Different methods of calculating body sway area

Thomas Wollseifen

i3, Wiesbaden, Germany

Posturography is used to assess the steadiness of the human body by measuring the movement of the centre of pressure of a standing subject on a force platform (stabilometry). This paper presents three different methods of calculating the centre of pressure trajectory. The first method ('Convex hull') is characterized by the area enclosed by the path of movement (body sway area), approximated by the area of a convex hull. PROC G3GRID is applied for the triangulation of the data points necessary for calculating the convex hull. This approach is compared with the second and most common procedure ('principal component analysis, PCA) which calculates an ellipse enclosing the sample points. PROC PRINCOMP is used to calculate the eigenvectors that represent the derived ellipse of the PCA. A third approach used in clinical studies ('Mean of Circle Areas') calculates the body sway area by summarizing the mean of the circle areas defined by the sample points and their distance to the point of origin. Simulation data are used to compare the different analysis methods. The SAS® annotate facility displays the results graphically.

Keywords: Principal component analysis, Body sway area, Posturography, Convex hull

Introduction

Analysis of postural body sway patterns has been used to describe the postural stability in the elderly and in patients with Parkinson's disease, with sleeping problems or with other pathological conditions. Upright posture is inherently unstable. Stabilometry can give hints about the functioning of the human balance control system.

In clinical studies, patients are asked to stand still on a platform (force platform) that is mounted on pressure sensors which transmit data of the trajectory to a computer (Fig. 1). The trajectory represents the amount of body movement during a certain time interval (Fig. 2). In addition to the body sway area (the area which encloses the data points of the trajectory), further parameters, such as trajectory length, velocity, and frequency distributions, are calculated. These parameters can be used to describe the amount of body sway.

In this paper, the following analysis methods that calculate the body sway area of the trajectory are presented and compared:

1. area of convex hull;
2. principal component analysis — body sway area (approximated by an ellipse) is defined by the principal component analysis of the covariant matrix;
3. mean of circle areas defined by the distance (radius) of the sample point to the origin.

Movement of the body sway trajectory will be simulated and the different approaches will be tested with these data. The absolute values of calculated areas of the three methods will be compared.

Methods to Calculate Body Sway Area

The following three methods describe the calculation of the body sway area.

Convex hull

The convex hull of the data points is the approximation of the body sway area (Fig. 3). In this example, the point set of Fig. 2 is used to describe the method. Polygon triangulation is used to calculate the area of the convex hull of the trajectory. The area of the convex hull is the summary of the lower and upper trapezoids.

In this case, the determinant of the (x,y) vector is used to calculate the area (see formula below). PROC G3GRID with OUTTRI option calculates the triangulation data. Outer triangles define the convex hull.

In computational geometry, a number of algorithms are known for computing the convex hull for a finite set of points and for other geometric objects. For example, the gift wrapping algorithm is an algorithm for computing the convex hull of a given set of points. In two dimensions, the gift wrapping algorithm is similar to the process of winding a string (or wrapping paper) around the set of points.

Correspondence to: Thomas Wollseifen, i3, Taunusstrasse 9, Wiesbaden 65183, Germany. Email: Thomas.wollseifen@i3global.com

Published by Maney
DOI 10.1179/175709311X13166801334271

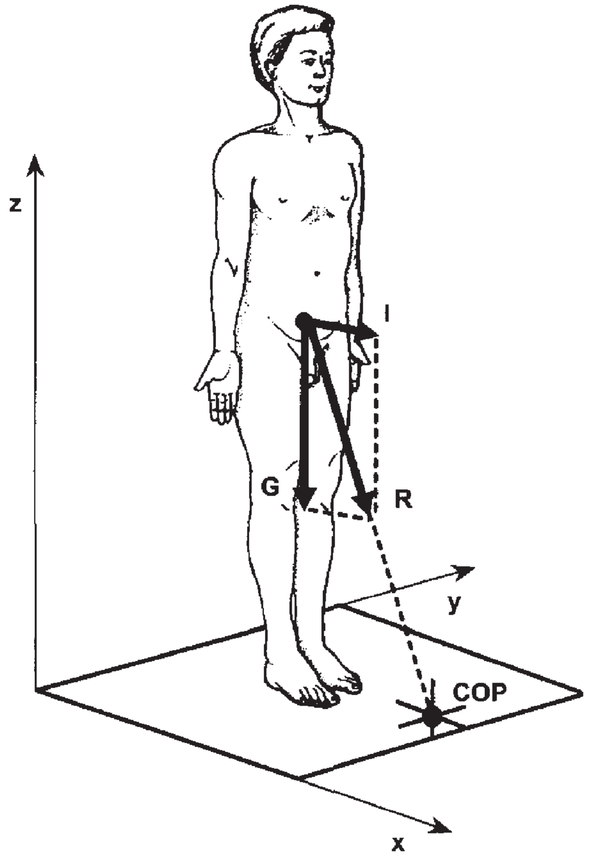

Figure 1 Centre of pressure (COP): subject standing on a force platform[1]

In this case, the triangulation of a set of points is a very simple method to use for the calculation of the convex hull.

The convex hull of a point set *P* could be calculated using the following algorithm:

1. triangulation of the point set *P* (Proc G3GRID);
2. a data point is on the convex hull if the vertex is part of an outer triangle which has only one edge;

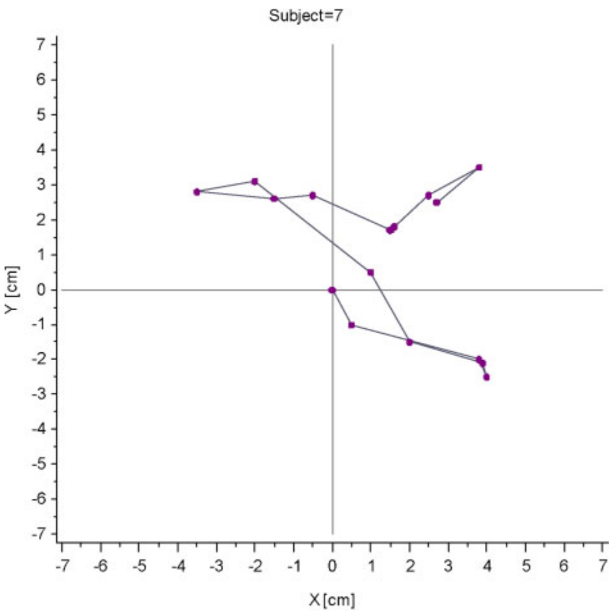

Figure 2 Posturography: sway path with simulation data

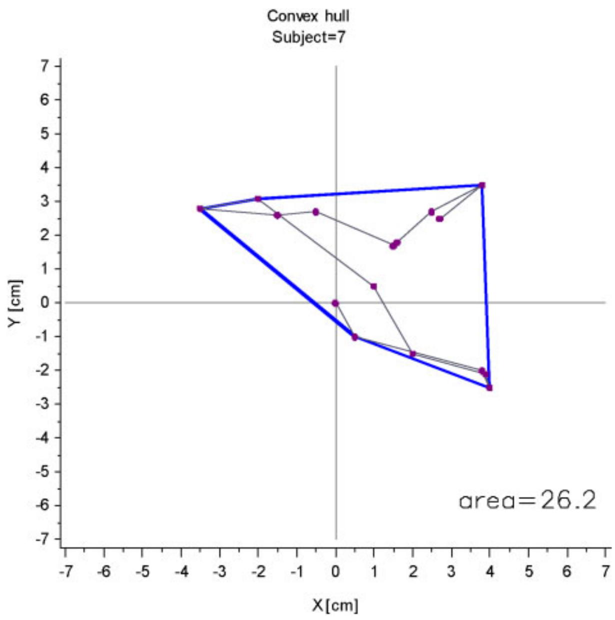

Figure 3 Convex hull

3. calculate the area of the convex hull:
   - 1.1 sort point set $vec_{convex\ hull}$ of the convex hull in counter clockwise order;
   - 1.2 find the center (balance point) as the average of the vertex set;
   - 1.3 choose a vertex arbitrarily and make a reference vector as: ref_vec=vertex[0]−center;
   - 1.4 transform convex hull into polar coordinates;
   - 1.5 sort angles of the coordinates in decreasing order;
   - 1.6 set order of the point set of the convex hull into counterclockwise order;
   - 1.7 calculate determinant of the point set vector; of points sorted in counterclockwise order;
4. the determinant of the vector (sorted in counter-clockwise order) is the area of the convex hull: $\det(vec_{convex\ hull}) = area$.

