第43卷第6期 四川大学学报(工程科学版) Vol. 43 No. 6
2011年11月 JOURNAL OF SICHUAN UNIVERSITY (ENGINEERING SCIENCE EDITION) Nov. 2011

文章编号: 1009-3087(2011)06-0127-06

SIRD: 一个同步整数关系探测算法

陈经纬[1,2],冯 勇[1*],秦小林[1],张景中[1]

(1. 中国科学院 重庆绿色智能技术研究院,重庆 401122; 2. 中国科学院 成都计算机应用研究所,四川 成都 610041;
3. 中国科学院 研究生院,北京 100049)

摘 要: 为了解决一组实数向量的整数关系探测问题,通过广义的 Hermite 约化方法来约化超平面矩阵,基于著名的 PSLQ 算法,给出了一个同步整数关系探测的新算法 SIRD; 并且在计算机代数系统 Maple 中采用软件精度数据类型"sfloat"实现了 SIRD 算法和另一个同步整数关系探测算法 HJLS,数值实验说明本文的算法相比 HJLS 算法更高效; 最后,部分采用硬件精度数据类型"hfloat"给出了 SIRD 算法在 Maple 中的另一种的实现,并将其应用到代数数极小多项式的重构问题中,进一步拓展了张景中和冯勇提出的"采用近似计算获得准确值"这一思想的应用范围.

关键词: 整数关系; 代数数; 极小多项式; 符号数值计算

中图分类号: TP301 **文献标志码:** A

SIRD: An Algorithm for Simultaneous Integer Relations Detection

CHEN Jing-wei[1,2], *FENG Yong*[1*], *QIN Xiao-lin*[1], *ZHANG Jing-zhong*[1]

(1. Chongqing Inst. of Green and Intelligent Technol., CAS, Chongqing 401122, China;
2. Chengdu Inst. of Computer Application, CAS, Chengdu 610041, China; 3. Graduate Univ., CAS, Beijing 100049, China)

Abstract: In order to reduce the hyperplane matrix when detecting simultaneous integer relations for several real vectors, a generalized Hermite reduction was presented. Based on generalized Hermite reduction and patial sum-lower trapezoidal orthogonal decomposition (PSLQ) algorithm, the algorithm of simultaneous integer relations detection (SIRD) was proposed. SIRD was implemented in computer algebra system Maple in two different routes of software float-point data type "sfloat" and hardware float point data type "hfloat". The SIRD was compared with HJLS, and the results showed that SIRD is better. Furthermore, SIRD was applied to get a complete method for finding the minimal polynomial of an unknown complex algebraic number from its approximation.

Key words: integer relation; algebraic number; minimal polynomial; symbolic-numeric computation

对实数向量 $\boldsymbol{x} = (x_1, \cdots, x_n)^{\mathrm{T}} \in \mathbb{R}^n$,称非零整数向量 $\boldsymbol{m} = (m_1, \cdots, m_n)^{\mathrm{T}} \in \mathbb{Z}^n$ 为 $\boldsymbol{x}$ 的整数关系,如果 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{m} = \sum_{i=1}^{n} x_i m_i = 0$。怎样探测一个实数向量的整数关系是一个十分古老的问题。当 $n = 2$ 时,Euclid 算法或与之等价的连分数算法便能够解决此问题。但对 $n > 2$ 的情形,直到20世纪70年代末才由Ferguson和Forcade给出了第一个行之有效的方法[1]。此后又经过了近20年的完善,形成了一个高效的算法 PSLQ[2]。正是由于这一算法解决了诸如计算数学和量子场论等领域的许多问题[3-4],在2000年被 *Computing in Science and Engineering* 杂志评为20世纪十大算法之一[5]。尽管 PSLQ 算法功能强大,并且能够推广到复数域,但是它也有不足之处。比如,对复数向量 $\boldsymbol{x} = (1 + \mathrm{I}, 1 + 2\mathrm{I}, 2 + \mathrm{I})^{\mathrm{T}}$ 应用 PSLQ 算法只能得到 Gauss 整数向量 $\boldsymbol{m} = (1, \mathrm{I}, -\mathrm{I})^{\mathrm{T}}$,易知 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{m} = 0$,其中,$\mathrm{I} = \sqrt{-1}$。在实际应用中,往往需要一个整数向量,而不是一个 Gauss 整数向量,此时 PSLQ 算法便无能为力了。比如,由复代数数的近似值获得其准确的极小多项式。

作者通过推广 PSLQ 算法中的 Hermite 约化过程得到SIRD算法。将复向量 $\boldsymbol{x} \in \mathbb{C}^n$ 分为实部 $\mathrm{Re}(\boldsymbol{x})$ 和虚部 $\mathrm{Im}(\boldsymbol{x})$ 两个实向量,利用 SIRD 算法

收稿日期: 2011-05-04

基金项目: 国家"973"计划资助项目(2011CB302400); 国家自然科学基金资助项目(10771205); 中国科学院西部之光资助项目

作者简介: 陈经纬(1984-),男,博士生. 研究方向: 符号与数值混合计算.

