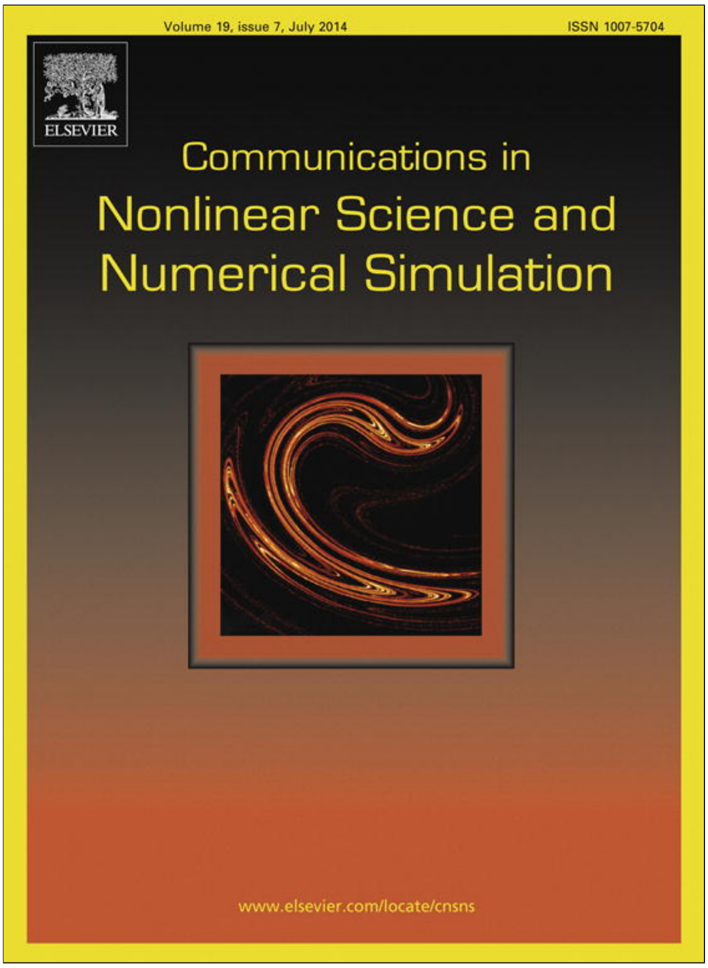
Volume 19, issue 7, July 2014
ISSN 1007-5704
ELSEVIER
Communications in
Nonlinear Science and
Numerical Simulation
www.elsevier.com/locate/cnsns

Commun Nonlinear Sci Numer Simulat 19 (2014) 2294–2308

Contents lists available at ScienceDirect

Commun Nonlinear Sci Numer Simulat

journal homepage: www.elsevier.com/locate/cnsns

An explicit recursive formula for computing the normal forms associated with semisimple cases

Yun Tian, Pei Yu *

Department of Applied Mathematics, Western University London, Ontario, Canada N6A 5B7

A R T I C L E I N F O

Article history:
Received 4 October 2013
Accepted 28 November 2013
Available online 6 December 2013

Keywords:
Differential system
Semisimple singularity
Normal form
Center manifold
Computational efficiency
Maple

A B S T R A C T

This paper presents an explicit, computationally efficient, recursive formula for computing the normal forms, center manifolds and nonlinear transformations for general n-dimensional systems, associated with semisimple singularities. Based on the formula, we develop a Maple program, which is very convenient for an end-user who only needs to prepare an input file and then execute the program to "automatically" generate the result. Several examples are presented to demonstrate the computational efficiency of the algorithm.

1. Introduction

Normal form theory has been used for several decades as one of the important tools in simplifying the study of nonlinear differential systems. Its basic idea is to introduce a near-identity transformation into a given differential system to eliminate as many of the nonlinear terms as possible, which are usually called non-resonant terms. The terms retained in the resulting system are normal form terms, called resonant terms. Since normal forms keep the fundamental dynamical characteristics of the original system in the vicinity of a singular point, it can be used to study the local bifurcations and stability/instability properties of the original system. There are various of books which have extensive discussions on normal form theory, for example, see [1–3]. More recent progress can be found in the article [4].

For higher-dimensional dynamical systems, normal form theory is usually applied together with center manifold theory, see [5–9]. If the Jacobian matrix of a differential system evaluated at a singular point contains eigenvalues with zero real part and non-zero real part, then center manifold theory should be considered in the study of the local dynamics of the system, and the dimension of the center manifold is equal to the number of eigenvalues with zero real part. Center manifold theory plays an important role in simplifying the analysis of local dynamical behavior of nonlinear differential systems near a singular point, because it allows us to determine the behavior by study the flow on a lower dimensional manifold.

Several computer algebra systems such as Maple, Mathematica, Macsyma, etc., have been widely used for the computation of normal forms. Even with the help of these computer algebra systems, it is still not easy to obtain higher-order normal forms since considerably more computer memory and computational time are demanded as the order of normal forms increases. Therefore, in the past two decades, various methods have been developed to compute normal forms for general n-dimensional differential systems. However, many methods are not computationally efficient because lots of unnecessary computations are involved, for example, see [6,10,11]. To be precise, in order to get an expression for the kth-order normal

* Corresponding author. Fax: +1 519 661 3523.
E-mail addresses: ytian56@uwo.ca (Y. Tian), pyu@uwo.ca (P. Yu).

1007-5704/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cnsns.2013.11.019

form computation, $(k-1)$th-order normal forms, center manifolds and near-identity transformation are substituted into the original system. Thus, besides the kth-order terms, the obtained expression also contains lower-order ($< k$) and higher-order ($> k$) terms, which are not desirable for efficient computation. To overcome this problem, Yu [7,12] developed a recursive formula for computing the coefficients of normal forms and center manifolds, which avoid those lower-order ($< k$) and higher-order ($> k$) terms in the kth-order computation. However, these formulas are not given in explicit recursive expressions and may be not so efficient in computation. For general planar systems, [13] obtained an explicit recursive formula for computing Poincaré–Lyapunov constants (focus values), and the computation based on this formula is efficient.

In this paper, we consider general n-dimensional differential systems associated with semisimple cases, i.e., the Jacobian matrix of the linearized system evaluated at a singular point can be transformed into a diagonal Jordan canonical form. Around semisimple singularities, a rich variety of bifurcations, such as Hopf, double-zero, Hopf-zero, double-Hopf, etc. may occur. A detailed study for some types of these bifurcations can be found in [14, chap. 7] by applying normal form theory to simplifying the systems. Particularly, for some special bifurcations like Hopf-zero, double-Hopf without resonance, the normal forms are symmetric with respect to rotation in the direction associated with the imaginary eigenvalues. In this case, the normal forms can be decoupled, and the systems are further simplified. Many methods have been developed and used to compute the normal forms of systems with semisimple singularities, not only for the particular cases like Hopf [9,12,13], Hopf-zero [15] and double-Hopf [16,17], but also for general semisimple cases involving center manifold [6,7]. In order to provide a good algorithm to compute the normal forms of general cases, in this paper we will develop a computationally efficient method and a Maple program without restriction on the dimension of the center manifold. This paper is an extension of our recent work [9], which focuses on general differential systems associated with Hopf bifurcation.

In the next section, an explicit, computationally efficient, recursive formula is derived for computing the normal forms and center manifolds of dynamical systems associated with semisimple singularities. The explicit formula is given in terms of the system coefficients of the original differential system, which is easily used for developing a Maple program. In Section 3, several examples are presented to demonstrate the computational efficiency of the method and the Maple program. Finally, conclusion is drawn in Section 4.

2. Main result

Consider a system of differential equations in the general form,

$$\dot{\mathbf{y}} = A\mathbf{y} + \mathbf{G}(\mathbf{y}), \quad \mathbf{y} \in \mathbf{R}^n, \quad \mathbf{G}(\mathbf{y}) : \mathbf{R}^n \to \mathbf{R}^n, \tag{1}$$

where the dot represents differentiation with respect to time, t, the matrix A is diagonalizable, $\mathbf{G}(\mathbf{0}) = \mathbf{0}$ and $D_{\mathbf{y}}\mathbf{G}(\mathbf{0}) = \mathbf{0}$. Denote by $\lambda_i, i = 1, \ldots, n$, the eigenvalues of A. Without loss of generality, it is assumed that there are only k eigenvalues $\lambda_j, j = 1, \ldots, k$, having zero real part, implying that system (1) has a k-dimensional center manifold.

