NSWC TR 80-166

COMPUTATION OF THE INTEGRAL OF THE BIVARIATE NORMAL DISTRIBUTION OVER ARBITRARY POLYGONS

by
A. R. DiDONATO
R. K. HAGEMAN
Strategic Systems Department

JUNE 1980

NAVAL SURFACE WEAPONS CENTER

Dahlgren, Virginia 22448 Silver Spring, Maryland 20910

NAVAL SURFACE WEAPONS CENTER
Dahlgren, Virginia 22448

Paul L. Anderson, Capt., USN
Commander

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

14 REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NSWC TR-80-166 ✓ 2. GOVT ACCESSION NO. AD-A102466	3. RECIPIENT'S CATALOG NUMBER (9)
4. TITLE (and Subtitle) (6) COMPUTATION OF THE INTEGRAL OF THE BIVARIATE NORMAL DISTRIBUTION OVER ARBITRARY POLYGONS	5. TYPE OF REPORT & PERIOD COVERED Final rept.
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) A. R. DiDonato and R. K. Hageman	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Surface Weapons Center Dahlgren, Virginia 22448	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NIF
11. CONTROLLING OFFICE NAME AND ADDRESS (11) Naval Surface Weapons Center (K05) Dahlgren, Virginia 22448	12. REPORT DATE June 1980
	13. NUMBER OF PAGES 168
14. MONITORING AGENCY NAME & ADDRESS (If different from Controlling Office) (12) 172	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

...iate normal probability distribution
arbitrary polygons
automatic procedure
computer program
Fortran IV
self-intersecting polygons

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An efficient automatic procedure is given for evaluating the integral of the bivariate normal density function (IBND) over an arbitrary polygon (Π). The polygon (Π), defined by N points, falls into one or more of the following classes: {S}, simple polygons; ({S}), limit elements of sequences of uniformly bounded N-sided simple polygons of the same orientation; {Π}, arbitrary polygons, including self-intersecting (SI) ones, where {S} ⊆ ({S}) ⊆ {Π}. It is not necessary to specify the class beforehand. The method extracts from (Π) a set of N exterior angular regions. The IBND is evaluated

(Continues)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

...which coincides with ... belongs to ...

411567

20. Abstract (Continued)

over each of these, and the results are properly combined to yield IBND for Π. In case Π is SI, account must be taken of the number of its "primary circuits" and their orientations. A by-product of the analyses is the evaluation of a function $A(\Pi)$ for which $|A|$, when properly interpreted, gives the area of Π.

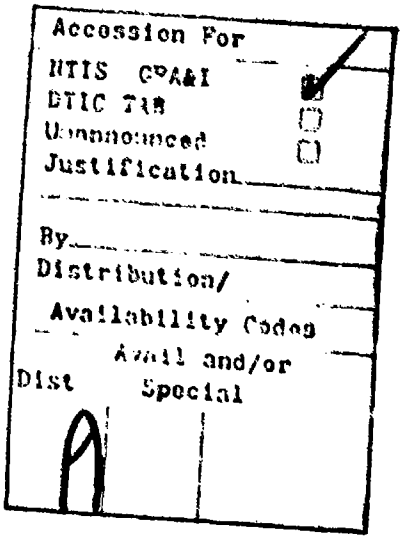
Another procedure for obtaining the same final results is described for completeness which is not as efficient. It treats an SI polygon by decomposing it into a finite set of S or $\bar{S}$ type elements. The IBND is evaluated over each of these; the results are properly summed to give the IBND for Π. In contrast to the first method, the smallest class $\{S\}$, $\{\bar{S}\}$, $\{\Pi\}$ to which Π belongs must be specified for computational efficiency.

The Fortran IV programs for both procedures are presently set to yield approximately 3, 6, or 9-decimal-digit accuracy. Fortran IV listings of the programs are given.

FOREWORD

The work described in this report was done in the Science and Mathematics Group of the Strategic Systems Department. It supplements the work reported in NSWC/DL TR-3886 of September 1978. We acknowledge Ann Howes' outstanding contributions and efforts towards the reproduction of this report for distribution.

R. T. RYLAND, JR., Head
Strategic Systems Department



ABSTRACT

An efficient automatic procedure is given for evaluating the integral of the bivariate normal density function (IBND) over an arbitrary polygon Π, defined by N points, falls into one or more of the following classes: $\{S\}$, simple polygons; $\{\bar{S}\}$, limit elements of sequences of uniformly bounded N-sided simple polygons of the same orientation; $\{\Pi\}$, arbitrary polygons, which includes self-intersecting (SI) ones, where $\{S\} \subsetneq \{\bar{S}\} \subsetneq \{\Pi\}$. It is not necessary to specify the class beforehand. The method extracts from Π a set of N exterior angular regions. The IBND is evaluated over each of these, and the results are properly combined to yield IBND for Π. In case Π is SI, account must be taken of the number of its "primary circuits" and their orientations. A by-product of the analyses is the evaluation of a function $A(\Pi)$ for which $|A|$, when properly interpreted, gives the area of Π.

Another procedure for obtaining the same final results is described for completeness which is not as efficient. It treats an SI polygon by decomposing it into a finite set of S or $\bar{S}$ type elements. The IBND is evaluated over each of these; the results are properly summed to give the IBND for Π. In contrast to the first method, the smallest class $\{S\}$, $\{\bar{S}\}$, $\{\Pi\}$ to which Π belongs must be specified for computational efficiency.

The Fortran IV programs for both procedures are presently set to yield approximately 3, 6, or 9-decimal-digit accuracy. Fortran IV listings of the programs are given.

CONTENTS

	Page
Glossary	vi
I. Introduction	1
II. Normal Probability over Convex Polygons (Summary)	4
III. Normal Probability over S and $\bar{S}$ Polygons	9
IV. Normal Probability over Arbitrary Polygons	13
V. Discussion of Computer Program B (Flow Charts Included)	25
VI. Numerical Results	46
References	78
Appendices	
A. Normal Probability over Arbitrary Polygons by (P − A)	A-1
B. Every Simple Polygon Contains an Interior Diagonal	B-1
C. An Alternative Method to Find P for Simple Polygons	C-1
D. Expressions for the Area of a Polygon	D-1
E. Program Parameters. Chebyshev Coefficients, erfc (x)/z(x), $x \geq 0$	E-1
F. Program Listings in Fortran IV	F-1
G. Triangle Checkout Program with Drezner	G-1
Distribution	

GLOSSARY

(See also page 39)

	Page
{C}	1
{S}	1
{$\bar{S}$}	1
{II}	1
{$S_n(N)$}	1
SI	1
N	1
polygonal element	1
polygon	1
IBND	2
Z(x, y)	2
P(II) Eq. (3)	1
PO, NO	2
(P − A), (P − B)	2
a, $a(R, \theta_1, \theta_2)$	4
R, θ_1, θ_2 Eq. (15)	4, 6
$P(a)$	6
u Eq. (8)	5
$\varepsilon(u)$ Eq. (8)	5
$\bar{S}$ = erfc (u) Eq. (8)	5
(5) Eq. (10)	6
a_k Eq. (10)	6
e Eq. (12)	6
$\Delta\theta$	5
g_1, g_2, h_1, h_2 Eq. (15)	6
G Eq. (18)	7
$C(v_1, v_2, \ldots, v_N)$	7
v_i	7
A, A(S) Eq. (22)	9
SAR (singular angular region)	12, 31
PAR (π-angular region)	12, 31
SDP (successive duplicate points)	12, 31
SCP (successive colinear points)	31
(j), j^{th} node	14
$\bar{j}$, j^{th} edge	14
MN, multiple node	14
SN, simple node	14
vertex	14
path	14
α-option	14
$J(j, \delta_j)$, J-disk	15
T, polygonal path	15
T_k	15
W, winding number Eq. (32), (37)	18
circuit	18
C_P, primary circuit	18
Ω Eq. (38)	20
β-option	24
IOP	25
ICV	25
IND	26
ψ	27
$\alpha_1, \alpha_2, \bar{R}$	28
s Eq. (47)	28
α_3	28
$\bar{E}$ = erf (h) Eq. (48)	28
α_4 Eq. (51)	29
V	31
WD (well-defined)	31
CU	32
Program Identification No.	26, 39
ϕ	39
∪, ∩	47
Δ_j	47
M	A-10
p, a	A-9
τ	4, A-10

I. INTRODUCTION

This report describes two automatic and efficient procedures for evaluating the integral of the general bivariate density function over an arbitrary polygon[1] $\hat{\Pi}$. Specifically, we evaluate $\hat{P}(\hat{\Pi})$, i.e.,

$$(1) \qquad \hat{P}(\hat{\Pi}) = \frac{(1-\rho^2)^{-1/2}}{2\pi\sigma_w\sigma_z} \iint_{\hat{\Pi}} \exp\left\{-\left[\left(\frac{w-\mu_w}{\sigma_w}\right)^2 - 2\rho\,\frac{(w-\mu_w)(z-\mu_z)}{\sigma_w\sigma_z} + \left(\frac{z-\mu_z}{\sigma_z}\right)^2\right] \Big/ 2(1-\rho^2)\right\} dw\,dz\,,$$

where (μ_w, μ_z) is the mean and

$$\begin{pmatrix} \sigma_w^2 & \rho\sigma_w\sigma_z \\ \rho\sigma_z\sigma_w & \sigma_z^2 \end{pmatrix}$$

is the covariance matrix of the normal random variable (w, z) with correlation coefficient ρ, $|\rho| < 1$.

Three main classes of polygonal elements[2] (S), $(\bar{S})$, (Π) are treated in the text. The set of simple polygons is denoted by (S) with the subset of convex polygons denoted by (C). The class (S) is enlarged to include elements which are limits $\bar{S}$ of sequences of uniformly bounded N-sided simple polygons with the same orientation, $(S_n(N))$. This class is denoted by $(\bar{S})$. A more extended class (Π) is obtained by adding self-intersecting (SI) polygons[3] to $(\bar{S})$.

Using the well-known linear transformation

$$(2) \qquad x = \left[\frac{w-\mu_w}{\sigma_w} - \rho\,\frac{z-\mu_z}{\sigma_z}\right] \Big/ \sqrt{1-\rho^2}\,, \quad y = \frac{z-\mu_z}{\sigma_z}\,, \quad |\rho| < 1\,,$$

in (1) results in a new integrand which has circular symmetry about the origin. In addition, since (2) maps straight lines into straight lines, $\hat{\Pi}$ transforms, by (2), to another polygon Π of the same class. Thus (1) can be written as

$$(3) \qquad \hat{P}(\hat{\Pi}) = P(\Pi) = \iint_{\Pi} Z(x, y)\,dx\,dy\,,$$

---

[1] A polygon or polygonal element will always mean a closed finite broken line, with its interior, in the plane. The last segment terminates at the first point. Its boundary is defined by an ordered set of N points in the plane. However, we specify a polygon by N + 1 points, where the (N + 1)st and first points are the same.

[2] For ease of language, polygon and polygonal element are used interchangeably.

[3] We say a polygon is self-intersecting if it is not in $(\bar{S})$. A characterization is given in Section III.

where

$$Z(x, y) \equiv \frac{1}{2\pi} \exp\left[-(x^2 + y^2)/2\right] . \tag{4}$$

Hereafter, we assume (1) has been transformed to (3), and we deal only with (3), the integral of the bivariate normal density function (IBND) for Π. Also, unless noted otherwise, we denote an element of a particular class or set by the letter in braces designating that set. For example, C refers to an element of {C}. Note that $\{C\} \subseteq \{S\} \subseteq \{\bar{S}\} \subseteq \{\Pi\}$.

We make the convention that if a simple polygon S is positively oriented, (PO), i.e., with its area on the left as one traverses the boundary continuously, (3) yields a positive result, whereas if S is negatively oriented (NO), i.e., with its area on the right as the boundary is traversed continuously, (3) yields the same result with a minus sign. If Π is SI, there can be both positive and negative contributions to P(Π). For example, in Figure 1 below, P(Π) is made up of the sum of the probabilities over triangles A and B, where A is specified by the point set (1, 2, 3, 1) and B by (3, 4, 5, 6). Thus P(Π) = P(A) + P(B), where P(A) > 0, but P(B) < 0. Clearly then, P(Π) may be negative and make no sense in terms of probability; nevertheless we often call P a probability, regardless of its sign, for ease of language. In Figure 2, we have an example of an element $\bar{S}$ of $\{\bar{S}\}$, where $P(\bar{A})$ and $P(\bar{B})$ are both positive. These concepts will be discussed more fully in Sections III and IV.

In Figure 3 we show an element of a sequence $\{S_n(11)\}$ of 11-sided simple polygons for which a limit element $\bar{S}$ is obtained by letting the points 3-9 converge onto the same straight line as shown in Figure 4.

A main objective of this report is to describe and discuss two procedures, (P - A) and (P - B), to evaluate (3). In either case, if Π is in {S}, P(Π) is found by integrating (4) over a set of exterior angular regions of Π, essentially in the same way as described in [2], [3] for convex polygons. The details are given in Section III. For background and completeness, the convex case is summarized in Section II.

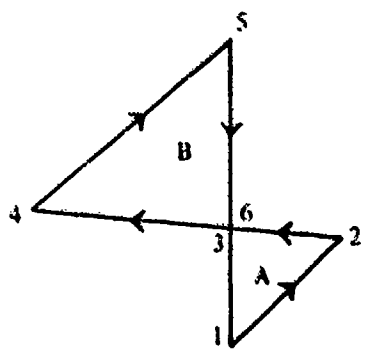


Figure 1. An SI Polygon of Class {Π}

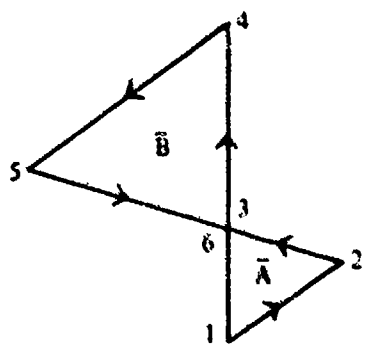


Figure 2. A Polygon of Class $\{\bar{S}\}$

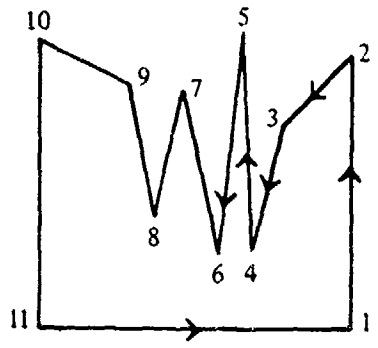


Figure 3. A Simple Polygon of $\{S_n(11)\}$

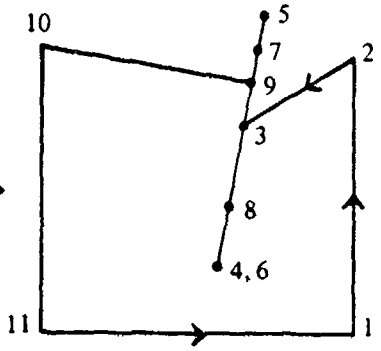


Figure 4. Limit Element $\bar{S}$ of $\{S_n(11)\}$

The same is true if Π is in $\{\bar{S}\}$. Some pre-processing of Π, which reduces it to an element of $\{S\}$, as described in Section III, may be necessary with (P−A) and desirable with (P−B). It is then treated as indicated in the previous paragraph.

If Π is self-intersecting (SI), procedures (P−A) and (P−B) are quite different. For (P−A), Π is first decomposed, by separate sub-procedures, into a set of disjoint isolated elements in $\{S\}$, or perhaps $\{\bar{S}\}$. The value of P for each of these isolated elements is computed and the results are summed to yield $P(\Pi)$. For (P−B) no decomposing procedure is necessary. It treats Π as if it were in $\{S\}$ or $\{\bar{S}\}$. This is possible because it keeps track of the number of "primary" circuits, or loops, in Π. Generally (P−B) is faster than (P−A), since there is no necessary pre-processing to do. Moreover, in (P−A), for efficiency, one must specify the smallest class to which Π belongs; if, by error, one specifies a class to which Π does not belong, then $P(\Pi)$ will be computed incorrectly. For (P−B), the smallest class need not be specified, and incorrectly computed results are not possible. Since (P−A) decomposes Π, it is useful in analyzing the configuration of Π. Nevertheless, (P−B), with some of its parts in common with (P−A), is the preferred procedure. It will be discussed in Section IV. The remainder of the discussion on (P−A), with some numerical results, is relegated to Appendix A.

We emphasize that the procedures described in Sections III and IV lead to a computer program which is completely automatic in the sense that one can simply specify, as input, the coordinates used to define Π in proper order, the number N of such points, and one of three accuracies desired for $P(\Pi)$. A by-product of the program is a function $A(\Pi)$, where $|A|$, properly interpreted, gives the area of Π.

In the most general case where Π is SI, we do not know of another program to compute P. Even if Π is simple we are not aware of an automatic program, although such a program would have many applications in probability and statistical studies. Of course, brute force methods can always be devised, such as decomposing Π into triangles and quadrilaterals [5, p. 956]. Even though it is easy to obtain a program for decomposing an arbitrary polygon into a set of triangles, the required number of such triangles would result in an inefficient procedure.[4] For example, in

[4]In Section VI, (see page 47), such a procedure is described. Combined with Dreznor's algorithm, [2, page 18] it gives us an independent method for checking our programs. A listing of the checkout program is given in Appendix G.

the case of an N-sided polygon, P would be required, in general, for $3(N-2)$ angular regions, whereas $(P-B)$ needs P for only N angular regions. The major time-consuming portion of the program for computing $P(\Pi)$ is the evaluation of P for each angular region needed.

The computer program and its flow charts are discussed in Section V. The Fortran IV CDC-6700[5] program listings are given in Appendix F. Section VI contains numerical results for polygonal elements displayed in Figures 30–58. These figures demonstrate the robustness of the program.

II. NORMAL PROBABILITY OVER CONVEX POLYGONS (SUMMARY)

The numerical method and computing program for evaluating P(C), where C denotes a convex polygon, are described in detail in an earlier report, [2]. We proceed to summarize the main ideas since most of them carry over to polygons in (S).

We use the notion of an angular region, i.e., the semi-infinite region bounded by two intersecting straight lines. There are four such angular regions associated with two intersecting straight lines and one must keep in mind which one is of interest. Let a denote the angular region of interest. As shown in Figure 5, it can be specified by the parameters R, θ_1, θ_2. Lines 1 and 2 form the boundaries of a. The quantity R denotes the distance from the origin to the vertex of a at V. When necessary we shall denote a by $a(R, \theta_1, \theta_2)$. Our objective is to give an efficient procedure to evaluate $P(a)$.

Because of the circular symmetry in the integrand of (3), it is convenient to perform a rotation of axes through the angle τ, such that the line L and the new positive x-axis coincide. The rotation for Figure 5 is shown in Figure 6. Hereafter, we assume such a rotation has been carried out and call the new axes x and y again.

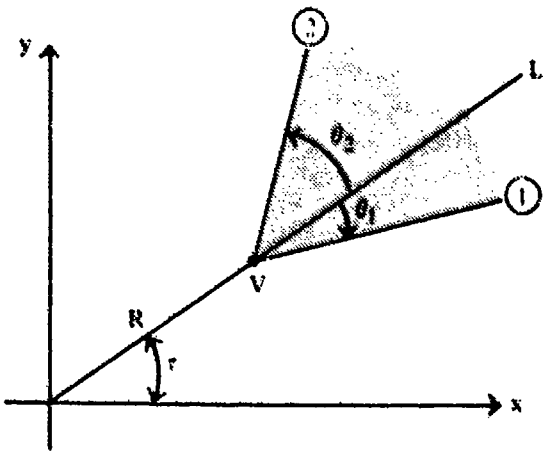


Figure 5. Angular Region $a(R, \theta_1, \theta_2)$. (Shaded Region)

[5] The CDC-6700 is a large-scale binary computer which does 10^6 arithmetic operations per second on 48-bit mantissas.

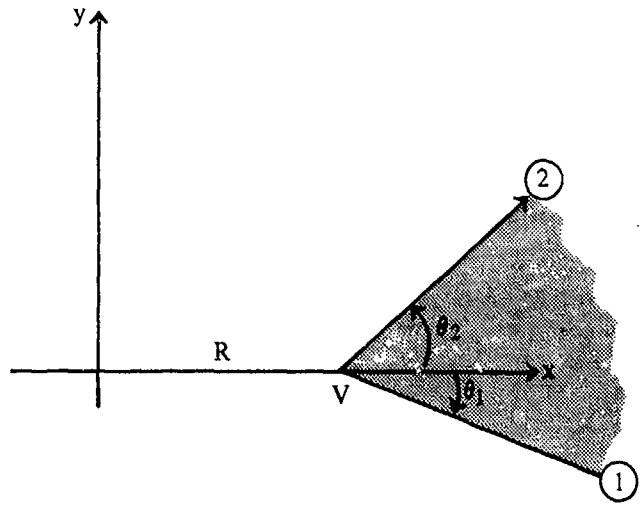


Figure 6. Showing Angular Region *a* Relative to Rotated Axes

Introducing polar coordinates (r, θ) centered at $V = (R, 0)$, we have

$$x = R + r\cos\theta, \quad y = r\sin\theta, \quad -\pi \leqslant \theta \leqslant \pi. \tag{5}$$

The variable θ will be measured, from the x-axis ($x \geqslant R$) about (R, 0), as positive in the counterclockwise direction. Using (5), (3) becomes, with a in place of Π,

$$P(a) = \frac{1}{2\pi}\int_{\theta_1}^{\theta_2}\int_0^{\infty} \exp\left[-1/2(R^2 + 2rR\cos\theta + r^2)\right] r\,dr\,d\theta. \tag{6}$$

An integration by parts in (6), on the integral in r, yields

$$\int_0^{\infty} e^{-r^2/2}e^{-rR\cos\theta}\,r\,dr = 1 - 2u\,\mathrm{erfc}(u)/z(u), \tag{7}$$

where

$$u \equiv \frac{R}{\sqrt{2}}\cos\theta, \quad z(u) \equiv \frac{2}{\sqrt{\pi}}\exp(-u^2), \quad \mathrm{erfc}(u) \equiv \int_u^{\infty} z(t)\,dt. \tag{8}$$

Using (7) in (6), and carrying out the obvious part of the θ-integration yields

$$P(a) = e^{-R^2/2}\left\{\frac{\theta_2 - \theta_1}{2\pi} - \frac{1}{\pi}\int_{\theta_1}^{\theta_2} u[\mathrm{erfc}(u)/z(u)]\,d\theta\right\}. \tag{9}$$

If $R = 0$, then $P(a) = \Delta\theta/2\pi$ as required, where $\Delta\theta \equiv \theta_2 - \theta_1$.

For exterior angular regions of polygons we can require that $-\pi \leqslant \Delta\theta \leqslant \pi$. For PO (NO) convex polygons, θ_2 will always precede θ_1 in the counterclockwise (clockwise) direction, so that $0 \leqslant \Delta\theta \leqslant \pi$ ($-\pi \leqslant \Delta\theta \leqslant 0$). Hence from (6), $P(a) \geqslant (<) 0$ for PO (NO) convex polygons.

We resolve the difficulty of evaluating the integral in (9) by using a minimax polynomial fit to $\mathrm{erfc}(u)/z(u)$ for $0 \leq u \leq c(\delta)$. That is, for a given $\delta > 0$, a set of real numbers, $\{a_k\}$, and a least positive integer K are found such that

$$\left| \mathrm{erfc}(u) - z(u) \sum_{0}^{K} a_k u^k \right| \leq \frac{2}{\sqrt{\pi}} \delta, \quad 0 \leq u \leq c(\delta). \tag{10}$$

It is shown in [2, page 6] that if (10) holds then

$$\left| \frac{e^{-R^2/2}}{\pi} \int_{\theta_1}^{\theta_2} \left\{ u[\mathrm{erfc}(u)/z(u)] - \sum_{0}^{K} a_k u^{k+1} \right\} d\theta \right| \leq \delta/\sqrt{\pi}. \tag{11}$$

Thus the constant c is chosen, once δ is specified, so that the probability over the semi-infinite region to the right of the line $x = \sqrt{2}\, c$ is equal to $\delta/\sqrt{\pi}$, i.e.,

$$\frac{1}{2} \mathrm{erfc}(c) = \epsilon = \delta/\sqrt{\pi}. \tag{12}$$

The coefficients a_k as well as K and $c(\delta)$ are given in [2] for 4 different values of ϵ corresponding to desired accuracies of 6, 9, and 12 decimal-digits in $P(a)$. They are also included in Appendix E of this report.

Recurrence relations allow us now to carry out the numerical integration of the integral in (9). We have, using (10), that $P(a)$ is given within $\pm\epsilon$ by

$$P(a) = \frac{e^{-R^2/2}}{\pi} \left[\frac{\Delta\theta}{2} - \sum_{0}^{K} a_k J_{k+1} \right], \quad |\theta_1| \leq \frac{\pi}{2}, \; |\theta_2| \leq \frac{\pi}{2}, \tag{13}$$

where

$$\begin{cases} J_k \equiv (R/\sqrt{2})^k \displaystyle\int_{\theta_1}^{\theta_2} \cos^k\theta \, d\theta \\ J_{k+1} = \dfrac{1}{k+1} \left[\left(h_2 g_2^k - h_1 g_1^k \right) + k \left(\dfrac{R^2}{2} \right) J_{k-1} \right], \end{cases} \tag{14}$$

with

$$\begin{cases} g_i = \dfrac{R}{\sqrt{2}} \cos\theta_i \qquad h_i = \dfrac{R}{\sqrt{2}} \sin\theta_i, \quad i = 1, 2 \\ R^2 = (x^2 + y^2), \qquad \text{with vertex of } a \text{ at } (x, y), \end{cases} \tag{15}$$

and

$$J_0 = \Delta\theta = \theta_2 - \theta_1, \quad J_1 = h_2 - h_1. \tag{16}$$

The constraints $|\theta_1| \leq \pi/2$, $|\theta_2| \leq \pi/2$ in (13) are required since (10) only holds for $u \geq 0$, which requires $\cos\theta \geq 0$, since $R \geq 0$. For arguments outside the range $|\theta| \leq \pi/2$, we make use of the relation

$$(17) \qquad P[a(R, 0, \theta)] = \frac{1}{2}\,\mathrm{erfc}\left(\frac{R}{\sqrt{2}}\sin\theta\right) - P[a(R, 0, \pi - \theta)], \quad |\theta| \leq \pi,$$

where $a(R, 0, \theta)$ denotes the angular region with its vertex at R, $\theta_1 = 0$, $\theta_2 = \theta$ (see Figure 6). The various situations in which (17) is needed are shown in Figure 5 of [2].

The program resulting from (13), (14), (17) is very efficient and takes advantage of situations where reduced computing effort is possible, namely when R is sufficiently large or small. For example, (18) is used when $R^2/2 \leq \alpha_2$, with G set to zero when $R^2/2 \leq \alpha_1$.

$$(18) \qquad P(a) \cong \frac{\Delta\theta}{2\pi} - G = \frac{\Delta\theta}{2\pi} - \left[\frac{1}{2\sqrt{\pi}}(h_2 - h_1) - \frac{1}{2\pi}(g_2h_2 - g_1h_1)\right].$$

See Section V (page 28) for added comments. Details are given in [2, pp. 6-8].

Letting $C(v_1, v_2, \ldots, v_N)$ denote a convex polygon with N vertices, $v_1, v_2, \ldots, v_N$, where $(v_i) \equiv (x_i, y_i)$, we have

$$(19) \qquad P(C) = 1 - \sum_{1}^{N} P(a_i).$$

This equation is fundamental to computing P(C) by the use of probabilities over appropriate angular regions a_i, $i = 1, \ldots, N$. These angular regions, quite simply, turn out to be the exterior angles of C as shown in Figure 7 for $N = 6$.[6]

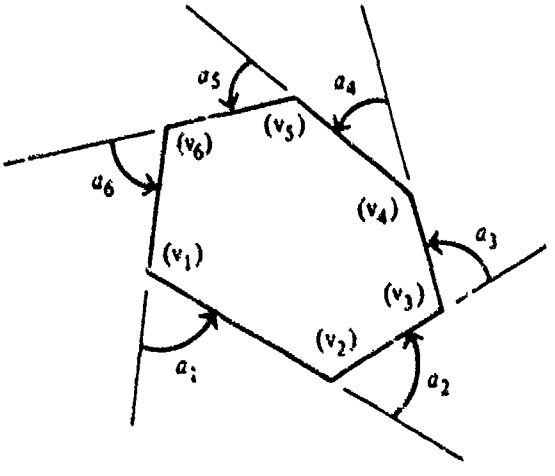


Figure 7. Convex Polygon, N = 6. Shows Angular Regions for (19).

[6] In [2], [3] we designated the angular regions for P(C) in a slightly different but equivalent way.

Clearly (19) is true since $\Sigma_1^N P(a_i) = 1 - P(C)$. We also note that the vertices are specified in counterclockwise order so that the area of C is on the left as one travels along C continuously from (v_k) to (v_{k+1}), k = 1, ..., N, $(v_{N+1}) \equiv (v_1)$.

When $\Delta\theta$ is very near 0 or π, the possibility of a catastrophic error due to round-off must be dealt with. As an example of this singular situation, suppose we are considering a polygon where one of the angular regions, say a, as shown by the solid lines in Figure 8, subtends an angle of nearly π radians with the sides of a at large perpendicular distances from the origin, so that $P(a) \sim 1.0$. But suppose that by rounding error line (1) is actually given by the computer as line (3) so that the angular region $\hat{a}$ subtends an angle $\hat{\theta}$, near $(-\pi)$ radians. The program would then yield a value $P(\hat{a})$ near zero. Moreover it would be negative since $\hat{\theta}$ is measured, from (3) to (2), clockwise rather than counterclockwise as required. The reader should note, as stated earlier, that for a positively oriented *convex* polygon all the angular regions a_i i = 1, ..., N are positive, i.e., rotating from θ_1 to θ_2 is always in the counterclockwise direction so that each $\Delta\theta_i$ is non-negative and no larger than π.

The program is alerted to a singular situation, such as the above example, whenever

$$\sin(\theta_2 - \theta_1) < 0, \tag{20}$$

for any angular region $a_1, \ldots, a_N$. If this inequality holds, and it can only hold through rounding error, a second inequality is tested, namely

$$\cos(\theta_2 - \theta_1) < 0. \tag{21}$$

If (20) and (21) are both satisfied, we resolve the difficulty by setting P(C) = 0, since if an angular region has $\Delta\theta = \pi$ for a convex polygon all vertices must be on the same line. If (20) holds but (21) does not, we set P(a) = 0, since $\Delta\theta \sim 10^{-14}$. However, in this case, a remote possibility can arise for which P(a) = 0 may be incorrect. In particular, when g_1 and g_2 are both negative and R is very large, say 10^9, P(a) may not be near zero even though $\Delta\theta \sim 10^{-14}$. Under these very unlikely circumstances, we cannot determine P(a) within the single precision capabilities of the CDC-6700, because of the inadequate precision in $\Delta\theta$.

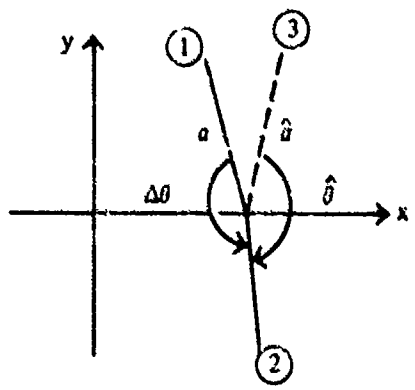


Figure 8. A Singular Case Situation

In the earlier report on convex polygons, [2], we remarked that a program was available for arbitrary polygons which made fundamental use of the probability program for convex polygons. Those ideas are briefly summarized in Appendix C. Since the completions of [2], [3] later studies revealed that a significant increase in efficiency could be made by dealing directly with simple polygons, rather than decomposing them into sets of disjoint convex polygons, as done originally. These results are the main topic of the next section.

III. NORMAL PROBABILITY OVER S AND $\bar{S}$ POLYGONS

In this section, we describe our method, based on computing normal probabilities over exterior angular regions, for evaluating P for elements in {S}. The same method, by continuity arguments, can be used to find P for elements in $\{\bar{S}\}$, provided certain precautions are taken, which will be discussed later. Just as for the class {C} (convex polygons), we shall show that P for an N-sided simple polygon requires the integration of (4) over its N exterior angular regions. Taking the precautions mentioned above into account, no more than N integrations are also needed to compute $P(\bar{S})$, where $\bar{S}$ is specified by N points.

It is important to keep in mind that the angular measure $\Delta\theta \equiv \theta_2 - \theta_1$ for an exterior angular region a of a polygon, satisfies $-\pi \leqslant \Delta\theta \leqslant \pi$.[7] Also one can see from (6) that $P(a)$ takes the same sign as $\Delta\theta$, where θ is taken positive (negative) measured from the x-axis, $x \geqslant R$, about (R, 0) in the counterclockwise (clockwise) direction, (see (5)).

We say S is *positively oriented* (PO) if its vertices (v_j), $j = 1, 2, \ldots, N$ are ordered such that the interior of S is on the *left* as the boundary is traversed continuously in the direction of increasing j. If, on the other hand, the interior of S is on the *right* as the boundary is traversed in the way just described, then we say S is *negatively oriented* (NO). For example, the polygons A and B, making up II, in Figure 1 are PO and NO, respectively.

One way to determine the orientation of S is by the sign of the expression for A(S),

$$A(S) = \frac{1}{2}\sum_{j=1}^{N} x_j(y_{j+1} - y_{j-1}), \quad y_0 \equiv y_N, \quad y_{N+1} \equiv y_1, \tag{22}$$

where (x_j, y_j) denotes the coordinates of the j^{th} vertex (v_j) of S. In fact, it can be shown that S is PO if and only if $A(S) > 0$, and NO if and only if $A(S) < 0$, (see Appendix D). The area of S is given by $|A|$. The expression in (22) yields an efficient procedure for computing A(S). The computer program for this computation, SMP-7, is listed in Appendix F. The derivation and orientation properties of (22) are given in Appendix D. Also the expression for A in the wz-plane is given there, (See page 1).

Consistent with earlier remarks that P(S) is taken positive if S is PO, we have

$$\begin{cases} P(S) > 0, & \text{if } A > 0, \\ P(S) < 0, & \text{if } A < 0, \quad (P(S) = 0, \text{ if } A = 0), \end{cases} \tag{23}$$

(See page 26).

[7] For computations, a strict inequality on the left is used. More on this later in the report.

Moreover, since A is a continuous function of the vertex coordinates of S, (22) must also hold for all $\bar{S}$ in $\{\bar{S}\}$. Thus S may be replaced by $\bar{S}$ in (23).

Consider first that S is PO; then we shall show that

$$P(S) = 1 - \sum_{i=1}^{N} P(a_i). \tag{24}$$

Here a_i, as before, denotes the exterior angular region at the i^{th} vertex of S, which is formed, as always, by extending the sides $(\overline{i-1,\ i})$ and $(\overline{i,\ i+1})$ as shown in Figures 7 and 9, where for i = 1, $(\overline{0,\ 1}) \equiv (\overline{N,\ 1})$ and for i = N, $(\overline{N,\ N+1}) \equiv (\overline{N,\ 1})$. A glance shows that (24) is the same as the expression for P(C) given by (19). In (19) each $P(a_i)$ is positive. In (24) this will not be the case if the interior angle at (v_i) of S exceeds π radians. For example, in Figure 9, the interior angle at (3) exceeds π, so that a_3 is measured in the clockwise direction rather than counterclockwise. Hence $\Delta\theta < 0$ for a_3 and $P(a_3) < 0$. Note that $P(a_i) > 0$ for i = 1, 2, 4.

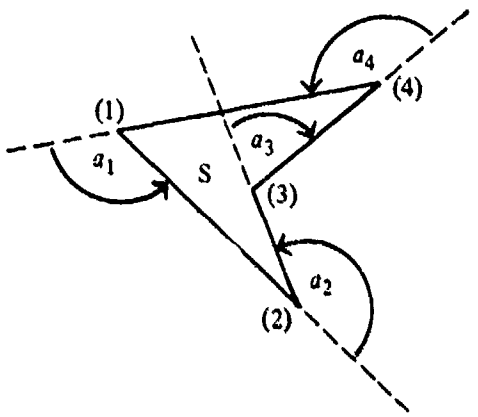


Figure 9. Polygon S. Shows Angular Regions a_i, i = 1, ..., 4.

Since the sign of $P(a)$ is determined by the sign of $\Delta\theta$, it may also be fixed by the sign of $\sin \Delta\theta$. Thus, we also have

$$\begin{cases} P(a) > 0 & \text{if } \sin \Delta\theta > 0, \\ P(a) < 0 & \text{if } \sin \Delta\theta < 0, \quad \Delta\theta \equiv \theta_2 - \theta_1, \quad |\Delta\theta| \leq \pi. \end{cases} \tag{25}$$

As a second example, in Figure 3, on page 3, S is PO with $P(a_i) > 0$ for i = 1, 2, 3, 5, 7, 9, 10, 11, and $P(a_i) < 0$ for i = 4, 6, 8. The ambiguous case, $\sin \Delta\theta = 0$, with $|\Delta\theta| = \pi$, is connected with the precautions mentioned earlier for $\bar{S}$, and will be discussed later in this section.

Let (S+) denote S when PO, and let (S−) denote the same configuration when NO. If (24) holds for (S+), then

$$P(S-) = -1 - \sum_{i=1}^{N} P(a_i), \tag{26}$$

where a_{N+2-i} from (24) and a_i from (26), with (N + 1) = (1), are vertical angles with their measures of opposite sign.

Indeed, since $P(S-) = -P(S+) = -1 + \Sigma P(a_i)$, and $\Sigma P(a_i)$ for $(S-)$ has the same absolute value, but opposite sign as the corresponding quantity for $(S+)$, (26) follows directly. By continuity arguments, (26) also holds for NO elements in $\{\bar{S}\}$ provided certain precautions are taken.

The truth of (19) for convex polygons is obvious. In the case of (24) we give a heuristic argument for its validity, which can be made rigorous.

The argument is inductive. Certainly (24) holds for N = 3. Some insight is gained by considering N = 4 with S not convex, as in Figure 9. We see that $1 - P(S)$ is obtained by considering $\Sigma_1^4 P(a_i)$, where $P(a_3)$, which is negative, compensates exactly for the excessive positive contribution from $P(a_2)$. Thus (24) holds for N = 4.

Now assume (24) is true for $N = J - 1$, $J \geqslant 4$. We shall show that (24) holds also for N = J. We look at the special case J = 8, with Figure 10, since the essentials of a rigorous proof are contained in the arguments for this case.

First a diagonal $\bar{L}$ is drawn from vertex (3) to vertex (7) which remains inside S. Such a line can always be found for any simple polygon; a proof of this fact is given in Appendix B. This line separates S into two simple polygons with the same orientation as S. Call the separated polygons D and E. Each has fewer than J vertices. From Figure 10, D is defined by the vertices, of S, (1), (2), (3), (7), (8), and E by the vertices (3), (4), (5), (6), (7). Note S, D, and E are all PO. By assumption, (24) holds for both D and E, where we have a_i, $i = 1, 2, \ldots, 8$, the angular regions of S; $\mathcal{S}_j$ and $\mathcal{E}_j$ $j = 1, 2, 3, 4, 5$, the angular regions of D and E. Hence

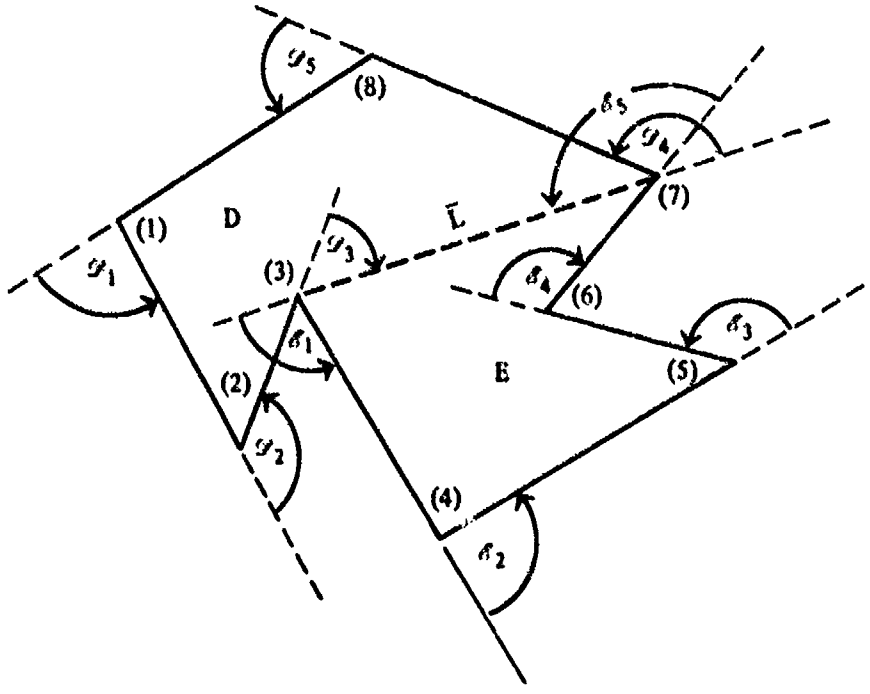


Figure 10. Shows Angular Regions of Simple Polygons D and E

$$(27)\qquad P(D) = 1 - \sum_{i=1}^{5} P(\mathscr{D}_i), \quad P(E) = 1 - \sum_{i=1}^{5} P(\mathscr{E}_i), \quad P(D) + P(E) = P(S),$$

where

$$(28)\qquad \mathscr{D}_1 = a_1, \mathscr{D}_2 = a_2, \mathscr{D}_5 = a_8, \mathscr{E}_2 = a_4, \mathscr{E}_3 = a_5, \mathscr{E}_4 = a_6.$$

We also have $\mathscr{D}_3$, $\mathscr{D}_4$, $\mathscr{E}_1$, and $\mathscr{E}_5$ as shown in Figure 10. Moreover, with $P(a_3) < 0$ and $P(a_7) > 0$, clearly

$$(29)\qquad P(a_3) = P(\mathscr{D}_3) - [P(\bar{L}) - P(\mathscr{E}_1)], \quad P(a_7) = P(\mathscr{E}_5) + P(\mathscr{D}_4) - [1 - P(\bar{L})],$$

where $P(\bar{L})$ denotes the (positive) probability over the half-plane *below* the extended line $\bar{L}$. From (27),

$$(30)\qquad P(D) + P(E) = 2 - \sum_{i=1}^{5} P(\mathscr{D}_i) - \sum_{i=1}^{5} P(\mathscr{E}_i).$$

By using (28) and (29) in (30) we have

$$(31)\qquad \begin{aligned} P(S) = P(D) + P(E) = {} & 2 - P(a_1) - P(a_2) - [P(a_3) + P(\bar{L}) - P(\mathscr{E}_1)] \\ & - [P(a_7) - P(\mathscr{E}_5) + 1 - P(\bar{L})] - P(a_8) - P(\mathscr{E}_1) - P(a_4) - P(a_5) \\ & - P(a_6) - P(\mathscr{E}_5) = 1 - \sum_{i=1}^{8} P(a_i). \end{aligned}$$

This completes the argument based on Figure 10. In order to make the proof rigorous, it is necessary to consider all the basically different possibilities for the measures of the angular regions at the two vertices on the diagonal $\bar{L}$. In Figure 10, the interior angle at (3) was greater than π radians and the one at (7) less than π radians. The three other possibilities were checked. The arguments in these cases required nothing new and are not included.

In discussing the sign associated with $P(a)$, see (25), the ambiguous case arose where $\sin \Delta\theta = 0$, $|\Delta\theta| = \pi$. The precautions we mentioned at the beginning of this section, in order to extend our results from the class $\{S\}$ to the class $\{\tilde{S}\}$, are necessary because of this ambiguous case for $P(a)$. To specify these precautions explicitly, we introduce several definitions.

We say an angular region a is *singular* (SAR) if (a) or (b) holds:

(a) Three consecutive points specifying a, say $(k-1)$, (k), $(k+1)$ are colinear, i.e., are on the same line L, with $|\Delta\theta| = \pi$. Such an angular region is called a *π-angular region*, (PAR).

(b) Two successive points of a, say $(k-1)$, (k), coincide. We call them *successive duplicate points*, (SDP).

Examples of (a) occur in Figure 4 for k = 4, 5, 6, 7, 8. Examples of (b) are shown in Figure 31, where (1), (2) and (3), (4) and (15), (16) are SDP.

The problem of SDP is easily treated by discarding one of the points from the xy-array specifying the polygon $\bar{S}$.[8] Clearly dropping such points does not affect the value of $P(\bar{S})$.

The PAR, say a, which is our prime concern here, has the property that its angular measure $\Delta\theta$ can take either a plus or minus value of π and $P(a)$ given, with a proper choice of signs, by $\pm 1/2$ erfc $(\pm t/\sqrt{2})$, where t denotes the normal distance from the extended line L to the origin. Once the sign of $\Delta\theta$ is chosen "correctly," and this is not trivial to do, the signs of $P(a)$ and of the argument t are determined. In addition, from a computational viewpoint, the correct choice of signs for a PAR must take into account the fact that the 4-quadrant arctangent subroutine returns $\Delta\theta = \pi$ which may be wrong. If the correct choice is not made, then *computationally* such an element of $\{\bar{S}\}$ is considered to be SI. The reasoning for this must be postponed until we give a characterization of SI polygons in the next section.

As a consequence of SAR(s), our main program package contains two subroutines for evaluating $P(\bar{S})$ which treat SAR(s) in different ways. One, VALR-7, cannot evaluate $P(\bar{S})$ directly if SAR(s) exist in $\bar{S}$. It resolves the problem by pre-processing $\bar{S}$ with another subroutine SORT III which eliminates SAR(s). This is permissible since the deletion of such regions does not affect the value of $P(\bar{S})$. For example, SORT III in processing $\bar{S}$ of Figure 4, would delete points (4), (5), (6), (7), (8) which clearly would not change the value of $P(\bar{S})$. The deletions by SORT III reduce $\bar{S}$ to a simple polygon which can then be treated by VALR-7 to compute $P(\bar{S})$.

The other subroutine for evaluating $P(\bar{S})$, VALR-2, based on (P − B), is the more versatile routine. In Sections IV and V, we shall show that it can find $P(\Pi)$ for any Π in $\{\Pi\}$. It does not need to pre-process Π, and yet it handles PAR(s) and SDP so that $P(\Pi)$ is always computed correctly. Thus, it has no problem with singular situations mentioned on page 8. Like VALR-7, it integrates (4) over N-angular regions to determine $P(\Pi)$, where Π is specified by N points.

The subroutine VALR-7 is included in the program package, because it can be, on the average, slightly faster than VALR-2 for simple polygons, where, of course, most applications occur. It uses A(S), (22), to decide whether (24) or (26) is needed. The reason for its greater speed will be given in Section V, page 38.

In the next section, the discussion is extended to arbitrary polygons.

IV. NORMAL PROBABILITY OVER ARBITRARY POLYGONS

In this section we show how to evaluate (3) for SI polygons. By our earlier remarks, these elements belong to $\{\Pi\}$ but not to $\{\bar{S}\}$.

[8] We limit our discussion here to elements in $\{\bar{S}\}$, but certainly SAR(s) can also occur with elements of $\{\Pi\}$ not in $\{\bar{S}\}$.

Recall that (P−A), to be discussed in Appendix A, is based on decomposing an SI polygon Π into a set of disjoint elements[9] in $\{S\}$ or $\{\bar{S}\}$. In addition, if care is not taken, and a class is specified to which Π does not belong, then a wrong value of P will result; moreover for computational efficiency, it is necessary to specify the smallest class to which Π belongs.

For (P−B), on the other hand, we shall show it is not necessary to specify the class to which Π belongs, and that P for any Π in $\{\Pi\}$ is evaluated by computing P over its N exterior angular regions, where N, as usual, denotes the number of points specifying Π.

We first characterize SI polygons. Then we describe in detail and establish procedure (P−B), (see page 3) for evaluating P(Π).

One way to note an SI polygon is to show it is not in $\{\bar{S}\}$. Our classification procedure is straightforward, and it is easy to conclude from it, in principle, if a polygon is SI. Before supporting these statements, some additional definitions and notation are given.

The *j^{th} node* (j) associates the integer j with the j^{th} xy-point, (x_j, y_j) of the ordered point set V which defines Π. The set V is denoted by the set of nodes $(1, 2, \ldots, j-1, j, j+1, \ldots, N+1)$. Let the *$j^{th}$* edge of Π, denoted by $\bar{j}$, mean the directed line segment of Π originating at $(j-1)$ and terminating at (j), so that $\bar{j}$ terminates and $\overline{j+1}$ $(\overline{N+1} \equiv \bar{1})$ begins at (j). We say $\bar{j}$ and $\overline{j+1}$ are *associated* with the j^{th} node. Of course, more than one node can exist at the same xy-point, in which case that point is called a *multiple node* (MN). If only one node occurs, it is called a *simple node* (SN). For example, in Figure 13, page 16, the first node has two edges $\bar{1}$, $\bar{2}$ associated with it. Nodes (4) and (7) at that same xy-point have the edges $\bar{4}$, $\bar{5}$ and $\bar{7}$, $\bar{8}$ associated with them, respectively. We identify a particular MN by MN(i), where (i) refers to the first node met at that xy-point. In the example above of Figure 13, we would refer to MN(1). A *vertex (j) of* Π is a point of V such that $\bar{j}$ and $\overline{j+1}$ have different slopes. We define a *path* [j, k] of Π as a line made up of consecutive edges $\bar{j}, \overline{j+1}, \ldots, \bar{k}$, $j < k$.

In order to characterize an SI polygon, we use an *α-numbering* scheme, sometimes simply designated as an *α-option*, for specifying Π. By this scheme, all vertices, points where two segments cross, and initial and terminal points of overlapping segments, are numbered in their natural order as Π is traced; i.e., starting from a point (1), each time such a situation is met it is numbered, in sequential order, until Π is completely traced. Some polygons numbered under the α-option are shown in Figures 13, 15, 18, 19, 20, 24, 25, and one that is not is given in Figure 26.[10] We will give a brief further discussion on Figures 25 and 26 at the end of this section, (Page 24).

To establish whether Π is SI, we use the fact that Π cannot be a limit element of $\{S_n(N)\}$, a sequence of uniformly bounded N-sided simple polygons with the same orientation, if the path formed by two of its edges associated with a nodal point at an MN *penetrates* another such path at the same MN. By penetrate, we mean pass through rather than just meet. Such a situation is shown in Figure 11. If two paths [j, j + 1], [k, k + 1] just meet at an MN, as shown in Figure 12, Π may be reached by a sequence $\{S_n(N)\}$, as in Figure 2, page 2.

---

[9]Two polygonal elements of Π are disjoint if they have no more than one node in common. A node is defined in the fifth paragraph on this page. A set of elements is disjoint if its elements are pairwise disjoint.

[10]Often, for computations, the number of nodes under the α-option can be reduced.

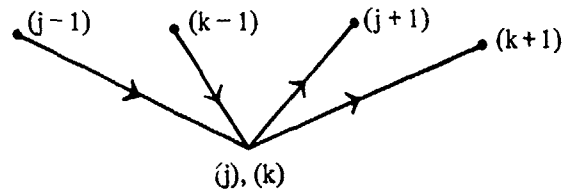


Figure 11. A Situation for an SI Polygon

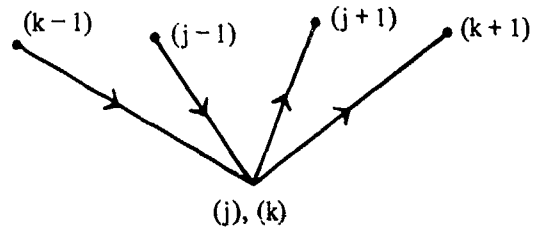


Figure 12. Situation Does Not Imply an SI Polygon

Using these notions, it is easy to see that the polygon of Figure 1 is SI, whereas the one in Figure 2 is in $\{\bar{S}\}$. There are, however, configurations with overlapping edges, such as appear in Figures 15, 18, and 19, (page 17), which need a more precise description for SI polygons.

With this in mind and with MN(j) denoting the MN at (j), where (j) is the first numbered node at that point, let $J(j, \delta_j)$ represent a disk centered at (j) with radius δ_j, where δ_j is chosen so small that $J(j, \delta_j)$ intersects only those edges of Π which originate or terminate at MN(j). Often, we simply refer to such a disk as a *J-disk*.

The approach now is to make a T-construction, i.e., to construct a closed polygonal path T, which is "close" to Π, and which is then used to classify Π, as to type, $\bar{S}$ or SI. So, consider two successive nodes (k − 1), (k)[11] with $2 \leq k \leq N + 1$, and $(N + 1) \equiv (1)$. Let T_k denote a segment of T which will be taken "close" to edge $\bar{k}$. This segment is constructed as follows: Begin with k = 2.

(A) $T_k = \bar{k}$, if (k − 1) and (k) are SN points, $(N + 1) \equiv (1)$.

