Electronic Transactions on Numerical Analysis.
Volume 18, pp. 188-197, 2004.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

A NEW SOURCE OF STRUCTURED SINGULAR VALUE DECOMPOSITION PROBLEMS *

ANA MARCO† AND JOSÉ-JAVIER MARTÍNEZ‡

Abstract. The computation of the Singular Value Decomposition (SVD) of structured matrices has become an important line of research in numerical linear algebra. In this work the problem of inversion in the context of the computation of curve intersections is considered. Although this problem has usually been dealt with in the field of exact rational computations and in that case it can be solved by using Gaussian elimination, when one has to work in finite precision arithmetic the problem leads to the computation of the SVD of a Sylvester matrix, a different type of structured matrix widely used in computer algebra. In addition only a small part of the SVD is needed, which shows the interest of having special algorithms for this situation.

Key words. curves, intersection, singular value decomposition, structured matrices.

AMS subject classifications. 14Q05, 65D17, 65F15.

1. Introduction and basic results. The singular value decomposition (SVD) is one of the most valuable tools in numerical linear algebra. Nevertheless, in spite of its advantageous features it has not usually been applied in solving curve intersection problems. Our aim in this paper is to consider the problem of *inversion* in the context of the computation of the points of intersection of two plane curves, and to show how this problem can be solved by computing the SVD of a *Sylvester matrix*, a type of structured matrix widely used in computer algebra. It must also be stressed that only a small part of the SVD will be needed.

Our approach to the intersection of parametric curves uses *algebraic methods*, and an important step of the process consists of the *implicitization* of one of the curves by using resultants. The process leads to the computation of the roots of a polynomial of (usually) high degree, and so these roots will usually be floating point numbers.

Therefore, although we employ algebraic methods the use of floating point arithmetic is unavoidable, contrarily to the idea expressed in the last section of [20], where it is indicated that the algebraic methods *assume the procedure is carried out using exact (integer or rational number) arithmetic.*

In fact, it is clear that the solution of practical problems of not small size needs the combination of symbolic computations and numerical computations, although historically those two classes of algorithms were developed by two distinct groups of people having very little interaction with each other.

In connection with the problem we are dealing with, it can be read in [20] that *inversion -computing the parameter t for a point (x, y) known to lie on the curve- can be performed using Gauss elimination or Cramer's rule.*

While that assert may be correct when working with exact arithmetic and problems of small size, the suggested procedure is far from being adequate with problems of large size which lead to polynomial roots given as floating point numbers.

As it will be seen in Section 2, the use of algebraic methods will allow us to formulate the problem of inversion in terms of the computation of the *nullspace* of a Sylvester matrix of

* Received November 10, 2003. Accepted for publication May 15, 2004. Recommended by F. Marcellán. This work was supported by Research Grant BFM 2003-03510 from the Spanish Ministerio de Ciencia y Tecnología.
† Departamento de Matemáticas, Universidad de Alcalá, Campus Universitario, 28871-Alcalá de Henares, Madrid, Spain. E-mail: ana.marco@uah.es.
‡ Departamento de Matemáticas, Universidad de Alcalá, Campus Universitario, 28871-Alcalá de Henares, Madrid, Spain. E-mail: jjavier.martinez@uah.es.

ETNA
Kent State University
etna@mcs.kent.edu

order l and rank $l-1$, a task which for accuracy reasons is carried out by using the singular value decomposition.

Let us mention that the use of the SVD in the context of curve intersection problems has been advocated in [15], although only for the problem of rank determination. The use of the SVD for a nullspace computation is not considered in [15], since the authors use a different approach involving the solution of *eigenvalue* problems, usually of large size.

The computation of the SVD of a Sylvester matrix has also been suggested as a tool in an important computer algebra application: the computation of polynomial greatest common divisors when the coefficients are inexact or imperfectly known [4].

For the sake of completeness, we include some basic results which are used in the next sections.

1.1. The Sylvester resultant. Following [20], we recall some important facts about resultants, focusing on the *Sylvester resultant*, which due to its simplicity in form and construction is perhaps the best known resultant expression for two polynomials $f(t) = a_0 + a_1 t + \cdots + a_m t^m$ (with $a_m \neq 0$) and $g(t) = b_0 + b_1 t + \cdots + b_n t^n$ (with $b_n \neq 0$).

