AD-A105 858 PRINCETON UNIV NJ DEPT OF STATISTICS F/G 9/2
GEOMETRIC DATA ANALYSIS: AN INTERACTIVE GRAPHICS PROGRAM FOR SH--ETC(U)
APR 81 A F SIEGEL DAAG29-79-C-0205
UNCLASSIFIED TR-193-SER-2 ARO-16669.11-M NL

END
DATE
FILMED
11 81
DTIC

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

| REPORT DOCUMENTATION PAGE | | READ INSTRUCTIONS BEFORE COMPLETING FORM |
|---|---|---|
| 1. REPORT NUMBER 16669.11-M | 2. GOVT ACCESSION NO. AD-A105858 | 3. RECIPIENT'S CATALOG NUMBER |
| 4. TITLE (and Subtitle) Geometric Data Analysis: An Interactive Graphics Program for Shape Comparison | | 5. TYPE OF REPORT & PERIOD COVERED Technical |
| | | 6. PERFORMING ORG. REPORT NUMBER |
| 7. AUTHOR(s) Andrew F. Siegel | | 8. CONTRACT OR GRANT NUMBER(s) DAAG29-79-C-0205 |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS Princeton University Princeton, NJ 08544 | | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS |
| 11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office Post Office Box 12211 Research Triangle Park, NC 27709 | | 12. REPORT DATE Apr 81 |
| | | 13. NUMBER OF PAGES 29 |
| 14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) LEVEL II | | 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)

NA

DTIC ELECTE OCT 20 1981 S D H

18. SUPPLEMENTARY NOTES

The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

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

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

Two shapes, each consisting of n homologous points, can be rotated, scaled, and translated to obtain a close fit to each other by several methods. An interactive graphical computer program is presented here that implements two methods: least squares and repeated medians, a robust method. Examples are given and the use of the system is discussed.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 406 8[illegible]

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

# GEOMETRIC DATA ANALYSIS: AN INTERACTIVE GRAPHICS PROGRAM FOR SHAPE COMPARISON

by

Andrew F. Siegel
Princeton University

Technical Report No. 193, Series 2
Department of Statistics
Princeton University

April 1981

Research was supported in part by U.S. Army Research Office Contract DAAG29-79-C-0205 awarded to the Statistics Department, Princeton University.

# Geometric Data Analysis: An Interactive Graphics Program for Shape Comparison

by

Andrew F. Siegel

## A B S T R A C T

Two shapes, each consisting of $n$ homologous points, can be rotated, scaled, and translated to obtain a close fit to each other by several methods. An interactive graphical computer program is presented here that implements two methods: least squares and repeated medians, a robust method. Examples are given and the use of the system is discussed.

## 1. INTRODUCTION

The quantitative study of shape and form has been considered by many authors since the fundamental descriptive work of Thompson (1917). The method of least squares was used by Sneath (1967) to solve the problem of finding a common location and orientation of two shapes in order to compare their similarities and differences. A general discussion of the numerical aspects of such least squares calculations may be found in Huber (1980). There are situations in which robust methods such as the repeated median technique (Siegel, 1980) are far superior to least squares, especially in the detection of localized shape differences (Siegel and Benson, 1979).

Least squares and robust methods are both considered here, and an interactive computer program written in FORTRAN with graphics capabilities is presented. Section 2 outlines the mathematical methods involved in each fitting process; further details can be found in the references. Two examples are provided in Section 3 to illustrate the use of the program and to show the kinds of graphic results that can be obtained. Section 4 outlines the work involved in setting up the program to run on a computer installation and provides an overall description of how to use the system. The interactive commands are described in detail in Section 5, and the program is listed in the Appendix.

## 2. METHODS

Each of the two shapes is represented by a sequence of $n$ points in two dimensions. Homology is assumed, so that the $i^{th}$ point of shape 1 corresponds to the $i^{th}$ point of shape 2 because, for example, this might be the location of the base of the skull in each specimen. Establishment of homology can be straightforward in some situations, but is difficult in others. We will assume it has been done, even if only tentatively, and will also assume that any needed reflection has also already been done. Thus our data are

| Shape 1 | Shape 2 |
|---|---|
| $(x_1,y_1)$ | $(u_1,v_1)$ |
| $(x_2,y_2)$ | $(u_2,v_2)$ |
| $\vdots$ | $\vdots$ |
| $(x_n,y_n)$ | $(u_n,v_n)$ |

Holding shape 1 fixed, we transform shape 2 by a rotation angle $\theta$, a magnification factor $\rho$, and a translation $(a,b)$. The transformed coordinates of shape 2 are therefore

$$\begin{pmatrix} u_i' \\ v_i' \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} + \rho \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} u_i \\ v_i \end{pmatrix} \qquad (2.1)$$

Defining $c=\rho\cos(\theta)$ and $d=\rho\sin(\theta)$ we can reparametrize to an expression that is linear in the parameters:

$$\begin{pmatrix} u_i' \\ v_i' \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} + \begin{pmatrix} c & -d \\ d & c \end{pmatrix} \begin{pmatrix} u_i \\ v_i \end{pmatrix} . \qquad (2.2)$$

