A REDUCE program for the normalization of polynomial Hamiltonians ☆

Yu.A. Ukolov [a], N.A. Chekanov [a], A.A. Gusev [b], V.A. Rostovtsev [c], S.I. Vinitsky [d,*], Y. Uwano [e]

[a] *Belgorod State University, Studentcheskaja St. 14, Belgorod 308007, Russia*
[b] *Scientific Center for Applied Research, Joint Institute for Nuclear Research, Dubna, Moscow Region 141980, Russia*
[c] *Laboratory of Information Technologies, Joint Institute for Nuclear Research, Dubna, Moscow Region 141980, Russia*
[d] *Bogoliubov Laboratory of Theoretical Physics, Joint Institute for Nuclear Research, Dubna, Moscow Region 141980, Russia*
[e] *Department of Applied Mathematics and Physics, Kyoto University, Kyoto 606-8501, Japan*

Abstract

The program LINA01 is proposed for the direct and the inverse normalization of Hamiltonian systems and for the calculation of formal integrals of motion of them. The calculations required in LINA01 are made on the basis of Lie canonical transformation method. The program package of LINA01 is written on REDUCE.

Program summary

Title of program: LINA01
Catalogue identifier: ADUV
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/ADUV
Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland
Computer: IBM PC PENTIUM 4/2.40 GHz 512 Mb
Operating systems under which the program has been tested: Windows XP
Programming language used: REDUCE vs. 3.7
No. of lines in distributed program, including test data, etc.: 485
No. of bytes in distributed program, including test data, etc.: 4320
Distribution format: tar.gz

* Corresponding author.
E-mail addresses: ukolov@bsu.edu.ru (Yu.A. Ukolov), chekanov@bsu.edu.ru (N.A. Chekanov), gusev_baatar@mail.ru (A.A. Gusev), rost@jinr.ru (V.A. Rostovtsev), vinitsky@thsun1.jinr.ru (S.I. Vinitsky), uwano@amp.i.kyoto-u.ac.jp (Y. Uwano).

Nature of physical problem. The transformation bringing a given Hamiltonian function into the normal form (namely, the normalization) is one of the conventional methods for non-linear Hamiltonian systems [A.J. Lichtenberg, M.A. Lieberman, Regular and Stochastic Motion, Springer-Verlag, Berlin, 1983; G.D. Birkhoff, Dynamical Systems, A.M.S. Colloquium Publications, New York, 1927; F. Gustavson, Astron. J. 71 (1966) 670; G.I. Hori, Astron. Soc. Japan 18 (1966) 287; A. Deprit, Cel. Mech. 1 (1969) 12; A.A. Kamel, Cel. Mech. 3 (1970) 90]. Recently, beyond classical mechanics, the normal form method has been applied to quantization of chaotic Hamiltonian systems with the aim of finding quantum signature of chaos [L.E. Reichl, The Transition to Chaos. Conservative Classical Systems: Quantum Manifestations, Springer, New York, 1992]. Besides those utilities, the normalization requires quite cumbersome algebraic calculations of polynomials, so that the computer algebraic approach is worth studying to promote further investigations around the normalization together with the ones around the inverse normalization.

Method of solution. The canonical transformation proceeding the normalization is expressed in terms of the Lie transformation power series, which is also referred to as the Hori–Deprit transformation. After (formal) power series expansion as above, the fundamental equation of the normalization is solved for the normal form together with the generating function of transformation recursively from degree-3 to the degree desired to be normalized. The generating function thus obtained is applied to the calculation of (formal) integrals of motion.

Restrictions due to the complexity of the problem. The computation time rises in a combinatorial manner as the desired degree of normalization does. Especially, such a combinatorial growth of computation is more significant in the inverse normalization than in the direct one. The hardware (processor and memory, for example) available for the computation may restrict either the degree of normalization or the computation time.

Keywords: Computer algebra; REDUCE; Hamiltonian systems; Normal form; Lie canonical transformation

1. Introduction

It is widely known that, in general, a given Hamiltonian system is hardly solved in an explicit way:[1] More precisely, one cannot reach to the solution of a given Hamiltonian system within quadrature, except for *integrable* systems; the Kepler problem, the harmonic oscillators, two-center-of-mass problem and the specific tops are typical and historically well-known examples of integrable systems. The various methods, including numerical ones [2–8], have been hence proposed to analyze non-integrable Hamiltonian systems.

As one of the conventional methods, the normalization of Hamiltonian systems around stable equilibrium points is well known, which is originated by Birkhoff [9,10] and by Gustavson [11]. The normalization of n-degree-of-freedom Hamiltonian systems is outlined in what follows [1]: Let the Hamiltonian for a given system is expanded in (formal)[2] power series form,

$$\mathcal{H}(q,p)=\sum_{\nu=1}^{n}\frac{\omega_\nu}{2}\left(p_\nu^2+q_\nu^2\right)+\sum_{j=3}^{\infty}\mathcal{H}_j(q,p)\quad(\omega_\nu>0,\ \nu=1,2,\ldots,n) \tag{1a}$$

around an equilibrium point, where $(q,p)=(q_1,\ldots,q_n,p_1,\ldots,p_n)$ are the canonical coordinates and each $H_j(q,p)$ is the homogeneous part of degree-j ($j=3,4,\ldots$). The normalization of $\mathcal{H}(q,p)$ into its normal form,

$$\mathcal{G}(\xi,\eta)=\sum_{\nu=1}^{n}\frac{\omega_\nu}{2}\left(\eta_\nu^2+\xi_\nu^2\right)+\sum_{j=3}^{\infty}\mathcal{G}_j(\xi,\eta) \tag{1b}$$

[1] One must be very careful that this statement does not contradict with the theorem on the existence of local solutions for differential equations.

[2] The power series appearing in the normalization may be formal, i.e. the convergent radius may be naught.

subject to the Poisson commuting property,

$$\left\{ \sum_{\nu=1}^{n} \frac{\omega_\nu}{2}(\eta_\nu^2+\xi_\nu^2), \mathcal{G}_j(\xi,\eta) \right\}_{\xi,\eta} = 0 \quad (j=3,4,\ldots), \tag{1c}$$

is made through a suitable canonical transformation $(q, p) \mapsto (\xi, \eta)$ giving rise in the form,[3]

$$\xi = \frac{\partial \mathcal{W}}{\partial \eta}(q,\eta), \qquad p = \frac{\partial \mathcal{W}}{\partial q}(q,\eta) \quad \text{with } \mathcal{W}(q,\eta) = \sum_{\nu=1}^{n} q_\nu \eta_\nu + \sum_{j=3}^{\infty} \mathcal{W}_j(q,\eta), \tag{1d}$$

where $\mathcal{G}_j(\xi,\eta)$ and $\mathcal{W}_j(q,\eta)$ are homogeneous polynomials of degree-j in $\mathcal{G}(\xi,\eta)$ and in $\mathcal{W}(q,\eta)$, respectively. The bracket $\{\cdot,\cdot\}_{\xi,\eta}$ denotes the Poisson bracket in the variables (ξ,η) defined to be

$$\{f_1(\xi,\eta), f_2(\xi,\eta)\}_{\xi,\eta} = \sum_{\nu=1}^{n} \left(\frac{\partial f_1}{\partial \eta_\nu}\frac{\partial f_2}{\partial \xi_\nu} - \frac{\partial f_1}{\partial \xi_\nu}\frac{\partial f_2}{\partial \eta_\nu} \right), \tag{1e}$$

for any smooth functions, $f_1(\xi,\eta)$ and $f_2(\xi,\eta)$, throughout this paper.