If we spawn $m+n$ polynomials of degree less or equal to $m+n-1$ by multiplying the degree m polynomial $f(t)$ with the n monomials $t^{n-1}, \cdots, 1$ and by multiplying the degree n polynomial $g(t)$ with the m monomials $t^{m-1}, \cdots, 1$, we can write this process in matrix notation as follows:

$$\begin{pmatrix} a_m & a_{m-1} & \cdots & a_1 & a_0 & & & \\ & a_m & a_{m-1} & \cdots & a_1 & a_0 & & \\ & & \cdots & \cdots & \cdots & & & \\ & & & \cdots & \cdots & \cdots & & \\ & & & & a_m & \cdots & \cdots & a_0 \\ b_n & b_{n-1} & \cdots & b_1 & b_0 & & & \\ & b_n & b_{n-1} & \cdots & b_1 & b_0 & & \\ & & \cdots & \cdots & \cdots & & & \\ & & & \cdots & \cdots & \cdots & & \\ & & & & b_n & \cdots & \cdots & b_0 \end{pmatrix} \begin{pmatrix} t^{m+n-1} \\ t^{m+n-2} \\ \vdots \\ \vdots \\ \vdots \\ t \\ 1 \end{pmatrix} = \begin{pmatrix} t^{n-1} f(t) \\ \vdots \\ t f(t) \\ f(t) \\ t^{m-1} g(t) \\ \vdots \\ t g(t) \\ g(t) \end{pmatrix}.$$

The coefficient matrix (of order $m+n$) is the *Sylvester matrix* for $f(t)$ and $g(t)$. Let us observe that we are showing the structure of the Sylvester matrix by rows, while the precise form of the columns will depend on the relationship between m and n. The *resultant* of $f(t)$ and $g(t)$ (with respect to t) is the determinant of the Sylvester matrix.

It is well-known that if there exists a common root t of the two polynomials $f(t)$ and $g(t)$, then the resultant must be zero. But from the above matricial expression we can derive another deep consequence: if t is a common root of $f(t)$ and $g(t)$, then $(t^{m+n-1}, t^{m+n-2}, \cdots, t, 1)^T$ must be an eigenvector of the Sylvester matrix corresponding to the eigenvalue 0. In other words, the vector $(t^{m+n-1}, t^{m+n-2}, \cdots, t, 1)^T$ must belong to the *nullspace* of the Sylvester matrix.

An alternative resultant formulation, with analogous properties, is the so-called *Bézout resultant*, whose construction and properties can be seen in [8]. The order of the Bézout matrix is $max\{m, n\}$.

1.2. The Singular Value Decomposition. Let us now recall the concept of *singular value decomposition (SVD)*:

Given an $m \times n$ real matrix A (not necessarily square), there exist orthogonal matrices U (of order m) and V (of order n), and a diagonal matrix Σ (of size $m \times n$) such that

ETNA
Kent State University
etna@mcs.kent.edu

$$A = U\Sigma V^T.$$

This factorization of A is called the *singular value decomposition (SVD)* of A.

The r (with $r \leq m, n$) nonzero diagonal entries of Σ are the *singular values* of A (*i.e.* the positive square roots of the eigenvalues of $A^T A$). If there are r (nonzero) singular values, then r is the *rank* of A.

But we also obtain a very important additional advantage from the computation of the SVD of a matrix A of size $m \times n$ (including V, not only Σ): *the last $n - r$ columns of of V form a basis of the nullspace of A* (see [21] and [5]).

The SVD provides the best way of estimating the rank of a matrix whose entries are floating point numbers. This is specially important in the presence of round-off errors. As one can read in [15], although in principle the rank of a matrix can be computed by Gaussian elimination, rounding errors make rank determination a non-trivial task. In this situation, the use of SVD is a more reliable tool.

This fact was clearly stated in a historical paper by Golub and Kahan [9], where we can read the following sentences: "In the past the conventional way to determine the rank of A was to convert A to a row-echelon form... But in floating-point calculations it may not be so easy to decide whether some number is effectively zero or not... *In other words, without looking explicitly at the singular values there seems to be no satisfactory way to assign rank to A*".

Good general references in connection with the computation of the SVD of a matrix are [11] and [5].

The rest of the paper is organized as follows. In Section 2 an algebraic algorithm for computing the intersection of two plane curves is presented, and an example for illustrating it is included. The description of the structure of the SVD problem involved in the intersection problem is the subject of Section 3.