Then, through a proper linear transformation, system (1) can be transformed into

$$\dot{\mathbf{x}} = J\mathbf{x} + \mathbf{f}(\mathbf{x}), \tag{2}$$

where J is a diagonal matrix, and $\mathbf{f}(\mathbf{x})$ is expanded as

$$\mathbf{f}(\mathbf{x}) = \sum_{m\geqslant 2} \mathbf{f}_m(\mathbf{x}), \quad \text{where } \mathbf{f}_m(\mathbf{x}) = \sum_{\{m(n)\}} \mathbf{f}_{m(n)} x_1^{m_1} x_2^{m_2} \ldots x_n^{m_n}$$

and $m(n)$ denotes a vector $(m_1, m_2, \ldots, m_n)$ of n nonnegative integers, which satisfies $\sum_{j=1}^{n} m_j = m$.

Suppose that the matrix J has the form $J = \mathrm{diag}(J_o, J_r)$, where

$$J_o = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_k), \quad J_r = \mathrm{diag}(\lambda_{k+1}, \lambda_{k+2}, \ldots, \lambda_n).$$

Let $\mathbf{x} = (\mathbf{x}_o^T, \mathbf{x}_r^T)^T$, where $\mathbf{x}_o = (x_1, x_2, \ldots, x_k)^T$ and $\mathbf{x}_r = (x_{k+1}, x_{k+2}, \ldots, x_n)^T$. Then, system (2) can be written as

$$\begin{aligned} \dot{\mathbf{x}}_o &= J_o\mathbf{x}_o + \mathbf{f}_o(\mathbf{x}_o, \mathbf{x}_r), \\ \dot{\mathbf{x}}_r &= J_r\mathbf{x}_r + \mathbf{f}_r(\mathbf{x}_o, \mathbf{x}_r). \end{aligned} \tag{3}$$

The center manifold of (3) may be defined as $\mathbf{x}_r = \mathbf{H}(\mathbf{x}_o)$, which satisfies $\mathbf{H}(\mathbf{0}) = \mathbf{0}$, $D\mathbf{H}(\mathbf{0}) = \mathbf{0}$. Then, the differential equation describing the dynamics on the center manifold is given by

$$\dot{\mathbf{x}}_o = J_o\mathbf{x}_o + \mathbf{f}_o(\mathbf{x}_o, \mathbf{H}(\mathbf{x}_o)). \tag{4}$$

Next, introduce a near-identity nonlinear transformation, given by

$$\mathbf{x}_o = \mathbf{u} + \mathbf{Q}(\mathbf{u}) = \mathbf{u} + \sum_{m\geqslant 2}\sum_{\{m(k)\}} \mathbf{q}_{m(k)} u_1^{m_1} u_2^{m_2} \ldots u_k^{m_k} \equiv \mathbf{q}(\mathbf{u}), \tag{5}$$

into (4) to obtain the normal form,

$$\dot{\mathbf{u}} = J_o\mathbf{u} + \mathbf{C}(\mathbf{u}), \quad \text{where } \mathbf{C}(\mathbf{u}) = \sum_{m\geqslant 2}\sum_{\{m(k)\}} \mathbf{c}_{m(k)} u_1^{m_1} u_2^{m_2} \ldots u_k^{m_k}. \tag{6}$$

Now the center manifold can be expressed in the new variable $\mathbf{u}$, as follows:

$$\mathbf{x}_r = \mathbf{H}(\mathbf{q}(\mathbf{u})) = \sum_{m\geqslant 2}\sum_{\{m(k)\}} \mathbf{h}_{m(k)} u_1^{m_1} u_2^{m_2} \dots u_k^{m_k} \equiv \mathbf{h}(\mathbf{u}). \tag{7}$$

Combining the above steps yields the following equations

$$D_{\mathbf{u}}\begin{pmatrix}\mathbf{Q}(\mathbf{u})\\ \mathbf{h}(\mathbf{u})\end{pmatrix} J_o\mathbf{u} - \begin{pmatrix}J_o\mathbf{Q}(\mathbf{u})\\ J_r\mathbf{h}(\mathbf{u})\end{pmatrix} = \begin{pmatrix}\mathbf{F}_o(\mathbf{u})\\ \mathbf{F}_r(\mathbf{u})\end{pmatrix} - D_{\mathbf{u}}\begin{pmatrix}\mathbf{Q}(\mathbf{u})\\ \mathbf{h}(\mathbf{u})\end{pmatrix}\mathbf{C}(\mathbf{u}) - \begin{pmatrix}\mathbf{C}(\mathbf{u})\\ 0\end{pmatrix}, \tag{8}$$

where $\mathbf{F}_o(\mathbf{u}) = \mathbf{f}_o(\mathbf{q}(\mathbf{u}), \mathbf{h}(\mathbf{u}))$, $\mathbf{F}_r(\mathbf{u}) = \mathbf{f}_r(\mathbf{q}(\mathbf{u}), \mathbf{h}(\mathbf{u}))$. Comparing the coefficients on both sides of (8), we obtain the recursive formulas for the coefficients of the center manifold and the normal form as well as the associated nonlinear transformation.

For convenience, we first introduce some notations. Suppose the powers of $\mathbf{q}(\mathbf{u})$ and $\mathbf{h}(\mathbf{u})$ can be expressed, for $j \geqslant 0$, as

$$\begin{aligned}\mathbf{q}^j(\mathbf{u}) &= \sum_{m=j}^{\infty}\sum_{\{m(k)\}} \mathbf{q}^j_{m(k)} u_1^{m_1} u_2^{m_2} \dots u_k^{m_k},\\ \mathbf{h}^j(\mathbf{u}) &= \sum_{m=2j}^{\infty}\sum_{\{m(k)\}} \mathbf{h}^j_{m(k)} u_1^{m_1} u_2^{m_2} \dots u_k^{m_k}.\end{aligned} \tag{9}$$

We have the following main result.

Theorem 1. *For any fixed $s(k), s \geqslant 2$, let $\Lambda = \sum_{i=1}^{k}\lambda_i s_i$. Then the recursive formulas for the coefficients of the nonlinear transformation (5), the normal form (6) and the center manifold (7) of system (3), i.e., $\mathbf{q}_{s(k)}, \mathbf{c}_{s(k)}$ and $\mathbf{h}_{s(k)}$, are given below.*

(1) For $\mathbf{q}_{s(k)}$ and $\mathbf{c}_{s(k)}$, if $\Lambda - \lambda_j = 0, j = 1, \dots, k$, then

$$q_{s(k),j} = 0, \quad c_{s(k),j} = a_{s(k),j} - b_{s(k),j},$$

otherwise,

$$q_{s(k),j} = (a_{s(k),j} - b_{s(k),j})/(\Lambda - \lambda_j), \quad c_{s(k),j} = 0.$$

(2) For $\mathbf{h}_{s(k)}$, we have

$$h_{s(k),j-k} = (a_{s(k),j} - b_{s(k),j})/(\Lambda - \lambda_j), \quad j = k+1, \dots, n;$$

where

$$\begin{aligned}\mathbf{a}_{s(k)} &= \sum_{m=2}^{s}\sum_{\{m(n)\}}\sum_{\{j(n)\}}^{j=s}\sum_{\{j_1(k)\}}\sum_{\{j_2(k)\}}\cdots\sum_{\{j_n(k)\}} \mathbf{f}_{m(n)} q^{m_1}_{j_1(k),1}\dots q^{m_k}_{j_k(k),k} h^{m_{k+1}}_{j_{k+1}(k),1}\dots h^{m_n}_{j_n(k),n-k},\\ \mathbf{b}_{s(k)} &= \sum_{i=1}^{k}\sum_{l=2}^{s-1}\sum_{\{l(k)\}}(s_i+1-l_i)\begin{pmatrix}\mathbf{q}_{s(k)-l(k)+e_i(k)}\\ \mathbf{h}_{s(k)-l(k)+e_i(k)}\end{pmatrix} c_{l(k),i},\\ \mathbf{q}^j_{s(k)} &= \sum_{l=j-1}^{s-1}\sum_{l(k)\leqslant s(k)} \mathbf{q}^{j-1}_{l(k)}\mathbf{q}_{s(k)-l(k)},\\ \mathbf{h}^j_{s(k)} &= \sum_{l=2j-2}^{s-2}\sum_{l(k)\leqslant s(k)} \mathbf{h}^{j-1}_{l(k)}\mathbf{h}_{s(k)-l(k)}.\end{aligned}$$

Proof. For any given integer $s \geqslant 2$, suppose that we have obtained $\mathbf{q}_{m(k)}$, $\mathbf{h}_{m(k)}$ and $\mathbf{c}_{m(k)}$ for $m < s$. Now, we want to derive the formulas for $\mathbf{q}_{s(k)}$, $\mathbf{h}_{s(k)}$ and $\mathbf{c}_{s(k)}$. We divide the proof in three steps, which can also be served as the guidelines for developing programs using a computer algebra system.