(B) $T_k = B_k$, if (k − 1) is an SN and (k) is at MN(j), $1 < j \leq k$, where B_k emanates from (k − 1) and terminates at a point t_k in $J(j, \delta_j)$. If $k = N + 1$, then we require $j = 1$, $T_{N+1} = T_1 = B_1 = \bar{1}$ so that T is closed, i.e., T_1 terminates and T_2 originates at (1).

(C) $T_k = D_k$, if (k − 1) is at MN(i) and (k) is an SN, $1 \leq i < k$, where D_k emanates from t_{k-1}, a point in $J(i, \delta_i)$, and terminates at (k). If i = 1 and k = 2, then $t_1 = (1)$ so that T will be closed.

(D) $T_k = L_k$, if (k − 1) is at MN(i) and (k) is at MN(j), $i \neq j$, $1 \leq i < k$, $1 \leq j \leq k$, where L_k emanates from t_{k-1} in $J(i, \delta_i)$ and terminates at t_k in $J(j, \delta_j)$. If $k = N + 1$, then $j = 1$, and for T to be closed, $T_{N+1} = T_1 = L_1$ with $t_{N+1} = t_1 = (1)$. Also, for the same reason, if $k = 2$, $i = 1$ and $t_1 = (1)$.

[11] With no loss in generality (k − 1) and (k) are assumed not to be SDP.

Repeat the procedure with k = 3, 4, ..., N + 1 to obtain T. Clearly by choosing the δ_i sufficiently small T can be obtained arbitrarily close to Π. Now if the t_k can be chosen, for any δ_i so that T is simple, then by choosing a sequence of δ_i approaching zero, for each δ_i, a sequence of T's can be constructed which make up $\{S_n(N)\}$ converging to Π. Hence in this case Π is in $\{\bar{S}\}$. If this cannot be done, i.e., if in some J-disk paths of T must cross then Π is SI since it cannot be obtained as the limit of a sequence $\{S_n(N)\}$. If an intersection takes place in $J(j, \delta_j)$ between paths $[T_k, T_{k+1}]$ and $[T_{k+m}, T_{k+m+1}]$ we say Π has a SI point at (k, k + m).

Clearly from this characterization of $\bar{S}$ and SI elements, by T, the polygon in Figure 1 is SI and the one in Figure 2 is in $\{\bar{S}\}$. A more complex example, given in Figure 13, shows it is necessary to consider all of the nodes at an MN. Accordingly, it is not determined that Π of Figure 13 is SI until $J(1, \delta_1)$ is entered for the sixth time, as shown in Figure 14, where B_{10} cannot possibly terminate at (1) without intersecting B_7 and/or D_8.

From Π of Figure 13, an interesting situation is reached by letting (5) and (6) converge, by sequences of points, to locations of (3) and (2), respectively, such that each polygon of the associated sequence is SI. The limit element, however, shown in Figure 15, is in $\{\bar{S}\}$. There is nothing contradictory about this with respect to our previous remarks. Figures (16) and (17) show that $\bar{S}$ of Figure 15 can be obtained as the limit of a sequence of SI elements, or as the limit of a sequence $\{S_n(9)\}$. Hence, it is still correct to say the element of Figure 15 is in $\{\bar{S}\}$, since T for it can be constructed as a simple polygon as shown in Figure 17. And, it is also correct to maintain that if T is SI, then there exists a sequence of SI elements which converge to an SI limit element since by the way T is made there cannot exist a sequence $\{S_n(N)\}$ converging to that same limit element. Note that $\bar{S}$ of Figure 15 has a PAR at (4) which is the reason for the phenomenon just described. More will be said on this near the end of this section.

We show two more interesting examples in Figures 18 and 19. The element of Figure 18 is in $\{\bar{S}\}$; i.e., it is not SI, whereas the polygon in Figure 19, by our characterization, is SI.

SI polygons should not appear often in practice. But, if the generation of a polygon is not under control of the user, say the nodes are computer assigned, then SI polygons can occur. For example, consider Figure 20. It is clearly SI at MN(3) with self-intersection point (3,6). Note that a renumbering $1 \to 1,\ 2 \to 2,\ 3 \to 3,\ 7 \to 4,\ 8 \to 5,\ 6 \to 6,\ 4 \to 7,\ 5 \to 8,\ 9 \to 9$ gives the same regions, but now the renumbered element is in $\{\bar{S}\}$.

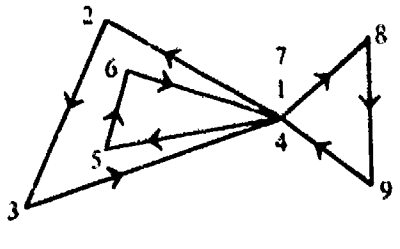


Figure 13. An SI Polygon, Π

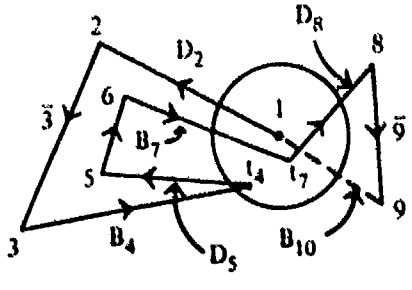
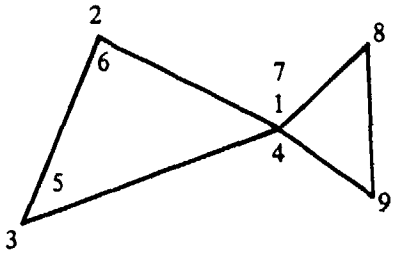


Figure 14. T for Π of Figure 13

Figure 15. A Polygon $\bar{S}$ Derived from Π of Figure 13



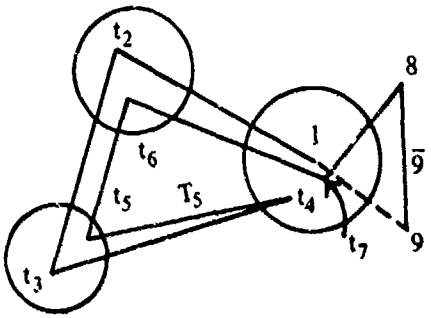


Figure 16. Incorrect T for $\bar{S}$ of Figure 15

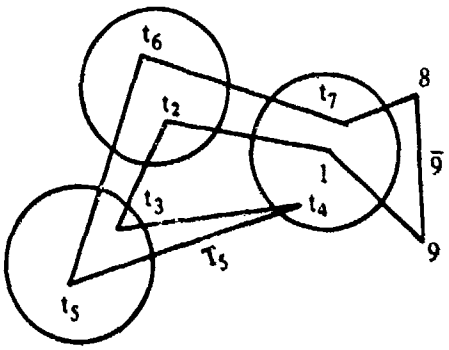


Figure 17. T for $\bar{S}$ of Figure 15

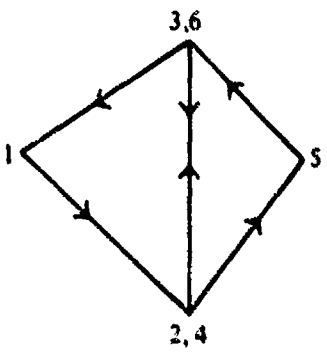


Figure 18. A Polygon in $(\bar{S})$

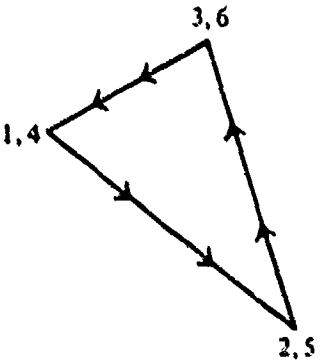


Figure 19. An SI Polygon

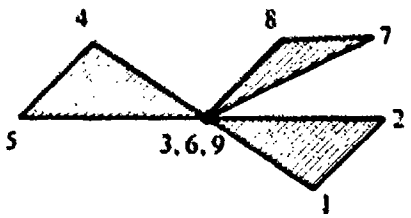


Figure 20. An SI Polygon

The basis for (P−B) is given by one equation, which is a main result of this section. Namely, for any element Π of {Π}

$$P(\Pi) = W - \sum_{1}^{N} P(a_i), \tag{32}$$

where $P(a_i)$ is the value of P for the i^{th} exterior angular region of Π, and W is a new quantity, which we define below and call the *winding number of* Π. Thus, there are two basic steps here. The first is to evaluate $P(a_i)$ for each i = 1, 2, ..., N, and the second is to compute the winding number of Π. The first has been discussed extensively in the earlier sections and offers no difficulties, except that there remains to discuss the evaluation of $P(a)$ when a is a PAR. We shall show below that W can be obtained by simply adding up the angular measures $\Delta\theta_i$ in radians of the a_i and dividing the sum by 2π. Thus, for an element in {S} or {$\bar{S}$} that is PO(NO) W = 1(−1), which gives agreement of (32) with (24) ((26)) for such elements.

We now need the following definitions and additional notation, where Π is numbered under the α-option:

A *circuit* of Π is a closed path of Π with no self-intersections. Thus a circuit is in {$\bar{S}$}, and its first and last points are located at the same MN. Note that if Π is in {S}, then MN(1) is the only MN in the sense that Π is closed and (N + 1) is at MN(1).

A *primary circuit of* Π, $C_p(\Pi)$, is the first circuit detected in tracing Π, starting at (1), which terminates at a self-intersection of Π,[12] say between paths [k, k + 1] and [j + m, j + m + 1] at MN(j), where $j \leq k < j + m$, (see Figure 11), with $C_p = (k, k + 1, \ldots, j + m - 1, j + m)$. It contains all other nodes (k + i) at MN(j) such that $k < k + i < j + m$. As an example, C_p of Figure 20 is (3, 4, 5, 6); note that k = j = 3, j + m = 6, there are no other nodes (3 + i) at MN(3) with $3 < 3 + i < 6$. In Figure 50, C_p = (3, 4, 5, 6, 7, 8, 9); we have k = 3 = j, j + m = 9, all other nodes (3 + i) at MN(3) with $3 < 3 + i < 9$ are included in C_p. Namely, node (6) with i = 3, i.e., $C_p(\Pi) \neq (6, 7, 8, 9)$. If Π is an $\bar{S}$ element, then $C_p(\Pi) \cong \Pi$.

The *winding number of a circuit C* is given by $\Sigma_1^r \Delta\theta_j/2\pi$, where $\Delta\theta_j$ is the angular measure, in radians, of the j^{th} exterior angular region of C. The integer r denotes the number of points specifying C. The winding number is +1(−1) if C is PO(NO).

In order to establish (32), let $\Pi_1 \cong \Pi$ and decompose Π as follows:

- (α) Obtain $C_p(\Pi_1)$. Set i = 1. Go to (δ).
- (β) Find $C_p(\Pi_i)$. Go to (δ).
- (γ) Π has been decomposed into a set of disjoint elements in {$\bar{S}$}. (See Footnote ·, page 14). The decomposition is complete. If i = K, we say Π has or decomposes into K primary circuits.
- (δ) If $C_p(\Pi_i) = \Pi_i$ go to (γ). Otherwise, delete $C_p(\Pi_i)$ from Π_i (except for its last node), call the result Π_{i+1}, set i + 1 = i, and go to (β).

[12]Care must be taken when PAR(s) are involved. More later. See footnote 13.

For example, for the SI polygon of Figure 20, we have

$\Pi_1 = (1, 2, \ldots, 9, 10)$, $C_P(\Pi_1) = (3, 4, 5, 6)$, $\Pi_2 = \Pi_1 - C_P(\Pi_1) = (1, 2, 6, 7, 8, 9, 10)$ $C_P(\Pi_2) = \Pi_2$. Thus $\Pi = C_P(\Pi_1) \cup C_P(\Pi_2)$, where $\cup$ denotes union.

The decomposition of Π into primary circuits can always be carried out. Indeed, by a slight modification of a proof by Knopp [4, page 15], one can state that any polygon can be decomposed into a finite set of disjoint elements in $\{\tilde{S}\}$. Knopp's proof is constructive, and, if it is followed, as described in Appendix A for (P − A), the polygon Π is decomposed into a set $\{\tilde{S}\} = (S^1, S^2, S^3, \ldots, S^J)$ of simple polygons and a set $\{L\} = (L^1, L^2, \ldots, L^Q)$ where each L^i denotes an overlapping line segment (a PAR). Now $C_P(\Pi_1)$ is made up of the union of a subset of $\{\tilde{S}\}$ and a subset of $\{L\}$. Deleting these elements from $\{\tilde{S}\} \cup \{L\}$ leaves Π_2. Then $C_P(\Pi_2)$ is found, for Π_2, in the same way from $\{\tilde{S}\} \cup \{L\} - C_P(\Pi_1)$. For example in Figure 22 we have $L^1 = (3, 4, 5)$, $S^1 = (1, 2, 3, 6)$ and $C_P(\Pi_1) = L^1 \cup S^1 = \Pi$, with $W_1 = W = 1$.

In general,

$$\Pi = \bigcup_{1}^{K} C_P(\Pi_i), \quad 1 \leqslant K < N, \tag{33}$$

where the $C_P(\Pi_i)$ are disjoint, (See Footnote 9, page 14). Hence

$$P(\Pi) = \sum_{1}^{K} P[C_P(\Pi_i)], \tag{34}$$

where, using (24) and (26),

$$P[C_P(\Pi_i)] = W_i - \sum_{n=1}^{K_i} P(a_{in}), \quad W_i \equiv \begin{cases} 1 \text{ if } C_P(\Pi_i) \text{ is PO} \\ -1 \text{ if } C_P(\Pi_i) \text{ is NO}, \end{cases} \tag{35}$$

with a_{ij} denoting the j^{th} of K_i exterior angular regions of $C_P(\Pi_i)$. Then, substituting (35) into (34) gives

$$P(\Pi) = \sum_{i=1}^{K} W_i - \sum_{i=1}^{K} \sum_{n=1}^{K_i} P(a_{in}). \tag{36}$$

The winding number of Π, W, is defined to be

$$W \equiv \sum_{i=1}^{K} W_i. \tag{37}$$

Now let

$$\Omega = \sum_{i=1}^{N} \Delta\theta_i, \quad -\pi \leqslant \Delta\theta_i \leqslant \pi,^{13} \tag{38}$$

where $\Delta\theta_i$ has the usual meaning, i.e., it denotes the angular measure, in radians, of the exterior angular region a_i of Π. To establish (32) and that W can be expressed in terms of Ω, we need to show that

$$\begin{cases} \Omega/2\pi = W \\ \sum_{k=1}^{N} P(a_k) = \sum_{i=1}^{K} \sum_{n=1}^{K_i} P(a_{in}). \end{cases} \tag{39}$$

We proceed to present the elements of a proof for (39). The argument goes as follows: Suppose $C_P(\Pi_1) = (j, j+1, \ldots, j+m)$, so that $\Pi_2 = (1, 2, \ldots, j-1, j, j+m+1, \ldots, N+1)$, where $\Pi_2 = \Pi_1 - C_P(\Pi_1)$. Denote the exterior angles of $C_P(\Pi_1)$ and Π_2 at (j) by $a_{1,1}$ and $a_{2,j}$, respectively. Denote their corresponding angular measures by ζ and λ. Also, let $\Delta\theta_j$ and $\Delta\theta_t$, $t = j+m$, denote the measures of a_j and a_t of Π.

With the aid of Figure 21 below, we have for one particular situation,

$$\zeta + \lambda = \Delta\theta_j + \Delta\theta_t. \tag{40}$$

Then from geometrical arguments, we must have also

$$a_{1,1} \cup a_{2,j} = a_j \cup a_{j+m}, \quad \text{and} \quad P(a_{1,1}) + P(a_{2,j}) = P(a_j) + P(a_{j+m}). \tag{41}$$

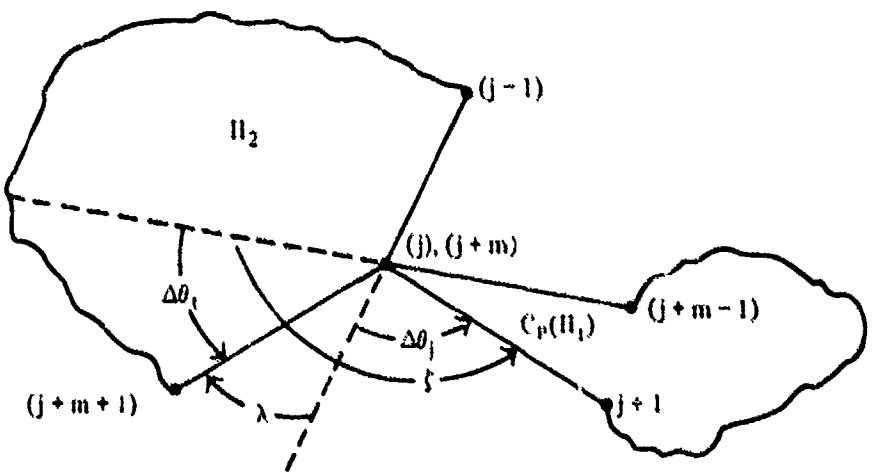


Figure 21. Parts of $C_P(\Pi_1)$ and Π_2 of SI Polygon at (j, j + m)

[13] To remain consistent with our characterization of $\bar{S}$ and SI elements from the T-construction, we need $|\Delta\theta_i| < \pi$, but computationally we always have $-\pi < \Delta\theta_i \leqslant \pi$ as a result of using the 4-quadrant arctangent routine to compute $\Delta\theta_i$.

In fact (40) and (41) hold for any other configuration with a self-intersection at (j, j + m). There are five other basically different configurations to check. This has been done; the details are not given.

Assume now that Π has only one self-intersection. Then by the relations given in (40) and (41), we see that the only angular regions affected are at (j) and (j + m), where the intersection takes place. Therefore with (34) and (35)

$$(42)\quad P(\Pi) = P[C_P(\Pi_1)] + P(\Pi_2), \quad (W = W_1 + W_2),$$

$$= W_1 - \left[P(a_{1,1}) + \sum_{k=j+1}^{j+m-1} P(a_k)\right] + W_2 - \left[\sum_{k=1}^{j-1} P(a_k) + P(a_{2,j}) + \sum_{k=j+m+1}^{N} P(a_k)\right].$$

Using (41) in (42) we have

$$(43)\qquad P(\Pi) = W - \sum_{k=1}^{N} P(a_k).$$

It remains to show the first equation of (39) in the case of one self-intersection. We have the winding numbers W_1 and W_2, for $C_P(\Pi_1)$ and Π_2, respectively, given by

$$(44)\qquad 2\pi W_1 = \zeta + \sum_{j+1}^{j+m-1} \Delta\theta_k, \quad 2\pi W_2 = \lambda + \sum_{1}^{j-1} \Delta\theta_k + \sum_{j+m+1}^{N} \Delta\theta_k,$$

where Π with only one intersection, implies that Π_2 is a circuit. Consequently, using (37) and (40)

$$(45)\qquad W = W_1 + W_2 = \frac{1}{2\pi}\left[\zeta + \lambda + \sum_{\substack{k=1 \\ k\neq j, j+m}}^{N} \Delta\theta_k\right] = \frac{1}{2\pi}\sum_{k=1}^{N} \Delta\theta_k = \Omega/2\pi.$$

To treat the case where Π decomposes into $K(>2)$ primary circuits, an induction argument can be used. Assume (32) holds for all elements Π with no more than $K - 1$ primary circuits. The essence of a proof that (32) holds for polygons with (decomposable into) K primary circuits is obtained from the argument above for $K = 2$.

Let Π have K primary circuits. Then by the decomposition procedure described above,

$$\Pi = C_P(\Pi_1) \cup \Pi_2,$$

where Π_2 can be decomposed into $K - 1$ primary circuits with winding numbers $W_2, W_3, \ldots, W_K$. But (32), by the induction hypothesis, holds for Π_2. Therefore, the remainder of the argument goes as above for $K = 2$, where W_2 is replaced by $\Sigma_{i=2}^{K} W_i$ in (42), (43), (44) and (45).

On pages 10, 12, we remarked that the evaluation of $P(a_k)$ requires some precaution, when a_k is a PAR, i.e., $|\Delta\theta_k| = \pi$. The basic difficulty is that the sign of $\Delta\theta_k$ cannot be determined from

the arctangent subroutine, because it always gives π for a PAR. Thus to determine the sign requires some additional analysis. This analysis amounts to correctly choosing the direction T_{k+1} should take in the construction of T (see page 15). For example, an attempt to construct T for $\bar{S}$ of Figure 15 by choosing T_5, as shown in Figure 16, would not be right. This amounts to choosing the sign of $\Delta\theta_4$ incorrectly since elements typical of the one in Figure 16 are SI. The correct direction for T_5 is shown in Figure 17 which indicates $\Delta\theta_4 = -\pi$ rather than π.

We can further elucidate the difficulty with the aid of Figures 22 and 23. The elements in both figures are in $\bar{S}$. Call them $\bar{S}_2$ and $\bar{S}_3$, respectively. Both elements have a PAR at (4). In Figure 22, the value for $\Delta\theta_4$ of $\bar{S}_2$ is π and for $\bar{S}_3$, $\Delta\theta_4 = -\pi$. We see for $\bar{S}_2$ that $1 - P(\bar{S}_2) = P(a_1) + P(a_2) + P(a)$, and $P(a) = P(a_4) + P(a_3) + P(a_5)$. Thus any routine, such as VALR-7, designed to compute P for elements in {S}, but not for SI polygons, would give the correct result for $\bar{S}_2$. For $\bar{S}_3$ however, the situation is different. The arctangent subroutine would give $\Delta\theta_4 = \pi$ which is incorrect for $\bar{S}_3$ to be in $\{\bar{S}\}$ and consequently $P(a_3) + P(a_4) + P(a_5) \neq P(a)$ and the result from VALR-7 for $P(\bar{S}_3)$ would be wrong.

The subroutine VALR-7 forms a part of our preferred program package, because of its slightly superior speed over VALR-2 in computing P for elements in {S}. So, in order to also use VALR-7 for elements in $\{\bar{S}\}$, rather than include the additional steps in the program to determine $\Delta\theta$ correctly for PAR(s), a routine SORT III was designed which pre-processes $\bar{S}$ by deleting from its specifying array V all SAR(s) (see page 13). The result is an element in {S}, say S, such that $P(S) = P(\bar{S})$, since SAR(s) do not affect the value of $P(\bar{S})$ by their removal.

In the case of VALR-2, which is based on (P−B), the winding number rectifies any wrong choice for $\Delta\theta$ at a PAR. For $\bar{S}_2$, in Figure 22, $\Delta\theta_4$ is computed correctly; therefore, W = 1 for $\bar{S}_2$. For $\bar{S}_3$ however, $\Delta\theta_4$ is computed incorrectly as noted above, so that from Figure 23, we have

$$P(a_4) + P(a_3) + P(a_5) = 1 + P(a),$$

where $P(a)$ is the correct value to add to $P(a_1) + P(a_2)$ to get $P(\bar{S}_3)$. Now by (32)

$$P(\bar{S}_3) = W - [P(a_1) + P(a_2) + 1 + P(a)].$$

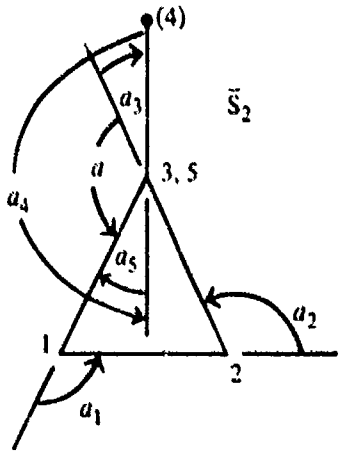


Figure 22. A Polygon $\bar{S}_2$ in $\{\bar{S}\}$. $\Delta\theta_4 = \pi$, W = 1

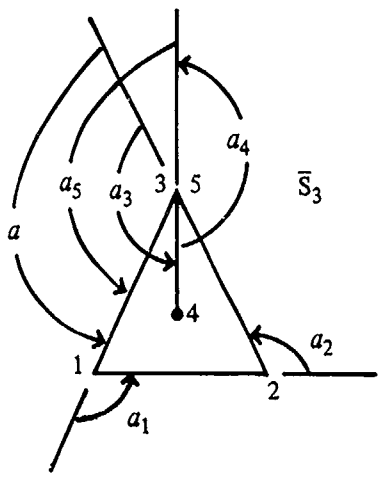


Figure 23. A Polygon $\bar{S}_3$ in $\{\bar{S}\}$, ($\Delta\theta_4$ incorrect), W = 2

But note that the computation of W using (38) with the arctangent routine is two, since $C_P(\Pi_1) = (3, 4, 5)$, $C_P(\Pi_2) = (1, 2, 3, 6)$ and $W_1 + W_2 = 1 + 1 = 2$. Hence $P(\bar{S}_3)$ is given correctly by

$$P(\bar{S}_3) = 1 - [P(a_1) + P(a_2) + P(a)].$$

Hence, an element with PAR(s) may be classified by the T-construction as in $\{\bar{S}\}$ and yet its winding number from (38), using the arctangent routine, (See Footnote 13) will not necessarily be ±1. Thus *computationally* it must be treated as SI.

It is to be noted that the value of P for a PAR requires an erfc function computation. For example, in Figure 30, W = 6 and 8 erfc functions are needed. Hence, it may be worth using SORT III also with VALR-2 to eliminate SAR(s), (See flow chart for P-2, page 40).

We conclude this section by giving two more examples of decomposing Π into primary circuits, and some remarks on numbering Π other than by the α-option.

Consider the polygon of Figure 24. We shall show that W = 3. Note, the first self-intersection occurs at (4,7):

$$C_P(\Pi_1) = (4, 5, 6, 7), \quad W_1 = 1, \quad \Pi_2 = (1, 2, 3, 7, 8, 9, 10, 11, 12, 13, 14, 15).$$

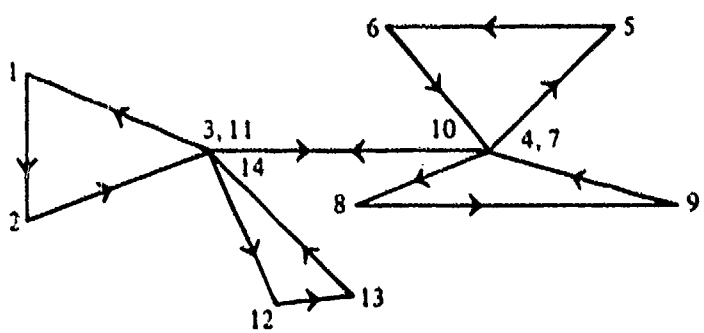


Figure 24. An SI Polygon, W = 3

The first SI point of Π_2 is at (3, 11); by the preceding discussion,

$$C_P(\Pi_2) = (3, 7, 8, 9, 10, 11), \quad W_2 = 1, \quad \Pi_3 = (1, 2, 11, 12, 13, 14, 15),$$

where $C_P(\Pi_2)$ is PO and in $\{\bar{S}\}$. With no remaining self-intersections, we have

$$C_P(\Pi_3) = \Pi_3, \quad W_3 = 1.$$

Hence W = 3.

For Figure 25 (note the α-numbering) the decomposition of Π is as follows:

$C_P(\Pi_1) = (6, 7, 8, 9, 10, 11)$, $W_1 = -1$, $\Pi_2 = (1, 2, 3, 4, 5, 11, 12, 13, 14, 15, 16, 17)$

$C_P(\Pi_2) = (4, 5, 11, 12)$, $W_2 = 1$, $\Pi_3 = (1, 2, 3, 12, 13, 14, 15, 16, 17)$

$C_P(\Pi_3) = (2, 3, 12, 13, 14, 15)$, $W_3 = -1$, $\Pi_4 = (1, 15, 16, 17)$

$C_P(\Pi_4) = \Pi_4$, $W_4 = 1$.

Hence W = 0.

In the actual computation of $P(\Pi)$, it is often not necessary to specify Π by an α-numbering (see page 14). However, one should not assume, for example, that all points for which $\Delta\theta = 0$ can be dropped when using (P−A), although this is permissible in (P−B). In Figure 55, if that element were treated as SI, an additional node at MN(1) would be required, following (6), for (P−A), using SORT I, to give the correct result. This will be clarified in Appendix A (See page A-7).

In Figure 25, six points can be dropped for computation purposes using (P−B), namely (2), (4), (6), (11), (12), (15). The reduced numbering scheme as shown in Figure 26 is called a *β-numbering scheme*, or simply a *β-option*.

An additional short discussion on the superiority of (P−B) over (P−A) is given on page A-15.

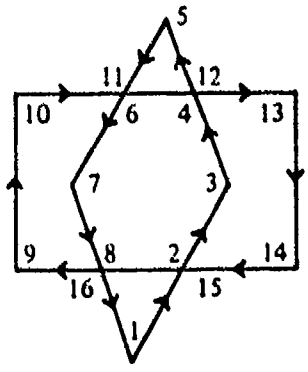


Figure 25. SI Polygon with α-Option. W = 0.

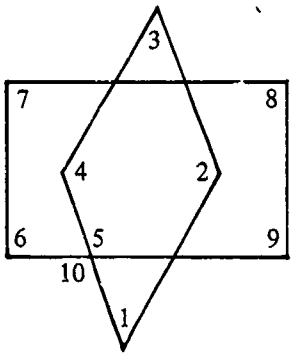


Figure 26. SI Polygon with β-Option. W = 0.

V. DISCUSSION OF COMPUTER PROGRAM B (FLOW CHARTS INCLUDED)

The program package described here, call it Program B, contains five subprograms, each in subroutine format, P-2, VALR-2, SORT III, VALR-7, SMP-7. The second, fourth, and fifth can be used independently; the first serves as a master routine. The last three are also used in the program of Appendix A, called Program A. We shall discuss each subprogram, and point out how each is used in connection with the others. VALR-7 has much in common with VALR-2; a detailed discussion on it is not given. However, the essential differences between it and VALR-2 are noted. Keep in mind that VALR-7 is on the average the slightly more efficient of the two for computing P for elements in {S}, but VALR-2 alone, which is based on (P−B), is capable of determining P for any element of {Π}.[14] Flow charts for the first four subprograms are included at the end of this section, pages 40-45. They will be used to aid in the discussion. No flow chart is given for SMP-7, which is used to compute A.

The given polygon, call it Π, for which P is wanted, is specified by its nodes at points (x_k, y_k), k = 1, 2, ..., N. The call line of each routine identifies these quantities by x, y, N.

The parameter IOP appears in the call line of P-2, VALR-2, VALR-7. It specifies the approximate accuracy to which P(Π) is computed. The user assigns IOP = 1, 2, or 3, so that P for each angular region of Π is computed with 3, 6, or 9-decimal-digit accuracy, respectively.

The parameter ICV appears in the call line of P-2. With this parameter, the user can specify, for maximum efficiency of computation, various combinations of the above routines or subprograms. The flow chart for P-2, page 40, summarizes the combinations one may choose. If Π is in {S}, then the user should set ICV = 0 and P-2 would call VALR-7 to compute P(Π). When N = 1 is specified, the normal probability, regardless of the ICV value, is given for the angular region a which is specified by three points. If the user is uncertain about the class to which Π belongs, ICV > 0 should be used. Then P-2 calls VALR-2 to find P(Π) where Π can be in {Π}. If Π has π-angular regions, PAR(s) (see page 12) and Π is not SI, then it may be more efficient to use SORT III with VALR-7 rather than VALR-2 alone. This can be done by setting ICV = −2. If Π has self-intersections as well as PAR(s), then VALR-2 with SORT III may be more efficient than VALR-2 alone since

[14] The package above gives an improvement in efficiency over using VALR-2 alone.

VALR-2 makes an erfc function computation for each PAR of Π; SORT III deletes such regions before VALR-2 is called. This combination can be called by setting ICV < 0, but not equal to -2. In using SORT III, N may be reduced below 3, in which case P = 0, A = 0 as shown in boxes 9 and 13 of the flow chart for P-2, page 40.

Later, in the discussion of VALR-2, the reason why VALR-7 can be slightly more efficient will be given. *But, VALR-7 can give grossly wrong values of P if it is used for SI polygons or without SORT III for an $\bar{S}$ element with SAR(s)*, (See page 31).

A by-product of VALR-2 and a necessary quantity for VALR-7 is A, which is given by

$$A = \frac{1}{2}\sum_{1}^{N} x_i(y_{i+1} - y_{i-1}), \quad y_0 \equiv y_N, \quad y_{N+1} \equiv y_1, \quad N \geqslant 3. \tag{46}$$

For VALR-7, A is computed by the subroutine SMP-7. It is used in VALR-7 to determine the orientation of Π, when Π is in $\{S\}$ or $\{\bar{S}\}$. The sign of A determines whether $P(\Pi)$ in VALR-7 is evaluated by (24) or (26), (See page 9).

No flow chart is given for SMP-7, but a listing of the program is given in Appendix F, page F-37. A derivation of (46) is given in Appendix D, where it is shown that $|A|$ when properly interpreted gives the area of Π.

For VALR-2, A is computed within VALR-2 itself. In VALR-7 it plays a crucial role as evidenced by (24) and (26). The winding number of Π, W, plays a similar but more complicated role for VALR-2 as (32) shows. In the previous section, it was shown that W is computed from $\Sigma_1^N \Delta\theta_i/2\pi$, where $\Delta\theta_i$ is the angular measure of a_i, the exterior angular region of Π at (i). The winding number in addition to $P(\Pi)$, $A(\Pi)$, and IND make up the output of VALR-2.

The error indicator IND is used in both VALR-2 and VALR-7. Its normal setting is zero. If VALR-2 is used to find P for an angular region (N = 1), and the x, y input contains a SDP, then IND is set to one and P is set to the absurd result of 5. If IND is set to two in VALR-7, or VALR-2, it means a PAR was encountered in evaluating $P(\Pi)$, and the result for $P(\Pi)$ from VALR-7 is not to be trusted, whereas the corresponding result from VALR-2 is good. If IND = 3, then a direct exit is taken since N has been set as smaller than one or equal to two. In either case such an assignment is not acceptable to VALR-2 or VALR-7. See boxes 7 and 1 in flow charts of VALR-2 and VALR-7, respectively.

For easy reference, the above programs are numbered accordingly: P-2 ↔ 1, VALR-2 ↔ 2, SORT III ↔ 3, VALR-7 ↔ 4, SMP-7 ↔ 8.

We proceed with a more detailed discussion of 2. We refer to the n^{th} box of the flow chart for 2 by 2-n.

Although 4 should be used when N = 1, this case is also handled by 2. When N = 1, P(a) as computed by 2 (or 4) always gives the normal probability for a; a negative value is never found for P(a) in this case. Three points are necessary to define a, with (1) always referring to the vertex of a with points (2) and (3) given in counterclockwise order with $0 \leqslant \Delta\theta \leqslant 2\pi$. Notice that this differs from $\Delta\theta$ for an exterior angular region of a polygon, where $|\Delta\theta| \leqslant \pi$. In Figures 27 and 28, the assignments of 3(x, y) coordinates are shown for two different angular regions when N = 1.

The sensing for N = 1 is carried out at 2-7 (flow chart for VALR-2, box 7).[15] If ψ, defined in 2-15 (see also pages 28 and 39), is nonnegative, we have an element of type shown in Figure 27, $0 \leqslant \Delta\theta \leqslant \pi$. If $\psi \leqslant 0$, Figure 28 represents a typical case, $\pi \leqslant \Delta\theta \leqslant 2\pi$. If $\psi \cong 0$, then a can have a vertex angle near 0, π, or 2π radians. Here the user must be careful, because if rounding error should interchange (2) and (3), a wrong result can be given for P(a), (See page 8). The error indicator IND is not used for this situation. The other boxes used only for N = 1 are 2-57, and 2-79.

If $N \geqslant 3$, 2 treats any polygon Π by finding $P(a_k)$, k = 1, 2, ..., N, for each exterior angular region a_k of Π. The analysis for computing $P(a_k)$, using (13) and (17), was given in Sections II and III. The $\Delta\theta_k$ term for a_k (see Figures 6, 9, pages 5, 10) is computed at 2-36 using a 4-quadrant arctangent routine which gives values in the half-closed interval $(-\pi, \pi]$. The sum of the $\Delta\theta_k$ is accumulated in Ω at 2-27. Individual terms for A, as given in (46), are computed at 2-22 (k = 1) and 2-68 and accumulated in A. The sum in (13) is computed at 2-35 and the loop 2-34, 54. The a_{m-1} in 2-34 are the Chebyshev coefficients for erfc(u)/z(u) as they appear in (13). They are tabulated in Appendix E for the three IOP settings, with an additional set for 12-decimal digits of accuracy which is not included in the program. The value of $P(a_k)$ is given by I in 2-47, where L refers to the erfc function contributions from using (17). Note if g_1 and g_2 from 2-19 are nonnegative 2-29, 30, then (17) is not needed and L = 0, 2-23, 2-37.

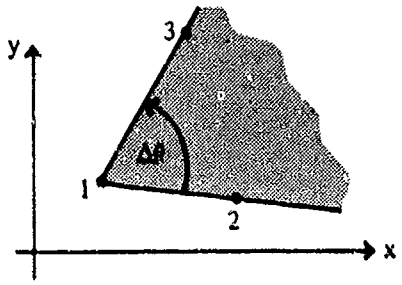


Figure 27. $\psi \geqslant 0$, P for Shaded Region

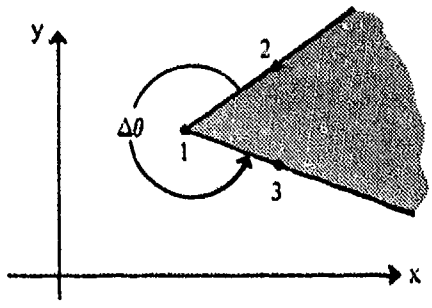


Figure 28. $\psi < 0$, P for Unshaded Region

[15]The boxes in each flow chart are numbered, usually at the upper right-hand corner.

For efficiency, yet retaining specified accuracy, if R is sufficiently small or large, then (13) is not used. This is reflected through the sensings at 2-18, 12, 28. If $R^2/2 \leqslant \alpha_1$ then $I \cong \Delta\theta/2\pi$ and if $R^2/2 \leqslant \alpha_2$, then $I \cong (\Delta\theta/2\pi) - G$, where G is computed at 2-10 from (18). If $R \geqslant \bar{R}$, then I = L, 2-28, 37. At this point, VALR-7 can be more efficient than 2. In 4 (VALR-7), $\Delta\theta_k$ is not computed until needed (see 4-16, 34) so that when (13) is not used an arctangent computation is saved. In 2, however, $\Delta\theta_k$ is computed regardless of whether (13) is used or not, because it is needed to find the winding number $W = \Omega/2\pi$. Thus, for a polygon with n vertices of S located such that $R \geqslant \bar{R}$ for each of them, 2-28, means n more arctangents would be computed by 2 than by 4. The parameters α_1, α_2, $\bar{R}^2/2$ as well as α_3 and α_4, which are discussed below, are given in Appendix E. They depend on the setting of IOP.

In 2-19, the rotation of axes for a_k is done, as discussed on page 4.

If s, given by

$$s \equiv 2\psi/D_1D_2 = \sin\Delta\theta \tag{47}$$

[16]

is sufficiently small in absolute value, 2-20, then a_k subtends an angle near 0 or $\pm\pi$. When it is sufficiently close to zero, with $\phi \geqslant 0$, 2-24, and $|\Delta\theta| \leqslant 7(-14)$, 2-14, then $0 \rightarrow I$, 2-11. If there is a no at 2-14 then $I \cong 0$ only if $g_1 \geqslant 0$, 2-13. If $g_1 < 0$, with $\Delta\theta \sim 0$, it may happen that a_k contains the origin, in which case $P(a_k)$ is not near zero for sufficiently large R.

If $|s| \leqslant \alpha_3/4$, 2-20, with $|\Delta\theta| \sim \pi(\phi < 0$, 2-24), and if $|s| \leqslant 7(-14)$, IND is set to two, 2-21. There is nothing to be concerned about here. IND is simply used to alert the user that a PAR has been encountered. Recall that in 4 if this occurs, P(Π) is not to be trusted. (This is discussed further on page A-3 of Appendix A). Now if $\psi < 0$, 2-6, then $P(a_k)$ is given by I of 2-4, and if $\psi \geqslant 0$, then $P(a_k)$ is given by I of 2-5. Boxes 4 and 5 are where the erfc function computations are made in the program for a PAR. Note the choice is made such that if $\psi = 0$ it is assumed $\Delta\theta_k = \pi$. Keep in mind that if |h| denotes the normal distance from an extended straight line, intersecting the nonnegative x-axis, to the origin, then $1/2\ \mathrm{erfc}(h/\sqrt{2})$ with $h \geqslant 0(<0)$ gives the probability over the half-plane to the right (left) of the line.

Assume now that the program moves from 2-20 or 2-13 to 2-12 and from there to 2-29. Then, in the next set of boxes, starting at 2-29, the necessary erfc computations, required by (17), or approximations to them are made and stored in L. As mentioned above if $g_1, g_2 \geqslant 0$, then (17) is not used, L is set to zero, and control is transferred to 2-28.

We use the notation

$$\begin{cases} \bar{E}(h) = \mathrm{erf}(h) = 1 - \mathrm{erfc}(h) = 1 - E(h), \quad \text{(see (8))}, \\ E(-h) = 1 + \bar{E}(h) = 2 - E(h) \end{cases} \tag{48}$$

[16] Variables appearing in the flow charts are defined, or cross-referenced to the text, on page 39, which precedes the flow charts of this section.

If $g_1 < 0$, $g_2 \geq 0$, then $-g_1 \to g_1$, $-h_1 \to h_1$, 2-29, 38 and $-\psi \to \psi$, 2-32.

If $\psi \leq 0$, then $\Delta\theta - \pi \to \Delta\theta$, 2-25, and $L = \frac{1}{2} E(h_1)$, 2-45.

If $\psi > 0$, then $\Delta\theta + \pi \to \Delta\theta$, 2-41, and $L = -\frac{1}{2} E(-h_1)$.

If $g_1 \geq 0$, $g_2 < 0$, then $-g_2 \to g_2$, $-h_2 \to h_2$, 2-29, 30, 39, and $-\psi \to \psi$, 2-32.

If $\psi \leq 0$, then $\Delta\theta - \pi \to \Delta\theta$, 2-25, and $L = \frac{1}{2} E(-h_2)$, 2-31.

If $\psi > 0$, then $\Delta\theta + \pi \to \Delta\theta$, 2-41 and $L = -\frac{1}{2} E(h_2)$

If $g_1 < 0$, $g_2 < 0$, then $-g_1 \to g_1$, $-h_1 \to h_1$, 2-29, 38, and $-g_2 \to g_2$, $-h_2 \to h_2$, 2-43, and

$L = \frac{1}{2} [E(h_1) - E(h_2)]$, 2-52.

An approximation for $E(h)$ or $\bar{E}(h)$ is used for efficiency of computations, if $|h|$ is sufficiently small, i.e., $|h| \leq \alpha_4$. The sensings on this inequality occur in 2-39, 43, 44, 49, 50. In each case, if the inequality is satisfied, $\bar{E}(h)$ is replaced by $(2/\sqrt{\pi})h$. The parameter α_4 depends on IOP, and is determined, to retain the accuracy specified, by using the mean value theorem on $\bar{E}(h)$. Indeed,

$$(49) \quad \bar{E}(h) \equiv \frac{2}{\sqrt{\pi}} \int_0^h \exp(-t^2)\, dt = \frac{2}{\sqrt{\pi}} \left[h + (4\xi^2 - 2) \frac{h^3}{6} \exp(-\xi^2) \right], \quad \xi \in (0, h).$$

Retaining the first term, the error $e(h)$ is bounded accordingly:

$$(50) \qquad |e(h)| = \frac{2}{\sqrt{\pi}} |4\xi^2 - 2| \frac{h^3}{6} \exp(-\xi^2) \leq \frac{2}{3} \frac{1}{\sqrt{\pi}} h^3 \leq \frac{\epsilon}{2}.$$

Thus

$$(51) \qquad |h| \leq \alpha_4 \equiv \left(\frac{3\sqrt{\pi}}{4} \epsilon \right)^{1/3},$$

with (49) and (50), implies

$$(52) \qquad \left| \bar{E}(h) - \frac{2}{\sqrt{\pi}} h \right| \leq \epsilon/2.$$

where ϵ and α_4 are given in Appendix E as a function of the IOP setting. For example, if (51) is satisfied for h_1 and h_2, with $g_1, g_2 < 0$, then L is computed from 2-48 rather than 2-52 with an error no larger than ϵ.

After L has been determined, control proceeds to 2-28 to check if R is sufficiently large so that the calculation of (13) can be omitted, or if the second term on the right-hand side of (17) can be dropped.

Consider the angular region a shown in Figure 29, which corresponds to case 11 in Figure 5 of [2, page 14]. The quantity $P(a)$ is found using (17), i.e., $P(a) = 1/2[E(h_2) - E(h_1)] + P(\bar{a})$, where, at 2-19, $g_1, g_2 < 0$, $h_1 > 0$, $h_2 < 0$. Now if R is sufficiently large, say $\geqslant \bar{R}$, 2-28, then $P(\bar{a})$ is negligible and its computation by (13), 2-35, 34, 47, can be bypassed by proceeding directly to 2-37.

In case $R < \bar{R}$, 2-28, the quantity (13) is computed from 2-35, 34, 47. Then $P(a_k)$ is obtained, as shown in 2-47, with the result stored in I. Recall that, for efficiency, $P(a_k)$ may be computed in various other ways as indicated at 2-2, 4, 5, 11, 37.

Control is now transferred to 2-55 to determine if N angular regions have been processed. As each $P(a_k)$ is computed, it is subtracted from P, which is initially set to zero, 2-66. If the answer is no at 2-55, the quantities w, z, D_1 at 2-59 and u, v, D_2 are updated at 2-64 to be used for the next angular region. Then $P - I \rightarrow P$ is carried out at 2-66, as noted above, and $\bar{x}(y_{k+1} - \hat{y}) + A \rightarrow A$, 2-68 also noted earlier. Control is then returned to 2-26.

When $\left[-\Sigma_1^N P(a_k)\right]$ has been computed, i.e., k = N, 2-55, 56, 60, then $W = \Omega/2\pi$ is computed, 2-61. The quantity $P(\Pi)$ is then found from

$$P(\Pi) = W - \sum_{1}^{N} P(a_k) \tag{53}$$

at 2-63 or 2-67 (see (32)), with W now an integer variable.

The remaining boxes of the flow chart 2 to discuss deal with the handling of SDP (successive duplicate points). Whenever two successive nodes of Π occur at the same xy-point, one of them is ignored in computing $P(\Pi)$. This feature is not contained in VALR-7, since it is handled by SORT III. It is included in VALR-2 so that this subroutine, entirely on its own, can find $P(\Pi)$ for any Π in $\{\Pi\}$.

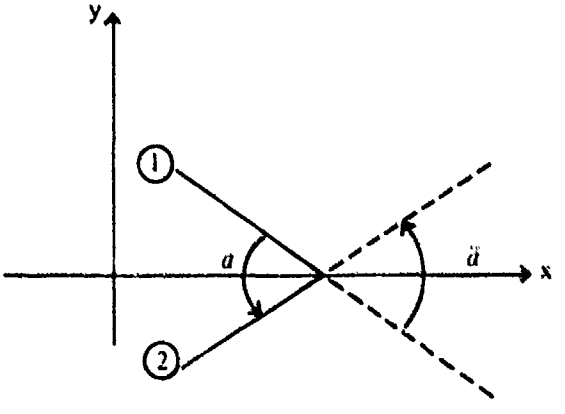


Figure 29. Shows Angular Regions a and $\ddot{a}$

If $D_1^2 \leqslant \omega^2$, where $\omega = 7(-14)$, k = 1, 2-16, then (1) and (N) are SDP (within ω) and control is transferred to 2-75. If initially N = 1, then IND is set to 1 and P to 5 since the angular region a is not defined, 2-79. If $N \geqslant 3$, then N = N − 1 and if N is reduced to two, P is set to zero for Π with an exit 2-77, 80. If N is not reduced to two, control is returned to 2-8, where new values of w and z are computed and D_1 is checked again, 2-16. If (1) and (N) are not SDP then $D_2^2 \leqslant \omega^2$ is checked at 2-17. If this inequality is satisfied, then (1) and (2) are SDP and control is transferred to 2-81. If N = 1, again the angular region is not well defined, IND = 1, P = 5 and EXIT, 2-79. If $N \geqslant 3$, then k + 1 = k and new u and v are computed, 2-78, 72, and the inequality $D_2^2 \leqslant \omega^2$ is checked, 2-73. If it is satisfied return to 2-78, increase k by one, and repeat 2-72, 73. A no eventually must occur at 2-73, because at k = N, points (1) and (N) are checked, but these cannot be SDP since for k = 1 they were checked at 2-16. With a no at 2-73, is k = N − 1? If the answer is yes, then points (2), (3), ..., (N − 1) are each, *with* (1), SDP, so P = 0, and EXIT is made, 2-74, 83, 80. If the answer is no at 2-74, then (N) and (1), (1) and (k + 1) in the array specifying Π, are not SDP; control is returned to 2-22, and 2 proceeds to compute $P(a_1)$. Computation of new u, v, D_2 quantities and updating of w, z, D_1 at 2-59, 64 for $k > 1$; also includes a check to see if (k) and (k + 1) are SDP.

The barred x and y, 2-1, 59, 64, 68, and the $\hat{y}$, 2-59, 68, are used so that once two SDP are found testing for more such points at the same k can be done with reference to the same point, namely $(\bar{x}, \bar{y})$.

We next look at SORT III. Its function is to remove points from V, the xy-array which specifies Π, when either of the following conditions hold in considering a_k.

(A) Either $\overline{k}$ or $\overline{k+1}$ (see page 14) has a length no larger than $\omega = 7$ (-14).

(B) The angle $\Delta\theta_k$ subtended by a_k, satisfies one of the inequalities $\pi - \omega \leqslant \Delta\theta_k \leqslant \pi$, $-\pi \leqslant \Delta\theta_k \leqslant \omega - \pi$.

If (A) holds, we say (k − 1) and (k) or (k) and (k + 1) are *successive duplicate points (SDP)*. If (B) holds, we say (k − 1), (k), and (k + 1) form a *PAR*. In either case, we say a_k is a singular angular region, (SAR),[17] because if (A) holds $D_1^2 = w^2 + z^2$ or $D_2^2 = u^2 + v^2$ is essentially zero, yet they must be used as divisors in 2-19 (or 4-24); if (B) holds then, because the arctangent subroutine gives values on $(-\pi, \pi]$, 4 cannot determine whether $\Delta\theta = \pi$ or $(-\pi)$ for a PAR (this is discussed on page 22). For 2, (B) holds no difficulty as explained on page 22. If neither (A) nor (B) holds, we say a_k is well defined (WD).

Given an array V made up of the ordered set of N xy-coordinates which define Π, SDP are eliminated by dropping one of those points from V. In the case of a PAR, it is removed by dropping its vertex point (k) from V,[18] where condition (B) is checked by sensing on $s^2 \equiv \sin^2 \Delta\theta_k \leqslant \omega^2$. (See (47), page 28). When this inequality is satisfied at a_k we say points (k − 1), (k), (k + 1) are *successive colinear points* (SCP). Note this inequality is also satisfied if $|\Delta\theta_k| < \omega$, so vertex points of such angular regions are also dropped.[18]

[17] Definitions here for PAR and SDP are slightly changed from those given on page 12, to account for the finite precision of the CDC-6700.

[18] The value of P(Π) is not changed by dropping such points.

After a point, or a succession of points, are deleted from V, it is "closed up" (CU). This means the array is brought together so that no gaps occur with the points renumbered in order. For example, in Figure 4, 4, 5, 6, 7, 8 would be dropped from V by SORT III, 3, and 2 or 4 would evaluate P over the CU array (1, 2, 3, 9, 10, 11, 12), which we again call V. Thus, 3 must not only detect when (A) or (B) holds, but it must also delete points from V and CU the array. In giving the details of 3, we make use of its flow chart on page 43.

At each stage of 3, an angular region a_k is under examination. It is made up of a vertex point (k), a preceding point (k − 1), and a following point (m) (initially m = k + 1). Points (k − 1), (k), (m) refer to their order as listed in V, where V may no longer be the original array due to previous deletions.

The program 3 is started with a_1, k = 1, m = 2, i.e., with points (N), (1), (2), 3-2. If (N) and (1) are SDP, then (N) is dropped from V by setting N = N − 1, 3-8, 9. This is repeated until (N) and (1) are not SDP. Similarly, (1) and (m) are tested. If they are SDP, m is increased by one (m = m + 1), and (1) and (m) are tested, where (m) now would refer to (3) of V. This is repeated until (1) and (m) are not SDP, 3-15, 16. The array V is then reduced by deleting the proper points and then CU by replacing points of V starting at (2) by points (m) through (N). Then N is replaced by N − m + 2, 3-17, 22. The value of N now refers to the number of elements in the updated CU array V. Assuming, at this point, that (A) is not satisfied for a_1, i.e., neither (N) and (1) nor (1) and (2) are SDP in V, then condition (B) is checked, 3-18. If (N), (1), (m)(=2) are SCP ($s^2 \leqslant \omega^2$), then m is increased by one until (1) and (m) are not SDP and (N), (1), (m) are not SCP, 3-19, 20, 18. If m > N, 3-19, then all points are colinear. N is set to 2 and 3 exits to P-2. If $2 < m \leqslant N$, 3-23, V is reduced and CU by replacing elements starting at (1) with elements (m − 1) through (N). The updated V will now contain N = N − m + 2 elements, 3-23, 27. Control is returned to 3-2.

