Journal of Computational and Applied Mathematics 29 (1990) 129–143
North-Holland

Computation of normal forms

Shui-Nee CHOW *
School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332, USA

Byron DRACHMAN **
Mathematics Department, Michigan State University, Wells Hall, East Lansing, MI 48824, USA

Duo WANG
Institute of Mathematics, Academia Sinica, Beijing 100871, People's Rep. of China

Received 22 October 1987
Revised 9 August 1988

Abstract: In this paper a method of computing a normal form for a system of ordinary differential equations is given. A program using the symbolic manipulator MACSYMA is used on several examples.

Keywords: Differential equations, normal forms, symbolic, MACSYMA, Klein–Gordon.

1. Introduction

One of the basic tools in the study of qualitative behavior of orbit structures of vector fields near equilibria is the theory of normal forms. (See for examples, [1,2,6,9]). It is also well known that it is in general not a simple task to find a normal form for a given equation. Furthermore, normal forms are generally not uniquely defined. This makes computation more complicated.

The purpose of this paper is to present a method for computing normal forms which is simple to use. Our work is related to the recent work of Rand and Keith [8], who give a simple and elegant program to compute a normal form for a differential equation, assuming the "form" of a fixed normal form is already known. However, they do not compute the "form" of the form. You must know this in advance. Our first contribution is to provide a method and computer program to compute the "form" of a fixed normal form. We then show how to compute the normal form for a differential equation, having computed the normal form for the matrix A, and give a computer program.

We begin by giving a discussion of the basic ideas for normal forms. Since our main purpose is to present computer programs that compute normal forms, our discussion corresponds to our computer program and we give a short discussion of the basic ideas.

* Partially supported by DARPA and NSF grant DMS 8401719.
** Partially supported by DARPA.

We point out that in computing the normal form by using a different method and by judicious use of the functions RATCOEF and other functions in the symbolic manipulation system MACSYMA we are able to avoid solving large matrix problems. In the second part of the process, that of computing the normal form for a specific differential equation, we have avoided the necessity of the user having to make many tedious keyboard entries, which in practice we found difficult to do without committing errors. We also note that in our method computing the normal form for the differential equation with three variables at any degree requires solving a 3×3 matrix (whose entries are polynomials) problem, whereas in [9], at degree three a 30×30 matrix problem is supposed to be solved. Although the smaller matrix problem has more complicated entries than the larger (30×30) matrix problem with entries involving unknown constants and numbers, we find that the smaller matrix problem can be solved with MACSYMA.

Different methods for computing normal forms can also be found in [4,5]. The method in [4] is based on representation theory for nilpotent matrices and the method in [5] on a special choice of inner product structure in the vector space of homogeneous polynomials.

2. Normal forms

The basic idea of normal forms theory is to perform a sequence of transformations of a given differential equation, each successive sequence simplifying the terms of the next degree, without affecting the simplified terms of lower degree, so that the end result yields a transformed differential equation in a normal (simple) form which is easier to solve or analyze than the original.

We begin by giving a discussion on normal form theory. In the following, for each $i = 2, 3, \ldots$, $H^i[x_1, \ldots, x_n]$ denotes the vector space of all homogeneous polynomials in the variables $x_1, \ldots, x_n$ of degree i. Let

$$\dot{x} = Ax + f^2(x) + f^3(x) + \cdots + f^m(x) \tag{1}$$

be a given differential equation, where each $f^i(x) \in H^i[x_1, \ldots, x_n]$, $i = 2, 3, \ldots, m$. We assume the matrix A is already in some canonical form. Consider the following sequence of transformations:

$$x \to x + \xi^i(x), \quad i = 2, 3, \ldots, m,$$

where $\xi^i(x) \in H^i[x_1, \ldots, x_n]$. Consider the first transformation which we write as $x = \psi(y) = y + \xi^2(y)$, and $\xi^2(y) \in H^2[y_1, \ldots, y_n]$ is to be chosen later. Substitution into (1) yields

$$\xi_y(y)\dot{y} = A\xi(y) + f^2(\xi(y)) + f^3(\xi(y)) + \cdots, \tag{2}$$

where $\xi_y(y)$ denotes the (Jacobian) derivative of $\xi(y)$. Multiply both sides by $\xi_y(y)^{-1}$ to get

$$\dot{y} = \big(\xi_y(y)\big)^{-1} A\xi(y) + \big(\xi_y(y)\big)^{-1} f^2(\xi(y)) + \cdots. \tag{3}$$

Since

$$\psi(y) = y + \xi^2(y), \quad \psi(0) = 0 \text{ and } \psi_y(0) = I,$$

we have

$$\psi_y^{-1}(\psi(y)) = I + \mathrm{O}(|y|) \quad \text{as } |y| \to 0.$$

Thus

$$\xi_y^2(y)\dot{y} = \xi_y^2(y)Ay + \mathrm{O}(|y|^3).$$

Now returning to (2),

$$\big(I + \xi_y^2(y)\big)\dot{y} = Ay + A\xi^2(y) + f^2(y) + \mathrm{O}(|y|^3),$$

