Least squares fitting of conic sections with type identification by NURBS of degree two

I. SEUFER* AND H. SPÄTH†

Abstract. *Fitting of conic sections is used in reflectometry, aircraft industry, metrology, computer vision, astronomy and propagation of sound waves [5]. So far numerical algorithms assume the type of the conic section to be known in advance. We consider the problem of additionally identifying the type during fitting, i.e. deciding whether the given data are better fitted by an ellipse, a hyperbola or a parabola. To solve this problem we apply a well-known descent algorithm [3, 4, 6] to NURBS of degree two. Numerical examples will be given.*

Key words: *conic sections, least squares fitting*

AMS subject classifications: 65D10

Received June 5, 1999 Accepted July 1, 1999

1. Introduction

Let $((x_{ik}, i = 1, \ldots, \ell), k = 1, \ldots, m \geq 5)$ be given measured data points in the plane ($\ell = 2$) or in the space ($\ell = 3$). If some conic section of unknown type should fit the data, you need some representation where the type is not yet specified. For $\ell = 2$ such a representation is given by the implicit form

$$Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Ey + F = 0\,. \tag{1}$$

Putting

$$a = \det \begin{pmatrix} A & B & D \\ B & C & E \\ D & E & F \end{pmatrix}\,, \quad b = \det \begin{pmatrix} A & B \\ B & C \end{pmatrix}\,, \quad c = A + C\,, \tag{2}$$

you will get a hyperbola for $b < 0$, a parabola for $b = 0$, and an ellipse for $b > 0$ in the case of $a \neq 0$; for $a = 0$ there are some special cases like a pair of straight lines [1].

*Department of Mathematics, University of Oldenburg, Postfach 2503, D-26111 Oldenburg, Germany

†Department of Mathematics, University of Oldenburg, Postfach 2503, D-26111 Oldenburg, Germany, e-mail: spaeth@mathematik.uni-oldenburg.de

This implicit form cannot be used for a geometrical fit, where the sum of squared (orthogonal) distances from the given data points to the conic section is minimized. Only some parametric representation

$$x_i = f_i(t) \quad (i = 1, ..., \ell)$$

is suitable. An example for $\ell = 2$ is

$$x_1 = \alpha + p\cos t\,, \quad x_2 = \beta + q\sin t \quad (0 \le t < 2\pi)\,,$$

which represents an ellipse in the normal position with center (α, β) and axes p and q. The only parametrization known to us, including the type of the conic as parameter and being also valid for $\ell = 3$, is that one by NURBS of degree two [2], i.e. $\ell = 2$ or $\ell = 3$ functions

$$x_i = \frac{a_i(1-t)^2 + 2b_i wt(1-t) + c_i t^2}{(1-t)^2 + 2wt(1-t) + t^2}\,, \quad t \in [0,1],\ i = 1, \ldots, \ell\,. \tag{3}$$

Here $\mathbf{a} = (a_1, \ldots, a_\ell)^T$, $\mathbf{b} = (b_1, \ldots, b_\ell)^T$, $\mathbf{c} = (c_1, \ldots, c_\ell)^T$ are three defined control points in the plane ($\ell = 2$) or in the space ($\ell = 3$) and w is a parameter determining the type of the conic. For $0 < w < 1$ equation (3) gives ellipses, for $w = 1$ parabolas, and for $w > 1$ hyperbolas. (The case of $w = 0$ is not of interest to us.) In (3) you also have to consider $w > 0$ and $w < 0$ simultaneously. For $w > 0$ you get that part of the conic lying in the convex hull of the control points and for $w < 0$ the complementary part. In *Figure 1* this is shown for an ellipse.