2. Intersection of two plane curves. We will focus on the case of two plane curves given by their parametric equations. An intersection algorithm for the case when the two curves are given by their implicit equations can be seen in [17] and [20]. It must be noticed that the SVD can also be used in a similar way as a tool in the last step of that algorithm: given the x-coordinates of the intersection points, find the corresponding y-coordinates, or vice versa.

Let $P(t) = (x(t) = v_1(t)/w_1(t), y(t) = v_2(t)/w_2(t))$ ($t \in [a, b]$) be a proper rational parametrization of the planar curve C_t and let $Q(u) = (\overline{x}(u) = v_3(u)/w_3(u), \overline{y}(u) = v_4(u)/w_4(u))$ ($u \in [c, d]$) be a proper rational parametrization of the planar curve C_u. Our aim is to compute using algebraic techniques the points where these two curves intersect. Our procedure carries out the successive steps of the classical algebraic algorithm described, for example, in [8], [13], or [17], trying to circumvent some of the difficulties usually associated with it. The algorithm consists of the following steps:

Step 1. Using the *resultant*, compute the implicit polynomial equation $F(x, y) = 0$ of the curve $P(t)$.

Step 2. Substitute the parametric equations $Q(u) = (\overline{x}(u), \overline{y}(u))$ into the implicit equation $F(x, y) = 0$ to obtain a single rational polynomial equation $F(\overline{x}(u), \overline{y}(u)) = 0$.

Step 3. Using a polynomial solver, find all the roots of the numerator of $F(\overline{x}(u), \overline{y}(u)) = 0$ within the interval of interest $[c, d]$. These roots are the parameter values along $Q(u)$ of the intersection points.

ETNA
Kent State University
etna@mcs.kent.edu

Step 4. Using the parametric equations for $Q(u)$, compute the coordinates of the intersection points corresponding to the parameter values found in Step 3.

Step 5. Making use of an *inversion* algorithm for $P(t)$, compute the parameter values along $P(t)$ of the intersection points found in Step 4, and discard those points whose parameters do not belong to $[a, b]$.

Step 1 is called the *implicitization* of the curve $P(t)$. In [16] we present an algorithm to obtain the implicit equation, computing $F(x, y) = Res_t(v_1(t) - xw_1(t), v_2(t) - yw_2(t))$ (the resultant with respect to the variable t of the polynomials $v_1(t) - xw_1(t)$ and $v_2(t) - yw_2(t)$) by means of classical bivariate Lagrange polynomial interpolation. That algorithm reduces the computation of the symbolic determinant $Res_t(v_1(t) - xw_1(t), v_2(t) - yw_2(t))$, one of the most expensive steps in the intersection process, to the computation of the interpolation data and the solution of a linear system whose coefficient matrix is the Kronecker product of two Vandermonde matrices.

As shown in [16], it is a remarkable fact that the algorithm admits a natural parallelization, and it only involves arithmetic operations with numbers.

The construction of the Sylvester (or Bézout) matrix, which is the fastest part of the algorithm, can esily be carried out by using *Maple*.

Step 3 corresponds to the computation of the real roots of a univariate polynomial of high degree, a task in which problems of numerical instability may appear.

The approach we suggest is the use of *Maple* (or *Mathematica* or any other suitable computer algebra system) for maintaining the exactness of the coefficients of the polynomial. It is a remarkable fact that when the degrees of the curves being studied are not small, it is likely that the coefficients of the polynomial cannot be represented even using IEEE double precision arithmetic, as the example we include in this section shows.

The process of computing the real roots can be done with *Maple* by using the general function `fsolve`, specifying an appropriate number of significant digits by means of the command `Digits`. As a general rule, for higher degrees of the polynomial more significant digits will be necessary to obtain a good accuracy.

Another suitable tool for computing accurately the roots of this polynomial of high degree is the algorithm presented in [1], which uses the GNU multiprecision package GMP.

In this way, by using *exact coefficients* and *high precision* along the process of computation of the roots, we can mitigate the well-known problem of instability which is frequently associated with the solving of polynomial equations of high degree.