If $1 < k < N$, and if (k) and (m) are SDP, where m = k + 1 initially, then a new angular region is considered by increasing m by one (m = m + 1), 3-30 until (k) and (m) are not SDP, 3-29. The V array is then reduced and CU if m > k + 1, by replacing elements starting at (k + 1) by elements (m) through (N). The updated V now contains N = N − m + 1 + k elements, 3-28, 34. Once the (A) condition does not hold, it is possible to check if $\Delta\theta$, subtended by a_k, satisfies B, 3-35. If (B) does not hold and a_k is WD, k is increased by one (k = k + 1) and the procedure for $1 < k < N$ is iterated, 3-39, 21.

If (B) holds, then (k − 1), (k), (m) are SCP and m is increased by one, 3-36. The value of m is again increased by one if (k) and (m) are SDP. This is continued until (k) and (m) are not SDP so that (B) can be tested again, 3-36, 37, 35. Eventually (A) and (B) are ruled out; if m > k + 1, then V is reduced and CU by replacing elements starting at (k) by elements (m − 1) through (N). The updated V now contains N = N − m + 1 + k elements, 3-14. (Note if m = k + 1 at 3-39, then (A) and (B) do not hold and control goes to 3-21 to look at the next angular region.) If at 3-36 m > N, then k, k + 1, ..., N are colinear. At 3-32 the k^{th} point is replaced by the N^{th} point and N is replaced by k. The updated array will now have k elements. Control is passed to 3-10.

Since (k) has been deleted, (k − 1) and the new (k) element could be SDP. If they are not, then m is set to k + 1 and the procedure described above for $1 < k < N$ is repeated until k = N, 3-21, 30, 31. If, however (k − 1) and (k) are SDP, 3-14, then k is reduced by one (k = k − 1),

3-38, m is set to k + 1, 3-33, and the procedure for $1 < k < N$ is repeated until k = N, 3-21, 3-30, or 3-36. In the event, when k is reduced, that it takes the value one, the entire procedure is restarted with the updated V at k = 1, m = 2, 3-38, 3-2.

When k = N, the N^{th} angular region, specified by (N − 1), (N), (1) with respect to the updated V is examined. It is treated in much the same way as a_1. The details may be gleaned from 3-30, 36, 31, 32, 21, 10, 11, 5, 12.

A final possibility exists that the N^{th} point (N) used to make up a previously WD a_1 is, subsequently, deleted. Therefore, after a_N has been accepted as WD, a_1 is checked again to assure that (N), (1), (2) are not SCP, 3-4, 6. If they are not, then an exit is made to P-2 with the updated V available to VALR-2 or VALR-7 depending on the value of ICV. If, however, they are SCP, then the entire procedure starting with k = 1, m = 3 must be carried out again with the updated V, 3-7, 20. This takes place in the decomposition of the element in Figure 30.

Although the ideas, and general description, given here appear straightforward, their implementation into a computer program that handles general situations, i.e., for any element of $\{\bar{S}\}$ using 3 with 4, or for any element of $\{\Pi\}$ using 3 with 2, requires an intricate code which is reflected in the flow chart of SORT III.

If 3 is used with 2, there is some duplication of effort, since both routines check for SDP. Of course, 2 can be used alone, as mentioned before, for any Π in $\{\Pi\}$, but it may be more efficient to use 3 with 2 if Π contains many PAR(s), since an erfc function computation is required for each of them when 2 is used alone, which does not occur if SORT III is used first.

We elaborate the discussion of 3 by processing the polygonal elements of Figures 30 and 31 on pages 34–37. The element in Figure 30 is in $\{\bar{S}\}$. The element in Figure 31 is also in $\{\bar{S}\}$, but *computationally*, on the basis of the discussion on pages 22, 23 (Figures 22 and 23), both 30 and 31 must be considered as self-intersecting. We shall refer to both of them as $\bar{S}$. Their processing involves every box of Flow Chart 3.

The description is given in tabulated form on pages 34-37. The first column contains N, the number of elements in V at certain stages of the processing. The second column lists the value of k, where (k) denotes the k^{th} node or point of V. It refers to the nodal point (k) which with (k − 1) and (m) define a_k. The value of m is shown, at intermediate stages, in column 3. The fourth column displays the numbers of flow chart boxes in the order they are processed. Column five shows which points are deleted from V. The letters preceding the dropped points are helpful to establish the updated version of V after deletions. For example, (a) at the head of the sixth column refers to the original V with N = 24 for $\bar{S}$ and the seventh column, headed (b), represents the CU array after points (1), (2), (3), shown in the fifth column have been deleted from V as shown under (a). Note, element (4) of the original V is the first element of the updated CU array V, in column (b), which now contains 21 elements, i.e., N = 21 at that stage. The updated V at each stage, where one or more points is dropped, is shown in CU form by columns (b), (c), ..., (q). The numbering of the elements in these columns retains the original numbering of the elements in V.

PROGRAM 3 FOR $\bar{S}$ FROM FLOW CHART BASED ON FIGURE 30

N	k	m	BOXES – SORT III	POINTS DELETED	CU, V($\bar{S}$) ARRAYS						
					(a)	(b)	(c)	(d)	(e)	(f)	(g)
24	1	2	2, 8, 15		(1)	4	4	4	4	4	4
24	1	2	17, 18		(2)	5	5	6	6	6	10
24	1	3	19, 20, 18		(3)	6	6	7	9	10	11
24	1	4	19, 20, 18		(4)	7	7	8	10	11	12
24	1	5	19, 20, 18		(5)	8	8	9	11	12	13
21	1	5	23, 27	(b): (1), (2), (3)	(6)	9	9	10	12	13	14
20	1	2	2, 8, 9	(c): (24)	(7)	10	10	11	13	14	15
20	1	2	8, 15, 17, 18, 23		(8)	11	11	12	14	15	16
20	2	3	24, 29, 28, 35		(9)	12	12	13	15	16	17
20	2	4	36, 37, 35, 39		(10)	13	13	14	16	17	18
19	2	4	14	(d): (5)	(11)	14	14	15	17	18	19
19	2	3	25, 29, 28, 35, 39		(12)	15	15	16	18	19	20
19	3	3	21, 26		(13)	16	16	17	19	20	21
19	3	4	25, 29, 28, 35		(14)	17	17	18	20	21	22
19	3	5	36, 37, 35		(15)	18	18	19	21	22	23
19	3	6	36, 37, 35, 39		(16)	19	19	20	22	23	
17	3	6	14	(e): (7), (8)	(17)	20	20	21	23		
17	2	3	38, 33		(18)	21	21	22			
17	2	4	30, 29, 28		(19)	22	22	23			
16	2	3	34, 35	(f): (9)	(20)	23	23				
16	2	4	36, 37, 35, 39		(21)	24					
15	2	4	14	(g): (6)	(22)						
15	2	3	25, 29, 28, 35, 39		(23)						
15	3	3	21, 26		(24)						
15	3	4	25, 29, 28, 35, 39								

PROGRAM 3 FOR $\bar{S}$ FROM FLOW CHART (Continued)
BASED ON FIGURE 30

N	k	m	BOXES – SORT III	POINTS DELETED	CU, V($\bar{S}$) ARRAYS			
					(h)	(i)	(j)	
15	4	4	21, 26		4	4	4	
15	4	5	25, 29, 28, 35, 39		10	10	10	
15	5	5	21, 26		11	11	11	
15	5	6	25, 29, 28, 35, 39		12	12	12	
15	6	6	21, 26		13	13	13	
15	6	7	25, 29, 28, 35, 39		14	14	14	
15	7	7	21, 26		15	15	15	
15	7	8	25, 29, 28, 35, 39		16	16	16	
15	8	8	21, 26		17	17	21	
15	8	9	25, 29, 28, 35, 39		20	21	22	
15	9	9	21, 26		21	22	23	
15	9	10	25, 29, 28, 35, 39		22	23		
15	10	10	21, 26		23			
15	10	11	25, 29, 28, 35					
15	10	12	36, 37, 35					
15	10	13	36, 37, 35, 39					
13	10	13	14	(h): (18), (19)				
13	9	10	38, 33					
13	9	11	30, 29, 28					
12	9	10	34, 35	(i): (20)				
12	9	11	36, 37, 35, 39					
11	9	10	14	(j): (17)				
11	9	10	25, 29, 28, 35, 39					
11	10	10	21, 26					
11	10	11	25, 29, 28, 35					

PROGRAM 3 FOR $\bar{S}$ FROM FLOW CHART (Continued)
BASED ON FIGURE 30

N	k	m	BOXES – SORT III	POINTS DELETED	CU, V($\bar{S}$) ARRAYS						
					(k)	(l)	(m)	(n)	(o)	(p)	(q)
11	10	12	36		4	4	4	10	10	11	11
10	10	12	32	(k): (22)	10	10	10	11	11	12	12
10	10	12	10, 11, 5		11	11	11	12	12	13	13
9	10	12	12, 10	(l): (23)	12	12	12	13	13	14	14
8	10	12	11, 12, 10, 11, 5, 4, 6	(m): (21)	13	13	13	14	14	15	
8	10	3	7, 20, 18, 23		14	14	14	15	15		
7	10	3	27	(n): (4)	15	15	15	16			
6	1	2	2, 8, 9	(o): (16)	16	16	16				
6	1	2	8, 15, 17, 18		21	21					
6	1	3	19, 20, 18, 23		23						
5	1	3	27	(p): (10)							
4	1	2	2, 8, 9	(q): (15)							
4	1	2	8, 15, 17, 18, 23								
4	2	3	24, 29, 28, 35, 39								
4	3	3	21, 26								
4	3	4	25, 29, 28, 35, 39								
4	4	4	21, 11, 5, 4, 6, 3								
			EXIT								

PROGRAM 3 FOR $\bar{S}$ FROM FLOW CHART BASED ON FIGURE 31

N	k	m	BOXES – SORT III	POINTS DELETED	CU, V($\bar{S}$) ARRAYS (a)	(b)	(c)	(d)	(e)	(f)	(g)
17	1	2	2, 8, 9	(b): (17)	(1)	1	1	1	1	1	1
16	1	2	8, 15		(2)	2	3	3	5	6	6
16	1	3	16, 17		(3)	3	4	5	6	7	7
15	1	2	22, 18, 23	(c): (2)	(4)	4	5	6	7	8	8
15	2	3	24, 29		(5)	5	6	7	8	9	12
15	2	4	30, 29, 28		(6)	6	7	8	9	10	13
14	2	3	34, 35	(d): (4)	(7)	7	8	9	10	11	14
14	2	4	36, 37, 35, 39		(8)	8	9	10	11	12	15
13	2	4	14	(e): (3)	(9)	9	10	11	12	13	16
13	1	4	38		(10)	10	11	12	13	14	
13	1	2	2, 8, 15, 16		(11)	11	12	13	14	15	
13	1	3	15, 17		(12)	12	13	14	15	16	
12	1	2	22	(f): (5)	(13)	13	14	15	16		
12	1	2	18, 23		(14)	14	15	16			
12	2	3	24, 29, 28, 35, 39		(15)	15	16				
12	3	4	21, 26, 25, 29, 28, 35, 39		(16)	16					
12	4	5	21, 26, 25, 29, 28, 35, 39		(17)						
12	5	6	21, 26, 25, 29, 28, 35								
12	5	7	36, 37								
12	5	8	36, 37, 35								
12	5	9	36, 37, 35, 39								
9	5	9	14	(g): (9), (10), (11)							
9	5	6	25, 29, 28, 35, 39								
9	6	7	21, 26, 25, 29, 28, 35, 39								

N	k	m	BOXES – SORT III	POINTS DELETED	CU, V($\bar{S}$) ARRAYS (h)	(i)	(j)	
9	7	8	21, 26, 25, 29, 28, 35, 39		1	1	1	
9	8	9	21, 26, 25, 29		6	6	6	
9	8	10	30		7	7	7	
8	8	10	31, 10, 11, 5	(h): (16)	8	8	8	
7	8	10	12, 10, 11	(i): (15)	12	12	12	
6	8	10	12, 10, 11, 5, 4, 6	(j): (14)	13	13	13	
6	8	10	3 (EXIT)		14	14		
					15			

As mentioned earlier, there is no need to discuss 4 (VALR-7) extensively, because much of the coding in 2 (VALR-2) is shared by 4 (VALR-7). Routine 4 yields P(Π) when Π = S or $\bar{S}$ with no SAR(s), such as in Figures 35 and 36. Of course, pre-processing $\bar{S}$ by 3 to remove SAR(s) allows 4 to be used with the reduced element which will then be in {S}. Routine 2 is more robust than 4, because it can find P(Π) for any element in {Π}. Routine 4, when it can be used, is preferred to 2, because it may be more efficient. This requires, however, that the user must know, a priori, that he has an element in {S}. *VALR-7 cannot be used alone for an element in* $\{\bar{S}\}$ *with SAR(s), and cannot be used at all for an SI element. If it is used alone under any of these circumstances, the value for P will most likely be wrong.*

Some specific differences between 4 and 2 are:

(1) 4 uses SMP-7 to compute A, 4-9. The sign of A determines whether (24) or (26) is used to find P, 4-58, 59, 62. The quantity A is computed internally in 2 as a by-product; it has no specific use there, 2-22, 68.

(2) The setting of the error indicator IND is normally set to zero. If IND = 2 in 2 or 4, then the input polygon contains a PAR. In this case the output for 2 is correct, but for 4, P and A are probably wrong. If IND = 1 in 2, then the input N was specified as one and SDP occurred. If IND = 3 in 2 or 4, then the input N < 1 or N = 2. For IND = 1 or IND = 3 the output is meaningless.

(3) A winding number W is not computed in 4 ((38) and (39)), since 4 never treats an SI element alone. This has the advantage that if $\Delta\theta$ is not used for a particular a_k, say if R is large or sin $\Delta\theta$ is small, a call to the arctangent subroutine can be bypassed, 4-20, 18, 11, 12, 13, 4, 5, 6, 7, 35, 38. For 2, all $\Delta\theta$ must be computed to evaluate W which is needed, since 2 is based on (32). Thus 4 should be used instead of 2 for simple polygons, and also for $\bar{S}$ elements without SAR(s).

Finally, it is recalled that a polygon may often be specified by either of two numbering schemes called α and β-options (see page 24). Generally β is the desired option for computation, since it may require fewer points to specify the given polygon (see page 24). However if the user wishes to determine beforehand the class to which a polygon belongs, it should always be numbered using the α-option. See pages 18-25 for more discussion.

Definitions and Page References for Flow Chart Quantities

IBND – Integral of the bivariate normal density function, page 2

P – Value of P-function for a polygon or an angular region, page 1

A – Value of A-function for a polygon, page 9, Appendix D

ICV – Program input parameter, page 25, Appendix F

IND – Program output error parameter, page 26, Appendix F

IOP – Program input accuracy parameter, page 25, Appendix F

W – Winding number, page 18

$\alpha_1, \alpha_2, \alpha_3, \alpha_4, \bar{R}^2/2$ – Program parameters which depend on IOP – Appendix E

$B = R^2/2$, page 28

$g_i = R/\sqrt{2} \cos\theta_i$, $i = 1, 2$, page 6

$h_i = R/\sqrt{2} \sin\theta_i$, $i = 1, 2$, page 6

$G = \frac{1}{2\sqrt{\pi}} (h_2 - h_1) - \frac{1}{2\pi} (g_2 h_2 - g_1 h_1)$, page 7

u, v, w, z – Defined in Flow Charts 2 and 4

a_{m-1} = Chebyshev coefficients for erfc (u), page 6

$\omega \equiv 7 \times 10^{-14}$; $\sigma \equiv 5 \times 10^{-14}$ (used in SORT I), Appendix E

$E(h) \equiv$ erfc (h), page 5

$\bar{E}(h) \equiv$ erf (h), page 28

Ω = Multiple of $\pm 2\pi$, page 20

$\Delta\theta = \tan^{-1} (\psi/\phi)$ page 5. (See Footnote 13, page 20)

$\psi \equiv vw - uz$

$\phi \equiv uw + vz$

$s = \sin \Delta\theta = 2\psi/D_1 D_2$, page 28

$D_1 = [2(w^2 + z^2)]^{1/2}$

$D_2 = [2(u^2 + v^2)]^{1/2}$

$c = (R^2/2) \cos \Delta\theta = g_1 g_2 + h_1 h_2$

Program Identification Number:

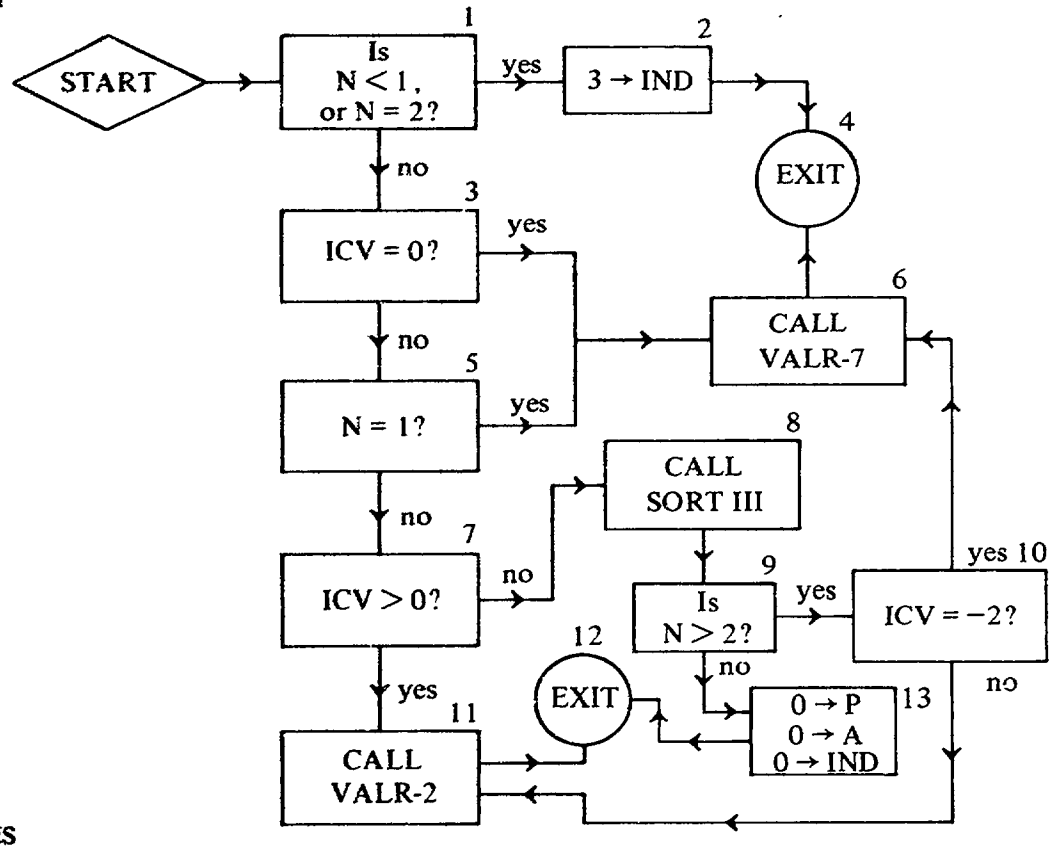
P-2	1	P-7	5
VALR-2	2	SORT I	6
SORT III	3	SORT II	7
VALR-7	4	SMP-7	8

FLOW CHART 1

P-2 MASTER PROGRAM FOR (P−B)

Input: x, y, N, ICV, IOP

Output: P, A, W, IND



BOXES

1, 3, 6: User wants P for a simple polygon or an $\bar{S}$ element with no SAR (s), (ICV = 0).

1, 3, 5, 6: User wants P for an angular region, (N = 1).

1, 3, 5, 7, 11: User wants P for an arbitrary polygon, (ICV > 0).

1, 3, 5, 7, 8, 9, 10, 11: User wants P for an arbitrary polygon, and wants to reduce number of erfc function computations by removing SCP, (ICV ≠ −2, ICV < 0).

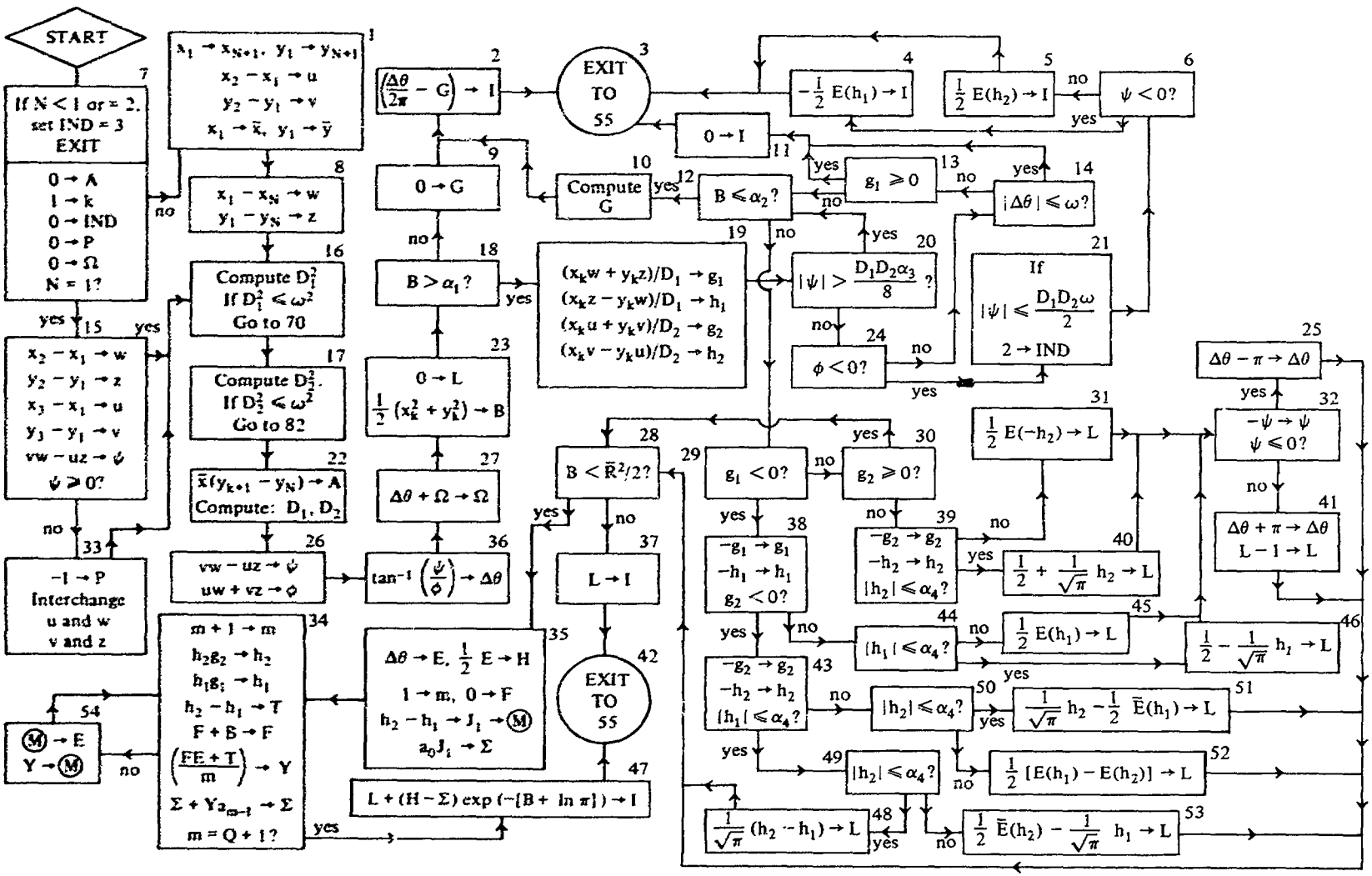
1, 3, 5, 7, 8, 9, 10, 6: User wants P for an $\bar{S}$ polygon which is specified with SAR(s), (ICV = −2).

FLOW CHART 2

VALR-2

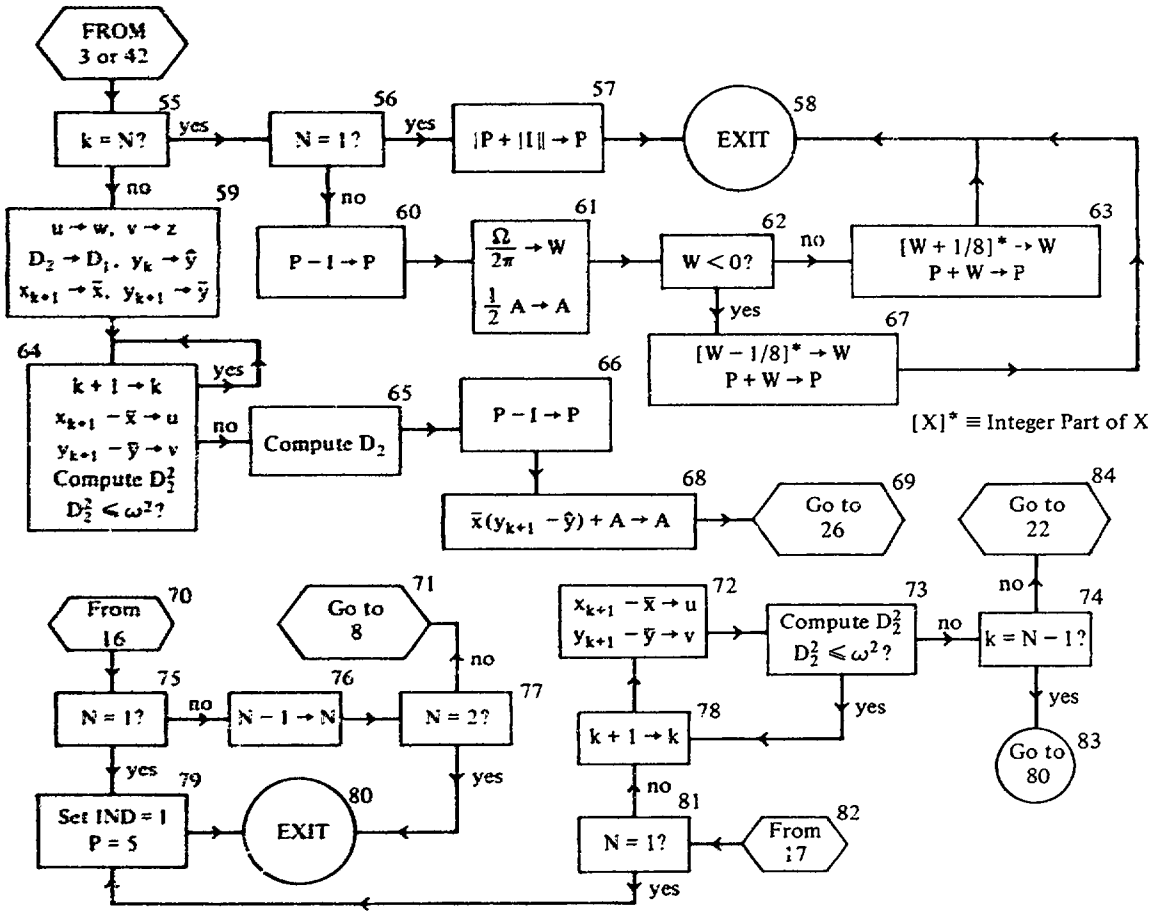
Input: N, $(x_k, y_k)_1^N$, IOP

Output: P, A, IND, W



FLOW CHART 2 (Continued)

VALR-2 (Continued)



FLOW CHART 3

SORT III

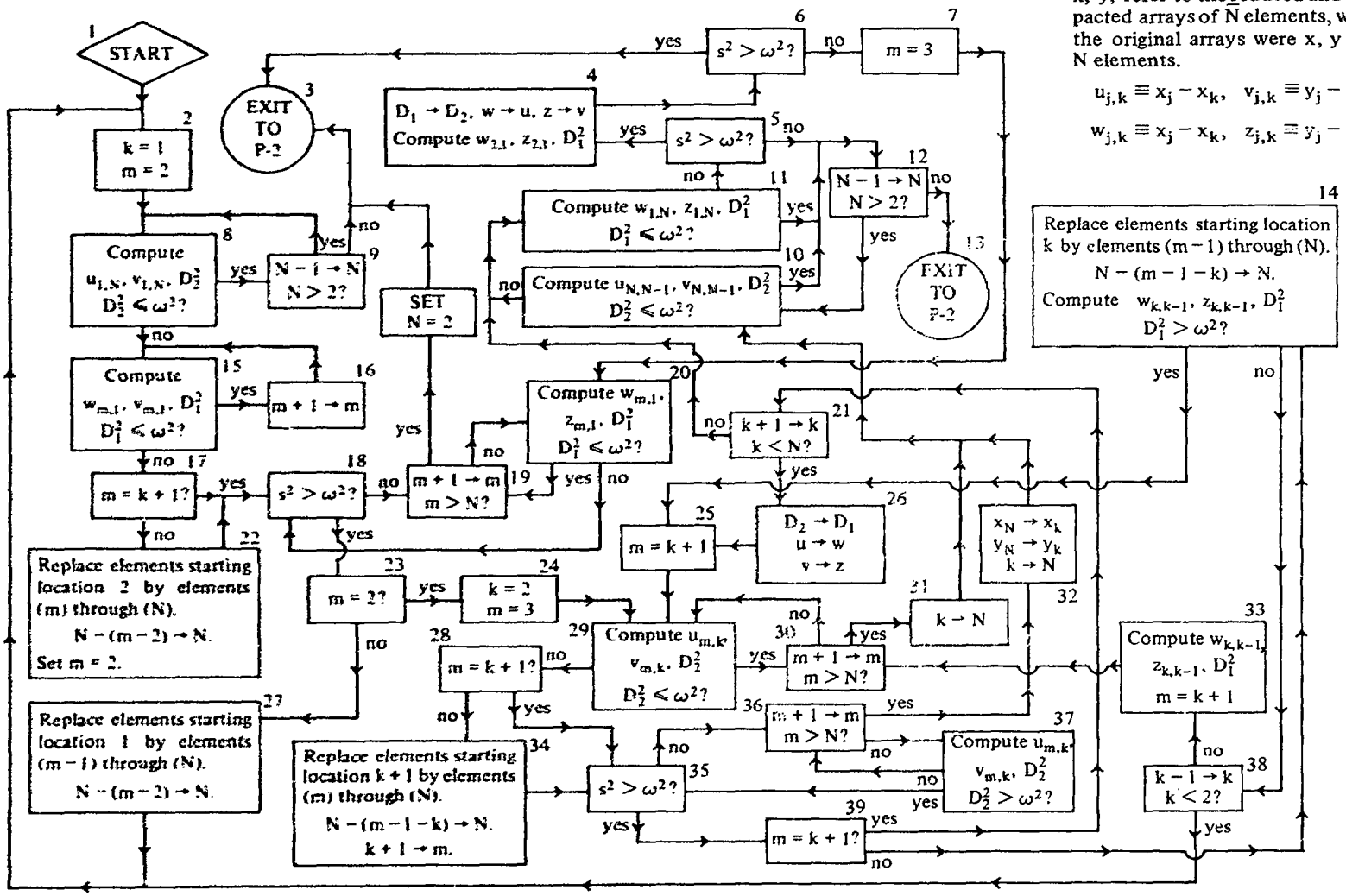
Input: x, y, N

Output: $\bar{x}$, $\bar{y}$, $\bar{N}$

$\bar{x}$, $\bar{y}$, refer to the reduced and compacted arrays of $\bar{N}$ elements, where the original arrays were x, y with N elements.

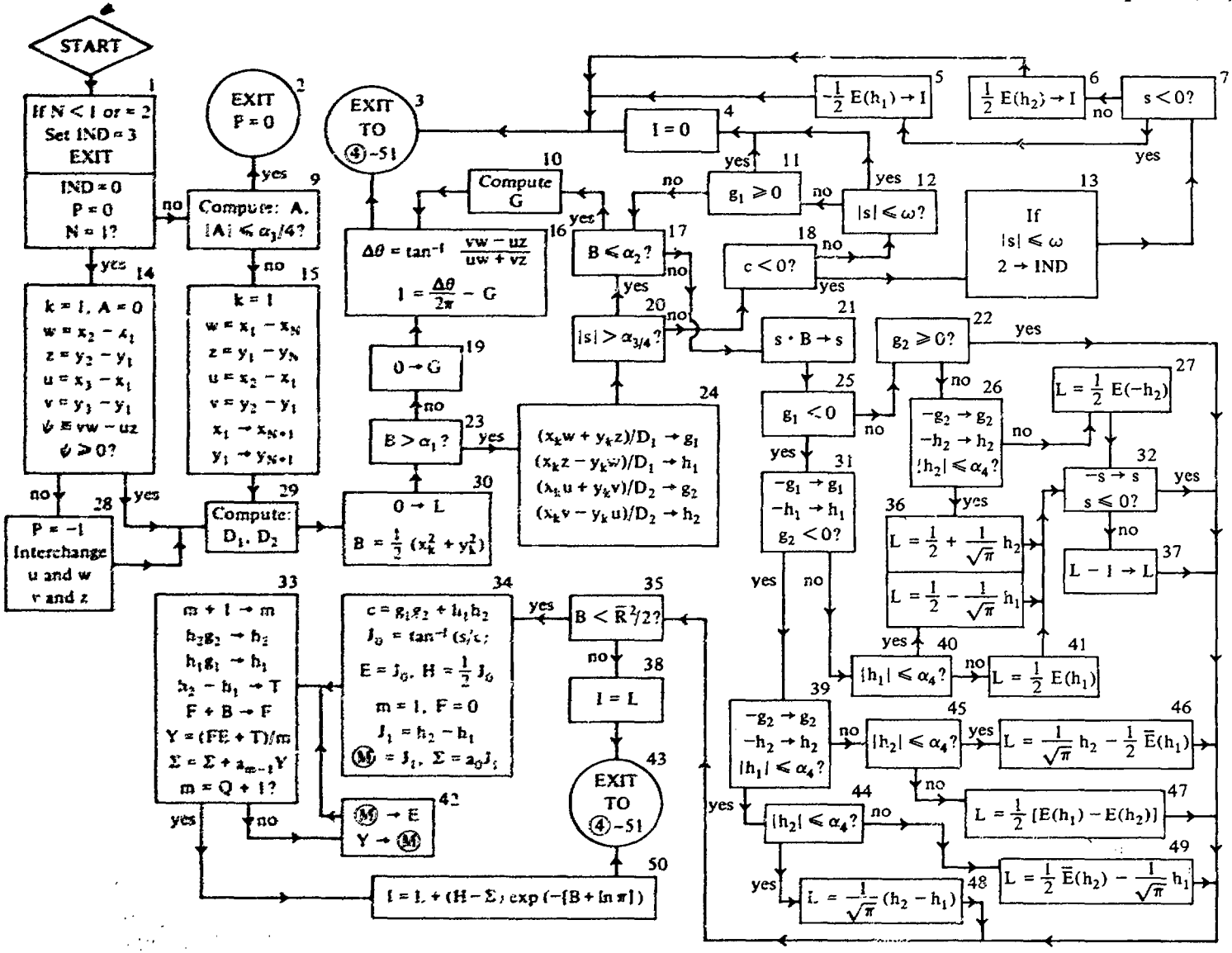
$u_{j,k} \equiv x_j - x_k$, $v_{j,k} \equiv y_j - y_k$

$w_{j,k} \equiv x_j - x_k$, $z_{j,k} \equiv y_j - y_k$



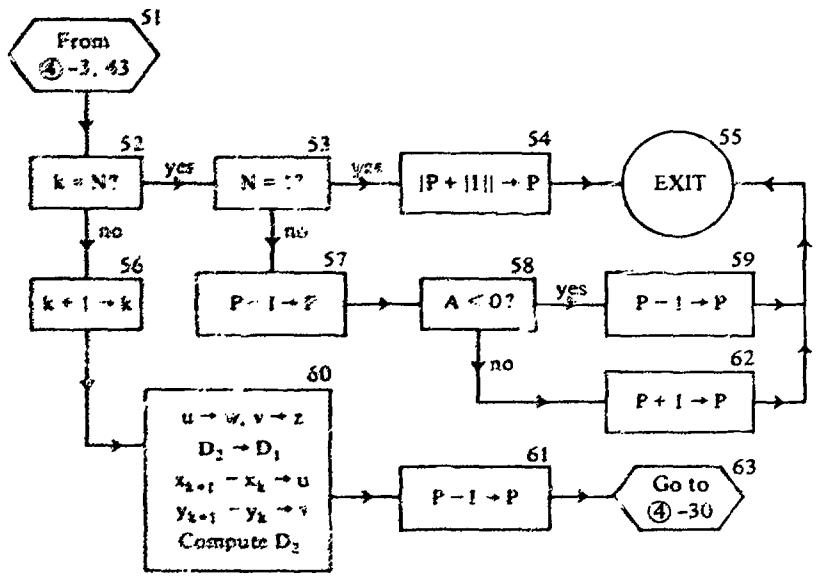
FLOW CHART 4

Input: $N, (x_k, y_k)_1^N$, IOP — VALR-7 — Output: P, A, IND



FLOW CHART 4 (Continued)

VALR-7



Note: $x_{N+1} = x_1$, $y_{N+1} = y_1$.

VI. NUMERICAL RESULTS

In this section, a variety of polygon elements are shown through Figures 30–58. Each element has its vertices and relevant edge intersections numbered sequentially, from 1 to N, in the order it is generated.[19] The numerals are not always placed optimally for viewing; nevertheless the xy-coordinates to which a numeral corresponds can always be decided. A numeral (node) is generally, but not always, located below and slightly to the left of the xy-point to which it belongs. The xy-coordinate values are always rational numbers that can be read from the figures using a ruler graduated in tenths. Each figure lists the following information:

ICV, P, A, Classification (__, __), N.

All the computations were performed using P-2 as the master routine. If there is a most efficient way to compute P, ICV is assigned one value. If there are two ways which may be equally good, or if it is difficult to decide which is better, then ICV is assigned two values. For example, Figure 37 shows a simple polygon, so ICV = 0. A glance at the P-2 flow chart, page 40, confirms that P-2 calls VALR-7 to compute P(S). In Figure 32, an SI element is shown with a PAR as well. We have ICV = 1, −1, which, from P-2, calls first VALR-2 to find P and subsequently VALR-2 preceded by SORT III to obtain the same result. The rounded value of P given in each figure is correct to the number of digits shown. The value of A is given next. The classification of an element (according to the T-construction, page 15) follows and is designated by S, $\bar{S}$, or SI. The two blanks, in parentheses, following the classification, as noted above, are used to denote the *computed* winding number W. It is only listed if VALR-2 is called. Thus, for Figure 32 two winding numbers are listed since ICV = 1, −1 which both use VALR-2. In the first case the arctangent subroutine yields π for the angular measure of the PAR at (11) instead of $-\pi$ according to the T-construction. This results in a computed winding number of one instead of zero. See pages 22 and 23. Note there is also a PAR at (19); however in this case W is not affected since π is its measure according to the T-construction. In Figure 37, W = 1, since the element is PO, but it is not listed because VALR-2 was not used to compute P for this element. Finally, N is listed which refers to the number of points used to define the configuration as shown.

By our T construction, page 15, an element of $(\bar{S})$ has a winding number W of ±1. However, because of the range of the arctangent routine, page 22, this need not be the case computationally as for example in Figures 30 and 31, see page 33 also. The element of Figure 30 (31) is in $(\bar{S})$ according to the T-construction, but must be considered SI for computations. Thus P is computed using VALR-2 with a computed W of 6 (2), using ICV = 1. Then P is computed again using VALR-7 preceded by SORT III, (ICV = −2).

In Figure 34, we have the case of a simple polygon in the form of a triple spiral. Another simple polygon is shown in Figure 37. Figures 30, 31, 35, 36, 38, 39, 42, 43, 44, 45, 48, 49, 51, 55, 56 contain elements in $(\bar{S})$. The remaining figures: 32, 33, 40, 41, 46, 47, 50, 52, 53, 54, 57, 58 display SI elements.

---

[19] It should be understood that an additional node (N + 1) is located at MN(1).

It is our objective in presenting these figures, that there is enough variety to resolve for the reader any remaining uncertainties regarding the robustness of (P−B), how a polygon is specified and classified, and how a winding number is determined. Finally, for completeness and as a further clarification of the role of exterior angular regions, we tabulate on page 48 a detailed listing of the values of $P(a_k)$, k = 1, 2, ..., N, that are needed to compute $P(\Pi)$ for the element displayed in Figure 33.

This polygon is interesting in its own right, since $P(\Pi)$ represents, here, the probability for an event, governed by a bivariate normal distribution, occurring in S_1 and/or S_2, where S_1 = (1, 2, 3, 4, 5, 6, 7), S_2 = (8, 9, 10, 11, 12). From probability theory, we can write

$$P(\Pi) = P(S_1 \cup S_2) = P(S_1) + P(S_2) - |P(S_1 \cap S_2)|, \tag{54}$$

where $\cup$, $\cap$ denote union and intersection symbols for sets and $S_1 \cap S_2$ = (8, 13, 14, 15, 16, 17, 18) which is NO. The values of $P(S_1)$, $P(S_2)$, $P(S_1 \cap S_2)$ are given in Appendix A, where $P(\Pi)$ is found by decomposing Π with SORT I, which is based on (P−A). See page (A-14).

The tabulation, page 48, lists in the first column the value of k for the k^{th} node of Π. The second and third columns list the x and y coordinates, respectively, for each node in the order Π is generated (the numbering used is under the β-option, see page 24). The fourth column lists the value of $P(a_k)$, for each k, obtained from VALR-2 with ICV = 1 and IOP = 3 in P-2. The corresponding angular measures $\Delta\theta_k$ for each a_k are given in the fifth column. In the sixth and seventh columns $P(a_k)$ and $\Delta\theta_k$ are listed for ICV = −1, i.e., Π is treated by SORT III first, and then values in the sixth and seventh columns are computed, with IOP = 3, from VALR-2. (Note, IOP = 3 implies an accuracy of 9-decimal digits for each $P(a_k)$.) The reduced polygon as a result of SORT III treating Π is given by (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 14, 15, 16, 17, 18, 19). SORT III has deleted (12) since (11), (12), (13) are SCP. Then it drops (13), because (11) and (13) are now SDP. Then it removes (11) because (10), (11), (14) are now SCP. The columns 4, 5, 6, 7 are summed at the bottom of the tabulation. Note that W = 2 for both situations. This is not always the case. The primary circuits (see page 18) can be gleaned from the figure. For ICV = 1, the circuits are given by $C_P(\Pi_1)$ = (16, 4, 5, 6, 7, 8, 9, 16) with W_1 = 1, $C_P(\Pi_2)$ = (1, 2, 3, 16, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19)[20] with W_2 = 1. Hence W = W_1 + W_2 = 2. For ICV = −1, the primary circuits are given by $C_P(\Pi_1)$ = (16, 4, 5, 6, 7, 8, 9, 16) with W_1 = 1, $C_P(\Pi_2)$ = (1, 2, 3, 16, 10, 14, 15, 16, 17, 18, 19), with W_2 = 1. Hence again W = 2. A π-angular region, PAR, occurs at k = 12 initially. Hence with ICV = 1 an erfc calculation is required at k = 12, namely 1/2 erfc (h_2) = 1/2 erfc (−1) = .92135 03965. If ICV = −1, then point (12) is deleted and the erfc computation, at the expense of using SORT III, is not necessary.

The time of computation per angular region is given in Appendix E.

All of the numerical results in this report, as well as many that are not given, were checked by an independent procedure. It consists of decomposing Π, regardless of its class, into a set of triangles $\{\Delta_j\}$. The triangle Δ_j has vertices (1), (j), (j + 1). With j = 2, 3, ..., N − 1 we have

[20]The order of the nodes appears unusual because of the use of the β-option numbering scheme.

$P(\Pi) = \Sigma_{j=2}^{N-1} P(\Delta_j)$. The proof of this result follows the lines of proof given for A in Appendix D; it is not given here.

The value of $P(\Delta_j)$ is computed from a routine we developed which uses Drezner's scheme [2, page 18] for evaluating P over an angular region. His method is much slower than ours but gives very good accuracy. It is described in [2].

In this checkout program, which is listed in Appendix G, N-2 triangles Δ_j are obtained for each II, and consequently P is required for 3(N-2) angular regions.

TABULATION OF RESULTS FOR FIGURE 33 USING P-2. ($\epsilon = 5 \times 10^{-10}$)

k	x	y	ICV = 1, $P(a_k)$	ICV = 1, $\Delta\theta_k$	ICV = −1, $P(a_k)$	ICV = −1, $\Delta\theta_k$
1	−3	0	1.6803 81909 (−2)	3π/4	1.6803 81909 (−2)	3π/4
2	0	−3	7.8697 69659 (−3)	1.4288 99272 (0)	7.8697 69659 (−3)	1.4288 99272 (0)
3	4	0	4.9999 79129 (−1)	2.4980 91545 (0)	4.9999 79129 (−1)	2.4980 91545 (0)
4	0	0	−3/8	−3π/4	−3/8	−3π/4
5	2	2	3.0600 67674 (−3)	1.8925 46881 (0)	3.0600 67674 (−3)	1.8925 46881 (0)
6	0	3	1.6551 27610 (−2)	1.2490 45772 (0)	1.6551 27610 (−2)	1.2490 45772 (0)
7	−3	0	4.9985 63923 (−1)	3π/4	4.9985 63923 (−1)	3π/4
8	−2	0	−3.1610 42924 (−1)	−4.6364 76090 (−1)	−3.1610 42924 (−1)	−4.6364 76090 (−1)
9	0	−1	9.5914 28393 (−2)	9.2729 52180 (−1)	9.5914 28393 (−2)	9.2729 52180 (−1)
10	4	1	1.5865 44813 (−1)	2.6779 45045 (0)	1.5865 44813 (−1)	2.6779 45045 (0)
11	−1	1	2.6739 05696 (−2)	π/4		
12	−2	0	9.2135 03965 (−1)	π		
13	−1	1	−1.0674 47074 (−1)	−π/4		
14	1	1	−4.8741 42552 (−1)	−3π/4	3.5393 04909 (−1)	π/4
15	0	0	3/8	3π/4	3/8	3π/4
16	2	0	−1.8389 57076 (−1)	−2.6779 45045 (0)	−1.8389 57076 (−1)	−2.6779 45045 (0)
17	0	−1	−9.5914 28393 (−2)	−9.2729 52180 (−1)	−9.5914 28393 (−2)	−9.2729 52180 (−1)
18	−2	0	1.6509 77199 (−3)	4.6364 76090 (−1)	1.6509 77199 (−3)	4.6364 76090 (−1)
$P(\Pi) = 2 - \Sigma P(a_k) = 0.94162$ 48130				$[\Sigma \Delta\theta_k/2\pi] = W = 2$	$P(\Pi) = 0.94162$ 48129	$[\Sigma \Delta\theta_k/2\pi] = W = 2$

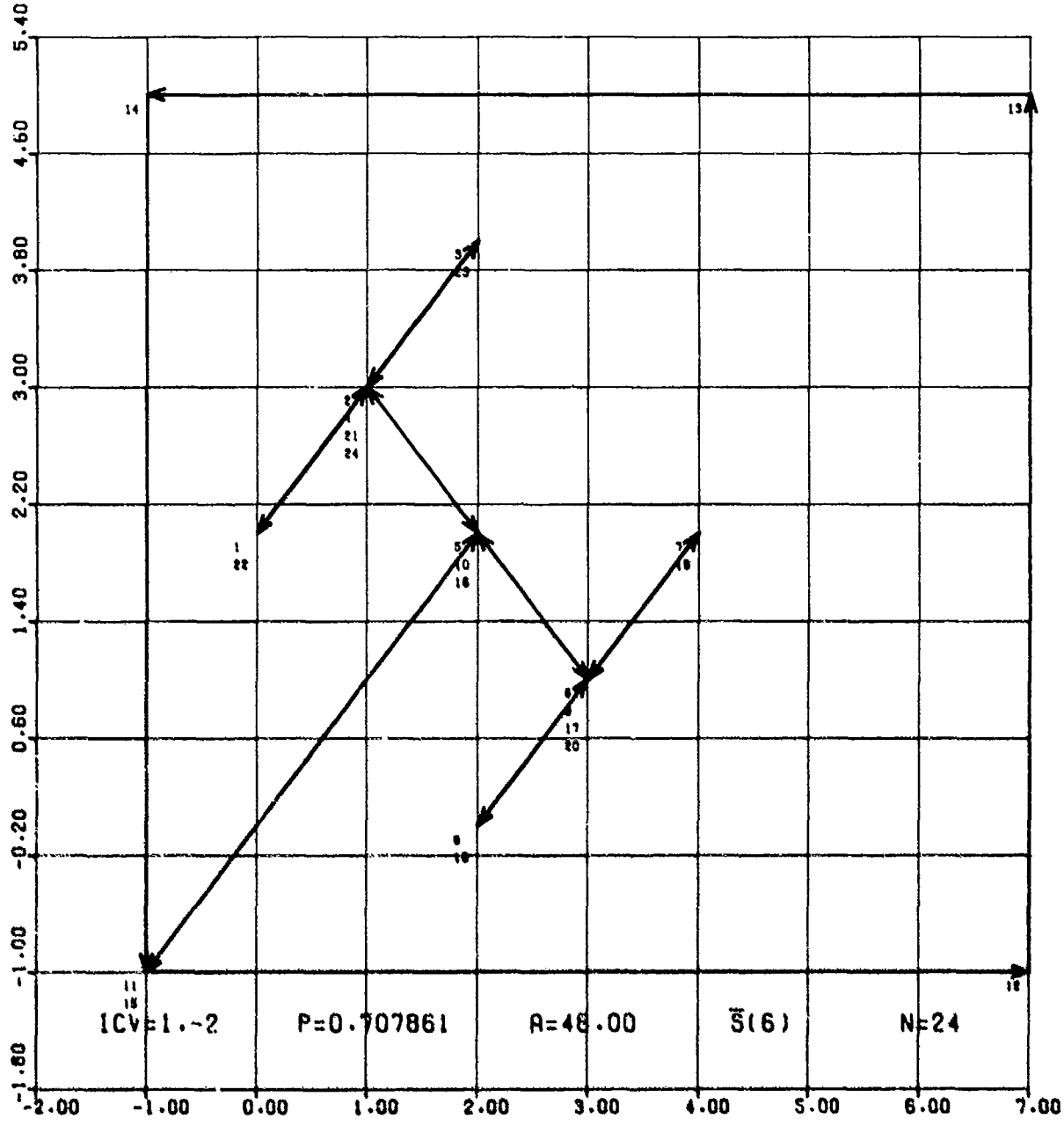

5.40
4.60
3.80
3.00
2.20
1.40
0.60
-0.20
-1.00
-1.80
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
5.00
6.00
7.00
14
13
1
22
21
24
5
10
16
17
20
9
19
18
ICV=1.-2
P=0.707861
A=48.00
S(6)
N=24

Figure 30

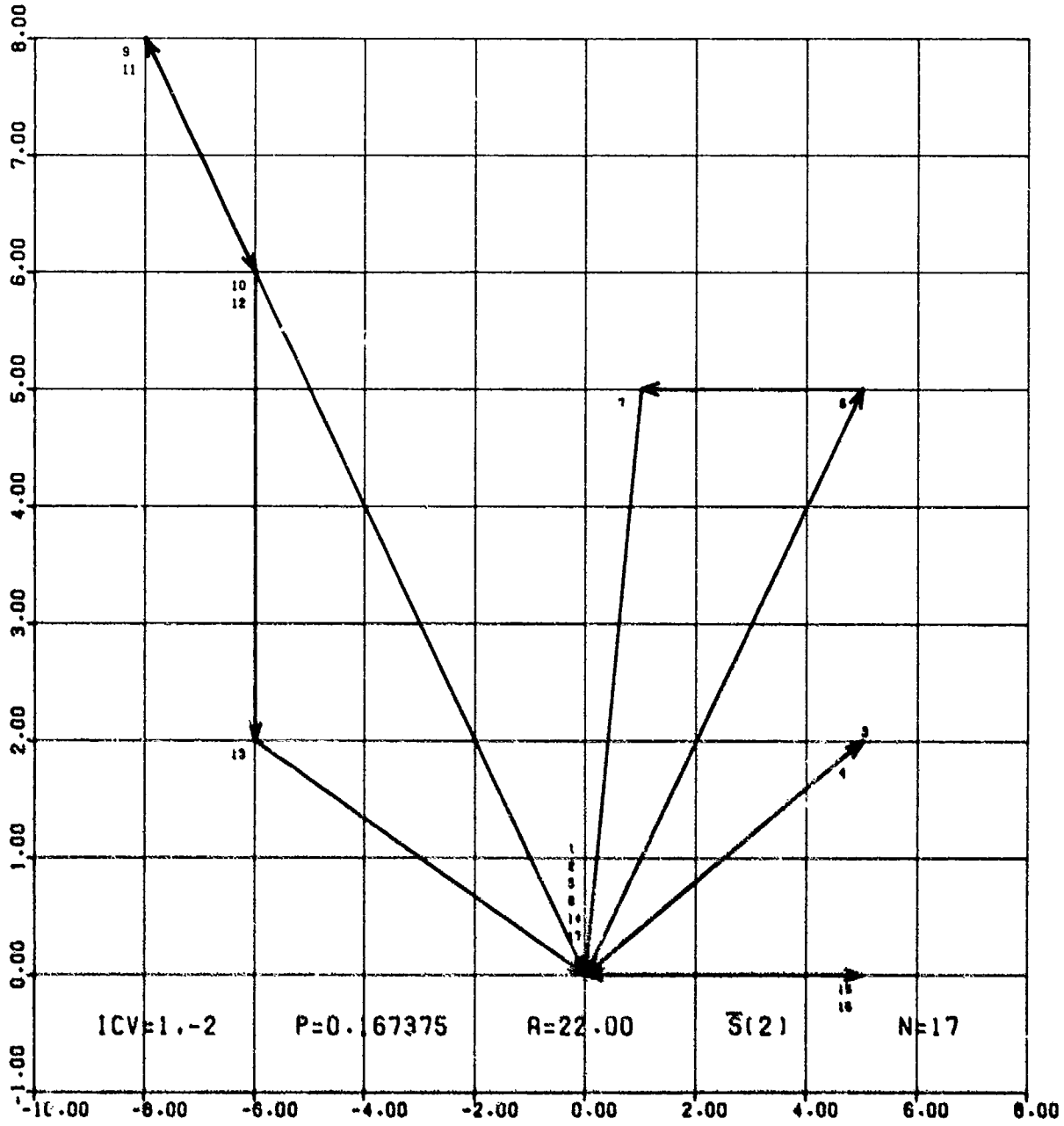

8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
-1.00
-10.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
ICV=1,-2
P=0.167375
R=22.00
S(2)
N=17