Finally, to calculate the area of the convex hull, a 'determinant' calculation method or cross-product method is used. The coordinates (x_1,y_1), …, (x_n,y_n) of the convex polygon are arranged in the 'determinant'. The coordinates must be taken in counterclockwise order around the polygon, beginning and ending at the same point. Thus, the area could be calculated with the following formula:

$$\mathrm{Area}_{\mathrm{Convex\ hull}} = \frac{1}{2}\begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \\ x_1 & y_1 \end{vmatrix}$$

$$= \frac{1}{2}[(x_1y_2 + x_2y_3 + x_3y_4 + \cdots + x_ny_1) - (y_1x_2 + y_2x_3 + y_3x_4 + \cdots + y_nx_1)]$$

If the points of the convex hull are sorted in clockwise order, the resulting area is negative; otherwise, it is positive.

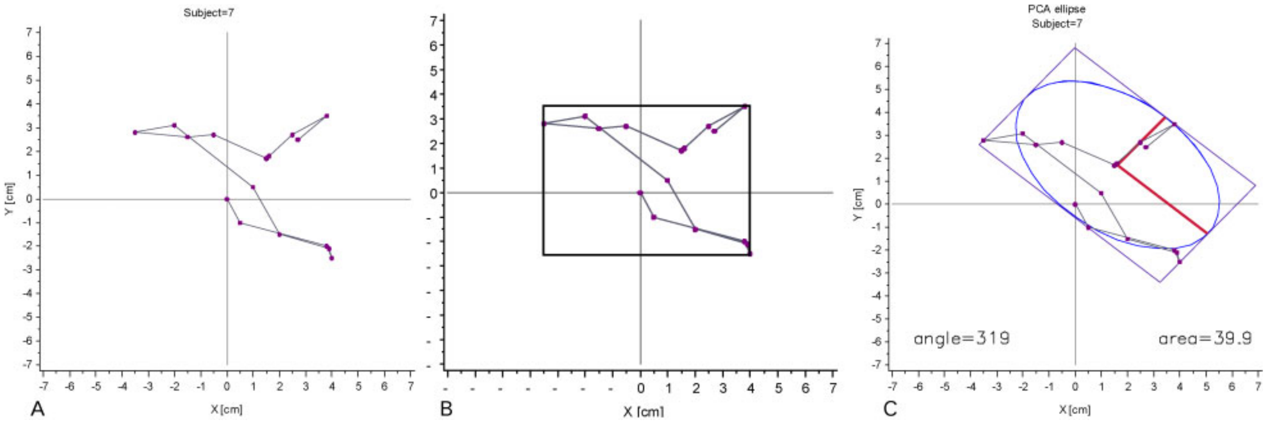

Figure 4 (A) Example body sway path; (B) standard bounding box; (C) ellipse and bounding box calculated with PCA

An example of the SAS macro for calculating the convex hull and its area is given in the Appendix (Example 1).

The calculation of the determinant is presented in Example 2 and the transformation from Cartesian coordinates to polar coordinates is shown in Example 3.

Principal component analysis (PCA)

An ellipse calculated by the PCA describes the approximation of the trajectory of the second approach. A common procedure to determine the area of the body sway trajectory is confined by the PCA of the covariant matrix. The PCA is a mathematical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called principal components.

The eigenvalues (σ_0^2) are calculated from the covariant matrix (σ_{xy}^2) with

$$\sigma_{xy}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})(y_i - \overline{y})$$

where $\overline{x}$ and $\overline{y}$ are the mean values and the summation is carried out over all measured data points. The body sway area can be calculated by the area of the ellipse with the two principal axes $1.96\sigma_0$ at the angle θ with $\tan\theta = \sigma_{xy}^2 / \left(\sigma_0^2 - \sigma_{yy}^2\right)$. The eigenvectors e_1 and e_2 of the component analysis build the main axes of the ellipse.

In Fig. 4C, the resulting ellipse derived by PRINCOMP procedure is displayed which is an approximation of the body sway area. The eigenvectors and the bounding rectangle of the data points are also displayed.

Suppose that one is given a set of points P of Fig. 4A, $p_1, p_2, \ldots, p_n$. It is often important to enclose the points with a bounding box. Usually, the bounding box is calculated by taking the minimum and maximum x–y coordinates of the data points. But this box does not hug the points very closely (Fig. 4B). If we could find a box in a more general position, it could give a much tighter fit to the points; such a box is given in Fig. 4C.

The main problem is finding the correct orientation of the bounding box and the assigned ellipse with axes e_1 and e_2. This can be solved using PCA. First, find the centroid of the point set and translate the whole point set such that the centroid is moved to the origin. Transform the point set into polar coordinates and rotate each point by angle θ of the first eigenvector. Then the bounding box could be easily calculated by the minimum and maximum x–y coordinates of the transformed point set. This allows the bounding box to be re-transformed with the angle θ to the original coordinates. The axes of the rotated bounding box are the eigenvectors e_1 and e_2 of the point set. The corresponding ellipse is also built by the eigenvectors e_1 and e_2 and has the same orientation as the bounding box. The ellipse fits completely into the bounding box. In Fig. 4C, the resulting bounding box rotated by the eigenvectors is displayed.

The following algorithm is applied to calculate the bounding box and the ellipse of the point set:

1. calculate the eigenvalues and eigenvectors e_1 and e_2 of the point set P (PROC PRINCOMP);
2. calculate the centroid (balance point) of the point set P;
3. translate the point set P to a set P' such that the centroid is moved to the origin;
4. transform point set P' into polar coordinates (with angle and radius r);
5. rotate the point set by angle θ of the first eigenvector;
6. re-transform the point set into Cartesian coordinates (x,y);
7. calculate the bounding box and the length of the eigenvectors;
8. calculate the area of the ellipse ($A = \pi|e'_1||e'_2|$) with the corrected length of the eigenvectors (e'_1 and e'_2) of the length of the bounding box.