The least squares solution is found by minimizing the sum of squared distances from each point $(x_i,y_i)$ of shape 1 to the transformed point $(u_i',v_i')$ of shape 2. This sum of squares is

$$SS(a,b,c,d) = \sum_{i=1}^{n} [(x_i-u_i')^2+(y_i-v_i')^2] . \qquad (2.3)$$

Minimizing (2.3) we obtain the least squares parameter values:

$$\hat{c} = \frac{\sum_{i=1}^{n} [(u_i-\bar{u})(x_i-\bar{x})+(v_i-\bar{v})(y_i-\bar{y})]}{\sum_{i=1}^{n} [(u_i-\bar{u})^2+(v_i-\bar{v})^2]} \qquad (2.4)$$

$$\hat{d} = \frac{\sum_{i=1}^{n} [(u_i-\bar{u})(y_i-\bar{y})-(v_i-\bar{v})(x_i-\bar{x})]}{\sum_{i=1}^{n} [(u_i-\bar{u})^2+(v_i-\bar{v})^2]} \qquad (2.5)$$

$$\hat{a} = \bar{x} - (c\bar{u} - d\,\bar{v}) \qquad (2.6)$$

$$\hat{b} = \bar{y} - (c\bar{v} + d\,\bar{u}) \qquad (2.7)$$

The repeated median procedure estimates the magnification ($\rho$) and the rotation ($\theta$) separately, using a double median for each. These estimates are

$$\tilde{\rho} = \underset{1 \leqslant i \leqslant n}{\text{Median}} \left\{ \underset{\substack{1 \leqslant j \leqslant n \\ j \neq i}}{\text{Median}} \quad \rho_{ij} \right\} \tag{2.8}$$

$$\tilde{\theta} = \underset{1 \leqslant i \leqslant n}{\text{Median}} \left\{ \underset{\substack{1 \leqslant j \leqslant n \\ j \neq i}}{\text{Median}} \quad \theta_{ij} \right\} \tag{2.9}$$

where $\rho_{ij}$ and $\theta_{ij}$ are the magnification and rotation parameters computed using only points $i$ and $j$ of each shape. The transformation factors can be estimated using a single median:

$$\tilde{a} = \underset{1 \leqslant i \leqslant n}{\text{Median}} \{x_i - \tilde{\rho}[u_i \cos(\tilde{\theta}) - v_i \sin(\tilde{\theta})]\} \tag{2.10}$$

$$\tilde{b} = \underset{1 \leqslant i \leqslant n}{\text{Median}} \{y_i - \tilde{\rho}[u_i \sin(\tilde{\theta}) + v_i \cos(\tilde{\theta})]\} \tag{2.11}$$

The match obtained using repeated medians, unlike least squares, will not be led astray by a few atypical points and localized changes will be more easily identified. In general, if more than $(n+1)/2$ (i.e. just over half) of the points can be made to match closely, then the repeated median method will produce a transformation that does match them closely.

## 3. EXAMPLES

Two examples are given here in order to illustrate the possible results obtainable through the use of this system. The first example is a square (shape 1) compared to a deformed square (shape 2). Using the line drawing capability to connect points 1 to 2, 2 to 3, ..., 7 to 8, and 8 to 1, the initial shapes can be drawn using the commands d1 and d2 . This is shown in Figure 1 in which numbers have been added in order to identify homologous points.

Typing the command ℓs causes shape 2 to be transformed to the least squares fit. A display superimposing both shapes can be obtained with the command ds , resulting in Figure 2a. Residual vectors starting from the points of shape 1 and ending at the homologous points of shape 2 can be drawn using the command dr . Figure 2b shows these residuals superimposed on shape 1, using commands dr and d1 .

The robust fit by repeated medians is produced by the command rs . Using commands ds , dr , and d1 as before, we can obtain the displays of the fitted shapes and residuals shown in Figure 3 for the robust fit.

Figures 2 and 3 show that the least squares and robust fits can be very different. The least squares comparison indicates a complicated relationship between the two forms, with differences of various sizes and in various directions at all points. In contrast, the robust fit shows clearly the localized