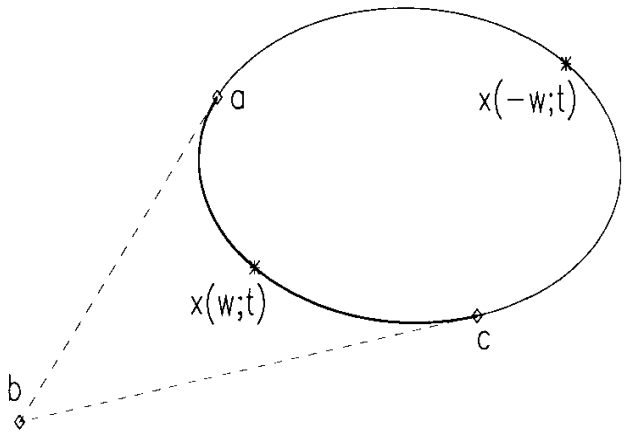

Figure 1. *An ellipse represented by a NURB of degree 2*

In the following we develop a descent algorithm similar to that one shown in [3, 4, 5, 6]. It can be used for $\ell = 2$ or $\ell = 3$, i.e. for conic sections in the plane or in the space. The control points $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$, the type parameter w and the parameter values $\mathbf{t} = (t_1, \ldots, t_m)^T$, corresponding to those values $t_k \in [0,1]$ where the perpendicular through the k-th given data point onto the conic meets it, have to be determined. As the conic consists of two parts we have to associate with each t_k some $v_k = w$ or $v_k = -w$. Without loss of generality, we will always assume $w > 0$.

2. Normalization of the control points

For a given conic K you will find the NURBS parametrization as follows, see [2]. Choose two points $\mathbf{a}$ and $\mathbf{c}$ on K (together on one branch if it were hyperbola) such that the tangents of K through these points have a finite crossing point $\mathbf{b}$. Then $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are possible control points. The corresponding value of w is easily calculated (see [2, p. 226]). Thus some NURBS parametrization is far from being unique (see *Figure 2*). Only $w < 1$ or $w = 1$ or $w > 1$ is significant. This can also be seen from the numerical examples in *Section 5*.

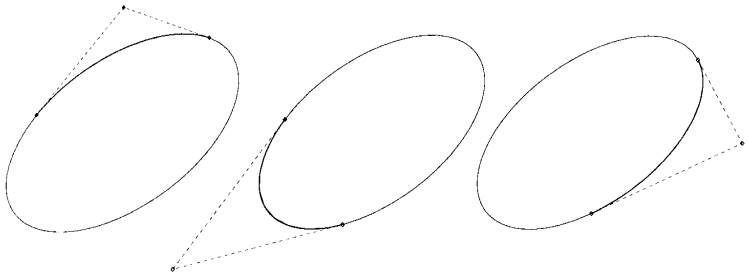

Figure 2. *Different NURBS parametrizations of an ellipse*

Depending on the given starting values for w and $\mathbf{t}$ the iterative method will end up with corresponding values of $\mathbf{a}, \mathbf{b}, \mathbf{c}, w$, and $\mathbf{t}$.

In order to be able to compare results for $w < 1$ or $w > 1$ it is necessary to normalize $\mathbf{a}, \mathbf{b}, \mathbf{c}$, and w in some way. We will explain this for an ellipse in the normal position in the plane, i.e. for

$$\frac{x_1^2}{p^2} + \frac{x_2^2}{q^2} = 1\,.$$

Defining

$$\mathbf{a} = \begin{pmatrix} 0 \\ q \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} -p \\ q \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} -p \\ 0 \end{pmatrix}$$

we get $w = \pm\frac{\sqrt{2}}{2}$ which can be verified when putting (3) into the above implicit representation. A generalization for rotated ellipses (see *Figure 3*) is possible. Also similar considerations can be done for the hyperbola and the parabola.

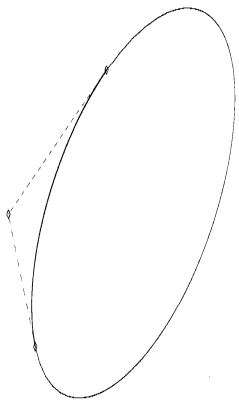

Figure 3. *An ellipse in a normalized NURBS parametrization*

3. Descent algorithm

The function to be minimized for a geometrical fit is

$$S(\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{v};\mathbf{t}) = \sum_{i=1}^{\ell}\sum_{k=1}^{m} S_{ik}(a_i,b_i,c_i,v_k;t_k) \tag{4}$$

where

$$S_{ik}(a_i,b_i,c_i,v_k;t_k) = \left(x_{ik} - \frac{a_i(1-t_k)^2 + 2b_i v_k t_k(1-t_k) + c_i t_k^2}{(1-t_k)^2 + 2v_k t_k(1-t_k) + t_k^2}\right)^2 . \tag{5}$$