Figure 31

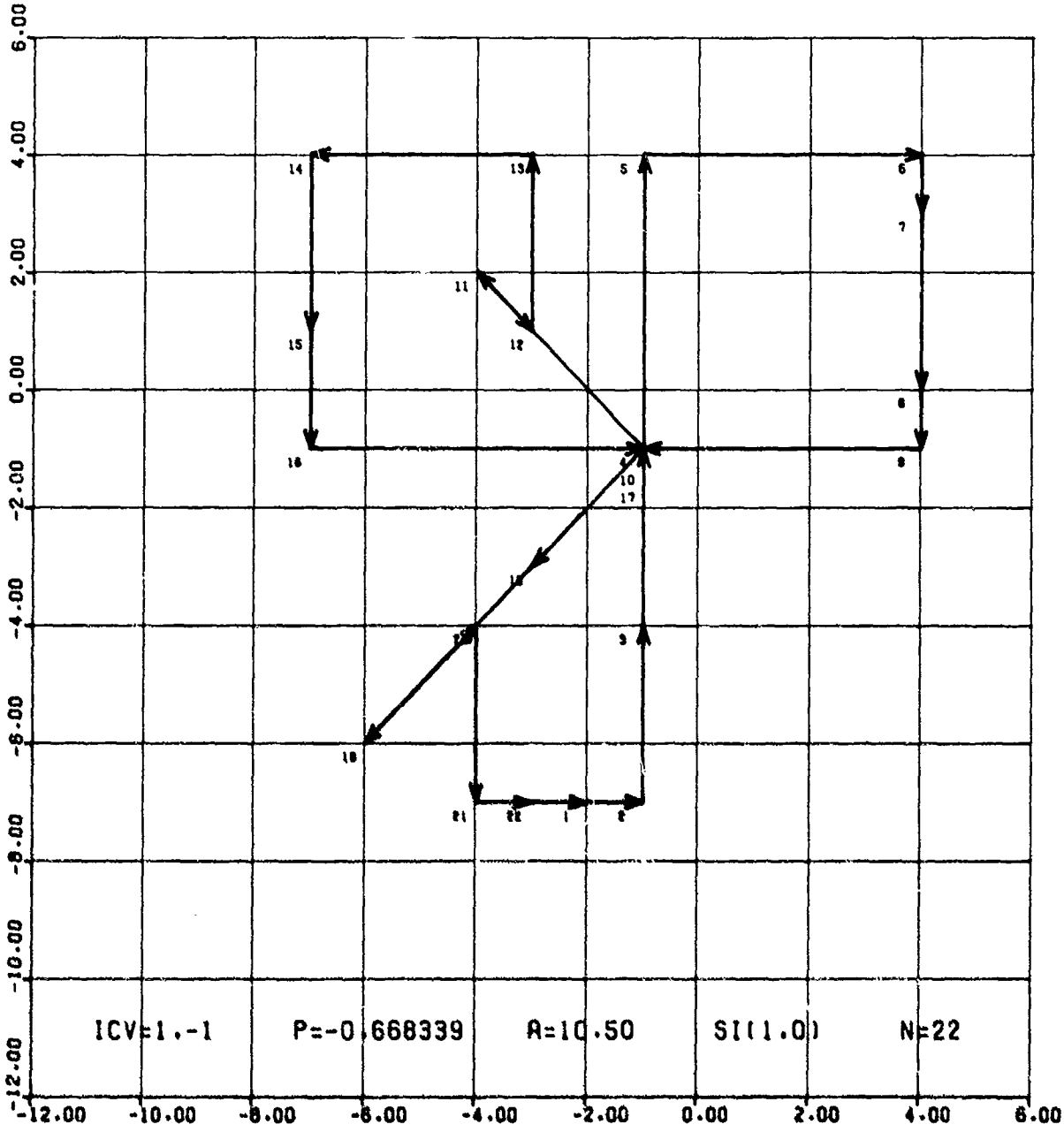

6.00
4.00
2.00
0.00
-2.00
-4.00
-6.00
-8.00
-10.00
-12.00
-12.00
-10.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
ICV=1.-1
P=-0.668339
A=10.50
SI(1.0)
N=22

Figure 32

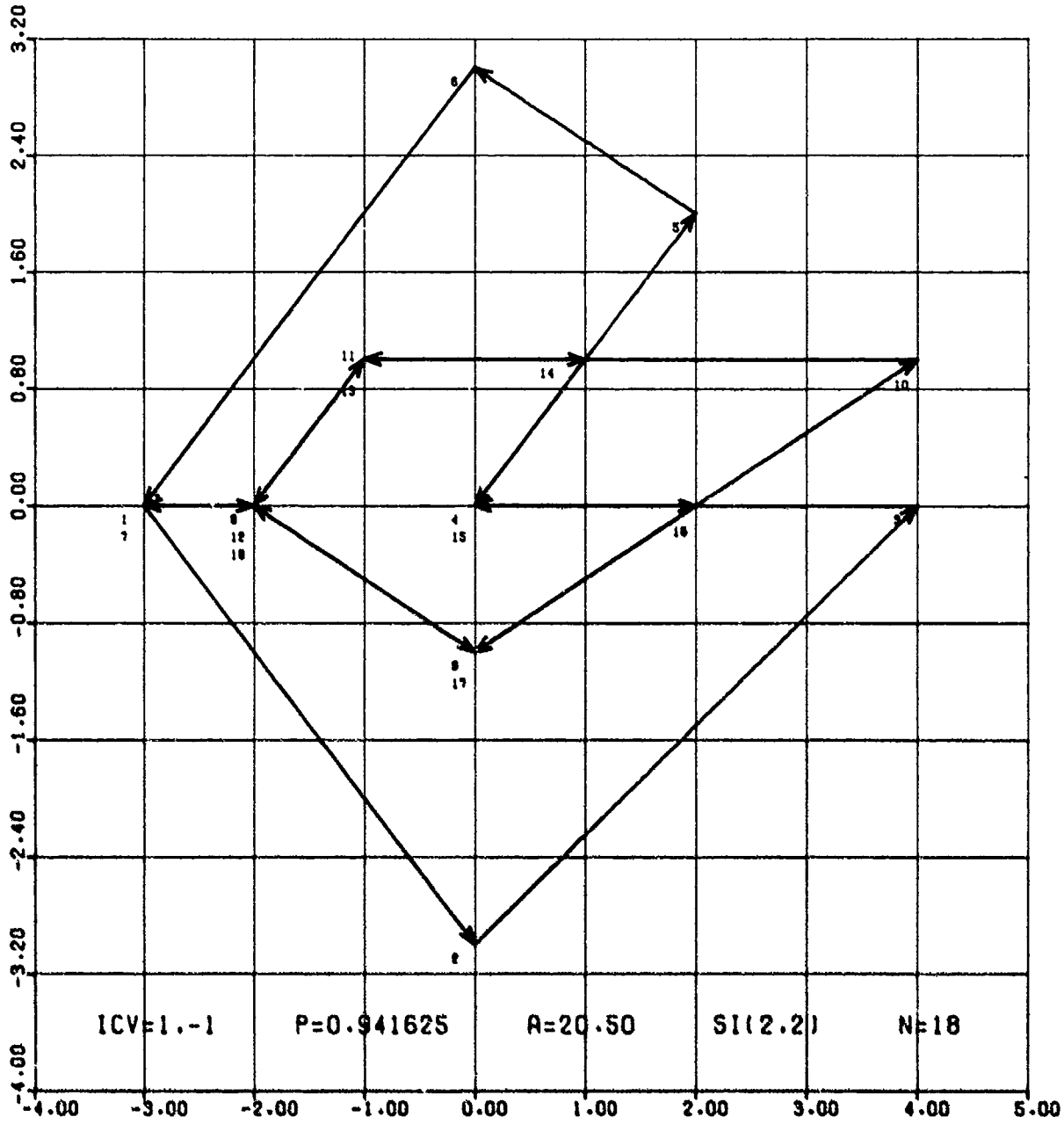

3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-4.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
5.00
ICV=1.-1
P=0.941625
A=20.50
SI(2.2)
N=18

Figure 33

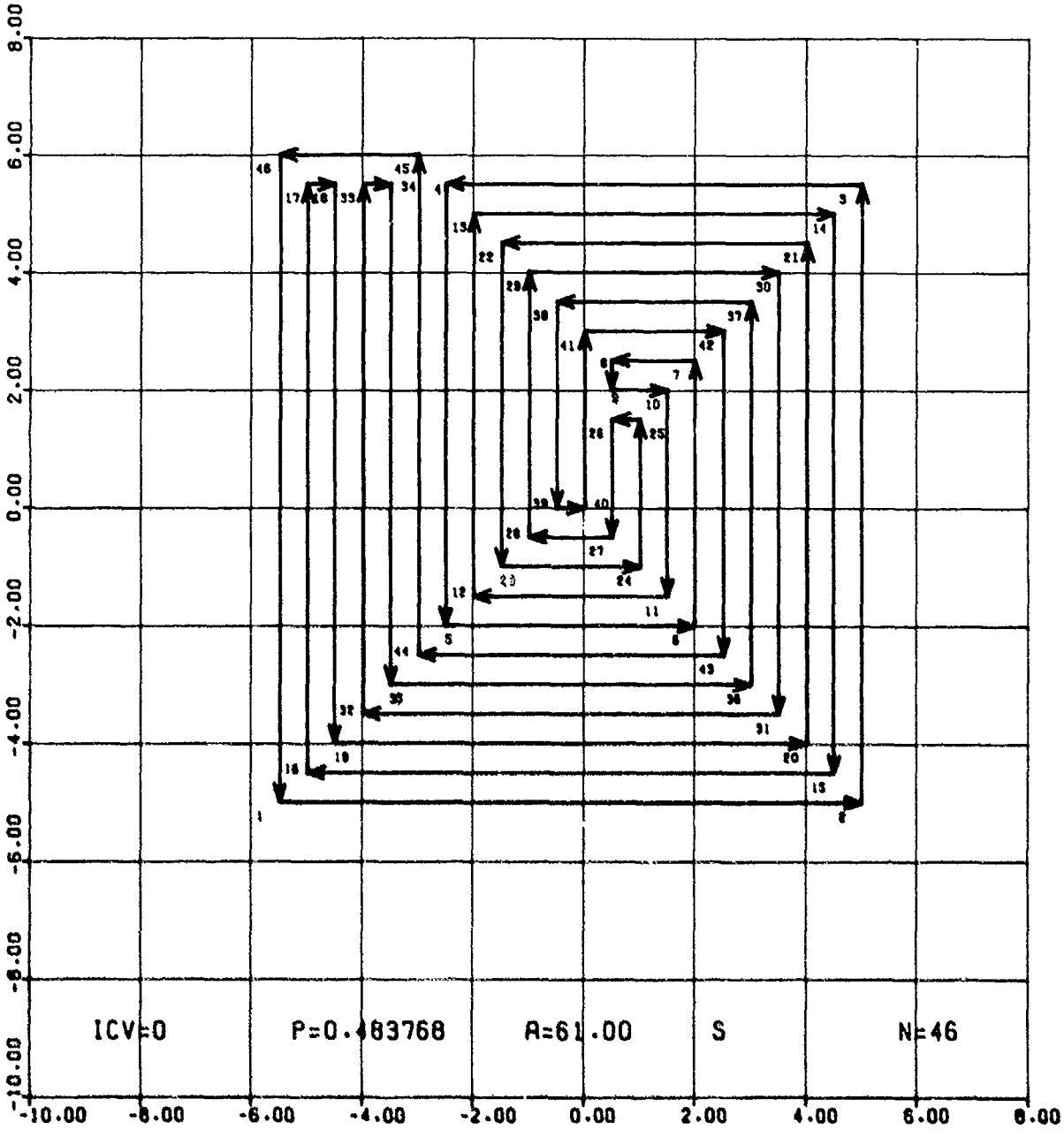

8.00
6.00
4.00
2.00
0.00
-2.00
-4.00
-6.00
-8.00
-10.00
-10.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
ICV=0
P=0.483768
A=61.00
S
N=46

Figure 34

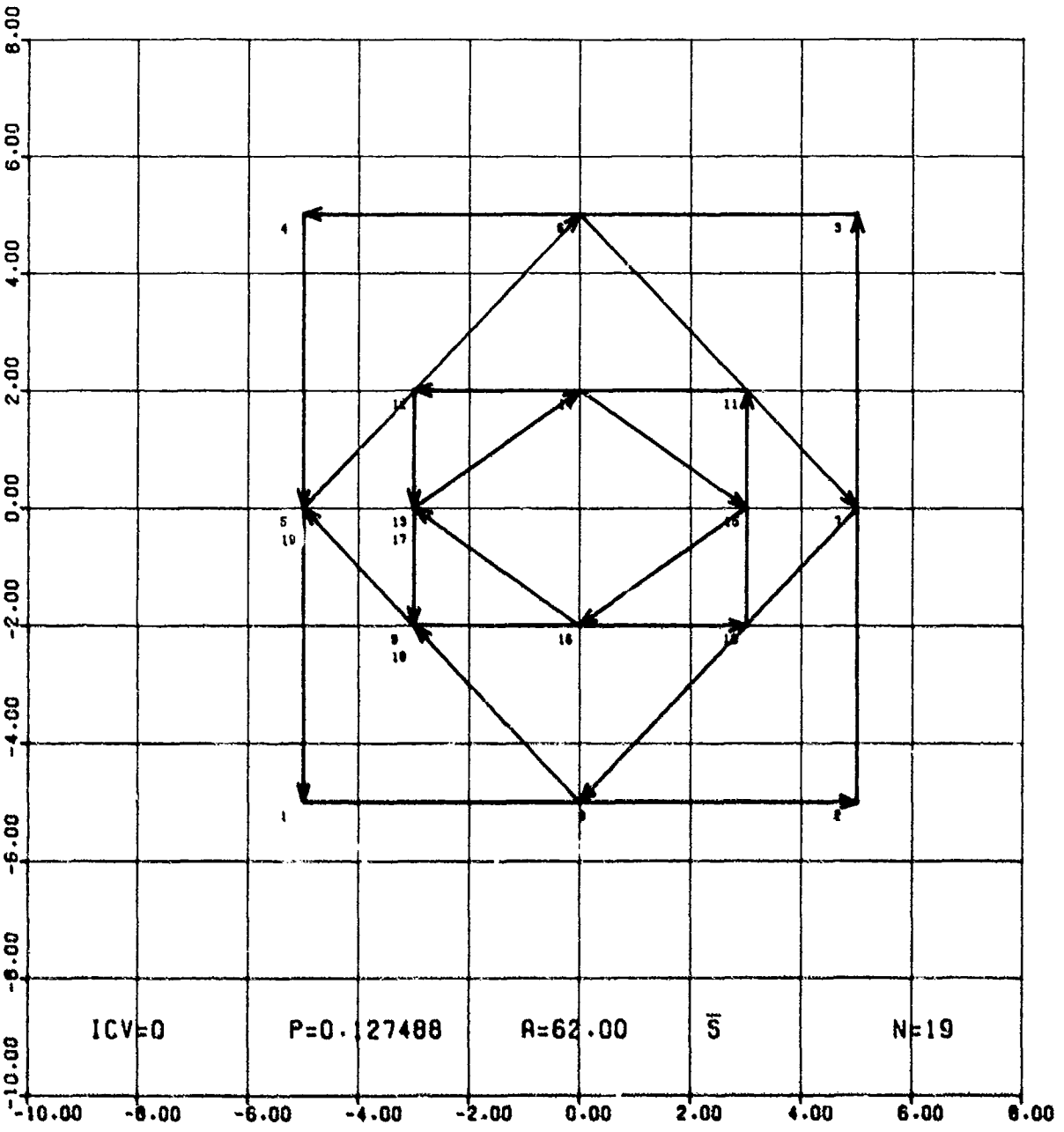

8.00
6.00
4.00
2.00
0.00
-2.00
-4.00
-6.00
-8.00
-10.00
-10.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
ICV=0
P=0.127488
A=62.00
S
N=19

Figure 35

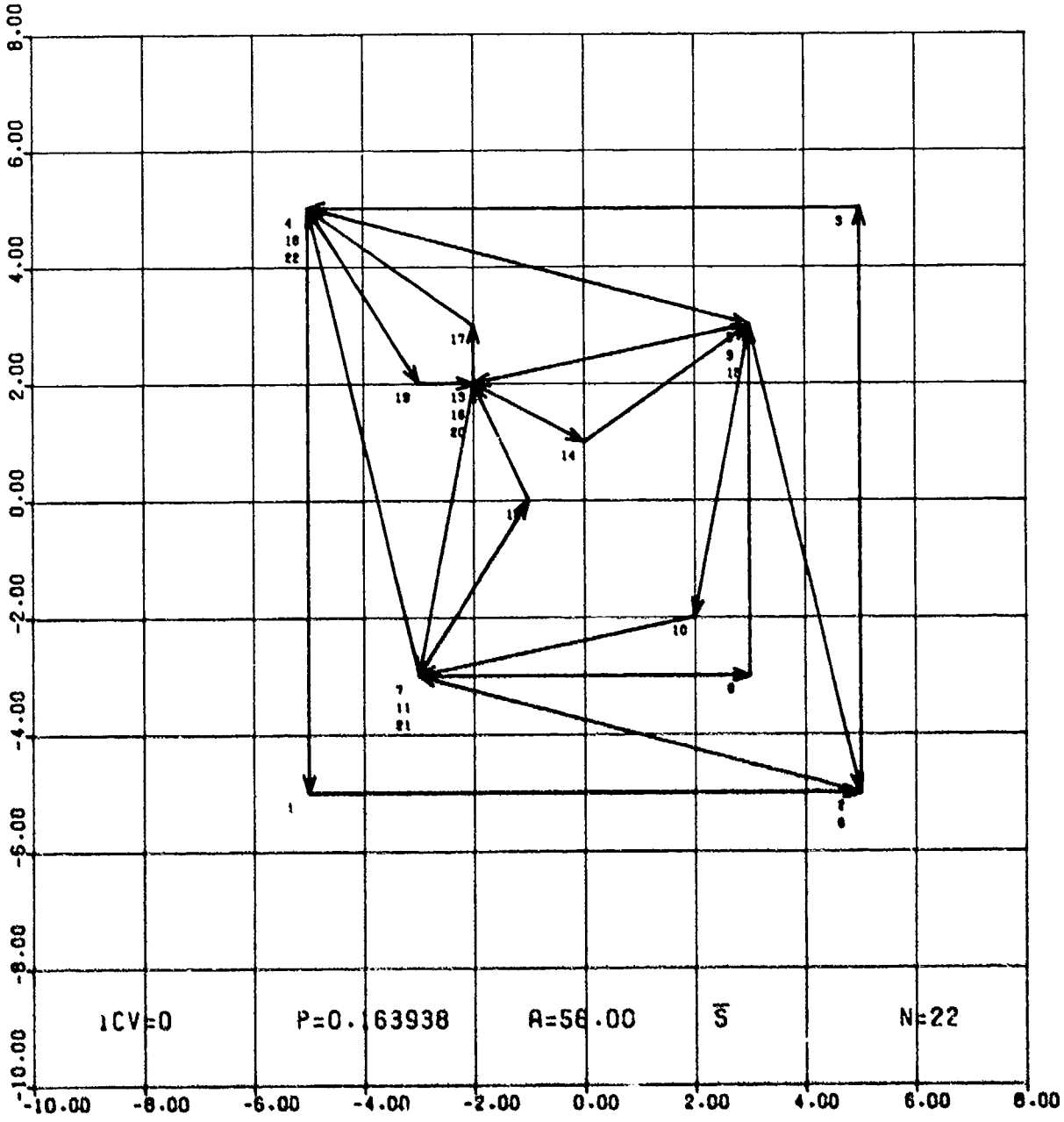

8.00
6.00
4.00
2.00
0.00
-2.00
-4.00
-6.00
-8.00
-10.00
-10.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
ICV=0
P=0.163938
A=56.00
S̄
N=22

Figure 36

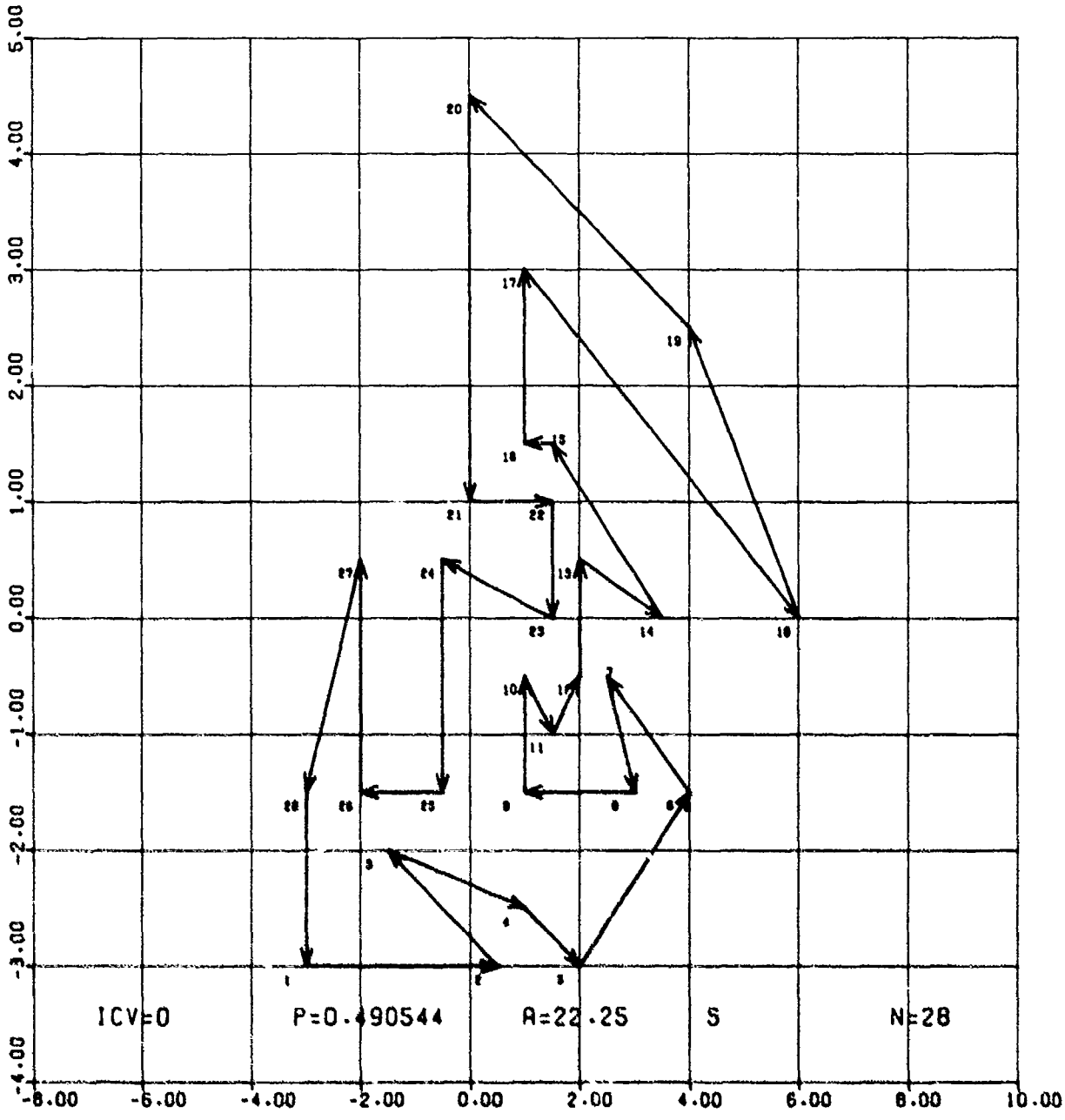

ICV=0
P=0.490544
A=22.25
S
N=28
5.00
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
10.00

Figure 37

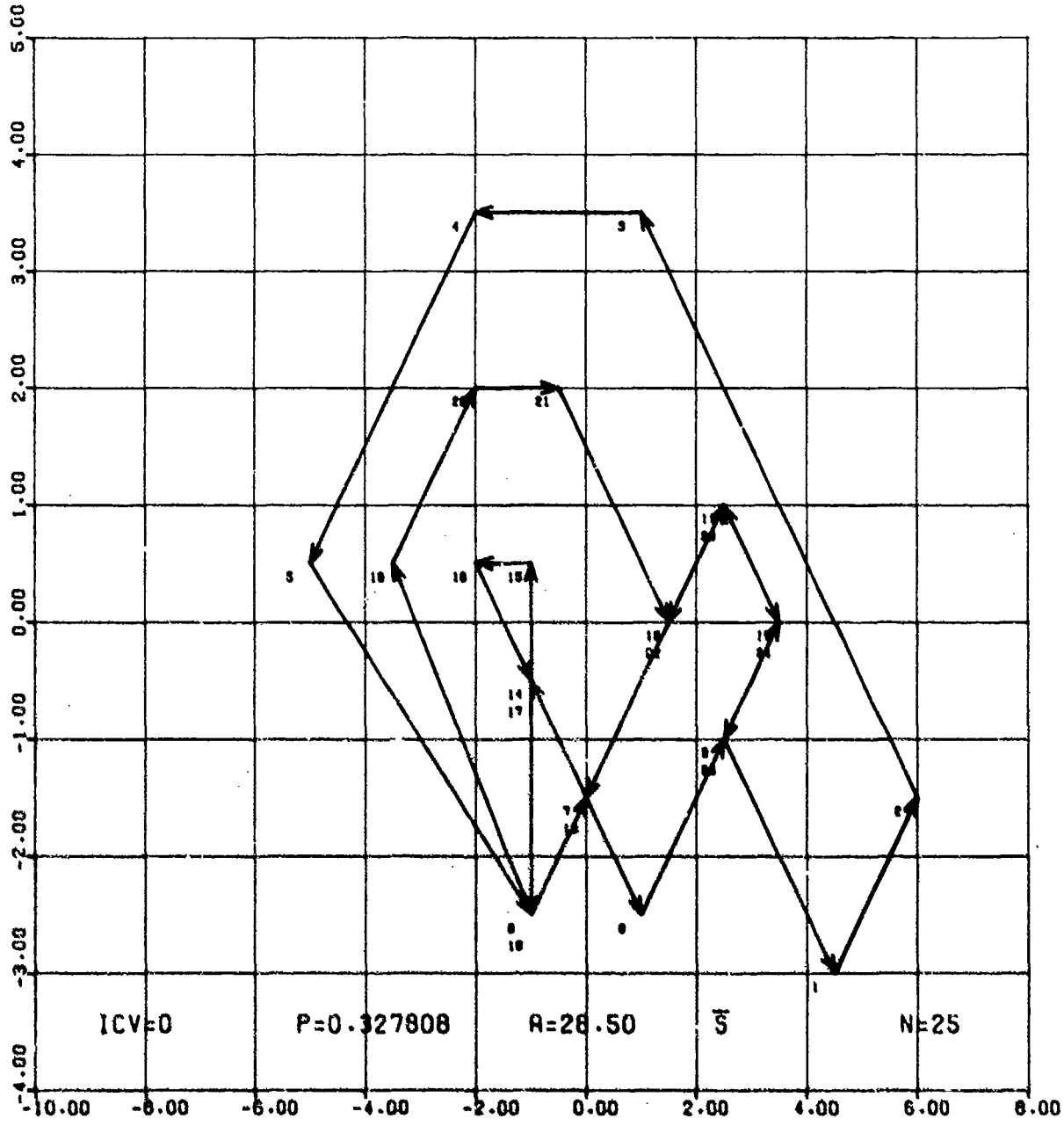

5.00
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-10.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
ICV=0
P=0.327808
A=28.50
S
N=25

Figure 38

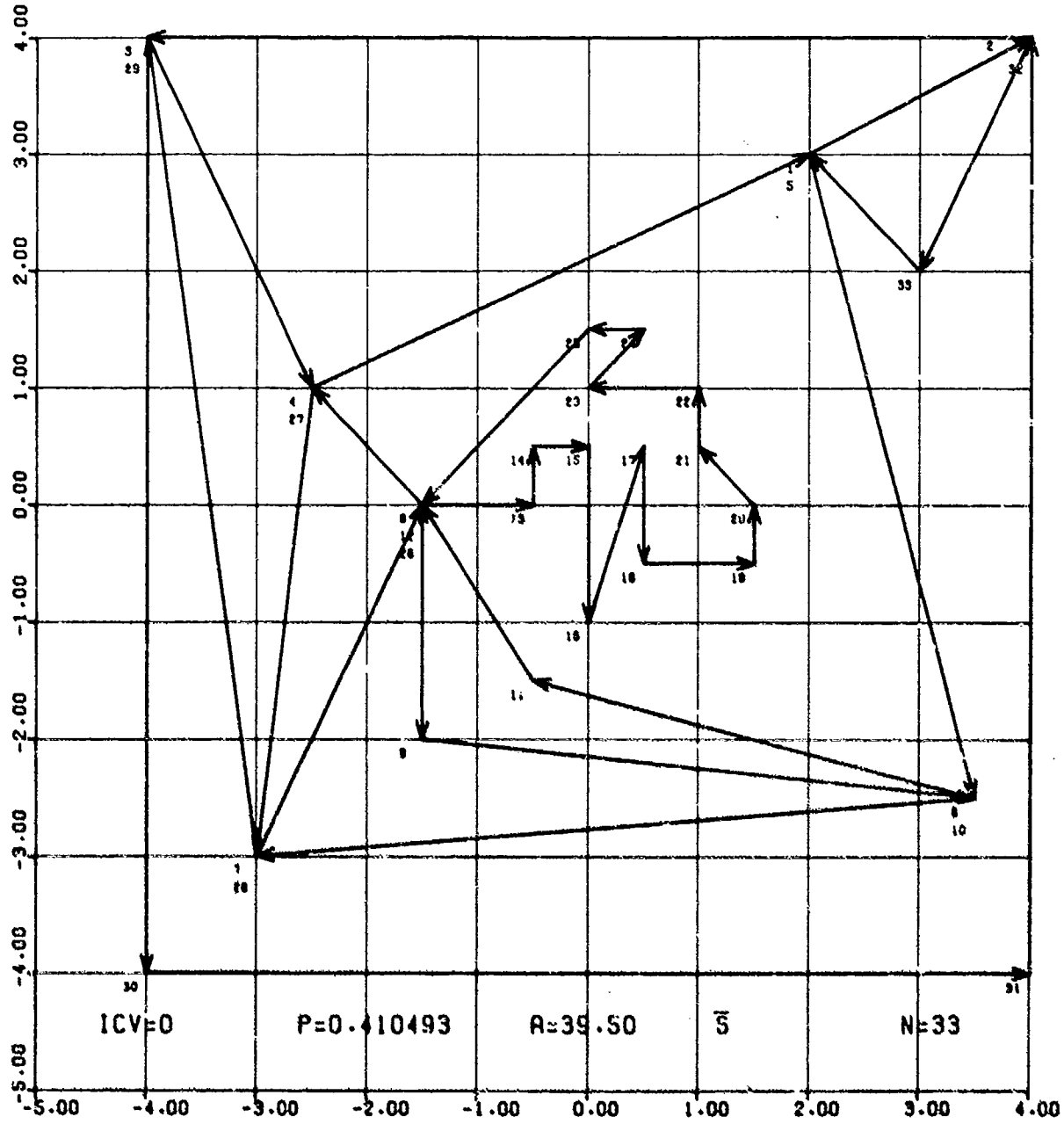

4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-5.00
-5.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
ICV=0
P=0.410493
A=39.50
S
N=33

Figure 39

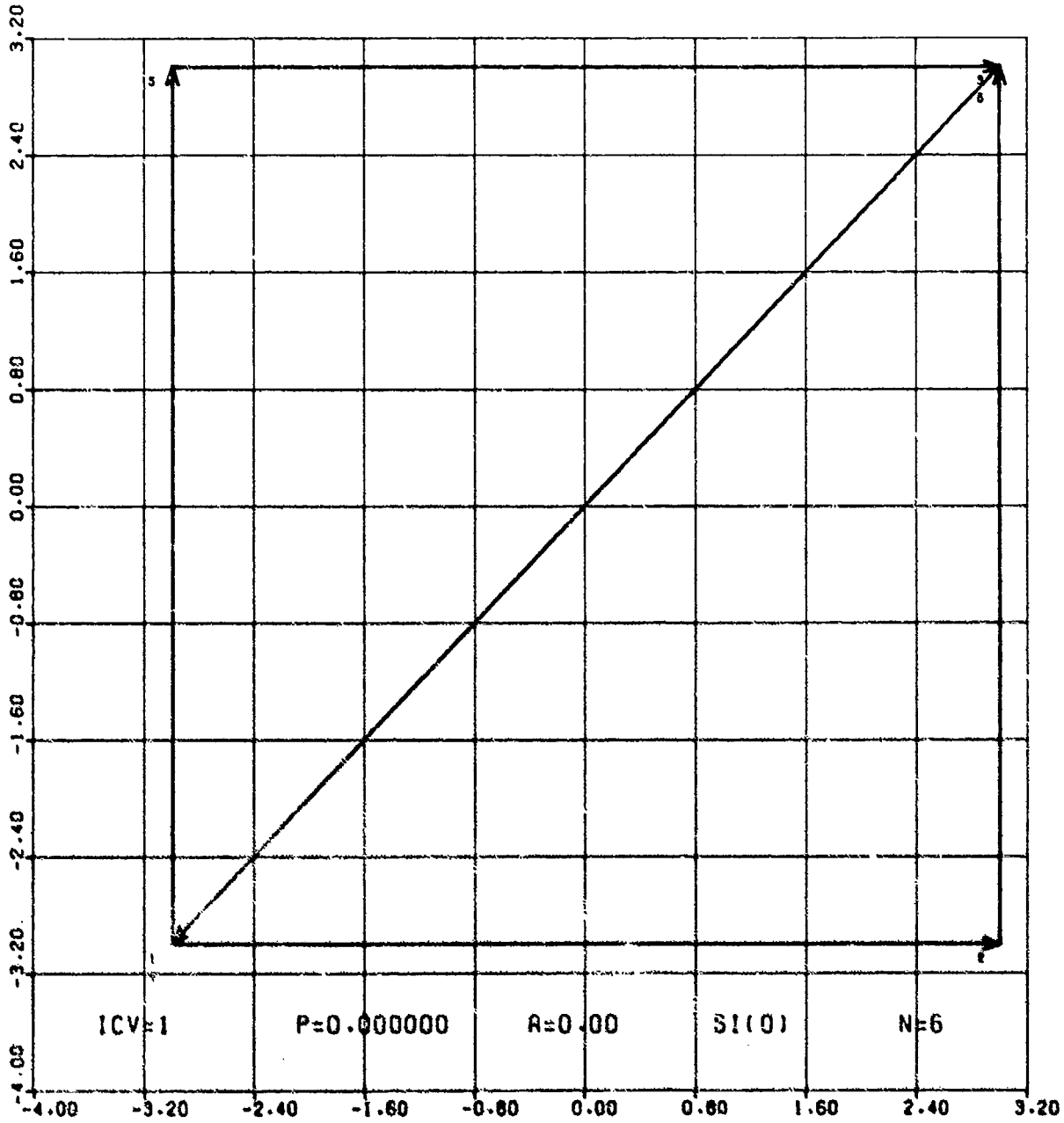

3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-4.00
-4.00
-3.20
-2.40
-1.60
-0.80
0.00
0.80
1.60
2.40
3.20
ICV=1
P=0.000000
A=0.00
S!(0)
N=6

Figure 40

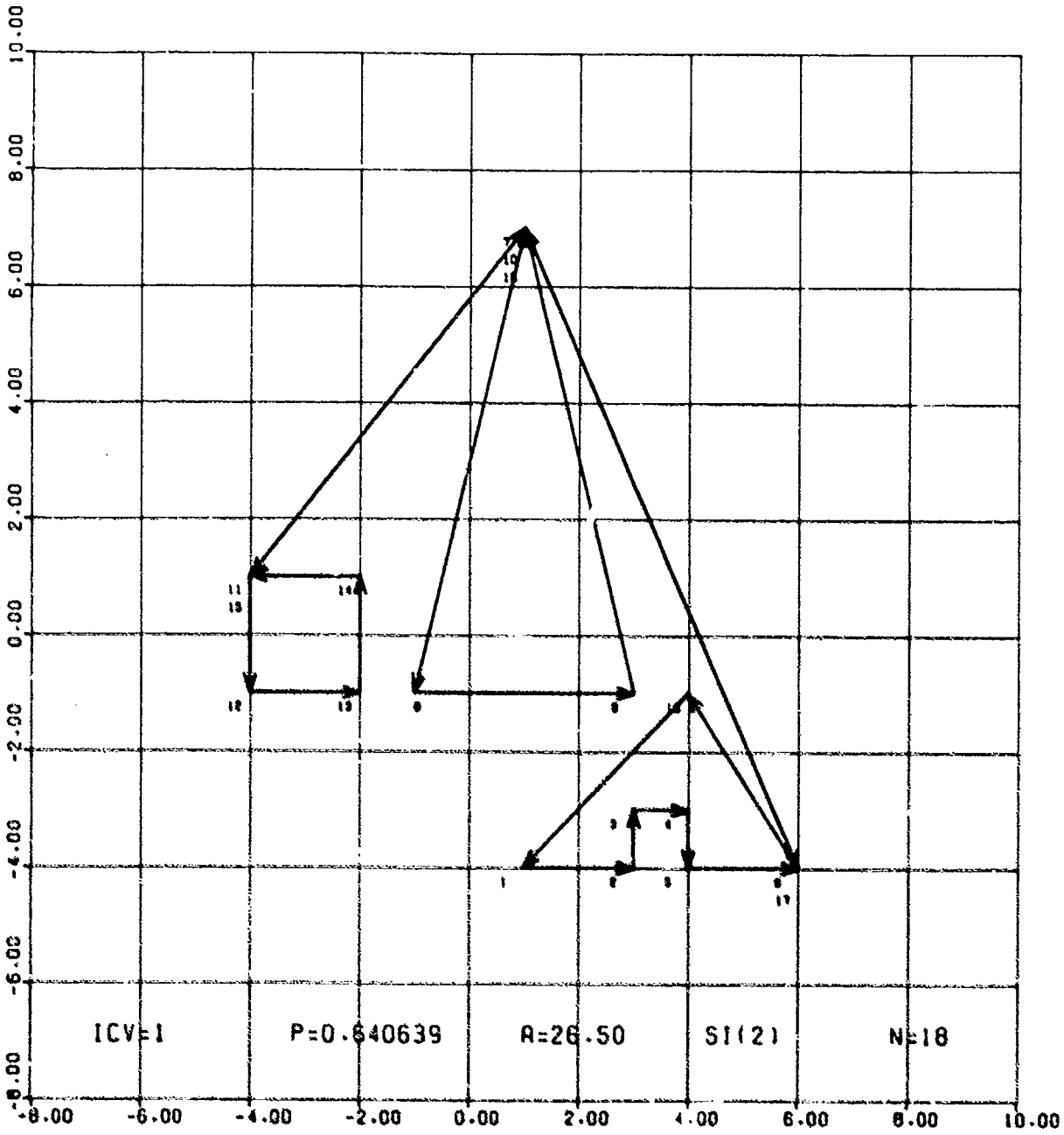

10.00
8.00
6.00
4.00
2.00
0.00
-2.00
-4.00
-6.00
-8.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
10.00
ICV=1
P=0.640639
A=26.50
SI(2)
N=18

Figure 41

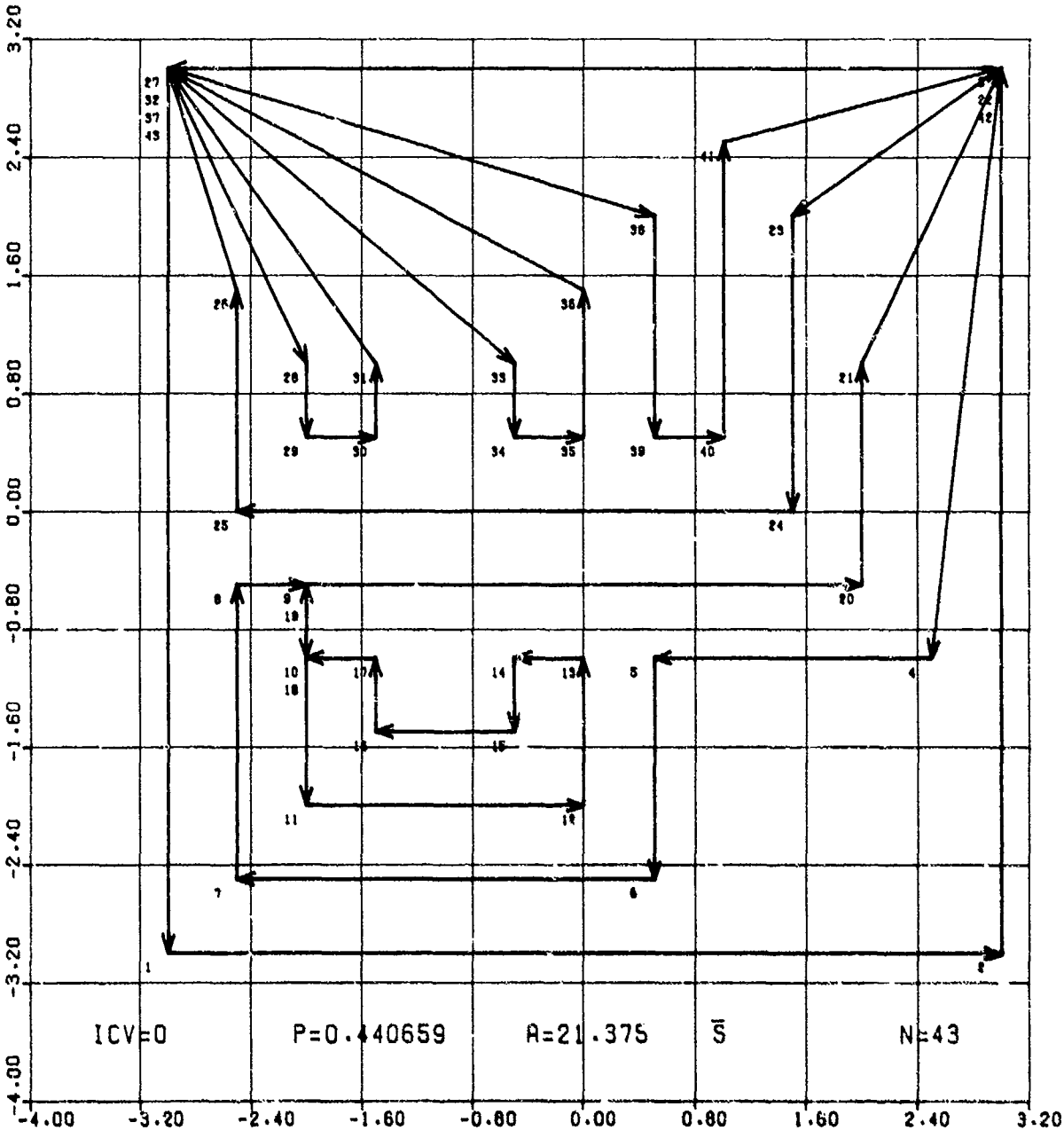

ICV=0
P=0.440659
A=21.375
S̄
N=43
3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-4.00
-4.00
-3.20
-2.40
-1.60
-0.80
0.00
0.80
1.60
2.40
3.20

Figure 42

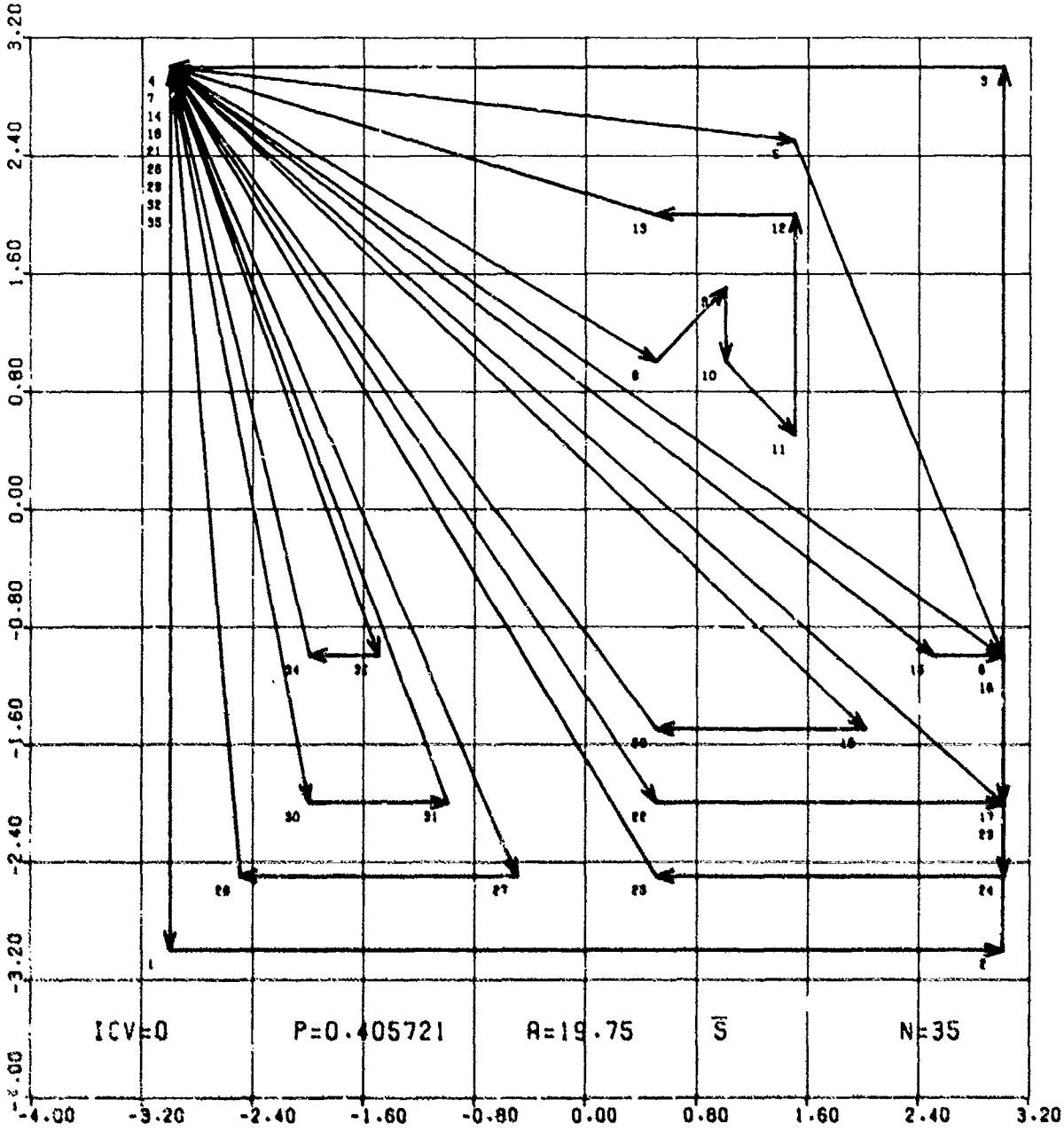

3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-4.00
-3.20
-2.40
-1.60
-0.80
0.00
0.80
1.60
2.40
3.20
ICV=0
P=0.405721
A=19.75
S
N=35

Figure 43

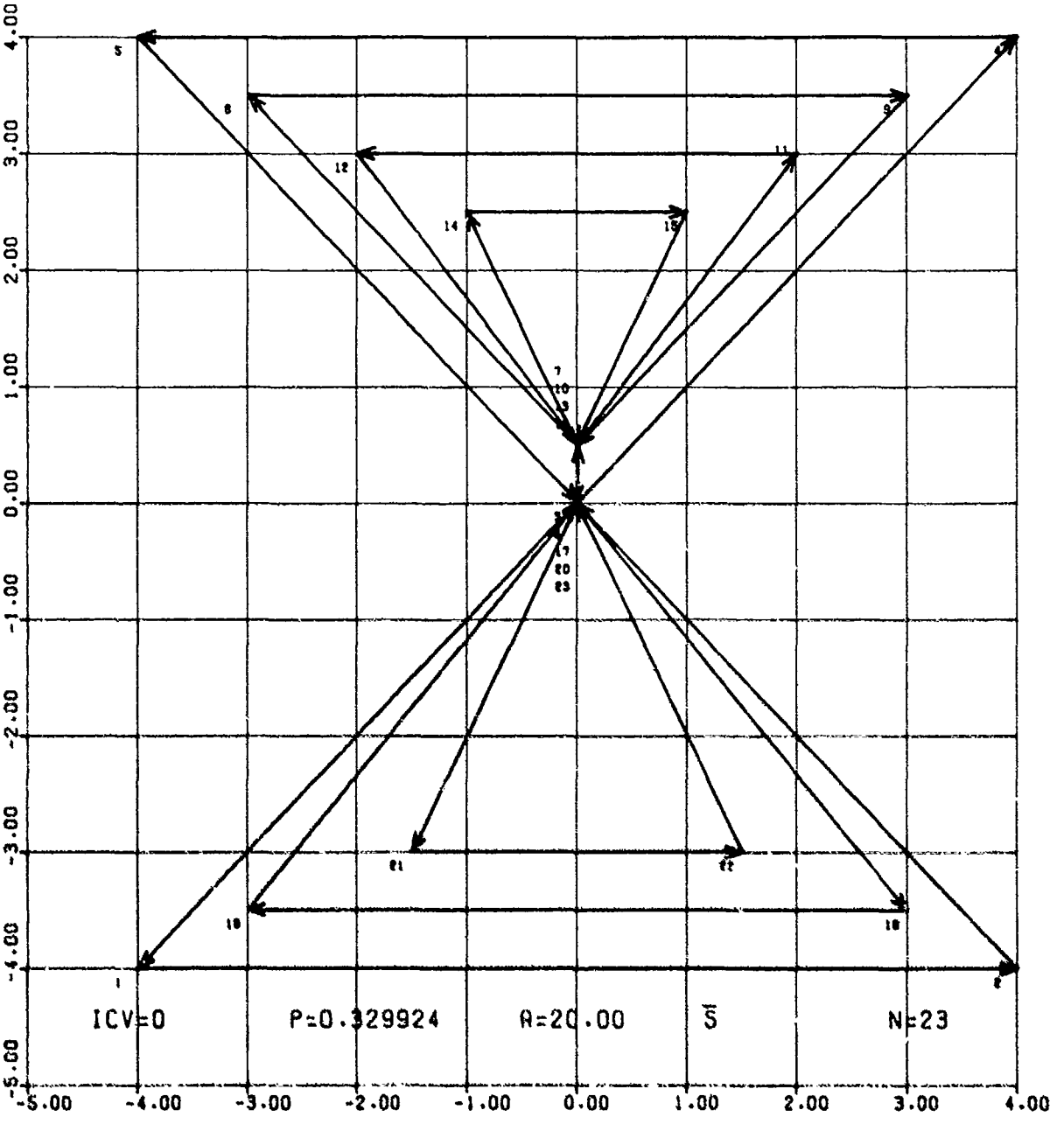

ICV=0
P=0.329924
R=20.00
S
N=23
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-5.00
-5.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00

