УДК 004.942

Я.А. Калиновский [1], д-р техн. наук, **Ю. Е. Бояринова** [1,2], канд. техн. наук,
А.С. Сукало[1], аспирантка, **Я.В. Хицко**[2], канд. техн. наук
[1] Ин-т проблем регистрации информации НАН Украины
(Украина, 03113, Киев, ул. Н. Шпака, 2,
тел. 4542138, e-mail: kalinovsky@i.ua),
[2] Национальный технический университет Украины
«Киевский политехнический ин-т»
(Украина, 03113, Киев, пр-т Победы, 37, e-mail: ub@ua.fm)

Программный комплекс для гиперкомплексных вычислений

Изложены основные принципы, положенные в основу программного комплекса гиперкомплексных вычислений. Описаны структура комплекса и состав функциональных подсистем. Рассмотрены наиболее важные процедуры, выполняемые подсистемами, приведены листинги программ и примеры их применения.

К л ю ч е в ы е с л о в а: гиперкомплексная числовая система, процедура, компьютерная алгебра, операция, Maple.

Викладено основні принципи, покладені в основу програмного комплексу гіперкомплексних обчислень. Описано структура комплексу і склад функціональних підсистем. Розглянуто найважливіші процедури, які виконуються підсистемами. Наведено лістинги програм та приклади їх застосування.

К л ю ч о в і с л о в а: гіперкомплексна числова система, процедура, комп'ютерна алгебра, операція, Maple.

Гиперкомплексные числовые системы (ГЧС) широко применяются в различных областях науки и техники [1, 2]. Теоретическая и прикладная механика, навигация, криптография, цифровая обработка сигналов — далеко не полный перечень областей науки и техники, где использование методов ГЧС является высокоэффективным.

Оперирование с гиперкомплексными числами, особенно в символьной форме, вызывает значительные трудности [3, 4], связанные с их многомерностью. Так, например, для перемножения двух кватернионов с численными коэффициентами, являющихся четырехмерными гиперкомплексными числами, необходимо выполнить 16 вещественных умножений и 12 сложений. Но коэффициенты в гиперкомплексных числах могут быть не только числами, но и различными алгебраическими выражениями (например, полиномами), различными функциями и гиперкомплексными числами, в том числе и с символьными переменными и коэффициентами. Поэтому для успешного оперирования с такими объектами требуются специализированные программные комплексы.

Для реализации подобных комплексов широко распространенные языки программирования, такие как Си, Паскаль, Фортран, Java, Basic, — неэффективны. Наиболее подходящими для создания такого типа комплексов являются системы компьютерной алгебры, или системы символьных вычислений [5], так как многие из них имеют возможность оперирования с некоторыми ГЧС, например с комплексными числами, кватернионами, клиффордовыми алгебрами и др.

Из существующих многочисленных систем компьютерной алгебры наиболее распространенными являются MatLab, Mathkad, Mathematica и Maple. Будем использовать систему компьютерной алгебры Maple, как одну из наиболее развитых и доступных для создания информационной технологии, которая повышает эффективность математического моделирования различных научно-технических задач с использованием ГЧС различной размерности и научных исследований в области теории ГЧС.

Основные принципы построения программного комплекса (ПК) гиперкомплексных вычислений. Поскольку система компьютерной алгебры Maple позволяет создавать частные пакеты различных вычислительных процедур, ПК гиперкомплексных вычислений представляет собой пакет, имеющий свой идентификатор. Такой ПК можно вызывать, присоединять к программе и транспортировать на другие компьютеры. Из процедур ПК можно формировать программы вычислений, используя средства алгоритмического языка Maple.

Программный комплекс может быть инсталлирован на любом компьютере с операционной системой Windows и системой компьютерных вычислений Maple не ниже пятой версии. Он открыт для пополнения новыми процедурами и редактирования существующих процедур любым пользователем, владеющим Maple.

Вызов и присоединение ПК имеет следующий вид:

read («имя устройства: \\путь\\имя ПК»)

with (имя ПК)

После этого будет выведен список процедур ПК, например:

[Add, AddHNS, Conjug, ConvertA, DirSum2, DirSumN, Divis, GenIso, HNSnumber, LibHNS, ListHNS, MultiDim, Norma, Rad2, Refill, RefillHNS, SearchHNS, SqrtEq, Subtr, Trans, Unit, VizHNS, VizInA, VizLibHNS, InAdd, InConvertHNS, InMulti, NameBas, NatMulti, RenamA].

Поскольку ПК предназначен для оперирования с данными в гиперкомплексном виде, при его разработке особое внимание было уделено способам и структурам представления данных.