so

$$\dot{y} = Ay + f^2(y) + A\xi^2(y) - \xi_y^2(y)Ay + \mathrm{O}(|y|^3).$$

Now define the linear transformation

$$L_A^2 : H^2[y_1, \ldots, y_n] \to H^2[y_1, \ldots, y_n]$$

by

$$L_A(p^2(y)) = p_y^2(y)Ay - Ap^2(y)$$

for all $p^2(y) \in H^2[y_1, \ldots, y_n]$.
Then

$$\dot{y} = Ay + f^2(y) - L_A^2(\xi^2(y)) + \mathrm{O}(|y|^3). \tag{4}$$

The method is to choose $\xi^2(y)$ such that the term in degree two, $f^2(y) - L_A(\xi^2(y))$ is in a simple form. In other words, let $\mathscr{R}^2 = \text{range}\,(L_A^2)$. We will choose a complementary space $\mathscr{C}^2$ of $\mathscr{R}^2$ in $H^2[y_1, \ldots, y_n]$ and decompose $f^2(y) = \bar{f}^2(y) + g^2(y)$ where $\bar{f}^2(y) \in \mathscr{R}^2$ and $g^2(y) \in \mathscr{C}^2$. Then we will choose $\xi^2(y)$ so that $L_A^2(\xi^2(y)) = \bar{f}^2(y)$. With this choice (4) becomes

$$\dot{y} = Ay + g^2(y) + \mathrm{O}(|y|^3). \tag{5}$$

Our goal is to devise a computer program that will choose a complementary space $\mathscr{C}^2$ so that $g^2(y)$ has as few nonzero components as possible, namely, the number of nonzero components will be equal to the codimension of $\mathscr{R}^2$ in H^2. Now rewrite the transformed equation (5) using x's

$$\dot{x} = Ax + g^2(x) + f^3(x) + \cdots, \tag{6}$$

where $g^2(x)$ is the same as just discussed, but $f^3(x)$ is not necessarily the same as in the original differential equation (1).

The general step is similar: having performed $m-1$ transformations so that the transformed equation is

$$\dot{x} = Ax + g^2x + \cdots + g^{m-1}(x) + f^m(x) + \cdots. \tag{7}$$

Solve the linear system

$$L_A^m \xi^m(y) = f^m(y) \tag{8}$$

for (nonunique) $\xi^m(y) \in H^m(y)$, where $\bar{f}^m(y)$ is the projection of $f^m(y)$ onto the range of L_A^m, and $L_A^m : H^m[x_1, \ldots, x_n] \to H^m[x_1, \ldots, x_n]$ is defined by

$$L_A^m(p^m(x)) = p_x^m(x)Ax - Ap^m(x). \tag{9}$$

One continues the process up to a desired degree, say k. The final form is

$$\dot{x} = Ax + g^2x + \cdots + g^k(x) + \mathrm{O}(|x|^{k+1}). \tag{10}$$

Equation (8) is called a normal form for (1). Note that since the complementary subspaces $\mathscr{C}^2$, $\mathscr{C}^3$, ... are not uniquely defined, normal forms of (1) are not unique.

3. Choice of bases for $\mathscr{R}^k$ and $\mathscr{C}^k$

Take the basis for $H^k[x_1, \ldots, x_n]$:

$$\begin{pmatrix} x_1^k \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad \begin{pmatrix} x_1^{k-1}x_2 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ x_n^k \end{pmatrix}. \tag{11}$$

The linear transformation L_A^k is first represented in this basis. The matrix is column reduced (the transpose is row reduced) and put into upper echelon form. The columns of the reduced matrix give a basis for the complement of the range $\mathscr{R}^k$ of L_A^k, and the zero columns determine a complementary subspace $\mathscr{C}^k$.

4. Normal forms of differential equations

Now that we have computed the normal form for the matrix A, we show how to compute the normal form for a given differential equation. We suppose that (1) is given.

By using the matrix A from the linear term in the differential equation, we compute bases for the complementary subspaces $\mathscr{C}^i$, $i = 2, \ldots, k$. We now know the "form" of the normal form for the differential equation for degree 2, 3, ..., k, in the sense that after a sequence of substitutions $x = y + \xi^i(y)$, $i = 2, 3, \ldots, k$, the final form of the differential equation will be as in (8). However, up to now we merely know that each $g^i(x) \in \mathscr{C}^i$, i.e., that each $g^i(x)$ is a linear combination of the chosen basis elements. We do not yet know the coefficients in the expansion of $g^i(x)$.

One approach would be to write $\xi^i(x)$ as a linear combination, with unknown coefficients, of the basis elements $H^i[x_1, \ldots, x_n]$ and then to try to solve for the coefficients of $\xi^i(x)$ by making the substitution into the differential equation and then using the fact that the terms in degree i must lie in $\mathscr{C}^i$. Once the coefficients are known, one would proceed to the next degree. In relatively simple examples, the computer ran out of memory trying these steps, since large matrix problems and very complicated "formal substitutions" were attempted.

Instead, at each degree i, let $f^i(x)$ be the component of the right-hand side of the transformed differential equation up to that point in the process. Since we have already computed a basis for $\mathscr{C}^i$, it is a simple matter to decompose

$$f^i(x) = \bar{f}^i(x) + g^i(x). \tag{12}$$

Thus we find the component $g^i(x)$ before we attempt to compute $\xi^i(x)$. Now find $\xi^i(x)$ by solving

$$L_A^i(\xi^i(x)) = \bar{f}^i(x) \quad \text{for } \xi^i(x). \tag{13}$$

Once $\xi^i(x)$ has been explicitly computed, one now substitutes into the differential equation. Then form $\psi(x) = x + \xi^i(x)$. Following (3), we have the new differential equation, after rewriting in terms of x,

$$\dot{x} = (\psi_x(x))^{-1}(A\psi(x) + f^2(\psi(x)) + \cdots). \tag{14}$$

This involves solving a 3×3 matrix problem, with somewhat messy entries (polynomials). Then one proceeds to the next degree. In our main example we will stop at degree 4. Note that in order to get $g^4(x)$ we only need the right-hand side of the differential equation (just formed) and the basis elements for $\mathscr{C}^4$, $\mathscr{R}^4$. We do not need to solve for $\xi^4(x)$. In Section 6 we will illustrate our method with some examples.