Here we have $v_k = w$ or $v_k = -w$ and $t \in [0,1)$. Regarding some fixed value of w the necessary conditions for a local minimum of (4) are

$$\frac{\partial S}{\partial a_i} = \frac{\partial S}{\partial b_i} = \frac{\partial S}{\partial c_i} = 0 \quad \text{for} \quad i = 1,\dots,\ell \tag{6}$$

and

$$\frac{\partial S}{\partial t_k} = 0 \quad \text{for} \quad k = 1,\dots,m\,. \tag{7}$$

The conditions (6) give for each $i \in \{1,\dots,\ell\}$ a linear system of three equations with three unknowns. If we define

$$N_k := (1-t_k)^2 + 2v_k t_k(1-t_k) + t_k^2\,, \tag{8}$$

then for $i \in \{1, \dots, \ell\}$ we have

$$\frac{\partial S}{\partial a_i} = \sum_{k=1}^{m} \frac{\partial S_{ik}}{\partial a_i} = 0$$

$$\Leftrightarrow \sum_{k=1}^{m} \left(x_{ik} - \frac{a_i(1-t_k)^2 + 2b_i v_k t_k (1-t_k) + c_i t_k^2}{N_k} \right) \cdot \frac{(1-t_k)^2}{N_k} = 0$$

$$\Leftrightarrow a_i \sum_{k=1}^{m} \frac{(1-t_k)^4}{N_k^2} + b_i \sum_{k=1}^{m} \frac{2v_k t_k (1-t_k)^3}{N_k^2} + c_i \sum_{k=1}^{m} \frac{t_k^2 (1-t_k)^2}{N_k^2} = \sum_{k=1}^{m} x_{ik} \frac{(1-t_k)^2}{N_k}.$$

Similar calculations for b_i and c_i give altogether ℓ systems

$$A\mathbf{z}_i = \mathbf{d}_i \tag{9}$$

with a common symmetric coefficient matrix

$$A = \begin{pmatrix} \sum_{k=1}^{m} \frac{(1-t_k)^4}{N_k^2} & \sum_{k=1}^{m} \frac{2v_k t_k (1-t_k)^3}{N_k^2} & \sum_{k=1}^{m} \frac{t_k^2 (1-t_k)^2}{N_k^2} \\ \sum_{k=1}^{m} \frac{2v_k t_k (1-t_k)^3}{N_k^2} & \sum_{k=1}^{m} \frac{4v_k^2 t_k^2 (1-t_k)^2}{N_k^2} & \sum_{k=1}^{m} \frac{2v_k t_k^3 (1-t_k)}{N_k^2} \\ \sum_{k=1}^{m} \frac{t_k^2 (1-t_k)^2}{N_k^2} & \sum_{k=1}^{m} \frac{2v_k t_k^3 (1-t_k)}{N_k^2} & \sum_{k=1}^{m} \frac{t_k^4}{N_k^2} \end{pmatrix} \in \mathbb{R}^{3\times 3},$$

and with vectors

$$\mathbf{d}_i = \begin{pmatrix} \sum_{k=1}^{m} x_{ik} \frac{(1-t_k)^2}{N_k} \\ \sum_{k=1}^{m} x_{ik} \frac{2v_k t_k (1-t_k)}{N_k} \\ \sum_{k=1}^{m} x_{ik} \frac{t_k^2}{N_k} \end{pmatrix} \in \mathbb{R}^3 \quad \text{and} \quad \mathbf{z}_i = \begin{pmatrix} a_i \\ b_i \\ c_i \end{pmatrix} \in \mathbb{R}^3.$$

As $A = A^T$ is a matrix of normal equations, it is nonsingular if at least two t_k are different. Thus the ℓ systems (9) normally are uniquely solvable. Numerically we solve the corresponding overdetermined linear systems with the modified Gram-Schmidt method (see [9, Subroutine MGS]).