Как известно, общий вид гиперкомплексного числа A таков:

$$A = a_1 e_1 + a_2 e_2 + \ldots + a_n e_n, \tag{1}$$

где n — размерность ГЧС; a_i — алгебраические выражения; e_i — элементы базиса ГЧС («мнимые единицы»). Такую форму гиперкомплексного числа будем называть натуральной. Как показывает опыт, оперирование с гиперкомплексными числами в натуральной форме — неудобно. Это связано с тем, что различные операции выполняются с коэффициентами при базисных элементах, которые необходимо выделять и идентифицировать.

В качестве примера рассмотрим такую простую операцию, как сложение гиперкомплексных чисел. Пусть наряду с (1) есть число $B = b_1 e_1 + b_2 e_2 + \ldots + b_n e_n$. Если в рамках Maple использовать операцию сложения, то получим следующее:

$$C = A + B = a_1 e_1 + a_2 e_2 + \ldots + a_n e_n + b_1 e_1 + b_2 e_2 + \ldots + b_n e_n. \tag{2}$$

Выражение (2) не является натуральной формой гиперкомплексного числа. В нем следует выполнить команду приведения подобных. В системе Maple есть такая команда — collect, но при ее применении необходимо указывать переменную, по которой проводится приведение подобных. Поэтому в данном случае придется n раз применить команду collect, каждый раз указывая, по какой переменной выполняется приведение подобных:

$$C = \text{collect}\,((\text{collect}\,(\text{collect}\,(\ldots(A+B), e_1), \ldots), e_n) =$$
$$= (a_1 + b_1)\, e_1 + \ldots + (a_n + b_n)\, e_n. \tag{3}$$

Как видим, (3) — достаточно громоздкая конструкция, особенно при больших размерностях ГЧС. Таких неудобств, связанных с использованием натуральной формы представления гиперкомплексного числа, очень много. В то же время, в системе Maple имеются средства, позволяющие избавиться от этих и многих других неудобств, связанных с использованием натуральной формы представления гиперкомплексного числа. В натуральной форме представления гиперкомплексного числа важны только коэффициенты при элементах базиса и их порядковые номера в изображении гиперкомплексного числа, т.е. гиперкомплексное число можно представить виде вектора. Однако векторно-матричная форма в данном случае не подходит, так как компоненты матрицы и вектора должны быть однотипними. В системе Maple есть такая форма представления данных, как список — list, т.е. упорядоченный набор разнотипных данных.

Для оперирования с данными в формате списков в Maple существуют многочисленные команды, позволяющие задавать список, определять его длину, складывать два списка одинаковой длины, определять член списка по его порядковому номеру, умножать все члены списка на любое выражение и др. Представление гиперкомплексного числа в виде списка называется списочным или внутренним представлением гиперкомплексного числа. Таким образом, вместо (1) используем представление

$$A=[a_1,a_2,...,a_n]. \tag{4}$$

Тогда сумма двух чисел определяется очень просто:

$$C=A+B=[a_1,...,a_n]+[b_1,...,b_n]=[a_1+b_1,...,a_n+b_n],$$

т.е. приведение подобных символьных коэффициентов в соответствии с их порядковыми номерами в числах выполняется автоматическими внутренними средствами Maple. Одной командой выполняется и умножение всех членов списка на одно и то же выражение: $\lambda A=[\lambda a_1,\lambda a_2, ...,\lambda a_n]$.

Таким образом, представление гиперкомплексных чисел в формате списков значительно упрощает разработку программных средств. Однако такое решение требует наличия в ПК процедур для взаимно-обратного преобразования натуральной и внутренней форм представления гиперкомплексных чисел. Тем более, что некоторые действия целесообразно выполнять над числами в натуральной форме. В связи с этим во многих процедурах ПК предусматривается выход в форме списка из двух элементов: первый элемент — результат в списочной форме, второй — в натуральной. Например, работа процедуры генерации гиперкомплексного числа восьмой размерности выглядит так:

> A = HNSnumber(8, a, e):
> A [1]

$$[a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8]$$

> A [2]

$$a_1e_1+a_2e_2+a_3e_3+a_4e_4+a_5e_5+a_6e_6+a_7e_7+a_8e_8.$$

Целесообразно придать списочный формат и более сложным гиперкомплексным структурам. Так, таблица Кэли умножения базисных элементов представляется трехуровневой списочной структурой: верхний уровень состоит из списка строк таблицы, второй, вложенный, уровень — список ячеек таблицы, третий, самый нижний уровень, — список структурных констант одной ячейки. Таблица Кэли для обобщенных кватернионов в натуральной форме имеет вид

