Editor's Corner: Gröbner Bases and Matrix Eigenproblems

Robert M. Corless

December 4, 1996

In order to fit into the current 32-page limit on the size of the SIGSAM BULLETIN, I have had to do some compression; in particular I have not been able to comment on the Formally Reviewed Articles in the last issue or on details of upcoming issues. This issue is devoted to the emerging excitement of the connection between polynomial systems and matrix eigenproblems, and I felt that the detailed overview by Manocha and Krishnan and the insightful comments of Stetter deserved this highlighting. In this regard see the call for papers for the special issue of the JSC, printed in the Bulletin no. 117; you may also find it on the new web page for the Bulletin at `http://pineapple.apmaths.uwo.ca/~rmc/sigsam`.

In this column we discuss a Maple routine to construct multiplication matrices for the residue class ring associated with the ideal $\mathcal{I}$ generated by a system of multivariate polynomial equations. We start by assuming that a Gröbner basis for the ideal has already been computed. The approach taken here works for any term ordering used to construct the Gröbner basis, including the total degree ordering. In the lexicographic order case, this approach does not usually gain us anything. The Maple routine we discuss here might be useful, for example, in the case where a total degree basis can be computed but the user is unwilling to wait for a lex order basis to be computed. It is well known that the latter is more difficult to compute.

We restrict attention to the zero-dimensional case, though it is pointed out in [15] that the approach taken here may generalize to higher dimensions through the use of Kronecker canonical forms; see [7] for a discussion of the numerical computation of Kronecker canonical forms.

This column contains very little new material, being really only a discussion of a trial Maple implementation of some of the matrix methods found in [9, 14, 15]. However, the column also contains a proof of the connection between multivariate polynomials and matrix eigenproblems which brings out the fundamental role commutativity plays. For a deeper look at the algebraic structure of this connection, see [16].

All of this material is at least implicit, where it isn't explicit, in the interesting paper [14], and indeed the proofs and algorithms of this present column are equivalent to a proper subset of those in that paper. Nonetheless I hope the slight variations on emphasis here will interest the reader, and stimulate the reader to look at [14] for a full, in-depth treatment. For another expository treatment, containing yet more material, I recommend the very readable papers [9, 10].

Related work includes the many papers of Manocha and co-authors, such as [13], and of course the groundbreaking work [8].

1 Notation and Background

We use the notation of [14]. Let $\mathbb{P}^s$ denote the ring of polynomials in $x_1, \ldots, x_s$ with complex coefficients:

$$\mathbb{P}^s := \mathbb{C}[x_1, \ldots, x_s]\,.$$

Given a finite set $F = \{f_1, \ldots, f_m\} \subset \mathbb{P}^s$, denote the variety of the ideal generated by F as $V(F)$. We assume in this paper that $V(F)$ is a finite set; that is, the ideal $\mathcal{I}$ is zero-dimensional. We assume also that a Gröbner basis G (for some fixed ordering) has been computed for F. Let $\mathrm{lt}(f)$ denote the leading term of f. Let the set of terms (power products) be denoted by

$$T := \{x_1^{i_1} \cdots x_s^{i_s} | i_1, \ldots, i_s \in \mathbb{N}_0\}\,.$$

Define the **normal set**

$$N := \{t \in T |\ \nexists g \in G : \mathrm{lt}(g) \text{ divides } t\}\,.$$

This paper is entirely about computations with this normal set. Knowledge of this set is in some sense fundamental. The most important properties of this set for our purposes are as follows. For proofs of these see e.g. [4].

1. The cardinality of N, say n, is the same as the number of roots of F.

2. N is a basis of the residue class ring $\mathbb{P}^s \bmod \mathcal{I}$: given $p \in \mathbb{P}^s$ we can use Buchberger's NormalForm algorithm (`normalf` in Maple) to find a unique remainder $r \in \mathrm{span}(N)$ such that $p \equiv r \bmod \mathcal{I}$. We denote the residue classes with square brackets: $[p] \equiv [r]$.