Step 5 carries out the *inversion process*, that is, the computation of the value of the parameter t corresponding to each one of the *possible* intersection points (x_i, y_i) computed at Step 4. Let (x_0, y_0) one of these points and let M be the matrix obtained when substituting $x = x_0$ and $y = y_0$ into the Sylvester matrix of $v_1(t) - xw_1(t)$ and $v_2(t) - yw_2(t)$ (considered as polynomials in t) constructed at Step 1.

The crucial result for computing the value of t corresponding to (x_0, y_0) is the one introduced at the end of Section 1.1, which in this case tells us that the vector $(t^{m+n-1}, \ldots, t, 1)$ belongs to the nullspace of M. An analogous result holds for the Bézout resultant [15].

So, if the matrix M has rank $m + n - 1$ then the vector $(t^{m+n-1}, \ldots, t, 1)$ is a basis of the nullspace of M. Since V is orthogonal, in the last column of that matrix in the SVD of M we obtain a multiple of that vector with euclidean norm 1: $\alpha(t^{m+n-1}, \ldots, t, 1)$. Taking this into account the value of t corresponding to (x_0, y_0) is $t = \alpha t/\alpha$.

In this way, the computation of the SVD for each matrix obtained by evaluating the Sylvester matrix at the point (x_i, y_i) of the curve gives us the corresponding value of t (*i.e.*

ETNA
Kent State University
etna@mcs.kent.edu

the solution of the inversion problem) when the rank is $m+n-1$, which is the usual case (i. e. the case of a regular point).

As the matrices whose SVD have to be computed are of order $m+n$ and rank $m+n-1$, and we are only interested in computing the right singular vector associated to its smallest singular value, we can consider, for computing this SVD, the partial singular value decomposition algorithm (PSVD) described in [22] and [23], and whose FORTRAN code is available in *netlib* (see `www.netlib.org/vanhuffel/psvd-doc`).

Finally, we illustrate the algorithm presented in this section with the following small example, taken from [3]. For the sake of completeness we show the application of the algorithm by using the computer algebra system *Maple*. When numerical computations are required (steps 3, 4 and 5 of the algorithm) they are performed fixing 16 digits of precision (by using the sentence `Digits:=16`). The first two steps are carried out by using exact rational arithmetic.

Example. Let $P(t)=(x(t),y(t))$ (with $t\in[0,1]$) be a rational Bézier curve of degree 5 where

$$x(t)=\frac{v_1(t)}{w_1(t)}=\frac{-\frac{1}{2}(1-t)^5-\frac{5}{4}t^2(1-t)^3+\frac{5}{4}t^4(1-t)+\frac{1}{2}t^5}{(1-t)^5+\frac{5}{2}t(1-t)^4+\frac{5}{2}t^2(1-t)^3+\frac{5}{2}t^3(1-t)^2+\frac{5}{2}t^4(1-t)+t^5},$$
$$y(t)=\frac{v_2(t)}{w_2(t)}=\frac{\frac{1}{2}(1-t)^5-\frac{5}{4}t(1-t)^4-\frac{5}{4}t^2(1-t)^3+\frac{5}{4}t^3(1-t)^2+\frac{5}{4}t^4(1-t)-\frac{1}{2}t^5}{(1-t)^5+\frac{5}{2}t(1-t)^4+\frac{5}{2}t^2(1-t)^3+\frac{5}{2}t^3(1-t)^2+\frac{5}{2}t^4(1-t)+t^5},$$

and let $Q(u)=(\overline{x}(u),\overline{y}(u))$ (with $u\in[0,1]$) be a polynomial Bézier curve of degree 8 where

$$\begin{aligned}\overline{x}(u)=&-24t(1-t)^7+182t^2(1-t)^6-392t^3(1-t)^5+392t^5(1-t)^3\\&-182t^6(1-t)^2+24t^7(1-t),\\\overline{y}(u)=&-\frac{1}{4}(1-t)^8+32t(1-t)^7-476t^2(1-t)^6+1960t^3(1-t)^5\\&-3010t^4(1-t)^4+1960t^5(1-t)^3-476t^6(1-t)^2+32t^7(1-t)-\frac{1}{4}t^8.\end{aligned}$$

Our aim is to compute the points where these two curves intersect (see Fig. 2.1).
The results obtained at each step of our algorithm are presented below.