* 通讯联系人 E-mail: yongfeng@casit.ac.cn

可以得到一个非零的整数向量m使得$\mathrm{Re}(x)^{\mathrm{T}}m = \mathrm{Im}(x)^{\mathrm{T}}m = 0$。比如，对上述的$x = (1+\mathrm{I},1+2\mathrm{I},2+\mathrm{I})^{\mathrm{T}}$，本文的SIRD算法将给出一个整数向量$m = (-3,1,1)^{\mathrm{T}}$使得$x^{\mathrm{T}}m = 0$成立。

对1组实数向量$x_1,\cdots,x_t \in \mathbb{R}^n (2 \le t < n)$，如果1个非零的整数向量$m$使得$x^{\mathrm{T}}m = 0, i = 1,\cdots,t$，那么称$m$为$x_1,\cdots,x_t$的同步整数关系。据作者所知，文献[6]不仅给出了第一个整数关系探测多项式时间算法的严格证明，也给出了一个同步整数关系探测算法HJLS。作者在计算机代数系统Maple中实现了HJLS算法和SIRD算法，实例和实验数据说明本文的算法效率更高(见第2节)。另外，基于改进的HJLS算法，Rössner和Schnorr在文献[7]中试图给出1个计算2个实向量x_1和x_2的同步整数关系的算法，但由于存在一些问题仍未解决，现在仍处于准备阶段。

最后，将SIRD算法应用到从代数数的近似值获得其准确的极小多项式这一问题中去。在文献[8-9]中，作者已经对实代数数的情况作了讨论。结合给出的SIRD算法的这一应用，便能将“采用近似计算获得准确值”[10]的应用领域扩大到代数数范围，并且该方法提供了一条不依赖于LLL算法(如文献[11-12])的新途径。

1 同步整数关系探测

定义1 设$x_i = (x_{i,1},x_{i,2},\cdots,x_{i,n})^{\mathrm{T}} \in \mathbb{R}^n, i = 1,\cdots,t (2 \le t < n)$。若存在非零整数向量$m = (m_1,m_2,\cdots,m_n)^{\mathrm{T}}$使得$\sum_{j=1}^{n} m_j x_{i,j} = 0$，则称$m$为$x_1,\cdots,x_t$的同步整数关系。

假设$x_1,\cdots,x_t$满足

$$\begin{vmatrix} x_{1,n-t+1} & x_{2,n-t+1} & \cdots & x_{t,n-t+1} \\ x_{1,n-t+2} & x_{2,n-t+2} & \cdots & x_{t,n-t+2} \\ \vdots & \vdots & \vdots & \vdots \\ x_{1,n} & x_{2,n} & \cdots & x_{t,n} \end{vmatrix} \neq 0 \qquad (1)$$

如果不满足，通过对矩阵$X = (x_1,\cdots,x_t)$实施初等行变换使得$X' = (x_1',\cdots,x_t') = CX$满足式(1)，其中$C \in GL(n,\mathbb{Z})$为整数幺模矩阵(行列式的绝对值为1)。若$m$为$x_1',\cdots,x_t'$的同步整数关系，则$C^{\mathrm{T}}m$为$x_1,\cdots,x_t$的同步整数关系。

定义2 设$x_i = (x_{i,1},x_{i,2},\cdots,x_{i,n})^{\mathrm{T}} \in \mathbb{R}^n, i = 1,\cdots,t (2 \le t < n)$。将矩阵$H \in \mathbb{R}^{n\times(n-t)}$称作是$x_1,\cdots,x_t$的超平面矩阵，如果$H$的各列构成向量空间$X^{\perp} = \{y \in \mathbb{R}^n : x_i^{\mathrm{T}}y = 0, i = 1,\cdots,t\}$的一组基。