A similar method is presented to quantify direction and magnitude of body sway using ellipses is presented in Ref. 2. Parameters of the calculated

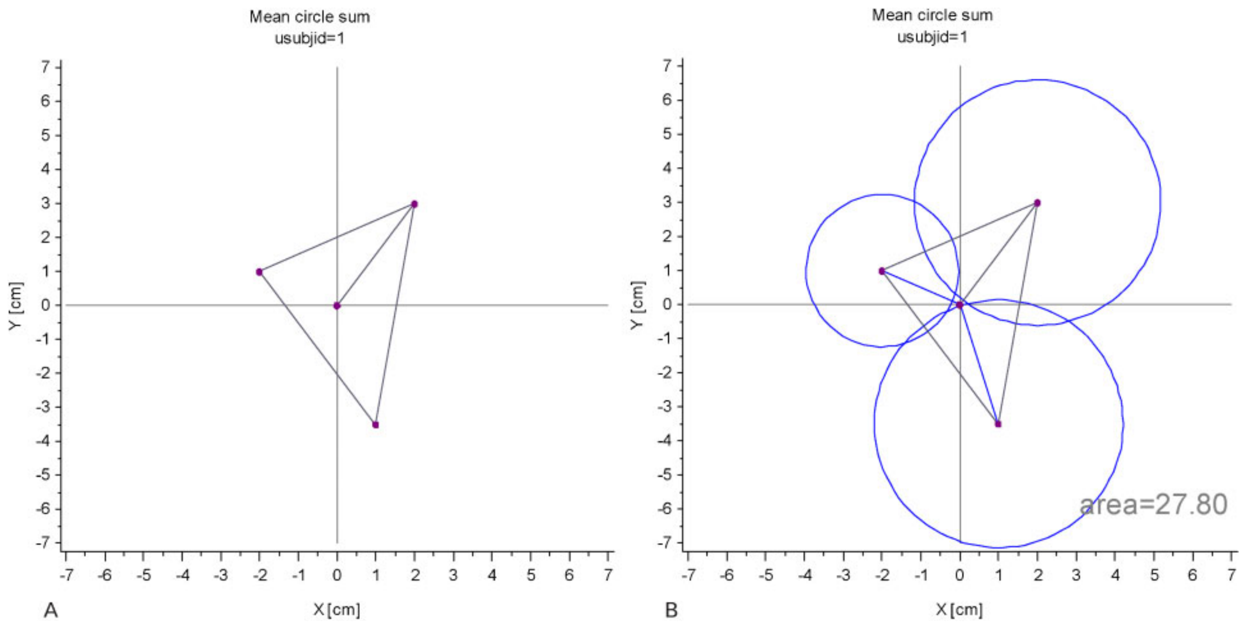

Figure 5 (A) Sway path with simple triangle data; (B) calculation of the mean circle area

ellipse (angle θ, and major and minor axes e_1 and e_2 of the ellipse) represent the direction and the magnitude of the body sway. In Ref. 3, the calculation method using PCA is compared with determination of the area outline described by Fourier coefficients.

The SAS macro code of the PCA method is presented in Appendix (Example 4).

Mean circle area

The third method that calculates the body sway area is an approximation of the trajectory area based on the summary of circle areas of each data point (Fig. 5). Each data point is expressed in polar coordinates (r,θ). For each data point, a circle area that is defined by the distance r to the origin and the data point coordinates (x,y) will be calculated. If a data point lies on the same angle θ with respect to the origin, only the maximum r will be taken into account (and the corresponding point) and the resulting circle area will be summarized. The resulting mean of the circle areas approximates the body sway area.

The following algorithm is applied:

1. transform data points P (x,y) into polar coordinates (r,θ);
2. calculate integer INT(θ) of each angle for each data point;
3. calculate the maximum radius r for each angle θ and save the polar coordinates of this data point (360 different angles possible);
4. for each remaining data point p' (r,θ), calculate the circle areas;
5. body sway area is the mean circle area of the remaining circles (with circle area $A=\pi r_2$).

Fig. 5 illustrates the calculation of the circle area. In this example (Fig. 5A), a simple triangle is used as the sway path. Four circles (three vertices + origin) for each data point are displayed in Fig. 5B. The resulting area is the mean of these circles. In this case, the origin which is the starting point of the sway path contributes also to the mean circle area with an area of zero.

In Fig. 6C, the mean circle area is presented using point set P from Fig. 2. The mean area is displayed as circle with center at the balance point of the data points.

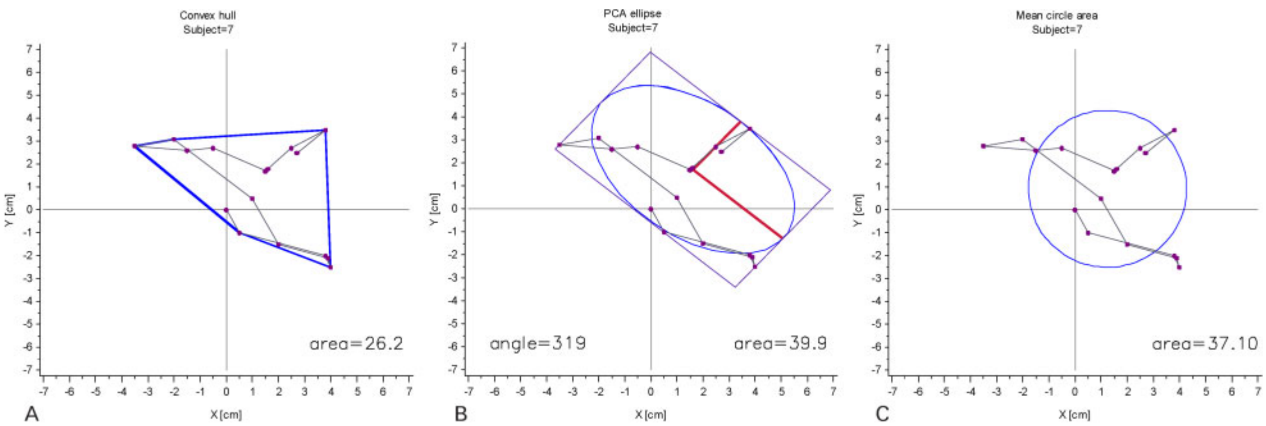

Figure 6 (A) Convex hull (area=26.2); (B) PCA method (area=39.9); (C) circle method (area=37.1)

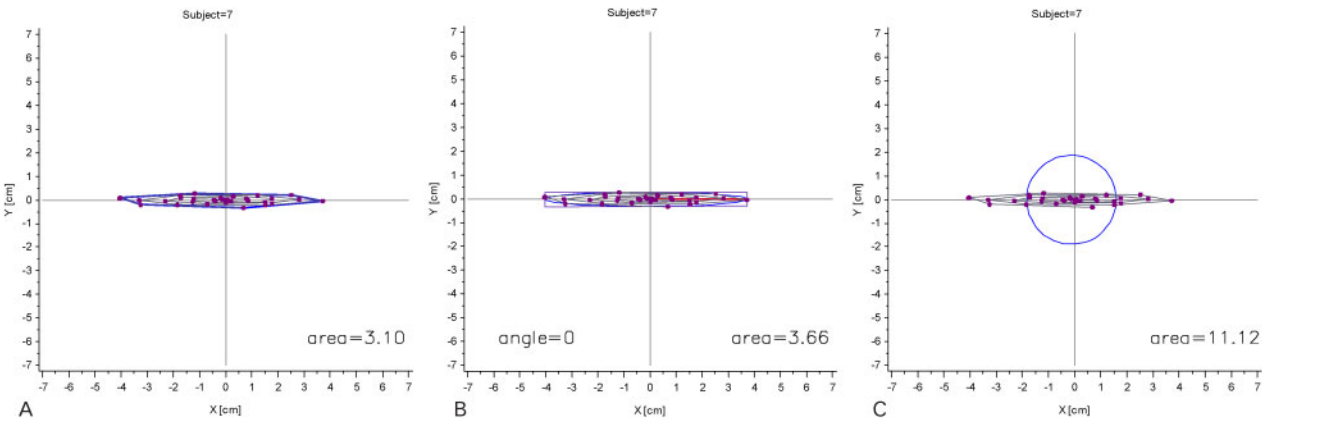

Figure 7 (A) Convex hull (area=3.10); (B) PCA method (area=3.66); (C) circle method (area=11.12)

In Example 5, the SAS macro code of the calculation of the body sway area by the mean circle method is displayed.

The main program for generating example data which calls the different methods is also presented in the appendix (Examples 6–9).

Comparison of the Methods

Overall, using example data point of Fig. 2, the area surrounded by convex hull is 26.2, ellipse area is 39.9, and area calculated by mean circle method is 37.1 (Fig. 6).