Step 1. First of all, we need to compute all the coefficients of terms with degree s for $\mathbf{x}_o^j = \mathbf{q}^j(\mathbf{u}), 2 \leqslant j \leqslant s$. Since $\mathbf{q}^j(\mathbf{u}) = \mathbf{q}(\mathbf{u})\mathbf{q}^{j-1}(\mathbf{u})$, we have

$$\begin{aligned}\mathbf{q}^j(\mathbf{u}) &= \left(\sum_{m=1}^{\infty}\sum_{\{m(k)\}}\mathbf{q}_{m(k)} u_1^{m_1} u_2^{m_2}\dots u_k^{m_k}\right)\left(\sum_{m=j-1}^{\infty}\sum_{\{m(k)\}}\mathbf{q}^{j-1}_{m(k)} u_1^{m_1} u_2^{m_2}\dots u_k^{m_k}\right)\\ &= \sum_{m=j}^{s}\sum_{\{m(k)\}}\sum_{l=j-1}^{m-1}\sum_{l(k)\leqslant m(k)} \mathbf{q}^{j-1}_{l(k)}\mathbf{q}_{m(k)-l(k)} u_1^{m_1} u_2^{m_2}\dots u_k^{m_k} + o(|\mathbf{u}|^s),\end{aligned}$$

where $l(k) \leqslant m(k)$ means $l_i \leqslant m_i$ for $i = 1, \dots, k$. Then, we obtain

$$\mathbf{q}^j_{s(k)} = \sum_{l=j-1}^{s-1}\sum_{l(k)\leqslant s(k)} \mathbf{q}^{j-1}_{l(k)}\mathbf{q}_{s(k)-l(k)}, \quad 2 \leqslant j \leqslant s.$$

Similarly, for $\mathbf{x}^j_r = \mathbf{h}^j(\mathbf{u})$, we have

$$\mathbf{h}^j_{s(k)} = \sum_{l=2j-2}^{s-2}\sum_{l(k)\leqslant s(k)} \mathbf{h}^{j-1}_{l(k)}\mathbf{h}_{s(k)-l(k)}, \quad 2 \leqslant j \leqslant s.$$

Step 2. Denote

$$\begin{pmatrix}\mathbf{F}_o(\mathbf{u})\\ \mathbf{F}_r(\mathbf{u})\end{pmatrix} = \sum_{m=2}^{s}\sum_{m(k)} \mathbf{a}_{\{m(k)\}} u_1^{m_1}u_2^{m_2}\dots u_k^{m_k} + o(|\mathbf{u}|^s). \tag{10}$$

In this step, we derive the formula for $\mathbf{a}_{s(k)}$. Let $\mathbf{q}^m_{l(k)} = (q^m_{l(k),1}, q^m_{l(k),2}, \dots, q^m_{l(k),k})^T$ and $\mathbf{h}^m_{l(k)} = (h^m_{l(k),1}, h^m_{l(k),2}, \dots, h^m_{l(k),n-k})^T$. For $2 \leqslant m \leqslant s$, substituting $\mathbf{q}(\mathbf{u})$ and $\mathbf{h}(\mathbf{u})$ into $\mathbf{f}_m(\mathbf{x})$ yields

$$\begin{aligned}
\mathbf{f}_m(\mathbf{x}) &= \sum_{\{m(n)\}} \mathbf{f}_{m(n)} x_1^{m_1}x_2^{m_2}\dots x_n^{m_n} = \sum_{\{m(n)\}} \mathbf{f}_{m(n)} \prod_{i=1}^{k} q_i^{m_i}(\mathbf{u}) \prod_{i=1}^{n-k} h_i^{m_{k+i}}(\mathbf{u}) \\
&= \sum_{\{m(n)\}} \mathbf{f}_{m(n)} \prod_{i=1}^{k}\left(\sum_{l=m_i}^{\infty}\sum_{\{l(k)\}} q^{m_i}_{l(k),i} u_1^{l_1}u_2^{l_2}\dots u_k^{l_k}\right)\prod_{i=1}^{n-k}\left(\sum_{l=2m_{k+i}}^{\infty}\sum_{\{l(k)\}} h^{m_{k+i}}_{l(k),i} u_1^{l_1}u_2^{l_2}\dots u_k^{l_k}\right) \\
&= \sum_{\{m(n)\}} \mathbf{f}_{m(n)} \left(\sum_{l=m}^{s}\sum_{\{l(k)\}}\sum_{j_1=m_1}^{l}\cdots\sum_{j_k=m_k}^{l}\sum_{j_{k+1}=2m_{k+1}}^{l}\cdots\sum_{j_n=2m_n}^{l}\sum_{\{j_1(k)\}}\sum_{\{j_2(k)\}}\cdots\sum_{\{j_n(k)\}} .q^{m_1}_{j_1(k),1}q^{m_2}_{j_2(k),2}\dots q^{m_k}_{j_k(k),k}h^{m_{k+1}}_{j_{k+1}(k),1}h^{m_{k+2}}_{j_{k+2}(k),2}\dots h^{m_n}_{j_n(k),n-k}u_1^{l_1}u_2^{l_2}\dots u_k^{l_k} + o(|\mathbf{u}|^s)\right) \\
&= \sum_{l=m}^{s}\sum_{\{l(k)\}}\sum_{\{m(n)\}}\sum_{\{j(n)\}}^{j=l}\sum_{\{j_1(k)\}}\sum_{\{j_2(k)\}}\cdots\sum_{\{j_n(k)\}} \mathbf{f}_{m(n)} q^{m_1}_{j_1(k),1}\dots q^{m_k}_{j_k(k),k}h^{m_{k+1}}_{j_{k+1}(k),1}\dots h^{m_n}_{j_n(k),n-k}u_1^{l_1}u_2^{l_2}\dots u_k^{l_k} + o(|\mathbf{u}|^s),
\end{aligned}$$

where $\sum_{i=1}^{n} j_i(k) = l(k)$.

Since $\mathbf{f}(\mathbf{x}) = \sum_{m\geq 2}\mathbf{f}_m(\mathbf{x})$, we consequently obtain

$$\mathbf{a}_{s(k)} = \sum_{m=2}^{s}\sum_{\{m(n)\}}\sum_{\{j(n)\}}^{j=s}\sum_{\{j_1(k)\}}\sum_{\{j_2(k)\}}\cdots\sum_{\{j_n(k)\}} \mathbf{f}_{m(n)} q^{m_1}_{j_1(k),1}\dots q^{m_k}_{j_k(k),k}h^{m_{k+1}}_{j_{k+1}(k),1}\dots h^{m_n}_{j_n(k),n-k},$$

where the vector $j(n)$ satisfies that

$$j_i\begin{cases} = 0 & \text{if } m_i = 0, \\ \left.\begin{aligned} &\geqslant m_i \quad \text{for } 1 \leqslant i \leqslant k \\ &\geqslant 2m_i \quad \text{for } k+1 \leqslant i \leqslant n \end{aligned}\right\} & \text{if } m_i \neq 0. \end{cases}$$

Step 3. Denote

$$D_{\mathbf{u}}\begin{pmatrix}\mathbf{Q}(\mathbf{u})\\ \mathbf{h}(\mathbf{u})\end{pmatrix}\mathbf{C}(\mathbf{u}) = \sum_{m=3}^{s}\sum_{\{m(k)\}} \mathbf{b}_{m(k)} u_1^{m_1}u_2^{m_2}\dots u_k^{m_k} + o(|\mathbf{u}|^s). \tag{11}$$