-*Step 1.* Let $p=v_1(t)-xw_1(t)$ and $q=v_2(t)-yw_2(t)$. The following *Maple* instruction

```
S:=sylvester(p,q,t);
```

computes the Sylvester matrix of p and q with respect to t. We will need it in Step 5 and we will denote it by S:

ETNA
Kent State University
etna@mcs.kent.edu

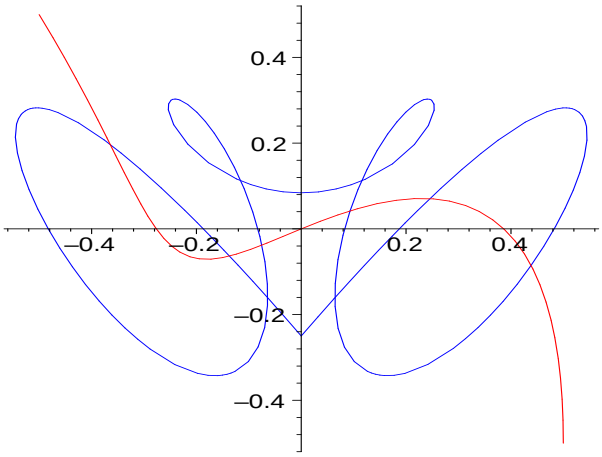

FIG. 2.1.

$$\begin{pmatrix}
1 & -5 & \frac{35}{4} & -\frac{5}{2}x-\frac{25}{4} & \frac{5}{2}x+\frac{5}{2} & -\frac{1}{2}-x & 0 & 0 & 0 & 0\\
0 & 1 & -5 & \frac{35}{4} & -\frac{5}{2}x-\frac{25}{4} & \frac{5}{2}x+\frac{5}{2} & -\frac{1}{2}-x & 0 & 0 & 0\\
0 & 0 & 1 & -5 & \frac{35}{4} & -\frac{5}{2}x-\frac{25}{4} & \frac{5}{2}x+\frac{5}{2} & -\frac{1}{2}-x & 0 & 0\\
0 & 0 & 0 & 1 & -5 & \frac{35}{4} & -\frac{5}{2}x-\frac{25}{4} & \frac{5}{2}x+\frac{5}{2} & -\frac{1}{2}-x & 0\\
0 & 0 & 0 & 0 & 1 & -5 & \frac{35}{4} & -\frac{5}{2}x-\frac{25}{4} & \frac{5}{2}x+\frac{5}{2} & -\frac{1}{2}-x\\
-1 & \frac{5}{2} & -\frac{15}{2} & \frac{35}{4}-\frac{5}{2}y & -\frac{15}{4}+\frac{5}{2}y & \frac{1}{2}-y & 0 & 0 & 0 & 0\\
0 & -1 & \frac{5}{2} & -\frac{15}{2} & \frac{35}{4}-\frac{5}{2}y & -\frac{15}{4}+\frac{5}{2}y & \frac{1}{2}-y & 0 & 0 & 0\\
0 & 0 & -1 & \frac{5}{2} & -\frac{15}{2} & \frac{35}{4}-\frac{5}{2}y & -\frac{15}{4}+\frac{5}{2}y & \frac{1}{2}-y & 0 & 0\\
0 & 0 & 0 & -1 & \frac{5}{2} & -\frac{15}{2} & \frac{35}{4}-\frac{5}{2}y & -\frac{15}{4}+\frac{5}{2}y & \frac{1}{2}-y & 0\\
0 & 0 & 0 & 0 & -1 & \frac{5}{2} & -\frac{15}{2} & \frac{35}{4}-\frac{5}{2}y & -\frac{15}{4}+\frac{5}{2}y & \frac{1}{2}-y
\end{pmatrix}$$

The polynomial defining the implicit equation of $P(t)$ is the determinant of S:

$$\begin{aligned}F(x,y) = &-\frac{507}{32}x^5-\frac{2145}{64}x^4y+\frac{22425}{128}x^4-\frac{5625}{8}x^3+\frac{975}{64}x^3y-\frac{345}{32}x^3y^2+\frac{58125}{1024}x^2\\
&-\frac{53025}{256}y^2x^2+\frac{50625}{128}x^2y+\frac{165}{16}x^2y^3-\frac{136875}{1024}xy+\frac{9375}{128}x+\frac{20625}{256}y^2x\\
&+\frac{33525}{256}xy^3-\frac{15}{8}xy^4-\frac{5325}{128}y^4-\frac{69375}{512}y^2-\frac{35625}{256}y^3-\frac{9375}{64}y-\frac{339}{64}y^5.\end{aligned}$$