Figure 44

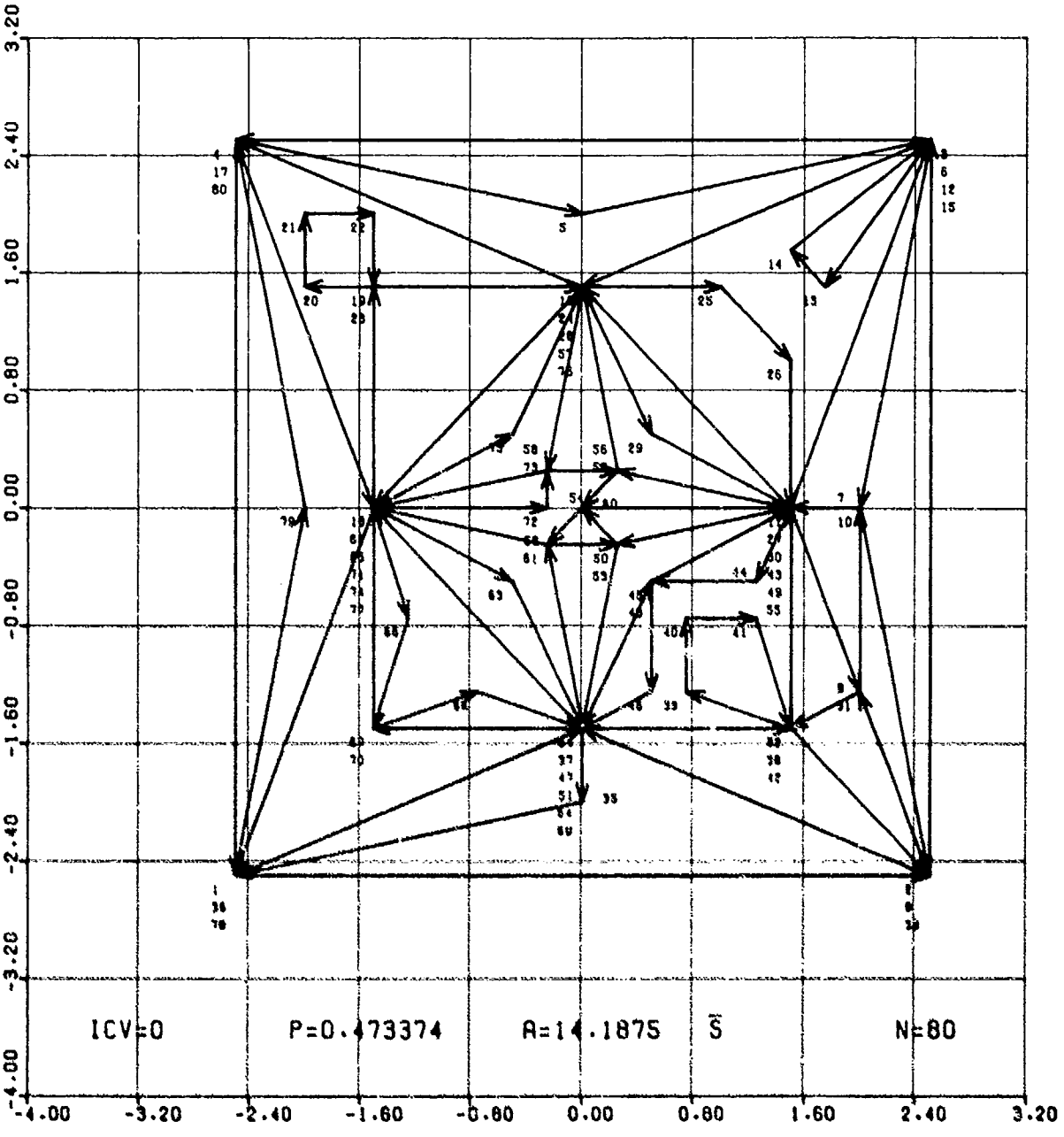

ICV=0
P=0.473374
A=14.1875
S
N=80
3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-4.00
-4.00
-3.20
-2.40
-1.60
-0.80
0.00
0.80
1.60
2.40
3.20

Figure 45

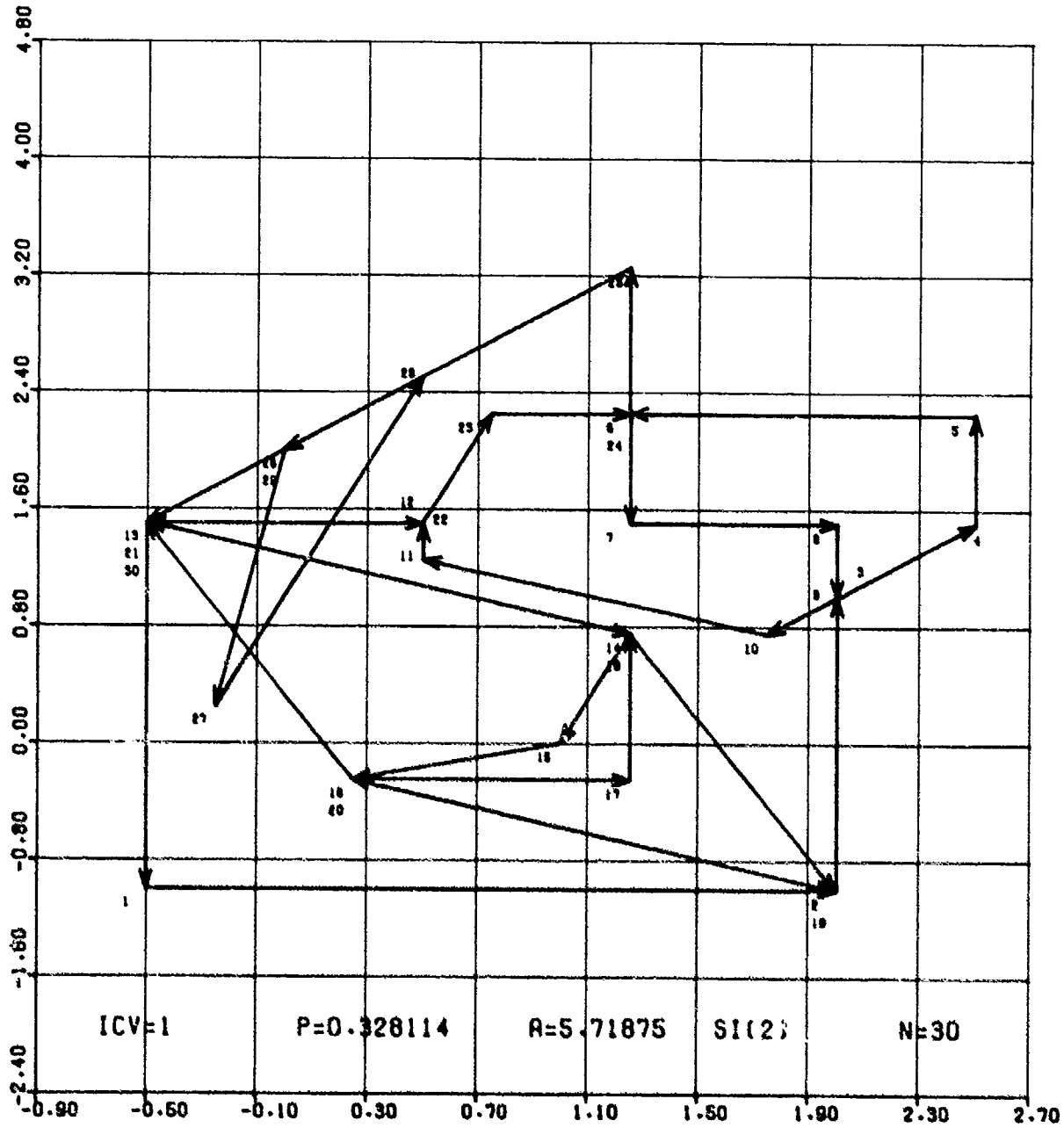

4.80
4.00
3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-0.90
-0.50
-0.10
0.30
0.70
1.10
1.50
1.90
2.30
2.70
ICV=1
P=0.328114
A=5.71875
SI(2)
N=30

Figure 46

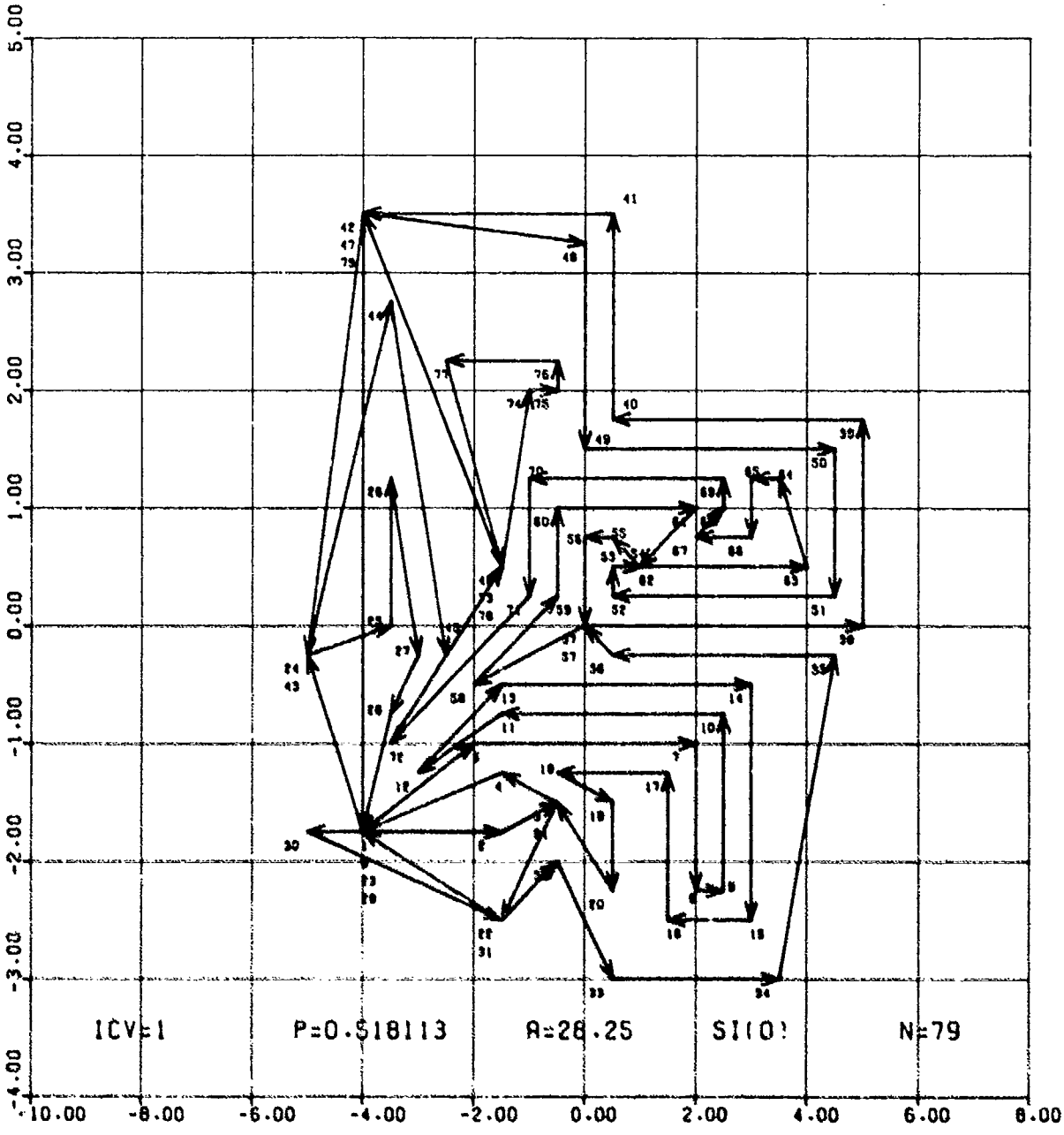

ICV=1
P=0.518113
A=28.25
SI(0)
N=79
5.00
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-10.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00

Figure 47

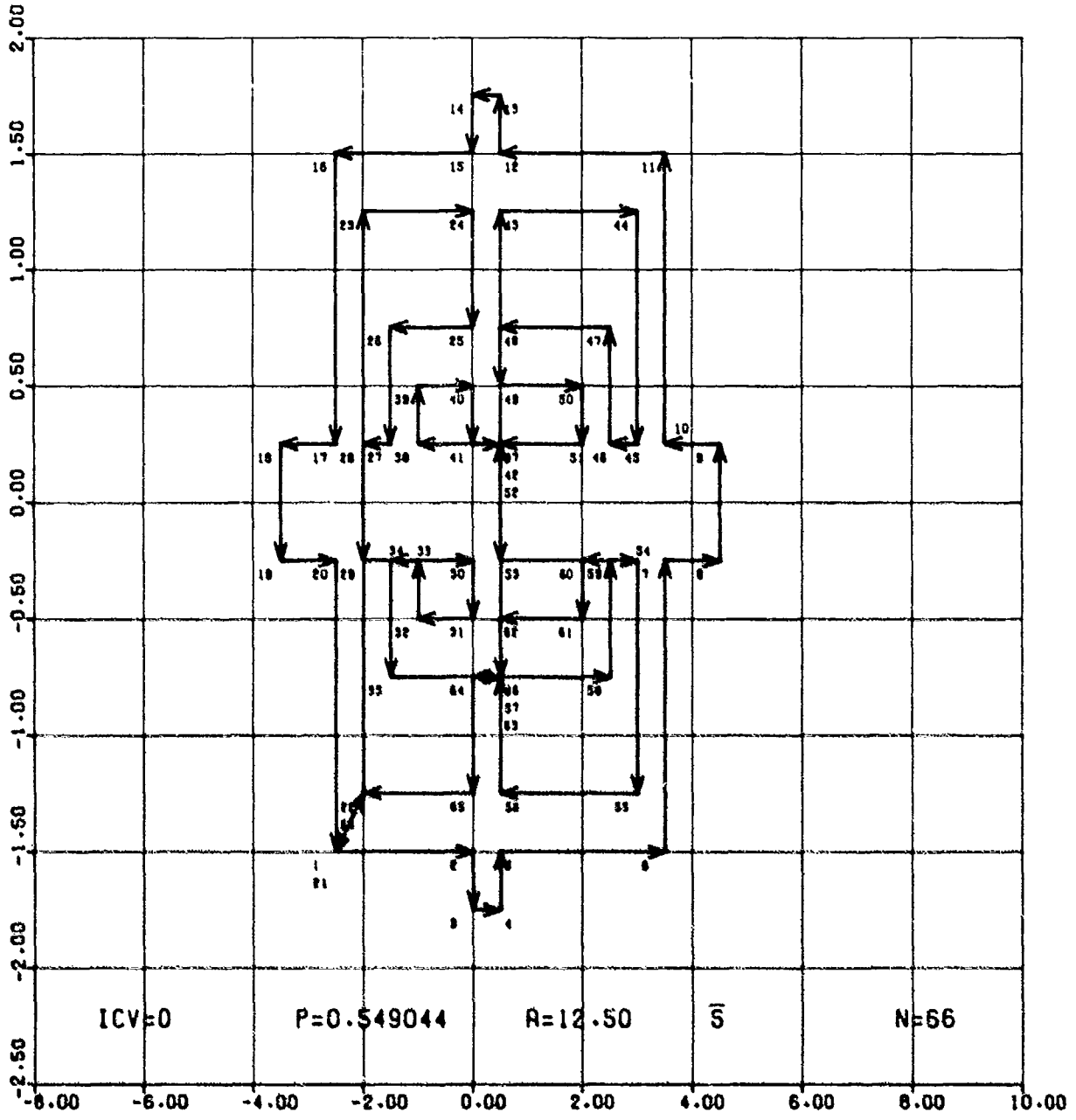

ICV=0
P=0.549044
A=12.50
S̄
N=66
2.00
1.50
1.00
0.50
0.00
-0.50
-1.00
-1.50
-2.00
-2.50
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
10.00

Figure 48

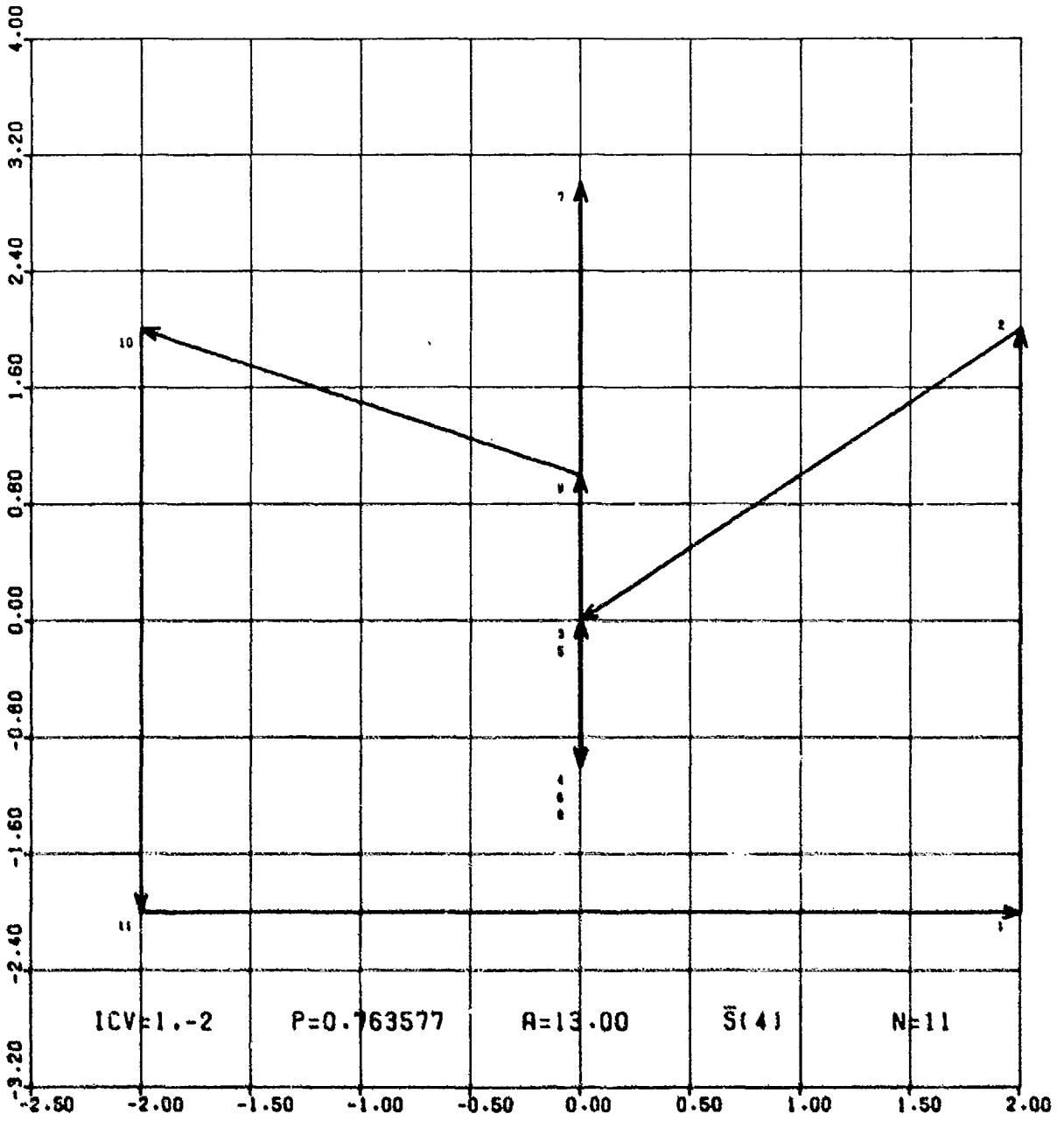

4.00
3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-2.50
-2.00
-1.50
-1.00
-0.50
0.00
0.50
1.00
1.50
2.00
10
ICV=1.-2
P=0.763577
A=13.00
S(4)
N=11

Figure 49

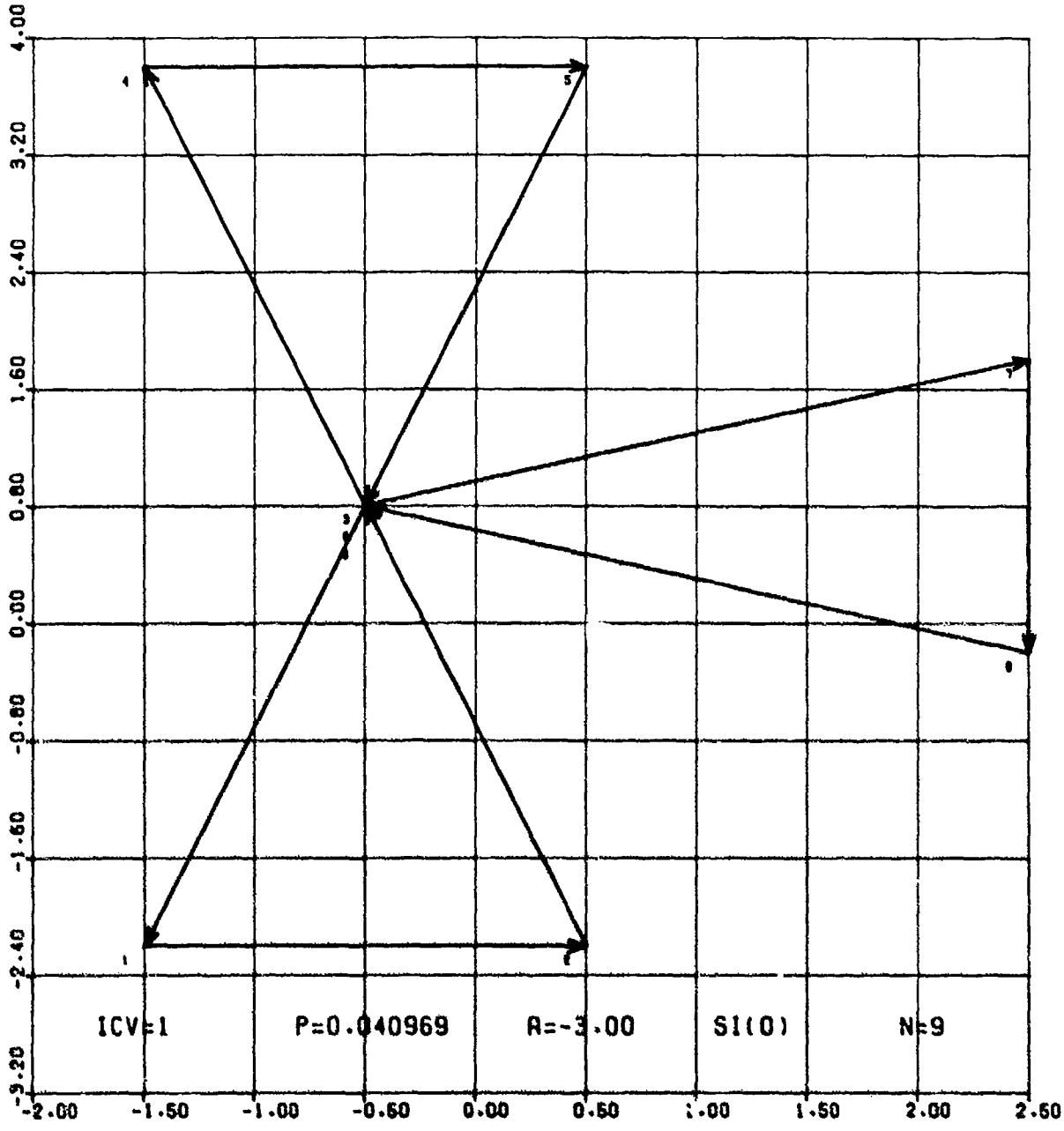

4.00
3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-2.00
-1.50
-1.00
-0.60
0.00
0.50
1.00
1.50
2.00
2.50
ICV=1
P=0.040969
R=-3.00
SI(0)
N=9

Figure 50

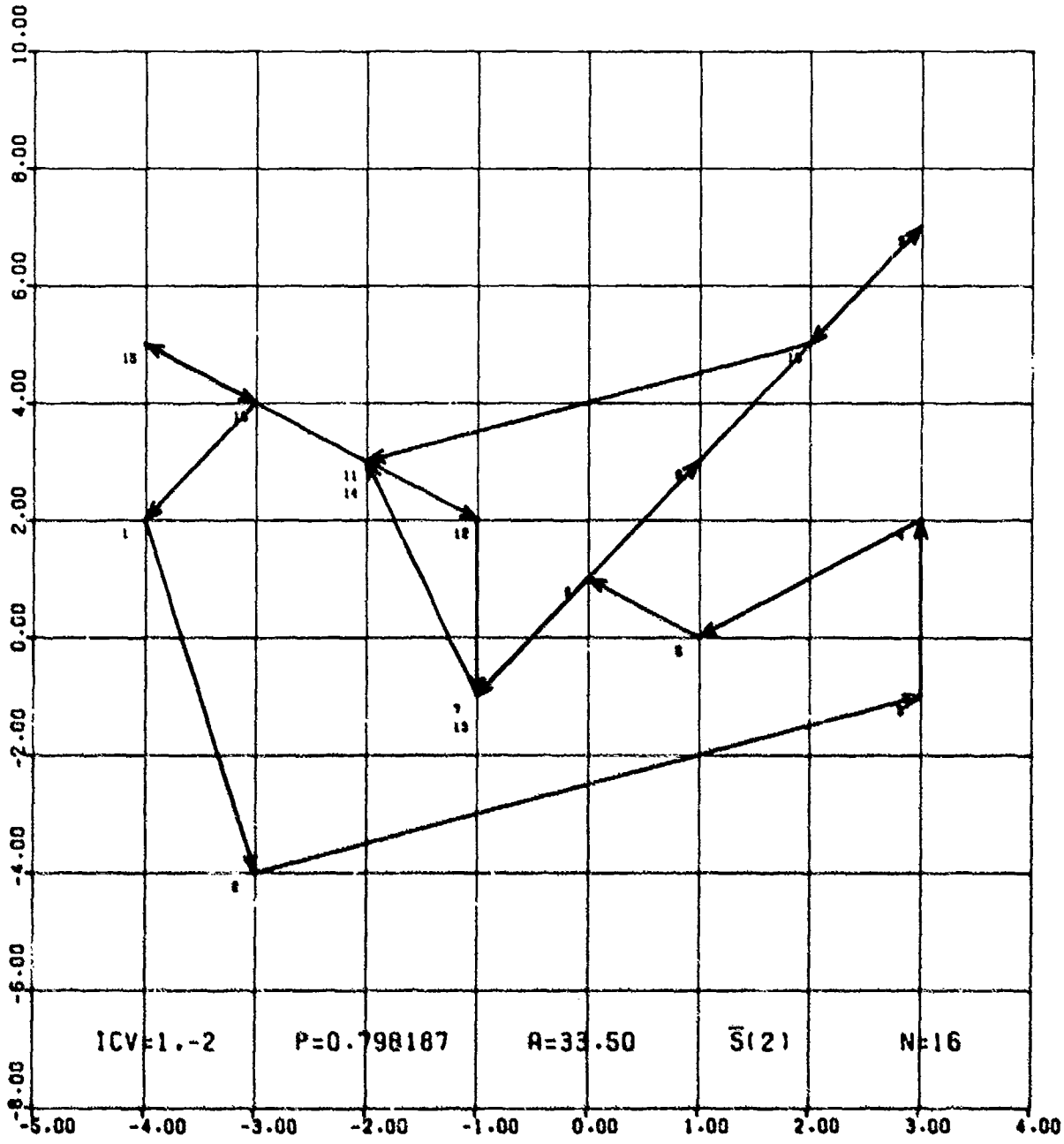

10.00
8.00
6.00
4.00
2.00
0.00
-2.00
-4.00
-6.00
-8.00
-5.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
ICV=1.-2
P=0.798187
A=33.50
S(2)
N=16

Figure 51

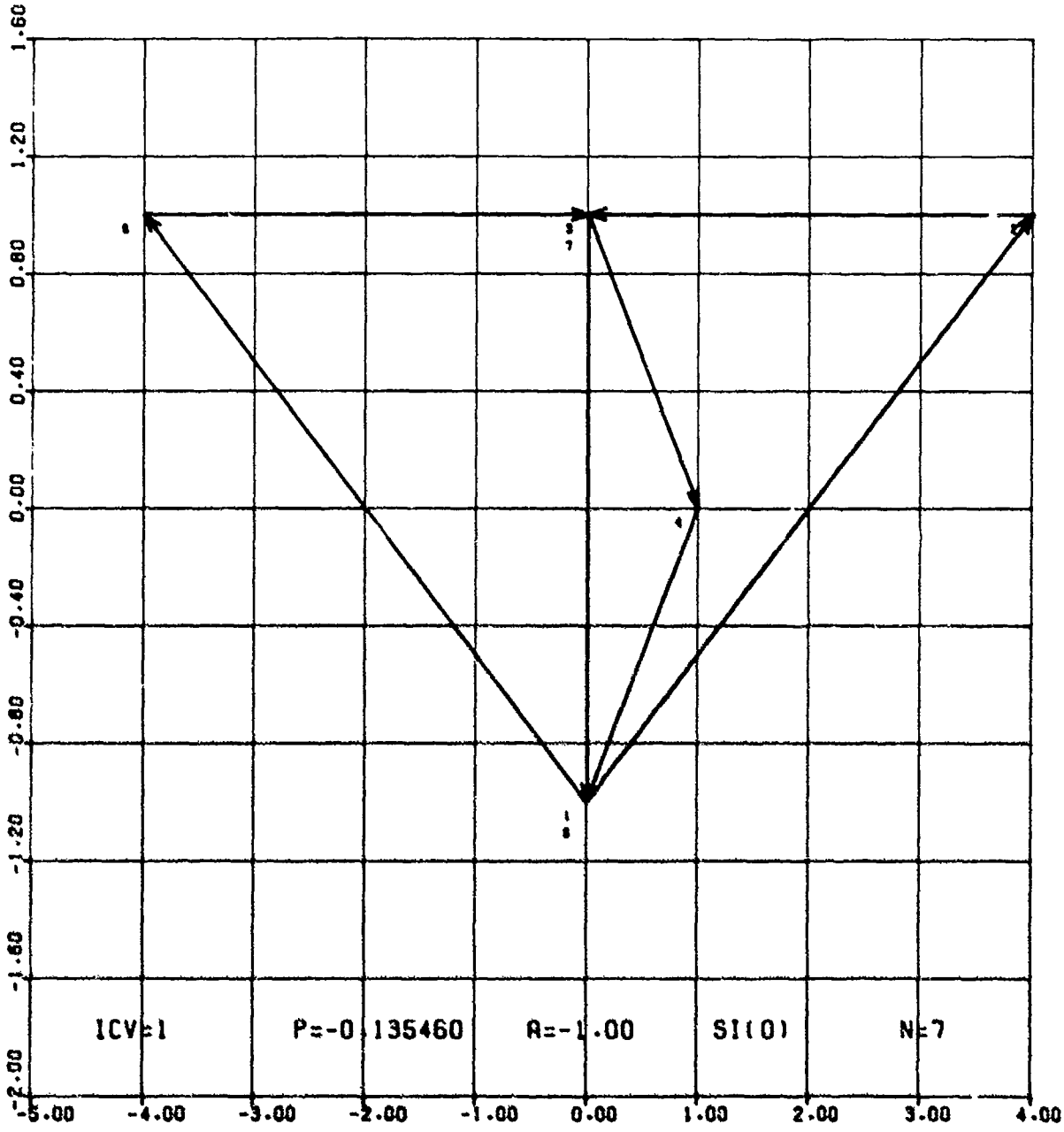

1.60
1.20
0.80
0.40
0.00
-0.40
-0.80
-1.20
-1.60
-2.00
-5.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
ICV=1
P=-0.135460
A=-1.00
SI(0)
N=7

Figure 52

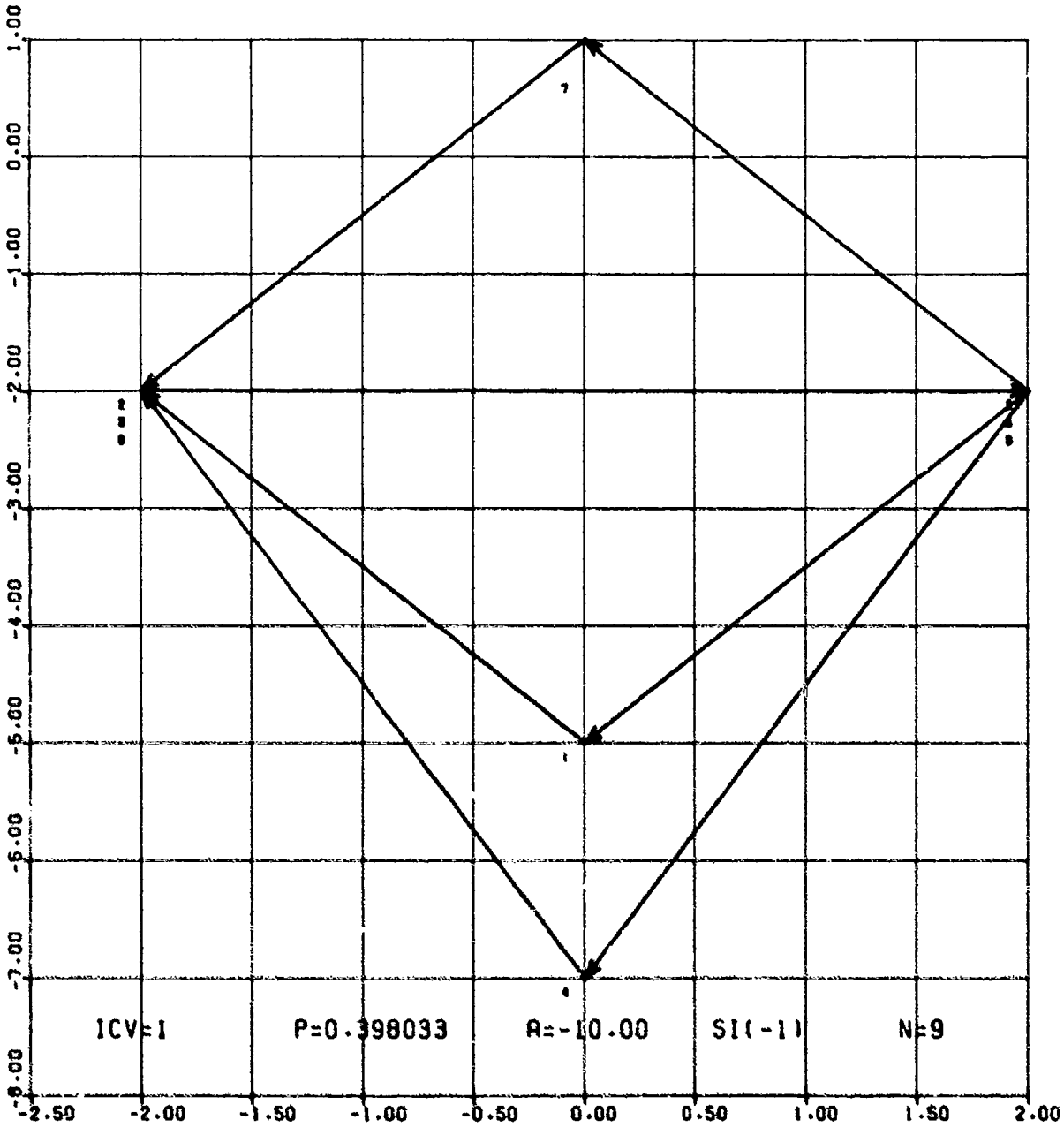

1.00
0.00
-1.00
-2.00
-3.00
-4.00
-5.00
-6.00
-7.00
-8.00
-2.50
-2.00
-1.50
-1.00
-0.50
0.00
0.50
1.00
1.50
2.00
ICV=1
P=0.398033
A=-10.00
SI(-1)
N=9

Figure 53

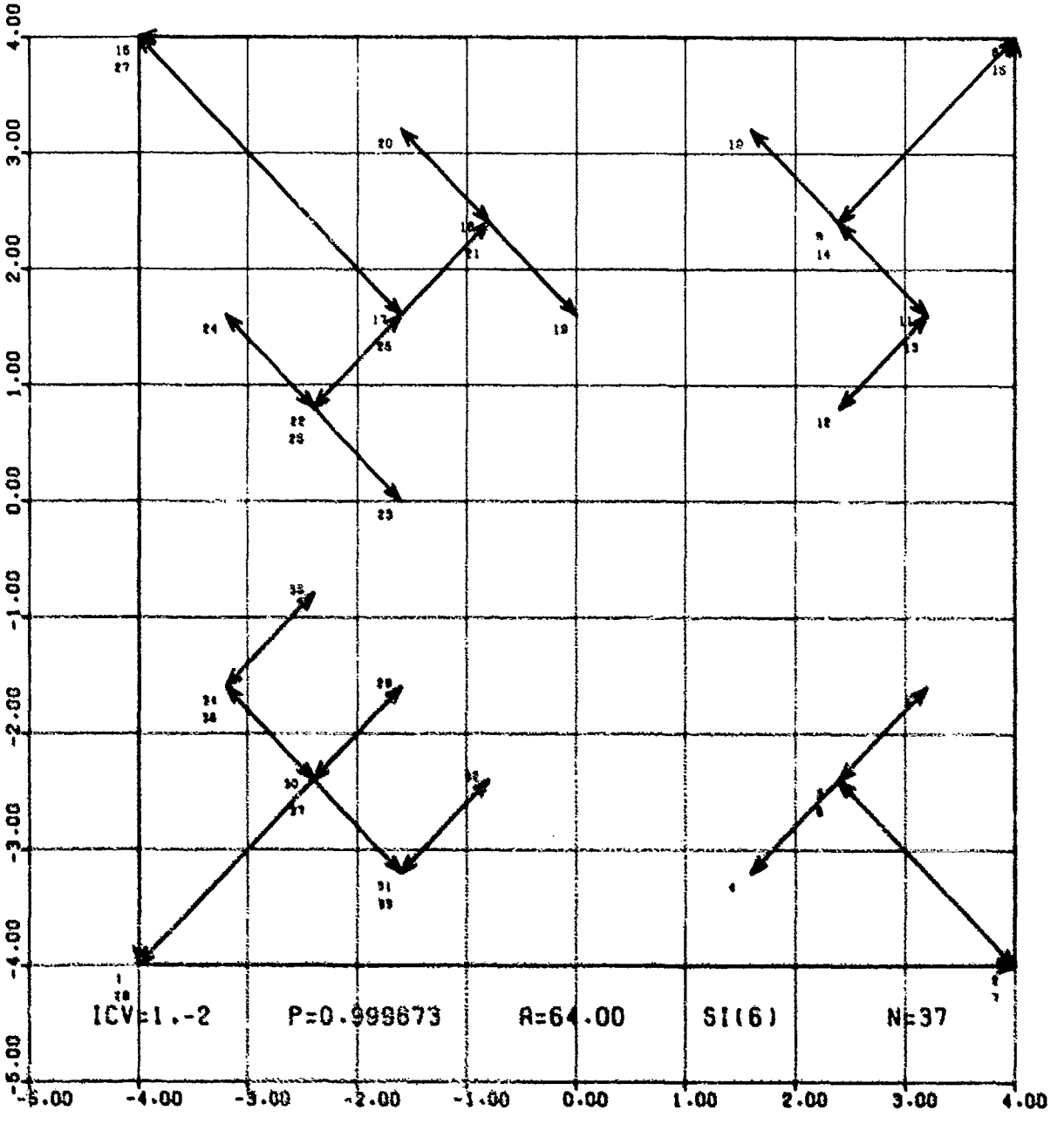

4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-5.00
-5.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
ICV=1.-2
P=0.999673
A=64.00
SI(6)
N=37

Figure 54

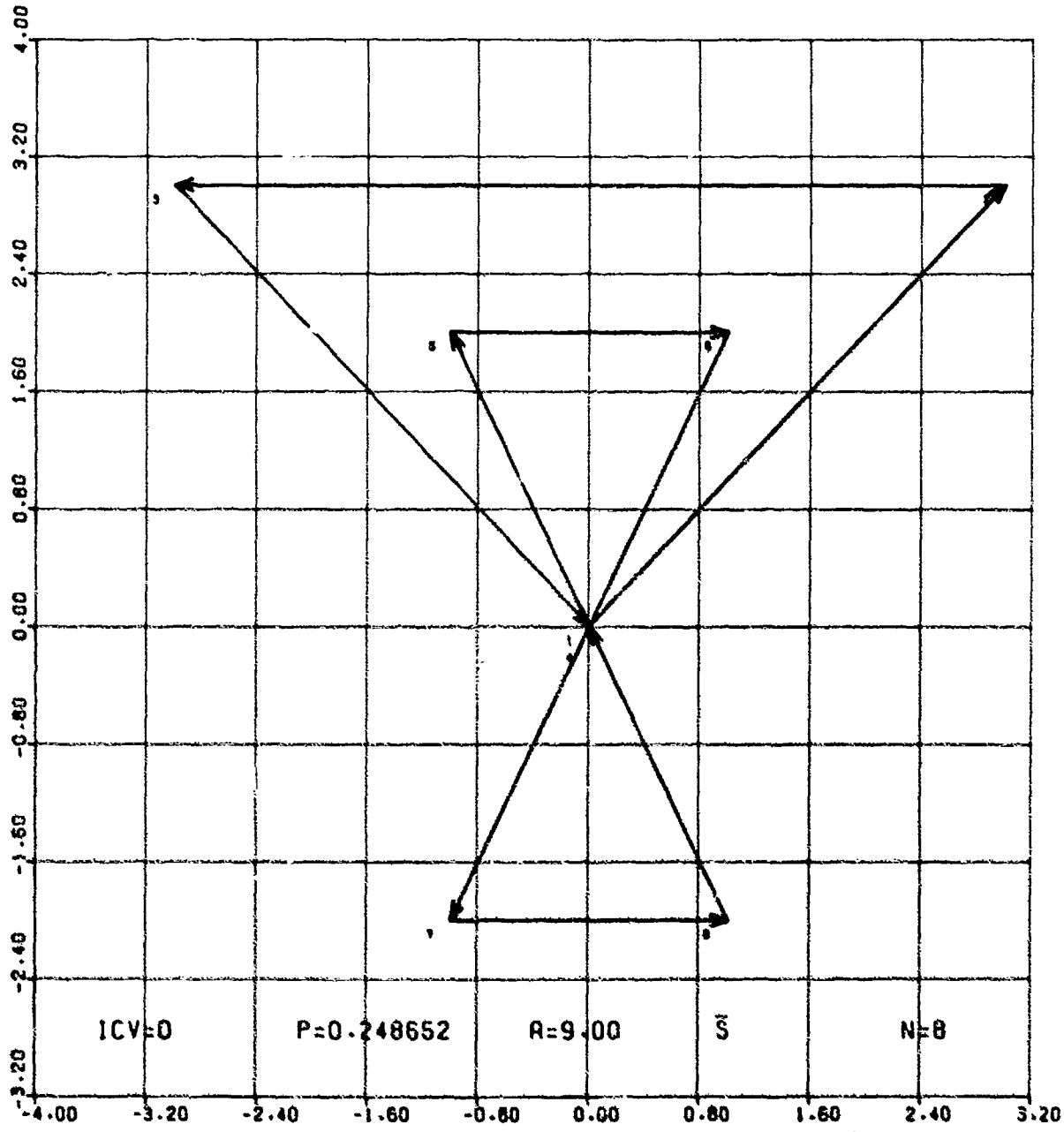

4.00
3.20
2.40
1.60
0.80
0.00
-0.80
-1.60
-2.40
-3.20
-4.00
-3.20
-2.40
-1.60
-0.80
0.00
0.80
1.60
2.40
3.20
ICV=0
P=0.248652
A=9.00
S̄
N=8

Figure 55

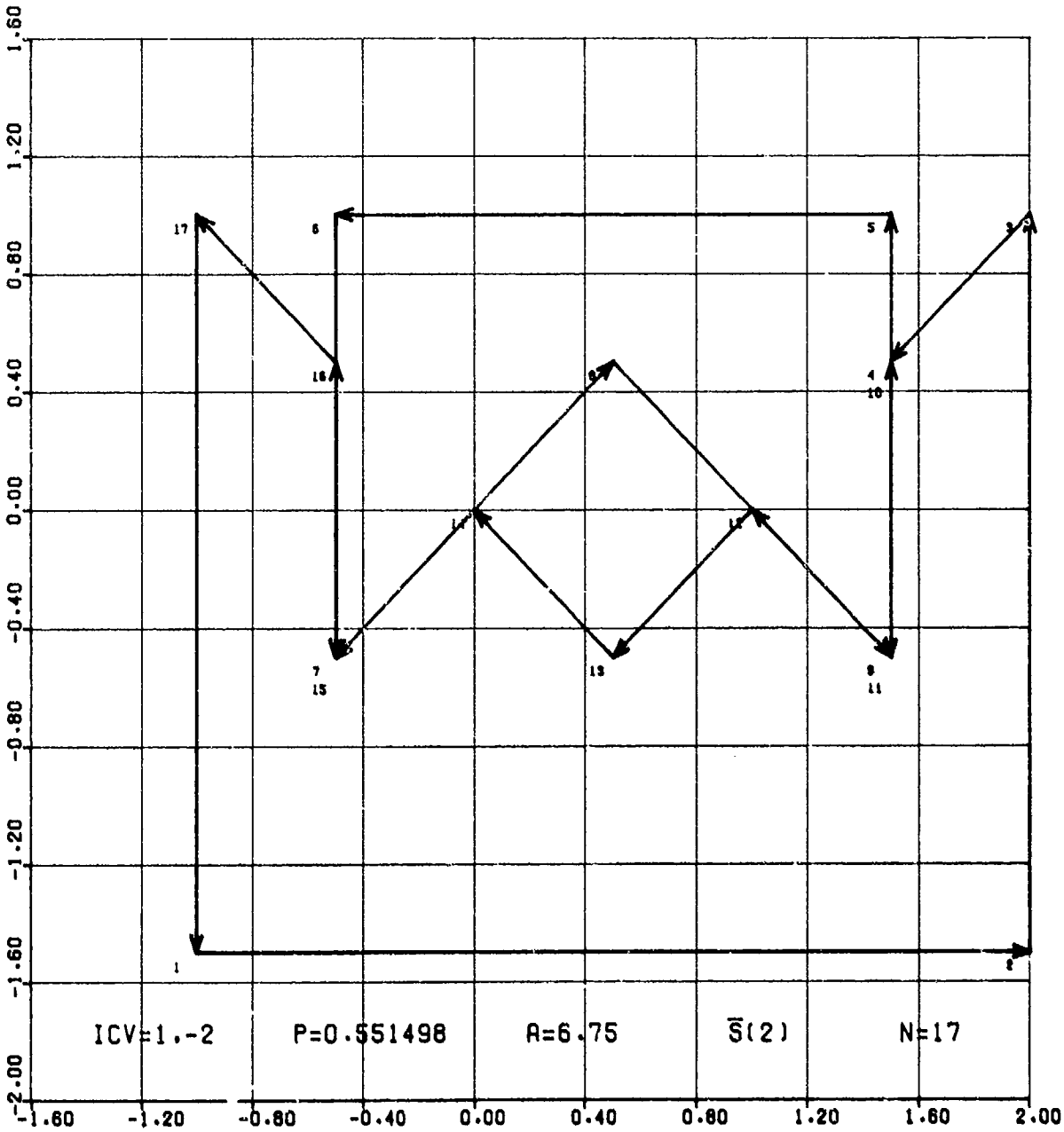

1.60
1.20
0.80
0.40
0.00
-0.40
-0.80
-1.20
-1.60
-2.00
-1.60
-1.20
-0.80
-0.40
0.00
0.40
0.80
1.20
1.60
2.00
ICV=1,-2
P=0.551498
A=6.75
S̄(2)
N=17

Figure 56

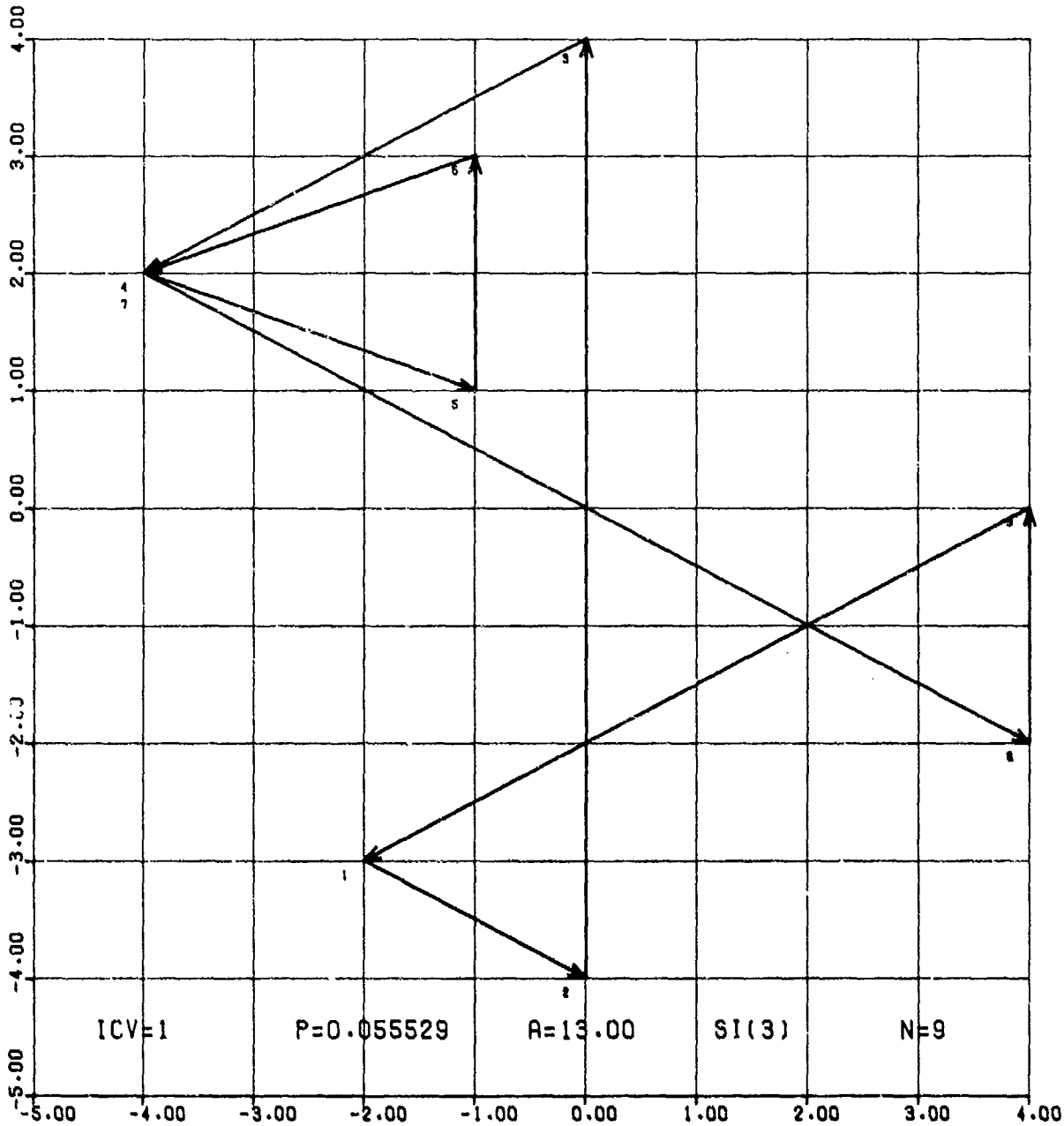

4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-5.00
-5.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
ICV=1
P=0.055529
A=13.00
SI(3)
N=9

Figure 57

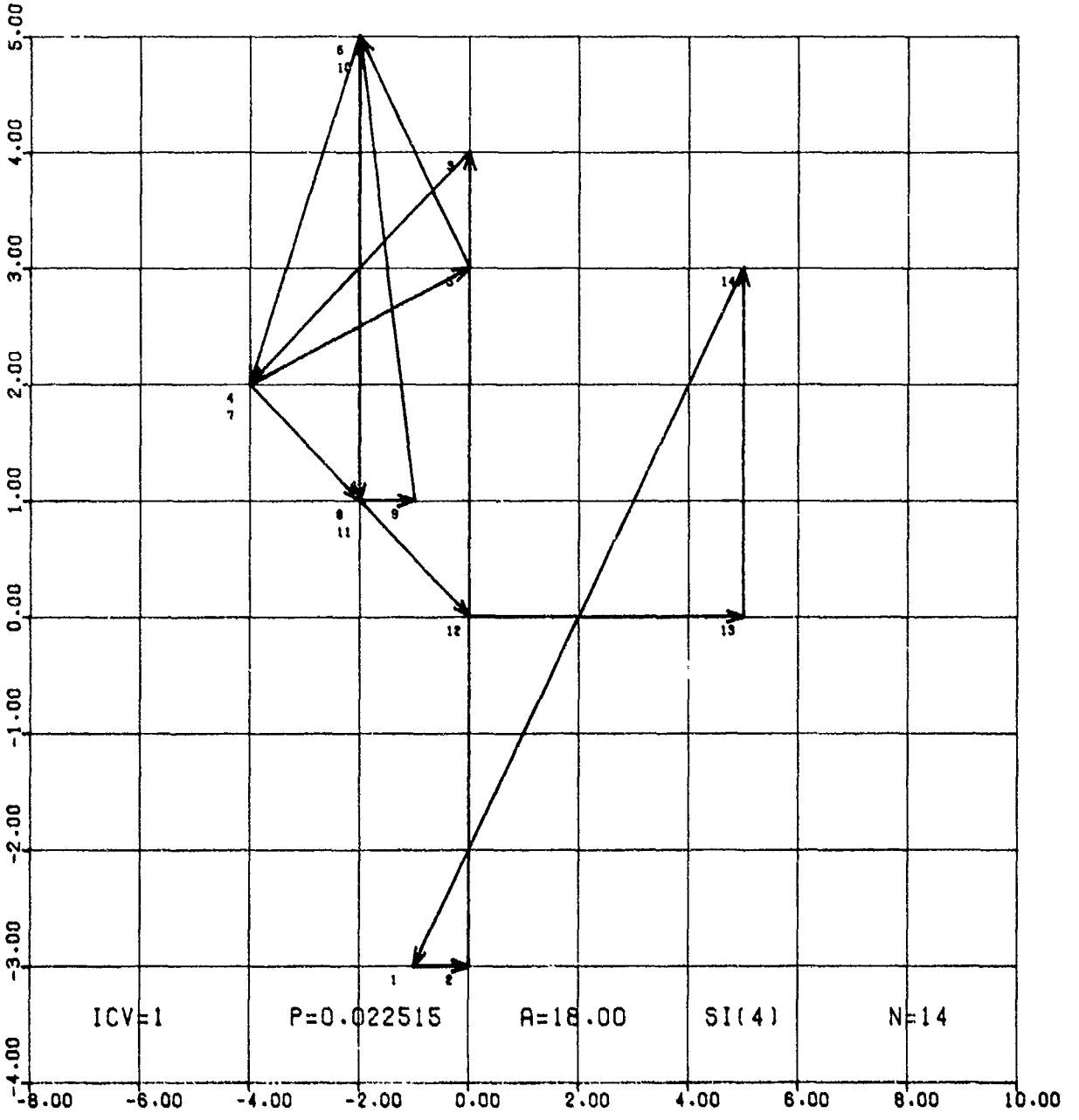

5.00
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
-8.00
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
10.00
ICV=1
P=0.022515
A=18.00
SI(4)
N=14

Figure 58

REFERENCES

1. A. R. Di Donato, R. K. Hageman, *An Algorithm for Finding the Convex Hull of a Finite Point Set in the Plane*, NSWC TN 79-42, Dahlgren, Va. 22448.