$H_{\alpha\beta}$	e_1	e_2	e_3	e_4
e_1	e_1	e_2	e_3	e_4
e_2	e_2	$-\alpha\, e_1$	e_4	$-\alpha\, e_3$
e_3	e_3	$-e_4$	$-\beta\, e_1$	$\beta\, e_2$
e_4	e_4	$\alpha\, e_3$	$-\beta\, e_2$	$-\alpha\,\beta\, e_1$

,

а в списочной форме — следующий вид

$$[[[1,0,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1]],[[0,1,0,0],[-\alpha,0,0,0],[0,0,0,1],[0,0,-\alpha,0]],$$

$$[[0,0,1,0],[0,0,0,-1],[-\beta,0,0,0],[0,\beta,0,0]],[[0,0,0,1],[0,0,\alpha,0],[0,-\beta,0,0],[-\alpha\beta,0,0,0]]].$$

Структура пакета гиперкомплексных вычислений включает следующие подсистемы:

выполнения алгебраических операций в ГЧС;

манипуляции с ГЧС и таблицами Кэли;

определения алгебраических характеристик гиперкомплексных выражений;

хранения часто употребляемых выражений;

выполнения модульных операций с гиперкомплексными выражениями;

визуализации и сервиса.

П о д с и с т е м а в ы п о л н е н и я а л г е б р а и ч е с к и х о п е р а ц и й в Г Ч С состоит из следующих процедур:

HNSnumber(n,Name,NameBas) — генерация гиперкомплексного числа n-й размерности;

inAdd(A,B) — сложение двух гиперкомплексных чисел, A и B, в списочном виде;

Add(A,B,dimHNS) — сложение двух гиперкомплексных чисел, A и B, размерности dimHNS в натуральном виде;

Subtr(A,B,dimHNS) — вычитание двух гиперкомплексных чисел, A и B, размерности dimHNS в натуральной форме;

inMulti(A,B,HNS) — умножение двух чисел в виде списка;

natMulti(A,B,HNS,nBas) — умножение двух гиперкомплексных чисел в натуральной форме;

Divis(A,B,nameHNS) — деление чисел в списочном виде;

Rad2(A,Name, nameBas) — извлечение квадратного корня из гиперкомплексного числа в любой форме;

SqrtEq(A,B,C,NameHNS) — решение гиперкомплексного квадратного уравнения.

Подсистема манипуляции с ГЧС и таблицами Кэли состоит из следующих процедур:

inConvertHNS (*M*,Name) — преобразование таблицы Кэли из естественного вида в списковый (список структурных констант);

VizHNS(Spis, nam) — визуализация списка ГЧС в таблицу Кэли с данным базисом;

nameBas(*A*) — определение идентификатора базиса ГЧС по гиперкомплексному числу в естественном виде;

renam*A*(*A*,nam,dimHNS) — переименование идентификатора базиса в гиперкомплексном числе в естественном виде;

VizHNS(Spis, nam) — визуализация списка ГЧС в таблицу Кэли с данным базисом;

LibHNS() — хранилище таблиц Кэли ГЧС;

SearchHNS(nameHNS, nameRepos) — процедура поиска ГЧС в хранилище по ее имени;

VizIn*A*(in*A*,*E*) — визуализация гиперкомплексного числа из списковой формы в естественную;

Convert*A*(*A*,DimHNS) — преобразование гиперкомплексного числа из естественной формы в списковую;

Refill(Spisok, Element) — пополнение списка на один элемент;

ListHNS(DimHNS) — генерация списка-шаблона для внутреннего представления ГЧС;

nameBas(*A*) — определение идентификатора базиса ГЧС по гиперкомплексному числу в естественном виде;

renam*A*(*A*,nam,dimHNS) — переименование идентификатора базиса в гиперкомплексном числе в естественном виде;

RefillHNS(nameLib, nameHNS) — удаление ГЧС из хранилища;

VizLibHNS(LibHNS) — просмотр всех ГЧС в естественном виде;

Trans(*M*,*s*,*t*) — транспозиция строк и столбцов таблицы Кэли ГЧС;

AddHNS(Name, Table, Comment, Type) — пополнение хранилища ГЧС;

GenIzo (*L*,nameHNS, newBas) — генерация изоморфной ГЧС путем линейного преобразования базиса;

DirSum2(Name1,Name2) — построение прямой суммы двух ГЧС;

DirSumN (Spisok, nameBas) — построение прямой суммы нескольких ГЧС;

MultiDim(nameHNS1,nameHNS2,nameBas,markKom,nameHNS) — умножение размерности ГЧС;

SysIzo(HNS1, HNS2) — генерация системы уравнений изоморфизма двух ГЧС;

SolIzo(SysEq) — решение системы уравнений изоморфизма двух ГЧС.