3. We can (and want to) specify the basis $N = \{t_1, t_2, \ldots, t_n\}$ by writing down all the monomials

which are not divisible by any term in $\mathrm{lt}(G)$, in some specific order. We will present the details of this computation in section 3.

4. We can describe the effect in $\mathrm{span}(N)$ of multiplication by any fixed $f \in \mathbb{P}^s$ by representing the image of multiplying f by every basis element $[t_i]$ in terms of $[t_1], \ldots, [t_n]$:

$$[f \cdot t_i] = \sum_{j=1}^{n} a_{ij}(f)[t_j] \qquad i = 1, \ldots, n\,.$$

The complex matrix

$$A_f = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$

is called the **multiplication matrix** corresponding to f. Note that this is a *dual* to the so-called *regular matrix representation* [11]. Clearly we can use `normalf` to compute the $a_{ij}(f)$ once we know the basis $\{t_1, \ldots, t_n\}$ of N. In [14] these matrices are called "multiplication tables", but to me a table is the whole set $\{f \cdot g\}$, so here we use 'matrix'.

5. The multiplication matrices for $f = x_1$, $f = x_2$, and so on, are particularly important. Once we know these we can construct the matrices for any polynomial f simply by matrix multiplication and addition. This uses the fact that multiplication by x_i commutes with multiplication by x_j, and hence the matrix M_i that is the multiplication matrix for x_i must commute with M_j. This commutativity, in fact, is central to the entire theory, as we will see.

2 A Theorem

Lemma: If $\{A_i\}_{i=1}^{m}$ is a commuting family of matrices, that is to say $A_iA_j = A_jA_i$ for every pair from the family, then there exists a unitary matrix V such that for all A_i in the family, $U_i = V^*A_iV$ is upper triangular. Note that a single matrix V suffices to upper-triangularize all members of the family at once. The matrix V is not unique: in particular the ordering of the diagonal entries of U_1 might be permuted (but of course in that case the diagonal entries of all the U_i are correspondingly permuted).

Proof: See [12, p. 81]. The proof is not constructive. Construction of V is not conceptually difficult in the exact arithmetic/exact data case, but there are interesting numerical problems in the floating-point/inexact-data case (for a discussion of the difficulties even for symmetric matrices see the beautiful paper [3]). Variants on this lemma are also useful; for example in the real case we may take V to be real orthogonal, and we use another variant for the diagonalizable family case later in this paper. But the use of unitary matrices is of interest for numerical stability reasons, and in the exact arithmetic case it gives a sense of elegance and generality.

Note: I do not yet have an implementation of any algorithm for the computation of V, even in the exact arithmetic case. One of the original points of this column was to point out why it would be useful to have such an implementation; however, P. Gianni, B. Trager, and I may now have a refinement which will allow us to use one Schur factorization of a generic combination of the A_i to quickly and stably give us the same information as V would give us [6].

Theorem: Let $M_1, \ldots, M_s$ be the multiplication matrices for $x_1, \ldots, x_s$. Let V be a unitary matrix such that $V^*M_iV = U_i$ is upper triangular for each $i = 1, 2, \ldots, s$. Denote the diagonal entries of U_i by u^i_{jj}, for $j = 1, 2, \ldots, n$. Then the roots of F, counting multiplicities, are

$$z_j = [x_1^j, x_2^j, \ldots x_s^j] = [u^1_{jj}, u^2_{jj}, \ldots, u^s_{jj}]\,,$$

for $j = 1, 2, \ldots, n$. That is, we may read the roots of F off from the diagonal entries of the U_i, which are the eigenvalues of the M_i.