设$b_1,\cdots,b_n$是$\mathbb{R}^n$的标准基，即b_i的第i个分量为1，其他分量为0。对$x_1,\cdots,x_t, b_1,\cdots,b_{n-t}$实施标准Gram-Schmidt正交化过程后得到$x_1^*,\cdots,x_t^*, b_1^*,\cdots,b_{n-t}^*$。由$x_1,\cdots,x_t$满足式(1)易知$n\times(n-t)$矩阵$b_1^*,\cdots,b_{n-t}^*$就是$x_1,\cdots,x_t$的超平面矩阵，记为$H_X$。若记$X = (x_1,\cdots,x_t)$，则超平面矩阵$H_X$满足$X^{\mathrm{T}}H_X = 0$，其中，0表示零矩阵。

定理1 若对任意的$x_1,\cdots,x_t$的同步整数关系m和任意的矩阵$A \in GL(n,\mathbb{Z})$，存在1个正交矩阵$Q \in \mathbb{R}^{(n-t)\times(n-t)}$使得$H = (h_{i,j}) = AH_XQ$是1个下梯形矩阵(对$j > i$，有$h_{i,j} = 0$)，且每个对角元$h_{j,j} \neq 0$，则

$$\frac{1}{\max\limits_{1\le j\le n-t} |h_{j,j}|} = \min_{1\le j\le n-t} \frac{1}{|h_{j,j}|} \le \|m\|_2 \qquad (2)$$

其中，$\|m\|_2 = \sqrt{m^{\mathrm{T}}m}$为$m$的2-范数。

将文献[2]中定理1的证明作相应修改便能得到上述定理的证明，在此不再赘述。定理1的意义在于给出了一个探测同步整数关系的基本思想：若能够以左乘幺模矩阵，右乘正交矩阵的形式对超平面矩阵对角元的模进行约化，则不等式(2)给出了$x_1,\cdots,x_t$的任意一个同步整数关系2-范数的不断增加的下界。另一方面，如果$x_1,\cdots,x_t$存在同步整数关系，那么这个下界便不会无限制地增加。无论是PSLQ算法还是HJLS算法都依赖于这一来自于文献[1]中的基本思想。下面考查如何约化超平面矩阵H_X。

在PSLQ算法中，对超平面矩阵的约化方法采用的是Hermite约化，现给出它的一个改进版本。

定义3 设$H = (h_{i,j}) \in \mathbb{R}^{n\times(n-1)}$为一下梯形阵且$h_{j,j} \neq 0$。令$D = (d_{i,j})$为$n$阶单位阵$I_n$。对$i$从2到$n$，$j$从$i-1$到1(步长为$-1$)，令$q = \lfloor h_{i,j}/h_{j,j} + 1/2 \rfloor$；对$k$从1到$n$，令$d_{i,k} = d_{i,k} - qd_{j,k}$。更新$H = DH$。称$H$为原矩阵的Hermite约化，$D$为Hermite约化矩阵。

PSLQ算法对于一个复数向量之所以会生成一个Gauss整数关系，其原因就在于对超平面矩阵约化时，相应的H为复数矩阵，这导致约化过程中的约化矩阵D中产生了Gauss整数。尽管不再适合探测同步整数关系，但Hermite约化仍有一些好的性质可以利用。采用如下广义Hermite约化过程对超平面矩阵H_X进行约化。

定义4 设$H = (h_{i,j}) \in \mathbb{R}^{n\times(n-t)}$满足$h_{j,j} \neq 0$，

且对 $j > i$, 有 $h_{i,j} = 0$。初始化 $\boldsymbol{D} = (d_{i,j})$ 为 n 阶单位阵 $\boldsymbol{I}_n$。对 i 从 2 到 n, j 从 $\min\{i-1, n-t\}$ 到 1 (步长为 -1), 令 $q = \lfloor h_{i,j}/h_{j,j} + 1/2 \rfloor$; 对 k 从 1 到 n, 令 $d_{i,k} := d_{i,k} - qd_{j,k}$。更新 $\boldsymbol{H} = \boldsymbol{DH}$。若存在 2 个整数 s_1, $s_2 \in \{n-t+1, \cdots, n\}$ 满足 $s_1 < s_2, h_{s_1,n-t} = 0$ 且 $h_{s_2,n-t} \neq 0$, 则交换 $\boldsymbol{D}$ 和 $\boldsymbol{H}$ 的第 s_1 行和第 s_2 行。称 $\boldsymbol{H}$ 为原矩阵的广义 Hermite 约化, $\boldsymbol{D}$ 为广义 Hermite 约化矩阵。