In this step, we derive the formula for $\mathbf{b}_{s(k)}$. Note that

$$\begin{aligned}
D_{\mathbf{u}}\begin{pmatrix}\mathbf{Q}(\mathbf{u})\\ \mathbf{h}(\mathbf{u})\end{pmatrix}\mathbf{C}(\mathbf{u}) &= \sum_{i=1}^{k}\begin{pmatrix}\mathbf{Q}_{u_i}(\mathbf{u})\\ \mathbf{h}_{u_i}(\mathbf{u})\end{pmatrix} C_i(\mathbf{u}) = \sum_{i=1}^{k}\left(\sum_{m=2}\sum_{\{m(k)\}} m_i \begin{pmatrix}\mathbf{q}_{m(k)}\\ \mathbf{h}_{m(k)}\end{pmatrix} u_1^{m_1}u_2^{m_2}\dots u_k^{m_k}u_i^{-1}\right)\left(\sum_{m=2}\sum_{\{m(k)\}} c_{m(k),i}u_1^{m_1}u_2^{m_2}\dots u_k^{m_k}\right) \\
&= \sum_{m=3}^{s}\sum_{\{m(k)\}}\sum_{i=1}^{k}\sum_{l=2}^{m-1}\sum_{l}(m_i+1-l_i)\begin{pmatrix}\mathbf{q}_{m(k)-l(k)+e_i(k)}\\ \mathbf{h}_{m(k)-l(k)+e_i(k)}\end{pmatrix} c_{l(k),i}u_1^{m_1}u_2^{m_2}\dots u_k^{m_k},
\end{aligned}$$

where $e_i(k)$ is a unit vector with a 1 in the ith place. Therefore, comparing the above equation with (11) we have

$$\mathbf{b}_{s(k)} = \sum_{i=1}^{k}\sum_{l=2}^{s-1}\sum_{\{l(k)\}}(s_i+1-l_i)\begin{pmatrix}\mathbf{q}_{s(k)-l(k)+e_i(k)}\\ \mathbf{h}_{s(k)-l(k)+e_i(k)}\end{pmatrix} c_{l(k),i}.$$

Finally, from the left-hand side of (8), we obtain

$$\begin{aligned}
D_{\mathbf{u}}\mathbf{Q}(\mathbf{u})J_o\mathbf{u} - J_o\mathbf{Q}(\mathbf{u}) &= \sum_{i=1}^{k}\lambda_i u_i \mathbf{Q}_{u_i} - J_o\mathbf{Q}(\mathbf{u}) = \sum_{i=1}^{k}\sum_{m=2}\sum_{\{m(k)\}}\lambda_i m_i \mathbf{q}_{m(k)}u_1^{m_1}u_2^{m_2}\dots u_k^{m_k} - \sum_{m=2}\sum_{\{m(k)\}} J_o\mathbf{q}_{m(k)}u_1^{m_1}u_2^{m_2}\dots u_k^{m_k} \\
&= \sum_{m=2}^{s}\sum_{\{m(k)\}}\left(\sum_{i=1}^{k}\lambda_i m_i I_k - J_o\right)\mathbf{q}_{m(k)}u_1^{m_1}u_2^{m_2}\dots u_k^{m_k} + o(|\mathbf{u}|^s)
\end{aligned} \tag{12}$$

and similarly,

$$D_{\mathbf{u}}\mathbf{h}(\mathbf{u})J_o\mathbf{u} - J_r\mathbf{h}(\mathbf{u}) = \sum_{m=2}^{s}\sum_{\{m(k)\}}\left(\sum_{i=1}^{k}\lambda_i m_i I_{n-k} - J_r\right)\mathbf{h}_{m(k)}u_1^{m_1}u_2^{m_2}\ldots u_k^{m_k} + o(|\mathbf{u}|^s). \tag{13}$$

Substituting (6) and (10)–(13) into (8) and comparing the coefficients of the same order results in the formulas in Theorem 1, and we thus complete the proof. □

The source code of the Maple program developed using the formulas in Theorem 1 is given in Appendix for the convenience of readers.

3. Application

In this section, we present several examples to demonstrate the applicability and the computational efficiency of the Maple program (see the source code in Appendix) developed in this paper. We show three examples associated with Hopf, Hopf-zero and double Hopf singularities, and compute their normal forms and center manifolds, as well as the corresponding nonlinear transformations. We have tested a number of systems for comparing the algorithm developed in this paper with that given in [6]. It is shown that for most cases the method developed in this paper is better than that given in [6]. Only in some special cases, the situation is reversed. The program given in [6] can only deal with the cases where the dimension of the center manifold is less than seven. All the Maple programs are executed on a desktop machine with CPU 3.4 GHZ and 32G RAM memory to generate the normal forms as needed.

Example 1. We consider a 5-dimensional system:

$$\begin{aligned}
\dot{x}_1 &= x_2 + x_1^2 - x_1x_3 + x_5^2,\\
\dot{x}_2 &= -x_1 + x_2^2 + x_1x_4 + x_2^3,\\
\dot{x}_3 &= -x_3 + x_1^2,\\
\dot{x}_4 &= -x_4 + x_5 + x_1^2 + x_4x_5,\\
\dot{x}_5 &= -x_4 - x_5 + x_2^2 - 2x_4^2.
\end{aligned} \tag{14}$$

The Jacobian matrix of this system evaluated at the origin has eigenvalues $\pm i$, -1 and $-1 \pm i$. So the origin is a Hopf singularity and system (14) has a 2-dimensional center manifold. The normal form given in polar coordinates up to 5th order is given as follows:

$$\begin{aligned}
\dot{r} &= \frac{3}{40}r^3 - \frac{25633}{102000}r^5 - \frac{163441769}{2663424000}r^7 + \cdots\\
\dot{\theta} &= 1 - \frac{7}{12}r^2 + \frac{6692923}{14688000}r^4 - \frac{47098141289}{299635200000}r^6 + \cdots
\end{aligned} \tag{15}$$

The lengthy expressions for the center manifold and nonlinear transformation are omitted here for brevity.

Remark 1. The coefficients of the terms r^3 and r^5, etc., in the first equation of (15) are called the first, second, etc., *focus values*. In general, the normal form of system (3), given in polar coordinates, is in the form of

$$\begin{aligned}
\dot{r} &= r(v_0 + v_1r^2 + v_2r^4 + \cdots v_kr^{2k} + \cdots),\\
\dot{\theta} &= 1 + t_0 + t_1r^2 + t_2r^4 + \cdots t_kr^{2k} + \cdots,
\end{aligned} \tag{16}$$

where v_k is called the kth-order focus value, which is a function of the system parameters of (3). Small limit cycles bifurcating from the origin and their stability can be determined from the first equation of (16). The second equation of (16) can be used to determine the frequency of the bifurcating periodic motion (limit cycle).

Example 2. The second example is a 6-dimensional differential system, described by

$$\begin{aligned}
\dot{x}_1 &= -x_1^2 + 2x_1x_2 + 3x_1x_4 - x_1x_5 - x_2^2 + x_2x_4,\\
\dot{x}_2 &= x_3 - x_1^2 + 2x_1x_3 + 8x_1x_4 + x_3x_5,\\
\dot{x}_3 &= -x_2 - x_3^2 + 3x_1x_6 - x_3x_4 - 6x_4^2 - x_4x_6 + 2x_5^2,\\
\dot{x}_4 &= -x_4 - x_1^2 + 2x_1x_2 + 3x_1x_4 - x_1x_5 - x_2^2,\\
\dot{x}_5 &= -x_5 + x_6 - 7x_1^2 + 2x_1x_3 + 3x_1x_6 - x_3x_4 - x_4x_6,\\
\dot{x}_6 &= -x_5 - x_6 + x_1x_4 - 5x_3^2 + x_3x_5 - 4x_4^2 + x_5^2.
\end{aligned} \tag{17}$$