Подсистема определения алгебраических характеристик гиперкомплексных выражений состоит из следующих процедур:

Norma(A,nameHNS) — определение псевдонормы гиперкомплексного числа в списочном виде;

Unit(nameHNS,name) — определение единичного элемента ГЧС;

Conjug(A,nameHNS,nam) — построение сопряженного числа;

Divis(A,B,nameHNS) — процедура деления;

HNSnumber(n,Name,NameBas) — процедура генерации гиперкомплексного числа.

Подсистема хранения часто употребляемых выражений содержит готовые формулы выполнения различных операций и вычислений для фиксированных ГЧС. Эта подсистема может пополняться и сохраняться пользователем.

Подсистема выполнения модульных операций с гиперкомплексными выражениями состоит из процедур построения системы остаточных классов по гиперкомплексным модулям, определения представимости гиперкомплексного числа, алгоритма Евклида для гиперкомплексных чисел и др.

Подсистема визуализации и сервиса состоит из следующих процедур:

VizInA(in$A$,E) — визуализация гиперкомплексного числа из списковой формы в естественную;

Convert$A$(A,DimHNS) — преобразование гиперкомплексного числа из естественной формы в списковую;

Refill(Spisok, Element) — пополнение списка на один элемент;

ListHNS(DimHNS) — генерация списка-шаблона для внутреннего представления ГЧС;

inConvertHNS (M,Name) — преобразование таблицы Кэли из естественного вида в списковый (список структурных констант);

RefillHNS(nameLib, nameHNS) — процедура удаления ГЧС из хранилища.

Такая структура и состав комплекса гиперкомплексных символьных вычислений в среде Maple позволяет значительно упростить процессы создания программного обеспечения для математического моделирования различных научно-технических задач.

Принципы построения некоторых процедур ПК. Рассмотрим следующие процедуры преобразования гиперкомплексного числа из естественной формы в списочную.

Процедура преобразования гиперкомплексного числа из естественной формы в списочную. Задача: число A в форме (1) перевести в форму (4). На первый взгляд, сделать это очень просто: выполнить подстановки $e_i = 1,\ i = 1, \dots, n,$ с помощью команды subs, после чего полученное выражение $A = a_1 + a_2 + \dots + a_n$ с помощью команды convert(A,list) превратить в (4). Однако, если в числе A один или более коэффициентов a_i равны нулю, т.е. гиперкомплексное число A — неполное, то получится список, длина которого меньше n, что приведет в дальнейшем к неправильным результатам.

Идея алгоритма, работающего правильно как с полными гиперкомплексными числами, так и неполными, заключается в следующем. Сначала генерируется список длиной n — заготовка результата:

$$\mathrm{in}\,A = [0, 0, \dots, 0]. \tag{5}$$

Затем генерируется список той же длины, состоящий из равенств $e_i = 0$:

$$\mathrm{sp} = [e_1 = 0, \dots, e_i = 0, \dots, e_n = 0].$$

Для каждого значения индексов строится список равенств

$$\mathrm{sp1} = [e_1 = 0, \dots, e_i = 1, \dots, e_n = 0] \tag{6}$$

и выполняется подстановка в исходное число списка (6). Если в числе A компонент $a_i = 0$, то он таким войдет в список (5). Таким образом, эта подстановка выделяет коэффициент a_i, который присваивается i-му элементу списка (5).

Полный текст программы, где формальными параметрами процедуры являются A — гиперкомплексное число в естественной форме, DimHNS — размерность ГЧС, в которой задано число A, имеет следующий вид:

```
Convert A := proc (A, DimHNS)
    local inA, sp, i, sp1, e, A1;
    inA := [seq (0, i = 1.. DimHNS)];
    if A < > 0 then
        A1 := subs (nameBas (A) = e, A);
        sp := [seg (e[i] = 0, i = 1.. DimHNS)];
        for i to DimHNS do
            sp1 := sp; sp1[i]:= e[i] = 1; inA[i] := subs (sp1, A1)
        end do
    end if;
    RETURN (inA)
end proc
```

П р о ц е д у р а у м н о ж е н и я г и п е р к о м п л е к с н ы х ч и с е л в с п и с о ч н о й ф о р м е выполняет умножение двух гиперкомплексных чисел, A и B, которые заданы в виде списков. Умножение проводится в соответствии с законами композиции системы HNS. Таким образом, процедура реализует обобщенную формулу умножения гиперкомплексных чисел [1, 6]

$$A\,B = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} a_i b_j \gamma_{ij}^{k}, \tag{7}$$

где n — размерность ГЧС HNS; γ_{ij}^{n} — структурные константы законов композиции ГЧС HNS.