Remark. The minus sign is applied to the right-hand side of (1d) to define the Poisson bracket in the standard text [1] of the normalization and in the [12] organizing the concrete formulation of the inverse problem by the author (YUw).

The normalization of $\mathcal{H}(q,p)$ into $\mathcal{G}(\xi,\eta)$ through a transformation of the form (1d) amounts to determining $\mathcal{G}(\xi,\eta)$ and $\mathcal{W}(q,\eta)$ to satisfy

$$\mathcal{H}\left(q, \frac{\partial \mathcal{W}}{\partial q}(q,\eta)\right) = \mathcal{G}\left(\frac{\partial \mathcal{W}}{\partial \eta}(q,\eta), \eta\right). \tag{1f}$$

Remark. If one put the additional claim on $\mathcal{W}(q,\eta)$ to be in the image of the linear operator, $D(q,\eta)$, defined by (15) with $\xi = q$, then the pair of $\mathcal{G}(\xi,\eta)$ and $\mathcal{W}(q,\eta)$ is determined uniquely for a given $\mathcal{H}(q,p)$ [12].

A number of computer programs (see [11–17], for example) have been proposed for the normalization, which are organized to solve (1d) for $\mathcal{G}(\xi,\eta)$ together with $\mathcal{W}(\xi,\eta)$. Although those programs have contributed a lot to various works around the normalization, they involve an essential difficulty for computations if one wishes to have the normalization in the cases of higher degree of freedom (large ν in (1b)), of higher order normalization (large j in (1b)), and of $\mathcal{H}(q,p)$ being highly parametric. A plausible account for this difficulty is that the canonical transformation (1d), (1e) applied to the normalization takes an implicit expression; the *old* position variables $q = (q_\nu)$ and the *new* momentum variables $\eta = (\eta_\nu)$ are mixed up together in (1d), (1e).

As another way of dealing with canonical transformations, the expression based on the Lie transformation theory has been proposed by Hori [19] and by Deprit [20], which will be referred to as the Lie transformation method henceforth. The Lie transformation method has been successfully applied to practical studies like in [19–28]. The aim of this paper is to present a new computer program, LINA01 (Lie Normalization Algorithm), for the normalization, which is organized on the basis of Lie transformation method as the name of the program suggests. In Section 2, a theoretical setting-up of the normalization is made on the basis of the Lie transformation method, with which the program LINA01 is written up. Section 3 is devoted to the description of the program and Section 4 is for showing an efficiency of LINA01 in three intuitive examples, the one-dimensional perturbed harmonic oscillator with a homogeneous quartic potential, a parametric two-dimensional perturbed oscillator with a homogeneous cubic potential and a parametric three-dimensional perturbed oscillator with a homogeneous cubic potential.

[3] The function $\mathcal{W}(q,\eta)$ in (3) is called a generating function of the 2nd type.

2. The Lie transformation method of the normalization

2.1. Setting-up

Let (q, p) be the Cartesian coordinates in $\mathbf{R}^n \times \mathbf{R}^n$ with

$$q = (q_1, q_2, \ldots, q_n), \qquad p = (p_1, p_2, \ldots, p_n). \tag{2}$$

Let the initial n-degree-of-freedom Hamiltonian be written in polynomial form of degree-$(j_{\max} + 2)$,

$$H(q, p; \epsilon) = H_0(q, p) + \sum_{j=1}^{j_{\max}} \frac{\epsilon^j}{j!} H_j(q, p), \tag{3a}$$

where each $H_j(q, p)$ $(j = 0, 1, \ldots)$ is the homogeneous polynomial of degree-$(j + 2)$ in (q, p) with

$$H_0(q, p) = \sum_{\nu=1}^{n} \frac{\omega_\nu}{2}(q_\nu^2 + p_\nu^2) \quad (\omega_\nu > 0,\ \nu = 1, \ldots, n). \tag{3b}$$

The ϵ is used as a formal parameter not only in (2) but also throughout this paper.

The normalization by the Lie transformation method is made in the following way [27]. Let $W(\xi, \eta; \epsilon)$ be the generating function in the form,

$$W(\xi, \eta; \epsilon) = \sum_{s=0}^{s_{\max}} \frac{\epsilon^s}{s!} W_{s+1}(\xi, \eta), \tag{4}$$

where each $W_j(\xi, \eta)$ $(j = 1, \ldots, s_{\max} + 1)$ is a homogeneous polynomial of degree-j in (ξ, η). By $W(\xi, \eta; \epsilon)$, we generate the canonical transformation, $(q, p) \to (\xi, \eta)$, in the manner,

$$q = \xi + \sum_{s=1}^{s_{\max}} \frac{\epsilon^s}{s!} (L_W)^s \xi, \qquad p = \eta + \sum_{s=1}^{s_{\max}} \frac{\epsilon^s}{s!} (L_W)^s \eta, \tag{5}$$

where L_W is the Lie derivation,

$$L_W = \sum_{\nu=1}^{n} \left(\frac{\partial W}{\partial \eta_\nu} \frac{\partial}{\partial \xi_\nu} - \frac{\partial W}{\partial \xi_\nu} \frac{\partial}{\partial \eta_\nu} \right) = \{ W(\xi, \eta), \cdot \}_{\xi, \eta}, \tag{6}$$

associated with $W(\xi, \eta; \epsilon)$ (see remark after (1e) for the Poisson bracket). Along with the canonical transformation, $(q, p) \to (\xi, \eta)$, given in (5) and (6), we bring the Hamiltonian $H(q, p; \epsilon)$ into its normal form Hamiltonian,

$$G(\xi, \eta; \epsilon) = G_0(\xi, \eta) + \sum_{s=1}^{s_{\max}} \frac{\epsilon^s}{s!} G_s(\xi, \eta), \quad \text{with } G_0(\xi, \eta) = \sum_{\nu=1}^{n} \frac{\omega_\nu}{2} \left(\eta_\nu^2 + \xi_\nu^2 \right), \tag{7}$$

that satisfies the basic equation,

$$H\left(q(\xi, \eta; \epsilon), p(\xi, \eta; \epsilon); \epsilon\right) = G(\xi, \eta; \epsilon) \quad \text{up to degree-}s_{\max} \text{ in } \epsilon, \tag{8}$$

together with the normal-form condition,

$$\left\{ G_0(\xi, \eta), G_s(\xi, \eta) \right\} = 0 \quad (s = 1, \ldots, s_{\max}), \tag{9}$$

where each $G_s(\xi, \eta)$ denotes the homogeneous polynomial part of degree-s in (ξ, η).