This system has a singular point at the origin, with its Jacobian matrix evaluated at the origin having three eigenvalues, 0 and $\pm i$, with zero real part, and three eigenvalues, -1 and $-1 \pm i$, with negative real part, implying that system (17) contains a 3-dimensional center manifold associated with a Hopf-zero singularity at the origin. Executing our Maple program gives the normal form (in cylindrical coordinates) up to 5th order,

$$\begin{aligned}
\dot{y} &= -y^2 - \frac{1}{2}r^2 + \frac{1}{2}y^3 - \frac{5}{4}yr^2 + \frac{59}{4}y^4 - \frac{259}{40}y^2r^2 + \frac{1}{36}r^4 + 84y^5 + \frac{18509}{400}y^3r^2 + \frac{11483}{4800}yr^4 + \cdots \\
\dot{r} &= \frac{29}{10}y^2r + \frac{9}{40}r^3 - \frac{1171}{25}y^3r - \frac{1371}{200}yr^3 - \frac{19331}{80}y^4r - \frac{263299}{2250}y^2r^3 - \frac{576761}{1224000}r^5 + \cdots \\
\dot{\theta} &= 1 + y - \frac{61}{20}y^2 - \frac{163}{240}r^2 + \frac{4501}{200}y^3 - \frac{1357}{800}yr^2 + \frac{4579}{160}y^4 + \frac{123833}{2250}y^2r^2 - \frac{102206489}{58752000}r^4 + \cdots
\end{aligned}$$

Example 3. The last example is a 7-dimensional differential system,

$$\begin{aligned}
\dot{x}_1 &= x_2 + x_1^3 - x_1^2x_5 + x_1^2x_7, \\
\dot{x}_2 &= -x_1 - 2x_1x_3^2, \\
\dot{x}_3 &= \sqrt{2}x_4 + x_1^2x_3 - 4x_5^3, \\
\dot{x}_4 &= -\sqrt{2}x_3, \\
\dot{x}_5 &= -x_5 + (x_1 - x_5)^2, \\
\dot{x}_6 &= -x_6 + x_7 + (x_1 - x_4)^2, \\
\dot{x}_7 &= -x_6 - x_7 + (x_2 - x_6)^2,
\end{aligned} \tag{18}$$

whose Jacobian matrix evaluated at the origin has eigenvalues $\pm i$, $\pm\sqrt{2}i$, -1 and $-1 \pm i$, and four of them have zero real part. So the center manifold of system (7) is four dimensional. System (18) was studied by [6] and the normal form in polar coordinates up to 5th order was also given. We executed the Maple programs developed in this paper as well as that given in [6] on the desktop machine. We have found that the Maple program given in [6] failed when it was executing to find the 9th-order normal form, since the Maple was unable to allocate enough memory to complete the computation. While the program developed in this paper only took 122 s and 13938 MB memory to finish the 9th-order normal form computation. The normal form up to 7th order given in polar coordinates is listed below.

$$\begin{aligned}
\dot{r}_1 &= \frac{3}{8}r_1^3 + \frac{157}{1360}r_1^5 - \frac{9}{40}r_1^3r_2^2 - \frac{428923841}{3847168000}r_1^7 - \frac{433291}{832320}r_1^5r_2^2 - \frac{612973}{8921600}r_1^3r_2^4 + \cdots \\
\dot{\theta}_1 &= 1 + \frac{1}{2}r_2^2 - \frac{5543}{21760}r_1^4 - \frac{3}{80}r_1^2r_2^2 - \frac{1}{16}r_2^4 - \frac{888039}{9617920}r_1^6 + \frac{1744833}{5178880}r_1^4r_2^2 - \frac{1448249}{93676800}r_1^2r_2^4 + \frac{3}{32}r_2^6 + \cdots \\
\dot{r}_2 &= \frac{1}{4}r_1^2r_2^2 - \frac{1}{16}r_1^2r_2^3 + \frac{10213}{348160}r_1^6r_2 - \frac{3457}{446080}r_1^4r_2^3 + \frac{27}{256}r_1^2r_2^5 + \cdots \\
\dot{\theta}_2 &= \sqrt{2} - \frac{1}{32}\sqrt{2}r_1^4 + \frac{125}{89216}\sqrt{2}r_1^4r_2^2 + \cdots
\end{aligned}$$

4. Conclusion

In this paper, we have derived an explicit, recursive formula for computing the normal forms, center manifolds and nonlinear transformations for general n-dimensional systems, associated with semisimple singularities. A Maple program is also developed on the basis of the formula, which is very convenient for practical applicants who may not be familiar with normal form theory. It only needs a user to prepare an input file and the Maple program will be "automatically" executed to generate the desired result. Three examples are presented to show the applicability of the new method and new program, and in particular, one of the examples demonstrates the advantage of the new method over the existing methods and programs.

Acknowledgment

This work was supported by the Natural Science and Engineering Research Council of Canada (NSERC).

Appendix A

In this appendix, for the convenience of readers, we list the symbolic Maple program developed in this paper using the recursive formulas in Theorem 1, which can be used for computing the normal forms of general n-dimensional systems associated with semisimple cases. The input here takes the third example in the section of application.

```
with (LinearAlgebra):
M1        :=  0:          #  No. of zero eigenvalues
M2        :=  2:          #  No. of pairs of purely imaginary eigenvalues
M3        :=  1:          #  No. of non-zero real eigenvalues
M4        :=  1:          #  No. of pairs of complex conjugate eigenvalues
N         :=  3:          #  Highest order in the system
Ord        :=  5:
Mc        :=  M1 + 2*M2:
M        :=  Mc + M3 + 2*M4:
L        :=  M1 + M2 + M3 + M4:
f[1]  := x[2] + x[1]^3 − x[1]^2*x[5] + x[1]^2*x[7]:
f[2]   := −x[1] − 2*x[1]*x[3]^2:
f[3]  := sqrt (2)*x[4] + x[1]^2*x[3] − 4*x[5]^3:
f[4]   := − sqrt (2)*x[3]:
f[5]  := − x[5] + (x[1] − x[5])^2:
f[6]   := − x[6] + x[7] + (x[1] − x[4])^2:
f[7]   := − x[6] − x[7] + (x[2] − x[6])^2:
L3seq   := proc ()
  global l12,S3,p:
  if l12 = 0 then
    S3[p+1] := S3[p+1]+1: l12 := S3[p]−1:
    S3[p] := 0: p := max (0,sign (−l12))*p+1:
  else S3[1] := S3[1]+1: l12 := l12−1: fi:
end:
L3product := proc (sl,sr,q2r,q2i)
    local l3rmx,qpmx,qpr,qpi,ctpo,l12,l12r,p,pr,ctl,ctr,ctp,l3,l3r,sb,
      sp,S3,S3r,i,temp:
    l3rmx := binomial (sr+Mc−2,Mc−2): ctpo := 1:
    qpmx := binomial (sl+sr+Mc−1,Mc−1):
    qpr := Array (1..qpmx): qpi := Array (1..qpmx):
    S3 := [seq (0,i=1..Mc−1)]:
    p := 1: ctl := 1: l12 := sl:
    for l3 to binomial (sl+Mc−2,Mc−2) do
      S3r := [seq (0,i=1..Mc−1)]:
      pr := 1: ctr := 1: l12r := sr: ctp := ctpo:
      for l3r to l3rmx do
        for i from ctp to ctp+l12+l12r do
          sb := max (0,i−ctp−l12): sp := min (l12r,i−ctp):
          qpr[i] := qpr[i]+add (q2r[ctr+j]*q1r[ctl+i−ctp−j]
                 −q2i[j+ctr]*q1i[ctl+i−ctp−j],j=sb..sp):
          qpi[i] := qpi[i]+add (q2r[ctr+j]*q1i[ctl+i−ctp−j]
                 +q2i[j+ctr]*q1r[ctl+i−ctp−j],j=sb..sp):
      od:
      ctp := i: ctr := ctr+l12r+1:
      if l12r =0 then
      ctp := ctp−binomial (S3r[pr]+1,2)−S3r[pr]*l12:
      temp := l12+S3[1]:
      for i from 2 to pr do
        ctp := ctp+binomial (temp+i,i+1)
            −binomial (temp+S3r[pr]+i−1,i+1):
        temp := temp+S3[i]:
      od:
      ctp := ctp+binomial (temp+S3r[pr]+i−1,i):
      S3r[pr+1] := S3r[pr+1]+1: l12r := S3r[pr]−1:
      S3r[pr] := 0: pr := max (0,sign (−l12r))*pr+1:
    else S3r[1] := S3r[1]+1: l12r := l12r−1: fi:
  od:
  ctl := ctl+l12+1:
  if l12 =0 then
```

```
    ctpo := ctpo+binomial (sr+p+1,p+1):
    S3[p+1] := S3[p+1]+1: 112 := S3[p]−1:
    S3[p] := 0: p := max (0,sign (−112))*p+1:
  else ctpo := ctpo+112+sr+1: S3[1] := S3[1]+1: 112 := 112−1: fi:
od:
return [qpr,qpi]:
end:

for i to M1 do x[i] := v[i]: od:
j := M1+1: k := L+1:
for i from M1+1 to M1+M2 do
  x[j] := (v[i]+v[k])/2:
  x[j+1] := I*(v[i]−v[k])/2:
  f[i] := simplify (f[j]−I*f[j+1]):
  j := j+2: k := k+1:
od:
for i from M1+M2+1 to L−M4 do
  x[j] := v[i]:
  f[i] := simplify (f[j]):
  j := j+1:
od:
for i from L−M4+1 to L do
  x[j] := (v[i]+v[k])/2:
  x[j+1] := I*(v[i]−v[k])/2:
  f[i] := simplify (f[j]−I*f[j+1]):
  j := j+2: k := k+1:
od:
for j to L do
  f[j] := simplify (f[j]):
  IEf[j] := diff (f[j],v[j]):
  for k to M do IEf[j] := subs (v[k]=0,IEf[j]): od:
  REf[j] := subs (I=0,IEf[j]):
  IEf[j] := subs (I=1,IEf[j]−REf[j]):
od:
Qd := [seq (1,j=1..M1+M2),seq (2,j=1..M3+M4),seq (1,j=1..M2),seq (2,j=1..M4)]:
Qc := [seq (j,j=1..M1),seq (L+j,j=1..M2),seq (M1+M2+j,j=1..M3),
      seq (M−M4+j,j=1..M4),seq (M1+j,j=1..M2),seq (L−M4+j,j=1..M4)]:
Qb := [seq (j,j=1..M1),seq (seq (M1+i*M2+j,i=0..1),j=1..M2)]:
SizeIndex := Array (1..2*N): Mr. := [seq (1,i=1..L)]:
vecf := Vector ([seq (f[j],j=1..L)]):
for m from 2 to N do
  M1 := [m+1,−1,seq (0,i=1..M−2)]: i := 1:
  while M1[M] <> m do
    M1[i+1] := 1 +M1[i+1]: M1[1] := M1[i]−1:
    if i <> 1 then M1[i] := 0: fi:
    if M1[1] = 0 then i := i+1: else i := 1: fi:
    M1c := M1: ji := 0:
    for l to 2 do
      coef[l] := vecf: cterm := 1:
      for k to M do
        coef[l] := coeff (coef[l],v[k],M1c[k]):
        cterm := cterm*v[k]^M1c[k]:
      od:
      if coef[l] = 0 then coef[l] := Vector (L): fi:
      vecf := vecf-cterm*coef[l]:
      if Norm (coef[l],2) <> 0 then
        ji := ji+1:
        if ji =1 then
```