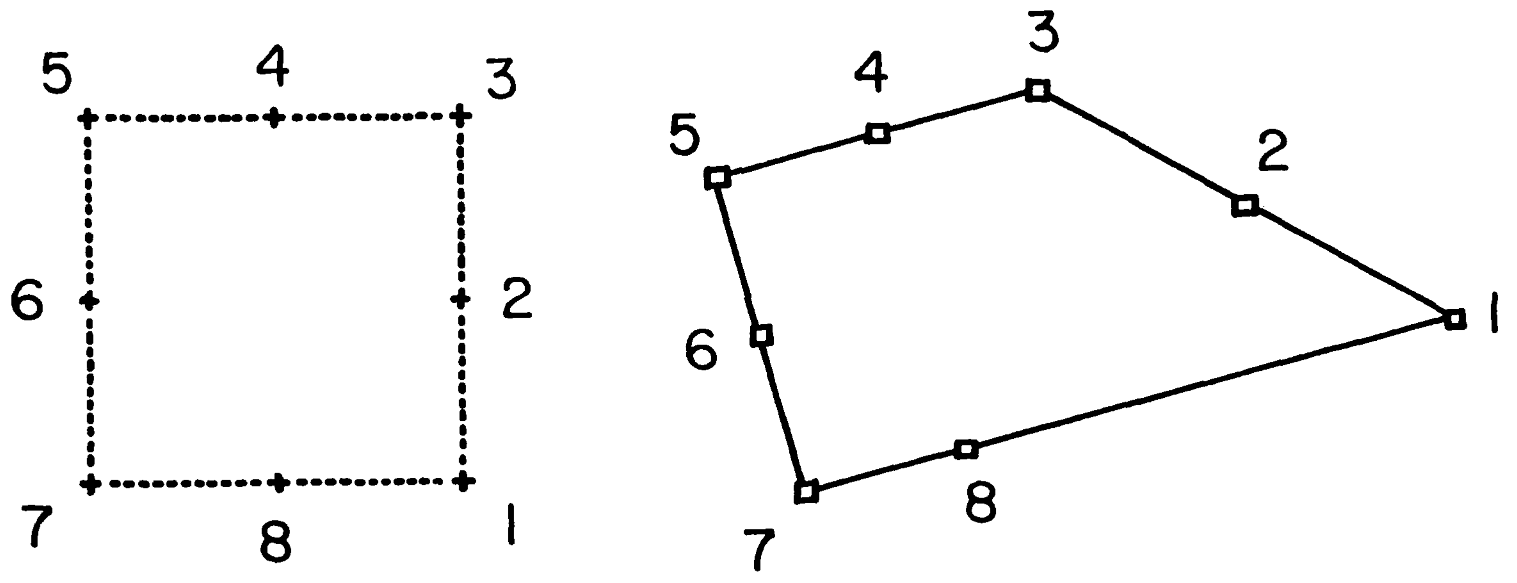

FIGURE 1. Two shapes to be compared.

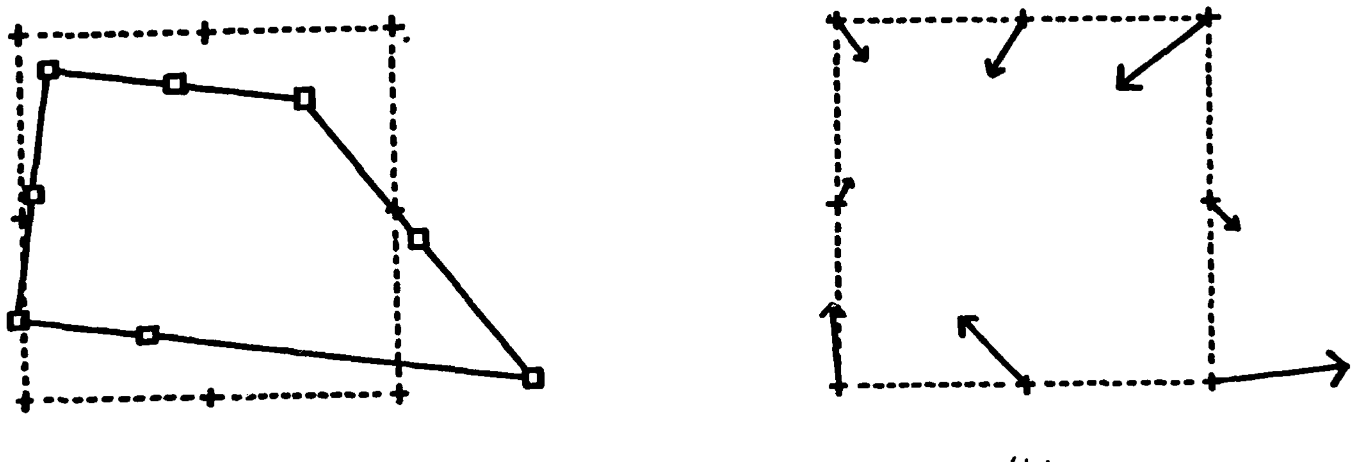

FIGURE 2. Least squares fit.

Shapes superimposed (a); residuals superimposed on shape 1 (b).

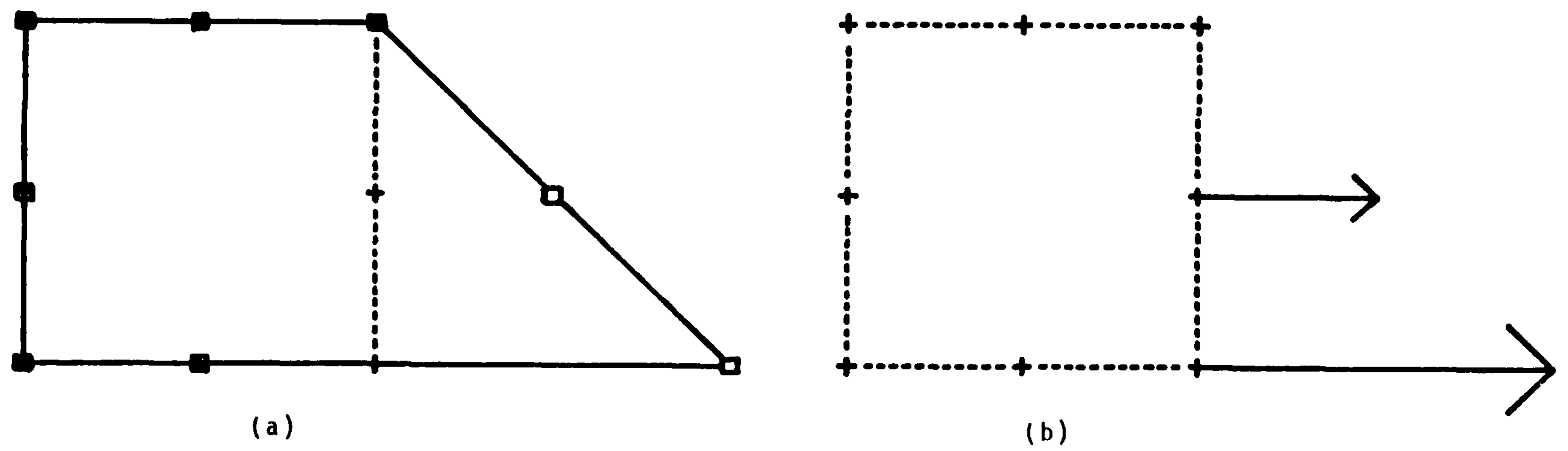

FIGURE 3. Robust fit.