-*Step 2.* The polynomial obtained by substituting $Q(u) = (\overline{x}(u), \overline{y}(u))$ into $F(x,y)$ is an irreducible polynomial of degree 40 whose leading term is

$$\frac{34373924217136700875292 7}{2048}u^{40}.$$

-*Step 3.* The polynomial computed in Step 2 has the following roots:

$$\begin{aligned}u_1 &= 0.6263135462704812\cdot 10^{-2}\\
u_2 &= 0.2078384532955000\cdot 10^{-1}\\
u_3 &= 0.2356938205913398\end{aligned}$$

ETNA
Kent State University
etna@mcs.kent.edu

$$u_4 = 0.7493929838366283$$
$$u_5 = 0.9061596468247985$$
$$u_6 = 0.9877287235407141$$
$$u_7 = -0.1390025326221537$$
$$u_8 = -0.1565456997018447 \cdot 10^{-1}.$$

They have been computed by using the *Maple* command `fsolve`.

-*Step 4.* As $Q(u)$ is a Bézier curve and $u_7, u_8 \notin [0,1]$, there are only 6 *possible* points in the intersection of the two curves:

$$(x_1, y_1) = (-0.1370658337732254, -0.6346706453153401 \cdot 10^{-1})$$
$$(x_2, y_2) = (-0.3644703079173818, 0.1968785910610758)$$
$$(x_3, y_3) = (-0.7523503060796923 \cdot 10^{-1}, -0.3785917023135303 \cdot 10^{-1})$$
$$(x_4, y_4) = (0.931399241331760 \cdot 10^{-1}, 0.4193972385076171 \cdot 10^{-1})$$
$$(x_5, y_5) = (0.4384598684117452, -0.852719535168224 \cdot 10^{-1})$$
$$(x_6, y_6) = (0.2453565154051313, 0.704612477501064 \cdot 10^{-1}).$$

-*Step 5.* Let us consider the point (x_1, y_1). The following *Maple* code returns the singular values of M and the value of the parameter t corresponding to (x_1, y_1):

```
M:=subs({x=-0.1370658337732254,y=-0.06346706453153401, op(S));
evalf(Svd(M,V,'right')); t1:=V[9,10]/V[10,10];
```

$$[29.80548615701261, 20.79518553103211, 11.71479043767689, 5.873738271013132,$$
$$2.979034891894657, 1.271701392114136, 0.5114521101863383, 0.4195631347621551,$$
$$0.2213148780388505, 0.1079996523540034 \cdot 10^{-16}]$$

$$t_1 = 0.4060841000766337$$

Let us observe that the system is only computing the singular values and the matrix V, and not the full singular value decomposition.

As $t_1 \in [0,1]$, the interval where Bézier curves are defined, (x_1, y_1) is a point in $P(t)$, and therefore it is a point in the intersection of the two curves.

Proceeding in the same way for the other five points we obtain

$$t_2 = 0.1323072487300945$$
$$t_3 = 0.4521172363813014$$
$$t_4 = 0.5537900358956320$$
$$t_5 = 0.8075967418165704$$
$$t_6 = 0.6425966835471886$$

ETNA
Kent State University
etna@mcs.kent.edu

and therefore the 6 points calculated in Step 3 are all the points in which $P(t)$ and $Q(u)$ intersect.

Remark 1. Let us observe that if we solve the homogeneous linear system $Mx = 0$ by means of the following *Maple* code

```
v:=vector([0,0,0,0,0,0,0,0,0,0]); linsolve(M, v);
```

instead of computing the SVD of M we obtain the trivial solution $x = 0$, and the value of the parameter t_1 cannot be computed.

It is illustrative to observe that although nothing is said in the description of the command `nullspace` of *Maple*, if the user asks for more information from the system (for example by writing the instruction `infolevel[nullspace]:=5`), then the system admits Gaussian elimination is not being used, by showing the following message:

```
nullspace:  doing a singular value decomposition
```

Of course, the cost of using the singular value decomposition is much higher that that of using Gaussian elimination, but it must be taken into account that here reliability is much more important than efficiency, as it is stressed in Section 1.2 in a general setting. This is the fact that *Maple* is implicitly recognizing.

Remark 2. Let us point out that in this example multiple precision arithmetic is needed in order to represent the exact coefficients of the polynomial obtained in Step 2. This necessity has also been recognized in [3].