广义 Herimite 约化保留了 Hermite 约化的两个性质: 约化矩阵 $\boldsymbol{D} \in GL(n, \mathbb{Z})$ 为幺模矩阵; 对所有的 $k > i$, 约化后的矩阵 $\boldsymbol{H}' = (h_{i,j}')$ 满足 $|h_{k,i}'| \leq |h_{i,i}'|/2 = |h_{i,i}|/2$。它们的主要区别在于: 广义 Hermite 约化能够约化 $\boldsymbol{H} \in \mathbb{R}^{n\times(n-t)}$ 的最后 $t-1$ 行, 而 Hermite 约化不能; 若存在 2 个整数 $s_1, s_2 \in \{n-t+1, \cdots, n\}$ 满足 $s_1 < s_2$, $h_{s_1,n-t} = 0$ 且 $h_{s_2,n-t} \neq 0$, 则交换 $\boldsymbol{D}$ 和 $\boldsymbol{H}$ 的第 s_1 和 s_2 两行, 这是原来 Hermite 约化中没有的。这也意味着下面的命题成立。

命　题　在经过广义的 Herimite 约化后, 如果 $h_{n-t+1,n-t} = 0$, 那么 $h_{n-t+2,n-t} = \cdots = h_{n,n-t} = 0$。

至此, 基于定理 1 给出的基本思想和 PSLQ 算法, 利用广义 Hermite 约化便可以得到一个探测同步整数关系的算法 SIRD。

算法(SIRD)　输入: $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t) \in \mathbb{R}^{n\times t}$ 满足式(1), 参数 $\gamma > 2/\sqrt{3}$, $M > 0$。或者输出 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t$ 的同步整数关系, 或者断言 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t$ 不存在 2 - 范数小于 M 的同步整数关系。

S1: 计算超平面矩阵 $\boldsymbol{H}_X$, 初始化 $\boldsymbol{H} := \boldsymbol{H}_X$, $\boldsymbol{B} := \boldsymbol{I}_n$。

S2: 计算 $\boldsymbol{H}$ 的广义 Hermite 约化矩阵 $\boldsymbol{D}$。令 $\boldsymbol{X}^{\mathrm{T}} := \boldsymbol{X}^{\mathrm{T}}\boldsymbol{D}^{-1}$, $\boldsymbol{H} := \boldsymbol{DH}$, $\boldsymbol{B} := \boldsymbol{BD}^{-1}$。

S3: 迭代。

S31: 选择 r 使得 $\gamma^r |h_{r,r}| \geq \gamma^i |h_{i,i}|$ $(1 \leq i \leq n-t)$。交换 $\boldsymbol{H}$ 的第 r 行和第 $r+1$ 行, 交换 $\boldsymbol{X}^{\mathrm{T}}$ 和 $\boldsymbol{B}$ 的第 r 列和第 $r+1$ 列。

S32: 若上一步选择的 $r < n-t$, 则 $\boldsymbol{H}$ 便不再是下梯形的, 此时对 $\boldsymbol{H}$ 作 LQ 分解 (等价于对 $\boldsymbol{H}^{\mathrm{T}}$ 作 QR 分解), 更新 $\boldsymbol{H}$ 为分解后的下梯形矩阵。

S33: 计算 $\boldsymbol{H}$ 的广义 Hermite 约化矩阵 $\boldsymbol{D}$。令 $\boldsymbol{X}^{\mathrm{T}} := \boldsymbol{X}^{\mathrm{T}}\boldsymbol{D}^{-1}$, $\boldsymbol{H} := \boldsymbol{DH}$, $\boldsymbol{B} := \boldsymbol{BD}^{-1}$。

S34: 计算 $G := 1/\max\limits_{1\leq j\leq n-t} |h_{j,j}|$。若 $G > M$, 则断言 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t$ 不存在 2 - 范数小于 M 的同步整数关系, 终止算法。

S35: 若 $\boldsymbol{X}^{\mathrm{T}}$ 的第 j 列全为 0, 则输出 $\boldsymbol{B}$ 的第 j 列; 若 $h_{n-t,n-t} = 0$, 则输出 $\boldsymbol{B}$ 的第 $n-t$ 列。

定理 2　若 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t \in \mathbb{R}^n$ 存在同步整数关系, 记 $\lambda(\boldsymbol{X})$ 为 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t$ 的同步整数关系所具有的最小 2 - 范数, 令 $\gamma > 2/\sqrt{3}$, $M > \lambda(\boldsymbol{X})$, 则 SIRD 算法一定能找到一个同步整数关系 $\boldsymbol{m}$, 且

$$\|\boldsymbol{m}\|_2 \leq \gamma^{n-t}\lambda(\boldsymbol{X}) \tag{5}$$

SIRD 算法是通过推广 PSLQ 算法中的 Hermite 约化而来的, 其正确性(定理 2)的证明可以参照文献 [2] 中的定理 2 和 3 的证明过程, 不同的地方在于广义 Hermite 约化的性质命题在证明过程中起了至关重要的作用。现简要描述如下。