Shapes superimposed (a); residuals superimposed on shape 1 (b).

systematic nature of the shape differences. All but two of the homologous pairs are matched closely, and the simple relationship between the shapes is apparent.

Figure 4 shows the comparison of a human skull to that of a chimpanzee. These data are from Sneath (1967) and the example is discussed in detail by Siegel and Benson (1979). It is included here as a demonstration of the graphics enhancement capability. This capability allows for outlines and other details that will not affect the fitting process itself (only the homologous points are used for that) but that will be transformed appropriately in order to produce a useful display.

No dashed lines are seen in Figure 4 because the line capability was not used. However, both the line and enhancement capabilities can be used simultaneously if desired.

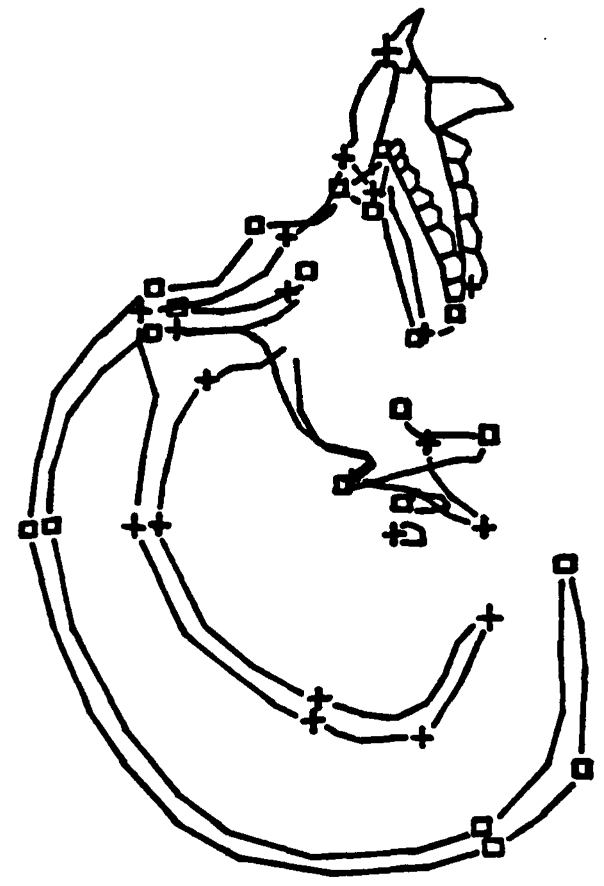

FIGURE 4. Robust comparison of primate skulls, illustrating the enhancement capability.

## 4. IMPLEMENTING AND USING THE SYSTEM

To facilitate the use of this program on different computer systems, the program is written in standard FORTRAN but four subroutines must be supplied by the user: INPUT, MOVE, DRAW, and FINISH. These subroutines are used to link to the user's database and graphics capabilities, which vary greatly from one installation to another. Detailed requirements for each subroutine are included in the program listing the Appendix after each of these four subroutine statements.

The system responds interactively to commands issued by the user. Each command is two letters (additional letters on the same line are ignored) and commands fall into four groups: fit, draw, print, and set. Commands can be issued in any order desired. The fit and set commands can be used to transform shape 2; draw and print commands will be based on the most recent orientation specified (e.g. least squares, robust, or other). For example, the sequence of commands ℓs dr d2 rs pc (one per line) will draw the residuals from the least squares fit, draw shape 2 (still in least squares configuration), and finally print the coordinates of the robust fit.

## 5. COMMAND DESCRIPTIONS

There are three fit commands: ℓs(least squares), rs(robust), and da(original data). Commands ℓs and rs respectively produce the least squares and robust fits discussed in Section 2. The original data configuration can always be brought back by using da . No printed or graphic output results from these commands, but the internal configuration of shape 2 is transformed accordingly.

There are four draw commands: ds(draw shapes), d1(draw shape 1), d2(draw shape 2), and dr(draw residual vectors). Both shapes are drawn when ds is used; shape 1 alone results when d1 is used; shape 2 alone is drawn when d2 is used. There are three components to a shape drawing: the homologous points (crosses for shape 1 and squares for shape 2), straight lines connecting pairs of points (dashed lines for shape 1, solid lines for shape 2), and enhancements (points and lines that do not affect the fitting process but allow the inclusion of outlines and details to be transformed appropriately). The residual vectors are arrows pointing from the points of shape 1 to the homologous points of shape 2. These vectors are drawn when command dr is given.

Four print commands are included: pc(print coordinates), pp(print parameters), pr(print residuals), and help(print command summary). The command pc prints the coordinates of the points of shape 1, the current values of the transformed homologous points of shape 2, and the current parameter values. The current parameter values $a,b,c,d,\rho$, and $\theta$, as defined in Section 2, are printed when pp is used. The command pr prints the residual vectors $(u(i)-x(i),v(i)-y(i))$ together with their lengths, their average length, their root mean square length, and the current parameter values. A summary of valid commands is printed when help is used.

There are four set commands: sp(set parameters), sm (modify parameters), ss(set scale), and se(set point and dash size). These commands prompt you for values to be typed in one per line with five or fewer characters per value, including the required decimal point. You may choose the transformation of shape 2 directly by specifying the parameter values $a,b,\rho$, and $\theta$ using command sp . To modify the current parameter values $a,b,\rho$, and $\theta$, the command sm will prompt you for changes $a',b',\rho'$, and $\theta'$ . The new parameter values will then be $a+a'$, $b+b'$, $\rho\rho'$, and $\theta+\theta'$ , unless $\rho'=0$ in which case $\rho$ will be left unchanged. The overall graphics size can be specified using the ss command; the default size of both shapes will be multiplied by the number entered. The size of the homologous points of both shapes and the spacing between dashes of the lines of shape 1 will both be changed by a factor entered using the se command.