5. Outlines of the computer programs

Program 1. Compute the normal form for the given matrix A.

(a) Set the number of variables (nvar), create A, and form the vector

$$\begin{bmatrix} x_1 \\ \vdots \\ x_{\text{nvar}} \end{bmatrix}.$$

For each degree k from 2 to 4:

(b) Compute a basis for $H^k[x_1, \ldots, x_{\text{nvar}}]$ (getbas ( )).

(i) Expand $(x_1 + \cdots + x_{\text{nvar}})^k$ (cp).

(ii) Extract the monomial terms (part (cp, i)).

(iii) Normalize the basis (den[i]).

(iv) Form basis elements

$$\text{hbasis}(1) = \begin{pmatrix} x_1^k \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \text{hbasis(ndimbas)} = \begin{pmatrix} 0 \\ \vdots \\ x_{\text{nvar}}^k \end{pmatrix}.$$

(c) Make the matrix c representing

$$L_A^k(\xi(y)) = \xi_y(y)Ay - A\xi(y) \quad \text{(makelv( ))}.$$

(d) Put the matrix into standard form (stdform ( )).

(e) Write the bases to disk.

Program 3. Compute the normal form for a differential equation.

(a) Enter number of variables (nvar) and the right-hand side of the differential equation by entering the appropriate coefficients co[i] for the basis elements hbasis[i] loaded from the file created in Program 1.

(b) For $k = 2$ to 4:

(i) Decompose $f^k(x) = \bar{f}^k(x) + g^k(x)$ where $g^k(x)$ is the component of $f^k(x)$ in the chosen complementary space $\mathscr{C}^k$ (get gbar ( )).

(ii) Solve $L_A^k(\xi^k(x)) = \bar{f}^k(x)$ for $\xi^k(x)$ (get xsi ( )).

(iii) Substitute $x + \xi^k(x)$ for x into the differential equation (makesubst ( )).

6. Examples

In this section we will consider several examples.

Example 1. Consider the following equations:

$$\begin{aligned} \dot{x}_1 &= -x_2 - (x_3 - x_1)^2, \\ \dot{x}_2 &= x_1, \\ \dot{x}_3 &= -(x_3 - x_1)^2. \end{aligned} \tag{15}$$

This equation is related after some change of variables to an equation in [7]. We are actually interested in $\dot{r}$ and $\dot{x}_3$ where (x_1, x_2) is put into polar coordinates $re^{i\theta}$. Before using our program to find the normal form for the matrix A, we substitute

$$\begin{aligned} z &= x_1 + \mathrm{i}x_2, \qquad \bar{z} = x_1 - \mathrm{i}x_2, \\ x_1 &= \frac{z+\bar{z}}{2}, \qquad x_2 = \frac{z-\bar{z}}{2\mathrm{i}}, \end{aligned} \tag{16}$$

and compute $\dot{z}$, $\dot{\bar{z}}$, and $\dot{x}_3$. The above is easily done with paper and pencil. We used a MACSYMA program to avoid careless errors. This "scratch work" is dignified with the name Program 2. The listing and its output are in the Appendix. Until our final step, x_1 now stands for z and x_2 stands for $\bar{z}$. Our differential equation to be normalized is then

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{pmatrix} = \begin{pmatrix} \mathrm{i} & 0 & 0 \\ 0 & -\mathrm{i} & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} -\frac{1}{4}x_1^2 - \frac{1}{2}x_1x_2 + x_3x_1 + x_3x_2 - \frac{1}{4}x_2^2 - x_3^2 \\ -\frac{1}{4}x_1^2 - \frac{1}{2}x_1x_2 + x_3x_1 + x_3x_2 - \frac{1}{4}x_2^2 - x_3^2 \\ -\frac{1}{4}x_1^2 - \frac{1}{2}x_1x_2 + x_3x_1 + x_3x_2 - \frac{1}{4}x_2^2 - x_3^2 \end{pmatrix}. \tag{17}$$

We use Program 1 to compute the normal form for the matrix A for this differential equation up to degree four and Program 3 to compute the normal form for the differential equation (15).

The output of the program is right after the program listing in the Appendix. In order to convince the reader that this program does a computation that would be no fun to do by hand, we note that the print-out of $(\psi_x(x))^{-1}$, where $\psi(x) = x + \xi^3(x)$, is nine pages long. We recall that x_1 stands for z and x_2 stands for $\bar{z}$, and we want $\dot{r}$ and $\dot{x}_3$. This is not difficult to do by hand, but it is even easier to use MACSYMA. This "scratch work" is called Program 4. It, along with its output, is also in the Appendix. Thus our normal form for the differential equation (15) is

$$\begin{aligned} \dot{r} &= rx_3 - \tfrac{153}{36}x_3r + \tfrac{43}{36}x_3r^3, \\ \dot{x}_3 &= -\tfrac{1}{2}r^2 - x_3^2 - \tfrac{271}{288}r^4 + 9x_3^2r^2 + 4x_3^4. \end{aligned} \tag{18}$$

This concludes the discussion of the main example.