2. A. R. Di Donato, M. P. Jarnagin, Jr., R. K. Hageman, *Computation of the Bivariate Normal Distribution Over Convex Polygons*, Technical Report NSWC/DL TR-3866, Naval Surface Weapons Center, Dahlgren, Va. 22448, Sept. 1978.

3. A. R. Di Donato, M. P. Jarnagin, Jr., and R. K. Hageman, *Computation of the Integral of the Bivariate Normal Distribution Over Convex Polygons*, **SIAM J. Scientific and Statistical Computing**, v. 1, #2 (June 1980), pp. 179–186.

4. K. Knopp, *Theory of Functions*, PART ONE, translated by F. Bagemihl, Dover Publications, 1945.

5. U. S. Dept. of Commerce, National Bureau of Standards, *NBS Handbook of Mathematical Functions*, Appl. Math. Series, v. 55, U.S. Government Printing Office, Washington, D.C., 1964.

6. S. I. Warshaw, *Area of Intersection of Arbitrary Polygons*, Report No. UCID-17430, Lawrence Livermore Laboratory, U. of California, Livermore, Ca. 94550, March 1977.

APPENDIX A

NORMAL PROBABILITY OVER ARBITRARY POLYGONS BY (P – A)

APPENDIX A

NORMAL PROBABILITY OVER ARBITRARY POLYGONS BY (P−A)

In this appendix, we describe Procedure (P−A) for computing P(Π), Π in {Π}. It differs significantly from (P−B) which is discussed in Sections IV, V and VI. In (P−B), the concept of a winding number was introduced. A program package, called Program B, was developed made up of five subprograms: P-2, VALR-2, SORT III, VALR-7, and SMP-7. The (P−A) procedure, like (P−B), was developed to treat SI elements or $\bar{S}$ elements with SAR(s). It decomposes such an element into a set of disjoint elements in $\{\bar{S}\}$ or {S} depending upon whether the β or α-option is used to specify Π (See page 24). If the decomposed elements are in $\{\bar{S}\}$ they are treated by removing any PAR(s). The resulting decomposed elements are then in {S}, and P for such elements is computed by using VALR-7. A program package is also described in this appendix, using (P−A), which is comprised of six subprograms: P-7, VALR-7, SORT I, SORT II, SORT III and SMP-7. This program package is called Program A.

Procedure (P−A) has merit, because its decomposition of Π into disjoint elements in $\{\bar{S}\}$ (or {S}) allows the analyst to gather a more detailed picture of the type of region Π represents. Moreover since it permits the decomposition to be carried out in a backward (from (N) to (1)) as well as forward direction, it gives an additional means of checking final results (The decomposition is not necessarily the same in the forward and backward directions). Nevertheless (P−B) is deemed the better overall procedure. Some summarizing remarks comparing (P−A) and (P−B) are given on page A-15.

For computational efficiency with Program A, the pre-processing of Π must be kept to a minimum by specifying beforehand the smallest class to which Π belongs. However if Π is erroneously assigned to a class to which it does not belong, then, because VALR-7 cannot treat SI elements nor elements in $\{\bar{S}\}$ with SAR(s) a wrong result for P(Π) is likely. Recall, on the other hand, that VALR-2 of Program B can treat any polygon with the same computational efficiency, although small improvements in efficiency can sometimes be achieved by pre-processing Π with SORT III and by using VALR-7 for VALR-2 as indicated in P-2 (see page 40).

We assume throughout this appendix that (P−A) or Program A are under discussion unless stated otherwise.

If Π is in {S}, then by computing $P(a_k)$ for each a_k of S, P(S) is obtained by using (24) if $A > 0$, and (26) if $A < 0$. In the first case Π is PO and in the second Π is NO.

If Π is in $\{\bar{S}\}$, then SAR(s) can occur (see pages 12, 31). The SDP and SCP are treated by removing appropriate points from the V-array which specifies Π. The reduced polygon is in {S}, and P(S) (=$P(\bar{S})$) is obtained as explained in the previous paragraph.

If Π is SI, then Π is decomposed into a set of disjoint elements, often referred to as *isolated elements*, in {S} or $\{\bar{S}\}$ depending on whether Π has been specified by the α or β numbering scheme. If Π is decomposed into $(S^1 \cdots S^n)$ then $P(\Pi) = \Sigma_1^n P(S^i)$ and if the decomposition results in $(\bar{S}^1 \cdots \bar{S}^n)$, then each $\bar{S}^i$ must be processed first, as explained in the preceding paragraph, before $P(\bar{S}^i)$ can be computed such that $P(\Pi) = \Sigma_1^n P(\bar{S}^i)$.

The α and β-options come into play if Π is in $\{\bar{S}\}$ or $\{\Pi\}$ (see page 14). Generally the β-option is the more efficient, since it will often require the treatment of fewer angular regions (one must be careful, however, that it specifies Π correctly). In Figure A-1,[1] 16 angular regions occur with the α-option. For the same Π using the β-option, as shown in Figure A-2, only ten regions occur, and two of those are of no consequence since $\Delta\theta = 0$ for them.

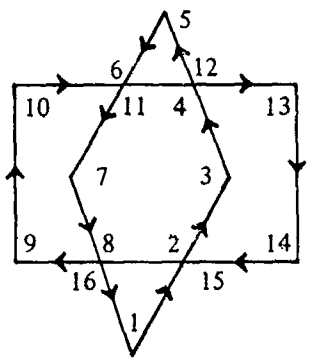


Figure A-1. Polygon with α-Option

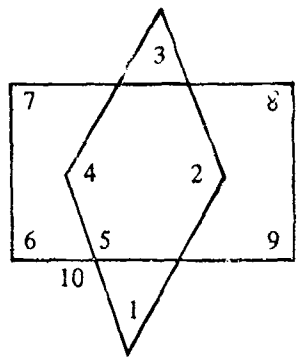


Figure A-2. Polygon with β-Option

The two figures above also show that to determine whether Π is SI every vertex, meeting or intersection of two edges must be numbered each time it occurs in the order it occurs (see page 14). Subsequently, the polygon can be numbered with the β-option, if the opportunity exists to do so, for the actual computation of P(Π). Note that although Π is SI in Figures A-1 and A-2, this would not be concluded from our T-characterization (page 15) by examining the numbered points in Figure A-2.

If Π is SI and numbered under the α or β-options, then by (P−A) Π is decomposed in the following way:

Starting at node (1), we look for the *first* MN (see page 14) that is met for the *second* time, say MN(k) is met for the second time, say at node (k + m), $1 \leqslant k \leqslant N - 1$, $2 \leqslant k + m \leqslant N + 1$, ($k + m = N + 1$ means $(k + m) = (1)$). In Figure A-3, this occurs at MN(3), since this point is first encountered by node (3) and for the second time by node (7). The same property also holds for Figure A-4 at MN(3), which is met for the second time at node (6).

When this situation occurs, there exists two possibilities,[2] [4, p. 16]:

(a) Edges $\overline{k+1}$ and $\overline{k+m}$, $m \geqslant 2$, with $\overline{k+1}$ originating at MN(k) and $\overline{k+m}$ terminating there, have more than one point in common, (Under α-option $m = 2$).

[1] Figures A-1 and A-2 are the same as Figures 25 and 26.

[2] Actually, a third possibility exists, namely SDP. For each set of SDP, as soon as it is detected, one of the points is removed.

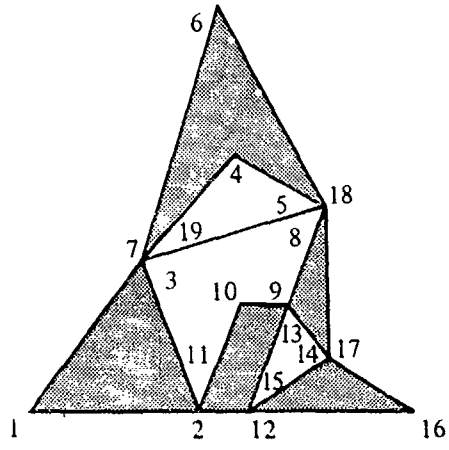


Figure A-3. An SI Polygon with IBND Required Over Shaded Areas, N = 19

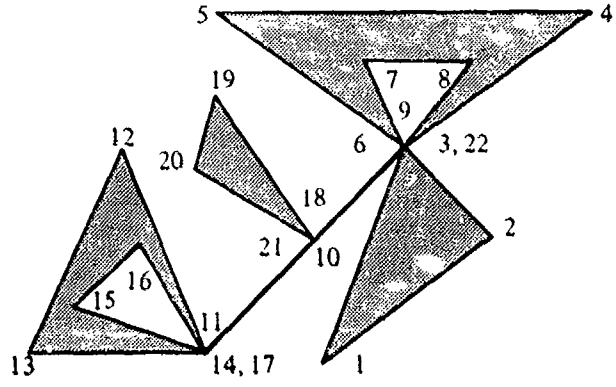


Figure A-4. An SI Polygon with Line Segments. IBND Required Over Shaded Areas, N = 22.

(b) Nodes (k), (k + 1), ..., (k + m), m > 2, specify a polygon in $\{\tilde{S}\}$ (Under α-option the polygon is in $\{S\}$.) with a clearly defined orientation (see page 9). Here also $\overline{k+1}$ originates at MN(k) and $\overline{k+m}$ ends there.

In case (a), a line segment of II exists where edges $\overline{k+1}$ and $\overline{k+m}$ completely overlap. The configuration of Figure A-4 contains examples of such line segments. They will be identified below where II in that figure is decomposed.

In case (b), an element of $\{\tilde{S}\}$ or $\{S\}$ is obtained with M = m nodes. The function P is computed for this element (using VALR-7). The nodes (k) to (k + m − 1) are then deleted from the original V-array specifying II, and the decomposition continues starting at (k + m), which is now the k^{th} element of the updated V-array. Since II has only N nodes, this will end after a finite number of such steps. P(II) is computed by adding up the positive contributions from PO isolated polygons and the negative contributions from the isolated NO polygons.

A proof that the above decomposition can always be carried out is essentially given in Knopp, [4, page 15]. His proof, which requires minor changes for our use, is constructive. We have used it as a guide in the decomposition procedure just described.

In detailing the decomposition of the polygons in Figures A-3 and A-4, the isolated simple polygons are superscripted in the order they are isolated, i.e., S^1, S^2, ..., S^n. They are identified, as usual, by their nodes. We also give their orientation. They are both specified by the α-option.

Figure A-3

S^1: (3, 4, 5, 6, 7) PO
S^2: (2, 7, 8, 9, 10, 11) NO
S^3: (12, 13, 14, 15) NO
S^4: (1, 11, 15, 16, 17, 18, 19, 20) PO

Note: Case (a) does not occur here.

Figure A-4

S^1: (3, 4, 5, 6) PO
S^2: (6, 7, 8, 9) NO
S^3: (11, 12, 13, 14) PO
S^4: (14, 15, 16, 17) NO
Line Segment: (10, 17, 18) Case (a)[3]
S^5: (18, 19, 20, 21) PO
Line Segment: (9, 21, 22) Case (a)
S^6: (1, 2, 22, 23) PO

An automatic formal procedure for decomposing an SI polygon Π is carried out by listing the integers corresponding to its ordered set of nodes. In general, after S^i is found, $i \neq n$, all the integers corresponding to the nodes of S^i, except the last, are dropped from the initial list V. However if S^i contains node (1) then one is retained, rather than the integer corresponding to the last node of S^i. For example, for Figure A-4, we would have after deleting S^1:

V = 1, 2, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23.

Starting at 6, S^2 is found, and the above list is reduced to

V = 1, 2, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23.

After S^3 and S^4 are found, we have

(•) 1, 2, 9, 10, 17, 18, 19, 20, 21, 22, 23.

At this stage, we get line segment (10, 17, 18) and the set (•) above contracts to

V = 1, 2, 9, 18, 19, 20, 21, 22, 23.

After S^5 is found, we have

V = 1, 2, 9, 21, 22, 23.

The removal of the line segment (9, 21, 22), leaves us with (1, 2, 22, 23) which is S^6 and concludes the decomposition.

Figures A-5 and A-6 contain the same polygon with the α and β-options, respectively. From the details of the decompositions given below, it will be clear PAR(s) under the α-option are removed during the decomposition, but they can be retained under the β-option as this example

[3]Note that the decomposition isolates SAR(s), if the α-option is used.

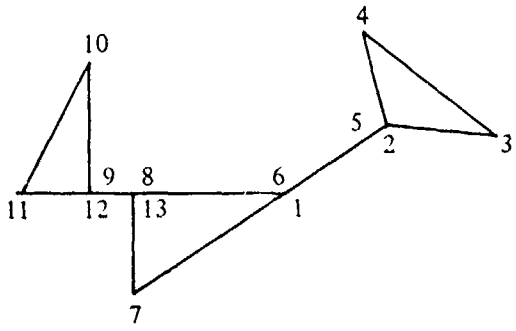


Figure A-5. SI Polygon, α-Option, N = 13

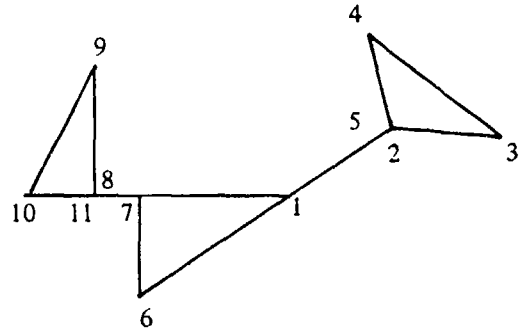


Figure A-6. SI Polygon, β-Option, N = 11

shows. Thus, with the β-option an additional program SORT II, is needed to eliminate PAR(s). However it actually does a little more by eliminating SCP(s) (see p. 31). This program works in the same way as SORT III only it need not check for SDP,[4] (see page 31).

For Figure A-5, the decomposition by SORT I gives:

S^1 = (2, 3, 4, 5) PO
Line Segment (1, 5, 6) (Eliminated by SORT I since a PAR requires p = 0)
S^2 = (9, 10, 11, 12) PO
Line Segment (8, 12, 13) (Eliminated by SORT I since a PAR requires p = 0)
S^3 = (1, 7, 13, 14) NO.

With Figure A-6, the decomposition by SORT I yields:

$\ddot{S}^1$ = (2, 3, 4, 5) PO
$\ddot{S}^2$ = (8, 9, 10, 11) PO
$\ddot{S}^3$ = (1, 5, 6, 7, 11, 12) NO

where points (5) and (11) are eliminated from $\ddot{S}^3$ by SORT II.

Note that in Figure A-6, it was necessary to include the point (11) coinciding with (8) or (7), otherwise the decomposition would have left the SI polygon (1, 5, 6, 7, 8, 9, 10, 12) with no coinciding points. Hence, after SORT II, which would remove (5), *the resulting element (1, 6, 7, 8, 9, 10, 12) is SI and VALR-7 applied to it would yield a wrong result.* This example serves to emphasize, with the (P−A) procedure, the care that must be taken in using the β-option to specify II. The (P−B) procedure would have no difficulty in this situation since VALR-2 can handle SI polygons directly.

We proceed with a description of the computer program package based on (P−A), i.e., Program A. Recall that for (P−B), Program B is composed of P-2, VALR-2, SORT III, VALR-7,

[4]It should be evident that SDP are always detected and removed by SORT I by testing if M ≤ 2 (see box 6 of the Flow Chart 6).

SMP-7. For easy reference, they were referred to as subprograms 1, 2, 3, 4 and 8 respectively. For Program A, the program package consists of P-7, VALR-7, SORT I, SORT II, SORT III, and SMP-7. For easy reference, we number them accordingly: 5 ↔ P-7, 4 ↔ VALR-7, 6 ↔ SORT I, 7 ↔ SORT II, 3 ↔ SORT III, 8 ↔ SMP-7. All of these subprograms are in subroutine format. Program P-7 serves as a master routine, VALR-7 is much like VALR-2, but it can only compute P for a single angular region, or polygons in $\{\bar{S}\}$, provided they contain no SAR(s). SORT I decomposes Π into a set of disjoint elements in $\{S\}$ or $\{\bar{S}\}$ depending on whether it is numbered with the α or β-option. It is primarily used if Π is SI. SORT II is used on the disjoint elements in $\{\bar{S}\}$, obtained from SORT I, to remove SCP(s). SORT III was taken up in Section V. It is used to delete SCP and SDP from Π, the original polygon, when Π is in $\{\bar{S}\}$. SMP-7 is used to compute the function A as given in (22); where $|A|$ is taken as the area of Π; the sign of A is used in VALR-7 to determine the orientation of Π when Π is in $\{S\}$ or $\{\bar{S}\}$.

The flow charts for 3 and 4 are given at end of Section V, pages 43-45, since they also make up part of Program B. Flow charts for 5, 6, 7 are given at the end of this appendix, pages (A-17-A-19). No flow chart is given for SMP-7. Fortran IV listings of all the programs are given in Appendix F.

Program 5 (see Flow Chart 5) uses as input x, y, N, ICV and IOP. These notations have all been used previously in Section V. The various values for ICV have slightly different meaning here. If ICV = 0, $P(S)$ or $P(\bar{S})$ is wanted where $\bar{S}$ has no SAR(s) such as in Figure 35. If ICV = 1, then $P(\bar{S})$ is wanted, where 3 is used before 4 to remove SCP and SDP. If N = 1, P for an angular region is wanted. If ICV = ±2 or ±3, P for an arbitrary polygon is wanted. IOP specifies the accuracy desired; it can be assigned the values 1, 2, or 3 to yield approximately 3, 6, or 9-decimal-digit accuracy, respectively, for P of each angular region.

If $|ICV| = 2$, it is assumed that the α-option has been used to specify an element in $\{\Pi\}$. If $|ICV| = |3|$, it is assumed the β-option has been used. In the first case the isolated elements are in $\{S\}$ and in the second case they may be in $\{\bar{S}\}$. If ICV = 2 or 3, the processing of Π, by SORT I, begins at (1) and progresses sequentially through nodes (2), (3), ..., (N). If ICV = −2 or −3, then Π is processed by SORT I in reverse order starting at (N) and progressing sequentially through (N − 1), (N − 2), ..., (1).

The parameter IND is discussed below.

We now consider 6 in more detail by using its flow chart, page A-18. Two points of $\{v_r \equiv (x_r, y_r)\}$, r = 1, 2, ..., K, are said to *coincide* or are *duplicates* if

$$(A\text{-}1) \qquad |x_i - x_k| \leq o, \quad |y_i - y_k| \leq o, \quad 1 \leq k < i \leq K, \quad o \equiv 5(-14).$$

Program 6 is started by setting K = N and then by sensing if v_1 and v_N of the V-array, which specifies Π, *coincide*, 6-3. If they do, then v_1 replaces v_N in the V-array. If they do not coincide, then v_1 is added to V as v_{N+1} and K = N + 1. Before proceeding with the decomposition of Π, 6 determines whether V is to be processed in increasing order of its elements or in decreasing order, 6-2. *The resulting decompositions are not necessarily identical*, i.e., they may not isolate the same set of

polygons. Figure 36 contains an example. The two decompositions for that example are given near the end of this appendix, page A-16. Of course, the result for P(Π) must be independent of which decomposition is used.

The procedure used by SORT I is an N^2-process, whereas SORT III is an N-process.

We focus our attention on the forward decomposition, (ICV > 0), 6-4, rather than the reverse procedure, (ICV < 0), 6-5.

The array V of data points is searched for a point v_k, $1 \leqslant k < i$, which *coincides* with v_i, starting with i = 2, i.e., where v_i and v_k satisfy (A-1). If v_i and v_k coincide, $1 \leqslant k < i$, then set IST = k and IEN = i, with $2 \rightarrow i \rightarrow L$ if k = 1, otherwise $k \rightarrow i \rightarrow L$, 6-4. In 6-6, M is set to NUMI; the inequality IEN − IST = NUMI ⩽ 2 is tested. If NUMI = 1 or NUMI = 2, then we have isolated either a set of SDP or a set of SCP, respectively. In either case, such elements do not contribute to P. Hence we set $p^5 = 0$, and a call to VALR-7, 6-9, can be averted. If the inequality is not satisfied, then an element of {S} or $\{\bar{S}\}$ has been isolated. In order to determine whether it is in {S}, i.e., if the α-option has been specified, a sensing on ICV is carried out at 6-10. If the answer is yes at this box, then the isolated element is assumed to be in {S}, (α-option), and SORT II, 6-13, is not called. If the answer is no, then SORT II will be called, 6-13, since it is assumed in this case that the isolated element is in $\{\bar{S}\}$, (β-option). In 6-13, the inequality NUMI ⩽ 2 is checked again, after SORT II has been used. It could happen that after deletions by SORT II, the isolated element $\bar{S}$ retains no more than 3 points, so that the inequality NUMI ⩽ 2 would be satisfied. Then p = 0, and VALR-7, 6-9, is bypassed; the program proceeds directly to 6-8. If the inequalities of 6-6 and/or 6-13 are not satisfied, then VALR-7 is called to compute P and A for the isolated element, which we denote here by p and a, respectively.

Following the computation of p and a, a query is made at 6-8. Is IEN = K? If the answer is no, II requires further processing, which is carried out after replacing elements L, ..., K − M of V by elements (L + M), (L + M + 1), ..., K, with K then reset to K = K − M as noted in 6-7. The replacement begins at L rather than L + 1 because (A-1) may also be satisfied by (x_L, y_L) and some point (x_m, y_m), where $m \leqslant L$. Hence, at this stage, V is reduced and closed-up (CU) for further processing. Control is returned to 6-2 and the search continues through the updated V-array, starting with i = k, or i = 2 if k = 1 for more "duplicate" points, 6-4.

If at 6-8, IEN = K, then we must have k = 1 (IST = 1) when ICV > 0, since v_1 and v_K are always the same. Thus, in this case, 6 proceeds from 6-12 to EXIT, 6-14, and return of control to P-7. If, on the other hand, ICV < 0, i.e., processing of V is from N − 1 to 1, 6-5, and IEN = K at 6-8, and IST = j ≠ 1 at 6-12, then an element has been isolated which is specified by (j, j + 1, ..., K). Consequently 6-7 can be bypassed with V reduced and CU by simply resetting $x_K \rightarrow x_{IST}$, $y_K \rightarrow y_{IST}$, and IST → K at 6-11. Figures 40, 52 contain examples of where this would occur.

SORT II, 7, is now considered in more detail with the aid of its flow chart 7 (page A-19). Recall, when the β-option is used, that this subroutine is called by SORT I to delete nodes, from an

[5] We use p and a for an isolated element and retain P and A for P(II) and A(II).

isolated element of $\{\bar{S}\}$ which contains SCP. They are detected, within rounding error, by testing the inequality (see page 28),

(A-2) $$|s| = |\sin \Delta\theta| \leqslant \omega, \quad \omega = 7(-14),$$

where s can be determined algebraically from the 3 points specifying the angular region (see (47) on page 28; 7-3, 12, 17). Recall also, that if (A-2) holds, then it is possible $|\Delta\theta|$ is near zero rather than π. In this case, although the angular region is well defined, (WD), the vertex node is deleted since the angular region does not contribute to p or a. Angular region a_1 in Figure 32 is an example.

It is assumed now that an isolated element $\bar{S}$ of $\{\bar{S}\}$ is available through the decomposition of Π by SORT I. We assume $\bar{S}$ is specified by an array τ of M coordinate points. Two integer-valued parameters k and m are introduced in 7-2 with k = 1, m = 2. Parameter k is associated with the vertex point (k) of the angular region a_k under consideration. The parameter m refers to the point of a_k following (k). It is denoted by (m). Initially m is set to k + 1, 7-2, 7-9. If, however, a_k specified by (k − 1, k, m) subtends an angle $\Delta\theta$ such that (A-2) holds, then m takes successive values above k + 1 until (A-2) is not satisfied or m = M, 7-4, 7-11, 7-16, 7-18.

The quantities u, v, D_2^2 and w, z, D_1^2 are computed initially at 7-2 and 7-3, respectively. (The quantities D_1 and D_2 are defined on page 39.) Then (A-2) is checked at 7-3 for a_1. If it holds, then m = m + 1, 7-4, with a return to 7-3 to compute new values of w, z, D_1^2. This is continued until (A-2) does not hold or m = M. If m = M a return is made, 7-5, to SORT I, box 13, with NUMI = 2. Hence, $p(\bar{S}) = 0$ for this particular isolated element $\bar{S}$, since it is a straight line within the tolerance ω of (A-2).

Assuming this does not occur, 7 proceeds to 7-7 with I = 2 and a query: Is m = 2? If m ≠ 2, then points (M), (1), ..., (m − 1) were found to be colinear, i.e., each 3 successive points generate an angular region for which (A-2) holds. In this case, the original array τ is reduced and CU by replacing elements of τ starting at (1) by elements (m − 1), (m − 2), ..., (M) and M is reset to M = M − (m − 2), 7-8. The program proceeds to 7-9, where k = 2, m = 3, and new values of w, z, and D_1^2 are computed. Then 7 would proceed to 7-12.

If m = 2 at 7-7, then a_1 is WD and 7 proceeds to 7-6, without disturbing τ, with m = 3. Proceeding to 7-12, new values of u, v, D_2^2 are computed, where as noted above k = 2, m = 3. (Observe that at this stage w, z, D_1^2 from 7-3 are based on k = 1, m = 2, and therefore have the correct values for looking next at a_2.) The program is now set to look at a_2, where the subscript refers to element (2) of the updated τ array.

At 7-12 (A-2) is checked. If it holds, m = m + 1 in 7-18 and a return is made to 7-12, where new values of u, v, D_2^2 are computed and (A-2) is checked again. This is continued until (A-2) is not satisfied or m = M + 1. If (A-2) is not satisfied for some m, $3 \leqslant m \leqslant M$, then from 7-12, the program proceeds to 7-13. If m = k + 1, then a_2 is WD, no alterations are made to τ, k = k + 1, and if k < M, the program goes from 7-16 to 7-15, where D_2^2, u, v are used for the new values of D_1^2, w, z, respectively, m = k + 1, and a return is made to 7-12. The angular region a_3 is now investigated as was done previously with a_2. If k = M, then 7 goes from 7-16 to 7-17 to process a_M.

If at 7-13 $m \neq k + 1$, then elements of τ starting at (k) are replaced by elements $(m - 1)$, (m), ..., (M), with M reset to $M = M - (m - k - 1)$, 7-10. The program proceeds to 7-11, where $k = k + 1$. If $k < M$, 7 returns to 7-9 and is ready to look at the next angular region. If $k = M$, then 7 proceeds from 7-11 to 7-14 to treat the last angular region a_M.

An answer of no to the query at 7-18 implies a_k is made up of points $(k - 1, k, M)$; consequently all the points $(k - 1)$, (k), ..., (M) taken as successive triplets $(k - 1, k, k + 1)$, $(k - 1, k, k + 2)$, ..., $(k - 1, k, M)$ are SCP, i.e., satisfy (A-2). Therefore points $(k + 1)$, ..., $(M - 1)$ are ignored, with the M^{th} point replacing the k^{th} point and M reset to k, 7-19. It remains to process a_M. For this, we go to 7-14.

Note that when 7 goes from 7-16 to 7-17 to compute w, z, D_1^2 for a_M that u, v, D_2^2 are already available from processing a_{M-1}.

If a_M satisfies (A-2), 7-17 then (M) is dropped from τ by setting $M = M - 1$, 7-20 and control is returned to SORT I. If (A-2) does not hold, then control is returned directly to SORT I. The final coordinates, as contained in the τ array, at exit, and the number of them are specified on the flow chart of 7 as output $\bar{x}$, $\bar{y}$, $\bar{M}$, respectively.

By processing polygons Π_1 and Π_2 of Figures 32 and 33, respectively, a more detailed description of SORT I and SORT II is given. We assume the β-option numbering scheme, in order to bring SORT II into play for Π_1. Also, Π_1 and Π_2 are processed in the order of increasing numbered nodes, starting with node 1. Thus ICV = 3 (see P-7, page (A-17)). The descriptions are presented in tabulated form in the same way as was done for SORT III (page 34). Each node in the tabulation will be identified by its number in the original V-array specifying the given polygon.

The first column, on page A-12, contains the value of N, the number of elements in V, when SORT I, 6, is involved, and it contains the value of M, the number of elements in the τ array when SORT II, 7, is operating. The τ array specifies an isolated element S from the decomposition procedure by 6. The second and third columns refer to integers i and k, and k and m of the preceding discussions on 6 and 7, respectively. The fourth column displays the boxes used, by their numbers on the flow charts, in the order they come into play. Column four, when referring to 6, also shows the particular $\bar{S}^1$ isolated at that stage. Column five, when referring to 7, shows the points deleted from each of the $\bar{S}^1$ as a result of SCP(s). The numerical data, in column five, preceded by a letter is associated with a subsequent column headed by the same letter which shows the reduced CU V or τ-arrays at particular stages of the programs. The sixth column, headed V_1, refers to the original V-array. The seventh column, headed $\bar{S}_1$, refers to the original τ array for the first isolated element $\bar{S}_1$. Subsequent elements isolated by 6 have their initial τ arrays listed under columns $\bar{S}^2$ and $\bar{S}^3$. Columns headed (b), (c), etc., refer to the reduced compacted arrays as determined by 7, for $\bar{S}^1$, $\bar{S}^2$ and $\bar{S}^3$. For example, for Π_1, 6 first isolates $\bar{S}^1$ given by (4, 5, 6, 7, 8, 9, 10). Then $\bar{S}^1$ is modified by deletion of (7, 8). The reduced compacted array returned to VALR-7 is listed in column headed (b). Numerical results of Π_1 are given, following its tabulation, at the end of page A-13.

For Π_2, Figure 33, SORT I decomposes it into 3 elements of the class $\{\bar{S}\}$, and a PAR, specified by: $\bar{S}^1 = (1, 2, 3, 4, 5, 6, 7)$, $\bar{S}^2 = (8, 9, 10, 11, 12)$, $\bar{S}^3 = (12, 13, 14, 15, 16, 17, 18)$, and

PROGRAMS 6 AND 7 FOR Π_1 FROM FLOW CHARTS BASED ON FIGURE 32

BOXES					Compacted V and τ-Arrays							
N	i	k	SORT I	Points Deleted	V_1	$\bar{S}^1$	(b)	V_2	$\bar{S}^2$	(c)	(d)	V_3
22	10	4	3, 2, 4		1	4	4	1	10	10	10	1
			$\bar{S}^1$ isolated by 6		2	5	5	2	11	12	12	2
			6, 10, 13 (Call 7)		3	6	6	3	12	13	13	3
M	k	m	SORT II		4	7	9	10	13	14	14	17
6	1	2	2, 3		5	8	10	11	14	15	16	18
	2	2, 3	7, 6, 12, 13		6	9		12	15	16	17	19
	3	3, 4	16, 15, 12		7	10		13	16	17		20
	3, 4	4, 5	13, 16, 15, 12		8			14	17			21
	4	6, 7	18, 12, 18		9			15				22
4	4	7	19, 14, 17, 21	(b): (7), (8)	10			16				23
N	i	k	SORT I		11			17				
			9, 8, 7	4-9 of V_1	12			18				
16	5	4	2, 4		13			19				
			6, 10, 13		14			20				
			$\bar{S}^2$ isolated by 6		15			21				
M	k	m	SORT II		16			22				
7	1	2	2, 3		17			23				
	2	2, 3	7, 6, 12		18							
	2	4	18, 12, 13, 10	(c): 11	19							
6	3	4	11, 9, 12, 13		20							
	4	4, 5	16, 15, 12, 13		21							
	5	5, 6	16, 15, 12		22							
6, 5	5	6, 7	18, 19, 14, 17	(d): 15	23							
N	i	k	SORT I									
			9, 8, 7	10-16 of V_2								

V_1: Denotes original V-array for Π_1.

V_2, V_3: Denote reduced compacted arrays from $V = V_1$.

PROGRAMS 6 AND 7 FOR Π_1 FROM FLOW CHARTS (Continued)
BASED ON FIGURE 32

BOXES					Compacted V and τ-Arrays				
N	i	k	SORT I	Points Deleted	$\bar{S}^3$	(e)	(f)	(g)	(h)
	5	4	(V_3): 1–3, 17–22		1	2	2	2	2
9	10	1	2, 4, 6		2	3	17	17	17
			$\bar{S}^3$ isolated by 6		3	17	18	20	20
			10, 13		17	18	19	21	21
M	k	m	SORT II		18	19	20	22	2
9	1	2	2, 3		19	20	21	2	
9	1	3	4, 3		20	21	22		
8	2	3	7, 8, 9, 12	(e): 1	21	22	2		
8	2	4	18, 12, 13		22	2			
7	3	4	10, 11, 9. 12	(f): 3	23				
7	3	5	18, 12						
7	3	6	18, 12, 13						
5	4	5	10, 11, 9, 12, 13	(g): 18, 19					
5	5	5	16, 17						
4	5	5	20, 21	(h): 22					
N	i	k	SORT I						
4	5	1	9, 8, 12, 14						
			EXIT TO P-7						

$\bar{S}^4$ = (1, 18, 19). It is worth noting that although the β-option was used, SORT II is not needed. This is so because $\bar{S}^1$, $\bar{S}^2$ and $\bar{S}^3$ are actually in (S) and $\bar{S}^4$ has only three points with zero area (NUMI $\leq$ 2 in 6-6). Consequently, SORT II can be bypassed by setting ICV = 2. The tabulation for Π_2 is given with ICV = 2.

The 3 final polygons resulting from the decomposition of Π_1 by SORT I and the removal of PAR(s) by SORT II are listed under columns (b), (d) and (h). VALR-7 yields values of p and a for each of these polygons, namely, $p(\bar{S}^1) = -.7078\ 0769$, $a(\bar{S}^1) = -25$, $p(\bar{S}^2) = .0268\ 8323$, $a(\bar{S}^2) = 22$, $p(\bar{S}^3) = .0125\ 8574$, $a(\bar{S}^3) = 13.5$. Thus $P(\Pi_1) = -.6683\ 3872$, $A(\Pi_1) = 10.5$. It is interesting to note that $A > 0$ but $P < 0$. In Figure 53, $A < 0$ and $P > 0$.

PROGRAM 6 FOR Π_2 FROM FLOW CHARTS
BASED ON FIGURE 33

BOXES					Compacted V-Arrays and τ-Arrays						
N	i	k	SORT I	Points Deleted	V_1	S^1	V_2	S^2	V_3	S^3	V_4
18	7	1	3, 2, 4		1	1	1	8	1	12	1
			S^1 isolated by 6		2	2	8	9	12	13	18
	(M = 6)		6, 10, 9		3	3	9	10	13	14	19
12	2	1	8, 7	2-7 of V_1	4	4	10	11	14	15	
	6	2	2, 4		5	5	11	12	15	16	
	(M = 4)		S^2 isolated by 6		6	6	12		16	17	
			6, 10, 9		7	7	13		17	18	
8	3	1	8, 7	8-11 of V_2	8		14		18		
	8	2	2, 4		9		15		19		
	(M = 6)		S^3 isolated by 6		10		16				
			6, 10, 9		11		17				
2	3	1	8, 7	12-17 of V_3	12		18				
	(M = 2)		2, 4, 6		13		19				
			8, 12, 14		14						
			EXIT TO P-7		15						
					16						
					17						
					18						
					19						

The decomposition of Π_2 (Figure 33) by SORT I results in 3 polygons, S^1, S^2, S^3. Since all the angular regions of these polygons are well defined, SORT II is not needed. The final array V_4 consists of a singular angular region; for this region the program proceeds from 6-6 directly to 6-8 setting p = 0 and then exiting. VALR-7 yields the values $p(\bar{S}^1)$ = .8308 6076, $a(\bar{S}^1)$ = 18; $p(\bar{S}^2)$ = .5378 8935, $a(S^2)$ = 6.5; $p(\bar{S}^3)$ = −.4271 2530, $a(S_3)$ = −4.0. Hence $P(\Pi_2)$ = .9416 2481, $A(\Pi_2)$ = 20.5, (see page 48).

Letting $\bar{S}$ denote the element shown in Figure 36, we list, on page A-16, $P(a_k)$ for each angular region a_k, k = 1, 2, ..., N(=22). The computations were carried out with IOP = 3 (from P-7), i.e.,

with 9-decimal-digit accuracy for $P(a_k)$, for each k = 1, 2, ..., N. The program was run with ICV = 0, i.e., P-7 called only VALR-7 to evaluate $P(\bar{S})$. Subsequently, the program was also run with ICV = 2 and −2. Recall that when ICV = 2, P-7 calls SORT I which decomposes Π (=$\bar{S}$ here) into a set of simple disjoint polygons $\{S^i\}$, (in this case); SORT I, in turn, calls VALR-7 to evaluate $p(S^i)$ for each isolated element, S^i, of the decomposition. The decomposition by SORT I starts at point (1) of $\bar{S}$, and $\bar{S}$ is processed from (2) to (N), sequentially. When ICV = −2, the procedure starts at point N of $\bar{S}$ and carries out the decomposition in the backward direction, i.e., sequentially from (N − 1) to (1). Observe in the tabulation that the decompositions with ICV = 2, and ICV = −2, are not the same, although, of course, the final results for $P(\bar{S})$ and $A(\bar{S})$ must be identical within the accuracy specified.

The first column of the tabulation shows the node number of $\bar{S}$; the second and third columns give the xy-coordinate values associated with the node number. The fourth column, headed ICV = 0, lists $P(a_k)$ for each node number (k) of the first column, k = 1, 2, ..., N. Summing the $P(a_k)$, and using (A-3), below gives $P(\bar{S})$ beneath columns 1-4. The next two columns refer to finding $p(\bar{S})$ with ICV = 2. The fifth column contains the node numbers for each isolated S^i, i = 1, ..., 6; the next column, headed ICV = 2, contains the $P(a_k)$ associated with the node numbers of a particular S^i. At the end of the listing for each S^i, $p(S^i)$ is given. For example, S^3 is specified by (13, 14, 15, 16); the value of P for the angular region of S^3 at node 13 is 8.1418 04138 × 10^{-1}. The value of $p(S^3)$ = 7.3866 73215 × 10^{-2}. The 7th and 8th columns refer to nodes and corresponding angular regions with ICV = −2. For example, S^4 in this case, is specified by (4, 5, 6, 7, 12, 13, 17, 18) with $P(a_7)$ of S^4, with ICV = −2, given by −2.0268 86540 × 10^{-1} and $p(S_4)$ = −9.1654 62410 × 10^{-1}. The results were checked by using an independent decomposition procedure, with Drezner's method [2], described on page 47.

It was shown in Section III, ((24) and (26)) that

(A-3) $$P(S) = 1 - \Sigma P(a_k), \quad \text{if} \quad A(S) > 0,$$

(A-4) $$P(S) = -1 - \Sigma P(a_k), \quad \text{if} \quad A(S) < 0, \quad \text{(See pages 10-12)}$$

We note that for ICV = −2, $a(S^1)$, $a(S^4)$ are negative, (their values are given in the lower right-hand corner of page A-16), i.e., S^1 and S^4 are negatively oriented, (NO), and therefore $p(S^1)$ and $p(S^4)$ are also negative. Note again, that the decompositions for ICV = 2 and ICV = −2 are different. ICV = 0 can be used, because $\bar{S}$ has no SAR. It is generally preferred when no SAR's occur for $\bar{S}$, since it does not use SORT I nor SORT III and is therefore more efficient.

Summarizing here, we can say that (P−B) is significantly better than (P−A) for complex SI polygons in the following ways:

(1) Great care must be exercised when using the β-option with (P−A) as the example in Figures A-5 and A-6 shows. Figure 58 is another example, where the numbering shown while appropriate for (P−B), since VALR-2 handles any polygon, is inadequate for (P−A) for the same reason as for Figure A-6.

(2) Program B, based on (P−B), is generally more efficient than the Program A based on (P−A), i.e., VALR-7 with SORT I and II, because often fewer points are needed to specify II as in Figure 58, and in addition, (P−A) uses an N^2 process to decompose II (SORT I), which is relatively slow.

TABULATION OF RESULTS FOR FIGURE 36

Node	x	y	ICV = 0, $P(a_k)$
1	−5	−5	2.8665 15719 (−7)
2	5	−5	2.8665 15719 (−7)
3	5	5	2.8665 15719 (−7)
4	−5	5	9.9986 26366 (−1)
5	3	3	−1.3449 27576 (−4)
6	5	−5	−1.3736 33819 (−4)
7	−3	−3	−9.9864 95777 (−1)
8	3	−3	1.3480 75949 (−3)
9	3	3	9.9069 82439 (−1)
10	2	−2	−8.5731 82606 (−3)
11	−3	−3	−2.0268 94695 (−1)
12	−1	0	1.1979 54136 (−1)
13	−2	2	−1.8536 42054 (−1)
14	0	1	1.0265 18201 (−1)
15	3	3	9.3010 33998 (−3)
16	−2	2	−6.2292 55125 (−4)
17	−2	3	1.6245 62987 (−5)
18	−5	5	8.2758 92935 (−2)
19	−3	2	8.9450 71843 (−1)
20	−2	2	−9.6857 13286 (−1)
21	−3	−3	−1.3654 79275 (−4)
22	−5	5	2.8665 15719 (−7)
$P(\vec{S})$ = 1.6393 83633 (−1)			

ICV = 0

$$P(\vec{S}) = 1 - \sum_{1}^{22} P(a_k), \quad A(\vec{S}) = 56.0$$

ICV = 2

$$P(\vec{S}) = \sum_{1}^{6} p(S^1), \quad A(\vec{S}) = \sum_{1}^{6} a(S^1)$$

ICV = −2

$$P(\vec{S}) = \sum_{1}^{5} p(S^1), \quad A(\vec{S}) = \sum_{1}^{5} a(S^1)$$

Node	ICV = 2, $P(a_k)$
2	2.8665 15719 (−7)
3	2.8665 15719 (−7)
4	9.9986 26366 (−1)
5	−1.3449 27576 (−4)
$p(S^1)$	2.7128 28364 (−4)*
7	1.3496 06848 (−3)
8	1.3480 75949 (−3)
9	9.9069 82439 (−1)
10	−8.5731 82606 (−3)
$p(S^2)$	1.5177 25594 (−2)
13	8.1418 04138 (−1)
14	1.0265 18201 (−1)
15	9.3010 33998 (−3)
$p(S^3)$	7.3866 73215 (−2)
16	2.2232 56327 (−2)
17	1.6245 62987 (−5)
18	8.2758 92935 (−2)
19	8.9450 71843 (−1)
$p(S^4)$	4.8507 74211 (−4)
11	7.9730 92908 (−1)
12	1.1979 54136 (−1)
20	9.0285 63470 (−3)
$p(S^5)$	7.3866 73215 (−2)
1	2.8665 15719 (−7)
6	9.9986 26366 (−1)
21	−1.3449 27576 (−4)
22	2.8665 15719 (−7)
$p(S^6)$	2.7128 28364 (−4)
$a(S^1) = 20$, $a(S^2) = 6$, $a(S^3) = 3.5$	
$a(S^4) = 3$, $a(S^5) = 3.5$, $a(S^6) = 20$	

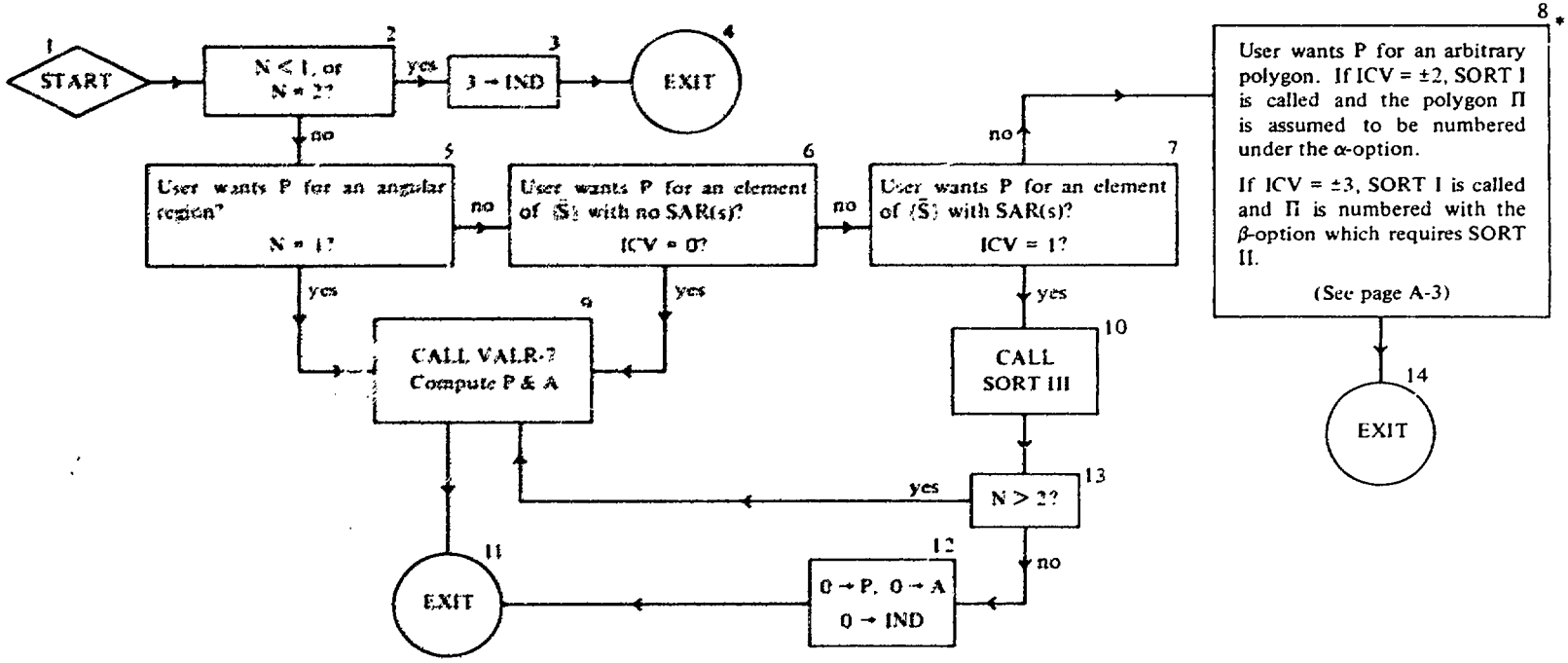
Node	ICV = −2, $P(a_k)$
18	−91724 10707 (−1)
19	8.9450 71843 (−1)
20	−9.6857 13286 (−1)
21	−1.3654 79275 (−4)
$p(S^1)$	−8.5582 37140 (−3)*
13	8.1418 04138 (−1)
14	1.0265 18201 (−1)
15	9.3010 33998 (−3)
$p(S^2)$	7.3866 73215 (−2)
7	1.3496 06848 (−3)
8	1.3480 75949 (−3)
9	9.9069 82439 (−1)
10	−8.5731 82606 (−3)
$p(S^3)$	1.5177 25594 (−2)
4	−1.3736 33819 (−4)
5	−1.3449 27576 (−4)
6	−1.3736 33819 (−4)
7	−2.0268 86540 (−1)
12	1.1979 54136 (−1)
13	−1.6754 46491 (−4)
17	1.6245 62987 (−5)
$p(S^4)$	−9.1654 62410 (−1)
1	2.8665 15719 (−7)
2	2.8665 15719 (−7)
3	2.8665 15719 (−7)
4	2.8665 15719 (−7)
$p(S^5)$	9.9999 88534 (−1)
$a(S^1) = -7.5$, $a(S^2) = 3.5$, $a(S_3) = 6$	
$a(S^4) = -46$, $a(S^5) = 100$.	

*Note: See (A-3) and (A-4) of page (A-15).

FLOW CHART 5
P-7 MASTER PROGRAM FOR (P − A)

Input: x, y, N, ICV, IOP

Output: P, A, IND



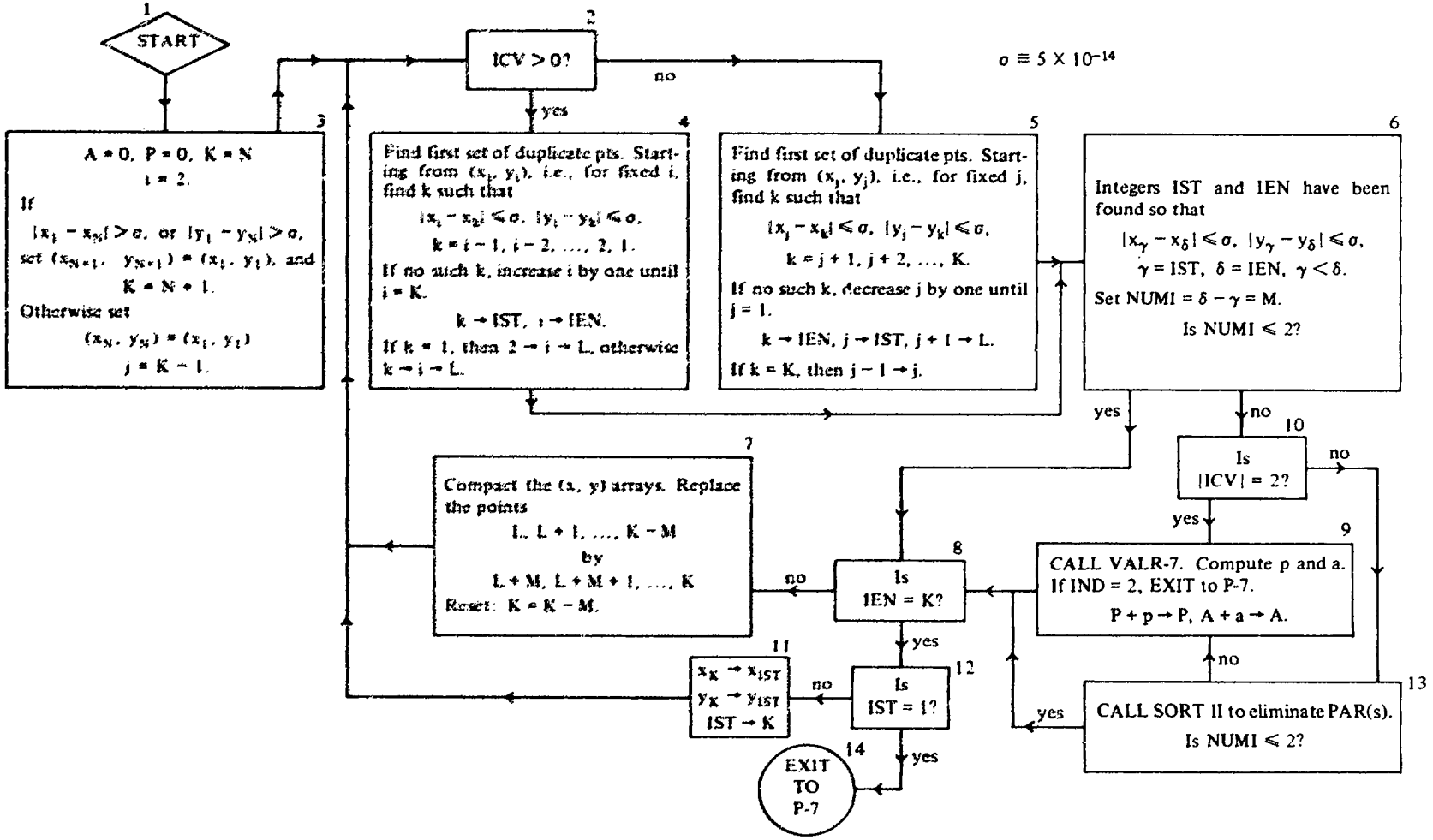
*When SORT I is called, it processes the arbitrary polygon Π, decomposing it into elements $S^1, \ldots, S^J$ with α-option, or $\bar{S}^1 \ldots, \bar{S}^L$ with β-option. VALR-7 is called to compute $P(S^1)$ or $P(\bar{S}^1)$ and sums the results to get $P(\Pi)$. In case of $\bar{S}^1$, SORT I calls SORT II to remove SCP. Details in Flow Chart 7.