The command en stops execution of the program.

## APPENDIX: FORTRAN PROGRAM

```
c  interactive system for fitting and plotting shapes.
c
c               programmed 10/80 by Andrew F. Siegel.
c
c  commands are listed by entering "help".
c
c
c  variables:
c    n = number of h-points (homologous points) of each shape.
c    m = number of pairs of h-points to be connected.
c        set m=0 if no such lines are desired.
c    nn = 500 = dimension of z, w, and ww (used for enhancement
c               of drawings).
c    ifit = 1, 2, 3, or 4 according to the current fit:
c           original data, least squares, robust, or special choice
c    coms(16) = table of commands.
c    sf = scale factor. Initially set to one, can be reset to
c         change size of drawings.
c    e = point size and dash spacing.
c    a,b = translation parameters to fit shape 2 to shape 1.
c    rho,theta = magnification and rotation parameters
c                to fit shape 2 to shape 1.
c    c,d = parameters of linear form of magnification and rotation
c          to fit shape 2 to shape 1.  (c=rho*cos(theta),
c          d=rho*sin(theta)).
c    x(n),y(n) = coordinates of h-points of shape 1.
c    uu(n),vv(n) = coordinates of original h-points of shape 2.
c    u(n),v(n) = coordinates of transformed h-points of shape 2.
c    lines(m,2) = pairs of h-points to be connected in drawing:
c                 for example, if lines(k,1)=i and lines(k,2)=j
c                 then the kth line drawn will connect the
c                 ith to the jth h-points of each shape,
c                 using a solid line for shape 1 and a dotted
c                 line for shape 2.
c    z(nn) = piecewise linear enhancements for shape 1.  for
c            example, to draw an outline connecting (a1,b1) to
c            (a2,b2) to ... to (ak,bk), store k,a1,b1,...,ak,bk
c            in the vector z().  Additional outlines can be
c            added one after another in this same form, ending
c            finally with a zero.
c    ww(nn) = original piecewise linear enhancements for shape 2.
c    w(nn)  = transformed piecewise linear enhancements for shape 2.
```

```
c     ut(n),vt(n) = temporary storage for repeated median fitting.
c
c
       dimension x(100),y(100),u(100),v(100),uu(100),vv(100),lines(200,2)
      *  ,z(500),w(500),ww(500),coms(16),ut(100),vt(100)
       data coms/'ls','rs','da','ds','dl','d2','dr','pc','pp','pr','he'
      *         ,'se','sp','sm','ss','en'/
       write(6,1)
1      format(' enter ''help'' for a list of commands.')
       nn=500
       z(1)=0.
       ww(1)=0.
       sf=1.
       e=.0075
       call input(n,nn,m,x,y,uu,vv,lines,z,ww)
       ifit=1
       a=0.
       b=0.
       c=1.
       d=0.
       call coord(n,nn,a,b,c,d,u,v,uu,vv,w,ww)
10     read(5,2)com
2      format(a2)
       if(com.eq.coms(1)) call ls(n,nn,a,b,c,d,ifit,x,y,u,v,uu,vv,w,ww)
       if(com.eq.coms(2)) call rs(n,nn,a,b,c,d,ifit,x,y,u,v,uu,vv,w,ww,
      *                             ut,vt)
       if(com.eq.coms(3)) call data(n,nn,a,b,c,d,ifit,u,v,uu,vv,w,ww)
       if(com.eq.coms(4)) call ds(0,n,nn,m,ifit,sf,e,x,y,u,v,lines,w,z)
       if(com.eq.coms(5)) call ds(1,n,nn,m,ifit,sf,e,x,y,u,v,lines,w,z)
       if(com.eq.coms(6)) call ds(2,n,nn,m,ifit,sf,e,x,y,u,v,lines,w,z)
       if(com.eq.coms(7)) call dr(n,nn,ifit,sf,e,x,y,u,v,z)
       if(com.eq.coms(8)) call pc(n,a,b,c,d,ifit,x,y,u,v)
       if(com.eq.coms(9)) call pp(a,b,c,d)
       if(com.eq.coms(10)) call pr(n,a,b,c,d,ifit,x,y,u,v)
       if(com.eq.coms(11)) call help
       if(com.eq.coms(12)) call se(e)
       if(com.eq.coms(13)) call sp(n,nn,a,b,c,d,ifit,u,v,uu,vv,
      *                             w,ww)
       if(com.eq.coms(14)) call sm(n,nn,a,b,c,d,ifit,u,v,uu,vv,
      *                             w,ww)
      if(com.eq.coms(15)) call ss(sf)
      if(com.eq.coms(16)) stop
      go to 10
      stop
      end
c
c
      subroutine input(n,nn,m,x,y,uu,vv,lines,z,ww)
c  reads in coordinate and enhancement data.
      dimension x(n),y(n),uu(n),vv(n),lines(m,2),z(nn),ww(nn)
c
c         ******************************************************
c         the user must supply this subroutine to obtain the
c         number of points, n; the initial coordinates x(),y(),
c         uu(), and vv(); the values of  m  and  lines(,); and
c         the enhancement values  z()  and  ww().