(continued on next page)

```
          Mlc := [seq (Mlc[Qc[k]],k=1..M)]:
          mlmx := max (Mlc−Ml):
          if mlmx =0 then l := l+1: fi:
        fi:
      else l := l+1: fi:
  od:
  if ji > 0 then
    Mr. := [seq (max (Mr[n],Ml[n]),n=1..L)]:
    qdg := m+add (Ml[n],n=M1+M2+1..L)+add (Ml[n],n = L+M2+1..M):
    jr := 0: jc := 0:
    for k from i to M do
      if Ml[k] <> 0 then
        if k < M1+1 or (k < L−M4 and k > M1+M2) then
          jr := jr+1: j := −jr:
        else jc := jc+1: j := jc: fi:
        Kvt[j] := Ml[k]: Ivt[j] := k: Qvt[j] := Qd[k]*Ml[k]:
      fi:
  od:
    kV := [seq (Kvt[j],j=1..jc),seq (Kvt[−j],j=1..jr)]:
    Iv := [seq (Ivt[j],j=1..jc),seq (Ivt[−j],j=1..jr)]:
    Qv := [seq (Qvt[j],j=1..jc),seq (Qvt[−j],j=1..jr)]:
    SizeIndex[qdg] := SizeIndex[qdg]+1:
    N := max (N,qdg): sqdg := SizeIndex[qdg]:
    Index[qdg,sqdg] := [kV,Iv,Qv,jc,jr,ji]:
  fi:
  for l to ji do
    eql := []:
    for k to L do
      if coef[l][k] <> 0 then eql := [op (eql),k]: fi:
    od:
    coefi := [seq (coef[l][eql[k]],k=1..nops (eql))]:
    coefr := subs (I=0,coefi):
    coefi := subs (I=1,coefi-coefr):
    sqdgn := (−1)^(l−1)*sqdg:
    Coef[qdg,sqdgn] := [eql,coefr,coefi]:
    od:
  od:
od:

for j to M do
    Ih[j,1,1] := Array (1..Mc): Rh[j,1,1] := Array (1..Mc):
od:
for j to M1 do Rh[j,1,1][j] := 1: od:
for j to M2 do
    Rh[M1+j,1,1][M1+2*j−1] := 1:
    Rh[L+j,1,1][M1+2*j] := 1:
od:
for s from 2 to Ord do
print (‘order=‘,s):
    smx := binomial (s+Mc−1,Mc−1):
    Ku := [seq (min (Mr[j],s),j=1..L)]:
    for j to L do
      for k from 2 to Ku[j] do
        Rh[j,k,s] := Array (1..smx): Ih[j,k,s] := Array (1..smx):
      od:
    od:
    for s1 to s−1 do
      l12 := s1: sr := s−s1: l3rmx := binomial (sr+Mc−2,Mc−2):
      S3 := [seq (0,i=1..Mc−1)]: p := 1: ctl := 1: ctpo := 1:
```

```
      for l3 to binomial (sl+Mc−2,Mc−2) do
        for j to L do
          Lslr[j] := [seq (Rh[j,l,sl][i],i=ctl..ctl+ll2)]:
          Lsli[j] := [seq (Ih[j,l,sl][i],i=ctl..ctl+ll2)]:
        od:
        S3r := [seq (0,i=1..Mc−1)]:
        pr := 1: ctr := 1: ll2r := sr: ctp := ctpo:
          for l3r to l3rmx do
          for l to L do
            for k to min (Ku[l]−1,sr) do
                Lsrr := [seq (Rh[l,k,sr][i],i=ctr..ctr+ll2r)]:
                Lsri := [seq (Ih[l,k,sr][i],i=ctr..ctr+ll2r)]:
                for i from ctp to ctp+ll2+ll2r do
                  sb := max (0,i−ctp−ll2): sp := min (ll2r,i−ctp):
                  Rh[l,k+l,s][i] := Rh[l,k+l,s][i]
                                    +add (Lsrr[j+1]*Lslr[l][i−ctp+1−j]
                                    −Lsri[j+1]*Lsli[l][i−ctp+1−j], j=sb..sp):
                  Ih[l,k+l,s][i] := Ih[l,k+l,s][i]
                    +add (Lsri[j+1]*Lslr[l][i−ctp+1−j]
                    +Lsrr[j+1]*Lsli[l][i−ctp+1−j], j=sb..sp):
                od:
            od:
          od:
          ctp := i: ctr := ctr+ll2r+1:
          if ll2r =0 then
              ctp := ctp-binomial (S3r[pr]+1,2)−S3r[pr]*ll2:
              temp := ll2+S3[1]:
              for i from 2 to pr do
                ctp := ctp+binomial (temp+i,1+i)
                  −binomial (temp+S3r[pr]+i−1,1+i):
                temp := temp+S3[i]:
        od:
        ctp := ctp+binomial (temp+S3r[pr]+i−1,i):
        S3r[pr+1] := S3r[pr+1]+1: ll2r := S3r[pr]−1:
        S3r[pr] := 0: pr := max (0,sign (−ll2r))*pr+1:
      else S3r[1] := S3r[1]+1: ll2r := ll2r−1: fi:
    od:
    ctpo := ctpo+binomial (sr+ll2+p+max (0,sign (−ll2)),sr+ll2):
    ctl := ctl+ll2+1: L3seq ():
  od:
od:

Tt := Array ([seq (j,j=1..smx)]):
Lm := M1:
for L5t from 2*M2−2 by −2 to 0 do
  S5 := [seq (0,j=1..L5t+1)]:
  ct := 1: ll4 := s: p := 1:
  for l5 to binomial (s+L5t,L5t) do
    for lm2 from 0 to iquo (ll4−1,2) do
      ct := ct+binomial (ll4+Lm,Lm)−binomial (ll4−lm2−1+Lm,Lm):
      dml := binomial (ll4+Lm,Lm+1):
      for lml from ll4−lm2−1 by −1 to 0 do
        lmmx := binomial (lml+Lm−1,Lm−1):
        dmcm := dml−binomial (lml+lm2+Lm,Lm+1):
        for j from ct to ct+lmmx−1 do
          temp := Tt[j]: Tt[j] := Tt[j+dmcm]:
          Tt[j+dmcm] := temp:
        od: ct := ct+lmmx:
```