Remark. The restriction 'up to degree-$s_{\max}$ in ϵ' in (8) is equivalent to 'up to degree-$(s_{\max}+2)$ in (ξ, η)'. If setting ϵ to be equal to 1 in (2) and (7), one gets the relations,

$$\begin{aligned} &H_j(q, p) = j!\mathcal{H}_{j+2}(q, p) \quad (j = 0, \ldots, j_{\max}), \\ &G_s(\xi, \eta) = s!\mathcal{G}_{s+2}(q, p) \quad (s = 0, \ldots, s_{\max}), \end{aligned} \tag{10}$$

between the setting above and the conventional setting introduced in Section 1.

2.2. *Solving the basic equation*

The normal form $G(\xi, \eta; \epsilon)$ is determined together with $W(\xi, \eta; \epsilon)$ by equating the 'coefficient functions in (ξ, η)' of ϵ^ss on the left-hand side of (8) with those on the right-hand side of (8). By calculation, we obtain the recursion formula,

$$G_s(\xi, \eta) - \{H_0|_{(\xi,\eta)}, W_s(\xi, \eta)\} = H_s|_{(\xi,\eta)} - T_s(\xi, \eta) \quad (s = 0, 1, 2, \ldots, s_{\max}), \tag{11}$$

where $T_s(\xi, \eta)$s are given by

$$T_s(\xi, \eta) = \sum_{j=1}^{s-1} \left(C_{s-1}^{j-1} (L_{W_j} H_{s-j})(\xi, \eta) + C_{s-1}^{j} K_{j,s-j}(\xi, \eta)\right) + (L_{W_s} H_0)(\xi, \eta) \tag{12}$$

$(s = 0, 1, \ldots, s_{\max})$ with

$$K_{j,i}(\xi, \eta) = (L_{W_j} G_i)(\xi, \eta) - \sum_{m=1}^{j-1} C_{j-1}^{m-1} (L_{W_m} K_{j-m,i})(\xi, \eta) \tag{13}$$

$(i, j = 1, \ldots, s-1)$. The symbol, $C_k^{k'}$, appearing in (12) and (13) stands for the binomial coefficient [29]; in other conventions, we express it as

$$C_k^{k'} = {}_kC_{k'} = \binom{k}{k'} \quad (k' = 0, 1, \ldots, k). \tag{14}$$

We are now in a position to solve Eq. (11) recursively for $G_s(\xi, \eta)$ and $W_s(\xi, \eta)$ $(s = 0, 1, \ldots, s_{\max})$. A key to solving (11) is to decomposition of any homogeneous polynomial into the kernel component and the image component of the linear differential operator

$$D(\xi, \eta) = \sum_{\nu=1}^{n} \omega_\nu \left(\eta_\nu \frac{\partial}{\partial \xi_\nu} - \xi_\nu \frac{\partial}{\partial \eta_\nu}\right) = \{G_0(\xi, \eta), \cdot\}_{\xi,\eta} = \{H_0|_{(\xi,\eta)}, \cdot\}_{\xi,\eta}, \tag{15}$$

by which the normal-form condition (9) can be rewritten as

$$(D(\xi, \eta) G_s)(\xi, \eta) = 0 \quad (s = 0, \ldots, s_{\max}). \tag{16}$$

Then it is a very easy matter to see

$$G_s(\xi, \eta) \in \ker D(\xi, \eta) \quad (s = 1, \ldots, s_{\max}) \tag{17a}$$

and

$$\{H_0|_{(\xi,\eta)}, W_s(\xi, \eta)\} = (D(\xi, \eta) W_s)(\xi, \eta) \in \text{image } D(\xi, \eta) \quad (s = 1, \ldots, s_{\max}). \tag{17b}$$

The decomposition,

$$T_s(\xi, \eta) - H_s|_{(\xi,\eta)} = N_s(\xi, \eta) + R_s(\xi, \eta),$$

$$N_s(\xi,\eta)\in\ker D(\xi,\eta),\qquad R_s(\xi,\eta)\in\operatorname{image} D(\xi,\eta)\quad(s=1,\dots,s_{\max}),\tag{18}$$

is put together with (17) to yield

$$G_s(\xi,\eta)=-N_s(\xi,\eta)\quad(s=1,\dots,s_{\max})\tag{19a}$$

and

$$\big\{H_0|_{(\xi,\eta)},W_s(\xi,\eta)\big\}=\big(D(\xi,\eta)W_s\big)(\xi,\eta)=R_s(\xi,\eta)\quad(s=1,\dots,s_{\max}).\tag{19b}$$

2.3. Algorithm

Now that we have the expression (18) of the solution of basic equation (8) with (9), we move on to present algorithmic expression of the solution. Since the major part of realizing the solution from (18) in a explicit way is based on the polynomial computer algebra, it would be better to regard ξ and η as formal complex variables in this subsection. We introduce another pair of formal complex variables (x,y)

$$\xi=\frac{1}{\sqrt{2}}(x+\sqrt{-1}\,y),\qquad \eta=\frac{\sqrt{-1}}{\sqrt{2}}(x-\sqrt{-1}\,y),\tag{20}$$

so that the differential operator $D(\xi,\eta)$ (see (15)) is expressed in a diagonal form:

$$D(\xi,\eta)=\tilde{D}(x,y)=\sqrt{-1}\sum_{\nu}\omega_\nu\left(x_\nu\frac{\partial}{\partial x_\nu}-y_\nu\frac{\partial}{\partial y_\nu}\right).\tag{21}$$