Proof: $\rightarrow$ to prove all roots are eigenvalues:
Suppose that $z = [z^1, z^2, \ldots, z^s]$ is a root of F. Then

$$f(z)t_i(z) = \sum_{j=1}^{n} a_{ij}(f)t_j(z)$$

where f and the t_j are evaluated at z, because the difference between the left hand side and the right hand side is in the ideal and hence zero after evaluation at z. Denoting the vector of values of the basis of N at z by $B(z) = [t_1(z), \ldots, t_n(z)]$ (clearly $B(z)$ is not the zero vector as $t_1(z) = 1$), we have

$$[A_f - f(z)I]\,B(z) = 0$$

and hence $f(z)$ is an eigenvalue of A_f, and $B(z)$ is a common eigenvector[1]. Thus z^j is an eigenvalue of M_j (just take $f(x_1, \ldots, x_s) = x_j$). ♮

$\leftarrow$ to prove all eigenvalues are roots:
Suppose $p \in \mathbb{P}^s$. Then $p(M_1, M_2, \ldots, M_s)$ makes sense because the matrices commute, and we can evaluate this polynomial on these matrices by a Horner-like rule, for example. We also have $p(M_1, \ldots, M_s) = Vp(U_1, \ldots, U_s)V^*$

[1]Hans Stetter pointed out to me that if z^j is a simple eigenvalue and if each x_ν is in the basis for N, then we can simply read off all the roots from this eigenvector. If some x_ν is missing, then $x_\nu + p(\ldots)$ is in the Gröbner basis and we can still compute the value for x_ν from this eigenvector. This powerful observation is given in [15], and also in the works of Manocha.

because the same V works for all M_j, and e.g. $M_1M_2 = VU_1V^*VU_2V^* = VU_1U_2V^*$.

Now suppose $p \in \mathcal{I}$. Then $[p] = 0$ and so $Vp(U_1, \ldots, U_s)V^* = 0$ or $p(U_1, U_2, \ldots, U_s) = 0$. But a simple computation shows that the diagonal entries of $p(U_1, U_2, \ldots, U_s)$ are

$$p(u^1_{jj}, u^2_{jj}, \ldots, u^s_{jj}) = 0$$

and hence the diagonal entries of the upper triangular matrices are roots of $p \in \mathcal{I}$. $\natural$

Remarks. By the theorem, then, if we have an algorithm to compute such a matrix V, we have an algorithm to compute the roots of F. In exact arithmetic this is not hard, particularly if the roots are simple. We remark that in the simple-root case if numerical computations are used we must also check the *conditioning* of the eigenvalues, for example by computing the left eigenvectors.

If the roots are multiple more care must be taken, and an algorithm equivalent to that of [13] or [14] is necessary. We don't discuss this further here. Indeed, the proof above is not quite complete because we didn't show in the $\to$ case that multiple roots occur as eigenvalues with the right multiplicity (they do).

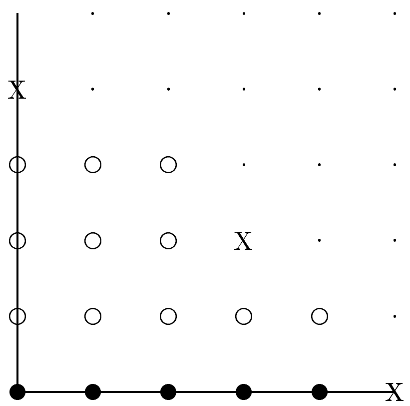
3 Maple code

We assume that we have already computed a `tdeg` (grevlex) Gröbner basis. Given the cost of that computation, minor inefficiencies in the cost of computing the M_i don't matter. However, it seems reasonable to use sparse matrices for the M_i as many of the rows will be simple unit vectors.

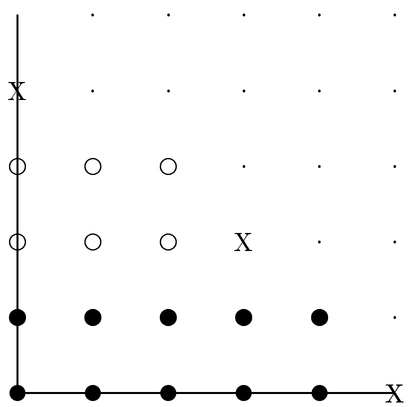
The hardest part of this algorithm (it's still pretty simple, though I am ashamed to admit to how much time it took me) is to get a list of all the terms in N, starting from a list of the leading terms in G.