3. Remarks on the special features of the SVD problem. Our aim in this paper has been to show how in the inversion problem, very common in CAGD applications, the computation of the SVD can be a valuable tool. The special structure of the SVD problem which arises in the last step of the intersection algorithm should motivate the research of algorithms adapted to it. In this last section we will remark some of the special features of the problem and we will give a brief overview of some related work.

To begin with, let us observe that the Sylvester matrix of $v_1(t) - xw_1(t)$ and $v_2(t) - xw_2(t)$ evaluated at each *possible* intersection point has the following block structure:

$$M = \begin{pmatrix} M_1 \\ M_2 \end{pmatrix}.$$

In addition, the blocks M_1 and M_2 are *upper triangular Toeplitz matrices.*

On the other hand, it can easily be seen that M has a *displacement rank* equal to two (see [14]).

Algorithms for the computation of very accurate singular value decompositions of matrices with unit displacement rank have recently been presented in [6]. Although the author claims those results cannot be extended to matrices with displacement rank higher than one, we think the case of displacement rank equal to two may be an interesting problem. According to [6], it is likely that in that case higher precision will be needed.

It is also important to take into account that, as we explained in Section 2, only a small part of the SVD of the evaluated Sylvester matrix M is needed. In fact we know that M has the singular value zero and *only the last column of V is needed.*

For this reason we include in this section a brief review of some papers related to the computation of *partial* singular value decompositions which have appeared in the last few

ETNA
Kent State University
etna@mcs.kent.edu

years, and which can serve as a guide to find improved algorithms adapted to the specific problem we have described.

Of course, the computation of the largest singular values and associated singular vectors is the natural choice when one considers the problem of approximating a matrix by another matrix of lower rank ([11], [2]). In addition, it must be taken into account that the computation of the smallest singular values is usually a more difficult task than the computation of the largest ones.

A portion of the singular values and vectors of a large sparse matrix A is computed in [12] by using a Jacobi-Davidson type method which makes use of the block structure of the augmented matrix

$$\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}.$$

In [10], a block Lanczos method for computing the greatest singular values and associated vectors of a large and sparse matrix is presented.

Another procedure for the computation of a few of the largest singular values and the associated singular vectors of large sparse matrices has been given in [24]. In this case the method has its roots in the theory of *orthogonal polynomials* (see [7], in particular §5.5) . In addition, an overview of a *parallel* implementation is presented.

In [18], a modification of the preconditioning step of Davidson method is used for computing the smallest singular values and the corresponding right singular vectors of large sparse matrices.

The subject of the computation of the smallest singular value and the corresponding singular vectors of a matrix has also been considered in [19]. In this case, as in our application to the inversion problem, the matrix is assumed to be a square matrix of order n, and for its rank the cases $r = n$ and $r = n - 1$ are treated.

Among the recent research devoted to the subject of the computation of a few of the smallest singular values, the work presented in [22] and [23] deserves a special attention, due to two important reasons: the algorithm is available in *netlib* and the study of the computational complexity is carried out in detail in [23]. In addition, the application to the computation of the *nullspace* of a matrix is stressed, as well as the applicability to *total least squares* problems ([2], [11]).

The partial singular value decomposition algorithm (PSVD) presented in [22] and [23] is specially suitable in our case because only the smallest singular value and its associated right singular vector are required, and the gap between the singular values associated to the desired and undesired singular vectors is large. It must be noticed that if in this situation we consider a square matrix of order n, the computational cost of the PSVD algorithm is of $4n^3/3 + O(n^2)$ multiplications (or divisions) while the computational cost of the classical algorithm for the computation of the full SVD (i.e. Σ, U and V) is of $21n^3/2 + O(n^2)$ multiplications, and the computational cost of the classical SVD for computing Σ and V is of $6n^3 + O(n^2)$ multiplications. It is also a remarkable fact that the PSVD algorithm is only 4 times more expensive than Gaussian elimination (which has a cost of $n^3/3 + O(n^2)$ multiplications) despite it is much more reliable when floating point arithmetic is used [23].

REFERENCES

[1] D. A. BINI AND G. FIORENTINO, *Design, analysis, and implementation of a multiprecision polynomial rootfinder*, Numerical Algorithms 23 (2000), pp. 127–173.

