Tv”, “TP”, 0 Tr” are necessary as they are.

Tv indicates line orientation, TP specify end of text path, Tr declare start of render.

4) Text is drawn in such a way “(Text to be drawn) Tx”

5)”(¥r) Tx” also shows text, but, “¥r” indicates carriage return.

6) TO declare drawing text.

6.2 Various works

Some lines may be inserted between *5 and *6 to conduct various special works.

1) Text movement in the vertical direction (for super and subscript)

Insert a line such that “5 Ts” before line *6. Here 5 denotes following character will be written 5 points upper from the base line. Once they are declared, it holds until next. Therefore, we usually need “0 Ts” before TO command.

2) Several Text block

Put line similar to *6, for example,

(This is a pen) Tx

(. I am young.) Tx

comes like “This is a pen. I am young” in print

3) Change font

Insert a line before Tx command.

Example : /_Times-Roman 7 6.1421 -1.9927 Tf

Here, “/_” is always necessary, then font name comes. First digit is height by points. Then two number comes, which is proportional to point size in the ordinary usage and the ration depends on font. Considering the use in MS Windows, we prepare “Arial”, “TimeNewRoman” and “Symbol”. In addition, we support Kanji character set “MS-Mincho”.

6.3 Alignment

Insert following line

Example : 0 Ta

Here first number indicates position where x and y in Tp command. 0=Left, 1=Center, 2=Right.

6.4 Functions related to Text drawing

Function	Example	meaning are arguments	
Text Objects			
To	0 To	begin text	bindType To
TO	-	end text	
Tp	1 0 0 -1 0 0 0 Tp	begin text path	a b c d tx ty startPt Tp
TP		end text path	TP
Render and Matrix Operators			

Tr	0 Tr	begin render	render Tr
iTm		internal set text matrix	iTm - (uses _tm as implicit argument)
Tm		set text matrix	a b c d tx ty Tm
Td		translate text matrix	tx ty Td
Te		end render	Te
Ta	0 Ta	set alignment	alignment Ta 0=Let, 1=Center, 2=Right
Tf		set font name and metrics	fontname size Tf
Tl	0 0 Tl	set leading	Line distance and paragraph distance
Tt	0 Tt	set user tracking	userTracking Tt
TW	100 100 200 TW	set word spacing	minSpace optSpace maxSpace TW
Tw	0 Tw	set computed word spacing	wordSpace Tw
TC	0 0 5 Tc	set character spacing	minSpace optSpace maxSpace TC
Tc	0 Tc	set computed char spacing	charSpace Tc
Ts		set super/subscripting	offset Ts
Ti		set indentation	firstStartIndent otherStartIndent stopIndent Ti
Tz	100 100 Tz	set line scaling	withStreamScalePercent crossStreamScalePercent Tz
TA	0 TA	set pairwise kerning	autoKern TA
Tq	0 Tq	set hanging quotes	hangingQuotes Tq
Tg		set binding punctuation	bindingPunctuation Tg
TG	25 TG	set binding space	bindingSpace TG
Tv	0 Tv	set line orientation	orientation Tv
TV		set character orientation	orientation TV
Tk		kern	autoKern kernValue Tk
TK		non-printing kern	autoKern kernValue TK
T*		carriage return and line feed	T*
T*-		carriage return and negative line feed	T*-
T-		discretionary hyphen (printed)	T-
T+		discretionary hyphen (not printed)	T+
TR		reset pattern matrix	a b c d tx ty TR
TS		special chars	textString justified TS

7 Kanji

Treatment does not change much in the Postscript. The difference is the method to express kanji. One kanji character is expressed by four or five character set starting from "¥" (back slash in English mode). The rest are numeral or codes depending on the kanji they express.

8 Color

Variable k and K controls color

cyan magenta yellow black k ex. 0 0 0 1 k

PLOTTER SUBROUTINES

This package is similar to CALCOMP format but output is a data equivalent to Illustrator V.3. Note that in some FORTRAN compiler (like UNIX) we should use "¥¥" (two backslashes in English mode) to express "¥". This package is written in such a way. However, in some FORTRAN compiler (like MS FORTRAN), just "¥" is enough. So you should modify the program by all exchanging from ¥¥ to ¥ depending on machine.

1 Minimum requirement

We need to call PLOTS in such a way before using other subroutines

CALL PLOTS (X, Y, IPEN)

Here arguments are meaningless. After finishing we need to call PLOT in such a way:

CALL PLOT(X, Y, 999)

Again, X and Y are meaningless, but 999 is necessary.