Calculating (7) you can see that t_k only appears in the k-the term of S. Thus

we temporarily omit the index k and write

$$\begin{aligned} t &:= t_k, \\ x_i &:= x_{ik}, \\ v &:= v_k, \\ Z_i &:= a_i(1-t)^2 + 2b_i vt(1-t) + c_i t^2, \\ N &:= N_k, \\ S_i &:= \left(x_i - \frac{Z_i}{N}\right)^2 = S_{ik}\,. \end{aligned}$$

Then for each $k \in \{1, \ldots, m\}$ we have

$$\frac{\partial S}{\partial t_k} = \sum_{i=1}^{\ell} \frac{\partial S_i}{\partial t} = 0$$

$$\Leftrightarrow \sum_{i=1}^{\ell} S_i \frac{\frac{\partial Z_i}{\partial t} N - Z_i \frac{\partial N}{\partial t}}{N^2} = \frac{1}{N^3} \sum_{i=1}^{\ell} (Nx_i - Z_i) \left(\frac{\partial Z_i}{\partial t} N - Z_i \frac{\partial N}{\partial t}\right) = 0\,.$$

After a multiplying by N^3 we get the equation

$$\sum_{i=1}^{\ell} (Nx_i - Z_i) \left(\frac{\partial Z_i}{\partial t} N - Z_i \frac{\partial N}{\partial t}\right) = 0\,. \tag{10}$$

As the functions N and Z_i are polynomials of degree two, the equation (10) also is a polynomial equation. After some lengthy calculations we get

$$\sum_{i=1}^{\ell} \sum_{j=0}^{4} p_{ij} t^j = \sum_{i=1}^{\ell} p_{i4} t^4 + \sum_{i=1}^{\ell} p_{i3} t^3 + \sum_{i=1}^{\ell} p_{i2} t^2 + \sum_{i=1}^{\ell} p_{i1} t + \sum_{i=1}^{\ell} p_{i0} = 0 \tag{11}$$

where

$$\begin{aligned} p_{i4} &= \alpha_i(v-1)(2(v-1)x_i - \alpha_i + \gamma_i), \\ p_{i3} &= -\alpha_i(v-1)(2(v-1)x_i + \gamma_i) + \delta_i(2(v-1)x_i - \alpha_i + \gamma_i), \\ p_{i2} &= \alpha_i(v-1)a_i - \delta_i(3(v-1)x_i + \gamma_i) + \beta_i v(b_i - x_i), \\ p_{i1} &= \delta_i(a_i - x_i) + \beta_i v(2(v-1)x_i + \gamma_i), \\ p_{i0} &= v(b_i - a_i)(x_i - a_i), \end{aligned}$$

and

$$\alpha_i := (a_i - c_i)\,, \quad \beta_i := (b_i - a_i)\,, \quad \gamma_i := 2(a_i - b_i v)\,, \quad \delta_i := \alpha_i + 2\beta_i v.$$

Thus the conditions (7) are m polynomial equations of degree four. Only for $v_k = 1$ the degree will be three. All zeros can be calculated either in closed form [1] or by some polynomial solver like RPOLY [7]. The existence of at least one real solution t_k can be assumed [6]. In the case of several real solutions t_k we select that one that minimizes the k-th term of S. (Thus the error vectors will be perpendicular to the actual conic.) This guarantees the descent property of the following algorithm for a fixed value of w:

Step 0. Let suitable starting values $t_1^{(0)}, \dots, t_m^{(0)}$ be given (see remark below). Set $it = 0$ (number of iterations).