Example 2. We note that the normal forms for the examples in [3] can all be done with Program 1. For example, to compute the normal form for the matrix

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{19}$$

in [3, case(1), p.13], we use Program 1 with the above entered for A, and the number of variables

set to two (nvar = 2), and stop at degree 5. The output is called Output 4 and is in the Appendix using this output:

$$\begin{aligned}\dot{x} = Ax + a\begin{pmatrix}0\\ x_1^2\end{pmatrix} + b\begin{pmatrix}0\\ x_1x_2\end{pmatrix} + c\begin{pmatrix}0\\ x_1^2x_2\end{pmatrix} + d\begin{pmatrix}0\\ x_1^3\end{pmatrix}\\ + e\begin{pmatrix}0\\ x_1^4\end{pmatrix} + f\begin{pmatrix}0\\ x_1^3x_2\end{pmatrix} + g\begin{pmatrix}0\\ x_1^4x_2\end{pmatrix} + h\begin{pmatrix}0\\ x_1^5\end{pmatrix},\end{aligned} \tag{20}$$

where $a, \ldots, h$ are constants.

Example 3. Finally, we point out that our Program 1 derives a result in [9] (see also [8]). Equations [8; (25),(26)] are

$$\begin{aligned}\dot{x}_1 &= x_2 + f(x_1, x_2),\\ \dot{x}_2 &= -x_1 + g(x_1, x_2),\end{aligned} \tag{21}$$

where f, g are of order at least 2 in x_1 and x_2. Under the substitution $z = x_1 + \mathrm{i}x_2$, $\bar{z} = x_1 - \mathrm{i}x_2$, (21) converts to

$$\begin{aligned}\dot{z} &= -\mathrm{i}z + f\left(\frac{z+\bar{z}}{2}, \frac{z-\bar{z}}{2\mathrm{i}}\right) + \mathrm{i}g\left(\frac{z+\bar{z}}{2}, \frac{z-\bar{z}}{2\mathrm{i}}\right),\\ \dot{\bar{z}} &= \mathrm{i}\bar{z} + f\left(\frac{z+\bar{z}}{2}, \frac{z-\bar{z}}{2\mathrm{i}}\right) - \mathrm{i}g\left(\frac{z+\bar{z}}{2}, \frac{z-\bar{z}}{2\mathrm{i}}\right).\end{aligned} \tag{22}$$

Thus the matrix A is

$$\begin{pmatrix}-\mathrm{i} & 0\\ 0 & \mathrm{i}\end{pmatrix}. \tag{23}$$

We use Program 1 with this value of A. The output is called Output 5 and is in the Appendix. In the printout, x_1 stands for z, x_2 stands for $\bar{z}$. Thus the normal form for (21) will be

$$\begin{aligned}\dot{z} &= -\mathrm{i}z + bz^2\bar{z} + cz^3\bar{z}^2,\\ \dot{\bar{z}} &= \mathrm{i}\bar{z} + az\bar{z}^2 + dz^2\bar{z}^3.\end{aligned} \tag{24}$$

In the first equation of (24), substitute $z = re^{\mathrm{i}\theta}$, $b = b_1 + \mathrm{i}b_2$, $c = c_1 + \mathrm{i}c_2$. Taking real and imaginary parts yields

$$\begin{aligned}\dot{r} &= b_1r^3 + c_1r^5 + \cdots,\\ \dot{\theta} &= -1 + b_2r^2 + c_2r^4 + \cdots.\end{aligned} \tag{25}$$