Output from FORTRAN is done in Unit 7, file name is "PLT007.ai". It is also noted that one dot (actually to move pen) is drawn at coordinate (0 0). It may be convenient to eliminate this line or delete this points in Illustrator.

Fundamental line width is 0.5 points. Unit system are cm without nay description.

A block common named "PLOTCM" is used in the package.

2 Fundamental subroutines

These are subroutines similar to CALCOMP and subroutines the user need not call.

2.1 PLOTS(X, Y, IPEN)

Declare use of plotter subroutine package.

2.2 PLOT(X, Y, IPEN)

Draw line

X	x coordinate
Y	y coordinate
IPEN	2=pen down (draw line), 3=pen up (move pen). negative sigh indicate to change origin into X, Y after drawing moving the pen. This subroutine is also to be called at the end of usage of plotter package with IPEN=999.

2.3 PLOT(X, Y, IPEN)

Similar to PLOT, but line type can be selected.

As(IPEN) = 2 or 3 are same as PLOT.

IPEN>10 specify line type. Here X are pitch length and Y is sometimes used for other control. IPEN=11 to 23 are line type.

IPEN	Line type	Pitch
11	Solid	
12	Dotted	0.1
13	Dashed	0.3
14	Dashed	0.3
15	Dashed	0.4
16	Dashed	0.4
17	Dashed	0.4
18	Dashed	0.4
19	Chained	0.4
20	Chained	0.5
21	Dashed	1.0
22	Chained	1.0
23	Chained	1.0

Generally, X denotes pitch, default values are shown in the above table.

In case IPEN=12、21、22、23, we can control other feature. In case IPEN=12, Y denotes length of solid part (default 0.03cm). In case IPEN=21 to 23, Y denotes solid length after second couple.

IPEN greater than 100 denotes line width. 100=0.1mm, 200=0.02mm, ...

2.4 PLOTCV(X1, Y1, X2, Y2, X3, Y3)

Draw Bezier curve. See explanation of Bezier curve for arguments

2.5 NEWPEN(IP)

Changes default pen width. Default = 0.5 and when IP>1, line width =(IP-1)/100 cm.

2.6 PLAPHA (CHR)

Change English font. Initial =“HELVETICA”. We prepare “ROMAN” and “SYMBOL”. They corresponds to Arial, TimesNewRoman, and Symbol in Windows

2.7 PLOTAL (ITA, TS)

Set alignment and subscript command. See Ta and Ts Illustrator command. Valid after next order.

2.8 PLOTSY (IP, X, Y, SIZE, ANG, NCR, CHR)

Internal subroutine. Draw part of text.

IP=1: ordinary

IP=2: first part of text set

IP=3: last part of text set

IP=4: super script

IP=5: subscript

IP=6: ordinary

2.9 PLOTKJ (IP, X, Y, SIZE, ANG, NCR, CHR)

Internal subroutine to draw Kanji. Use KANJI for user.

2.10 ARC1 (X, Y, R, SANG, EANG)

Draw circle or arc.

X	X coordinate
Y	Y coordinate
R	radius. In case R<0, interior is colored black as well.
SANG	Angle at starting point (deg.)
ENAG	Angle at end point (deg.) Note, Counter-clockwise direction

2.11 PLOTMK (ITP, XX, YY, RLEN)

Internal subroutine to draw symbol mark.

ITP	type =1 rectangular =2 trigonometric =3 inverse trigonometric
XX, YY	coordinate at the center of the symbol
RLEN	Size. In case RLEN<0, with black interior.

2.12 GRID (X0, Y0, XD, YD, MX, MY)

Draw grid

X0, Y0	coordinate
XD	increment in x-direction
YD	increment in y-direction

MX	number of grid in x-direction
MY	number of grid in y-direction

2.13 PATRN (XX, YY, NP, N0)

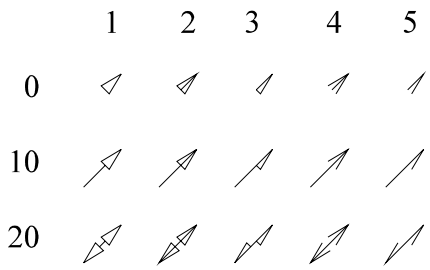
Draws multi-

XX, YY	one dimensional array storing the coordinates
NP	Number of points
N0	Gray scale from 0 (black) to 10 (white)

2.14 AROHD (XS, YS, XE, YE, S, W, KY)

Draw arrow

XS, YS	x-coordinate
XE, YE	y-coordinate
S, W	Length at arrow part
KY	10th digit (0, 1, 2) and 1st digit (1 to 5) have different meaning. See figure



2.15 CIRCLE (XX, YY, SANG, EANG, R)

Draw circle. Same as ARC1 except order of arguments are different.