Step 1. Solve the linear systems (9) with $t_k := t_k^{(it)}$, $k = 1, \dots, m$ and thus determine

$$a_i^{(it+1)} := a_i\,, \quad b_i^{(it+1)} := b_i\,, \quad c_i^{(it+1)} := c_i\,, \quad i = 1, \dots, \ell\,.$$

Step 2. Solve the m single equations (10) with

$$a_i := a_i^{(it+1)}\,, \quad b_i := b_i^{(it+1)}\,, \quad c_i := c_i^{(it+1)}\,, \quad i = 1, \dots, \ell$$

and select that real solution t_k that minimizes $S_k := S_{1k} + \cdots + S_{\ell k}$. Set

$$t_k^{(it+1)} := t_k\,, \quad it := it + 1\,.$$

If you are beyond a given maximal number of iterations or if the unknowns or the value of S do not longer change significantly, then stop; otherwise go back to Step 1.

Remark: A general method of estimating $t_k^{(0)}$ $(k = 1, \dots, m)$ cannot be recommended. If the data indicate only some part of some conic, then choosing the $t_k^{(0)}$ equidistant in $[0, 1)$ is one possibility whereas, when the data are indicating some whole conic, the interval $[-1, 2)$ was more successful for the first iteration. Also, ordering the given data points $((x_{ik}, i = 1, \dots, \ell), k = 1, \dots, m)$ in some way and numbering the $t_k^{(0)}$ $(k = 1, \dots, m)$ correspondingly may be a good advice. Nevertheless, different starting values may produce different local minima. Thus it is a natural mean to vary those values corresponding to several heuristics.

4. Identification of the type of the conic section

So far the algorithm only works for a fixed value of w. To determine also w, when minimizing S, requires to solve in addition to (6) and (7) the necessary condition

$$\frac{\partial S}{\partial w} = 0\,. \tag{12}$$

However, as w appears in each term of S, (12) would give a polynomial equation of degree $3m - 2$, where m is the number of given points. The numerical solution of polynomials of such high degree ($m = 20$ means degree 58) is not easy, but it empirically turned out that S as a function of w for given $\mathbf{a}, \mathbf{b}, \mathbf{c}$, and $\mathbf{t}$ is unimodal within a symmetrical interval with midpoint w_0, e.g. within $[w_0 - .05, w_0 + .05]$. Thus it is possible to minimize S with some method that does not need the derivative of S with respect to w. Such a method is realized, e.g. in the subroutine FMIN [8].

Consequently, our algorithm from *Section 3* is supplemented by

Step 3. For given values $a_i = a_i^{(it)}$, $b_i = b_i^{(it)}$, $c_i = c_i^{(it)}$, and $t_i = t_i^{(it)}$ we calculate in a suitable interval with midpoint $w_0 = w^{(it)}$ a minimum $w_{\min}$ of S. Then we put $w^{(it+1)} := w_{\min}$.

This additional step can be added after Step 1 or after Step 2 or even after both steps. The descent property of the algorithm is preserved in any case.

5. Numerical examples

The normalization discussed in *Section 2* could be inserted in each iteration, but practically it was sufficient to realize it after stopping the algorithm. For the examples given here we always used starting values for t_k which were equally distributed in $[0, 1]$.

At first we tested the algorithm with 10 data points lying exactly on an ellipse (*Table 1*). *Table 2* shows results for fixed values of w. It is clear that S cannot tend to zero for $w \geq 1$. *Table 3* presents results with w not being fixed. You can see that even for starting values $w^{(0)} \geq 1$ the algorithm finds out $w < 1$, i.e. the type identification works in this case.

k	1	2	3	4	5	6	7	8	9	10
x_{1k}	-1.799	-1.703	-1.573	-1.316	-0.414	-0.123	0.011	0.531	0.404	1.193
x_{2k}	-0.849	-0.262	-0.047	0.257	0.912	1.050	1.104	1.254	1.299	1.285