c         ******************************************************
c
      return
      end
c
c
      subroutine move(x,y)
c  moves cursor to  (x,y).
c         ******************************************************
c         the user must supply this subroutine to move the
c         graphics cursor to  (x,y)  but draw nothing.  generally,
c         x  and  y  will lie between  0  and  1, although these
c         limits can be exceeded.
c         ******************************************************
      return
      end
c
c
      subroutine draw(x,y)
c  draws a line to (x,y).
c         ******************************************************
c         the user must supply this graphics subroutine to draw
c         a line from the previous cursor location to the point
c         (x,y).  the cursor location should then be updated
```

```
c         to  (x,y).
c         ******************************************************
      return
      end
c
c
      subroutine finish
c  turns off graphics mode.
c         ******************************************************
c         the user must supply this subroutine to turn off
c         graphics mode, if needed.
c         ******************************************************
      return
      end
c
c
      subroutine ls(n,nn,a,b,c,d,ifit,x,y,u,v,uu,vv,w,ww)
      dimension x(n),y(n),u(n),v(n),uu(n),vv(n),w(nn),ww(nn)
      ifit=2
      an=n
      sx=0.
      sy=0.
      su=0.
      sv=0.
      sux=0.
      suy=0.
      suu=0.
      svx=0.
      svy=0.
      svv=0.
      do 10 i=1,n
        sx=sx+x(i)
        sy=sy+y(i)
        su=su+uu(i)
        sv=sv+vv(i)
        sux=sux+uu(i)*x(i)
        suy=suy+uu(i)*y(i)
        suu=suu+uu(i)*uu(i)
        svx=svx+vv(i)*x(i)
        svy=svy+vv(i)*y(i)
```

```
10        svv=svv+vv(i)*vv(i)
       ss=suu-su**2/an+svv-sv**2/an
       c=(sux-su*sx/an+svy-sv*sy/an)/ss
       d=(suy-su*sy/an-svx+sv*sx/an)/ss
       a=(sx-c*su+d*sv)/an
       b=(sy-c*sv-d*su)/an
       call coord(n,nn,a,b,c,d,u,v,uu,vv,w,ww)
       return
       end
c
c
       subroutine rs(n,nn,a,b,c,d,ifit,x,y,u,v,uu,vv,w,ww,ut,vt)
c   robust fit by repeated medians.
       dimension x(n),y(n),u(n),v(n),uu(n),vv(n),w(nn),ww(nn),ut(n),vt(n)
       call ls(n,nn,a,b,c,d,ifit,x,y,u,v,uu,vv,w,ww)
       ifit=3
c   get magnification rho
       do 20 i=1,n
         j1=0
         do 10 j=1,n
           if(i.eq.j) go to 10
           j1=j1+1
           ut(j1)=sqrt(((x(j)-x(i))**2+(y(j)-y(i))**2)/
     *                 ((u(j)-u(i))**2+(v(j)-v(i))**2))
10         continue
20       vt(i)=fmed(n-1,ut)
       rho=fmed(n,vt)
c   get rotation theta
       do 40 i=1,n
         j1=0
         do 30 j=1,n
           if(i.eq.j) go to 30
           j1=j1+1
           x1=x(j)-x(i)
           y1=y(j)-y(i)
           u1=u(j)-u(i)
           v1=v(j)-v(i)
           ut(j1)=atan2(u1*y1-v1*x1,u1*x1+v1*y1)
30         continue
40       vt(i)=fmed(n-1,ut)
```

```
      theta=fmed(n,vt)
c  get translation vector a,b
      ct=rho*cos(theta)
      dt=rho*sin(theta)
      do 50 i=1,n
        ut(i)=x(i)-ct*u(i)+dt*v(i)
50      vt(i)=y(i)-dt*u(i)-ct*v(i)
      at=fmed(n,ut)
      bt=fmed(n,vt)
      at=at+ct*a-dt*b
      b=bt+dt*a+ct*b
      a=at
      ctt=ct*c-dt*d
      d=ct*d+dt*c
      c=ctt
      call coord(n,nn,a,b,c,d,u,v,uu,vv,w,ww)
      return
      end
c
c
      subroutine data(n,nn,a,b,c,d,ifit,u,v,uu,vv,w,ww)
c  puts original data back for shape 2.
      dimension uu(n),vv(n),u(n),v(n),ww(nn),w(nn)
      ifit=1
      a=0.
      b=0.
      c=1.
      d=0.
      call coord(n,nn,a,b,c,d,u,v,uu,vv,w,ww)
      return
      end
c
c
      subroutine coord(n,nn,a,b,c,d,u,v,uu,vv,w,ww)
c  puts transformed coordinates into u,v,w for shape 2.
      dimension u(n),v(n),uu(n),vv(n),w(nn),ww(nn)
      do 10 i=1,n
        u(i)=a+c*uu(i)-d*vv(i)
10      v(i)=b+d*uu(i)+c*vv(i)
      i=1
```

```
20    j=ww(i)
        w(i)=j
        i=i+1
        if(j.eq.0) return
        do 30 k=1,j
          w(i)=a+c*ww(i)-d*ww(i+1)
          w(i+1)=b+d*ww(i)+c*ww(i+1)
30        i=i+2
        go to 20
      end
c
c
      subroutine ds(nshape,n,nn,m,ifit,sf,e,x,y,u,v,lines,w,z)
c  draw shapes (both if nshape=0, shape 1 only if nshape=1,
c  and shape 2 only if nshape=2).
      dimension x(n),y(n),u(n),v(n),lines(m,2),w(nn),z(nn)
      call desc(ifit)
      call scale(n,s,sf,xm,ym,x,y)
      if(nshape.eq.2) go to 40