Appendix: program listings and outputs

PROGRAM 1

```
loadprint:false$

/* getbas() is a subroutine to get a basis for the v. space of
homog. polynomials of degree=deg in
the variables x(1),...,x(nvar)
global variables used:
   nvar=      number of variables in x
   deg=       degree
   ndimbas=   the dimension of the vector space of homog. polynomials in
              the variables x(1),...,x(nvar) of degree = deg.
   len=       length of (x(1)+...+x(nvar))**deg when expanded (no. of terms).
   vful=      list of x(1)**deg,x(1)**deg-1*x(2),...,x(nvar)**deg.
   hbasis=    the basis elements of the v. space of hom. polynomials in
              degree=deg                                                  */

getbas():=block(
kill(x),
b:sum(x[i],i,1,nvar),
cp:expand(b**deg),
if nvar=1 then
     ( len:1,vful[1]:cp)
else ( len:length(cp),
       for i thru nvar do eqn[i]:x[i]=1,
       eqns2:makelist(eqn[i],i,1,nvar),
       for i thru len do  den[i]:subst(eqns2,part(cp,i)),
       for i thru len do vful[len-i+1]:part(cp,i)/den[i]      ),
ndimbas:nvar*len,
for i thru nvar do (
  for j thru len do (
    hbasis[(i-1)*len+j]:vful[j]*col(ident(nvar),i)     ))
                                                        )$

/* makelv is a subroutine to form the matrix representing the operator
L with respect to the basis elements hbasis(1),...,hbasis(ndimbas)
global variables used:
hbasis=basis elements of h
len=length of (x(1)+...+x(nvar))**deg
ndimbas=dimension of h
c=matrix representing L                                               */

makelv():=block(
kill(x),
for k thru ndimbas do (b3[i,j]:=diff(hbasis[k][i,1],x[j]),
grad:genmatrix(b3,nvar,nvar),
lv[k]:grad.a.xbas-a.hbasis[k],
remarray(grad),
kill(b3)),
remarray(c),
remarray(la),
kill(c),
/* form the columns of the matrix c representing L */
for k thru ndimbas do (
   for n thru nvar do (
      for i thru len do  (
             c[k,(n-1)*len+i]:ratcoef(lv[k][n,1],vful[i])  ))))$

/* pivot is a utility to move (pivot) zero-columns to the right in
the m by m matrix cech(it is applied to c which is in echelon form) */

pivot(cech,m):=block(
for i thru m do ipvt[i]:i,
i:0,
cont1,
i:i+1,
```

```
if i=m+1 then go(jumpout),
if cech[i,i]#0 then go(cont1),
j:i, if j=m then go(jumpout),
cont2,
j:j+1, if j=m+1 then go(cont1),
if cech[i,j]=0 then go(cont2),
(for L thru m do
(temp:cech[L,i], cech[L,i]:cech[L,j],cech[L,j]:temp)),
temp:ipvt[i],ipvt[i]:ipvt[j],ipvt[j]:temp,
go(cont1),
jumpout)$

/* stdform puts the matrix into standard form
global variables used:
c=       the matrix representing L
cech=    the matrix echelon reduced,and after calling pivot(cech,ndimbas)
         it is in standard form
nrankc=  the rank of the matrix c
nulrank= dimension of the complementary space
ndimbas= dimension of h (=homog. polynomials of degree=deg)
                                                                    */
stdform():=block(remarray(cech),cech:echelon(c),
                 pivot(cech,ndimbas),nulrank:ndimbas-nrankc)$

/* main program starts here */

block(
nvar:3,
a:matrix([%i,0,0],[0,-%i,0],[0,0,0]),
print("a=",a),
xbas:genmatrix(u,nvar,1),
for i thru nvar do xbas:subst(u[i,1]=x[i],xbas),
for deg from 2 thru 4 do (
  getbas(),
  makelv(),
  print("  for degree",deg,":"),
  c:genmatrix(c,ndimbas,ndimbas),
  la:transpose(c),
  nrankc:rank(c),
  if nrankc=ndimbas then (nog:-1,print("no terms")) else nog:0,
  if nrankc#ndimbas then (
  stdform(),
  for k thru nulrank do w[k]:hbasis[ipvt[ndimbas-k+1]],
    for k thru nulrank do print("w(",k,")=",w[k]),

    /* if the matrix L happens to be diagonal, set diagl=-1 for
    future savings in computations                        */

    diagl:-1,
    for i thru ndimbas do (
      for j thru ndimbas do (
          if i#j and la[i,j]#0 then diagl:0  )),
    if diagl=-1 then ( for i thru ndimbas do sv[i]:la[i,i],
                   remarray(la),kill(la),
                   for i thru ndimbas do la[i]:sv[i])  ),
  for k thru nrankc do (
    for j thru ndimbas do u[k,ipvt[j]]:cech[k,j]),
  for k thru nrankc do rangebas[k]:sum(u[k,j]*hbasis[j],j,1,ndimbas),
  if deg=2 then save([wfile2],la,ndimbas,vful,hbasis,ndimbas,diagl,w,
                   nulrank,a,xbas,nog,rangebas),
  if deg=3 then save([wfile3],la,ndimbas,vful,hbasis,ndimbas,diagl,w,
                   nulrank,nog,rangebas),
  if deg=4 then save([wfile4],la,ndimbas,vful,hbasis,ndimbas,diagl,w,
                   nulrank,nog,rangebas), endloop)                    );
quit();
```

Output (Edited) of Program 1:

for degree 2:

$$w(1) = \begin{bmatrix} 0 \\ 0 \\ x_3^2 \end{bmatrix} \quad w(2) = \begin{bmatrix} 0 \\ 0 \\ x_1 \;\; x_2 \end{bmatrix} \quad w(3) = \begin{bmatrix} 0 \\ x_2 \;\; x_3 \\ 0 \end{bmatrix} \quad w(4) = \begin{bmatrix} x_1 \;\; x_3 \\ 0 \\ 0 \end{bmatrix}$$

for degree 3:

$$w(1) = \begin{bmatrix} 0 \\ 0 \\ x_3^3 \end{bmatrix} \quad w(2) = \begin{bmatrix} 0 \\ x_1 \;\; x_2^2 \\ 0 \end{bmatrix} \quad w(3) = \begin{bmatrix} 0 \\ x_2 \;\; x_3 \\ 0 \end{bmatrix} \quad w(4) = \begin{bmatrix} x_1 \;\; x_3^2 \\ 0 \\ 0 \end{bmatrix}$$

$$w(5) = \begin{bmatrix} 0 \\ 0 \\ x_1 \;\; x_2 \;\; x_3 \end{bmatrix} \quad w(6) = \begin{bmatrix} x_1^2 \;\; x_2 \\ 0 \\ 0 \end{bmatrix}$$

for degree 4:

$$w(1) = \begin{bmatrix} 0 \\ 0 \\ x_3^4 \end{bmatrix} \quad w(2) = \begin{bmatrix} 0 \\ 0 \\ x_1^2 \;\; x_2^2 \end{bmatrix} \quad w(3) = \begin{bmatrix} x_1^2 \;\; x_2 \;\; x_3 \\ 0 \\ 0 \end{bmatrix} \quad w(4) = \begin{bmatrix} x_1 \;\; x_3^3 \\ 0 \\ 0 \end{bmatrix}$$

$$w(5) = \begin{bmatrix} 0 \\ 0 \\ x_1 \;\; x_2 \;\; x_3^2 \end{bmatrix} \quad w(6) = \begin{bmatrix} 0 \\ x_1 \;\; x_2^2 \;\; x_3 \\ 0 \end{bmatrix} \quad w(7) = \begin{bmatrix} 0 \\ x_2 \;\; x_3^3 \\ 0 \end{bmatrix}$$