Different random datasets with normally and not normally distributed point sets with 10 subjects (~30 data points per subject) were used to compare the different techniques of calculating body sway area. The methods were compared using pairwise comparisons (PROC GLM) of the means of the calculated areas with Tukey's method. The data were also tested for normality. In Table 1, the test results of a sample run are displayed. For each test, body sway data of 10 subjects were simulated following a cyclic movement. For the test01–test03, the data are very 'flat' which means that there is only one major direction of the body sway (Fig 7 shows one subject of these tests). For test04, the data are normally distributed and has no major body sway direction (see, for example, Fig. 8).

The area of the convex hull is always smaller than the area calculated by PCA. The area calculated by the circle method is dependent on the distribution of the data on the plane. The convex hull and the PCA ellipse are comparable, whereas the mean circle method is different when the data have only one major axis.

In addition to the calculation of the area, the PCA method also provides the major sway direction (angle) and the magnitude of the body sway (lengths of axes e_1 and e_1).

One drawback of the mean circle method is that if the body sway has only one major direction, the resulting mean circle area is very large compared to the calculated area by convex hull or PCA method. This effect is shown in Fig. 7. For the circle method, each data point has an influence on the mean circle area. Coordinates close to the balance point have the same influence on the area as coordinates far away from the balance point. If several data points lie on the same angle compared to the balance point, then the data point with the largest distance is taken into account and influences the result.

In contrast, the mean circle area is smaller than the area calculated by convex hull or PCA method when the data are evenly distributed in the x and y directions (Fig. 8).

Conclusion

All three methods proposed are suitable for data interpretation of posturography. The calculated areas

Table 1 Mean calculated area by test data and method

Test data	Calculation method	*N*	Mean	SD	Min.	Max.	Pairwise comparison	*P* value *F*-test	Normal distribution?
Test01	1 Convex hull	10	1.78	1.22	0.1	3.3	1–2: 0.9789		
	2 PCA ellipse	10	2.08	1.44	0.1	3.9	1–3: <0.0001	<0.0001	No
	3 Mean circle	10	13.88	5.65	11.1	29.6	2–3: <0.0001		
Test02	1 Convex hull	10	3.58	2.45	0.3	6.7	1–2: 0.8785		
	2 PCA ellipse	10	4.16	2.84	0.4	7.8	1–3: <0.0001	<0.0001	Yes
	3 Mean circle	10	11.85	2.66	10.1	18.7	2–3: <0.0001		
Test03	1 Convex hull	10	5.36	3.68	0.4	10.0	1–2: 0.8329		
	2 PCA ellipse	10	6.23	4.25	0.5	11.6	1–3: 0.0014	0.0010	No
	3 Mean circle	10	11.30	1.54	10.0	14.5	2–3: 0.0062		
Test04	1 Convex hull	10	8.93	6.16	0.6	16.7	1–2: 0.8367		
	2 PCA ellipse	10	10.33	7.12	0.7	19.3	1–3: 0.6492	0.6683	Yes
	3 Mean circle	10	11.11	0.95	10.1	12.9	2–3: 0.9444		

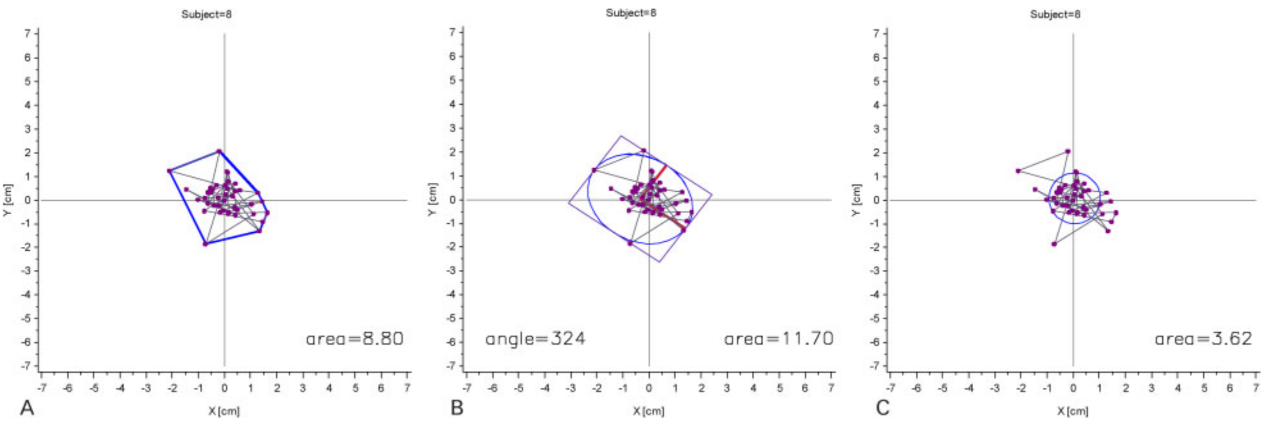

Figure 8 (A) Convex hull (area=8.80); (B) PCA method (area=11.70); (C) circle method (area=3.62)

differ. The calculation of the convex hull is easy to follow and the resulting area describes the outline of the trajectory very well. The PCA method is an approximation of the outline and uses an ellipsis to characterize the trajectory. The calculated area of the PCA method is comparable to the convex hull method. Moreover, the PCA method quantifies the direction and the magnitude of the body sway. The third method is very fast because only circle areas are calculated and summarized, but the resulting area is a crude approximation and depends on the input data.

The area calculated by the convex hull is usually smaller than the area calculated by the PCA method ($Area_{Convex\ hull} < Area_{PCAellipse}$). The size of the area calculated by the circle method is dependent on the data.

I would propose to use the PCA method because the area calculated is comparable to the area surrounded by the convex hull. Another benefit of the PCA method is that both magnitude and direction of sway are computed simultaneously.

Acknowledgements

I would like to thank Sandra Vermeulen (Associate Project Director, i3), Pam Giese (Principal Statistical Programmer, i3), Nancy Brucken (Principal Statistical Programmer, i3), and Hans-Dieter Spies (Senior Statistical Programmer, i3) for their inputs and their thoughtful review of this manuscript.

References

1 Witte H, Recknagel SJ. Ist die indirekte Posturographie mittels Kraftmeßplatten der direkten Posturographie durch Bewegungsanalyse. Biomed Tech 1997;**42**:280–3.
2 Sparto PJ, Redfern MS. Quantification of direction and magnitude of cyclical postural sway using ellipses. Biomed Eng – Appl Basis Commun 2001;**13**(5):213–7.
3 Rugelj D, Sevsek F. Postural sway area of elderly subjects. WSEAS Trans Signal Process 2007;**3**(2):213–9.