Further, using the transformation (20) with (11), (15) and (17b), we can put Eqs. (11), (18a) and (18b) into the form,

$$\tilde{G}_s(x,y)-\big(\tilde{D}(x,y)\tilde{W}_s\big)(x,y)=\tilde{H}_s(x,y)-\tilde{T}_s(x,y)\quad(s=1,\dots,s_{\max}),\tag{22}$$

$$\tilde{G}_s(x,y)=-\tilde{N}_s(x,y)\quad(s=1,\dots,s_{\max})\tag{23a}$$

and

$$\big(\tilde{D}(x,y)\tilde{W}_s\big)(x,y)=\tilde{R}_s(x,y)\quad(s=1,\dots,s_{\max}),\tag{23b}$$

respectively, where

$$\begin{aligned}&\tilde{H}_s(x,y)=H_s|_{(\xi,\eta)},\qquad &&\tilde{G}_s(x,y)=G_s(\xi,\eta),\\ &\tilde{W}_s(x,y)=W_s(\xi,\eta),\qquad &&\tilde{T}_s(x,y)=T_s(\xi,\eta),\\ &\tilde{N}_s(x,y)=N_s(\xi,\eta),\qquad &&\tilde{R}_s(x,y)=R_s(\xi,\eta)\quad(s=1,\dots,s_{\max})\end{aligned}\tag{24}$$

(see (18) for $N(\xi,\eta)$ and $R(\xi,\eta)$). To realize an explicit expression of $\tilde{G}_s(x,y)$ and $\tilde{W}_s(x,y)$, we define the monomial-basis, $\{\Phi_{(s)}^{\ell,m}\}$, to be

$$\Phi_s^{lm}=x^l y^m=x_1^{l_1}x_2^{l_2}\cdots x_n^{l_n}y_1^{m_1}y_2^{m_2}\cdots y_n^{m_n}\quad(|l|+|m|=s+2),\tag{25a}$$

where l and m are the multi-indices of the form,

$$l=(l_1,l_2,\dots,l_n),\qquad m=(m_1,m_2,\dots,m_n)\tag{25b}$$

with the conventional notations,

$$x^l=x_l^{l_1}\cdots x_n^{l_n},\quad y^l=y_l^{m_1}\cdots y_n^{m_n},\quad |l|=\sum_{\nu=1}^{n}l_\nu,\quad |m|=\sum_{\nu=1}^{n}m_\nu.\tag{25c}$$

The operator $\tilde{D}(x, y)$ acts on Φ_s^{lm} in the manner

$$\tilde{D}(x, y)\Phi_s^{lm} = \sqrt{-1}\sum_{\nu}\omega_{\nu=1}^{n}(m_\nu - l_\nu)\Phi_s^{lm} \quad \begin{pmatrix} s = 0, 1, \ldots, s_{\max} \\ |l| + |m| = s + 2 \end{pmatrix}. \tag{26}$$

If restricted to the space homogeneous polynomials of degree-$(s+2)$, the kernel and the image of $\tilde{D}(x, y)$ are then characterized to be

$$\ker \tilde{D}(x, y) = \mathrm{span}\left\{ \Phi_s^{lm} \,\middle|\, \sum_{\nu=1}^{n}\omega_\nu(m_\nu - l_\nu) = 0 \right\} \tag{27a}$$

and

$$\mathrm{image}\, \tilde{D}(x, y) = \mathrm{span}\left\{ \Phi_s^{lm} \,\middle|\, \sum_{\nu=1}^{n}\omega_\nu(m_\nu - l_\nu) \neq 0 \right\}. \tag{27b}$$

We are at the final stage to provide an explicit expression of $\tilde{G}_s(x, y)$ and $\tilde{W}_s(x, y)$ given by (24). In view of (27), it is of great use to prepare a pair of the sets of multi-indices,

$$\mathcal{I}_s^N = \left\{ (l, m) \,\middle|\, |l| + |m| = s + 2,\ \sum_{\nu=1}^{n}\omega_\nu(m_\nu - l_\nu) = 0 \right\} \tag{28a}$$

and

$$\mathcal{I}_s^R = \left\{ (l, m) \,\middle|\, |l| + |m| = s + 2,\ \sum_{\nu=1}^{n}\omega_\nu(m_\nu - l_\nu) \neq 0 \right\}. \tag{28b}$$

It is worth pointing out here that if $\{\omega_\nu\}$ is independent over the field of the rational numbers (so-called, the non-resonant case), we see from (28) that $\ker \tilde{D}(x, y)$ is spanned by $x^l y^m$ with $l = m$. It means that in the non-resonant case the normal form can be regarded as a polynomial of $\{\xi_j^2 + \eta_j^2\}$.

On expanding $\tilde{G}_s(x, y)$, $\tilde{W}_s(x, y)$, $\tilde{N}_s(x, y)$ and $\tilde{R}_s(x, y)$ $(s = 0, 1, \ldots, s_{\max})$ to be

$$\tilde{G}_s(x, y) = \sum_{(l,m)\in\mathcal{I}_s^N} \tilde{G}_s^{lm}\Phi_s^{lm}(x, y), \qquad \tilde{W}_s(x, y) = \sum_{(l,m)\in\mathcal{I}_s^R} \tilde{W}_s^{lm}\Phi_s^{lm}(x, y), \tag{29a}$$

and

$$\tilde{N}_s(x, y) = \sum_{(l,m)\in\mathcal{I}_s^N} \tilde{N}_s^{lm}\Phi_s^{lm}(x, y), \qquad \tilde{R}_s(x, y) = \sum_{(l,m)\in\mathcal{I}_s^R} \tilde{R}_s^{lm}\Phi_s^{lm}(x, y), \tag{29b}$$

in the monomial basis $\{\Phi_{(s)}^{\ell,m}\}$, the coefficients in $G_s(\xi, \eta)$ and $W_s(\xi, \eta)$ are solved to be

$$\tilde{G}_s^{lm} = -\tilde{N}_s^{lm} \quad \left(s = 0, 1, \ldots, s_{\max},\ (l, m) \in \mathcal{I}_s^N\right) \tag{30a}$$

and

$$\tilde{W}_s^{lm} = \frac{\tilde{R}_s^{lm}}{\sqrt{-1}\sum_{\nu=1}^{n}\omega_\nu(m_\nu - l_\nu)} \quad \left(s = 0, 1, \ldots, s_{\max},\ (l, m) \in \mathcal{I}_s^R\right). \tag{30b}$$

Remark. Precisely speaking, a kind of ambiguity is involved in the expansion of $\tilde{W}_s(x, y)$: If we add any homogeneous polynomial of degree-$(s+2)$ in $\ker \tilde{D}(x, y)$ to $\tilde{W}_s(x, y)$ with (30b), $\tilde{W}_s(x, y)$ after the addition also satisfies (23b). We are, however, able to get rid of such an ambiguity by restricting $\tilde{W}_s(x, y)$ to be in image $\tilde{D}(x, y)$ (see also [12]).