Program 2:

```
eq[1]:zdot    = -x[2] - (x[3] - x[1])**2 + %i*x[1]
eq[2]:zbardot = -x[2] - (x[3] - x[1])**2 - %i*x[1];
eq[3]:x3dot   = -(x[3] - x[1])**2;
eqs:makelist(eq[i],i,1,3);
xeq:[x[1] = (z + zbar)/2, x[2] = (z - zbar)/(2*%i)];
neweqns:expand(subst(xeq,eqs));
quit();
```

The output (edited):

$$(d1) \qquad \text{zdot} = -(x_3 - x_1)^2 - x_2 + \%i\, x_1$$

$$(d2) \qquad \text{zbardot} = -(x_3 - x_1)^2 - x_2 - \%i\, x_1$$

$$(d3) \qquad \text{x3dot} = -(x_3 - x_1)^2$$

$$(d5) \qquad [x_1 = \frac{\text{zbar} + z}{2},\ x_2 = -\frac{\%i\,(z - \text{zbar})}{2}]$$

$$(d6) \qquad [\text{zdot} = -\frac{\text{zbar}^2}{4} - \frac{z\ \text{zbar}}{2} + x_3\ \text{zbar} - \frac{z^2}{4} + x_3 z + \%i\, z - x_3^2,$$

$$\text{zbardot} = -\frac{\text{zbar}^2}{4} - \frac{z\ \text{zbar}}{2} + x_3\ \text{zbar} - \%i\ \text{zbar} - \frac{z^2}{4} + x_3 z - x_3^2,$$

$$\text{x3dot} = -\frac{\text{zbar}^2}{4} - \frac{z\ \text{zbar}}{2} + x_3\ \text{zbar} - \frac{z^2}{4} + x_3 z - x_3^2]$$