Appendix

```
/******************************************
* Program name    : m-convex_hull.sas
*
* Author                : Thomas
                          Wollseifen
* Date created          : 29.07.2011
* Purpose               : Convex hull
                          of a point set
                          (with x,y
                          coordinates)
* Template              :
* Inputs                :
* Outputs               :
* Program completed     : Yes/No
* Updated by            : (Name) - (Date):
*                            (Modification
                             and Reason)
******************************************/

%macro convex_hull(indata      =
                  ,x   =
                  ,y   =
                  ,z   = /*usually usubjid*/
                  ,outdata     =
                  ,outanno     =
);

%local idx i usubjid;

******************************************;
* create macro var list with usubjid;
******************************************;
proc sql noprint;
     select distinct &z. into :usubjid
     separated by '#'
     from &indata.
     order by &z.
     ;
quit;
******************************************;
* loop for each usubjid;
******************************************;
%let maxn=&sqlobs.;
%let idx=1;
%do %while (&idx.<=&maxn.);
     %let i=%scan(&usubjid.,&idx.,#);
     data _ch00;
          set &indata.(where=(&z.=
          &i.));
     run;
     proc sort data=_ch00;
          by &z.;
     run;
```

```
*************************************;
** 1 calculate triangulization of the
   point set;
*************************************;
proc g3grid data=_ch00 outtri=
_ch01 out=_temp;
     grid &y.*&x.=&z.;
run;
quit;
***check if dataset is empty;
proc sql noprint;
     select count(*)
     into :OBSCOUNT
     from _ch01
     ;
quit;
%let obscount=%sysfunc(compress
(&obscount.));
***if data set is empty then create
artificial data set with 1 triangle
(without vertices/edges);
%if &obscount=0 %then %do;
    data _ch01;
        xs=0; ys=0; triangle=1;
        output;
        xs=.; ys=.; triangle=1;
        output;
        xs=.; ys=.; triangle=1;
        output;
   run;
%end;
**************************************;
** 2 a point (x,y) is on the convex hull
if the vertex includes 2
** missings;
**************************************;
proc sql;
     create table _ch02 as
     select *
            ,nmiss(xs) as xn
            ,&i. as usubjid

     from _ch01
     group by triangle
     having xn=2 and xs ne.
     ;
quit;
**************************************;
** 3 sort data points in counter
clockwise order;
**************************************;
** 3.1 Find the center as the average
of the vertex set.;
**************************************
proc sql;
     create table _ch03 as
     select *
            ,mean(xs) as cx
            ,mean(ys) as cy
     from _ch02
     ;
quit;
**************************************;
** 3.2. Choose a vertex arbitrarily
and make a reference vector as:;
** ref_vec = vertex[ 0] - center;
**************************************;
data _ch04;
     set _ch03;
     xs_=xs-cx;
     ys_=ys-cy;
run;
**************************************;
** 3.3. Transform convex hull into
polar coordinates;
**************************************;
%polar_coordinates( indata    = _ch04
                   ,x        = xs_
                   ,y        = ys_
                   ,outdata  = _ch04
);
**************************************;
** 3.4. Sort angles of the coordinates
in decreasing order;
**************************************;
proc sort data=_ch04;
     by descending theta_;
run;
**************************************;
** 3.5 set order of the point set of the
convex hull
** into counter clockwise order;
**************************************;
data _ch04;
     set _ch04;
     order=_n_;
run;
data _ch04;
     set _ch04
         _ch04 (where=(order=1));
run;
**************************************;
** 4 calculate determinant of the
point set vector;
** of points sorted in counter
clockwise order;
** det(vec):=area;
**************************************;
%det(indata=_ch04, outdata=_det);
*calculate area of a convex polygon;
**************************************;
** draw convex polygon;
**************************************;
data _ch05;
     set _ch04;
    area=&det.;
run;
data _ch06;
     length function $20 color $10;
     set _ch05;
     %system (2,2,2);
     if _n_=1 then do;
          %move(xs,ys);
     end;
     else do;
          %draw(xs, ys, blue, 1, 1.0);
     end;
     %label(&areax.,&areay.,"area=
&det.",black,0, 0,&labelsize., Simplex,
2);
run;
data _ch06&idx.;
```

```
      set _ch06;
  run;
  data _ch06_&idx.;
    set _ch05;
  run;
  %let idx=%eval(&idx.+1);
%end; *end while;
  *************************************;
* concatenate data*;
  *************************************;
data &outanno.;
  length method $20;
  set %do idx=1 %to &maxn.;
        _ch06&idx.
      %end;;
  method=' Convex hull' ;
  methodn=1;
run;
data &outdata.;
  length method $20
            indata $30;
 set %do idx=1 %to &maxn.;
         _ch06_&idx.
    %end;;
  methodn=1;
  indata="&indata.";
run;
data area_&outdata.;
     set &outdata.(keep=usubjid area
     method methodn indata);
     by usubjid;
     if first.usubjid;
run;
%mend convex_hull;
```

Example 1: SAS macro for calculation of convex hull.

```
/****************************************
* Program name          : m-calc_
                          determinant.sas
*
* Author                : Thomas
                          Wollseifen
* Date created          : 29.07.2011
* Purpose               : Calculate
                          determinant of
                          a vector
                         (x1,y1)
*                                (...)
*                                (xn,yn)
* Template              :
* Inputs                :
* Outputs               :
* Program completed     : Yes/No
* Updated by            : (Name) - (Date):
*              (Modification and Reason)
****************************************/
%macro det( indata =
         ,outdata =
         ,x = xs
         ,y = ys
);
%global det;
data _det01;
     set &indata.;
     x2=lag(&x.);
     y2=lag(&y.);
     if not missing(x2) then d1=x2*&y.;
     else d1=0;
     if not missing(y2) then d2=y2*&x.;
     else d2=0;
run;
proc sql;
create table _det02 as
     select *
          ,sum(d1) as sumd1
          ,sum(d2) as sumd2
          ,round((calculated sumd2 -
calculated sumd1)*0.5, 0.1) as det
          from _det01
          ;
quit;
data _null_;
     set _det02;
     call symput(' det' , trim(left(put
(det, 8.2)))); *return det. as macro
variable;
run;
%mend det;
```

Example 2: SAS macro for calculation of determinant.

```
****************************************
* Program name          : m-polar_
                          coordinates.sas
* Author                : Thomas
                          Wollseifen
* Date created          : 29.07.2011
* Purpose               : Macro to
                          calculate
                          polar
                          coordinates
      Returns data set with radius r,
and angle theta
* Template              :
* Inputs                :
* Outputs               :
* Program completed     : Yes/No
* Updated by            : (Name) - (Date):
*              (Modification and Reason)
****************************************/
%macro polar_coordinates( indata  = /
*in data set*/
      ,x  = /*x coordinate variable*/
      ,y  = /*y coordinate variable*/
      ,outdata  = /*output data set*/
      ,theta  = theta /*angle theta of
the x,y coordinates in -pi
to pi*/
      ,theta  _deg=theta_ /*angle
theta in deg of the x,y
   coordinates (0-360°)*/
      ,r  = r /*radius r*/
);
data &outdata.;
     set &indata.;
     r=sqrt((&x**2)+(&y**2));
     if &x.> 0 then &theta.=(atan(&y./
&x.));
     else if &x.<0 and &y.>=0 then
&theta.=(atan(&y./&x.)+&pi.);
     else if &x.<0 and &y.<0 then
&theta.=(atan(&y./&x.)-&pi.);
```

```
    else if &x.=0 and &y.>0 then
&theta.=&pi./2;
  else if &x.=0 and &y.<0 then
&theta.=-&pi./2;
  else if &x.=0 and &y.=0 then
&theta.=0;
  if &theta.<0 then &theta.=&theta.+
2*&pi.;
  else &theta.=&theta.;
  &theta_deg.=round((180/&pi.)
*&theta.,0.001);
  pi=&pi.;
run;
%mend polar_coordinates;
```