Процедура выполняет вызов хранилища таблицы Кэли ГЧС (процедура LibHNS()), в которой осуществляется поиск информации о заданной ГЧС HNS (SearchHNS(HNS, LibHNS())). В ней содержится трехуровневый список структурных констант ГЧС HNS. Далее выполняется формула (7). Полный текст программы имеет следующий вид:

```
inMulti := proc (A, B, HNS)
    local X, k, i, j, Lib, HNSI;
    Lib := Lib HNS ( );
    HNSI := SearchHNS (HNS, Lib);
    X :=[seq ( 0, i = 1.. nops (HNSI[1]))];
    for k to nops (HNSI[1]) do
        X[k] := 0;
        for i to nops (HNSI[1]) do
            for j to nops (HNSI[1]) do
                X[k] := X[k] +A[i]*B[j]*HNSI [i, j, k]
            end do
        end do
    end do;
    RETURN (X)
end proc
```

Рассмотрим пример вызова и работы этой процедуры. Сгенерируем два гиперкомплексных числа четвертой размерности:

$$A := \text{HNSnumber}\,(4, a, e)\,[1],\; B := \text{HNSnumber}\,(4, a, e)\,[1];$$

$$A := [a_1, a_2, a_3, a_4],\; B := [b_1, b_2, b_3, b_4].$$

Построим их произведение в ГЧС $Q4N$ — некоммутативном автоудвоении обобщенной системы комплексных чисел [1, 6 — 8], таблица Кэли которой имеет вид

$Q4N$	E_1	E_2	E_3	E_4
E_1	E_1	E_2	E_3	E_4
E_2	E_2	pE_1+qE_2	E_4	pE_3+qE_4
E_3	E_3	$-E_4$	pE_1+qE_3	$-pE_2-qE_4$
E_4	E_4	$-pE_3-qE_4$	pE_2+qE_4	$-p^2E_1-pqE_2-pqE_3-q^2E_4$

Вызываем процедуру: $C:=\mathrm{inMulti}(A,B,Q4N)$. Получаем список из четырех компонентов результата:

$$[a_1b_1+a_2b_2p+a_3b_3p-a_4b_4p^2,\ a_1b_2+a_2b_1+a_2b_2q+$$
$$+a_3b_4p-a_4b_3p-a_4b_4p\,q,\ a_1b_3-a_2b_4p+a_3b_1+$$
$$+a_3b_3q+a_4b_2p-a_4b_4p\,q,\ a_1b_4-a_2b_3-a_2b_4q+$$
$$+a_3b_2+a_3b_4q+a_4b_1+a_4b_2q-a_4b_3q-a_4b_4q^2].$$

П р о ц е д у р а у м н о ж е н и я д в у х г и п е р к о м п л е к с н ы х ч и с е л в е с т е с т в е н н о й ф о р м е использует процедуру умножения в списочном виде. Сначала число преобразуется из естественной формы в списочную, затем выполняется умножение в списочной форме, после чего полученное произведение преобразуется из списочной формы в натуральную и визуализируется с заданным именем базиса. Текст процедуры:

natMulti := **proc** (*A*, *B*, HNS, *n*Bas)
VizIn*A* (inMulti (Convert*A*(*A*, nops (SearchHNS (HNS,
LibHNS ( )))), Convert*A*(*B*, nops (SearchHNS (HNS,
LibHNS ( )))), HNS), *n*Bas)
end proc

Пусть числа третьей размерности имеют естественную форму:

$$A:=a_1e_2+a_2e_2+a_3e_3\ ,\ B:=b_1e_2+b_2e_2+b_3e_3\ .$$

Найдем их произведение в системе триплексных чисел T [1] с таблицей умножения вида

T	e_1	e_2	e_3
e_1	e_1	e_2	e_3
e_2	e_2	$(e_3-e_1)/2$	$-e_2$
e_3	e_3	$-e_2$	e_1

ISSN 0204–3572. Electronic Modeling. 2017. V. 39. № 5

Тогда их произведение имеет вид

$$C := \mathrm{natMulti}(A, B, T, f),$$

$$C := \left(a_1 b_1 - \frac{1}{2} a_2 b_2 + a_3 b_3\right) f_1 + (a_1 b_2 + a_2 b_1 - a_2 b_3 - a_3 b_2) f_2 +$$

$$+ \left(a_1 b_3 + \frac{1}{2} a_2 b_2 + a_3 b_1\right) f_3,$$

где имя базиса e изменено на f. Вызов этой процедуры в ГЧС $R \oplus C$ [1] с таблицей умножения