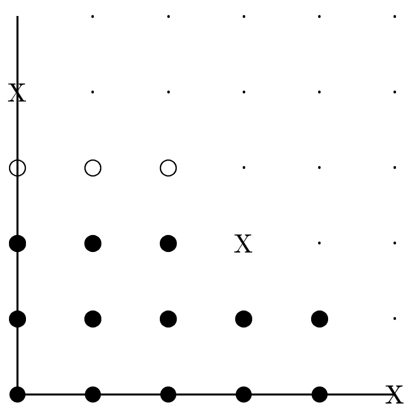
The idea I settled on using (after wasting some time looking for an elegant recursive characterization) is *projection*. This idea is best explained with some two-dimensional pictures, for a concrete example.

Suppose the leading terms in the Gröbner basis are x^5, x^3y^2, y^4. Then we begin with the terms $1, x, x^2, x^3, x^4$. We will work with a graph of the lattice of exponents of these monomials, as usual (see e.g. [4] or [9]). These initial monomials make a line of five dots along the x-axis.

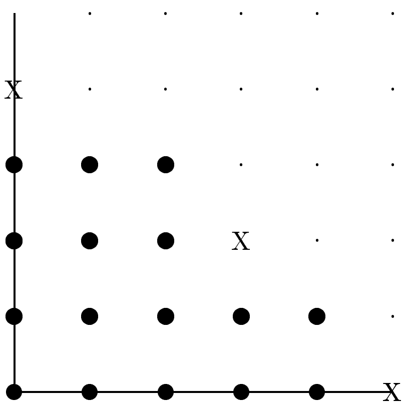


We project this line up onto the terms multiplied by y, and then thereafter to the terms multiplied by y^2.





When we project onto the terms multiplied by y^2 the line is diminished to the terms y^2, xy^2, x^2y^2 by the leading term x^3y^2. We then project this diminished line up two more, to y^4, but the line becomes empty, finishing this dimension.



Now suppose that there is a third variable; we will project, at each iteration, this 2-dimensional polygonal figure, cutting out rectangular regions on hitting divisor terms now instead of cutting out lines. Once this process finishes, we work on the fourth variable, making projections of this three-dimensional 'ziggurat' and removing box-like regions on encountering divisor terms.

The extension to higher dimensions is hard to visualize but is simple in essence. Here is the Maple code to implement it. **Note for Maple V Release 3 users:** you must include the code for the routine `remove`, which is included in Figure 1.

```
remove := proc(f)
option `Copyright (c) 1994 Wissenschaftliches
Rechnen, ETH Zurich. All rights reserved.`;
  select(subs('F' = f, proc() not F(args) end),
  args[2 .. nargs])
end:
```

Figure 1: A necessary update for Maple V Release 3 users

The code for `rembasis` can be found in Figure 2.

I found it convenient to return the basis and a table of pointers indexed by the basis, which makes construction of the matrices easier. The code to actually compute the multiplication matrices can be found in Figure 3. Barry Trager points out that a significant efficiency improvement in the reduction step is to re-use previously computed rows of the M_i whenever possible, as is done in [8]. This is not done here in this naive approach, but this is the first place for improvements.