[2] A. BJÖRCK, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

ETNA
Kent State University
etna@mcs.kent.edu

[3] G. CASCIOLA, F. FABBRI AND L. B. MONTEFUSCO, *An application of fast factorization algorithms in Computer Aided Geometric Design*, Linear Algebra and Its Applications 366 (2002), pp. 121–138.
[4] R. M. CORLESS, P. M. GIANNI, B. M. TRAGER AND S. M. WATT, *The singular value decomposition for polynomial systems*, Proc. ISSAC (1995), pp. 195–207.
[5] J. W. DEMMEL, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
[6] J. W. DEMMEL, *Accurate singular value decompositions of structured matrices*, SIAM J. Matrix Anal. Appl. 21 (2000), No. 2, pp. 562–580.
[7] W. GAUTSCHI, *The interplay between classical analysis and (numerical) linear algebra – A tribute to Gene H. Golub*, Electronic Transactions on Numerical Analysis 13 (2002), pp. 119–147.
[8] R. N. GOLDMAN, T. W. SEDERBERG AND D. C. ANDERSON, *Vector elimination: A technique for the implicitization, inversion and intersection of planar parametric rational polynomial curves*, Computer Aided Geometric Design 1(1984), pp. 327–356.
[9] G. H. GOLUB, AND W. KAHAN, *Calculating the singular values and pseudo-inverse of a matrix*, J. SIAM Numer. Anal., Ser. B, 2 (1965), No. 2, pp. 205–224.
[10] G. H. GOLUB F. T. LUK AND M. L. OVERTON, *A block Lanzos method for computing the singular values and corresponding singular vectors of a matrix*, ACM Transa. Math. Software 7 (1981), No. 2, pp. 149–169.
[11] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 3rd edition, Johns Hopkins University Press, Baltimore, 1996.
[12] M. E. HOCHSTENBACH, *A Jacobi-Davidson type SVD method*, SIAM J. Sci. Comput. 23 (2001), No. 2, pp. 606–628.
[13] J. HOSCHEK AND D. LASSER, *Fundamentals of Computer Aided Geometric Design*, A. K. Peters, Wellesley, 1993.
[14] T. KAILATH AND A. H. SAYED, *Displacement structure: Theory and applications*, SIAM Review 37 (1995), No. 3, pp. 297 –386.
[15] D. MANOCHA AND J. DEMMEL, *Algorithms for intersecting parametric and algebraic curves I: simple intersections*, ACM Transactions on Graphics 13 (1994), pp. 73–100.
[16] A. MARCO AND J. J. MARTÍNEZ, *Using polynomial interpolation for implicitizing algebraic curves*, Computer Aided Geometric Design 18 (2001), pp. 309–319.
[17] N. M. PATRIKALAKIS AND T. MAEKAWA, *Shape Interrogation for Computer Aided Design and Manufacturing*, Springer-Verlag, Berlin, 2002.
[18] B. PHILIPPE AND M. SADKANE, *Computation of the fundamental singular subspace of a large matrix*, Linear Algebra Appl. 257 (1997), pp. 77–104.
[19] H. SCHWETLICK AND U. SCHNABEL, *Iterative computation of the smallest singular value and the corresponding singular vectors of a matrix*, Linear Algebra Appl. 371 (2003), pp. 1–30.
[20] T. W. SEDERBERG AND J. ZHENG, *Chapter 15: Algebraic methods for Computer Aided Geometric Design*, in: Farin, G., Hoscheck, J. and Kim, M. S. (Eds.), Handbook of Computer Aided Geometric Design, Elsevier, 2002.
[21] G. STRANG, *Linear Algebra and Its Applications*, 3rd edition, Harcourt Brace Jovanovich, San Diego, 1988.
[22] S. VAN HUFFEL, *Partial singular value decomposition algorithm*, J. Comput. Appl. Math. 33 (1990), pp. 105–112.
[23] S. VAN HUFFEL, J. VANDEWALLE AND A. HAEGEMANS, *An efficient and reliable algorithm for computing the singular subspace of a matrix, associated with its smallest singular values*, J. Comput. Appl. Math. 19 (1987), pp. 313–330.
[24] S. VARADHAN, M. W. BERRY AND G. H. GOLUB, *Approximating dominant singular triplets of large sparse matrices via modified moments*, Numer. Algorithms 13 (1996), pp. 123–152.