(*continued on next page*)

```
      od: ll4 := ll4-1:
    od:
    ct := ct+binomial (ll4+Lm+1,Lm+1):
    ll4 := ll4+lm2-1:
    if ll4 =0 then
      l5 := l5+p: ct := ct+p: S5[p+1] := S5[p+1]+1: ll4 := S5[p]:
      S5[p] := 0: p := max (0,sign (1-ll4)*p)+1:
    else S5[1] := S5[1]+1: fi:
    od: Lm := Lm+2:
  od:
  for j from 1 to M2 do
    for k from 2 to Ku[M1+j] do
      Rh[L+j,k,s] := Array ([seq (Rh[M1+j,k,s][Tt[i]],i=1..smx)]):
      Ih[L+j,k,s] := Array ([seq (-Ih[M1+j,k,s][Tt[i]],i=1..smx)]):
    od:
  od:
  for j from 1 to M4 do
    for k from 2 to Ku[L-M4+j] do
      Rh[M-M4+j,k,s] := Array ([seq (Rh[L-M4+j,k,s][Tt[i]],i=1..smx)]):
      Ih[M-M4+j,k,s] := Array ([seq (-Ih[L-M4+j,k,s][Tt[i]],i=1..smx)]):
    od:
  od:
  T[s] := copy (Tt):
  if s =Ord then L := M1+M2: fi:
  for j to L do Rht[j] := Array (1..smx): Iht[j] := Array (1..smx): od:
  for m from 2 to min (s,N) do
    sm := s-m:
    for mi to SizeIndex[m] do
      kV := Index[m,mi][1]: Iv := Index[m,mi][2]: Qv := Index[m,mi][3]:
      jc := Index[m,mi][4]: jr := Index[m,mi][5]: ji := Index[m,mi][6]:
      slg := jc+jr: l3mx := binomial (sm+slg-1,slg-1):
      ll2 := sm: p := 1: S3 := [seq (0,i=1..slg+1)]:
      for l3 to l3mx do
        Sv := [ll2+Qv[1],seq (S3[j]+Qv[j+1],j=1..slg-1)]:
        q1r := copy (Rh[Iv[1],kV[1],Sv[1]]):
        q1i := copy (Ih[Iv[1],kV[1],Sv[1]]):
        s1 := Sv[1]:
        for j from 2 to jc do
          qp := L3product (s1,Sv[j],
               Rh[Iv[j],kV[j],Sv[j]],Ih[Iv[j],kV[j],Sv[j]]):
          q1r := copy (qp[1]): q1i := copy (qp[2]): s1 := s1+Sv[j]:
        od:
        s1mx := binomial (s1+Mc-1,Mc-1):
        if ji =2 then
          if jc > 1 then
        q3r := Array ([seq (q1r[T[s1][i]],i=1..s1mx)]):
        q3i := Array ([seq (-q1i[T[s1][i]],i=1..s1mx)]):
          else ivc := Qc[Iv[1]]:
        q3r := copy (Rh[ivc,kV[1],Sv[1]]):
        q3i := copy (Ih[ivc,kV[1],Sv[1]]):
      fi:
    fi:
    for i to ji do
      s1c := s1:
      for j from max (jc,1)+1 to slg do
        qp := L3product (s1c,Sv[j],
            Rh[Iv[j],kV[j],Sv[j]],Ih[Iv[j],kV[j],Sv[j]]):
        q1r := copy (qp[1]): q1i := copy (qp[2]):
        s1c := s1c+Sv[j]:
```

```
    od:
    lfa := Coef[m, (−1)^(i−1)∗mi]:
    for l to nops (lfa[1]) do
      jl := lfa[1,l]:
      if jl > L then break: fi:
      Rht[jl] := Array ([seq (Rht[jl][j]+lfa[2,l]∗q1r[j]
              −lfa[3,l]∗q1i[j],j=1..smx)]):
      Iht[jl] := Array ([seq (Iht[jl][j]+lfa[2,l]∗q1i[j]
            +lfa[3,l]∗q1r[j],j=1..smx)]):
    od:
    if ji =2 then q1r := copy (q3r): q1i := copy (q3i): fi:
  od: L3seq ():
   od:
  od:
od:
for sl from 2 to s−1 do
  ll2 := sl: sr := s−sl: l3rmx := binomial (sr+Mc−2,Mc−2):
  S3 := [seq (0,i=1..Mc−1)]: ctpo := 1: p := 1: ctl := 1:
  for l3 to binomial (sl+Mc−2,Mc−2) do
    for j to Mc do
      Lslr[j] := [seq (Ren[j,sl][i],i=ctl..ctl+ll2)]:
      Lsli[j] := [seq (Imn[j,sl][i],i=ctl..ctl+ll2)]:
    od:
    S3r := [seq (0,i=1..Mc−1)]:
    ll2r := sr: ctp := ctpo: pr := 1: ctr := 1:
    for l3r to l3rmx do
      for j to L do
        for wri to Mc do
          jw := Qb[wri]:
          Lsrr := [seq (dRh[j,sr+1,wri][i],i=ctr..ctr+ll2r)]:
          Lsri := [seq (dIh[j,sr+1,wri][i],i=ctr..ctr+ll2r)]:
          for jl to ll2+ll2r+1 do
            sb := max (1,jl−ll2): sp := min (ll2r+1,jl):
            Lsrt[wri][jl] := add (Lsrr[i]∗Lslr[jw][jl+1−i]
              −Lsri[i]∗Lsli[jw][jl+1−i],i=sb..sp):
            Lsit[wri][jl] := add (Lsrr[i]∗Lsli[jw][jl+1−i]
              +Lsri[i]∗Lslr[jw][jl+1−i],i=sb..sp):
        od:
      od:
      for i from ctp to ctp+ll2+ll2r do
        Rht[j][i] := Rht[j][i]−add (Lsrt[wri][i−ctp+1],wri=1..Mc):
        Iht[j][i] := Iht[j][i]−add (Lsit[wri][i−ctp+1],wri=1..Mc):
    od:
  od:
  ctp := i: ctr := ctr+ll2r+1:
  if ll2r =0 then
    ctp := ctp−binomial (S3r[pr]+1,2)−S3r[pr]∗ll2:
    temp := ll2+S3[1]:
    for i from 2 to pr do
      ctp := ctp+binomial (temp+i,1+i)
              −binomial (temp+S3r[pr]+i−1,1+i):
      temp := temp+S3[i]:
    od:
    ctp := ctp+binomial (temp+S3r[pr]+i−1,i):
    S3r[pr+1] := S3r[pr+1]+1: ll2r := S3r[pr]−1:
    S3r[pr] := 0: pr := max (0,sign (−ll2r))∗pr+1:
    else S3r[1] := S3r[1]+1: ll2r := ll2r−1: fi:
  od:
```