Table 1. *Ten given points on an ellipse (rounded)*

	it	1	10	100	400	1000
w=0.2	$S^{(it)}$	0.05970	0.00245	0.00020	0.00005	0.00000
w=0.4	$S^{(it)}$	0.03903	0.00015	0.00004	0.00000	0.00000
w=0.6	$S^{(it)}$	0.03826	0.00249	0.00098	0.00051	0.00003
w=0.8	$S^{(it)}$	0.04373	0.00762	0.00316	0.00300	0.00259
w=1.0	$S^{(it)}$	0.05063	0.01202	0.00532	0.00532	0.00532
w=1.2	$S^{(it)}$	0.05761	0.01580	0.00725	0.00716	0.00699
w=1.6	$S^{(it)}$	0.07088	0.02176	0.01053	0.01003	0.00923
w=2.0	$S^{(it)}$	0.08321	0.02622	0.01323	0.01236	0.01097

Table 2. *Values of $S^{(it)}$ for fixed w*

	it	1	10	100	400	1000
$w^{(0)}$=0.5	$w^{(it)}$	0.5	0.4611	0.4348	0.4218	0.4238
	$S^{(it)}$	0.03732	0.00061	0.00002	0.00000	0.00000
$w^{(0)}$=1.0	$w^{(it)}$	1.0	0.8930	0.6552	0.4301	0.4116
	$S^{(it)}$	0.05063	0.00983	0.00182	0.00003	0.00000
$w^{(0)}$=2.0	$w^{(it)}$	2.0	1.7985	1.2520	0.5718	0.3818
	$S^{(it)}$	0.08322	0.02388	0.00765	0.00137	0.00000

Table 3. *Values of $S^{(it)}$ with type identification*

Next we considered two sets of data points given in *Tables 4* and *6* which only differ in four points. The first set ought to be best fitted by an ellipse, the second one by a hyperbola (see also *Figure 4* and *5*). Indeed, when increasing the iterations, the values of w develop in both cases correspondingly (see *Tables 5* and *7*).

k	1	2	3	4	5	6	7	8	9	10
x_{1k}	-2	-3	-4	-5	-4	-3	-1	1	2	4
x_{2k}	4	3	2	0	-1	-2	-3	-4	-3	-2

Table 4. *Scattered data points (set 1)*

	it	1	10	100	400	1000
$w^{(0)}$=0.5	w	0.5	0.4611	0.2671	0.2280	0.2270
	$S^{(it)}$	2.01361	1.77644	1.01452	0.88163	0.88163
$w^{(0)}$=1.0	$w^{(it)}$	1.0	0.9183	0.5192	0.2244	0.2235
	$S^{(it)}$	3.27108	2.31068	1.86630	0.88163	0.88163
$w^{(0)}$=2.0	$w^{(it)}$	2.0	1.7600	0.8175	0.3270	0.2177
	$S^{(it)}$	5.54759	2.93128	2.23468	1.42053	0.88163

Table 5. *Fitting with type identification*

k	1	2	3	4	5	6	7	8	9	10
x_{1k}	-2	-3	-4	-5	-4	-3	-1	1	2	4
x_{2k}	6	4	2	0	-1	-2	-3	-4	-4	-4

Table 6. *Scattered data points (set 2)*

	it	1	10	100	400	1000
$w^{(0)}$=0.5	$w^{(it)}$	0.5	0.5168	0.6917	1.3854	1.5600
	$S^{(it)}$	1.10691	0.79854	0.67971	0.54138	0.53791
$w^{(0)}$=1.0	$w^{(it)}$	1.0	1.0291	1.2360	1.5341	1.5723
	$S^{(it)}$	1.43181	0.57987	0.55142	0.53807	0.53791
$w^{(0)}$=2.0	$w^{(it)}$	2.0	1.9700	1.8344	1.6855	1.6529
	$S^{(it)}$	2.95856	0.54929	0.54008	0.53798	0.53791