$R \oplus C$	e_1	e_2	e_3
e_1	e_1	0	0
e_2	0	e_2	e_3
e_3	0	e_3	$-e_2$

имеет вид

$$C := \mathrm{natMulti}(A, B, R \oplus C, f),$$

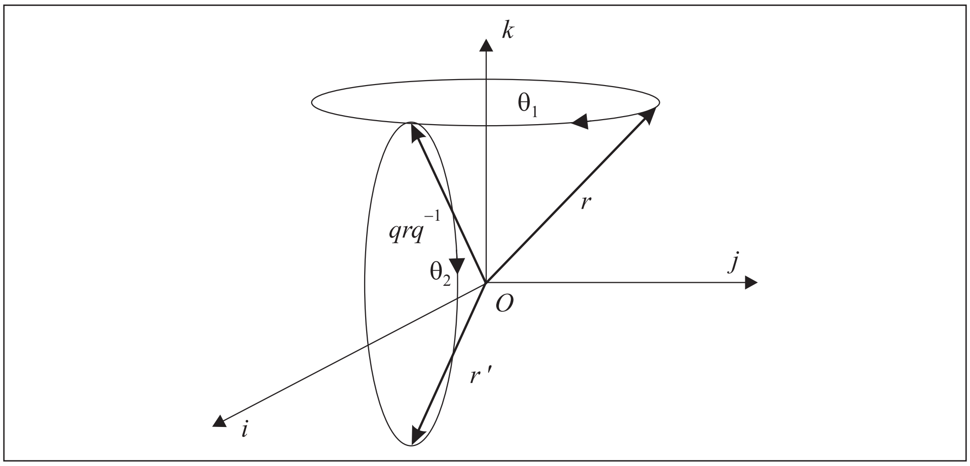
$$C := a_1 b_1 f_1 + (a_2 b_2 - a_3 b_3) f_2 + (a_2 b_3 + a_3 b_2) f_3.$$

Пример решения практической задачи с помощью ПК. Сравним решение задачи поворота вектора с помощью кватерниона традиционными средствами и с использованием процедур ПК. Эта задача часто возникает в системах ориентации в пространстве, навигации, компьютерной анимации и др. [2]. Формула поворота вектора r в пространстве с помощью кватерниона q имеет вид [9]

$$r' = q r q^{-1}, \tag{8}$$

где r и r' — начальные и конечные координаты поворачиваемой точки в кватернионном виде, т.е. этот кватернион — векторный; q и q^{-1} — прямой и обратный кватернионы, определяющие ось вращения; все умножения — кватернионные. В выражении (8) кватернион q должен быть нормирован, т.е. его норма должна быть равна единице. Если кватернион поворота имеет вид $Q = a_1 e_1 + a_2 e_2 + a_3 e_3 + a_4 e_4$, то нормированный кватернион имеет следующий вид:

$$q = \frac{a_1}{\sqrt{a_1^2 + a_2^2 + a_3^2 + a_4^2}} e_1 + \frac{a_2 e_2 + a_3 e_3 + a_4 e_4}{\sqrt{a_1^2 + a_2^2 + a_3^2 + a_4^2}}. \tag{9}$$



Геометрический смысл элементов выражения (9) таков: угол поворота вокруг оси (см. рисунок)

$$\theta = 2\arccos\frac{a_1}{\sqrt{a_1^2 + a_2^2 + a_3^2 + a_4^2}},$$

ее направляющие косинусы —

$$\cos\theta_i = \frac{a_i}{\sqrt{a_2^2 + a_3^2 + a_4^2}}, \quad i = 2, 3, 4.$$

Рассмотрим задачу определения координат точки, полученной последовательными поворотами вектора вокруг двух осей: сначала вокруг оси, определяемой кватернионом q, а затем — вокруг оси, определяемой кватернионом p, как показано на рисунке. Такой сложный поворот определяется формулой

$$r' = pqrq^{-1}p^{-1}, \tag{10}$$

где все умножения — кватернионные [1, 3]; q и q^{-1} — прямой и обратный кватернионы, определяющие первую ось вращения; p и p^{-1} — то же для второй оси вращения.