(continued on next page)

```
      ctpo := ctpo+binomial (sr+ll2+p+max (0,sign (−ll2)),sr+ll2):
      ctl := ctl+ll2+l: L3seq ():
    od:
  od:

  lic := Array (l..smx):
  S3 := [seq (0,i=l..Mc)]: p := l: ll2 := s:
  for l5 to smx do
    S5 := [ll2,op (S3)]:
    lic[l5] := add (IEf[i]*(S5[2*i−Ml−l]−S5[2*i−Ml]),i=Ml+l..Ml+M2):
    L3seq ():
  od:
  for j to Ml+M2 do
    Ren[j,s] := Array (l..smx): Imn[j,s] := Array (l..smx):
    Rh[j,l,s] := Array (l..smx): Ih[j,l,s] := Array (l..smx):
    Iy := −IEf[j]:
    for l5 to smx do
      Il := Iy+lic[l5]:
      if Il <> 0 then
        Rh[j,l,s][l5] := factor (Iht[j][l5]/Il):
        Ih[j,l,s][l5] := −factor (Rht[j][l5]/Il):
        else Ren[j,s][l5] := factor (Rht[j][l5]):
      Imn[j,s][l5] := factor (Iht[j][l5]): fi:
    od:
  od:
  if s < Ord then
    for j from Ml+M2+l to L do
      Rh[j,l,s] := Array (l..smx): Ih[j,l,s] := Array (l..smx):
      Ry := −REf[j]: Iy := −IEf[j]:
      for l5 to smx do
        Il := Iy+lic[l5]: temp := Ry*Ry+Il*Il:
        Rh[j,l,s][l5] := factor ((Rht[j][l5]*Ry+Iht[j][l5]*Il)/temp):
        Ih[j,l,s][l5] := factor ((Iht[j][l5]*Ry−Rht[j][l5]*Il)/temp):
      od:
  od:
  for j from Ml+l to Ml+M2 do
      Ren[M2+j,s] := Array ([seq (Ren[j,s][Tt[i]],i=l..smx)]):
      Imn[M2+j,s] := Array ([seq (−Imn[j,s][Tt[i]],i=l..smx)]):
      Rh[L−Ml+j,l,s] := Array ([seq (Rh[j,l,s][Tt[i]],i=l..smx)]):
      Ih[L−Ml+j,l,s] := Array ([seq (−Ih[j,l,s][Tt[i]],i=l..smx)]):
    od:
  for j from L−M4+l to L do
      Rh[M2+M4+j,l,s] := Array ([seq (Rh[j,l,s][Tt[i]],i=l..smx)]):
      Ih[M2+M4+j,l,s] := Array ([seq (−Ih[j,l,s][Tt[i]],i=l..smx)]):
    od:
    qdemx := binomial (s+Mc−2,Mc−l):
    for wri to Mc do
      for j to L do
        dRh[j,s,wri] := Array (l..qdemx):
        dIh[j,s,wri] := Array (l..qdemx):
      od:
      temp := Mc−wri:
      Sil := [seq (0,j=l..temp+2)]:
      lsimx := binomial (s+temp,temp);
      lli := s: kst := l: oml := 0: po := l:
      for lsi from l to lsimx do
        if wri > l then
          oml := oml+binomial (lli+wri−2,wri−2):
          for li from l to lli do
```

```
        limx := binomial (lli−li+wri−2,wri−2):
        for j from kst to kst+limx−1 do
          for j1 to L do
            dRh[j1,s,wri][j] := li*Rh[j1,l,s][j+om1]:
              dIh[j1,s,wri][j] := li*Ih[j1,l,s][j+om1]:
          od:
        od: kst := kst+limx:
      od:
    else
      for j1 to L do
        dRh[j1,s,wri][kst] := lli*Rh[j1,l,s][kst+om1]:
        dIh[j1,s,wri][kst] := lli*Ih[j1,l,s][kst+om1]:
      od: kst := kst+1:
    fi:
    if lli =1 then
      om1 := om1+po: Sil[po+1] := Sil[po+1]+1: lli := Sil[po]:
      Sil[po] := 0: lsi := lsi+po: po := max (0,sign (1−lli)*po)+1:
      else Sil[1] := Sil[1]+1: lli := lli−1: fi:
    od:
  od:
  fi:
od:
ZC := [seq (0,j=1..M1)]:
RC := [seq (0,j=1..M2)]:
IC := [seq (IEf[M1+j],j=1..M2)]:
for s from 2 to Ord do
  l12 := s: p := 1: l3mx := binomial (s+Mc−1,Mc−1):
  S3 := [seq (0,i=1..Mc)]:
  for l3 to l3mx do
    Sl := [l12,op (S3)]: term := 1:
    for j from 1 to M1 do term := term*y[j]^Sl[j]: od:
    thetan := 0:
    for j from M1+1 to M1+M2 do
      term := term*r[j−M1]^(Sl[2*j−M1−1]+Sl[2*j−M1]):
      thetan := thetan+theta[j−M1]*(Sl[2*j−M1−1]−Sl[2*j−M1]):
    od:
    for j from 1 to M1 do
      ZC[j] := ZC[j]+term*(factor (Ren[j,s][l3])*cos (thetan)
            −factor (Imn[j,s][l3])*sin (thetan)):
    od:
    for j from 1 to M2 do
      RC[j] := RC[j]+term*(factor (Ren[j+M1,s][l3])*cos (thetan−theta[j])
            −factor (Imn[j+M1,s][l3])*sin (thetan−theta[j])):
      IC[j] := IC[j]+term/r[j]*(factor (Ren[j+M1,s][l3])*sin (thetan−theta[j])
            +factor (Imn[j+M1,s][l3])*cos (thetan−theta[j])):
    od:
    L3seq ():
  od:
  od:
for i from 1 to M1 do
  ZC[i] := combine (ZC[i],trig): print (‘‘y",i,ZC[i]):
od:
for i from 1 to M2 do
    RC[i] := combine (RC[i],trig): print (‘‘r",i,RC[i]):
    IC[i] := combine (IC[i],trig): print (‘‘theta",i,IC[i]):
od:
save M1,M2,ZC,RC,IC, output:
```

References

[1] Chow S-N, Li C, Wang D. Normal forms and bifurcations of planar vector fields. Cambridge: Cambridge University Press; 1994.
[2] Kahn P, Zarmi Y. Nonlinear dynamics: exploration through normal forms. NY: John Wiley Sons; 1998.
[3] Han M, Yu P. Normal forms, Melnikov functions and bifurcations of limit cycles. London: Springer; 2012.
[4] Stolovitch L. Progress in normal form theory. Nonlinearity 2009;22:R77–99.
[5] Carr J. Applications of center manifold theory. NY: Springer-Verlag; 1981.
[6] Bi Q, Yu P. Symbolic computation of normal forms for semi-simple cases. J Comput Appl Math 1999;102:195–220.
[7] Yu P. A simple and efficient method for computing center manifold and normal forms associated with semisimple cases. Dyn Continuous Discrete Impuls Syst Ser B: Appl Algorithm 2003;10:273–86.
[8] Kuznetsov Y. Practical computation of normal forms on center manifolds at degenerate Bogdanov–Takens bifurcations. Int J Bif Chaos 2005;15(11):3535–46.
[9] Tian Y, Yu P. An explicit recursive formula for computing the normal form and center manifold of n-dimensional differential systems associated with Hopf bifurcation. Int J Bif Chaos 2013;23(6). 1350104 (18 pages).
[10] Leung A, Ge T. An algorithm for higher order Hopf normal form. J Shock Vib 1995;2:307–19.
[11] Zhang W, Huseyin K, Ye M. On the computation of the coefficients associated with high order normal forms. J Sound Vib 2000;232:525–40.
[12] Yu P. Computation of the normal forms via a perturbation technique. J Sound Vib 1998;211(1):19–38.
[13] Giné J, Santallusia X. On the Poincaré–Lyapunov constants and the Poincaré series. Appl Math (Warsaw) 2001;28(1):17–30.
[14] Guckenheimer J, Holmes P. Nonlinear oscillations, dynamical systems, and bifurcations of vector fields. NY: Springer-Verlag; 1983.
[15] Gazor M, Mokhtari F, Sanders A. Normal forms for Hopf-Zero singularities with nonconservative nonlinear part. J Differential Equ 2013;254:1571–81.
[16] Zhang Q, Leung A. Normal form of double Hopf bifurcation in forced oscillators. J Sound Vib 2000;231(4):1057–69.
[17] Yu P. Symbolic computation of normal forms for resonant double Hopf bifurcations using multiple time scales. J Sound Vib 2001;247(4):615–32.