Table 7. *Fitting with type identification (hyperbola)*

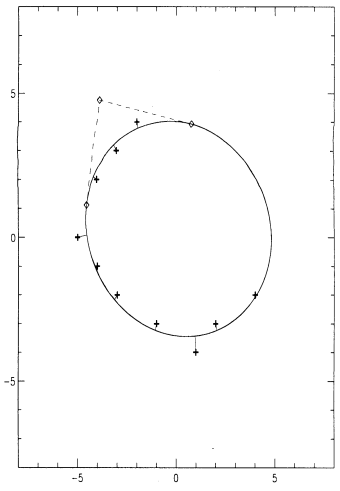

Figure 4. *A fitted ellipse in a normalized NURBS parametrization*

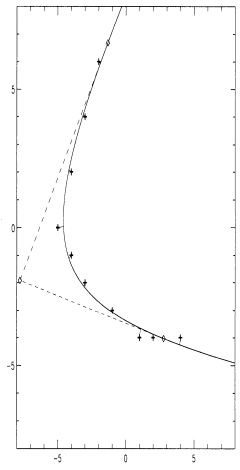

Figure 5. *A fitted hyperbola in a normalized NURBS parametrization*

Finally, we remark that the algorithm is easily adapted for $\ell = 3$. A set of 14 data points in the space is given in *Table 8*. The fitted ellipse is shown in *Figure 6*.

After 148 iterations we got

$$w = 0.789625\,, \quad S = 0.481444,$$
$$\mathbf{a} = (1.265, 2.159, 2.752)^T,$$
$$\mathbf{b} = (0.071, 1.502, 2.716)^T,$$
$$\mathbf{c} = (-0.658, 0.547, 2.349)^T$$

and after normalization

$$w = \tfrac{\sqrt{2}}{2}\,, \quad S = 0.481444,$$
$$\mathbf{a} = (0.299, 1.499, 2.643)^T,$$
$$\mathbf{b} = (-1.295, 0.208, 2.337)^T,$$
$$\mathbf{c} = (-0.907, -0.185, 1.971)^T\,.$$

k	1	2	3	4	5	6	7
x_{1k}	2.10	2.29	1.55	1.40	0.85	0.54	0.31
x_{2k}	2.52	2.23	2.25	2.16	1.84	1.69	1.35
x_{3k}	2.59	2.38	2.88	2.76	2.65	2.70	2.91

k	8	9	10	11	12	13	14
x_{1k}	0.07	-0.45	-0.75	-0.66	-0.77	-0.25	-0.47
x_{2k}	1.51	0.81	0.74	0.36	0.70	0.74	-0.28
x_{3k}	2.53	2.52	2.52	2.35	2.01	2.26	1.94

Table 8. *Data points in space*

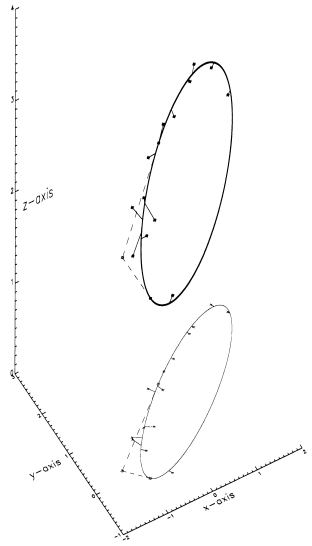

Figure 6. *A fitted ellipse in space*

References

[1] I. N. BRONSTEIN, K. A. SEMENDJAJEW, *Taschenbuch der Mathematik*, Verlag Harri Deutsch, 1987.

[2] G. FARIN, *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, 2nd ed., 1990.

[3] I. SEUFER, *Orthogonale Anpassung mit speziellen ebenen Kurven*, Diplomarbeit, Universität Oldenburg, 1996.

[4] H. SPÄTH, *Least-squares fitting by circles*, Computing **57**(1996), 179–185.

[5] H. SPÄTH, *Orthogonal least squares fitting by conic sections*, in: Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modelling, (S. van Huffel, Ed.), SIAM, Philadelphia, 1997, 259–264.

[6] H. SPÄTH, *Least-squares fitting of ellipses and hyperbolas*, Computational Statistics **12**(1997), 329–341.

[7] Subroutine RPOLY, From: *Collected Algorithms of the ACM, Nr. 493.*

[8] Subroutine FMIN, From: *G. E. Forsythe, M. A. Malcolm, C. . Moler – Computer Methods for Mathematical Computations*, Prentice Hall, 1977.

[9] Subroutine MGS, From: *H. Späth – Mathematical Algorithms for Linear Regression*, Academic Press, 1992.