FLOW CHART 6

Subroutine (SORT I)

Input: x, y, N, ICV, IOP

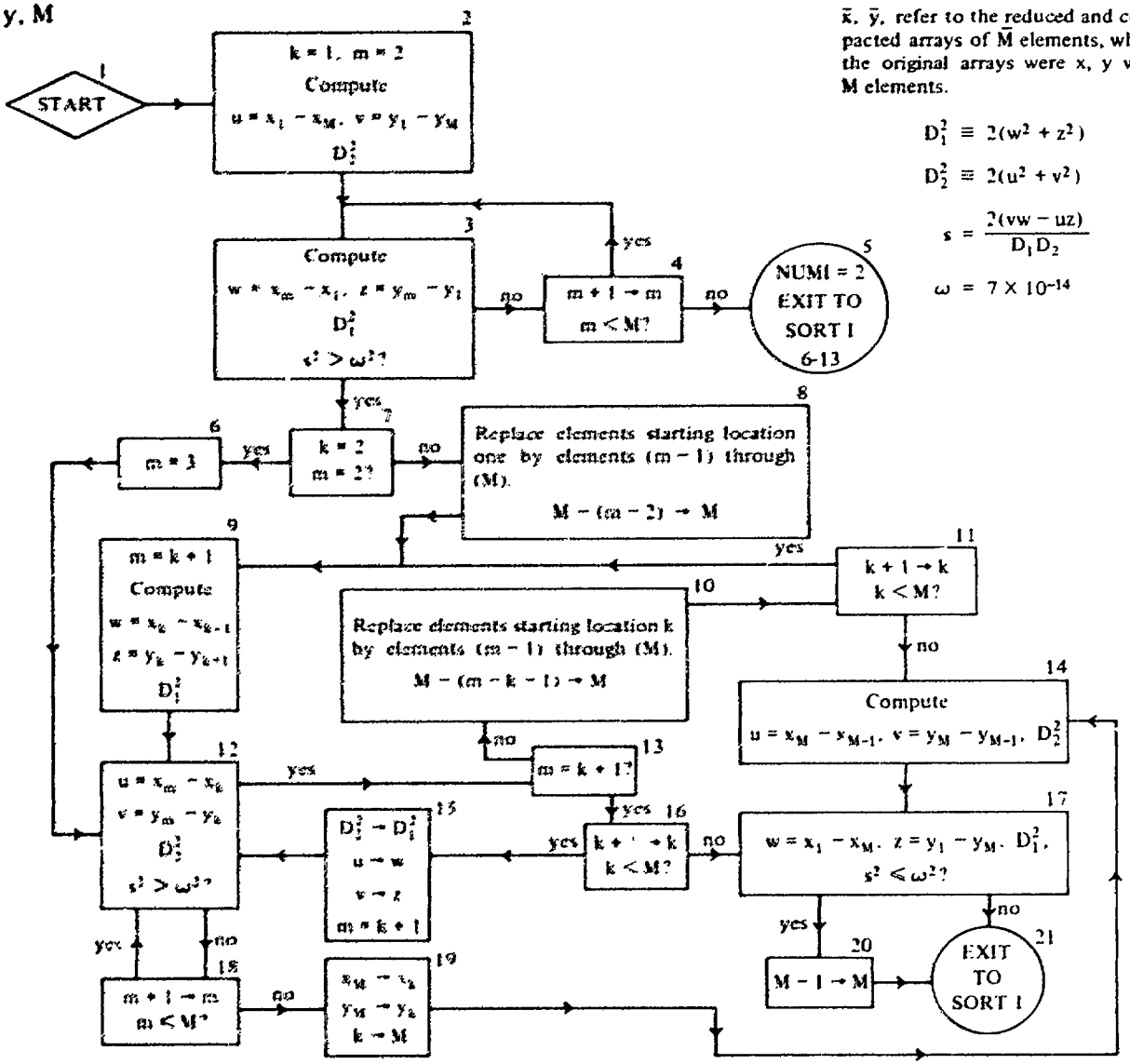
Output: P, IND, A



FLOW CHART 7

Subroutine (SORT II)

Input: x, y, M



$\bar{x}$, $\bar{y}$, refer to the reduced and compacted arrays of $\bar{M}$ elements, where the original arrays were x, y with M elements.

$$D_1^2 \equiv 2(w^2 + z^2)$$

$$D_2^2 \equiv 2(u^2 + v^2)$$

$$s = \frac{2(vw - uz)}{D_1 D_2}$$

$$\omega = 7 \times 10^{-14}$$

Output: $\bar{x}$, $\bar{y}$, $\bar{M}$

APPENDIX B
EVERY SIMPLE POLYGON CONTAINS AN INTERIOR DIAGONAL

APPENDIX B

EVERY SIMPLE POLYGON CONTAINS AN INTERIOR DIAGONAL

By an interior diagonal, we mean the open segment (L) of the closed line segment [L] extending from some vertex (k) to some nonadjacent vertex (j) of the simple polygon S, such that (L) is entirely in the interior of S.

The proof given in [4, p. 17, Lemma 2] is not correct. Knopp's proof is repeated here with a counter-example. An argument to correct the proof is then given.

Let a straight line which does not intersect or meet S be translated parallel to itself toward the polygon until they meet. Then the line necessarily contains a vertex A of S with the interior angle of A less than two right angles. Let B and C denote the adjacent vertices to A. Then one of the following is true:

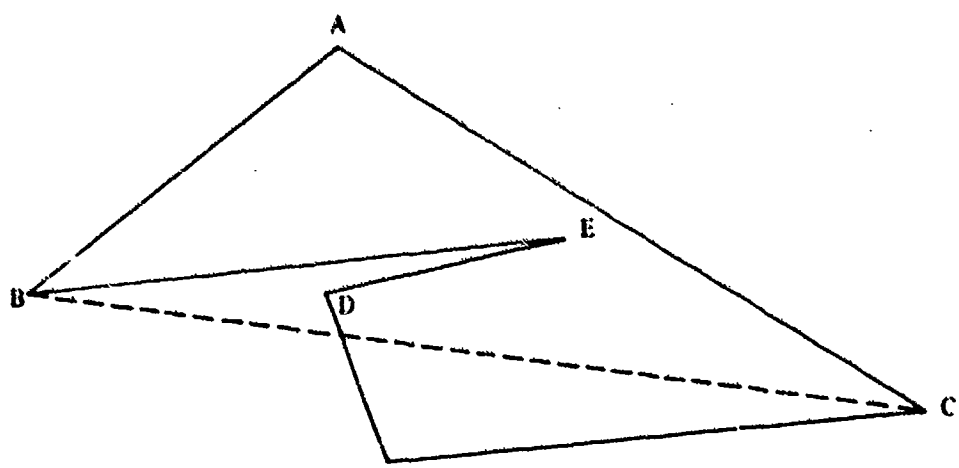
(1) BC is a diagonal lying in the interior of S.

(2) There is at least one vertex of S on the (open) segment BC (let one of these vertices be denoted by V) but no vertex in the interior of triangle ABC, (Δ ABC).

(3) There is at least one vertex of S in the interior of Δ ABC.

If (1) is true, there is nothing further to show. If (2) holds, then AV is an interior diagonal of S. If (3) is true, let a point X move from B to C along BC until AX encounters a vertex or vertices of S in the interior of Δ ABC. If V denotes that one of these vertices which is nearest to A, then AV is a diagonal interior to A.

The proof of part (3) is not correct. This is easily seen from the figure below. The vertex nearest to A in Δ ABC, following Knopp, is D, but the line AD contains points outside S. The proper vertex to have chosen was E which is in Δ ABC, but *farther* from A than D.



A Counter-Example to Knopp's Proof

The proof is easily corrected as follows: Starting at A, move an open segment, which extends from AB to AC, parallel to BC and towards BC until one or more vertices of S are met. If there is more than one such vertex, choose any one and call it V. Vertex V has the property that no other vertex of S in Δ ABC has, a greater (perpendicular) distance from BC. Now suppose AV is not an interior diagonal of S. Then there exists a point (z) where AV intersects another side of S, say side $\overline{(k, k+1)}$. Point (z) cannot be a vertex by the way V was chosen. Now either vertex (k) or (k + 1) must have at least as great a distance from BC as (z). Say it is (k). But, since S is simple (k) must be in the interior of Δ ABC. This contradicts the way V was chosen. Hence AV must be an interior diagonal of S.

This result is used on page 11 and in Appendix D.

APPENDIX C
AN ALTERNATIVE METHOD TO FIND P FOR SIMPLE POLYGONS

APPENDIX C
AN ALTERNATIVE METHOD TO FIND P FOR SIMPLE POLYGONS

At an early stage of our studies, we developed a method to compute the P-function over a simple polygon by using a program already available, which computed P for convex polygons, [2]. To put it another way, an automatic procedure was set up to represent any simple polygon by a finite set of convex polygons. Realization of our working program required, in addition to a program for computing P for convex polygons, a program to determine the convex hull of a finite point set in the plane. Such a program was available from previous work, [1]. By the convex hull $C(Z_N)$ of the point set $Z_N = \{(x_j, y_j), j = 1, 2, \ldots, N\}$, we mean the smallest convex polygon which contains all of Z_N. The vertices of $C(Z_N)$ are in Z_N.

A simple polygon S is shown in Figure C-1. We set forth the procedure by applying it to this polygon. It will be apparent to the reader that any N-sided simple polygon can be handled in the same way.

Procedure:

(A) Find the convex hull C of S. We obtain

$$C = (1, 2, 3, 6, 7, 13, 14).$$

The P-function, P(C), for C is computed by the program for evaluating P for convex polygons. Clearly since S and C are positively oriented, PO, we have

(C-1) $$0 < P(S) < P(C).$$

(B) The set of vertices of C is searched to determine which vertices of S are missing between adjacent vertices of C. Obviously, vertices (4) and (5) between (3) and (6), and vertices (8), (9), (10), (11), (12) between (7) and (13) are missing from C. In this way, we isolate 2 simple polygons from C, namely

$$S_1 = (3, 4, 5, 6, 3), \quad S_2 = (7, 8, 9, 10, 11, 12, 13, 7).$$

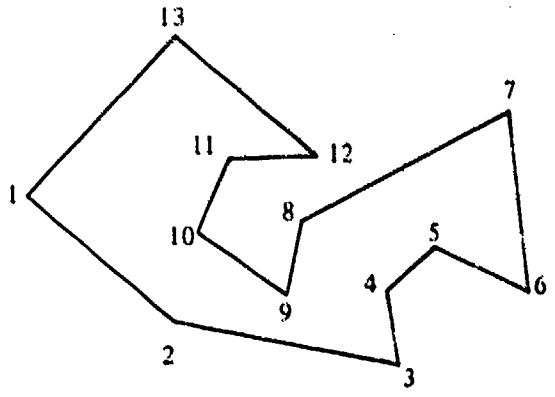


Figure C-1. A 13-Sided Simple Polygon, S

(C) The convex hull C_1 of S_1 is found to be identical to S_1. (When this occurs, that convex hull requires no further processing.) We note C_1 is negatively oriented so that $P(C_1) < 0$. The convex hull C_2 of S_2 is then determined to be

$$C_2 = (7, 9, 10, 13, 7),$$

as indicated in Figure C-2 by the dotted lines and the line segment (9, 10). C_2 is also NO, and hence $P(C_2) < 0$. Therefore, consideration of C, C_1, C_2 shows

(C-2) $$P(S) > P(C) + P(C_1) + P(C_2).$$

(D) Two simple polygons, both PO, are obtained from C_2, by noting the missing vertices, as explained in (B), namely

$$S_3 = (7, 8, 9, 7), \quad S_4 = (10, 11, 12, 13, 10).$$

(E) The convex hull C_3 for S_3 is again S_3. Thus, C_3 requires no further processing. Since C_3 is PO, we obtain $P(C_3) > 0$. Next, the convex hull C_4 for S_4 is found to be

$$C_4 = (10, 12, 13, 10)$$

which is also PO. Thus $P(C_4) > 0$ and we obtain, adding $P(C_3)$ and $P(C_4)$ to the right-hand side of (C-2),

(C-3) $$P(S) < P(C) + P(C_1) + P(C_2) + P(C_3) + P(C_4).$$

(F) Finally we isolate S_5 from C_4,

$$S_5 = (10, 11, 12, 10).$$

Its convex hull C_5 is identical to it. Observing that C_5 is NO and that $P(C_5) < 0$, we obtain the final results by adding $P(C_5)$ to the right-hand side of (C-3), namely, with $C_0 \equiv C$,

$$\bigcup_{i=0}^{5} C_i = S, \quad P(S) = \sum_{i=0}^{5} P(C_i).$$

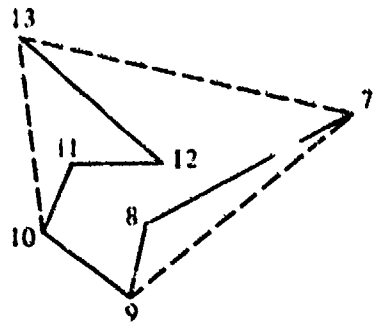


Figure C-2. Convex Hull, C_2, of $S_2 = (7, 8, 9, 10, 11, 12, 13, 7)$

Although no proofs are given the procedure can be put on a rigorous basis by induction type arguments.

This method, call it (O) (for old) is much slower in general than the procedure described in Section III, call it (N) (for new). This is so, because (in addition to finding convex hulls) the time consuming computation is finding P for an angular region by VALR-2 or VALR-7. By our present procedure, (N), we require the evaluation of P, in the example of Figure C-1, for 13 angular regions, whereas by the method of this appendix, (O), we require 6 angular regions for P(C), 4 for $P(C_1)$, 4 for $P(C_2)$, 3 for $P(C_3)$, 3 for $P(C_4)$ and 3 for $P(C_5)$ for a total of 23 angular regions.

The method (O) does have an advantage in that P(S) is alternately bounded above and below with improved bounds on each cycle of positive and negative contributions to estimating P(S). By a cycle, we mean a stage in (O) where each convex hull obtained is of the same orientation. The first cycle occurs with C is found. It is PO. At the second stage C_1 and C_2 are found and both are NO. The third stage is manifested by the appearance of C_3 and C_4, both PO. The fourth and final stage is when C_5 is found. It is NO. The improved bounds may allow the calculation for P(S) to be terminated early. Indeed, if at the end of any cycle of NO convex polygons, the last denoted by C_J, the quantity

$$\sum_{i=0}^{J} P(C_i) \tag{C-5}$$

is greater than $1 - \epsilon$, then $P(S) = 1$ within ϵ; if at the end of any cycle of PO convex polygons, the last denoted by C_J, the quantity corresponding to (C-5) is less than ϵ, then $P(S) = 0$ within ϵ.

APPENDIX D
EXPRESSIONS FOR THE AREA OF A POLYGON

APPENDIX D

EXPRESSIONS FOR THE AREA OF A POLYGON

We first show an expression for the area of a simple polygon in terms of vectors. There is nothing new about the result, but it is not as easily available as one would expect, [6]. Subsequently, by some straightforward algebraic manipulations we obtain an expression for the area, A, which leads to a very efficient machine computation for the area of a simple polygon, $A(\Pi)$.

Let S denote a simple polygon with its vertices numbered in the natural order from 1 to N, such that in tracing S continuously, the interior of S is always on the left. We say S is positively oriented (PO) in this case. The classical vector expression $\bar{A}$, for which $|\bar{A}(S)| = A(S)$, is then given by a sum of vector cross-products.

$$\text{(D-1)} \qquad \bar{A}(S) = \frac{1}{2}\sum_{i=1}^{N} (\bar{z}_i - \bar{Z}) \times (\bar{z}_{i+1} - \bar{Z}), \quad \bar{z}_{N+1} \equiv \bar{z}_1 ,$$

where $\bar{z}_i - \bar{Z}$ denotes the vector from $\bar{Z}$ to $\bar{z}_i$, with $\bar{Z}$ fixed, but arbitrary, and $A(S) \geq 0$ using the right-hand rule for cross products.

In order to establish (D-1) for simple polygons, we use induction, and the result of Appendix B that for any simple polygon S there exists a diagonal between two vertices of S which is entirely in S. We also need the fact that

$$\text{(D-2)} \qquad (\bar{z}_i - \bar{Z}) \times (\bar{z}_j - \bar{Z}) = -(\bar{z}_j - \bar{Z}) \times (\bar{z}_i - \bar{Z}).$$

Certainly for N = 3, S a triangle, (D-1) holds. Now assume (D-1) holds for all simple PO polygons with no more than N − 1 vertices. Let S denote a simple polygon of N vertices, PO. By the result of Appendix B, there exists a diagonal from vertex j to vertex j + k, $k > 1$, which is entirely within S, except for its end points at vertices j and j + k. This diagonal divides S into two disjoint simple PO polygons, except for the common side, each with no more than N − 1 vertices. Hence, by the induction hypothesis, (D-1) holds for each of these polygons, call them S_1 and S_2, where

$$S_1 = (1, 2, \ldots, j-1, j, j+k, j+k+1, \ldots, N, 1),$$

$$S_2 = (j, j+1, \ldots, j+k-1, j+k, j).$$

Hence

$$\text{(D-3)} \qquad \bar{A}(S_1) = \frac{1}{2}\left[\sum_{i=1}^{j-1} (\bar{z}_i - \bar{Z}) \times (\bar{z}_{i+1} - \bar{Z}) + (\bar{z}_j - \bar{Z}) \times (\bar{z}_{j+k} - \bar{Z}) + \sum_{i=j+k}^{N} (\bar{z}_i - \bar{Z}) \times (\bar{z}_{i+1} - \bar{Z})\right],$$

$$\text{(D-4)} \qquad \bar{A}(S_2) = \frac{1}{2}\left[\sum_{i=j}^{j+k-1} (\bar{z}_i - \bar{Z}) \times (\bar{z}_{i+1} - \bar{Z}) + (\bar{z}_{j+k} - \bar{Z}) \times (\bar{z}_j - \bar{Z})\right].$$

Since S_1 and S_2 are disjoint and PO, we have by adding (D-3) and (D-4), and using (D-2), the expression (D-1) for N-sided simple polygons.

Above, we have assumed S was PO. If S is NO, then each cross product in (D-1) is reversed, and by (D-2), A is given by (D-1) with a minus sign attached.

Now, since $\bar{A}(S)$ is a continuous function of the coordinates of the vertices of S, (D-1) also yields A for polygonal elements of $\{\bar{S}\}$, such as in Figures 35, 36, 45.

Actually (D-1) holds for arbitrary polygons, i.e., elements of $\{\Pi\}$. This follows by using the above results with a theorem given in [4, page 15], which states that every polygon can be decomposed into a finite set of polygons in $\{\bar{S}\}$.

An efficient expression for computing A can be obtained from (D-1). Since $\bar{Z}$ is arbitrary, choose $\bar{Z} = \bar{0}$. Then (D-1) reduces to

$$\text{(D-5)} \qquad \bar{A}(\Pi) = \frac{1}{2}\sum_{i=1}^{N} (\bar{z}_i \times \bar{z}_{i+1}).$$

In component form we have

$$\bar{z}_i \times \bar{z}_{i+1} \to x_i y_{i+1} - x_{i+1} y_i,$$

so that (D-5) can be written as

$$\text{(D-6)} \qquad A(\Pi) = \frac{1}{2}\sum_{i=1}^{N} (x_i y_{i+1} - x_{i+1} y_i), \quad (x_{N+1}, y_{N+1}) \equiv (x_1, y_1).$$

The number of multiplications can be halved by some algebra. From (D-6) take the second product of the $(i-1)^{st}$ term, $x_i y_{i-1}$, and combine it with the first product of the i^{th} term, $x_i y_{i+1}$ to obtain $x_i(y_{i+1} - y_{i-1})$. This can be done successively for each $i = 2, 3, \ldots, N$. The remaining elements, namely, the first product of the first term, $x_1 y_2$ and the second product of the N^{th} term, $x_1 y_N$ are combined to obtain $x_1(y_2 - y_N)$. Thus (D-6) becomes

$$\text{(D-7)} \quad A(\Pi) = \frac{1}{2}\sum_{i=1}^{N} x_i(y_{i+1} - y_{i-1}), \quad y_{N+1} \equiv y_1, \quad y_0 \equiv y_N. \quad \text{(See (22) and (46)).}$$

This expression appears in the text as (22) and (46).

A Fortran IV listing of the short program for computing A, SMP-7, is given in Appendix F, page (F-37).

The area of Π in the wz-plane, A(w, z), see page 1, is given by

$$\text{(D-8)} \qquad A(w, z) = \sigma_w \sigma_z (1 - \rho^2)^{1/2} A(\Pi).$$

APPENDIX E

PROGRAM PARAMETERS. CHEBYSHEV COEFFICIENTS, erfc (x)/z(x), x ≥ 0

APPENDIX E

PROGRAM PARAMETERS. CHEBYSHEV COEFFICIENTS, erfc (x)/z(x), $x \geqslant 0$

In this appendix, we list the constants that appear in the programs SORT I, SORT II, SORT III, VALR-2, and VALR-7. There are two constants, σ and ω, which depend only on the characteristics of the computer used. They are set at 5×10^{-d} and 7×10^{-d}, where d denotes the maximum number of decimal digits the machine uses to represent a real number. For our machine d = 14.

For VALR-2 and VALR-7, the additional parameters that appear are listed for 4 levels of accuracy, i.e., 3, 6, 9 and 12-decimal digits. The last is, at present, not incorporated into our programs, but it would be easy to do so. The values for all the parameters follow.

$\sigma = 5(-14) = 5 \times 10^{-14}$, in SORT I

$\omega = 7(-14)$ in SORT II and SORT III

$\omega = 7(-14)$ in VALR-2 and VALR-7

ADDITIONAL PARAMETERS FOR VALR-2, OR VALR-7

Acc.	ϵ	α_1	α_2	$(\alpha_3/4)$	α_4	$(\bar{R}/\sqrt{2})^2 = c^2$
Ⓐ	2.54 (-4)	2.02 (-7)	1.22 (-3)	5.625 (-5)	6.962 (-2)	6.05160
Ⓑ	2.57 (-7)	2.08 (-13)	1.23 (-4)	5.700 (-8)	6.990 (-3)	12.60605
Ⓒ	2.94 (-10)	2.71 (-19)	1.34 (-6)	6.512 (-11)	7.311 (-4)	19.201924
Ⓓ	1.00 (-13)	3.17 (-26)	6.58 (-9)	2.225 (-14)	5.111 (-5)	26.103925

$\epsilon = \delta/\sqrt{\pi}$	See page 6	$\alpha_3 = \sqrt{\pi}\,\epsilon/2$	See [2, page 15]
$\alpha_1 = \pi c^2$	See page 7	$\alpha_4 = \left(\frac{3}{2}\,\alpha_3\right)^{1/3}$	See page 29, Eq. (12) also,
$\alpha_2 = (9\alpha_1)^{1/3}$	See page 7	$\bar{R}^2/2$	See pages 6, 28 and [2, page 8]

The first column of the table labeled Acc. (for accuracy) lists Ⓐ Ⓑ Ⓒ Ⓓ referring to 3, 6, 9, 12-decimal-digit accuracy, respectively, for the probability over an angular region. Pages are indicated above where the parameters are discussed in the report.

The minimax coefficients, a_k, for approximating erfc (x)/z(x) on [0, c(δ)] (see page 6) are given below for the four accuracy levels associated with Ⓐ, Ⓑ, Ⓒ, Ⓓ as noted above. They were computed by a double precision minimax subroutine utilizing values of erfc (x) correct to 18 significant digits on (1/2, c(δ)] and values of erf (x) accurate to 25 digits on [0, 1/2].

For Ⓐ (Average time per angular region = 7.8×10^{-4} sec)

a_0 = .885777518572895D + 00
a_1 = −.981151952778050D + 00
a_2 = .759305502082485D + 00
a_3 = −.353644980686977D + 00
a_4 = .695232092435207D − 01

For Ⓑ (Average time per angular region = 1.1×10^{-3} sec)

a_0 = .886226470016632D + 00
a_1 = −.999950714561036D + 00
a_2 = .885348820003892D + 00
a_3 = −.660611239043357D + 00
a_4 = .421821197160099D + 00
a_5 = −.222898055667208D + 00
a_6 = .905057384150449D − 01
a_7 = −.254906111884287D − 01
a_8 = .430895168984138D − 02
a_9 = −.323377239693247D − 03

For Ⓒ (Average time per angular region = 1.3×10^{-3} sec)

a_0 = .886226924931465D + 00
a_1 = −.999999899776252D + 00
a_2 = .886223733186722D + 00
a_3 = −.666626670510907D + 00
a_4 = .442851899328568D + 00
a_5 = −.265638206366025D + 00
a_6 = .145060043403012D + 00
a_7 = −.7149098377998890 − 01
a_8 = .309199295521210D − 01
a_9 = −.112323532148441D − 01
a_{10} = .324944543171185D − 02
a_{11} = −.704260243309096D − 03
a_{12} = .105787574480633D − 03
a_{13} = −.971864864160461D − 05
a_{14} = .408335517232165D − 06

For Ⓓ (Average time per angular region = 1.5×10^{-3} sec)

a_0 = .886226925452593D + 00
a_1 = −.999999999948597D + 00
a_2 = .886226922786746D + 00
a_3 = −.666666611866661D + 00
a_4 = .443112868048919D + 00
a_5 = −.266662729091411D + 00
a_6 = .147687136321938D + 00
a_7 = −.761365855850292D − 01
a_8 = .368032849350860D − 01
a_9 = −.167195096888183D − 01
a_{10} = .710292625734052D − 02
a_{11} = −.278170932906224D − 02
a_{12} = .981112629090333D − 03
a_{13} = −.302588640752108D − 03
a_{14} = .789960968802448D − 04
a_{15} = −.168685181767046D − 04
a_{16} = .283646635409322D − 05
a_{17} = −.358314466908290D − 06
a_{18} = .317679497040006D − 07
a_{19} = −.175440651940430D − 08
a_{20} = .452534347337305D − 10

Average time per angular region a refers to the average computing time on the CDC-6700 to obtain $P(a)$.

APPENDIX F

PROGRAM LISTINGS IN FORTRAN IV

P-2, VALR-2, SORT III, VALR-7, P-7, SORT I, SORT II, SMP-7

(Flow charts on pages 40–45 and A-17 to A-19)

MASTER SUBROUTINE P-2

(FLOW CHART 1, page 40)

P-2 is used for computing P(Π) over an Arbitrary Polygon Π*

CALL: P-2 (x, y, N, P, ICV, IND, IOP, A, W),

where:

x is the array of abscissas of the numbered points of Π. x is dimensioned at N + 1.

y is the array of ordinates of the numbered points of Π. y is dimensioned at N + 1.

N is the number of points specifying Π, except if N = 1 when the IBND over an angular region is computed. Three input points are needed when N = 1, given in counterclockwise order, with the vertex at point one, (see pages 25, 27).

P is the location where the value of P(Π) is returned.

ICV must be set as an integer by the user according to the list below:

ICV = 0, Π is simple, or of $\bar{S}$ type with no SAR(s) (see pages 12, 31). VALR-7 used alone.

ICV > 0, Π is arbitrary. VALR-2 used alone.

ICV = −2, Π is of $\bar{S}$ type with possible SAR(s).

ICV < 0, ≠−2, Π is arbitrary with PAR(s).

IND is an error indicator. Normally, it is set to zero. If IND = 2, then PAR(s) have been detected by either VALR-2 or VALR-7. For VALR-2, (ICV > 0, ICV < 0, ≠ −2) the result for P(Π) is acceptable. For VALR-7 (ICV = 0, −2) however, this result of IND = 2, means the value for P(Π) is most likely wrong, unless N = 1 *VALR-7 is not to be used alone where SAR(s) are a possibility, unless N = 1.* If IND = 3, then N has not been specified as an integer equal to one or greater than two. Such values of N are not allowed.

IOP is an accuracy parameter. It is set by the user to 1, 2, 3 for approximately 3, 6, or 9 decimal digits of accuracy in P(Π).

A is the location where A(Π) is returned. |A| gives area of Π, (see pages 9, 26).

W is the location where the winding number of Π is returned. It is computed in VALR-2 and takes integer values (see pages 18, 19). W is defined as an integer variable. It is initialized to one, and is only computed if ICV > 0 or ICV < 0, ≠ −2.

*See footnote 1, page 1, for definition of an arbitrary polygon.

```
      SUBROUTINE P2  ( X,Y,NB,P,ICV,IND,IOP,A,KO )
      DIMENSION X(1),Y(1)
      IF ( NB.NE.2.AND.NB.GE.1 )  GO TO 3031
      IND=3
      RETURN
3031  CONTINUE
      N=NB
      KO=1
      IF ( ICV.EQ.0.OR.NB.EQ.1 )  GO TO 3091
      IF ( ICV.GT.0 )  GO TO 3071
      CALL SORT3 ( X,Y,N )
      IF ( N.GT.2 )  GO TO 3061
      P=0.
      A=0.
      IND=0
      RETURN
3061  CONTINUE
      IF ( ICV.EQ.-2 )  GO TO 3091
3071  CONTINUE
      CALL VALR2 ( X,Y,N,P,IOP,A,IND,KO )
      RETURN
3091  CONTINUE
      CALL VALR7 ( X,Y,N,P,IOP,A,IND )
      RETURN
      END
```

SUBROUTINE VALR-2

(FLOW CHART 2, page 41)

VALR-2 is used to compute P(Π) when Π is arbitrary

CALL: VALR-2 (x, y, N, P, IOP, A, IND, W),

where:

- x is the array of abscissas of the numbered points of Π. x is dimensioned at N + 1.
- y is the array of ordinates of the numbered points of Π. y is dimensioned at N + 1.
- N is the number of points specifying Π, except if N = 1 when the IBND over an angular region is computed. Three input points are needed, when N = 1, given in counterclockwise order, with the vertex at point one, (see pages 25, 27).
- P, A are the locations where the values of P(Π) and A(Π) are returned.
- IOP is an accuracy parameter. It is set by the user to 1, 2, or 3 for approximately 3, 6, or 9-decimal digits of accuracy in P(Π).
- IND is an error indicator. Normally, it is set to zero. If IND = 2, it informs the user that Π contains a PAR. The value for P(Π) is acceptable. If IND = 3, then N has not been specified as an integer equal to one or greater than two. Such values of N are not allowed.
- W is the location where the value of the winding number W for Π is returned. W is an integer variable.

This routine requires computation of erf (x) and erfc (x) which are defined on pages 5, 28 and 29. We have

$$\text{ERF}(1,x) = \text{erf}(x), \quad \text{ERFC}(1,0,x) = \text{erfc}(x),$$

where the subroutine listings for these functions are given on pages F-12 to F-15. They are identical to the NSWC (DL) math library functions ERF and ERFC as of June 1980 which are based on the reference below.

Cody, W. J., *Rational Chebyshev Approximations for the Error Function*, **Mathematics of Computation**, v. 23 (1969), pp. 631-637.

```
      SUBROUTINE VALR2 ( X,Y,N,P,IOP,A,IND,KO )
      DIMENSION X(1),Y(1),G(2),H(2),RSQ(4)
      DIMENSION E(5),E2(10),E3(15)
      DIMENSION APH1(3),APH2(3),CST(3)
      DIMENSION APH4(3),A3D8(3)
      REAL L
      REAL KOM
      DATA PI/3.1415 92653 5898 /
      DATA TWOPI/6.2831 85307 17958 /
      DATA ALNPI/1.1447 29885 84940 /
      DATA C1/.28209 47917 73877 /
      DATA C2/.15915 49430 91895 /
      DATA TAU/7.E-14 /
      DATA TAUSQ/4.9E-27 /
       DATA ( E(I),I=1,  5) /
     1        .8857775185728 95E+00 ,       -.9811519527780 50E+00 ,
     2        .7593055020824 85E+00 ,       -.3536449806869 77E+00 ,
     3        .6952320924352 07E-01 /
       DATA (E2(I),I=1, 10) /
     1        .8862264700166 32E+00 ,       -.9999507145610 36E+00 ,
     2        .8853488200038 92E+00 ,       -.6606112390433 57E+00 ,
     3        .4218211971600 99E+00 ,       -.2228980556672 08E+00 ,
     4        .9050573841504 49E-01 ,       -.2549061118842 87E-01 ,
     5        .4308951689841 38E-02 ,       -.3233772396932 47E-03 /
       DATA (E3(I),I=1, 15) /
     1        .8862269249314 65E+00 ,       -.9999998997762 52E+00 ,
     2        .8862237331867 22E+00 ,       -.6666266705109 07E+00 ,
     3        .4428518993285 69E+00 ,       -.2656382063660 25E+00 ,
     4        .1450600434030 14E+00 ,       -.7149098377998 89E-01 ,
     5        .3091992955212 10E-01 ,       -.1123235321484 41E-01 ,
     6        .3249445431711 85E-02 ,       -.7042602433090 96E-03 ,
     7        .1057875744806 33E-03 ,       -.9718648641604 61E-05 ,
     8        .4083355172321 65E-06 /
      DATA ( APH1(I),I=1,3 ) /
     1  2.02E-7,2.08E-13,2.71E-19 /
      DATA ( APH2(I),I=1,3 ) /
     1  1.22E-2,1.23E-4,1.34E-6 /
      DATA ( APH4(I),I=1,3 ) /
     1  .6962E-1,  .6990E-2,  .7311E-3 /
      DATA RTPII/.56418 95835 4776 /
      DATA (  RSQ(I),I=1,3 ) /
     1  6.0516,12.60605 ,19.201924 /
      DATA ( A3D8(I),I=1,3 ) /
     1  0.28125E-4,0.285E-7,0.32625E-10 /
      DATA ( CST(I),I=1,3 ) /
     1  .5625E-4,.57E-7,.6512E-10 /
      IF ( N.NE.2.AND.N.GE.1 )  GO TO 3011
      IND=3
```

```
      RETURN
3011  CONTINUE
      P=0.
      IND=0
      A=0.
      KOM=0.
      K=1
      IF ( N.NE.1 )  GO TO 3021
      W=X(2)-X(1)
      Z=Y(2)-Y(1)
      U    =X(3)-X(1)
      V    =Y(3)-Y(1)
      PSI1=V*W-U*Z
      IF ( PSI1.GE.0. )  GO TO 3041
      P=-1.
      T1=W
      W=U
      U=T1
      T1=V
      V=Z
      Z=T1
      GO TO 3041
3021  CONTINUE
      Y(N+1)=Y(1)
      X(N+1)=X(1)
      U    =X(2)-X(1)
      V    =Y(2)-Y(1)
      XK=X(1)
      YK=Y(1)
3031  CONTINUE
      W=X(1)-X(N)
      Z=Y(1)-Y(N)
3041  CONTINUE
      D1SQ=W*W+Z*Z
      IF ( D1SQ.GT.TAUSQ )  GO TO 3051
      IF ( N.EQ.1 )  GO TO 4011
      N=N-1
      IF ( N.EQ.2 )  RETURN
      GO TO 3031
3051  CONTINUE
      D2SQ=U*U+V*V
      IF ( D2SQ.GT.TAUSQ )  GO TO 3071
      IF ( N.EQ.1 )  GO TO 4011
3061  CONTINUE
      K=K+1
      U=X(K+1)-XK
      V=Y(K+1)-YK
      D2SQ=U*U+V*V
```

```
      IF ( D2SQ.LE.TAUSQ )  GO TO 3061
      IF ( K.EQ.(N-1) )  RETURN
3071  CONTINUE
      A=XK*(Y(K+1)-Y(N))
      BGD1=SQRT(2.*D1SQ)
      BGD2=SQRT(2.*D2SQ)
3081  CONTINUE
      PSI1=V*W-U*Z
      CEE=U*W+V*Z
      AJ0 =ATAN2(PSI1,CEE)
      KOM=KOM+AJ0
      L=0.
      B=.5*(X(K)*X(K)+Y(K)*Y(K))
      IF ( B.GT.APH1(IOP) )  GO TO  3111
      CAPG=0.
3101  CONTINUE
      P1  =AJ0 /TWOPI-CAPG
      GO TO 3621
3111  CONTINUE
      G(1)=(W*X(K)+Z*Y(K))/BGD1
      G(2)=(U*X(K)+V*Y(K))/BGD2
      H(1)=(-Y(K)*W+X(K)*Z)/BGD1
      H(2)=(-Y(K)*U+X(K)*V)/BGD2
      IF ( ABS(PSI1).GT.(BGD1*BGD2*A3D8(IOP)))  GO TO 3241
      IF ( CEE.LT.0. )  GO TO 3131
      IF ( ABS(AJ0).LE.TAU )  GO TO 3121
      IF ( G(1).GE.0. )  GO TO 3121
      GO TO 3241
3121  CONTINUE
      P1=0.
      GO TO 3621
3131  CONTINUE
      IF ( ABS(PSI1).LE.(.5*TAU*BGD1*BGD2) )  IND=2
      IF ( PSI1.LT.0. )  GO TO 3171
      P1  =.5*ERFC1(0,H(2))
      GO TO 3621
3171  CONTINUE
      P1  =-.5*ERFC1(0,H(1))
      GO TO 3621
3241  CONTINUE
      IF ( B.LE.APH2(IOP) )  GO TO 3301
      IF ( G(1).LT.0. )  GO TO 3261
      IF ( G(2).GE.0. )  GO TO 3471
      G(2)=-G(2)
      H(2)=-H(2)
      IF ( ABS(H(2)).LE.APH4(IOP) )  GO TO 3251
      L=.5*ERFC1(0,-H(2))
      GO TO 3461
```

```
3251  CONTINUE
      L=.5+RTPII*H(2)
      GO TO 3461
3255  CONTINUE
      L=.5-RTPII*H(1)
      GO TO 3461
3261  CONTINUE
      G(1)=-G(1)
      H(1)=-H(1)
      IF ( G(2).LT.0. )  GO TO 3271
      IF ( ABS(H(1)).LE.APH4(IOP) )  GO TO 3255
      L=.5*ERFC1(0,H(1))
      GO TO 3461
3271  CONTINUE
      G(2)=-G(2)
      H(2)=-H(2)
      IF ( ABS(H(1)).LE.APH4(IOP) )  GO TO 3291
      IF ( ABS(H(2)).LE.APH4(IOP) )  GO TO 3281
      L=.5*(ERFC1(0,H(1))-ERFC1(0,H(2)))
      GO TO 3471
3281  CONTINUE
      L=RTPII*H(2)-.5*ERF1(H(1))
      GO TO 3471
3291  CONTINUE
      IF ( ABS(H(2)).LE.APH4(IOP) )  GO TO 3295
      L=.5*ERF1(H(2))-RTPII*H(1)
      GO TO 3471
3295  CONTINUE
      L=RTPII*(H(2)-H(1))
      GO TO 3471
3301  CONTINUE
      CAPG=C1*(H(2)-H(1))-C2*(G(2)*H(2)-G(1)*H(1))
      GO TO 3101
3461  CONTINUE
      PSI1=-PSI1
      IF ( PSI1.LE.0. )  GO TO 3465
      L=L-1.
      AJ0=PI+AJ0
      GO TO 3471
3465  CONTINUE
      AJ0=AJ0 -PI
3471  CONTINUE
      IF ( B.GE.RSQ(IOP) )  GO TO 3501
      CAPE=AJ0
      CAPH=.5*AJ0
      M=1
      F=0.
      AJ1=H(2)-H(1)
```

```
      CIRCM=AJ1
      IF ( IOP.EQ.3 )  GO TO 3681
      IF ( IOP.EQ.2 )  GO TO 3701
      SUM=E(M)*AJ1
3481  CONTINUE
      M=M+1
      H(2)=H(2)*G(2)
      H(1)=H(1)*G(1)
      T=H(2)-H(1)
      F=F+B
      CAPV=(F*CAPE+T)/M
      SUM=SUM+E(M)*CAPV
      IF( M .GE. 5    )   GO TO 3491
      CAPE=CIRCM
      CIRCM=CAPV
      GO TO 3481
3491  CONTINUE
      P1  =L+EXP(-(B+ALNPI))*(CAPH-SUM)
      GO TO 3621
3501  CONTINUE
      P1=L
3621  CONTINUE
      IF ( K.NE.N )  GO TO 3651
      IF ( N.NE.1 )  GO TO 3631
      P=ABS(P+ABS(P1))
      RETURN
3631  CONTINUE
      P=P-P1
      KOM=KOM/TWOPI
      A=.5*A
      IF ( KOM.LT.0. )  GO TO 3641
      KO=INT(KOM+.125 )
      GO TO 3645
3641  CONTINUE
      KO=INT(KOM-.125 )
3645  CONTINUE
      P=P+FLOAT(KO)
      RETURN
3651  CONTINUE
      W=U
      Z=V
      BGD1=BGD2
      XK=X(K+1)
      YK=Y(K+1)
      YKM1=Y(K)
3661  CONTINUE
      K=K+1
      U=X(K+1)-XK
```

```
      V=Y(K+1)-YK
      D2SQ=U*U+V*V
      IF ( D2SQ.LE.TAUSQ )  GO TO 3661
      BGD2=SQRT(2.*D2SQ)
      P=P-P1
      A=A+XK*(Y(K+1)-YKM1 )
      GO TO 3081
3681  CONTINUE
      SUM=E3(M)*AJ1
3691  CONTINUE
      M=M+1
      H(2)=H(2)*G(2)
      H(1)=H(1)*G(1)
      T=H(2)-H(1)
      F=F+B
      CAPV=(F*CAPE+T)/M
      SUM=SUM+E3(M)*CAPV
      IF ( M.GE.15 )  GO TO 3491
      CAPE=CIRCM
      CIRCM=CAPV
      GO TO 3691
3701  CONTINUE
      SUM=E2(M)*AJ1
3711  CONTINUE
      M=M+1
      H(2)=H(2)*G(2)
      H(1)=H(1)*G(1)
      T=H(2)-H(1)
      F=F+B
      CAPV=(F*CAPE+T)/M
      SUM=SUM+E2(M)*CAPV
      IF ( M.GE.10 )  GO TO 3491
      CAPE=CIRCM
      CIRCM=CAPV
      GO TO 3711
4011  CONTINUE
      P=5.
      IND=1
      RETURN
      END
```

```
      FUNCTION ERF1(X)
      DIMENSION A(4),B(4),P(8),Q(8),R(5),S(5)
      DATA A/2.42667955230532E02, 2.19792616182942E01,
     1       6.99638348861914E00,-3.56098437018154E-2/
      DATA B/2.15058875869861E02, 9.11649054045149E01,
     1       1.50827976304078E01, 1.00000000000000E00/
      DATA P/3.00459261020162E02, 4.51918953711873E02,
     1       3.39320816734344E02, 1.52989285046940E02,
     2       4.31622272220567E01, 7.21175825088309E00,
     3       5.64195517478974E-1,-1.36864857382717E-7/
      DATA Q/3.00459260956983E02, 7.90950925327898E02,
     1       9.31354094850610E02, 6.38980264465631E02,
     2       2.77585444743988E02, 7.70001529352295E01,
     3       1.27827273196294E01, 1.00000000000000E00/
      DATA R/2.99610707703542E-3, 4.94730910623251E-2,
     1       2.26956593539687E-1, 2.78661308609648E-1,
     2       2.23192459734185E-2/
      DATA S/1.06209230528468E-2, 1.91308926107830E-1,
     1       1.05167510706793E00, 1.98733201817135E00,
     2       1.00000000000000E00/
      DATA C/5.64189583547756E-1/
C     ----------------------------
      AX=ABS(X)
      X2=AX*AX
      IF (AX.GE.0.5) GO TO 20
      TOP=A(4)
      BOT=B(4)
      DO 10 I=1,3
      J=4-I
      TOP=A(J)+X2*TOP
   10 BOT=B(J)+X2*BOT
      ERF1=X*TOP/BOT
      RETURN
C
   20 IF (AX.GT.4.0) GO TO 30
      TOP=P(8)
      BOT=Q(8)
      DO 21 I=1,7
      J=8-I
      TOP=P(J)+AX*TOP
   21 BOT=Q(J)+AX*BOT
      ERF1=1.-EXP(-X2)*TOP/BOT
      IF (X.LT.0.)  ERF1=-ERF1
      RETURN
C
   30  ERF1=1.
      IF (AX.GE.5.54) GO TO 32
      TOP=R(1)
```

```
      BOT=S(1)
      DO 31 I=2,5
      TOP=R(I)+X2*TOP
   31 BOT=S(I)+X2*BOT
       ERF1=C-TOP/(X2*BOT)
       ERF1=1.-EXP(-X2)*ERF1/AX
   32 IF (X.LT.0.)  ERF1=-ERF1
      RETURN
      END
```

```
      FUNCTION  ERFC1(IND,X)
      DIMENSION A(4),B(4),P(8),Q(8),R(5),S(5)
      DATA A/2.42667955230532E02, 2.19792616182942E01,
     1       6.99638348861914E00,-3.56098437018154E-2/
      DATA B/2.15058875869861E02, 9.11649054045149E01,
     1       1.50827976304078E01, 1.00000000000000E00/
      DATA P/3.00459261020162E02, 4.51918953711873E02,
     1       3.39320816734344E02, 1.52989285046940E02,
     2       4.31622272220567E01, 7.21175825088309E00,
     3       5.64195517478974E-1,-1.36864857382717E-7/
      DATA Q/3.00459260956983E02, 7.90950925327898E02,
     1       9.31354094850610E02, 6.38980264465631E02,
     2       2.77585444743988E02, 7.70001529352295E01,
     3       1.27827273196294E01, 1.00000000000000E00/
      DATA R/2.99610707703542E-3, 4.94730910623251E-2,
     1       2.26956593539687E-1, 2.78661308609648E-1,
     2       2.23192459734185E-2/
      DATA S/1.06209230528468E-2, 1.91308926107830E-1,
     1       1.05167510706793E00, 1.98733201817135E00,
     2       1.00000000000000E00/
      DATA C/5.64189583547756E-1/
C     ------------------------------
      AX=ABS(X)
      X2=AX*AX
      IF (AX.GE.0.47) GO TO 20
      TOP=A(4)
      BOT=B(4)
      DO 10 I=1,3
      J=4-I
      TOP=A(J)+X2*TOP
   10 BOT=B(J)+X2*BOT
       ERFC1=1.-X*TOP/BOT
      IF (IND.NE.0)  ERFC1=EXP(X2)*ERFC1
      RETURN
C
   20 IF (AX.GT.4.0) GO TO 30
      TOP=P(8)
      BOT=Q(8)
      DO 21 I=1,7
      J=8-I
      TOP=P(J)+AX*TOP
   21 BOT=Q(J)+AX*BOT
       ERFC1=TOP/BOT
      IF (IND.EQ.0) GO TO 22
      IF (X.LT.0.0)  ERFC1=2.*EXP(X2)-ERFC1
      RETURN
   22  ERFC1=EXP(-X2)*ERFC1
      IF (X.LT.0.0)  ERFC1=2.-ERFC1
```

```
      RETURN
C
   30 IF (X.LE.-5.33) GO TO 32
      TOP=R(1)
      BOT=S(1)
      DO 31 I=2,5
      TOP=R(I)+X2*TOP
   31 BOT=S(I)+X2*BOT
       ERFC1=(C-TOP/(X2*BOT))/AX
      IF (IND.EQ.0) GO TO 22
      IF (X.LT.0.0)  ERFC1=2.*EXP(X2)-ERFC1
      RETURN
   32  ERFC1=2.
      IF (IND.NE.0)  ERFC1=EXP(X2)*ERFC1
      RETURN
      END
```

SUBROUTINE SORT III

(FLOW CHART 3, page 43)

Subroutine SORT III Used to Eliminate SDP and/or SCP from II

CALL: SORT III (x, y, N),

where:

x is the array of abscissas of the numbered points of the polygon Π. The array is dimensioned at N. Upon return to the calling program, P-2 (or P-7), the array of abscissas will be reduced by the number of consecutive duplicate points SDP and SCP eliminated. The array is compacted.*

y is the array of ordinates of the numbered points of the polygon Π. The array is dimensioned at N. Upon return to the calling program, P-2 (or P-7), the array of ordinates will be reduced by the number of points deleted due to SDP and SCP. The array is CU.

N is the number of points initially used to specify the polygon. Upon return to the calling program, P-2 (or P-7), N will be reduced by the number of points that were eliminated.

*Compact here means that whenever a point is eliminated all subsequent points of the array are moved up one location in the array, i.e., the array is closed up (CU).

```
      SUBROUTINE SORT3 ( X,Y,N )
      DIMENSION X(1),Y(1)
      DATA CST/4.9E-27 /
3041  CONTINUE
      IF ( N.LT.3 )  RETURN
      K=1
      L=2
3051  CONTINUE
      U=X(1)-X(N)
      V=Y(1)-Y(N)
      D2=U*U+V*V
      IF ( D2.GT.CST )  GO TO 3061
      N=N-1
      IF ( N.GT.2 )  GO TO 3051
      RETURN
3061  CONTINUE
      W=X(L)-X(1)
      Z=Y(L)-Y(1)
      D1=W*W+Z*Z
      IF ( D1.GT.CST )  GO TO 3071
      L=L+1
      GO TO 3061
3071  CONTINUE
      IF ( L.EQ.(K+1) )  GO TO 3091
      LM2=L-2
      N=N-LM2
      DO 3081  I=2,N
      I1=LM2+I
      X(I)=X(I1)
      Y(I)=Y(I1)
3081  CONTINUE
      L=2
3091  CONTINUE
      T=V*W-U*Z
      SN=(4.*T*T)/(D1*D2)
      IF ( SN.GT.CST )  GO TO 3121
3111  CONTINUE
      L=L+1
      IF ( L.GT.N )  GO TO 3341
3115  CONTINUE
      W=X(L)-X(1)
      Z=Y(L)-Y(1)
      D1=W*W+Z*Z
      IF ( D1.GT.CST )  GO TO 3091
      GO TO 3111
3121  CONTINUE
      IF ( L.EQ.2 )  GO TO 3141
      LM2=L-2
```

```
      N=N-LM2
      DO 3131  I=1,N
      I1=LM2+I
      X(I)=X(I1)
      Y(I)=Y(I1)
3131  CONTINUE
      GO TO 3041
3141  CONTINUE
      K=2
      L=3
      GO TO 3161
3151  CONTINUE
      D1=D2
      W=U
      Z=V
3155  CONTINUE
      L=K+1
3161  CONTINUE
      U=X(L)-X(K)
      V=Y(L)-Y(K)
      D2=U*U+V*V
      IF ( D2.GT.CST )  GO TO 3171
3165  CONTINUE
      L=L+1
      IF ( L.LE.N )  GO TO 3161
      N=K
      GO TO 3251
3171  CONTINUE
      IF ( L.EQ.(K+1) )  GO TO 3191
      N=N-((L-1)-K)
      KP1=K+1
      I2=L-KP1
      DO 3181  I=KP1,N
      I1=I2+I
      X(I)=X(I1)
      Y(I)=Y(I1)
3181  CONTINUE
      L=KP1
3191  CONTINUE
      T=V*W-U*Z
      SN=(4.*T*T)/(D1*D2)
      IF ( SN.GT.CST )  GO TO 3221
3201  CONTINUE
      L=L+1
      IF ( L.GT.N )  GO TO 3211
      U=X(L)-X(K)
      V=Y(L)-Y(K)
      D2=U*U+V*V
```

```
      IF ( D2.GT.CST )  GO TO 3191
      GO TO 3201
 3211 CONTINUE
      X(K)=X(N)
      Y(K)=Y(N)
      N=K
      GO TO 3251
 3221 CONTINUE
      IF ( L.EQ.(K+1) )  GO TO 3241
      I2=L-1-K
      N=N-I2
      LM2=L-2
      DO 3231  I=K,N
      I1=I2+I
      X(I)=X(I1)
      Y(I)=Y(I1)
 3231 CONTINUE
      W=X(K)-X(K-1)
      Z=Y(K)-Y(K-1)
      D1=W*W+Z*Z
      IF ( D1.GT.CST )  GO TO 3155
      K=K-1
      IF ( K.LT.2 )  GO TO 3041
      W=X(K)-X(K-1)
      Z=Y(K)-Y(K-1)
      D1=W*W+Z*Z
      L=K+1
      GO TO 3165
 3241 CONTINUE
      K=K+1
      IF ( K.LT.N )  GO TO 3151
      GO TO 3255
 3251 CONTINUE
      U=X(N)-X(N-1)
      V=Y(N)-Y(N-1)
      D2=U*U+V*V
      IF ( D2.LE.CST )  GO TO 3261
 3255 CONTINUE
      W=X(1)-X(N)
      Z=Y(1)-Y(N)
      D1=W*W+Z*Z
      IF ( D1.LE.CST )  GO TO 3261
      T=V*W-U*Z
      SN=(4.*T*T)/(D1*D2)
      IF ( SN.GT.CST )  GO TO 3351
 3261 CONTINUE
      N=N-1
      IF ( N.GT.2 )  GO TO 3251
```

```
      RETURN
3341  CONTINUE
      N=2
      RETURN
3351  CONTINUE
      D2=D1
      U=W
      V=Z
      W=X(2)-X(1)
      Z=Y(2)-Y(1)
      D1=W*W+Z*Z
      T=V*W-U*Z
      SN=(4.*T*T)/(D1*D2)
      IF ( SN.GT.CST )  RETURN
      L=3
      GO TO 3115
      END
```

SUBROUTINE VALR-7

(FLOW CHART 4, page 44)

Subroutine VALR-7 Used to Compute p($\bar{S}$), where $\bar{S}$ has no SAR(s), (See page 31)

CALL: VALR-7 (x, y, M, p, IOP, a, IND),*

where:

x is the input array of abscissas for $\bar{S}$. Dimensioned at M + 1.

y is the input array of ordinates for $\bar{S}$. Dimensioned at M + 1.