PROGRAM 3

```
/* gethom is a subroutine to get the terms of degree=deg from a
    given expression.  It is applied to rh=right hand side of a d.e.
    global variables used:
      deg=      the degree
      rh=       the expression (right hand side of the d.e.)
      hbasis=   basis for v.s. of homogeneous polynomials of degree=deg
      ndimbas=  dimension of above v.s.
      r =       component of rh of degree=deg                */

gethom():=block(
  for i thru ndimbas do (
    for j thru nvar do(
      if hbasis[i][j,1]#0 then coe[i]:subst(xq,ratcoef(expand(rh[j,1]),
      hbasis[i][j,1])) )),
  r:sum(coe[i]*hbasis[i],i,1,ndimbas),
  print("from gethom,r=",r)                )$

/* getgbar is a subroutine to get the component of the right hand
    side of the d.e. in the range of the operator L
    global variables used:
      nvar=       number of variables
      r-          component of r.h.s. of d.e. gotten from gethom
      deg=        degree
      w=          basis for complementary space
      rangebas=   basis for the range of L
      nulrank=    dimension of complementary space
      ndimbas=    dimension of v.s. of hom. polynomials of degree=deg
      vful=       x(1)**deg,...,x(nvar)**deg
      fbr=        component of r in the range of L
      gbr=       component of r in the complementary space
    local variable:
      nrankc=     rank of the operator L=rank of the matrix c in Program 1
      dd,co=      coefficients of r in terms of new basis {w,rangebas}
      len=        ndimbas/nvar= number of terms in vful
                                                               */
getgbar():=block(nrankc:ndimbas-nulrank,
                 len:ndimbas/nvar,
                 kill (co,dd,eq),
                 for j thru nvar do (
                 rformal[j]:sum(co[i]*rangebas[i][j,1],i,1,nrankc)
                          +sum(dd[i]*w[i][j,1],i,1,nulrank),
                 q[j]:expand(rformal[j]-r[j,1])  ),
                 q[j]:expand(rformal-r),
                 neq:0,
```

```
                for i thru nvar do (
                  for j thru len do (
                    t:ratcoef(q[i],vful[j]),
                    if t#0 then ( neq:neq+1,eq[neq]:t)  )),
                eqns:makelist(eq[i],i,1,neq),
                unks:makelist(co[i],i,1,nrankc),
                unks:append(unks,makelist(dd[i],i,1,nulrank)),
                ans:solve(eqns,unks),
                for i thru nrankc do co[i]:rhs(part(ans,1,i)),
                for i thru nulrank do dd[i]:rhs(part(ans,1,nrankc+i)),
                fbr:sum(co[i]*rangebas[i],i,1,nrankc),
                gbr:sum(dd[i]*w[i],i,1,nulrank)
                                                              )$

nulleqns:[%r1=0,%r2=0,%r3=0,%r4=0,%r5=0,%r6=0,%r7=0,%r8=0,%r9=0,
%r10=0,%r11=0,%r12=0,%r13=0,%r14=0,%r15=0,%r16=0,%r17=0,%r18=0,
%r19=0,%r20=0,%r21=0,%r22=0,%r23=0,%r24=0,%r25=0,%r26=0,%r27=0,
 %r28=0,%r29=0,%r30=0,%r31=0,%r32=0,%r33=0,%r34=0,%r35=0,%r36=0]$

/* enter number of variables and right hand side of d.e. using
basis elements stored in file from first program and appropriate
coefficients co[i] */

block( nvar:3,
for i thru nvar do xeqn[i]:x[i]=0,
xq:makelist(xeqn[i],i,1,nvar),
deg:2,
loadfile(wfile2),
rh:a.xbas,
temp:-x[1]**2/4-x[1]*x[2]/2+x[3]*x[1]+x[3]*x[2]-x[2]**2/4-x[3]**2,
rh[1]:rh[1]+temp,
rh[2]:rh[2]+temp,
rh[3]:rh[3]+temp,
print("rhs of original de="),
print(rh)                          )$

block(
for deg from 2 thru 4 do (
  gethom(),
  getgbar(),
  g[deg]:gbr,
  if deg=4 then ( print("after subst's xi[1],..,xi[4], rhs(de)="),
                  print(a.xbas+sum(g[i],i,2,deg)), quit() ),
  print("g[",deg,"]=",g[deg]),
  fbar[deg]:fbr,
 print("fbar[",deg,"]=",fbar[deg]),
  for i thru ndimbas do (
    for j thru nvar do (
      if hbasis[i][j,1]#0 then
         fbcol[i]:subst(xq,ratcoef(expand(fbar[deg][j,1]),hbasis[i][j,1])))),

  if diagl=-1 then  (
                  for i thru ndimbas do (
                    if la[i]#0 then
                       xicol[i]:fbcol[i]/la[i]
                    else
                       xicol[i]:0)),
  if diagl=0 then (
    for i thru ndimbas do
      eqns[i]:sum(la[i,j]*xicol[j],j,1,ndimbas)=fbcol[i],
    ans:solve(makelist(eqns[i],i,1,ndimbas),makelist(xicol[i],i,1,ndimbas)),
    ans:subst(nulleqns,ans),
    print("ans="),
    print(ans),
    save([temp],all),
    for i thru ndimbas do xicol[i]:rhs(part(ans,1,i))),
  xi[deg]:sum(xicol[i]*hbasis[i],i,1,ndimbas),
  xsi:xbas+xi[deg],
```

```
  kill(b4),
  b4[i,j]:=diff(xsi[i,1],x[j]),
  gradxsi:genmatrix(b4,nvar,nvar),
  print("gradxsi=",gradxsi),
  gradinv:invert(gradxsi),
  fy:subst([x[1]=y[1],x[2]=y[2],x[3]=y[3]],rh),
  fxi:subst([y[1]=xsi[1,1],y[2]=xsi[2,1],y[3]=xsi[3,1]],fy),
  fxit:taylor(fxi,makelist(x[i],i,1,3),0,4),
  rh:taylor(gradinv.fxit,makelist(x[i],i,1,nvar),0,4),
  print("after substitution ,new rhs of de up to order 4 =  "),
  print(rh),
  kill(la,ndimbas,hbasis,ndimbas,w,nulrank,xicol,gradxsi),
  if deg=2 then loadfile(wfile3),
  if deg=3 then loadfile(wfile4),
enddoloop                                                    )
                                                             )$
quit();
```

Output 3 (Edited)

rhs of original de =

$$\begin{bmatrix} -x_3^2 + x_2x_3 + x_1x_3 - \frac{x_2^2}{4} - \frac{x_1x_2}{2} - \frac{x_1^2}{4} + \%i\,x_1 \\ -x_3^2 + x_2x_3 + x_1x_3 - \frac{x_2^2}{4} - \frac{x_1x_2}{2} - \%i\,x_2 - \frac{x_1^2}{4} \\ -x_3^2 + x_2x_3 + x_1x_3 - \frac{x_2^2}{4} - \frac{x_1x_2}{2} - \frac{x_1^2}{4} \end{bmatrix}$$

$$g[2] = \begin{bmatrix} x_1 & x_3 \\ x_2 & x_3 \\ -x_3^2 - \frac{x_1x_2}{2} \end{bmatrix}$$

$$\text{'fbar}[2] = \begin{bmatrix} -x_3^2 + x_2x_3 - \frac{x_2^2}{4} - \frac{x_1x_2}{2} - \frac{x_1^2}{4} \\ -x_3^2 + x_1x_3 - \frac{x_2^2}{4} - \frac{x_1x_2}{2} - \frac{x_1^2}{4} \\ x_2x_3 + x_1x_3 - \frac{x_2^2}{4} - \frac{x_1^2}{4} \end{bmatrix}$$

$\vdots$

(11 pages of printout)