4 Examples

Example 1: (from [15]) (note there is a typo in the equation for f_2 in [15, p. 359]; $-6x_1^3$ should be $+6x_1^3$)

```
> f1 := 3*x1^2*x2+9*x1^2+2*x1*x2+5*x1+x2-3:
```

```
rembasis := proc( gb, vars, ordering)
  local monomials, nvars, m, i, z, j,im, t, x,
        scope, P, S, rvlex, rev;
  monomials := [seq(grobner[leadmon](gb[i],
            vars, ordering)[2],i=1..nops(gb))];
  nvars := nops(vars);
  rvlex := subs(DY=vars, (a,b) ->
      evalb(a = op(2, sort(a+b, DY, plex))));
  monomials := sort(monomials,rvlex);
  t := monomials[1];
  x := vars[nvars];
  m := degree(t, x);
  # t has to be a pure power of x[k].
  if t <> x^m then
    ERROR(`Sort failed, perhaps there is
           not a finite number of solutions.`)
  fi;
  im := 2;
  S := { seq(x^j, j=0..m-1) };
  P := S; # initial projection
  for i from nvars-1 by -1 to 1 do
    x := vars[i];
    while P <> {} and im <= nops(monomials) do
      t := monomials[im];
      im := im+1;
      m := degree(t,x);
      scope := (p,z) -> p*z;
      for z in {seq(x^j,j=0..m-1)} do
        S := S union map(scope,P,z);
      od;
      P := remove((b,a)->divide(b,a),P,t/x^m);
    od;
    P := S; # reset projection to entire S
            # when the dimension changes
  od;
S := [op(sort(convert(S,`+`),vars,ordering))];
S := [seq(S[nops(S)+1-i],i=1..nops(S))];
rev := table():
for i to nops(S) do
  rev[S[i]] := i;
od:
S,eval(rev)
end:
```

Figure 2: Code to enumerate monomials in the normal set N. The input is 'gb', a grobner basis in 'vars' with the ordering 'ordering'.

```
mulmatrix := proc(x,S,rev,gb,vars,ordering)
  local xw, X, n, i, j, c, t;
  n := nops(S);
  X := array(1..n, 1..n, sparse);
  xw := map( (t,x)->t*x, S, x);
  xw := map(grobner[normalf], xw, gb,
            vars, ordering);
  for i to n do
    c := [coeffs(xw[i], S, 't')];
    t := [t];
    for j to nops(t) do
        X[i,rev[t[j]]] := c[j];
    od;
  od;
eval(X)
end:
```

Figure 3: Code to compute the multiplication matrices. The input is `x`, the variable whose matrix we want, `S`, the list of monomials in N, `rev`, the table of reversed indices, `gb`, the Gröbner basis in `vars` with the ordering `ordering`.

```
> f2:=2*x1^3*x2+6*x1^3-2*x1^2-x1*x2-3*x1-x2+3:
```

```
> f3 := x1^3*x2 + 3*x1^3 + x1^2*x2 + 2*x1^2:
```

```
> with(grobner):
```

```
> gb := gbasis({f1,f2,f3}, [x1, x2],tdeg);
```

$$\mathrm{gb} := [x_1\,x_2 + x_1 - x_2 + 3,\, 2\,{x_1}^2 - 3\,x_1 + 2\,x_2 - 6,\\ -8\,x_1 - 5\,x_2 - 3 + 2\,{x_2}^2]$$

```
> read `rembasis`;
```

```
> rm := rembasis(gb, [x1,x2], tdeg);
```

$$\mathrm{rm} := [1,\, x_2,\, x_1], \mathrm{table}([\\ 1 = 1\\ x_1 = 3\\ x_2 = 2\\ ])$$

Note that rm is an expression sequence with two elements. This means that using it as an argument to a procedure will send two actual arguments to that procedure. This is a feature of Maple that I probably should not use as it makes for less readable code.

```
> m1 := mulmatrix(x1, rm, gb, [x1,x2], tdeg);
```

$$m_1 := \begin{bmatrix} 0 & 0 & 1 \\ -3 & 1 & -1 \\ 3 & -1 & 3/2 \end{bmatrix}$$

```
> m2 := mulmatrix(x2, rm, gb, [x1,x2], tdeg);
```

$$m_2 := \begin{bmatrix} 0 & 1 & 0 \\ 3/2 & 5/2 & 4 \\ -3 & 1 & -1 \end{bmatrix}$$

```
> evalm(m1 &* m2 - m2 &* m1);
```