2.4. *Solving the inverse problem: an outline*

To compare this section, we wish to mention of the inverse transformation of normalization very briefly. The inversion is, in principle, made by inverting the roles of $H(q, p)$ and $G(\xi, \eta)$. Namely, we regard $H(q, p)$ as *a given normal form* and $G(\xi, \eta)$ as the solution of the inverse problem. According to the treatment above, we have to modify the expression (29a) for $\tilde{G}(x, y)$ to be

$$\tilde{G}_s(x, y) = \sum_{(l,m)\in\mathcal{I}_s^N} \tilde{G}_s^{lm} \Phi_s^{lm}(x, y) + \sum_{(l,m)\in\mathcal{I}_s^R} \tilde{G}_s^{lm} \Phi_s^{lm}(x, y), \tag{31}$$

in order that $\tilde{G}(x, y)$ express the solution to the inverse problem.

Remark. If one wish to apply the program code in the present paper to the inversion, a modification along (31) is necessary.

The solution thereby takes the form,

$$\tilde{G}_s^{lm} = \begin{cases} -\tilde{N}_s^{lm} & (s = 1, \ldots, s_{\max},\ (l, m) \in \mathcal{I}_s^N), \\ \text{chosen arbitrarily} & (s = 1, \ldots, s_{\max},\ (l, m) \in \mathcal{I}_s^R) \end{cases} \tag{32a}$$

and

$$\tilde{W}_s^{lm} = \frac{\tilde{R}_s^{lm}}{\sqrt{-1}\sum_{\nu=1}^{n} \omega_\nu (m_\nu - l_\nu)} \quad \big(s = 0, 1, \ldots, s_{\max},\ (l, m) \in \mathcal{I}_s^R\big). \tag{32b}$$

We remark again that, in this subsection, $\tilde{H}(x, y)$ plays a role of a given normal form and $\tilde{G}(x, y)$ the solution to the inverse problem. As shown in (32a), we can choose $\tilde{G}_s^{lm}$ with $(l, m) \in \mathcal{I}_s^R$ $(s = 1, \ldots, s_{\max})$ arbitrarily. This fact is a very nature of the inverse problem of the normalization.

3. Description of the program

The program LINA01 is written up in the language REDUCE according to the algorithm described above.

3.1. *The global structure of the program, LINA01*

In this section, we present the structure of LINA01 together with its elements, the main program and a set of procedures for computation. The main program is labeled with the file name LINA01.RED and the procedures around the main program is packed in the file named LINA01PROC.RED.

For executing the main program LINA01.RED, users have to prepare file LINA01IN beforehand as follows: The user sets initial data in the form of a sequence of REDUCE assignment operators, i.e. the file LINA01IN consists of lines of the type:

$$\langle variable \rangle := \langle expression \rangle; \tag{33}$$

The degrees-of-freedom, n, and frequencies, $\omega_\nu > 0$ $(\nu = 1, \ldots, n)$, of the input Hamiltonian are assigned to be the values of the variables N and WW(1), ..., WW(N), respectively. To specify the highest degree, say $j_{\max} + 2$, of the input Hamiltonian $H(q, p; 1)$ (see (3a)) that the users wish to normalize, we set the value of the variable JMAX to be equal to $j_{\max}$. Similarly, to specify the degree, say $s_{\max} + 2$, of the normalization (see (3)–(8)) that

the users wish to make, we set the value of the variable SMAX to be equal to $s_{\max}$. Then the value of each variable $H(j)$ $(j = 0, 1, \ldots,$ JMAX) in LINA01IN expresses $H_j(q, p)$.

Remark. If the initial Hamiltonian is given in the form (1a) user will take into account relation (10).

To execute LINA01.RED in the MS WINDOWS environment, we type the command `IN "LINA01.RED";` (see [18], for the available system options). The structure of the core program LINA01.RED will be described in the next subsection.

The results produced through the execution of LINA01.RED are written out in the file named LINA01OUT in the following way. Each homogeneous part, $W_s(\xi, \eta)$, of degree-$(s+2)$ in $W(\xi, \eta; 1)$ is written out to the variable W(S), S = 0, 1, ..., SMAX, and each $G_s(\xi, \eta)$ in $G(\xi, \eta; 1)$ to the variable G(S), S = 0, 1, ..., SMAX;

W(0) := ···

W(1) := ···

...

W(SMAX) := ···

G(0) := ···

G(1) := ···

...

G(SMAX) := ···

Remark. Instead of (ξ, η) and (x, y) used in Section 2, the variables (q, p) are applied to the expressions in W(0), ..., W(SMAX), G(0), ..., G(SMAX) (see TEST RUN OUTPUT), in order to reduce memory consumption through the program.

After the completion of the normal form calculation, the computation time (in ms unit) is written out to LINA01OUT.

There exist several options for checking the computation: If the value of the variable TEST is set to be 1, then the program writes out a message 'G(s) is normal form;' to LINA01OUT.

If the value of the variable SUBST is set to be 1 then the program computes the direct and inverse transformations of coordinates (5), which are written out to the file LINA01OUT1 together with the input Hamiltonian, the normal form obtained by direct transformation and the Hamiltonian obtained from normal form through the inverse transformation by recursion formula from Appendix A.

3.2. The structure of the main program, LINA01.RED

The main program consists of six stages. The first stage is devoted to the initial setting-up. The first stage starts with the setting-up of several switches and the definition of the program variables as operators. The variables which the users can set their values in LINA01IN are assigned to take the default values. After those procedures, the procedure definitions are read from the file LINA01PROC.RED and the initial data from the file LINA01IN. The initial data are written out to the file LINA01OUT.

At the second stage, the initial values of internal operators H(i), i = 0, ..., SMAX, are assigned. If necessary, the quadratic part H(0) of the original Hamiltonian is brought into the standard form equal to $H_0(q, p)$ through a scaling transformation. The quadratic part G(0) of the Hamiltonian is thereby calculated to be equal to $G_0(\xi, \eta)$ with $(\xi, \eta) = (q, p)$.

From the third stage, the construction of the generating function $W(\xi, \eta; 1)$ and the normal form $G(\xi, \eta; 1)$ starts: Namely, W(s) and G(s) (s = 1, ..., SMAX) are computed. The computation is made by the call of the procedure, NORMFORM(MAX), with parameter SMAX.

At the fourth stage, W(S) for the generating function, G(S) for the normal form are written out to the file LINA01OUT. Further, in the case that the variable TEST is set to be equal to 1 at the first stage, the test results of the computations are written to the file LINA01OUT.

At the fifth stage, the program provides the direct and inverse transformations if the variable SUBST is set to be 'on' at the first stage.

At the last stage, the initial setting-up of the switches and the variables, the direct and inverse transformations are written out to the file LINA01OUT1. If TEST = 1 then a normal form evaluated by the direct transformation, the Hamiltonian evaluated from the normal form by the inverse transformation and some test results are written out to the file LINA01OUT1 too.

3.3. The structure of the program, LINA01PROC.RED

The file LINA01PROC.RED consists of fifteen procedures listed in the following.

C(b, a): This procedure calculates the binomial coefficient C_a^b.