Example 3: SAS macro for transformation of Cartesian coordinates to polar coordinates.

```
/****************************************
* Program name        : m-pca_
                        ellipse.sas
*
* Author              : Thomas
                        Wollseifen
* Date created        : 29.07.2011
* Purpose             : Calculate area
                        of point set
                        with PCA method
* Template            :
* Inputs              :
* Outputs             :
* Program completed   : Yes/No
* Updated by          : (Name) - (Date):
*              (Modification and Reason)
****************************************/
%macro pca_ellipse( indata    =
          ,outdata  =
          ,x    = xs
          ,y    = ys
);
%local idx i usubjid maxn;
ods graphics on;
data _pca00;
     set &indata.;
run;
proc sql noprint;
     select distinct usubjid into :
     usubjid separated by '#'
     from _pca00
     order by usubjid
     ;
quit;
%let maxn=&sqlobs.;
%put &usubjid.;
%let idx=1;
%do %while (&idx.<=&maxn.); *loop for
every usubjid;
     %let i=%scan(&usubjid.,&idx.,
     #);
     %put ****** usubjid=&i.;

     data _pca01;
          set _pca00(where=(usubjid=
          &i.));
  run;
     proc sql;
          create table _pca01 as select
               *
               ,mean(xs) as mxs
               ,mean(ys) as mys
          from _pca01
          ;
     quit;
     data _null_;
          set _pca01;
          call symput('mxs', mxs);
          call symput('mys', mys);
     run;
     ***********************************
**;
     ** 1 Principal component analysis;
     ** Calculate eigenvectors and
eigenvalues of the point set;
     ***********************************
**;
     proc princomp data=_pca01 COV OUT
     = _prins ;
          var &y. &x.;
          ods output eigenvectors
=_eigenv eigenvalues=_eigenvalues;
     run;
     ***********************************
**;
     *** calculate ellipse;
     ***********************************
**;
     proc transpose data=_eigenv
     out=_eigenvt;
          var prin1 prin2;
          id variable;
          idlabel variable;
     run;
     data anno_eigenv;
          set _eigenvt;
          usubjid=&i.;
          %system(2,2,2);
          %line(0,0,xs,ys,green,1,0.1);
     run;
     %polar_coordinates( indata  =
_eigenvt
               ,x    = xs
               ,y    = ys
               ,outdata = _eigenvt
     );
     data _eigenvt01;
          set _eigenvt(where=(_name_
='Prin1')); *select first eigenvector*;
          usubjid=&i.;
     run;
     ***********************************
**;
     ***merge data with eigenvector
data;
     ***********************************
**;
     data _eigenvt02;
          merge      _eigenvt01
                     _pca01;
          by usubjid;
          theta_e_=theta_; *angle of
the eigenvector;
          theta_e=theta;
```

```
          r_e=r;   *radius (=1) of
eigenvector;
     run;
     ***********************************;
     ** 2 calculate mean (x), mean(y)
coordinates (centroid);
     ***********************************;
     proc sql;
          create table _eigenvt03 as
select
                *
                ,mean(xs) as meanx
                ,mean(ys) as meany
          from _eigenvt02
          ;
     quit;
     ***********************************;
     ** 3 translation of the point set by
centroid;
     ***********************************;
     data _eigenvt04;
          set _eigenvt03;
          xsm=xs-meanx;
          ysm=ys-meany;
     run;
     ***********************************;
     ** 4 Transform point set into polar
coordinates;
     ***********************************;
     %polar_coordinates( indata=
_eigenvt04
                  ,x    = xsm
                  ,y    = ysm
                  ,outdata = _eigenvt04
     );
     ***********************************;
     ** 5+6 rotation of the data points
by angle theta_e;
     ** of the first eigenvector e1;
     ** and re-transformation to x,y
Cartesian coordinates;
     ***********************************;
     data _eigenvt05;
          set _eigenvt04;
          xsm_=r*cos(theta-theta_e);
          ysm_=r*sin(theta-theta_e);
     run;
     ***********************************;
     ** rotation of the original data
points by angle theta;
     ***********************************;
     %polar_coordinates(indata  =
_eigenvt04
                      ,x    = xs
                       ,y   = ys
                       ,outdata=
_eigenvt04_
     );
     data _eigenvt05_;
          set _eigenvt04_;
          xs=r*cos(theta-theta_e);
          ys=r*sin(theta-theta_e);
     run;
     ***********************************;
     ** calculate the min and max coor-
dinates of the bounding box
     ***********************************;
     proc sql;
               create table _eigenvt06_
as select
                    *
                   ,max(xs) as maxxs
                   ,min(xs) as minxs
                   ,max(ys) as maxys
                   ,min(ys) as minys
               from _eigenvt05_
               ;
     quit;
     ***********************************;
     ** calculate the edges of the
bounding box;
     ***********************************;
     data _eigenvt07_;
          set _eigenvt06_;
          lenx=round((maxxs-minxs)/2,
0.01); *x length of rectangle;
          leny=round((maxys-minys)/2,
0.01); *y length of rectangle;
          centerx=round(minxs+lenx,
0.01); *center (x) of the rectangle;
          centery=round(minys+leny,
0.01); *center (y) of the rectangle;
          call symput('centerx',
trim(left(put(centerx, 8.2))));
          call symput('centery',
trim(left(put(centery, 8.2))));
          call symput('lenx', trim
(left(put(lenx, 8.2))));
          call symput('leny', trim
(left(put(leny, 8.2))));
     run;
     ***********************************;
     ** retransformation of the center
of the rectangle;
     ** and calculate area of the
ellipse;
     ***********************************;
     %polar_coordinates( indata  =
_eigenvt07_
                   ,x    = centerx
                   ,y    = centery
                   ,outdata  =
_eigenvt07_
     );
     ***********************************;
     ** 7 Calculate area of the ellipse;
     ***********************************;
     data _eigenvt08;
          set _eigenvt07_;
          centerx_=r*cos(theta+
theta_e);
          centery_=r*sin(theta+
theta_e);
          call symput('centerx_',
trim(left(put(centerx_, 8.2))));
          call symput('centery_',
trim(left(put(centery_, 8.2))));
          area=round(lenx*leny*pi,
0.01);***area of the ellipse;
     run;
     ***********************************;
     ** calculate resulting ellipse;
```

```
    ***********************************;
    %calc_ellipse(indata=_eigenvt08
(obs=1)
        ,x0=centerx_ /*meanx*/
        ,y0=centery_ /*meany*/
        ,alen=lenx
        ,blen=leny
        ,angle=theta_e
        ,outdata=_pca
        );
    data _pca_&idx._;
   length method $20;
   set _pca;
   method=' PCA ellipse';
   methodn=3;
   call symput(' area',trim(left(put
(area,8.2))));
   call symput(' theta_e_',trim(left
(put(theta_e_,8.))));
    run;
    data _pca_&idx.;
   length function $20 color $10 style
$20 text $15;
   set _pca;
   by usubjid;
   %system(2,2,2);
   if first.usubjid then do;
   %move(x,y);
   end;
   else do;
    %draw(x, y, blue, 1, 0.8); *draw
ellipse;
   end;
   %label(&areax.,&areay.,"ar-
ea=&area.",black,0, 0,&label-
size.,        Simplex, 2);
      %label(&anglex.,&angley.,"angle
=&theta_e_.°",black,0,
0,&labelsize., Simplex, 2);
    run;
    proc transpose data=_eigenv
out=_eigenv_t;
     var prin1 prin2;
     id variable;
     idlabel variable;
    run;
    %polar_coordinates( indata  =
_eigenv_t
        ,x    = xs
        ,y    = ys
        ,outdata  = _eigenv_t
    );
***************************************;
    ** calculate x-y coordinates of the
eigenvectors and;
    ** calculate the x-y coordinates of
the bounding rectangle;
***************************************;
    data _eigenv_t;
        set _eigenv_t;
        usubjid=&i.;
        *calculate x-y coord of the
resulting eigenvectors with
adapted      length;
        if _name_=' Prin1' then do;
             eigenvx=&lenx.*cos
(theta)+&centerx_.;
             eigenvy=&lenx.*sin
(theta)+&centery_.;
         end;
         if _name_=' Prin2' then do;
             eigenvx=&leny.*cos
(theta)+&centerx_.;
             eigenvy=&leny.*sin
(theta)+&centery_.;
         end;
         *coordinates of bounding box;
         ax=&centerx.-&lenx.;
         ay=&centery.+&leny.;
         bx=&centerx.+&lenx.;
         by=&centery.+&leny.;
         cx=&centerx.+&lenx.;
         cy=&centery.-&leny.;
         dx=&centerx.-&lenx.;
         dy=&centery.-&leny.;

     run;
   ************************************;
     ** rotate x-y coordinates of the
bounding rectangle;
     ** according to the angle of the 1st
eigenvector;
   ************************************;
     data _rect01;
         set _eigenv_t(obs=1);
         theta_e=theta;
     run;
     %polar_coordinates( indata  =
_rect01,x= ax,y=ay,outdata=
_rect01);
     data _rect01;
         set _rect01;
         ax=r*cos(theta+theta_e);
         ay=r*sin(theta+theta_e);
     run;
     %polar_coordinates( indata  =
_rect01,x= bx,y=by,outdata=
_rect01);
     data _rect01;
         set _rect01;
         bx=r*cos(theta+theta_e);
         by=r*sin(theta+theta_e);
     run;
     %polar_coordinates( indata  =
_rect01,x= cx,y=cy,outdata=
_rect01);
     data _rect01;
         set _rect01;
         cx=r*cos(theta+theta_e);
         cy=r*sin(theta+theta_e);
     run;
     %polar_coordinates( indata=
_rect01,x= dx,y=dy,outdata=
_rect01);
     data _rect01;
         set _rect01;
         dx=r*cos(theta+theta_e);
         dy=r*sin(theta+theta_e);
     run;
   ************************************;
    ** draw eigenvectors;
```

```
   ***********************************;
    data anno05_eigv;
         set _eigenv_t;
         usubjid=&i.;
         %system(2,2,2);
         %move(&centerx_.,&centery_.);
         %line(&centerx_. ,&centery_.,
eigenvx,eigenvy, vipk,1,1.0);
    run;
   ***********************************;
   **draw bounding rectangle;
   ***********************************;
    data anno06_rect;
      set _rect01(obs=1);
      %system(2,2,2);
      %move(ax,ay);
      %line(ax,ay,bx,by,bipb,1,0.8);
      %line(bx,by,cx,cy,bipb,1,0.8);
      %line(cx,cy,dx,dy,bipb,1,0.8);
      %line(dx,dy,ax,ay,bipb,1,0.8);
    run;
    data _pca_&idx.;
         set _pca_&idx.
         anno05_eigv
         anno06_rect;
    run;
    %let idx=%eval(&idx.+1);
%end; *end while;
*************************************;
***prepare annotation data set for
gplot;
*************************************;
data anno_pca;
    set %do idx=1 %to &maxn.;
         _pca_&idx.
        %end;;
run;
data &outdata.;
    length indata $30;
    set %do idx=1 %to &maxn.;
             _pca_&idx._
        %end;;
    indata="&indata.";
run;
*************************************;
***prepare area outdata set with cal-
culated area;
*************************************;
data area_&outdata.;
    set &outdata.(keep=usubjid area
method methodn indata);
    by usubjid;
    if first.usubjid;
run;
%mend pca_ellipse;
```