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Since the roots are simple, here, we may use Maple's `eigenvects` command to find them:

```
> with(linalg):
> e1 := eigenvects(m1);
```

$$e_1 := [\frac{5}{4} + \frac{1}{4}\sqrt{65},\, 1,\, \{\left[1,\, -\frac{3}{4} - \frac{1}{4}\sqrt{65},\, \frac{5}{4} + \frac{1}{4}\sqrt{65}\right]\}],\\ [\frac{5}{4} - \frac{1}{4}\sqrt{65},\, 1,\, \{\left[1,\, -\frac{3}{4} + \frac{1}{4}\sqrt{65},\, \frac{5}{4} - \frac{1}{4}\sqrt{65}\right]\}],\\ [0,\, 1,\, \{[1,\, 3,\, 0]\}]$$

Knowing from the output of `rembasis` that the eigenvectors are of the form $[1, x_2, x_1]$, we can simply read off the roots.

Alternatively, we can use Theorem 1.3.19 from [12] instead of the unitary matrix lemma; this theorem states that a diagonalizable family is simultaneously diagonalizable if and only if it is a commuting family. So by computing a matrix to diagonalize one of the matrices, we can diagonalize them all. A look at the proof of the theorem above shows that we can use $P^{-1}M_iP$ wherever we used V^*M_iV before; this is of course less numerically stable in the floating-point case but we don't care about that here.

Note for Maple V Release 3 users: The `jordan` routine has changed in Release 4 and now returns P such that $P^{-1}AP = J$, whereas in Release 3 it returned P such that $P^{-1}JP = A$. The code below uses Release 4.

```
> j1 := jordan(m1, 'P');
```

$$j_1 := \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{5}{4} + \frac{1}{4}\sqrt{65} & 0 \\ 0 & 0 & \frac{5}{4} - \frac{1}{4}\sqrt{65} \end{bmatrix}$$

Since the matrix P is in this case simply the matrix of eigenvectors, we could read the roots off from it, after normalization of each column. It illustrates the point about simultaneous diagonalization, however, if we compute $P^{-1}m_iP$ instead.

```
> evalm(P^(-1) &* m1 &* P):
```

```
> map(radsimp,",'ratdenom');
```

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{4}\sqrt{5}\sqrt{13} + \frac{5}{4} & 0 \\ 0 & 0 & -\frac{1}{4}\sqrt{5}\sqrt{13} + \frac{5}{4} \end{bmatrix}$$

```
> evalm(P^(-1) &* m2 &* P):

> j2 := map(radsimp,",'ratdenom');
```

$$j_2 := \begin{bmatrix} 3 & 0 & 0 \\ 0 & -\frac{1}{4}\sqrt{5}\sqrt{13}-\frac{3}{4} & 0 \\ 0 & 0 & \frac{1}{4}\sqrt{5}\sqrt{13}-\frac{3}{4} \end{bmatrix}$$

We can verify by direct substitution that the values

$$\begin{aligned} (x_1, x_2) &= (\frac{1}{4}\sqrt{5}\sqrt{13}+\frac{5}{4}, -\frac{1}{4}\sqrt{5}\sqrt{13}-\frac{3}{4}) \\ (x_1, x_2) &= (-\frac{1}{4}\sqrt{5}\sqrt{13}+\frac{5}{4}, \frac{1}{4}\sqrt{5}\sqrt{13}-\frac{3}{4}) \\ (x_1, x_2) &= (0, 3) \end{aligned}$$

satisfy the original polynomials.

Example 2: Consider the polynomial system $F = \{x^2+y^2-1, (3x+4y)^2+(-4x+3y)^2/2-25\}$. A total-degree Gröbner basis for F is

$$[x^2+y^2-1, -16+24\,xy+7\,y^2, -688\,y+625\,y^3+384\,x]$$

and applying `rembasis` gives $N=[1,y,x,y^2]$. Applying `mulmatrix` gives m_x as

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 2/3 & 0 & 0 & -\frac{7}{24} \\ 1 & 0 & 0 & -1 \\ 0 & \frac{216}{625} & \frac{112}{625} & 0 \end{bmatrix}.$$

Maple's `eigenvects` routine tells us that this 4 by 4 matrix has only two eigenvectors, $[-5/3, 4/3, 1, -\frac{16}{15}]^T$ and $[5/3, 4/3, 1, \frac{16}{15}]^T$, each corresponding to an eigenvalue of multiplicity 2. Since there is only one eigenvector for each eigenvalue, it must be the common eigenvector; since here $B(z)=[1,y,x,y^2]^T$ it must be that the (double) roots are $(x,y)=(-3/5,-4/5)$ and $(x,y)=(3/5,4/5)$.