设 $\boldsymbol{H}(k)$ 为 SIRD 算法中经过 k 次迭代之后的矩阵 $\boldsymbol{H}$。如果存在 $1 \leq j \leq n-t$ 使得 $h_{j,j}(k) = 0$ 且 k 是满足上述条件的最小正整数, 那么 $j = n-t$。这是因为广义 Hermite 约化不会将对角元化为 0。此时, 矩阵 $\boldsymbol{B}$ 的第 $n-t$ 列必为 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t$ 的同步整数关系。在 SIRD 算法中, 有

$$0 = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{H}_X = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{BB}^{-1}\boldsymbol{H}_X = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{BB}^{-1}\boldsymbol{H}_X\boldsymbol{Q} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{BH}(k-1),$$

其中, $\boldsymbol{Q}$ 为一适当的 $n-t$ 阶正交阵。记 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B} = (\boldsymbol{z}_1, \cdots, \boldsymbol{z}_t)^{\mathrm{T}}$, 其中, $\boldsymbol{z}_i = (z_{i,1}, \cdots, z_{i,n})^{\mathrm{T}}$, 则

$$\begin{pmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{pmatrix}_{t\times(n-t)} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{BH}(k-1) =$$

$$\begin{pmatrix} \boldsymbol{z}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{z}_t^{\mathrm{T}} \end{pmatrix}\boldsymbol{H}(k-1) = \begin{pmatrix} \cdots & \sum\limits_{j=n-t}^{n} z_{1,j}h_{j,n-t}(k-1) \\ \cdots & \cdots \\ \cdots & \sum\limits_{j=n-t}^{n} z_{t,j}h_{j,n-t}(k-1) \end{pmatrix} =$$

$$\begin{pmatrix} \cdots & z_{1,n-t}h_{n-t,n-t}(k-1) \\ \cdots & \cdots \\ \cdots & z_{t,n-t}h_{n-t,n-t}(k-1) \end{pmatrix}。$$

由 $h_{n-t,n-t}(k) = 0$ 知 $h_{n-t+1,n-t}(k-1) = 0$ 且 $h_{n-t,n-t}(k-1) \neq 0$。而命题意味着上式的最后一个等号成立, 从而 $z_{1,n-t} = \cdots = z_{t,n-t} = 0$, 即矩阵 $\boldsymbol{B}$ 的第 $n-t$ 列是 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t$ 的同步整数关系。

至此, 对复数向量 $\boldsymbol{v} = (v_1, v_2, \cdots, v_n)^{\mathrm{T}} \in \mathbb{C}^n$ 的整数关系可以由 SIRD 算法探测 $\boldsymbol{v}$ 的实部向量 $(\mathrm{Re}(v_1), \mathrm{Re}(v_2), \cdots, \mathrm{Re}(v_n))^{\mathrm{T}}$ 和虚部向量 $(\mathrm{Im}(v_1), \mathrm{Im}(v_2), \cdots, \mathrm{Im}(v_n))^{\mathrm{T}}$ 的同步整数关系得到, 从而克服了 PSLQ 对复数向量仅能输出 Gauss 整数关系的不足。

2 算法实现与实验数据

给出的同步整数关系探测算法 SIRD 并不是第一个可用于探测一组向量同步整数关系的算法。早在 20 世纪 80 年代,Hastad、Just、Lagarias 和 Schnorr 已经给出了一个同步整数关系探测算法 HJLS[12]。这一算法与本文算法的基本思想是一致的,都是基于文献[1]中给出的方法。HJLS 算法也可以分为求超平面矩阵、初始约化超平面矩阵、迭代这 3 个步骤。与本文算法的主要区别在于对超平面矩阵 $\boldsymbol{H}_X$ 采用了不同的约化方式。HJLS 算法利用一个类似于 LLL 约化[13]的矩阵约化过程对超平面矩阵进行约化。在文献[2,14]中,已经指出这一约化过程的数值表现不稳定,同时也说明了基于数值稳定的 QR 分解(如 Householder 变换)的 Hermite 约化过程具有很好的数值特性。另一方面,由于采用了不同的约化过程,也会导致 HJLS 算法与 SIRD 算法的迭代次数有所差异。例如对 $\boldsymbol{v}_1 = (11,27,31)^{\mathrm{T}}$ 和 $\boldsymbol{v}_2 = (1,2,3)^{\mathrm{T}}$,HJLS 算法将在 5 次迭代后输出一个 $\boldsymbol{v}_1$ 和 $\boldsymbol{v}_2$ 的同步整数关系 $(19,-2,-5)^{\mathrm{T}}$,而 SIRD 算法仅需 3 次迭代便会输出同一个整数关系。