c  draw shape 1
      do 10 i=1,n
        p=(x(i)-xm)/s+.5
        q=(y(i)-ym)/s+.5
        call move(p,q+e)
        call draw(p,q-e)
        call move(p+e,q)
10      call draw(p-e,q)
      call enh(nn,xm,ym,s,z)
      if(m.eq.0) go to 40
      do 30 i=1,m
        i1=lines(i,1)
        i2=lines(i,2)
        x1=(x(i1)-xm)/s+.5
        y1=(y(i1)-ym)/s+.5
        x2=(x(i2)-xm)/s+.5
        y2=(y(i2)-ym)/s+.5
        s2=sqrt((x2-x1)**2+(y2-y1)**2)
        if(s2.lt.4.*e) go to 30
        j=(s2-3.*e)/(2.*e)
        xu=(x2-x1)/s2
```

```
         yu=(y2-y1)/s2
         exu=e*xu
         eyu=e*yu
         exu2=2.*exu
         eyu2=2.*eyu
         p=(s2-3.*e-2.*e*float(j))/2.
         x1=x1+p*xu+exu2
         y1=y1+p*yu+eyu2
         do 20 k=1,j
           call move(x1,y1)
           call draw(x1+exu,y1+eyu)
           x1=x1+exu2
20         y1=y1+eyu2
30       continue
40     if(nshape.eq.1) go to 70
c  draw shape 2
       do 50 i=1,n
         p=(u(i)-xm)/s+.5
         q=(v(i)-ym)/s+.5
         call move(p+e,q+e)
         call draw(p+e,q-e)
         call draw(p-e,q-e)
         call draw(p-e,q+e)
50       call draw(p+e,q+e)
       call enh(nn,xm,ym,s,w)
       if(m.eq.0) go to 70
       do 60 i=1,m
         i1=lines(i,1)
         i2=lines(i,2)
         x1=(u(i1)-xm)/s+.5
         y1=(v(i1)-ym)/s+.5
         x2=(u(i2)-xm)/s+.5
         y2=(v(i2)-ym)/s+.5
         s2=sqrt((x2-x1)**2+(y2-y1)**2)
         if(s2.lt.4.*e) go to 60
         xu=(x2-x1)/s2
         yu=(y2-y1)/s2
         call move(x1+2.*e*xu,y1+2.*e*yu)
         call draw(x2-2.*e*xu,y2-2.*e*yu)
60       continue
```

```
70    call finish
      return
      end
c
c
      subroutine enh(nn,xm,ym,s,z)
c  draws enhancements
      dimension z(nn)
      i=1
10    j=z(i)
        i=i+1
        if(j.eq.0) return
        x1=(z(i)-xm)/s+.5
        y1=(z(i+1)-ym)/s+.5
        call move(x1,y1)
        call draw(x1,y1)
        i=i+2
        if(j.eq.1) go to 10
        do 20 k=2,j
          call draw((z(i)-xm)/s+.5,(z(i+1)-ym)/s+.5)
20        i=i+2
        go to 10
      end
c
c
      subroutine dr(n,nn,ifit,sf,e,x,y,u,v,z)
c  draws residual vectors
      dimension x(n),y(n),u(n),v(n),z(nn)
      call desc(ifit)
      call scale(n,s,sf,xm,ym,x,y)
      do 10 i=1,n
        x1=(x(i)-xm)/s+.5
        y1=(y(i)-ym)/s+.5
        u1=(u(i)-xm)/s+.5
        v1=(v(i)-ym)/s+.5
        call move(x1,y1)
        call draw(u1,v1)
        dx=(u1-x1)/8.
        dy=(v1-y1)/8.
        call draw(x1+7.*dx+dy,y1+7.*dy-dx)
```

```
         call move(u1,v1)
10       call draw(x1+7.*dx-dy,y1+7.*dy+dx)
       call finish
       return
       end
c
c
       subroutine pc(n,a,b,c,d,ifit,x,y,u,v)
c   prints coordinates and parameters.
       dimension x(n),y(n),u(n),v(n)
       call desc(ifit)
       write(6,1) (x(i),y(i),u(i),v(i),i=1,n)
1      format(' coordinates x,y of shape 1 and u,v of shape 2'/
     *        (1x,4f10.5))
       call pp(a,b,c,d)
       return
       end
c
c
       subroutine pr(n,a,b,c,d,ifit,x,y,u,v)
c   print residuals
       dimension x(n),y(n),u(n),v(n)
       call desc(ifit)
       write(6,1)
1      format(' residuals: coordinates and lengths.')
       sr=0.
       srr=0.
       do 10 i=1,n
         rx=u(i)-x(i)
         ry=v(i)-y(i)
         r=sqrt(rx**2+ry**2)
         write(6,2) rx,ry,r
2        format(1x,3f10.5)
         sr=sr+r
10       srr=srr+r**2
       sr=sr/float(n)
       srr=sqrt(srr/float(n))
       write(6,3)sr,srr
3      format(/' average residual length = ',f10.5/
     *         ' r.m.s.  residual length = ',f10.5)
```

```
      call pp(a,b,c,d)
      return
      end
c
c
      subroutine pp(a,b,c,d)
c  print parameters.
      rho=sqrt(c**2+d**2)
      theta=atan2(d,c)*180./3.141593
      write(6,1)a,b,c,d,rho,theta
1     format(' parameters a, b, c, d, rho, theta (in degrees) are'
     *        /1x,6f10.5)
      return
      end
c
c
      subroutine help
      write(6,1)
1     format(' valid commands are:'//
     *        5x,'fit:'/10x,'least squares',17x,'ls'/
     *        10x,'robust (repeated median)',6x,'rs'/
     *        10x,'original data',17x,'da'//
     *        5x,'draw:'/10x,'shapes',24x,'ds'/
     *        10x,'shape  1  only',16x,'d1'/
     *        10x,'shape  2  only',16x,'d2'/
     *        10x,'residual vectors',14x,'dr'//
     *        5x,'print:'/10x,'coordinates',19x,'pc'/
     *        10x,'parameters',20x,'pp'/