Таким образом, программа должна предусматривать нормирование кватернионов, определение обратных кватернионов, кватернионные перемножения по (10) и упрощение полученного результата. Если даны исходные нормированные по (9) кватернионы q, p и r в списочном виде, а также обратные кватернионы $q^{-1} = Cq$, $p^{-1} = Cp$, то один из возможных вариантов программы в виде процедуры без применения средств ПК имеет следующий вид:

```
Pov := proc(r, q, p, Cq, Cp)
    local AB, M1, M2, M3, M4, i, M, r1;
    AB := [a[1]*b[1] – a[2]*b[2] – a[3]*b[3] – a[4]*b[4],
    a[1]*b[2] + a[2]*b[1] + a[3]*b[4] – a[4]*b[3],
    a[1]*b[3] – a[2]*b[4] + a[3]*b[1] + a[4]*b[2],
    a[1]*b[4] + a[2]*b[3] – a[3]*b[2] + a[4]*b[1]];
    M1 := subs (a[1] = p[1], a[2] = p[2], a[3] = p[3], a[4] = p[4],
    b[1] = q[1], b[2] = q[2], b[3] = q[3], b[4] = q[4], AB);
    M2 := subs (a[1] = M1[1], a[2] = M1[2], a[3] = M1[3], a[4] = M1[4],
    b[1] = r[1], b[2] = r[2], b[3] = r[3], b[4] = r[4], AB);
    M3 := subs (a[1] = M2[1], a[2] = M2[2], a[3] = M2[3], a[4] = M2[4],
    b[1] = Cq[1], b[2] = Cq[2], b[3] = Cq[3], b[4] = Cq[4], AB);
    M4 := subs (a[1] = M3[1], a[2] = M3[2], a[3] = M3[3], a[4] = M3[4],
    b[1] = Cp[1], b[2] = Cp[2], b[3] = Cp[3], b[4] = Cp[4], AB);
    for i to 4 do M[i] := factor (M4[i]) end do;
    r1 := [M[1], M[2], M[3], M4[4]];
    RETURN (r1)
end proc
```

Как видим, основной объем этой программы занимают вычисления произведений кватернионов, которые выполнены в виде их подстановок в общую формулу. Поскольку в ПК есть процедуры умножения гиперкомплексных чисел, эта программа существенно упрощается. Программа с использованием средств ПК выглядит так:

```
Pov1:= proc(r, q, p, Cq, Cp) for i from 1 to 4 do r1[i]
    := factor (inMulti(inMulti(inMulti(inMulti(p, q, H), r, H),
    Cq[1], H), Cp[1], H) [i]): end do;
    RETURN (r1)
end proc
```

Естественно, результаты вычислений по обеим программам — одинаковы. Так, если решается задача поворота точки с координатами [1, 2, 3] сначала на угол $\theta_1 = \pi/3$ вокруг оси, определяемой ортом [0, 0, 1], т.е. оси *Oz*, затем — на угол $\theta_2 = \pi/2$ вокруг оси, определяемой ортом [0, 1, 0], т.е. оси *Oy*, то конечное положение точки определяется векторным кватернионом $r' = 3e_2 + (\sqrt{3}+0{,}5)\, e_3 + (\sqrt{3}-0{,}5)\, e_4$.

Выводы

Разработанные программно-алгоритмические средства обеспечивают большую область вычислений, связанных с моделированием процессов, описываемых гиперкомплексными числами. Их использование значительно упрощает процесс разработки программного обеспечения и повышает его надежность, так как применяются многократно проверенные алгоритмы и программы. В то же время, ПК нуждается в дальнейшем пополнении и расширении, что является направлением дальнейшей научно-исследовательской работы в этой области.

СПИСОК ЛИТЕРАТУРЫ

1. *Синьков М.В.* Конечномерные гиперкомплексные числовые системы. Основы теории. Применения / М.В. Синьков, Я.А. Калиновский, Ю.Е. Бояринова. Киев: НАН України, Ін-т проблем реєстрації інформації, 2010, 389 с.
2. *Калиновский Я.А.* Высокоразмерные изоморфные гиперкомплексные числовые системы и их использование для повышения эффективности вычислений / Я.А.Калиновский, Ю.Е.Бояринова. Киев: Инфодрук, 2012, 183с.
3. *Каліновський Я.О.* Методи комп'ютерного моделювання та обчислень з використанням гіперкомплексних числових систем: Дис. … д-ра техн. наук : 01.05.02. ІПРІ НАН України. Київ, 2007, 417 с.
4. *Калиновский Я.А.* Гиперкомплексные числовые системы и быстрые алгоритмы цифровой обработки информации / Я.А. Калиновский, Д.В. Ландэ, Ю.Е. Бояринова, Я.В. Хицко. Киев: ИПРИ НАН Украины, 2014, 130 с.
5. *Von zur Gathen J.* Modern Computer Algebra // J. von zur Gathen, J. Gerhard. Cambrige: Cambrige University Press, 2013, 808 p.
6. *Kalinovsky Y.A., Boyarinova Yu.E., Sukalo A.S* .Study of relations between the generalized quaternions and procedure of doubling of hypercomplex numerical systems // Storage & Processing. 2015, Vol. 17, № 1, p. 36—45.
7. *Kalinovsky Ya.* The structure of a hyperfast calculation method linear convolution of discrete signals // Ibid. 2013, Vol. 15, № 1, p. 31—44.
8. *Kalinovsky Y.A., Lande D.V., Boyarinova Y.E., Khitsko Y.V.* Some isomorphic classes for noncanonical hypercomplex number systems of dimension 2, -arXiv preprint arXiv: 1403. 2273, 2014.
9. *Liefke H.* Quaternion Calculus for Modeling Rotations in 3D Space. [Electronic resource] : www.liefke.com/ hartmut (1998).