L(r, f): This procedure calculates the Lie derivative, $L_{W_r} f$ (see (6)), of function f generated by function W(r).

SUPQXY(A, n): This procedure provide the transformation, along with (20), of the argument A chosen from the canonical variables (Q(1), ..., Q(n), P(1), ..., P(n)) to the formal complex ones (X(1), ..., X(2), Y(1), ..., Y(2)) corresponding to A.

SUXYPQ(A, n): This procedure performs the inverse transformations from the formal complex variables (X(1), ..., X(n), Y(1), ..., Y(n)) to the canonical ones (Q(1), ..., Q(n), P(1), ..., P(n)).

TEST1(u): This procedure check the Poisson commuting property of the function u with by means H_0; the calculation Poisson bracket $\{H_0, u\}$.

BASIS (HT, U, V, n): This procedure solves the fundamental equation (22) and finds (in terms of the formal complex variables (U(1), ..., U(n), V(1), ..., V(n) from (20)) the generating function WT and the normal form GT of degree-$(s+2)$ by formulas (29) and (30).

NORMFORM(MAX): This procedure performs the actual construction of the normal form for degrees from s = 1 up to s = SMAX + 2 (in terms of the canonical variables $\xi_1, \ldots, \xi_n, \eta_1, \ldots, \eta_n$) denoted here by (Q(1), ..., Q(n), P(1), ..., P(n)).

TS(n): This procedure calculates the term T_s along with (12).

K(j, i): This procedure calculates $K_{j,i}$ along with (13), where j and i are the summation indices.

KSI(m, n): This procedure performs the direct transformations $q \to \xi$ from (A.2), where m ranges over the order of expansion minus 2, m = 0 ... SMAX and n is the degree of freedom.

KSI1(j, i, n): This procedure picks up the term $\xi_{j,i}$ from (A.3) for direct and inverse transformations, where i, j are indices of summing, n is the number of degree of freedom.

ETA(m, n): This procedure performs the direct transformations $p \to \eta$ from (A.2), where m ranges over the order of expansion minus 2, m = 0 ... SMAX and n is the degree of freedom.

ETA1(j, i, n): This procedure picks up the term $\eta_{j,i}$ from (A.3) for direct and inverse transformations, where i and j are the summation indices and n the degree of freedom.

QQS(m, n): This procedure performs the inverse transformations $\xi \to q$ from (A.6), where m ranges over the order of expansion minus 2, m = 0 ... SMAX and n is the degree of freedom.

PPS(m, n): This procedure performs the inverse transformations $\eta \to p$ from (A.6), where m ranges over the order of expansion minus 2, m = 0 ... SMAX and n is the degree of freedom.

4. Examples

With LINA01, we compute the normal forms for a 1-degree-of-freedom (abbreviated to 1D) Hamiltonian (Example 1), a 2D-Hamiltonian (Example 2), and a 3D Hamiltonian (Example 3) up to the order, $s_{\max}$. In what follows, we present the input Hamiltonians and their normal forms *not* in terms of programming codes:

Example 1. The input Hamiltonian is $H(q, p) = \frac{1}{2}(p_1^2 + q_1^2) + \gamma q_1^4$.
The normal form G up to $s_{\max} = 6$ reads as $G = G_0 + G_2/2! + G_4/4! + G_6/6!$, where

$$G_0 = I_1, \quad G_2 = 3\gamma I_1^2, \quad G_4 = -102\gamma^2 I_1^3, \quad G_6 = 16875\gamma^3 I_1^4.$$

Example 2. The input Hamiltonian is $H(q, p) = \frac{\omega_1}{2}(p_1^2 + q_1^2) + \frac{\omega_2}{2}(p_2^2 + q_2^2) + aq_1^2 q_2$.
The normal form G up to $s_{\max} = 2$ reads as $G = G_0 + G_2/2!$, where

$$G_0 = \omega_1 I_1 + \omega_2 I_2, \quad G_2 = -\frac{a^2(-3\omega_2^2 + 8\omega_1^2)}{2\omega_2(-\omega_2^2 + 4\omega_1^2)} I_1^2 - \frac{4\omega_1 a^2}{-\omega_2^2 + 4\omega_1^2} I_1 I_2.$$

Example 3. The input Hamiltonian is $H(q, p) = \sum_{\nu=1}^{3} \frac{\omega_\nu}{2}(p_\nu^2 + q_\nu^2) + \alpha q_1 q_3^2 + \beta q_2 q_3^2$.
The normal form G up to $s_{\max} = 2$ reads as $G = G_0 + G_2/2!$, where

$$G_0 = \omega_1 I_1 + \omega_2 I_2 + \omega_3 I_3,$$
$$G_2 = \frac{4\alpha^2\omega_3}{\omega_1^2 - 4\omega_3^2} I_1 I_3 + \frac{4\beta^2\omega_3}{\omega_2^2 - 4\omega_3^2} I_2 I_3 - \frac{\beta^2(3\omega_2^2 - 8\omega_3^2)}{2\omega_2(\omega_2^2 - 4\omega_3^2)} I_3^2 - \frac{\alpha^2(3\omega_1^2 - 8\omega_3^2)}{2\omega_1(\omega_1^2 - 4\omega_3^2)} I_3^2.$$

Here $I_\nu = \frac{1}{2}(p_\nu^2 + q_\nu^2)$ $(\nu = 1, 2, 3)$.

Remarks. The normal forms for these Hamiltonians were calculated also by the program GITA [16]. In Examples 2 and 3, the frequencies, ω_νs, are assumed to be the formal parameters which are incommensurable; i.e. if $\sum_{\nu=1}^{n} \omega_\nu b_\nu = 0$ holds for some integers, b_νs, then all the b_νs must vanish. The ω_νs can, of course, be specified explicitly to be positive real numbers in the program, too. In the case with commensurable ω_νs, the normal forms in Examples 2 and 3 will differ from the above ones due to the commensurability.

We wish to discuss the performances of the programs, LINA and GITA, in Examples 1, 2 and 3: In Examples 2 and 3, we choose the frequencies, ω_νs, not only to be formal incommensurable parameters but also to be the commensurable values $\omega_1 = \omega_2 = \omega_3 = 1$. Both of the programs are implemented in REDUCE on a Pentium-IV-based computer with 512 RAM under Windows XP.

Table 1
The run-times (in seconds) of the programs LINA01 and GITA for Examples 1, 2 and 3

Example	ω_ν	SMAX	LINA01	GITA
1	1	38	12.8	25.5
2	ω_ν	4	4.4	4.0
2	1	10	4.0	7.6
3	ω_ν	2	8.9	42.5
3	1	8	13.7	14.7

Column 1 shows the labels (in digit) of the examples. Column 2 shows the choice of frequencies ω_ν: The Arabic number '1' expresses the case of $\omega_1 = \omega_2 = \omega_3 = 1$ and 'ω_ν' does the case that ω_νs are formal and incommensurable parameters. Column 3 shows the order of normalization SMAX. Columns 4 and 5 show the execution time (in seconds) of the normalization procedure made by programs LINA01 and by GITA, respectively.