Example 4: SAS Macro to calculate the bounding box and the corresponding ellipse and the area of the ellipse with PCA.

```
/*************************************
* Program name        : m-calc_bs_
                        area_circle.sas
*
* Author              : Thomas
                        Wollseifen
* Date created        : 29.07.2011
* Purpose             : Calculation of
                        an area of data
                        points with mean
                        circle method
* Template            :
* Inputs              :
* Outputs             :
* Program completed   : Yes/No
* Updated by          : (Name) - (Date):
*            (Modification and Reason)
**************************************/

%macro calc_bs_area_circlemethod
( indata  =
             ,x       = xs
             ,y       = ys
             ,outdata  = );

    data _cm01;
         set &indata.;
    run;
    **********************************
**;
    ** 1 Convert x,y Cartesian coordi-
nates into polar coordinates;
    **********************************
**;
  %polar_coordinates( indata  = _cm01
             ,x       = xs
             ,y       = ys
             ,outdata  = _cm02);
    **********************************
**;
    ** 2 Calculate the integer of each
angle for each data point;
    **********************************
**;
    data _cm03;
         set _cm02;
         theta_int=int(theta_);
         z=1;
    run;
    **********************************
**;
  ** Calculate the mean of the x,y
coordinates;
    **********************************
**;
    proc sql;
         create table _cm04_ as
         select *
             ,mean(xs) as meanx
             ,mean(ys) as meany
         from _cm03
         group by usubjid
         ;
  quit;
    **********************************
**;
  * 3 Calculate the max radius for each
angle theta;
    **********************************
**;
  proc sql;
   create table _cm04 as
```

```
        select usubjid
              ,max(r) as maxr
              ,xs
              ,ys
              ,r
              ,theta
              ,theta_int
              ,pi
              ,meanx
              ,meany
        from _cm04_
        group by usubjid, theta_int
        having r=max(r)
        order usubjid, theta_int
        ;
     quit;
     ***********************************
**;
     * 4+5 Calculate mean area for each
data point with max radius;
     * Calculate the mean circle area
     ***********************************
**;
     proc sql;
       create table _cm05 as
           select *
                  ,count(usubjid) as n
                  ,r**2*pi as area_i
                  ,sum(calculated
area_i)/calculated n as mean_area
           from _cm04
           group by usubjid
           order usubjid
           ;
     quit;
     ***********************************
**;
     * prepare outdata set;
     ***********************************
**;
     data &outdata.;
          length method $20
            indata $30;
           set _cm05;
           area=round(mean_area,0.01);
           r_meancircle=sqrt(area/
&pi.);
           method='Mean Circle';
           methodn=2;
           indata="&indata.";
     run;
     ***********************************
**;
     * prepare data for graphical out-
put;
     ***********************************
**;
     proc sql noprint;
           select distinct usubjid into
:usubjid separated by '#'
           from &outdata.
           order by usubjid
    ;
     quit;
     %let maxn=&sqlobs.;
     %let idx=1;
     %do %while (&idx.<=&maxn.);
          %let i=%scan(&usubjid.,
&idx.,#);
          data _cmg&idx.;
               set &outdata.(where=
(usubjid=&i.));
               length function $20
color $10 style $20 areac $20;;
               areac="area="||
trim(left(put(area,8.2)));
               %system(2,2,2);
               %circle(meanx, meany,
r_meancircle, blue);
          *display circle with radius
of resulting mean circle area;
               %label(&areax.,
&areay.,areac,black,0, 0,&labelsize.,
               Simplex, 2);
          run;

             %let idx=%eval(&idx.+1);
     %end;

     data &outdata._;
            length method $20;
            set %do idx=1 %to &maxn.;
            _cmg&idx.
            %end;;
     run;
     ***********************************
**;
     * get area of mean circles as macro
var;
     ***********************************
**;
     data _null_;
          set _cm05;
          call symput('area', trim(left
(put(mean_area, 8.2))));
     run;

     data anno_circle1;
          length function $8;
          set _cm05;
          %system(2,2,2);
          %circle(xs, ys, r, blue);

          %line(0,0,xs,ys,blue,1,0.8);
     run;

     data anno_circle2;
          length function $8;
          set _cm05(obs=1);
          %system(2,2,2);
          %label(&areax.,&areay.,"area
=&area.",black,0, 0,&labelsize.,
             Simplex, 2);
     run;

  data anno_circle;
    set anno_circle1
    anno_circle2;
  run;

  data area_&outdata.;
    set &outdata.(keep=usubjid area
method methodn indata);
```

```
    by usubjid;
    if first.usubjid;
  run;

%mend calc_bs_area_circlemethod;
```