Example 3: Consider $F=\{x^2+y^2-1, x^3+(2+z)xy+y^3-1, z^2-2\}$. The basis for N that we get is

$$[1,z,y,x,yz,y^2,xz,xy,y^2z,y^3,xyz,y^3z]^T$$

so there are 12 roots. There are 4 simple roots and 4 double roots. The double roots corresponding to $x=0$ and $y=1$ are difficult to sort out automatically because these double roots each have two eigenvectors, meaning that we have to form a linear combination to find the common eigenvector for m_x, m_y, and m_z. While this can be done simply enough in this case ($u+\sqrt{2}v$ does the trick with the ordering I got), in general it requires more work. There are two other double roots, each with only a single eigenvector[2]. In one case the eigenvector is

$$[1, \sqrt{2}, -1/\sqrt{2}, -1/\sqrt{2}, \ldots, -1/2]^T$$

which gives $z=\sqrt{2}$, $y=-1/\sqrt{2}$, and $x=-1/\sqrt{2}$. The other one changes the signs of x, y, and z. This example comes from the problem where $f_2=x^3+axy+y^3-1$ where I chose the values of $a=2\pm\sqrt{2}$ (after a discriminant calculation to find these values) so as to force multiple roots.

5 Application to a Flow-Induced Vibration Problem

Consideration of a semi-empirical model for flow-induced vibration gives rise to the following problem [1]. Let

$$\begin{aligned} f_1 = \quad & 4\,Y\nu_1\,\hat{S}\,C-4\,MYC+4\,MXY^2C+4\,M^2g \\ & -3\,M^2Xh+4\,\hat{S}^2g-3\,X\hat{S}^2h-4\,Y^2C^2 \\ & +4\,XY^3C^2+4\,YCMg-3\,XYCMh \end{aligned}$$

and

$$\begin{aligned} f_2 = \quad & 16\,Y^2C^2-32\,XY^3C^2-32\,YCMg \\ & +24\,XYCMh+16\,X^2Y^4C^2+32\,XY^2CMg \\ & -24\,X^2Y^2CMh+16\,M^2g^2-24\,M^2Xgh \\ & +9\,M^2X^2h^2+16\,\hat{S}^2g^2-24\,X\hat{S}^2gh \\ & +9\,X^2\hat{S}^2h^2-16\,\hat{S}^2YC^2 \end{aligned}$$

where the variables we wish to solve for are X and Y and the rest are parameters. (Of course we are aware that certain algebraic relations amongst the parameters may qualitatively change the Gröbner basis we compute.)

We started with a larger system than this one, for which a lex-order Gröbner basis was impossible to compute; we succeeded using total degree. We have since applied reduction techniques to the original system to get the one above, for which a tdeg basis takes only 12 minutes on a 32meg Pentium 90. We find now that a lex order basis for this reduced system is possible to compute, and takes 17 minutes on the same machine; so in part the motivation for using the matrix methods has disappeared, although they played a large role in finding the reduction methods we used, because partial successes with intermediate systems gave us clues as to how to simplify.

We can now write down polynomials giving each of the roots of the system. There is one 'trivial' root (where $Y=0$ and $X=4g/3h$) and 6 other roots. The methods of this column allow us to write down a very efficient program to evaluate X and Y given values for the other parameters.

[2]It is curious that this 'most degenerate' case is the simplest to deal with.

We may also investigate stability and bifurcations of this system. This will be reported on in more detail in an upcoming paper [1].