2.16 PLOTGR (IP)

=1	Start of group
=2	End of group
=3	Stroke of closed lines
=4	Stroke of open lines
=5,6	Same as IP=3,4, but interior is black.

2.17 SYMBOL(XX, YY, SIZE, CHR, ANGLE, NCH)

Same as CALCOMP when NCH>0.

XX, YY	coordinates
SIZE	Height. Converted by points, therefore actual appearance is smaller
CHR	Text
ANGLE	Angle (deg.)
NCH	Number of character

2.18 FACTOR(FACT)

No work

2.19 WHERE(XX, YY, FAC)

No work

2.20 SPACE(SP)

No work

3 Application routines

Common arguments are:

XGAGE, YPAGE coordinates

SIZE height

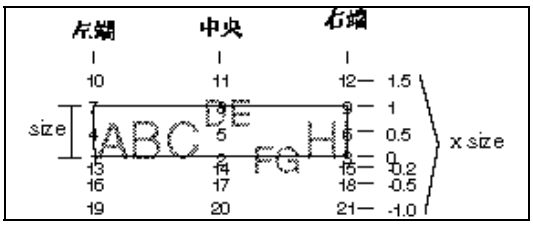
CHR text (character)

FPN, PFPN number to be drawn

ANGLE Angle

NCHAR Number of character to be drawn. Control character is also to be counted.

IPOSIT Flag between the specified coordinate and actual appearance. See figure



NDEC Number of digid after decimal point. Negative is also OK.

3.1 PSYM (XPAGE, YPAGE, SIZE, CHR, ANGLE, NCHAR, IPOSIT)

Similar to SYMBOL in CALCOMP,except that it has arguments IPOSIT. We can use several control character for special work. It is "$" and an alphabet.

$A Change character set into HELVETICA

$B Change character set into ROMAN

$C Change character set into SYMBOL

$$ character "$"

$2 Begin superscript

$3 begin subscript

$4 Begin normal

$(other) neglected

Example: AABCBDECFG212$334$445 appears ABCDEΦΓ$^{12}_{34}$45 where NCHR=25

3.2 PNUM (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)

Same as NUMBER in CALCOMP, except that it has argument IPOSIT.

3.3 PNUMEX (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)

Draw number by E-format in FORTRAN

3.4 PNUMPW (XPAGE, YPAGE, SIZE, FPN, NDEC, PFPN, IPNUM, ANGLE, IPOSIT)

Draw number in such a way FPN^{PFPN}

3.5 PNUM10 (XPAGE, YPAGE, SIZE, FPN, ANGLE, NDEC, IPOSIT)

Draw FPN in such a way $A \times 10^B$ where $1 \leq A < 10$ and B is integer.

3.6 PSYMSZ (SIZE, CHR, NCHAR, XLEN, YMIN, YMAX)

Showing the length when drawing PSYM with SIZE, CHR, and NCHAR

3.7 PNLN (SIZE, FPN, NDEC, XLEN)

Showing the length when drawing PNUM with SIZE, FPN, and NDEC

3.8 PNLNEX (SIZE, FPN, NDEC, XLEN)

Showing the length when drawing PNEX with SIZE, FPR, and NDEC

3.9 PNLNPW (SIZE, FPN, NDEC, PFPN, IPNUM, XLEN, YMAX)

Showing the length when drawing PNPW with SIZE, FPN, NDEC, PFPN, IPNUM

3.10 PNLN10 (SIZE, FPN, NDEC, XLEN, YMAX)

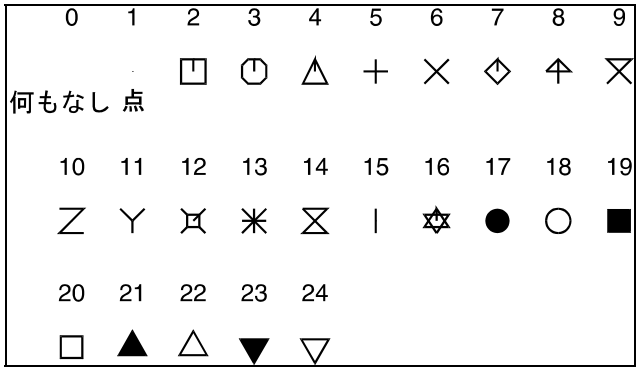
Showing the length when drawing PN10 with SIZE, FPN, NDEC

3.11 PALPHA (CSET)

Specify character set. We can use “HELVETICA”, “TIMES-ROMAN” and “SYMBOL”

3.12 PMARK (XPAGE, YPAGE, SIZE, IMK, ANGLE)

Draw symbol marks.