From Table 1, one can see that LINA01 works more effectively than GITA does especially in the normalization of higher-degrees and in the normalization with parametric-frequencies. The performances of the programs, GITAN0, LINA01 and LINA03, are discussed in detail in [27,28], all of those are implemented in REDUCE 3.7, in MAPLE 7 and in MATHEMATICA 4.0.

5. Conclusion

The REDUCE program LINA01 is proposed on the basis of the Lie transformation theory, that allows us to make computer algebraic calculations very effectively for bringing polynomial Hamiltonians into their normal form. A program code unifying the direct and the inverse normalization will be published in a further paper.

Acknowledgements

All the authors thank Professor V.P. Gerdt at LIT, JINR for his encouragement. Thanks are also to RFBR for support by grants Nos. 03-02-16263 and 03-02-17695. YUw is supported by Grants-in-Aid Scientific Research Nos. 13660065 and 16560050 from JSPS. YUw thanks JINR for the hospitality and support during his stay at JINR in October, 2004, where the primary manuscript has been revised.

Appendix A. Lie transformation of the canonical coordinates

The canonical transformation $(q, p) \to (\xi, \eta)$ from (5) and (6) up to degree-$s_{\max}$ in ϵ are realized by recursion formula

$$q = \xi + \sum_{s=1}^{s_{\max}} \frac{\epsilon^s}{s!} \xi^{(s)}(\xi, \eta), \qquad p = \eta + \sum_{s=1}^{s_{\max}} \frac{\epsilon^s}{s!} \eta^{(s)}(\xi, \eta), \tag{A.1}$$

where

$$\xi^{(n)} = \frac{\partial W_n}{\partial \eta} + \sum_{j=1}^{n-1} C_{n-1}^{j} \xi_{j,n-j}, \qquad \eta^{(n)} = -\frac{\partial W_n}{\partial \xi} + \sum_{j=1}^{n-1} C_{n-1}^{j} \eta_{j,n-j}, \tag{A.2}$$

$$\xi_{j,i} = L_j \xi^{(i)} - \sum_{m=1}^{j-1} C_{j-1}^{m-1} L_m \xi_{j-m,i}, \qquad \eta_{j,i} = L_j \eta^{(i)} - \sum_{m=1}^{j-1} C_{j-1}^{m-1} L_m \eta_{j-m,i}. \tag{A.3}$$

Here L_j is the Lie differential operator

$$L_j \cdot = \{\cdot, W_j(\xi, \eta)\}_{(\xi,\eta)}, \tag{A.4}$$

the symbol $\{\cdot, \cdot\}$ means the Poisson bracket of (1e). The components $W_j(\xi, \eta)$ of generated function $W(\xi, \eta; \epsilon)$ of (4) are calculated in advance by recursion formulas (11)–(13).

The inverse transformation $(\xi, \eta) \to (q, p)$ reads as

$$\xi = q + \sum_{n=1}^{s_{\max}} \frac{\epsilon^n}{n!} q^{(n)}(q, p), \qquad \eta = p + \sum_{n=1}^{s_{\max}} \frac{\epsilon^n}{n!} p^{(n)}(q, p), \tag{A.5}$$

where

$$q^{(n)} = -\xi^{(n)} + \sum_{j=1}^{n-1} C_{n-1}^{j} \xi_{j,n-j}, \qquad p^{(n)} = -\eta^{(n)} + \sum_{j=1}^{n-1} C_{n-1}^{j} \eta_{j,n-j}. \tag{A.6}$$

Appendix B. TEST RUN OUTPUT

The test run output is presented for the Hamiltonian taken as Example 2 in Section 4 with $\omega_1 = \omega_2 = 1$. Since the frequencies, $\omega_1 = \omega_2 = 1$, chosen here are commensurable, the output given below differs from the normal form G with $\omega_1 = \omega_2 = 1$ in Example 2.

The output file, LINA01OUT, contains the initial data, the generating function, the resulting normal form, and the confirmation that the normal form obtained is Poisson commuting with $G(0)$

LINA01OUT:

```
n:= 2$

smax:= 2$

ww(1) := 1$

ww(2) := 1$

jmax := 1$

h(0) := 1/2*(p(2)**2 + p(1)**2 + q(2)**2 + q(1)**2)$

h(1) := q(2)*q(1)**2*a$

test := 1$

subst := 1$

w(1):=a*( - 2/3*p(2)*p(1)**2 - 1/3*p(2)*q(1)**2 - 2/3*p(1)*q(2)*q(1))$

w(2):=a**2*(5/12*p(2)**2*p(1)*q(1) + 5/12*p(2)*p(1)**2*q(2) + 1/4*p(2)*
q(2)*q(1)**2 + 5/24*p(1)**3*q(1) + 1/4*p(1)*q(2)**2*q(1) + 1/8*p(1)*q(1)
**3)$

g(0):=1/2*(p(2)**2 + p(1)**2 + q(2)**2 + q(1)**2)$

g(1):=0$

g(2):=a**2*( - 5/6*p(2)**2*p(1)**2 + 1/6*p(2)**2*q(1)**2 - 2*p(2)*p(1)*
q(2)*q(1) - 5/24*p(1)**4 + 1/6*p(1)**2*q(2)**2 - 5/12*p(1)**2*q(1)**2 - 5
/6*q(2)**2*q(1)**2 - 5/24*q(1)**4)$
```

```
test:=0; & g(0) is normal form!!!$

test:=0; & g(2) is normal form!!!$

Time: 64 ms
```

The output file LINA01OUT1 contains the initial data, the direct and inverse transformations, the normal form for initial Hamiltonian, and the Hamiltonian obtained from the normal form through the inverse transformation. The coincidence of the initial Hamiltonian with the Hamiltonian inverted from the normal form is also confirmed.