由于这 2 个算法在计算机实现时都需要调用高精度的软件包,而计算机代数系统 Maple 提供了这样的功能,作者在计算机代数系统 Maple 13 中实现了 SIRD 和 HJLS 这两个同步整数关系探测算法的 $t = 2$ 的情形。在实现时,对相同的功能均采用相同的技术,例如涉及矩阵 QR 分解的模块,均采用基于 Householder 变换的数值稳定的 QR 分解算法。在这两个实施中都采用 Maple 中自带的软件精度数据类型“sfloat”。所有实验数据都是在 AMD Athlon 7750 处理器(2.70 GHz)2 GB 内存的 PC 中获得的。

表 1 给出了利用 HJLS 和 SIRD 对 Maple 中 rand 命令产生的两个长度为 n 的有理向量探测其同步整数关系的实验数据,其中,itr_{HJLS} 和 itr_{SIRD} 表示二者实际的迭代次数,t_{HJLS} 和 t_{SIRD} 表示二者的运行时间(单位:s)。前 6 个例子是维数较小时的比较,后面的例子维数逐渐升高。可以发现随着维数的不断增加,HJLS 和 SIRD 的表现差别也越来越显著。这不仅体现在迭代次数 HJLS 比 SIRD 多,更直接的体现在了运算时间上。以表 1 中的数据来看,SIRD 算法探测两个实数向量的同步整数关系所需要的时间平均约为 HJLS 所需时间的 1/10。

表 1 HJLS 和 SIRD 算法的实验数据

Tab. 1 Experimental data for HJLS and SIRD

no.	n	itr_{HJLS}	itr_{SIRD}	t_{HJLS}	t_{SIRD}
1	4	15	8	0.063	0.000
2	4	13	6	0.062	0.000
3	4	21	11	0.094	0.015
4	5	25	12	0.109	0.016
5	5	27	7	0.141	0.000
6	5	21	10	0.094	0.000
7	30	51	7	0.922	0.125
8	54	34	9	2.203	0.453
9	79	34	5	4.860	0.625
10	97	37	5	7.438	1.047
11	128	45	5	13.765	1.687
12	149	29	2	19.016	1.610
13	173	26	3	26.812	2.421
14	192	29	5	34.218	3.563
15	278	28	5	85.797	8.860
16	290	35	4	95.656	8.328
17	293	23	4	98.062	8.750
18	305	22	3	109.187	8.063
19	316	19	3	120.187	8.766
20	325	18	2	129.031	6.953

另外,SIRD 算法中的参数 γ 也会对算法的效率产生影响。比如对 $\boldsymbol{v}_1 = (86,6,8,673)^{\mathrm{T}}$ 和 $\boldsymbol{v}_2 = (83,5,87,91)^{\mathrm{T}}$,如果在运行时令 $\gamma = 2$,则在完成 10 次迭代后输出整数向量 $(-32,-747,63,10)^{\mathrm{T}}$,但是如果增大参数 γ 的值,令 $\gamma = 93$,SIRD 算法将在 7 次迭代后输出 $(-35,-2624,157,26)^{\mathrm{T}}$,可以验证这两个输出的整数向量都是 $\boldsymbol{v}_1$ 和 $\boldsymbol{v}_2$ 的同步整数关系。如果继续增加参数 γ 的值,会发现迭代次数总是 7 次,不再减少。对更多的实验数据观察,可以发现参数 γ 的值越大,得到正确整数关系所需要的精度就越高,这会影响计算的效率。基于这样的观察,所呈现的实验数据都是在 $\gamma = 1\,000$ 时得到的。同时,如何选取参数 γ 的值使得算法效率最优仍然需要进一步的研究。

3　应用举例

由于上述对 HJLS 和 SIRD 算法的实现中数据类型采用的是软件精度，当一些复杂的实例要求的精度很高时，效率不佳。为降低这一缺陷的影响，作者也给出了一个 SIRD 算法的快速实现 TLSIRD: 对算法中只涉及整数操作的那些矩阵一律采用硬件精度的数据类型 "hfloat"，这些硬件精度的数据类型在算法执行过程中将直接采用 IEEE 单精度硬件算术操作，而不通过软件平台进行高精度的算术操作。这个策略不仅节约了每次迭代过程的运算时间，甚至也使得实际的迭代次数略为减少。以 SIRD 算法在 "从代数数的近似值获得其准确极小多项式" 中的应用为例，描述 SIRD 算法这两种基于不同策略的 Maple 程序的效率。