M is the number of input points for $\bar{S}$. When M = 1, IBND over an angular region is computed. Three input points in counterclockwise order are used to specify the region with the vertex at (1).

p is the location where the function value for p($\bar{S}$) will be returned.†

IOP is set by the user to 1, 2, or 3 for approximately 3, 6, or 9-decimal-digit accuracy, respectively, in p($\bar{S}$).

a is the location where the value of the function a($\bar{S}$) is returned. The absolute value of a gives the area of $\bar{S}$.

IND is an error indicator normally set to zero. If PAR(s) are detected by VALR-7, then IND is set to two and the result for p($\bar{S}$) is most likely wrong, unless M = 1. See Flow Chart 4-24, 20, 21, 22. VALR-7 should never be used alone if SAR(s) are a possibility, unless M = 1. If M = 2 or M < 1, then IND = 3 and an EXIT is made. Such M are not allowed.

This routine requires computation of erf (x) and erfc (x) which are defined on pages 5, 28 and 29. We have

$$\text{ERF 1 (x)} = \text{erf (x)}, \quad \text{ERFC 1 (0, x)} = \text{erfc (x)},$$

where the subroutine listings for these functions are given on pages F-12 to F-15. They are identical to the NSWC (DL) math library functions ERF and ERFC as of June 1980 which are based on the reference below.

Cody, W. J., *Rational Chebyshev Approximations for the Error Function*, Mathematics of Computation, v. 23 (1969), pp. 631-637.

*We use p, a, M here in place of P, A, N to avoid ambiguity with results in P-7, if SORT I is used with VALR-7.
†The IBND over $\bar{S}$, p($\bar{S}$), will be positive if $\bar{S}$ is PO and it will be negative if $\bar{S}$ is NO.

```
      SUBROUTINE VALR7 ( X,Y,N,P,IOP,A,IND )
      DIMENSION RSQ(4)
      DIMENSION X(1),Y(1),G(2),H(2)
      DIMENSION E(5),E2(10),E3(15)
      DIMENSION APH1(3),APH2(3),APH4(3),CST(3)
      REAL L
      DATA TWOPI/6.2831 85307 17958 /
      DATA ALNPI/1.1447 29885 84940 /
      DATA C1/.28209 47917 73877 /
      DATA C2/.15915 49430 91895 /
      DATA TAU/7.E-14 /
       DATA ( E(I),I=1,  5) /
     1        .8857775185728 95E+00 ,       -.9811519527780 50E+00 ,
     2        .7593055020824 85E+00 ,       -.3536449806869 77E+00 ,
     3        .6952320924352 07E-01 /
       DATA (E2(I),I=1, 10) /
     1        .8862264700166 32E+00 ,       -.9999507145610 36E+00 ,
     2        .8853488200038 92E+00 ,       -.6606112390433 57E+00 ,
     3        .4218211971600 99E+00 ,       -.2228980556672 08E+00 ,
     4        .9050573841504 49E-01 ,       -.2549061118842 87E-01 ,
     5        .4308951689841 38E-02 ,       -.3233772390932 47E-03 /
       DATA (E3(I),I=1, 15) /
     1        .8862269249314 65E+00 ,       -.9999998997762 52E+00 ,
     2        .8862237331867 22E+00 ,       -.6666266705109 07E+00 ,
     3        .4428518993285 69E+00 ,       -.2656382063660 25E+00 ,
     4        .1450600434030 14E+00 ,       -.7149098377998 89E-01 ,
     5        .3091992955212 10E-01 ,       -.1123235321484 41E-01 ,
     6        .3249445431711 85E-02 ,       -.7042602433090 96E-03 ,
     7        .1057875744806 33E-03 ,       -.9718648641604 61E-05 ,
     8        .4083355172321 65E-06 /
      DATA ( APH1(I),I=1,3 ) /
     1  2.02E-7,2.08E-13,2.71E-19 /
      DATA ( APH2(I),I=1,3 ) /
     1  1.22E-2,1.23E-4,1.34E-6 /
      DATA ( APH4(I),I=1,3 ) /
     1  .6962E-1,  .6990E-2,  .7311E-3 /
      DATA RTPII/.56418 95835 4776 /
      DATA (  RSQ (I),I=1,3 ) /
     1  6.0516,12.60605 ,19.201924 /
      DATA ( CST(I),I=1,3 ) /
     1  .5625E-4,.57E-7,.6512E-10 /
      IF ( N.NE.2.AND.N.GE.1 )  GO TO 3061
      IND=3
      RETURN
3061  CONTINUE
      P=0.
      IND=0
      IF ( N.NE.1 )  GO TO 3071
```

```
      K=1
      A=0.
      W=X(2)-X(1)
      Z=Y(2)-Y(1)
      U   =X(3)-X(1)
      V   =Y(3)-Y(1)
      PSI1=V*W-U*Z
      IF ( PSI1.GE.0. )  GO TO 3081
      P=-1.
      T1=W
      W=U
      U=T1
      T1=V
      V=Z
      Z=T1
      GO TO 3081
3071  CONTINUE
      CALL SMP7  ( N,A,X,Y )
      IF ( ABS(A   ).LE.CST(IOP) )  RETURN
      K=1
      W=X(1)-X(N)
      Z=Y(1)-Y(N)
      U   =X(2)-X(1)
      V   =Y(2)-Y(1)
      X(N+1)=X(1)
      Y(N+1)=Y(1)
3081  CONTINUE
      BGD1=SQRT( 2.*(W*W+Z*Z))
      BGD2=SQRT( 2.*(U*U+V*V))
3091  CONTINUE
      L=0.
      B=.5*(X(K)*X(K)+Y(K)*Y(K))
      IF ( B.GT.APH1(IOP) )  GO TO  3111
      CAPG=0.
3101  CONTINUE
      T1=V*W-U*Z
      T2=U*W+V*Z
      PHIK=ATAN2(T1,T2)
      P1  =PHIK/TWOPI-CAPG
      GO TO 3621
3111  CONTINUE
      G(1)=(W*X(K)+Z*Y(K))/BGD1
      G(2)=(U*X(K)+V*Y(K))/BGD2
      H(1)=(-Y(K)*W+X(K)*Z)/BGD1
      H(2)=(-Y(K)*U+X(K)*V)/BGD2
     SN=(2.*(V*W-U*Z))/(BGD1*BGD2)
      IF ( ABS(SN).GT.CST(IOP) )  GO TO 3241
      CN=G(1)*G(2)+H(1)*H(2)
```

```
      IF ( CN.LT.0. )  GO TO 3131
      IF ( ABS(SN).LE.TAU )  GO TO 3121
      IF ( G(1).GE.0. )  GO TO 3121
      GO TO 3241
3121  CONTINUE
      P1=0.
      GO TO 3621
3131  CONTINUE
      IF ( ABS(SN).LE.TAU )  IND=2
      IF (  SN .LT.0. )  GO TO 3171
      P1  =.5*ERFC1(0,H(2))
      GO TO 3621
3171  CONTINUE
      P1  =-.5*ERFC1(0,H(1))
      GO TO 3621
3241  CONTINUE
      IF ( B.LE.APH2(IOP) )  GO TO 3301
      SN=B*SN
      IF ( G(1).LT.0. )  GO TO 3261
      IF ( G(2).GE.0. )  GO TO 3471
      G(2)=-G(2)
      H(2)=-H(2)
      IF ( ABS(H(2)).LE.APH4(IOP) )  GO TO 3251
      L=.5*ERFC1(0,-H(2))
      GO TO 3461
3251  CONTINUE
      L=.5+RTPII*H(2)
      GO TO 3461
3255  CONTINUE
      L=.5-RTPII*H(1)
      GO TO 3461
3261  CONTINUE
      G(1)=-G(1)
      H(1)=-H(1)
      IF ( G(2).LT.0. )  GO TO 3271
      IF ( ABS(H(1)).LE.APH4(IOP) )  GO TO 3255
      L=.5*ERFC1(0,H(1))
      GO TO 3461
3271  CONTINUE
      G(2)=-G(2)
      H(2)=-H(2)
      IF ( ABS(H(1)).LE.APH4(IOP) )  GO TO 3291
      IF ( ABS(H(2)).LE.APH4(IOP) )  GO TO 3281
      L=.5*(ERFC1(0,H(1))-ERFC1(0,H(2)))
      GO TO 3471
3281  CONTINUE
      L=RTPII*H(2)-.5*ERF1(H(1))
      GO TO 3471
```

```
3291  CONTINUE
      IF ( ABS(H(2)).LE.APH4(IOP) )  GO TO 3295
      L=.5*ERF1(H(2))-RTPII*H(1)
      GO TO 3471
3295  CONTINUE
      L=RTPII*(H(2)-H(1))
      GO TO 3471
3301  CONTINUE
      CAPG=C1*(H(2)-H(1))-C2*(G(2)*H(2)-G(1)*H(1))
      GO TO 3101
3461  CONTINUE
      SN=-SN
      IF ( SN.LE.0. )  GO TO 3471
      L=L-1.
3471  CONTINUE
      IF ( B.GE.RSQ(IOP) )  GO TO 3501
      CN=G(1)*G(2)+H(1)*H(2)
      AJ0=ATAN2(SN,CN)
      CAPE=AJ0
      CAPH=.5*AJ0
      M=1
      F=0.
      AJ1=H(2)-H(1)
      CIRCM=AJ1
      IF ( IOP.EQ.3 )  GO TO 3681
      IF ( IOP.EQ.2 )  GO TO 3701
      SUM=E(M)*AJ1
3481  CONTINUE
      M=M+1
      H(2)=H(2)*G(2)
      H(1)=H(1)*G(1)
      T=H(2)-H(1)
      F=F+B
      CAPV=(F*CAPE+T)/M
      SUM=SUM+E(M)*CAPV
      IF( M .GE. 5    )   GO TO 3491
      CAPE=CIRCM
      CIRCM=CAPV
      GO TO 3481
3491  CONTINUE
      P1  =L+EXP(-(B+ALNPI))*(CAPH-SUM)
      GO TO 3621
3501  CONTINUE
      P1=L
3621  CONTINUE
      IF ( K.NE.N )  GO TO 3651
      IF ( N.NE.1 )  GO TO 3631
      P=ABS(P+ABS(P1))
```

```
      RETURN
3631  CONTINUE
      P=P-P1
      IF (     A.LT.0. )  GO TO 3641
      P=P+1.
      RETURN
3641  CONTINUE
      P=P-1.
      RETURN
3651  CONTINUE
      K=K+1
      W=U
      Z=V
      U=X(K+1)-X(K)
      V=Y(K+1)-Y(K)
      BGD1=BGD2
      BGD2=SQRT( 2.*(U*U+V*V))
      P=P-P1
      GO TO 3091
3681  CONTINUE
      SUM=E3(M)*AJ1
3691  CONTINUE
      M=M+1
      H(2)=H(2)*G(2)
      H(1)=H(1)*G(1)
      T=H(2)-H(1)
      F=F+B
      CAPV=(F*CAPE+T)/M
      SUM=SUM+E3(M)*CAPV
      IF ( M.GE.15 )  GO TO 3491
      CAPE=CIRCM
      CIRCM=CAPV
      GO TO 3691
3701  CONTINUE
      SUM=E2(M)*AJ1
3711  CONTINUE
      M=M+1
      H(2)=H(2)*G(2)
      H(1)=H(1)*G(1)
      T=H(2)-H(1)
      F=F+B
      CAPV=(F*CAPE+T)/M
      SUM=SUM+E2(M)*CAPV
      IF ( M.GE.10 )  GO TO 3491
      CAPE=CIRCM
      CIRCM=CAPV
      GO TO 3711
      END
```

MASTER SUBROUTINE P-7

(FLOW CHART 5, page A-17)

SUBROUTINE P-7 is Used for Computing P(Π) for an Arbitrary Polygon Π*

CALL: P-7 (x, y, N, P, ICV, IND, IOP, A),

where:

x is the array of abscissas of the numbered points of Π. x is dimensioned at N + 1.

y is the array of ordinates of the numbered points of Π. y is dimensioned at N + 1.

N is the number of points specifying Π, except if N = 1 when the IBND over an angular region is computed. Three input points are needed, for N = 1, given in counterclockwise order, with the vertex at point one.

P is the location where the value of P(Π) is returned.

ICV must be specified as an integer by the user according to the list below:

ICV = 0 Π is simple or of $\bar{S}$ type with no SAR(s) (pages 12, 31). VALR-7 used alone.

ICV = 1 Π is in $\{\bar{S}\}$. SORT III used with VALR-7.

ICV = 2 Π is in {Π}. SORT I is used to search for duplicate points** of Π in increasing digital order from point (2) to point (N). Π is numbered with the α-option (see page 14), so Π is decomposed into simple polygons, S^1, S^2, ..., S^J. SORT II is not needed. VALR-7 is used to find $p(S^j)$, which are summed in SORT I to give P(Π).

ICV = −2 Π is in {Π}. SORT I is used to search for duplicate points of Π in decreasing digital order from point (N − 1) to point (1). Π is numbered with the α-option.

ICV ⩾ 3 Π is in {Π}. SORT I is used to search for duplicate points in increasing digit order of the numbered points from point (2) to point (N). Π is numbered with the β-option (see page 24), so Π is decomposed into $\bar{S}$ type elements $\bar{S}^1$, ..., $\bar{S}^L$. These elements require SORT II to eliminate any SCP, so that VALR-7 can be used on each $\bar{S}^l$ to obtain $p(\bar{S}^l)$, which are summed in SORT I to give P(Π).

ICV < 0 ≠ −2 This has the same function as ICV = 3, except that SORT I searches for duplicate points of Π in decreasing digital order of the numbered points starting at point (N − 1) and finishing at point (1).

---

*See footnote 1, page 1 for definition of an arbitrary polygon.

**Duplicate points are not to be confused with SDP(s), see pages 43 and A-8.

Generally ICV = 3 is preferable to ICV = 2 and ICV = −3 is preferable to ICV = −2, because the computing time may be less since often fewer angular regions of Π will need processing.

IND is an error indicator. It is normally set at zero. However, if VALR-7 is used alone (ICV = 0) on a polygon containing PAR(s), then IND is set to two and, unless N = 1, the result for P is probably wrong. This will never occur if SORT III or SORT I and SORT II are used to eliminate SAR(s) before using VALR-7, provided Π is in $\{\tilde{S}\}$. If N is not set to one or greater than two, as an integer, then IND is set to three with direct exit from VALR-7. Such N are not allowed.

IOP is an accuracy parameter. It is set by the user to 1, 2, or 3 for approximately 3, 6, or 9-decimal digits of accuracy in P(Π).

A is the location where A(Π) is returned. |A| gives the area of Π. (See Appendix D, also (46).)

```
      SUBROUTINE P7   ( X,Y,NB,P,ICV,IND,IOP,A )
      DIMENSION X(1),Y(1)
      IF ( NB.NE.2.AND.NB.GE.1 )  GO TO 3031
      IND=3
      RETURN
3031  CONTINUE
      N=NB
      IF ( N  .EQ.1 )   GO TO 3041
      IF ( ICV .EQ.0 )  GO TO 3041
      IF ( ICV.EQ.1 )  GO TO 3061
      CALL  SORT1 ( X,Y,N,P,ICV,IND,IOP,A )
      RETURN
3041  CONTINUE
      CALL VALR7   ( X,Y,N,P,IOP,A,IND )
      RETURN
3061  CONTINUE
      CALL SORT3 ( X,Y,N )
      IF ( N  .GT.2 )  GO TO 3071
      A=0.
      IND=0
      P=0.
      RETURN
3071  CONTINUE
      CALL VALR7   ( X,Y,N,P,IOP,A,IND )
      RETURN
      END
```

SUBROUTINE SORT I (See Appendix A)

(FLOW CHART 6, page A-18)

Subroutine SORT I Used to Decompose Π Into S or $\bar{S}$ Type Elements

CALL: SORT I (x, y, N, P, ICV, IND, IOP, A),

where:

x is the array of abscissas of the numbered points specifying the polygon, Π. x is dimensioned at N + 1.

y is the array of ordinates of the numbered points specifying the polygon, Π. y is dimensioned at N + 1.

N is the number of points numbered on the polygon.

P is the location where P(Π) is returned.

ICV is a user specified integer according to the listing below:

ICV = 2 for a polygonal element of the class {Π} (see page 1). SORT I searches for duplicate points of Π in increasing digital order from point (2) to point (N). Π is specified by numbering points of Π according to the α-option. (See page 14.) Π is decomposed into $S^1, \ldots, S^J$. VALR-7 is called to compute $p(S^i)$. These quantities are summed to give P(Π).

ICV = −2 for a polygonal element of {Π}. SORT I searches for duplicate points of Π in decreasing digital order from point (N − 1) to point (1). Π is numbered according to the α-option.

ICV ⩾ 3 for a polygonal element of {Π}. SORT I proceeds in the same way as for ICV = 2, except that Π is numbered according to the β-option rather than the α-option (see page 24). The β-option numbering requires that SORT II be used to eliminate SCP in any of the $\bar{S}$ elements obtained from the decomposition of Π by SORT I (see Flow Chart 6, page A-18). Π is decomposed after using SORT II into $S^1, \ldots, S^J$. VALR-7 is called to compute each $p(S^i)$. The $p(S^i)$ are summed to give P(Π).

ICV ⩽ 0 ≠ −2 for a polygonal element of {Π}. SORT I proceeds in the same way as for ICV = −2 except that Π is numbered according to the β-option rather than the α-option. The β-option numbering requires that SORT II be used to eliminate SCP in any of the $\bar{S}$ elements obtained from the decomposition of Π by SORT I.

If N does not differ using the α or β-option, then ICV = ±2 is preferable to ICV ≠ ±2. However, if N is reduced by using the β-option, then ICV ≠ ±2 is preferable since fewer calls to VALR-7 will be needed.

IND is an error indicator. Normally it is set to zero. If a π-angular region, PAR, is detected by VALR-7, IND is set to two, and $p(\bar{S})$ is very likely wrong, unless N = 1; consequently also $P(\Pi)$ will be wrong. The routine P-7 is designed, if properly used, so that this cannot happen under the α-option, nor can it occur under the β-option since SORT II removes SCP before a call is made to VALR-7 (see Flow Chart 6). If $N \neq 1$ or is not greater than two, as an integer, IND is set to three and an exit is made. Such values of N are not allowed.

IOP is set by the user to 1, 2, or 3 to obtain approximately 3, 6, or 9-decimal-digit accuracy, respectively, for $P(\Pi)$.

A is the location where the A-function value for Π is returned. The area of Π is given by $|A|$ (see SMP-7, pages 9, F-37).

```
      SUBROUTINE  SORT1 ( X,Y,N,P,ICV,IND,IOP,A )
      DIMENSION X(1),Y(1)
      DATA CST/5.E-14 /
      P=0.
      A=0.
      IC=IABS(ICV)
      IF ( ABS(X (N )-X (1)).GT.CST )  GO TO 2311
      IF ( ABS(Y (N )-Y (1)).GT.CST )  GO TO 2311
      GO TO 2321
 2311 CONTINUE
      N=N+1
 2321 CONTINUE
      X(N)=X(1)
      Y(N)=Y(1)
      J1ST=2
      I1=2
 2331 CONTINUE
      IF ( ICV.GT.0 )  GO TO 2361
      NUMP1=N+1
      DO 2351  J1=J1ST,N
      J =NUMP1-J1
      JP1=J+1
      DO 2341  K=JP1,N
      IF ( ABS(X (J )-X (K )).GT.CST )  GO TO 2341
      IF ( ABS(Y (J )-Y (K )).GT.CST )  GO TO 2341
      IST=J
      IEN=K
      J1ST=N-K+1
      IF ( K.EQ.N )  J1ST=2
      LST=IST+1
      GO TO 2531
 2341 CONTINUE
 2351 CONTINUE
 2361 CONTINUE
      DO 2521  I=I1,N
      IM1=I-1
      DO 2511  K1=1,IM1
      K=I-K1
      IF ( ABS(X (I )-X (K )).GT.CST )  GO TO 2511
      IF ( ABS(Y (I )-Y (K )).GT.CST )  GO TO 2511
      IST=K
      IEN=I
      I1=K
      LST=IST
      IF ( K.NE.1 )  GO TO 2531
      I1=2
      LST=LST+1
      GO TO 2531
```

```
2511  CONTINUE
2521  CONTINUE
2531  CONTINUE
      NUM1=IEN-IST
      NSAV=NUM1
      IF ( NUM1.LE.2 )  GO TO 2575
      IF (  IC.EQ.2 )  GO TO 2565
      CALL SORT2 ( X (IST),Y (IST),NUM1 )
      IF ( NUM1.LT.3 )  GO TO 2575
2565  CONTINUE
      CALL VALR7 ( X(IST),Y(IST),NUM1,SMP,IOP,SMA,IND )
      IF ( IND.EQ.2 )  RETURN
      A=A+SMA
      P=P+SMP
      X(IST)=X(IEN)
      Y(IST)=Y(IEN)
2575  CONTINUE
      IF ( IEN.NE.N   )  GO TO 2577
      IF ( IST.EQ.1 )  RETURN
      X(IST)=X(N)
      Y(IST)=Y(N)
      N=IST
      GO TO 2331
2577  CONTINUE
      N=N-NSAV
      DO 2581  L=LST,N
      K=L+NSAV
      X(L)=X(K)
      Y(L)=Y(K)
2581  CONTINUE
      GO TO 2331
      END
```

SUBROUTINE SORT II (See Appendix A)

(FLOW CHART 7, page A-19)

Subroutine SORT II Used to Eliminate Successive Colinear Points in $\bar{S}$

CALL: SORT II (x, y, M),

where:

x is the array of abscissas of the numbered points of the polygon $\bar{S}$. The array is dimensioned at M. Upon return to the calling program SORT I, the array of abscissas will be reduced by the number of points deleted, because of SCP. The x array is compacted or closed up.

y is the array of ordinates of the numbered points of the polygon $\bar{S}$. The array is dimensioned at M. Upon return to the calling program SORT I, the array of ordinates will be reduced by the number of points deleted, because of SCP. The reduced y array is compacted or closed up.

M is the number of points of the polygon $\bar{S}$ that are numbered. Upon return to the calling program, SORT I, M will be reduced by the number of successive colinear points that were eliminated.

```
      SUBROUTINE SORT2 ( X,Y,N )
      DIMENSION X(1),Y(1)
      DATA CST/4.9E-27 /
      K=1
      L=2
      U=X(1)-X(N)
      V=Y(1)-Y(N)
      D2=U*U+V*V
3051  CONTINUE
      W=X(L)-X(1)
      Z=Y(L)-Y(1)
      D1=W*W+Z*Z
      T=V*W-U*Z
      SN=(4.*T*T)/(D1*D2)
      IF ( SN.GT.CST )  GO TO 3071
      L=L+1
      IF ( L.LT.N )  GO TO 3051
      N=2
      RETURN
3071  CONTINUE
      K=2
      IF ( L.NE.2 )  GO TO 3081
      L=3
      GO TO 3111
3081  CONTINUE
      LM2=L-2
      N=N-(LM2)
      DO 3091  I=1,N
      I1=LM2+I
      X(I)=X(I1)
      Y(I)=Y(I1)
3091  CONTINUE
3101  CONTINUE
      L=K+1
      W=X(K)-X(K-1)
      Z=Y(K)-Y(K-1)
      D1=W*W+Z*Z
3111  CONTINUE
      U=X(L)-X(K)
      V=Y(L)-Y(K)
      D2=U*U+V*V
      T=V*W-U*Z
      SN=(4.*T*T)/(D1*D2)
      IF ( SN.GT.CST    )  GO TO 3121
      L=L+1
      IF ( L.LE.N )  GO TO 3111
      X(K)=X(N)
      Y(K)=Y(N)
```

```
      N=K
      GO TO 3151
3121  CONTINUE
      IF ( L.EQ.(K+1) )  GO TO 3171
      LM2=L-2
      I3=LM2-(K-1)
      N=N-I3
      DO 3131  I=K,N
      I1=I3+I
      X(I)=X(I1)
      Y(I)=Y(I1)
3131  CONTINUE
      K=K+1
      IF ( K.LT.N )  GO TO 3101
3151  CONTINUE
      U=X(N)-X(N-1)
      V=Y(N)-Y(N-1)
      D2=U*U+V*V
3161  CONTINUE
      W=X(1)-X(N)
      Z=Y(1)-Y(N)
      D1=W*W+Z*Z
      T=V*W-U*Z
      SN=(4.*T*T)/(D1*D2)
      IF ( SN.LE.CST    )  GO TO 3165
      RETURN
3165  CONTINUE
      N=N-1
      RETURN
3171  CONTINUE
      K=K+1
      IF ( K.GE.N )  GO TO 3161
      D1=D2
      W=U
      Z=V
      L=K+1
      GO TO 3111
      END
```

SUBROUTINE SMP-7

(No flow chart given)

SMP-7 is Used to Compute the a-Function*

CALL: SMP-7 (M, a, x, y)†,

where:

M is the number of input points specifying the polygon.

a is the location to which the a-function is returned.

x is the array of input abscissas. Dimensioned at M.

y is the array of input ordinates. Dimensioned at M.

(See Appendix D for value of a in the wz-plane.)

---

*The expression used to compute the a-function is given by

$$a = \frac{1}{2}\sum_{1}^{N} x_i(y_{i+1} - y_{i-1}), \quad y_0 = y_N, \quad y_{N+1} = y_1, \quad \text{(See Appendix D)}$$

(The area of the input polygon is given by |a|.)

†M, a are used in place of N, A to avoid confusion with the latter quantities in P-7 when it calls SORT 1, and SORT 1 in turn calls VALR-7. See Flow Charts 5 and 6, pages (A-17, 18).

```
      SUBROUTINE SMP7  ( NB,ANS,X,Y )
      DIMENSION X(1),Y(1)
      IF ( NB.GT.3 )  GO TO 3151
      ANS=.5*((X(2)-X(1))*(Y(3)-Y(1))-(X(3)-X(1))*(Y(2)-Y(1)))
      RETURN
3151  CONTINUE
      NBM1=NB-1
      ANS=X(1)*(Y(2)-Y(NB) )+X(NB)*(Y(1)-Y(NBM1))
      DO 3161  I=2,NBM1
      ANS=ANS+X(I)*(Y(I+1)-Y(I-1) )
3161  CONTINUE
      ANS=.5*ANS
      RETURN
      END
```

APPENDIX G

TRIANGLE CHECKOUT PROGRAM WITH DREZNER

(No Flow Chart)

SUBROUTINE DZ

TRIANGLE CHECKOUT PROGRAM with DREZNER (See page 47)

CALL: DZ (x, y, N, P, A),

where:

x is the array of abscissas of the points specifying polygon Π. x is dimensioned at N.

y is the array of ordinates of the points specifying Π. y is dimensioned at N.

N is the number of points specifying Π.

P is the location where P(Π) is returned.

A is the location where A(Π) is returned.

This subroutine decomposes Π into N − 2 triangles Δj with the vertices given by (1), (j), (j + 1), j = 2, ..., N − 1. P(Δj) is computed by DZ − 1 and A(Δj) by SMP-7; the results are summed in DZ, i.e., $P(\Pi) = \Sigma_2^{N-1} P(\Delta j)$, $A(\Pi) = \Sigma_2^{N-1} A(\Delta j)$.

This routine requires computation of erf (x) and erfc (x) which are defined on pages 5, 28 and 29. We have

$$\text{ERF 1 (x)} = \text{erf (x)}, \quad \text{ERFC 1 (0, x)} = \text{erfc (x)},$$

where the subroutine listings for these functions are given on pages F-12 to F-15. They are identical to the NSWC (DL) math library functions ERF and ERFC as of June 1980 which are based on the reference below.

Cody, W. J., *Rational Chebyshev Approximations for the Error Function*, Mathematics of Computation, v. 23 (1969), pp. 631–637.

```
      SUBROUTINE DZ  ( X,Y,N,ANS,A ,IOP )
      DIMENSION X(1),Y(1),U(4),V(4)
      A=0.
      ANS=0.
      IF ( N.NE.1 )  GO TO 3031
      CALL DZ1 ( X,Y,N,ANS,IOP,A )
      RETURN
3031  CONTINUE
      IF ( N.LT.3 )  RETURN
      L=3
      U(1)=X(1)
      U(2)=X(2)
      U(3)=X(3)
      V(1)=Y(1)
      V(2)=Y(2)
      V(3)=Y(3)
3041  CONTINUE
      CALL DZ1 (U,V,3,ANS1,IOP,A1 )
      A=A+A1
      ANS=ANS+ANS1
3061  CONTINUE
      L=L+1
      IF ( L.GT.N )  RETURN
      U(2)=U(3)
      V(2)=V(3)
      U(3)=X(L)
      V(3)=Y(L)
      GO TO 3041
      END
```

SUBROUTINE DZ-1

Computes P(Δj) for DZ

CALL: DZ-1 (x, y, N, P, IOP, A)*,

where:

x is the array of abscissas of the points specifying a simple polygon S.

y is the array of ordinates of the points specifying a simple polygon S. x and y are dimensioned at N + 1.

N is the number of points specifying S.

IOP is specified by the user.

IOP = 1 for 3-decimal-digit accuracy for P(S).

IOP = 2 for 6-decimal-digit accuracy for P(S).

IOP = 3 for 9-decimal-digit accuracy for P(S).

P, A are the locations where the values of P(S) and A(S) are returned, respectively.

For each angular region a of S specified by R, θ_1, θ_2, DZ-1 computes the corresponding Drezner arguments m, k, ρ as indicated in [2, Eq. 60]. Subroutine PLAN uses Drezner's algorithm to determine which equation of [†] is used to find P(a). Functions EQ 7, EQ 8, EQ 9 and EQ 11 of PLAN compute P(a) using equations 7, 8, 9, and 11, respectively, of [†]. Subroutine BPHI uses equation 5 of [†] to compute P(a).

*DZ-1 computes P(3) by Drezner's procedure which is described in [2].

†Z. Drezner, *Computation of the Bivariate Normal Integral*, **Mathematics of Computation**, v. 32 (1978), pp. 277-279.

```
      SUBROUTINE DZ1  ( X,Y,N,ANS,IOP,A )
      DIMENSION X(1),Y(1),H(2),APH1(3)
      DATA ( APH1(I),I=1,3 ) /
     1  2.02E-7,2.08E-13,2.72E-19 /
      DATA RT2 / 1.4142 13562 3731/
      DATA TWOPI/6.2831 85307 17958 /
      K=1
      ANS=0.
      IF ( N.NE.1 )  GO TO 3071
      W=X(2)-X(1)
      Z=Y(2)-Y(1)
      U=X(3)-X(1)
      V=Y(3)-Y(1)
      PSI1=(V*W-U*Z)
      IF ( PSI1.GE.0.       )  GO TO 3081
      ANS=+1.
       T1=W
      W=U
      U=T1
      T1=V
      V=Z
      Z=T1
      GO TO 3081
 3071 CONTINUE
      X(N+1)=X(1)
      Y(N+1)=Y(1)
      CALL SMP7 ( N,A,X,Y )
      IF ( ABS(A).LE.0.6512E-10 )  RETURN
      W=X(1)-X(N)
      Z=Y(1)-Y(N)
      U=X(2)-X(1)
      V=Y(2)-Y(1)
 3081 CONTINUE
      BGD1=SQRT(2.*(W*W+Z*Z))
      BGD2=SQRT(2.*(U*U+V*V))
 3151 CONTINUE
      B=.5*(X(K)*X(K)+Y(K)*Y(K))
      IF ( B.GT.APH1(IOP) )  GO TO 3155
      T1=V*W-U*Z
      T2=U*W+V*Z
      PHIK=ATAN2(T1,T2 )
      ANS1=PHIK/TWOPI
      GO TO 3211
 3155 CONTINUE
      RTR=(2.*    (W*V-U*Z))/(BGD1*BGD2)
      H(1)=(-Y(K)*W+X(K)*Z)/BGD1
      H(2)=(-Y(K)*U+X(K)*V)/BGD2
      SGN=1.
```

```
      IF ( RTR.GE.0. )  GO TO 3161
      RTR=-RTR
      SGN=-1.
      T1=H(1)
      H(1)=H(2)
      H(2)=T1
3161  CONTINUE
      AM    =-RT2*H(2)
      AK    =RT2*H(1)
      RHO=(-2.*(W*U+V*Z))/(BGD1*BGD2)
      IF ( ABS(RHO).LT.(1.-1.E-13) )  GO TO 3181
      IF ( RHO.LT.0. )  GO TO 3171
      T1=AM
      IF ( AK.LE.AM )  T1=AK
      T2=-T1/RT2
      ANS1=.5*ERFC1(0,T2 )
      GO TO 3191
3171  CONTINUE
      ANS1=0.
      IF ( AK.LE.-AM )  GO TO 3191
      T1=-AK/RT2
      T2=AM/RT2
      ANS1=.5*(ERFC1(0,T1)-ERFC1(0,T2))
      GO TO 3191
3181  CONTINUE
      CALL PLAN ( AM    ,AK    ,RHO    ,ANS1,IOP,RTR )
3191  CONTINUE
      ANS1=SGN*ANS1
3211  CONTINUE
      IF ( K.NE.N )  GO TO 3651
      IF ( N.NE.1 )  GO TO 3631
      ANS=ABS(ANS-ABS(ANS1))
      RETURN
3631  CONTINUE
      ANS=ANS-ANS1
      IF ( A.LT.0. )  GO TO 3641
      ANS=ANS+1.
      RETURN
3641  CONTINUE
      ANS=ANS-1.
      RETURN
3651  CONTINUE
      K=K+1
      KP1=K+1
      W=U
      Z=V
      U=X(KP1)-X(K)
      V=Y(KP1)-Y(K)
```

```
BGD1=BGD2
BGD2=SQRT(2.*(U*U+V*V))
ANS=ANS-ANS1
GO TO 3151
END
```

```
      SUBROUTINE PLAN ( H,AK,R,ANS,IOP ,RTR )
      ANS=0.
      IF ( (H*AK*R).GT.0. )  GO TO 3155
      IF ( H.GT.0. )  GO TO 2031
      IF ( AK.GT.0. )  GO TO 2021
      IF ( R.GT.0. )  GO TO 2011
      ANS=BPHI(H,AK,R,IOP,RTR )
      RETURN
2011  CONTINUE
      IF ( AK.NE.0. )  GO TO 2061
      GO TO 2023
2021  CONTINUE
      IF ( R.LT.0. )  GO TO 2041
2023  CONTINUE
      ANS=EQ9(H,AK,R,IOP,RTR )
      RETURN
2031  CONTINUE
      IF ( AK.EQ.0. )  GO TO 2051
2035  CONTINUE
      IF ( AK.LT.0. )  GO TO 2061
2041  CONTINUE
      ANS=EQ7(H,AK,R,IOP,RTR )
      RETURN
2051  CONTINUE
      IF ( R.GT.0. )  GO TO 2061
      GO TO 2041
2061  CONTINUE
      ANS=EQ8(H,AK,R,IOP,RTR )
      RETURN
3155  CONTINUE
      ANS=EQ11(H,AK,R,IOP,RTR )
      RETURN
      END
```

```
      FUNCTION  EQ7 (H,AK,R,IOP,RTR )
      DATA RT2/1.4142 13562 3731/
      T=-H/RT2
      T1=-AK/RT2
      EQ7=BPHI(-H,-AK,R,IOP,RTR )
     1  +.5*(ERFC1(0,T)+ERFC1(0,T1))-1.
      RETURN
      END
```

```
      FUNCTION  EQ8 (H,AK,R,IOP,RTR )
      DATA RT2/1.4142 13562 3731/
      T=-AK/RT2
      EQ8=-BPHI(-H,AK,-R,IOP,RTR)+.5*ERFC1(0,T  )
      RETURN
      END
```

```
      FUNCTION  EQ9 (H,AK,R,IOP,RTR )
      DATA RT2/1.4142 13562 3731/
      T=-H/RT2
      EQ9=-BPHI(H,-AK,-R,IOP,RTR)+.5*ERFC1(0  )
      RETURN
      END
```

```
      FUNCTION  EQ11(H,AK,R,IOP,RTR )
      DATA RT2/1.4142 13562 3731/
      CST=SQRT(H*H-2.*R*H*AK+AK*AK )
      T1=R*H-AK
      C1=1.
      T2=SIGN(C1,H)
      T1=(T1*T2)    /CST
      T4=1.
      T3=H*AK
      T5=SIGN(T4,T3)
      TDEL=(1.-    T5)*.25
      T3=R*AK-H
      C1=1.
      T2=SIGN(C1,AK)
      T3=(T3*T2)     /CST
      RTR1=(RTR*ABS(H))/CST
      RTR3=(RTR*ABS(AK))/CST
      IF ( H.GT.0. )  GO TO 2031
      IF ( T1.GT.0. )  GO TO 2023
      T4=BPHI(H,0.,T1,IOP,RTR1)
      GO TO 2051
2023  CONTINUE
      T4=EQ9(H,0.,T1,IOP,RTR1)
      GO TO 2051
2031  CONTINUE
      IF ( T1.LT.0. )  GO TO 2041
      T4=.5-BPHI(-H,0.,-T1,IOP,RTR1 )
      GO TO 2051
2041  CONTINUE
      C1=-H/RT2
      T4=BPHI(-H,0.,T1,IOP,RTR1)-.5*ERF1(C1)
2051  CONTINUE
      IF ( AK.GT.0. )  GO TO 3031
      IF ( T3.GT.0. )  GO TO 3023
      T6=BPHI(AK,0.,T3,IOP,RTR3)
      GO TO 3051
3023  CONTINUE
      T6=EQ9(AK,0.,T3,IOP,RTR3 )
      GO TO 3051
3031  CONTINUE
      IF ( T3.LT.0. )  GO TO 3041
      T6=.5-BPHI(-AK,0.,-T3,IOP,RTR3 )
      GO TO 3051
3041  CONTINUE
      C1=-AK/RT2
      T6=BPHI(-AK,0.,T3,IOP,RTR3 )-.5*ERF1(C1)
3051  CONTINUE
      EQ11=T4+T6-TDEL
```

```
      RETURN
      END
```

```
      FUNCTION BPHI ( H,AK,R ,IOP,RTR )
      DIMENSION A(21),X(21),LLO(6),LHI(6)
      DIMENSION EPS1(11 )
      DIMENSION EPS3(11)
      DATA ( A(I),I=1,8 ) /
     1  4.4602 97704 66658E-1,  3.9646 82669 98335E-1,
     2  4.3728 88798 77644E-2,  2.4840 61520 28443E-1,
     3  3.9233 10666 52399E-1,  2.1141 81930 76057E-1,
     4  3.3246 66035 13439E-2,  8.2485 33445 15628E-4 /
      DATA ( X(I),I=1,8 ) /
     1  1.9055 41497 98192E-1,  8.4825 18675 44577E-1,
     2  1.7997 76578 41573E+0,  1.0024 21519 68216E-1,
     3  4.8281  39660 46201E-1,  1.0609 49821 52572E+0,
     4  1.7797 29418 52026E+0,  2.6697 60356 08766E+0 /
      DATA ( A(I),I=9,16 ) /
     1  1.3410 91884 53360E-1, 2.6833 07544 72640E-1,
     2  2.7595 33979 88422E-1, 1.5744 82826 18790E-1,
     3  4.4814 10991 74625E-2, 5.3679 35756 02526E-3,
     4  2.0206 36491 32407E-4, 1.1925 96926 59532E-6 /
      DATA ( X(I),I=9,16 ) /
     1  5.2978 64393 18514E-2, 2.6739 83721 67767E-1,
     2  6.1630 28841 82402 E-1, 1.0642 46312 11623E+0,
     3  1.5888 55862 27006E+0, 2.1839 21153 09586E+0,
     4  2.8631 33883 70808E+0, 3.6860 07162 72440E+0 /
      DATA ( EPS1(I),I=1,3 ) / -8.,-12.,-20. /
      DATA PI / 3.1415 92653 58979 /
      DATA ( LLO(I),I=1,3 ) / 1,4,9 /
      DATA ( LHI(I),I=1,3 ) / 3,8,16 /
      DATA RT2 / 1.4142 13562 3731/
      DATA ( EPS3(I),I=1,3 ) / 2.E-5,2.E-7,2.E-10 /
      ILO=LLO(IOP)
      IHI=LHI(IOP)
      EPS=EPS1(IOP)
      CST=RT2*RTR
      BPHI=0.
      H1=H/CST
      AK1=AK/CST
      SUM=0.
      DO 3361  I=ILO,IHI
      SUM1=0.
      DO 3351  J=ILO,IHI
      T1=H1*(2.*X(I)-H1)+AK1*(2.*X(J)-AK1)
     1  +2.*R*(X(I)-H1)*(X(J)-AK1)
      IF ( T1.LT.EPS )  GO TO 3351
      SUM1=SUM1+EXP(T1)*A(J)
3351  CONTINUE
      SUM=SUM+A(I)*SUM1
3361  CONTINUE
```

```
      BPHI=(SUM*RTR)/PI
      RETURN
      END
```

DISTRIBUTION LIST

Chief of Naval Operations
Department of the Navy
Washington, D.C. 20350
Attn: OP-980
OP-982
OP-982E
OP-982F
OP-983
OP-987
OP-961

Commander, David W. Taylor
Naval Ship Research and Development Center
Bethesda, MD 20034
Attn: Code 18
Code 154
Code 184
Code 1541
Code 1802
Code 1805
Library

Commander, Naval Facilities Engineering Command
Department of the Navy
Washington, D.C. 20390
Attn: 0322

ECOM Office Building
U.S. Army Electronic Command
Fort Monmouth, New Jersey 07703
Attn: Technical Library

Director, Naval Research Laboratory
Washington, D.C. 20390
Attn: Code 7800
Library

Office of Naval Research
Washington, D.C. 20360
Attn: Math and Information Sciences Division
Library

Commander, Naval Air Systems Command
Department of the Navy
Washington, D.C. 20360
Attn: Code NAIR-03
Code NAIR-03D
Code NAIR-5034

Commander, Naval Electronics Systems Command
Department of the Navy
Washington, D.C. 20360
Attn: Code NELEX-03A

Commander, Naval Sea Systems Command
Department of the Navy
Washington, D.C. 20362
Attn: SEA 03A
SEA 034E
SEA 035
SEA 035B

Fleet Analysis Center
Naval Weapons Station
Seal Beach
Corona, California 91720
Attn: Library

Commander, U.S. Naval Weapons Center
China Lake, California 93555
Attn: Code 6073
Code 60704
Library

Naval Sea Systems Command
Department of the Navy
Washington, D.C. 20362

U.S. Naval Observatory
34th Street and Massachusetts Avenue, N.W.
Washington, D.C. 20390
Attn: Library

U.S. Naval Oceanographic Office
Washington, D.C. 20390
Attn: Code 0814
Library

Commander, Harry Diamond Laboratory
2800 Powdermill Road
Adelphi, MD 20783

AFADSB Headquarters, U.S. Air Force
Washington, D.C. 20330

Director
Defense Research and Engineering
Washington, D.C. 20390
Attn: WSEG
Deputy Director, Tactical Warfare Programs
Deputy Director, Test and Evaluation

The Library of Congress
Washington, D.C. 20540
Attn: Exchange and Gift Division (4)

Director
Defense Intelligence Agency
Washington, D.C. 20301

Lawrence Radiation Laboratory
Technical Information Department
P.O. Box 808
Livermore, California 94550

Sandia Corporation
Livermore Branch
P.O. Box 969
Livermore, California 94550
Attn: Technical Library

Numerical Analysis Research Library
University of California
405 Hilgard Avenue
Los Angeles, California 90024

Superintendent
U.S. Naval Postgraduate School
Monterey, California 93940
Attn: Library, Tech Reports Section

Director
Office of Naval Research Branch Office
1030 East Green Street
Pasadena, California 91101

Commanding Officer
Marine Air Detachment
Naval Missile Center
Point Mugu, California 93041

Office of Naval Research
Branch Office, Chicago
219 South Dearborn Street
Chicago, Illinois 60604

Superintendent
U.S. Naval Academy
Annapolis, Maryland 21402
Attn: Library, Serials Division

Commanding Officer
U.S. Army Aberdeen R&D Center
Aberdeen, Maryland 21005
Attn: Dr. F. E. Grubbs
Library

Director
U.S. Army Munitions Command
Edgewood Arsenal, Maryland 21010
Attn: Operations Research Group

Director
National Security Agency
Fort George G. Meade, Maryland 20755
Attn: Dr M. Kupperman
Library

Director
National Bureau of Standards
Gaithersburg, Maryland 20760
Attn: Library

Director
National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, Maryland 20771
Attn: Library

Commanding Officer
Naval Weapons Evaluation Facility
Kirtland Air Force Base
Albuquerque, New Mexico 87117

Los Alamos Scientific Laboratory
P.O. Box 1663
Los Alamos, New Mexico 87544
Attn: Report Library

Commanding General
White Sands Missile Range
Las Cruces, New Mexico 88002
Attn: Technical Library, Documents Section

Air Force Armament Laboratory
Eglin Air Force Base, Florida 32542

Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439
Attn: Dr. A. H. Jaffey, Building 200

The RAND Corporation
4921 Auburn Avenue
Bethesda, Maryland 20014
Attn: Library

The RAND Corporation
1700 Main Street
Santa Monica, California 90406 (2)

University of Chicago
Chicago, Illinois 60637
Attn: Prof. W. Kruskel, Statistics Dept.

Journal of Mathematics and Mechanics
Mathematics Department, Swain Hall East
Indiana University
Bloomington, Indiana 47401

NASA Scientific and Technical Information Facility
P.O. Box 33
College Park, Maryland 20740

The Johns Hopkins University
Applied Physics Laboratory
Johns Hopkins Road
Laurel, MD 20810
Attn: Strategic Analysis Support Group
Document Librarian

Prof. George F. Carrier
Pierce Hall, Room 311
Harvard University
Cambridge, Massachusetts 02138

Massachusetts Institute of Technology
Cambridge, Massachusetts 02139
Attn: Computation Center

University of Michigan
Institute of Science and Technology
P.O. Box 618
Ann Arbor, Michigan 48107
Attn: Operations Research Division
Dr. E. H. Jebe

Joint Strategic Target Planning Staff
Offutt Air Force Base, Nebraska 68113 (2)

President
Naval War College
Newport, Rhode Island 02840

Prof. M. Albertson
Department of Civil Engineering
Colorado State University
Fort Collins, Colorado 80521

California Institute of Technology
Pasadena, California 91109
Attn: Prof. T. Y. Wu

Rutgers University
Statistics Center
New Brunswick, New Jersey 08903

Dr. George Ioup, Physics Department
Louisiana State University
Lake Front
New Orleans, Louisiana 70122

ETL Analysis
3460 Olney-Laytonsville Rd.
Olney, Maryland 20832
Attn: P. Schoenfeld

Pennsylvania State University
University Park, Pennsylvania 16802
Attn: Prof. P. C. Hammer, Computer Science Department

Defense Technical Information Center
Cameron Station
Alexandria, Virginia 22314 (12)

Langley Aeronautical Laboratory
National Aeronautics and Space Administration
Langley Field, Virginia 23365
Attn: Mr. J. B. Parkinson

Prof. Gary Makowski
Department of Math and Statistics
Marquette University
Milwaukee, Wisconsin 53233

University of Wyoming
Statistics Department
Box 3275, University Station
Laramie, Wyoming 82070
Attn: Dr. W. C. Guenther
Mr. J. Terragno

Mrs. Pamela M. Morse
Canada Department of Agriculture
Sir John Carling Building, Room E265
Statistical Research Service
C.E.F. Ottawa, Ontario
Canada

Aeronautical & Mechanical Engineering Library
National Research Council
Montreal Road
Ottawa, Ontario K1A ORG

Vitro Laboratories
14000 Georgia Avenue
Silver Spring, Maryland 20910
Attn: D. Abell

V.P.I. and State University
Blacksburg, Virginia 24060
Attn: Dr. J. Arnold, Statistics Dept.
Dr. R. H. Myers, Statistics Dept.

Dr. Donald Amos
Division 5122
Sandia Laboratories
Albuquerque, New Mexico 87115

Alan B. Bligh
Code 7810
Naval Research Laboratory
Washington, D.C. 20375

Gene H. Gleissner
Code 18
Naval Ship Research and Development Center
Bethesda, Maryland 20034

Allen Miller
Code 1723
Naval Research Laboratory
Washington, D.C. 20375

Dr. Richard Nance
Department of Computer Science
562 McBryde Hall
V.P.I. and State University
Blacksburg, Virginia 24061

B. L. Ball QEC.
Naval Weapons Station
Seal Beach, California 90740

Director Tradoc System Analysis Activity
USA TRASANA
ATAA – TDS
White Sands Missile Range
New Mexico 88002

Donald Baker Moore
Exploratory Technology
P.O. Box KK
Fairfield, California 94533

I. Sugai 8-100
The Johns Hopkins University
Applied Physics Laboratory
Johns Hopkins Road
Laurel, Maryland 20810

Dr. M. P. Jarnagin, Jr.
c/o Mrs. G. L. Griffeth
4282 Roswell Rd. – Apt. H2
Atlanta, Georgia 30342

Prof. George Marsaglia
Computer Science Department
Avery 451
Washington State University
Pullman, Washington 99164

U.S. Army Electronic Command
ECOM Office Building
Ft. Monmouth, New Jersey 07703
Attn: Technical Library

Prof. H. Goldstein
Columbia University
520 W. 120th Street, Engter Bldg.
New York, New York 10027

Prof. J. Gurland
University of Wisconsin-Madison
Department of Statistics
1210 West Dayton St.
Madison, Wisconsin 53706

Mr. John A. Simpson
Senior Systems Engineer
Norden, Div. of United Technologies
Norwalk, Connecticut 06856

M. Miller
Schering Corporation
Bloomfield, New Jersey 07003

Dr. David Giri
Air Force Weapons Lab./ELP
Kirtland Air Force Base
New Mexico 87117

Dr. Joo Koo
NIEHS, Biometry Branch
Box 12233
Research Triangle Park, North Carolina 27709

Harvey S. Picker
Physics Department
Trinity College
Hartford, Connecticut 06106

Energy Research & Development Admin.
Division of Military Applications
Washington, D.C. 20545
Attn: Library

Naval Ocean Systems Center
San Diego, California 92152
Attn: Library

H. Saunders
Building 41 – Room 319
General Electric Company
One River Road
Schenectady, New York 12345

R. F. Hausman
Lockheed Missile & Space Company
Department 6213 – Building 104
P.O. Box 504
Sunnyvale, California 94088

ASD/XRZ
Wright-Patterson Air Force Base
Ohio 45433
Attn: Dr. J. Becsey

Melvin Cohen
Computer Science Department
McGill University
805 Sherbrooke Street, West
Montreal, Quebec
Canada H3A – 2K6

Space Sciences Laboratory
University of California
Berkeley, California 94720
Attn: Dr. M. Lampton

Deputy Director Marine Corps
Operations Analysis Group
2000 N. Beauregard Street
Alexandria, Virginia 22311
Attn: Dr. B. Barfoot

AFRRI
National Naval Medical Center
Bethesda, Maryland 20014
Attn: Dr. S. Levin

Mr. Harry Weingarten
LMSC
O/9440-B/150
P. O. Box 504
Sunnyvale, California 94086

Dr. Harold Crutcher
35 Westall Avenue
Asheville, N.C. 28804

Prof. D. C. Hoaglin
Dept. of Statistics
Harvard University
1 Oxford Street
Cambridge, MA 02138

Local Distribution:

D
D1
E411
F
F14 (E. Morgan)
F42 (E. Spooner)
G
G10
G10 (F. Clodius)
G32 (G. Seidl)
K
K01
K02
K04
K05D(30)
K10
K11
K11 (Dr. B. Zondek)
K20
K21
K22
K23
K30
K30 (Dr. M. Thomas) (2)
K30 (D. Snyder)
K40
K50
K50 (Dr. E. Ball)
K50 (Dr. A. Evans)
K55
K60
K70
K71
K72
K73
K74
N
N10
N10 (S. Vittoria)
N20
N30
N40
R10 (Dr. I. W. Enis)
(Dr. A. H. Van Tuyl)
X210 (6)
X211 (2)