LINA01OUT1:

```
n:= 2$

smax:= 2$

ww(1) := 1$

ww(2) := 1$

jmax := 1$

h(0) := 1/2*(p(2)**2 + p(1)**2 + q(2)**2 + q(1)**2)$

h(1) := q(2)*q(1)**2*a$

test := 1$

subst := 1$

q(1)= - 17/72*p(2)**2*q(1)*a**2 + 5/12*p(2)*p(1)*q(2)*a**2 - 4/3*p(2)*
p(1)*a + 13/144*p(1)**2*q(1)*a**2 + 25/72*q(2)**2*q(1)*a**2 - 2/3*q(2)*
q(1)*a + 25/144*q(1)**3*a**2 + q(1)$
q(2)=5/12*p(2)*p(1)*q(1)*a**2 - 17/72*p(1)**2*q(2)*a**2 - 2/3*p(1)**2*
a + 25/72*q(2)*q(1)**2*a**2 + q(2) - 1/3*q(1)**2*a$

p(1)= - 47/72*p(2)**2*p(1)*a**2 - 1/4*p(2)*q(2)*q(1)*a**2 + 2/3*p(2)*
q(1)*a - 47/144*p(1)**3*a**2 + 7/72*p(1)*q(2)**2*a**2 + 2/3*p(1)*q(2)*
a - 11/144*p(1)*q(1)**2*a**2 + p(1)$

p(2)= - 47/72*p(2)*p(1)**2*a**2 + 7/72*p(2)*q(1)**2*a**2 + p(2) - 1/4*
p(1)*q(2)*q(1)*a**2 + 2/3*p(1)*q(1)*a$

q(1)= - 47/72*p(2)**2*q(1)*a**2 - 5/12*p(2)*p(1)*q(2)*a**2 + 4/3*p(2)*
p(1)*a - 77/144*p(1)**2*q(1)*a**2 + 7/72*q(2)**2*q(1)*a**2 + 2/3*q(2)*
q(1)*a + 7/144*q(1)**3*a**2 + q(1)$
```

```
q(2)= - 5/12*p(2)*p(1)*q(1)*a**2 - 47/72*p(1)**2*q(2)*a**2 + 2/3*p(1)**2*
a + 7/72*q(2)*q(1)**2*a**2 + q(2) + 1/3*q(1)**2*a$

p(1)= - 17/72*p(2)**2*p(1)*a**2 + 1/4*p(2)*q(2)*q(1)*a**2 - 2/3*p(2)*
q(1)*a - 17/144*p(1)**3*a**2 + 25/72*p(1)*q(2)**2*a**2 - 2/3*p(1)*q(2)*
a + 43/144*p(1)*q(1)**2*a**2 + p(1)$

p(2)= - 17/72*p(2)*p(1)**2*a**2 + 25/72*p(2)*q(1)**2*a**2 + p(2) + 1/4*
p(1)*q(2)*q(1)*a**2 - 2/3*p(1)*q(1)*a$

g:= - 5/12*p(2)**2*p(1)**2*a**2 + 1/12*p(2)**2*q(1)**2*a**2 + 1/2*p(2)**2
- p(2)*p(1)*q(2)*q(1)*a**2 - 5/48*p(1)**4*a**2 + 1/12*p(1)**2*q(2)**2*
a**2 - 5/24*p(1)**2*q(1)**2*a**2 + 1/2*p(1)**2 - 5/12*q(2)**2*q(1)**2*
a**2 + 1/2*q(2)**2 - 5/48*q(1)**4*a**2 + 1/2*q(1)**2$

h:=1/2*p(2)**2 + 1/2*p(1)**2 + 1/2*q(2)**2 + q(2)*q(1)**2*a + 1/2*q(1)**2$

Time: 63 ms plus GC time: 30 ms
```

References

[1] J.K. Moser, Lectures on Hamiltonian Systems, in: Memoirs of the American Math. S., vol. 81, 1968, pp. 1–60.
[2] A.P. Markeev, Theoreticheskaja Mekhanika, Nauka, Moscow, 1990 (in Russian).
[3] A.H. Nayfeh, Perturbation Methods, Wiley-Interscience Publication, New York, 1973.
[4] A.P. Markeev, Tochki Libracii v Nebesnoj Mekhanike i Kosmodinamike, Nauka, Moscow, 1978 (in Russian).
[5] G.E.O. Giacaglia, Perturbation Methods in Non-Linear Systems, Springer-Verlag, New York, 1972.
[6] A.H. Nayfeh, Introduction to Perturbation Techniques, Wiley-Interscience Publication, New York, 1981.
[7] A.J. Lichtenberg, M.A. Lieberman, Regular and Stochastic Motion, Springer-Verlag, Berlin, 1983.
[8] D. Kahaner, C. Moler, St. Nash, Numerical Methods and Software, Prentice-Hall International, Inc., New York, 1989.
[9] G.D. Birkhoff, Dynamical Systems, A.M.S. Colloquium Publications, New York, 1927.
[10] C.L. Siegel, J.K. Moser, Lectures of Celestial Mechanics, Springer-Verlag, Berlin, 1971.
[11] F. Gustavson, Astron. J. 71 (1966) 670.
[12] Y. Uwano, J. Phys. A 33 (2000) 6635.
[13] A. Giorgilli, Comput. Phys. Comm. 6 (1979) 331.
[14] T. Uzer, R.A. Marcus, J. Chem. Phys. 81 (1984) 5013.
[15] M. Robnik, J. Phys. A: Math. Gen. 17 (1984) 109.
[16] V. Basios, N.A. Chekanov, B.L. Markovski, V.A. Rostovtsev, S.I. Vinitsky, Comput. Phys. Comm. 90 (1995) 355.
[17] V.F. Edneral, O.A. Khrustalev, Sov. J. Program. 5 (1992) 73 (in Russian).
[18] A.C. Hearn, REDUCE User's Manual, Version 3.4. The Rand Corporation, Santa Monica, 1991.
[19] G.I. Hori, Astron. Soc. Japan 18 (1966) 287.
[20] A. Deprit, Cel. Mech. 1 (1969) 12.
[21] A.A. Kamel, Cel. Mech. 3 (1970) 90.
[22] W.A. Mersman, Cel. Mech. 3 (1970) 81.
[23] W.A. Mersman, Cel. Mech. 3 (1971) 384.
[24] J.A. Campbell, W.H. Jeffris, Cel. Mech. 2 (1970) 467.
[25] A.P. Markeev, A.G. Sokolsky, Some calculation algorithms for the normalization Hamiltonian systems, Preprint IPM AN USSR, No. 31, 1976 (in Russian).
[26] M.K. Ali, J. Math. Phys. 26 (10) (1985) 2565.
[27] A. Gusev, Yu. Ukolov, N. Chekanov, V. Rostovtsev, Y. Uwano, S. Vinitsky, in: V.G. Ganzha, E.W. Mayr, E.V. Vorozhtsov (Eds.), Computer Algebra in Scientific Computing, Technische Universitat Munchen, 2003, pp. 187–197.
[28] A.A. Gusev, N.A. Chekanov, V.A. Rostovtsev, S.I. Vinitsky, Y. Uwano, Programm. Comput. Softw. 30 (2004) 75.
[29] M. Abramowitz, I. Stegun, Handbook of Mathematical Functions, National Bureau of Standards, New York, 1964.