重构 $n-1$ 次近似代数数 α 极小多项式的基本思想是: 若 α 的近似值 $\bar{\alpha}$ 满足一定的误差控制条件(本文不作描述，可以参考文献[8-9, 15])，则探测 $(1,\bar{\alpha},\cdots,\bar{\alpha}^{n-1})^{\mathrm{T}}$ 的一个近似整数关系 $\boldsymbol{p}=(p_0,p_1,\cdots,p_{n-1})^{T}\in\mathbb{Z}^n$(当 α 为实代数数时，用 PSLQ 算法; 当 α 为虚代数数时，用本文的 SIRD 算法) 使得这组近似的整数关系恰好是 $(1,\alpha,\cdots,\alpha^{n-1})^{\mathrm{T}}$ 的一个准确的整数关系，即 $\sum_{i=0}^{n-1}p_i\alpha^i=0$。那么 $\sum_{i=0}^{n-1}p_ix^i\in\mathbb{Z}[x]$ 的本原部分便是 α 准确的极小多项式。

例 1: 设 $\alpha=2+\sqrt{3}\mathrm{I}$, $\mathrm{I}=\sqrt{-1}$。显然 α 在 $\mathbb{Z}[x]$ 中的极小多项式是 x^2-4x+7。若事先未知 α 的准确值，只知道 α 的一个具有 4 位有效数字(已满足误差控制) 的近似值 $\bar{\alpha}=2.000+1.732\mathrm{I}$，及其次数的上界 2，系数的最大值不超过 7。为了得到一个 $(1,\bar{\alpha},\bar{\alpha}^2)^{\mathrm{T}}$ 的近似整数关系，可以对其实部向量 $\boldsymbol{v}_1=(1,2,1)^{\mathrm{T}}$ 和虚部向量 $\boldsymbol{v}_2=(0,1.732,6.928)^{\mathrm{T}}$ 运用 SIRD 算法，仅需两次迭代就可以得出一个 $\boldsymbol{v}_1$ 和 $\boldsymbol{v}_2$ 的同步整数关系。迭代过程中的矩阵 $\boldsymbol{B}$ 分别为:

$$\begin{pmatrix}2&1&0\\-1&-1&0\\0&0&1\end{pmatrix},\begin{pmatrix}7&0&2\\-4&0&-1\\1&1&0\end{pmatrix}。$$

很明显后 1 个矩阵的第 1 列即为所求，同时它也是 $(1,\alpha,\alpha^2)^{\mathrm{T}}$ 的一个整数关系。从而得到 α 在 $\mathbb{Z}[x]$ 中的极小多项式。

表 2 中，r 和 s 定义了一个代数数 $\alpha=3^{1/r}-2^{1/s}\mathrm{I}$，易知 α 的次数为 $2rs$。为测试 SIRD 算法的 2 种实现在恢复近似代数数极小多项式中的效率，在 Digits 位十进制精度下，探测两个 $n=2rs+1$ 维实向量的整数关系。表 2 中 itr 和 t 分别表示迭代次数和运行时间(单位: s)，下标为 SIRD 的对应采用 "sfloat" 数据类型的程序，下标为 TLSIRD 的对应部分采用 "hfloat" 数据类型的程序。表中最后 2 个例子，由于采用 "sfloat" 数据类型的程序运行时间太多，未完成。由于 "hfloat" 数据类型采用 IEEE 硬件算术操作，效率远远高于对高精度的 "sfloat" 数据类型的操作，所以 TLSIRD 程序的效率应明显优于 SIRD 程序，这与表 2 中所给出的数据相符，高效地实现了 SIRD 算法也为更多 SIRD 算法的应用提供了可能。

表 2　SIRD 算法 2 种实现的效率比较

Tab. 2　Efficiency comparison between SIRD and TLSIRD

r	s	n	*Digits*	itr_{SIRD}	t_{SIRD}	itr_{TLSIRD}	t_{TLSIRD}
4	3	25	100	5 685	75.937	5 611	8.215
3	5	31	150	11 792	356.890	11 792	28.157
6	3	37	300	18 927	556.031	18 993	73.109
4	5	41	350	26 600	942.516	26 192	134.360
5	5	51	400	50 084	2 432.672	49 738	440.110
6	5	61	550	84 677	6 422.079	81 758	1 267.985
4	9	73	1 000			159 326	4 889.922
6	7	85	1 300			234 422	10 658.735