6 Concluding Remarks

We've seen that computation of a V that simultaneously upper-triangularizes a commuting family of matrices is desirable in general (to handle the multiple-root case when there is more than one eigenvector per eigenvalue) although we can go a long way with simpler techniques. Algorithms for the computation of V in exact arithmetic are in effect known, though not implemented, as far as I know. A numerically stable algorithm for the computation of V in the floating-point case would be nice to have, and to my knowledge there are no efficient algorithms.

Acknowledgements

Thanks go to Anne-Marie E. Allison for carefully reading earlier drafts of this paper and commenting, and for testing the code in Maple V Release 3 and uncovering the incompatibilities mentioned in the text. Hans Stetter, Barry Trager, and Patrizia Gianni also read an earlier draft and gave detailed and valuable advice; finally special thanks to Lalo Gonzalez-Vega for many discussions on this linear algebraic approach and for teaching me how it works.

References

[1] A.-M. E. Allison & R. M. Corless, "Bifurcation Study of a model of Flow-Induced Vibration, part II", *in preparation.*

[2] W. Auzinger and H. J. Stetter, "An Elimination Algorithm for the Computation of All Zeros of a System of Multivariate Polynomial Equations", Numerical Mathematics Singapore 1988, Ravi P. Agarwal, Y. M. Chow, & S. J. Wilson, eds, ISNM vol 86, Birkhäuser, *pp.* 11–30, 1988.

[3] Angelika Bunse-Gerstner, Ralph Byers, and Volker Mehrmann, "Numerical Methods for Simultaneous Diagonalization", SIAM Journal of Matrix Analysis and Applications, Volume 14, Number 4, *pp.* 927–949, 1993.

[4] David Cox, John Little, & Donal O'Shea, *Ideals, Varieties, and Algorithms*, Springer-Verlag, 1992.

[5] Robert M. Corless, Patrizia M. Gianni, Barry M. Trager, & Stephen M. Watt, "The SVD for Polynomial Systems", Proc. ISSAC 95, Montréal, A. H. M. Levelt, editor, *pp.* 195–207, 1995.

[6] Robert M. Corless, Patrizia M. Gianni, Barry M. Trager, "Schur form and generic combinations for multiple roots", *in progress.*

[7] J. W. Demmel & Bo Kågström, "Computing Stable Eigendecompositions of Matrix Pencils", Linear Algebra and its Applications, Volume 88/89, *pp* 139–186, 1987.

[8] J. C. Faugère, P. Gianni, D. Lazard, and T. Mora, "Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering", J. Symbolic Computation, **16**, 1993, *pp.* 329–344.

[9] L. Gonzalez-Vega, "Some examples of problem solving by using the symbolic viewpoint when dealing with polynomial systems of equations" In: Computer Algebra in Science and Engineering (editors J. Fleischer, J. Grabmeier, F. W. Hehl and W. Küchlin), World Scientific, *pp.* 102–116, 1995.

[10] L. Gonzalez-Vega, "Symbolic Recipes for Polynomial System Solving: Real Solutions", *notes for ISSAC '96 tutorial.*

[11] Nathan Jacobson, *Basic Algebra*, 2nd edition, W. H. Freeman, 1989.

[12] Roger Horn & Charles Johnson, *Matrix Analysis*, Cambridge University Press, 1985 (1991 printing used here).

[13] Dinesh Manocha & Shankar Krishnan, "Solving Algebraic Systems using Matrix Computations", *this* BULLETIN, *pp.* 4–21.

[14] H. Michael Möller and Hans J. Stetter, "Multivariate polynomial equations with multiple zeros solved by matrix eigenproblems", Numerische Mathematik, Volume 70, number 3, *pp* 311–330, 1995.

[15] Hans J. Stetter, "Multivariate Polynomial Equations as Matrix Eigenproblems", In: Contributions to Numerical Mathematics, World Scientific Series in Applicable Analysis, Volume 2, *pp.* 355–371, 1993.

[16] Hans J. Stetter, "Matrix Eigenproblems are at the Heart of Polynomial System Solving", this BULLETIN *pp.* 22–25.