     *        10x,'residuals',21x,'pr'/
     *        10x,'commands',20x,'help'//
     *        5x,'set:'/10x,'parameters (new)',14x,'sp'/
     *        10x,'parameters (modify)',11x,'sm'/
     *        10x,'scale',25x,'ss'/
     *        10x,'point size, dash spacing',6x,'se'//
     *        5x,'end',32x,'en'/)
      return
      end
c
c
      subroutine se(e)
```

```
c  set point size and dash spacing for drawing shapes.
      write(6,1)
1     format(' please enter desired dash and point size for drawing.')
      read(5,2)e
2     format(f5.2)
      e=.0075*e
      write(6,3)
3     format(' thank you.')
      return
      end
c
c
      subroutine ss(sf)
c  sets the scale factor to any desired magnification for drawing.
      write(6,1)
1     format(' please enter desired magnification factor'
     *       ,' for drawing.')
      read(5,2)sf
2     format(f5.2)
      write(6,3)
3     format(' thank you.')
      return
      end
c
c
      subroutine sp(n,nn,a,b,c,d,ifit,u,v,uu,vv,w,ww)
c  sets transformation parameters to any desired values.
      dimension u(n),v(n),uu(n),vv(n),w(nn),ww(nn)
      ifit=4
      write(6,1)
1     format(' desired transformation parameters...'/
     *       ' enter  a, b, rho, and  theta (degrees), one per line')
      read(5,2)a,b,rho,theta
2     format(f5.2)
      if(rho.eq.0.) rho=1.
      theta=theta*3.141593/180.
      c=rho*cos(theta)
      d=rho*sin(theta)
      call coord(n,nn,a,b,c,d,u,v,uu,vv,w,ww)
      write(6,3)
```

```
3     format(' thank you.')
      return
      end
c
c
      subroutine sm(n,nn,a,b,c,d,ifit,u,v,uu,vv,w,ww)
c  modify current values of transformation parameters.
      dimension u(n),v(n),uu(n),vv(n),w(nn),ww(nn)
      ifit=4
      write(6,1)
1     format(' modification of current transformation parameters...'
     *       /' enter  a, b, rho, and  theta (degrees), one per line.')
      read(5,2)at,bt,rho,theta
2     format(f5.2)
      if(rho.eq.0.) rho=1.
      theta=theta*3.141593/180.
      ct=rho*cos(theta)
      dt=rho*sin(theta)
      att=at+a*ct-b*dt
      b=bt+a*dt+b*ct
      a=att
      ctt=c*ct-d*dt
      d=c*dt+d*ct
      c=ctt
      call coord(n,nn,a,b,c,d,u,v,uu,vv,w,ww)
      write(6,3)
3     format(' thank you.')
      return
      end
c
c
      subroutine desc(ifit)
c  prints the current type of fit.
      go to (10,20,30,40)ifit
10    write(6,1)
1     format(' original data')
      return
20    write(6,2)
2     format(' least squares fit')
      return
```

```
30    write(6,3)
3     format(' robust fit by repeated medians')
      return
40    write(6,4)
4     format(' special fit')
      return
      end
c
c
      subroutine scale(n,s,sf,xm,ym,x,y)
c  finds center and scale for drawing so that shape 1 fits in square
c  from (.25,.25) to (.75,.75), then adjusts by the value of  sf.
      dimension x(n),y(n)
      xmin=x(1)
      xmax=x(1)
      ymin=y(1)
      ymax=y(1)
      do 10 i=1,n
        xmin=amin1(xmin,x(i))
        xmax=amax1(xmax,x(i))
        ymin=amin1(ymin,y(i))
10      ymax=amax1(ymax,y(i))
      s=2.*amax1(xmax-xmin,ymax-ymin)/sf
      xm=(xmin+xmax)/2.
      ym=(ymin+ymax)/2.
      return
      end
c
c
      function fmed(n,a)
c  finds the median of a(1),...,a(n).
c  a more efficient algorithm could be used
c  if  n  is large to decrease execution time.
      dimension a(n)
      m=n/2+1
      do 20 i=1,m
        k=i
        il=i+1
        do 10 j=il,n
10        if(a(j).lt.a(k)) k=j
```

```
          at=a(i)
          a(i)=a(k)
 20       a(k)=at
        if(n.eq.2*(n/2)) go to 30
        fmed=a(m)
        return
 30     fmed=(a(m)+a(m-1))/2.
        return
        end
```

## REFERENCES

HUBER, P.J. (1980). Comparison of Point Configurations. Technical Report PJH-1, Department of Statistics, Harvard University.

SIEGEL, A.F. (1980). Robust Regression Using Repeated Medians. Technical Report 172, Series 2, Department of Statistics, Princeton University.

SIEGEL, A.F., and BENSON, R.H. (1979). Resistant Fitting as a Basis for Estimating Allometric Change in Animal Morphology. Technical Report 522, Department of Statistics, University of Wisconsin at Madison.

SNEATH, P.H.A. (1967). Trend-Surface Analysis of Distortion Grids. Journal of Zoology, Proceedings of the Zoological Society of London 151, 65-122.

THOMPSON, D'A.W. (1917). On Growth and Form. First edition. Cambridge University Press.

DAT
ILM