3.13 SYMOPT (ICON, OPT)

No work

3.14 KANJI(XX, YY, SIZE, ITEXT, ANG, NCH)

Draw Kanji. ITEXT is to be one dimensional array with 3 or 4 digit integer expressing Kanji character.

3.15 KSPACE(SPS)

No work

3.16 PSMLEN (CHR, NCH, SIZE)

This is a function to output length of character in the figure

3.17 AXIS (JOB, XS, XE, ANGLE, X0, Y0, XBAS, NFG, SCALE, SCALES, NBR, NDEC, SIZE, SIZES, SMULT, SPS, NFGS, SLNS, SVAL, PEN)

Quite different or from the CALCOMP

3.18 LGAXIS (JOB, XS, XE, ANGLE, X0, Y0, XBAS, NFG, NBR1, NBR2, SCALE, SIZE, SIZES, SMULT, SPS, NFGS, SLNS, SVAL, PEN)

Quite different or from the CALCOMP

3.19 Bezier curve

Draw Bezier curve by connecting specified points. We first call SURVES. After that we call CURVE; one Bezier curve one CALL. Then CURVEE at the end of the curve. We can create continuous line.

(1) CURVES (IOPEN, ILINE, IB, SIZE, FNP, IDEC, INBR, ALINE)

IOPEN	=0: Open loop
	=1: Close loop
ILINE	=0: Bezier curve (at least 3 points are required)
	=1: piecewise linear line connecting the specified line
IB	=0: no cutoff.
	=1: Lines outside the region which is specified by CURVEO is not drawn
SIZE	height of number to be drawn in the line
FPN	number to be drawn in the line
IDEC	Number of digit after decimal point for number to be drawn in the line
INBR	Flag number specifying to draw number at the end of line
	=0: not drawn
	=1: both end
	=2: only start end
	=4 only last end
ALINE	In case ALINE≠0, a number FPN is drawn every ALINE cm. When ALINE>0, number is drawn on the line, wheras ALINE<0, number is drawn beside the line.

(2) CURVEO (XMIN, XMAX, YMIN, YMAX)

Rectangular region where line is not drawn outside the region when IB=1.

(3) CURVE (X, Y)

Coordinate of the points. CURVE is called sequencially.

(4) CURVEE

CURVEE indicates the end of the line, therefore is called after calling CURVE's.

4 Executation of the program

4.1 Draw characters

Conventional subroutine SYMBOL is avaialbe, but the use of PSYM is recommended. When calling PSYM, the user need to add one argument in addition to the arguments required in SYMBOL.

```
CALL SYMBOL (X, Y, SIZE, CHR, ANGLE, NC)
CALL PSYM (X, Y, SIZE, CHR, ANGLE, NC, 1)
```

Here it is noted that CHR should be character. In the case of CALCOMP SYMBOL, the user can draw symbol mark by specifying NC<0, but this function is not installed in PSYM; the user should use PMARK.

4.2 Draw numbers

The user should use PNUM instead of NUMBER, in which he should add one arguments.

```
CALL NUMBER (X, Y, SIZE, FPN, ANGLE, NC)
CALL PNUM (X, Y, SIZE, FPN, ANGLE, NC, 1)
```

It is also noted that, in the case of the NUMBER, only the upper digit larger than |NC|-th digit, but whole integer is drawn in the case of PSYM. If the user does not like it, he should change FPN into FPN*10**(NC+1) when NC<0.

4.3 Draw symbold

The user should change PSYM instead of CALCOMP SYMBOL.

```
CALL SYMBOL (X, Y, SIZE, IMK, ANGLE, NCH)
```

1) NCH=0

```
SS = SIZE / 2
THETA = ANGLE * 3.14159 / 180.
XX = X - SS * COS(THETA) + SS * SIN(THETA)
```

```
        YY = Y + SS * SIN(THETA) + SS * COS(THETA)
        CALL PMARK ( XX, YY, SIZE, IMK+1, ANGLE)
```
2) NCH=-1
```
        CALL PMARK ( X, Y, SIZE, IMK+1, ANGLE)
```
3) NCH=-2
```
        CALL PLOT(X, Y, 2)
        CALL PMARK ( X, Y, SIZE, IMK+1, ANGLE)
```

4.4 Others

The user cannnot use WHERE and FACT. Therefore the user should change the program when these functions are required.