Example 5: SAS macro to calculate the area of a point set with mean circle method.

```
/*****************************************
* Program name        : m-calc_
                        ellipse.sas
*
* Author              : Thomas
                        Wollseifen
* Date created        : 29.07.2011
* Purpose             : calculate
                        ellipse data
* Template            :
* Inputs              :
* Outputs             :
* Program completed   : Yes/No
* Updated by          : (Name) - (Date):
*            (Modification and Reason)
*****************************************/
%macro calc_ellipse( indata=
          ,x=x    /*x-coordinates*/
          ,y=y    /*y-coordinates*/
          ,x0=    /*x0 vector, center
of ellipse*/
          ,y0=    /*y0 vector, center
of ellipse*/
          ,alen=  /*a length of vec-
tor*/
          ,blen=  /*b length of vec-
tor*/
          ,angle=      /*angle of the
vector a*/
          ,mint=1      /*min of loop*/
          ,maxt=9      /*max of loop*/
          ,stept=0.2 /*step width*/
          ,outdata=
        );

data _ellipse01;
     set &indata.;
run;
data &outdata.;
     set &indata.;
     do t=&mint. to &maxt. by &stept.;
      x=&x0.+&alen.*cos(t)*cos(&angle.)
-&blen.*sin(t)*sin(&angle.);
      y=&y0.+&alen.*cos(t)*sin(&angle.)
+&blen.*sin(t)*cos(&angle.);
        output;
    end;

run;

%mend calc_ellipse;
```

Example 6: SAS macro to calculate data points of an ellipse.

```
/*****************************************
* Program name        : initialize_
                        options.sas
*
* Author              : Thomas
                        Wollseifen
* Date created        : 29.07.2011
* Purpose             : Initialize
                        options
* Template            :
* Inputs              :
* Outputs             :
* Program completed   : Yes/No
* Updated by          : (Name) - (Date):
*            (Modification and Reason)
*****************************************/
GOPTIONS
        ftext="Arial"
        htext=1
        noborder
        hsize=13cm
        vsize=13cm
        device = emf
          gsfmode = replace
          gsfname = bsfile
        reset=global
        ;

%annomac; *initiliazing annotate
macros;

%global pi;

%let xrange=7;
%let yrange=7;
%let step=1;

%global areax;
%global areay;
%let areax=%sysevalf(&xrange. -2.0);
*position x coordinate of area label;
%let areay=%sysevalf(-&xrange.+0.5);
*position y coordinate of area label;
%let anglex=%sysevalf(-&xrange.+3.0);
*position x coordinate of angle label;
%let angley=%sysevalf(-&xrange.+0.5);
*position y coordinate of anglelabel;
%let labelsize=%sysevalf(1-1/&xrange.);
*size of area label;

%let pi=3.1415927;

title;
footnote;

proc format;
   value method  1=' Convex hull'
           2=' Circle Area'
           3=' PCA'
   ;
run;
```

Example 7: Initialize options, macro variables, and formats.

```
/*****************************************
* Program name        : display_data.
                        sas
*
* Author              : Thomas
                        Wollseifen
* Date created        : 29.07.2011
```

```
* Purpose              : Display data
                         with
                         coordinate
                         system
* Template             :
* Inputs               :
* Outputs              :
* Program completed    : Yes/No
* Updated by           : (Name) - (Date):
*              (Modification and Reason)
****************************************/
%macro display_data( indata =
              ,filename =
              ,annodata =
              ,xs=xs
              ,ys=ys
            );

%local idx i usubjid maxn;

****************************************;
* prepare axis/symbol statements;
****************************************;
axis1 order=-&yrange. to &yrange. by
&step. minor=(number=1) label=(angle
=90 "Y[ cm] ");
axis2 order=-&xrange. to &xrange. by
&step. minor=(number=1) label=("X
[ cm] ") ;

symbol1 i=j           value=none
color=GRB             height=1;
symbol2 i=none        value=dot
color=PURPLE          height=0.6;

****************************************;
* prepare coordinate system with anno-
tate;
****************************************;
data _anno_coord_01;
     length function $20 color $10;
     %system(2,2,2);
     %line(0,-&yrange.,0,&yrange.,
gray,1,0.1);
     %line(-&xrange.,0,&xrange.,0,
gray,1,0.1);
run;
****************************************;
* while loop for each subject;
****************************************;
data _display01;
     set &indata.;
run;

%if &annodata. ne %then %do;
proc sql noprint;
     select distinct usubjid into
:usubjid separated by '#'
     from &annodata.
     order by usubjid
     ;
quit;
%let maxn=&sqlobs.;
%let idx=1;
%do %while (&idx.<=&maxn.);
     %let i=%scan(&usubjid.,&idx.,#);
     %put usubjid=&i.;
     data _display00;
          set &annodata.(where=(usub-
jid=&i.));
     run;
     data _anno_coord_02;
         length function $20 color $10;
          set _anno_coord_01
                _display00;

     run;
     proc sort data=_display01;
          by usubjid;
     run;

     filename bsfile "&path.\&filename.
_&i..emf";
     proc gplot data=_display01(where=
(usubjid=&i.)) annotate=_anno_
coord_02;
         plot &ys.*&xs=1 &ys.*&xs.=2 /
vaxis=axis1 overlay
         haxis=axis2
         noframe
         ;
         by usubjid;

     run;
     quit;
     %let idx=%eval(&idx.+1);

     %end;
%end;
%else %do;
   proc sql noprint;
         select distinct usubjid into
:usubjid separated by '#'
         from &indata.
         order by usubjid
         ;
     quit;
     %let maxn=&sqlobs.;
     %let idx=1;
     %do %while (&idx.<=&maxn.);
         %let
i=%scan(&usubjid.,&idx.,#);
         %put usubjid=&i.;

         proc sort data=&indata.;
              by usubjid;
         run;
         data _display00;
              set &indata.(where=
(usubjid=&i.));
         run;
         filename bsfile "&path.\
&filename._&i..emf";
         proc gplot data=_display00
annotate=_anno_coord_01;
              plot &ys.*&xs.=1
&ys.*&xs.=2 /
                        vaxis=axis1
                        haxis=axis2
                        overlay
                        noframe
         ;
              by usubjid;
```

```
        run;
        quit;
        %let idx=%eval(&idx.+1);
    %end;
%end;
%mend display_data;
```

Example 8: SAS macro to display point set and annotation for different body sway calculation methods.

```
/****************************************
* Program name        : main.sas
*
* Author              : Thomas
                        Wollseifen
* Date created        : 29.07.2011
* Purpose             : Different
                        methods for
                        calculating
                        Body Sway
* Template            :
* Inputs              :
* Outputs             :
* Program completed   : Yes/No
* Updated by          : (Name) - (Date):
*            (Modification and Reason)
****************************************/

%let path=U:\BodySway;

%include "&path.\body_sway_data.
sas";
%include "&path.\initialize_options.
sas";
%include "&path.\m-calc_determinant.
sas";
%include "&path.\m-convex_hull.sas";
%include "&path.\m-polar_coordinates.
sas";
%include "&path.\m-display_data.sas";
%include "&path.\m-calc_bs_area_
circlemethod.sas";
%include "&path.\m-calc_ellipse.sas";
%include "&path.\m-pca_ellipse.sas";
%include "&path.\m-evaluate_body_
sway_area.sas";

%generate_simulation_data(outdata
=sw01); *prepares data with x,y coord.
and usubjid;
%display_data(indata=sw01, filename
=bs01);

/*Convex hull*/
title "Convex hull";
%convex_hull(indata=sw01, x=xs, y
=ys, z=usubjid, outdata=ch, outanno
=ch01anno);
%display_data(indata=sw01, filename
=ch01, annodata=ch01anno);

/*Mean circle calculation*/
title "Mean circle area";
%calc_bs_area_circlemethod(indata
=sw01, outdata=cm);
%display_data(indata=sw01, filename
=cm01, annodata=cm_);

/*PCA ellipse*/
title "PCA ellipse";
%pca_ellipse(indata=sw01, outdata
=pca);

%display_data(indata=sw01, filename
=pca01, annodata=anno_pca);

data area;
  set area_ch
   area_cm
   area_pca;

run;
%evaluate_bs_area( indata=area, outfile
=mean_area_comparison.rtf); *compares
different methods
```

Example 9: Main program for calling different methods and displaying data.