$\vdots$

after subst's $xi[1], \ldots, xi[4]$, rhs(de) =

$$\begin{bmatrix} -\frac{17\,x_1x_3^3}{4} + \frac{3\,\%i\,x_1x_3^2}{2} + \frac{41\,x_1^2x_2x_3}{36} + x_1x_3 - \frac{\%i\,x_1^2x_2}{24} + \%i\,x_i \\ -\frac{17\,x_2x_3^3}{4} + \frac{3\,\%i\,x_2x_3^2}{2} + \frac{41\,x_1x_2^2x_3}{36} + x_2x_3 - \frac{\%i\,x_1x_2^2}{24} - \%i\,x_2 \\ 4\,x_3^4 + \frac{9\,x_1x_2x_3^2}{8} - x_3^2 - \frac{271\,x_1^2x_2^2}{288} - \frac{x_1x_2}{2} \end{bmatrix}$$

PROGRAM 4

```
de[1]:-17*x[1]*x[3]**3/4+3*%i*x[1]*x[3]**2/2+41*x[1]**2*x[2]*x[3]/36
        +x[1]*x[3]-%i*x[1]**2*x[2]/24+%i*x[1];
de[2]:-17*x[2]*x[3]**3/4-3*%i*x[2]*x[3]**2/2+41*x[2]**2*x[1]*x[3]/36
        +x[2]*x[3]+%i*x[2]**2*x[1]/24-%i*x[2];
de[3]:4*x[3]**4+9*x[1]*x[2]*x[3]**2/8-x[3]**2
        -271*x[1]**2*x[2]**2/288-x[1]*x[2]/2;
xeq:[ x[1]=r*exp(%i*theta),x[2]=r*exp(-%i*theta) ];
rhp:subst(xeq,[de[1],de[2],de[3]]);
eq[1]:rdot*exp(%i*theta)+%i*r*exp(%i*theta)*thetadot=rhp[1];
e[1]:realpart(eq[1]);
e[2]:imagpart(eq[1]);
ans:solve([e[1],e[2]],[rdot,thetadot]);
e[3]:x3dot=rhp[3];
print(trigsimp(ans));
print(trigsimp(e[3]));
quit();
```

Output of Program 4 (Edited)

$$[[\text{rdot} = \frac{41\,x_3 r^3 + (36\,x_3 - 153\,x_3^3)r}{36}, \text{thetadot} = -\frac{r^2 - 36x, x_3^2 - 24}{24}]]$$

$$\text{x3dot} = -\frac{271\,r^4}{288} + \frac{9\,x_3^2 r^2}{8} - \frac{r^2}{2} + 4\,x_3^4 - x_3^2$$

OUTPUT 4:

$$a = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

for degree 2:

$$w(1) = \begin{bmatrix} 0 \\ x_1^2 \end{bmatrix}$$

$$w(2) = \begin{bmatrix} 0 & \\ x_1 & x_2 \end{bmatrix}$$

for degree 3:

$$w(1) = \begin{bmatrix} 0 & \\ x_1^2 & x_2 \end{bmatrix}$$

$$w(2) = \begin{bmatrix} 0 \\ x_1^3 \end{bmatrix}$$

for degree 4:

$$w(1) = \begin{bmatrix} 0 \\ x_1^4 \end{bmatrix}$$

$$w(2) = \begin{bmatrix} 0 & \\ x_1^3 & x_2 \end{bmatrix}$$

for degree 5:

$$w(1) = \begin{bmatrix} 0 & \\ x_1^4 & x_2 \end{bmatrix}$$

$$w(2) = \begin{bmatrix} 0 \\ x_1^5 \end{bmatrix}$$

OUTPUT 5:

$$a = \begin{bmatrix} -\%i & 0 \\ 0 & \%i \end{bmatrix}$$

for degree 2:

no terms

for degree 3:

$$w(1) = \begin{bmatrix} x_1^2 & x_2 \\ 0 & \end{bmatrix}$$

$$w(2) = \begin{bmatrix} 0 & \\ x_1 & x_2^2 \end{bmatrix}$$

for degree 4:

no terms

for degree 5:

$$w(1) = \begin{bmatrix} 0 & \\ x_1^2 & x_2^3 \end{bmatrix}$$

$$w(2) = \begin{bmatrix} x_1^3 & x_2^2 \\ 0 & \end{bmatrix}$$

References

[1] V. Arnold, *Geometrical Methods in the Theory of Ordinary Differential Equations* (Springer, New York, 1982).
[2] S.-N. Chow and J.K. Hale, *Methods of Bifurcation Theory* (Springer, New York, 1982).
[3] S.-N. Chow and D. Wang, Normal forms of bifurcating periodic orbits, in: M. Golubitsky and J. Guckenheimer, Eds., *Multiparameter Bifurcation Theory*, Contemp. Math. **56** (Amer. Mathematical Soc., Providence, RI, 1986) 9–18.
[4] R. Cushman and J. Sanders, Splitting algorithm for nilpotent normal form, preprint, MSI Cornell Univ., Ithaca, NY, 1987.
[5] C. Elphick, E. Tirapegui, M.E. Brachet, P. Coullet and G. Iooss, A simple global characterization for normal forms of singular vector fields, preprint, Univ. Nice, 1986.
[6] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcation of Vector Fields* (Springer, Berlin, 1983).
[7] M.D. Kruskal and H. Segur, Asymptotics beyond all orders in a model of dentrites, to appear.
[8] R.H. Rand and W.L. Keith, Normal forms and center manifold calculation on MACSYMA, in: R. Pavelle, Ed., *Applications of Computer Algebra*.
[9] F. Takens, Singularities of vector fields, *Publ. Math. Inst. Hautes Etades Sci.* **43** (1974) 47–100.