Поступила 31.05.17;
после доработки 07.08.17

REFERENCES

1. Sinkov, M.V., Kalinovsky, Ya.A. and Boyarinova, Yu.E. (2010), *Konechno-mernyie giperkompleksnyie chislovyie sistemy. Osnovy teorii. Primeneniya* [Finite-dimensional hypercomplex number systems. Fundamentals of the theory. Applications], NAS of Ukraine, Institute for Information Recording of NAS of Ukraine, Kyiv, Ukraine.
2. Kalinovsky, Ya.A. and Boyarinova, Yu.E. (2012), *Vysokorazmernyie izomorfnyie giperkompleksnyie chislovyie sistemy i ikh ispolzovaniye dlya povysheniya effektivnosti vychisleniy* [High-dimensional isomorphic hypercomplex numerical systems and their use for increasing efficiency of computations], Infodruk, Kyiv, Ukraine.
3. Kalinovsky, Ya. A. (2007), "Development of the methods of the theory of HNS for a mathematical modeling and computer calculations", Dr. Sci. (Tech.) thesis, 01.05.02, Institute for Information Recording of NAS of Ukraine, Kyiv, Ukraine.

4. Kalinovsky, Ya.A., Lande, D.V., Boyarinova, Yu.E. and Khitsko, Ya.V. (2014), *Giperkompleksnye chislovye sistemy i bystrye algoritmy tsifrovoy obrabotki informatsii* [Hypercomplex number systems and fast algorithms for digital information processing], Institute for Information Recording of NAS of Ukraine, Kyiv, Ukraine.
5. Von zur Gathen, J. and Gerhard, J. (2013), Modern computer algebra, Cambrige University Press, Cambridge, UK.
6. Kalinovsky, Ya.A., Boyarinova, Yu.E. and Sukalo, A.S (2015), "Study of relations between the generalized quaternions and procedure of doubling of hypercomplex numerical systems", *Data Rec., Storage & Processing*, Vol. 17, no. 1, pp. 36-45.
7. Kalinovsky, Ya.A. (2013), "The structure of a hyperfast calculation method linear convolution of discrete signals", *Data Rec., Storage & Processing,* Vol. 15, no.1, pp. 31-44.
8. Kalinovsky, Ya.A., Lande, D.V., Boyarinova, Yu.E. and Khitsko, Ya.V. (2014), Some isomorphic classes for noncanonical hypercomplex number systems of dimension 2, -arXiv preprint arXiv:1403.2273, 2014.
9. Liefke, H. (1998), "Quaternion calculus for modeling rotations in 3D space", available at: www.liefke.com/ hartmut.

Received 31.05.17;
after revision 07.08.17

Ya.A. Kalinovsky, Yu.E. Boyarinova, A.S. Sukalo, Ya.V. Khitsko

SOFTWARE COMPLEX FOR HYPERCOMPLEX COMPUTATIONS

The principles assumed as a basis of the algorithmic-software complex of hypercomplex computations are stated. The complex structure and the composition of the functional subsystems are described. The most important procedures performed by the subsystems are considered, program listings and examples of their application are given.

K e y w o r d s: hypercomplex numerical system, procedure, computer algebra, operation, Maple.

КАЛИНОВСКИЙ Яков Александрович, д-р техн. наук, ст. науч. сотр. Ин-та проблем регистрации информации НАН Украины. В 1965 г. окончил Киевский политехнический ин-т. Область научных исследований — теория гиперкомплексных числовых систем и их применение в математическом моделировании.

БОЯРИНОВА Юлия Евгеньевна, канд. техн. наук, ст. науч. сотр. Ин-та проблем регистрации информации НАН Украины, доцент Национального технического университета Украины «Киевский политехнический ин-т», который окончила в 1997 г. Область научных исследований — теория гиперкомплексных числовых систем и их применение в математическом моделировании.

СУКАЛО Алина Сергеевна, аспирантка Ин-та проблем регистрации информации НАН Украины. В 2013 г. окончила Житомирский госуниверситет. Область научных исследований — математическое моделирование и вычислительные процессы.

ХИЦКО Яна Владимировна, канд. техн. наук, мл. науч. сотр. Национального технического университета Украины «Киевский политехнический ин-т», который окончила в 2005 г. Область научных исследований — теория гиперкомплексных числовых систем и их применение в математическом моделировании.