另外，作者之所以没有给出 HJLS 算法采用 "hfloat" 数据类型的实现是因为从表 1 中已经能够反映出 HJLS 算法和 SIRD 算法的效率。即使采用 "hfloat" 数据类型，可以预见其效率会比采用 "sfloat" 数据类型的效率高，但仍不及 TLSIRD。

4 结 论

给出了广义 Hermite 过程用来约化超平面矩阵,基于 PSLQ 算法得到了一个可以探测 t 个实数向量的同步整数关系的算法 SIRD,克服了 PSLQ 算法只能对复数向量输出 Gauss 整数关系的不足。并分别采用高精度的浮点运算和部分高精度浮点运算这两种策略给出了该算法在计算机代数系统 Maple 中的两个实现,与已有的 HJLS 算法比较说明本文的算法在运行效率上更具优势。

本文涉及的 Maple 程序见文献[16]。

参考文献:

[1] Ferguson H R P, Forcade R W. Generalization of the Euclidean algorithm for real numbers to all dimensions higher than two [J]. Bulletin of the American Mathematical Society, 1979, 1(6): 912 – 914.

[2] Ferguson H R P, Bailey D H, Arno S. Analysis of PSLQ, an integer relation finding algorithm [J]. Math Comput, 1999, 68(225): 351 – 369.

[3] Bailey D H, Broadhurst D J. Parallel integer relation detection: Techniques and applications [J]. Math Comput, 2000, 70(236): 1719 – 1736.

[4] Bailey D H, Borwein J M, Calkin N J, et al. Experimental Mathematics in Action [M]. Natick, MA: AK Peters, 2007.

[5] Dongarra J, Sullivan F. Guest editors' introduction: the top 10 algorithms [J]. Comput Sci Eng, 2000, 2(1): 22 – 23.

[6] Hastad J, Just B, Lagarias J C, et al. Polynomial time algorithms for finding integer relations among real numbers [J]. SIAM Journal on Computing, 1989, 18(5): 859 – 881.

[7] Rössner C, Schnorr C P. Diophantine approximation of a plane [EB/OL]. http://citeseer.ist.psu.edu/193822.html, 1997.

[8] Qin Xiaolin, Feng Yong, Chen Jingwei, et al. Finding exact minimal polynomial by approximations [C]. SNC, 2009, 125 – 131.

[9] Qin Xiaolin, Feng Yong, Chen Jingwei, et al. Exact representation of real algebraic number by approximations and its applications [J]. Journal of Sichuan University: Engineering Science Edition, 2010, 42(2): 126 – 131. [秦小林, 冯勇, 陈经纬, 等. 采用近似方法的实代数数准确表示及其应用 [J]. 四川大学学报: 工程科学版, 2010, 42(2): 126 – 131.]

[10] Zhang Jingzhong, Feng Yong. Obtainingexact value by approximate computations [J]. Science in China Series A: Mathematics, 2007, 50(9): 1361 – 1368.

[11] Kannan R, Lenstra A K, Lovász L. Polynomialfactorization and nonrandomness of bits of algebraic and some transcendental numbers [J]. Math Comput, 1988, 50(181): 235 – 250.

[12] van Hoeij M, Novocin A. Gradual sub-lattice reduction and a new complexity for factoring polynomials [C]//LATIN '10. LNCS, 2010, 6034: 539 – 553.

[13] Lenstra A K, Lenstra Jr H W, Lovász L. Factoring polynomials with rational coefficients [J]. Math Ann, 1982, 261: 515 – 534.

[14] Ferguson H R P, Bailey D H. Polynomial time, numerically stable integer relation algorithm [R]. Technical Report RNR – 91 – 032, NAS Applied Research Branch, NASA Ames Research Center, 1992.

[15] Chen Jingwei, Feng Yong, Qin Xiaolin, et al. Reconstructing minimal polynomial from approximate algebraic numbers [J]. J Sys Sci & Math Scis, 2011, 31(8): 903 – 912. [陈经纬, 冯勇, 秦小林, 等. 代数数极小多项式的近似重构 [J]. 系统科学与数学, 2011, 31(8): 903 – 912.]

[16] Chen Jingwei. SIRD Maple package [EB/OL]. http://cid-5dbb16a211c63a9b.skydrive.live.com/self.aspx/.Public/sird.rar, 2011.

(编辑 杨 蓓)