ГИПЕР-ДУАЛЬНЫЕ ТЕНЗОРЫ 2-ГО РАНГА С ГИПЕР-ДУАЛЬНОЙ АЛГЕБРОЙ $\varepsilon^2 = 2\omega, \varepsilon\omega = \omega^2 = 0$

Олифер В.И.

Цель работы

Целью настоящей работы является обобщение предыдущих исследований автора, посвящённых гипер-дуальным числам [1], гипер-дуальным собственным значениям [2] и гипер-дуальным собственным векторам [3]. А также создание более компактного и читаемого программного кода, реализующего полученные алгоритмы.

1. Гипер-дуальные тензоры

Алгебра гипер-дуальных чисел 2-го порядка или тензоров 0-го ранга (HDN) следует соотношениям, полученных в [1]:

$$\begin{aligned} &X = x + x_1\boldsymbol{\varepsilon} + x_2\boldsymbol{\omega} \text{ и } Y = y + y_1\boldsymbol{\varepsilon} + y_2\boldsymbol{\omega}, \\ &X \pm Y = x \pm y + (ax_1 \pm y_1)\boldsymbol{\varepsilon} + (x_2 \pm y_2)\boldsymbol{\omega}, \\ &X{\cdot}Y = x{\cdot}y + (x{\cdot}y_1 + y{\cdot}x_1)\boldsymbol{\varepsilon} + (x{\cdot}y_2 + 2x_1{\cdot}y_1 + y{\cdot}x_2)\boldsymbol{\omega}, \\ &(x, x_1, x_2), (y, y_1, y_2) \in \mathbb{R}, \end{aligned} \tag{1.1}$$

а алгебра гипер-дуальных векторов или тензоров 1-го ранга (HDV) определяется так:

$$\begin{aligned} &\vec{X} = \vec{x} + \vec{x}_1\boldsymbol{\varepsilon} + \vec{x}_2\boldsymbol{\omega} \text{ и } \vec{Y} = \vec{y} + \vec{y}_1\boldsymbol{\varepsilon} + \vec{y}_2\boldsymbol{\omega}, \\ &\vec{X} \pm \vec{Y} = \vec{x} \pm \vec{y} + (a\vec{x}_1 \pm y_1)\boldsymbol{\varepsilon} + (\vec{x}_2 \pm y_2)\boldsymbol{\omega}, \\ &\vec{X}{\cdot}\vec{Y} = \vec{x}{\cdot}\vec{y} + (\vec{x}{\cdot}y_1 + \vec{y}{\cdot}\vec{x}_1)\boldsymbol{\varepsilon} + (\vec{x}{\cdot}y_2 + 2\vec{x}_1{\cdot}y_1 + y{\cdot}\vec{x}_2)\boldsymbol{\omega}, \\ &(\vec{x}, \vec{x}_1, \vec{x}_2), (\vec{y}, y_1, y_2) \in \mathbb{R}_3, \end{aligned} \tag{1.2}$$

и наконец, алгебра гипер-дуальных тензоров 2-го ранга (HDT) определяется аналогично алгебре гипер-дуальных матриц [3] как:

$$\begin{aligned} &\boldsymbol{A} = \boldsymbol{A_0} + \boldsymbol{A_1}\boldsymbol{\varepsilon} + \boldsymbol{A_2}\boldsymbol{\omega}, \quad \boldsymbol{B} = \boldsymbol{B_0} + \boldsymbol{B_1}\boldsymbol{\varepsilon} + \boldsymbol{B_2}\boldsymbol{\omega}, \\ &\boldsymbol{A} \pm \boldsymbol{B} = \boldsymbol{A_0} \pm \boldsymbol{B_0} + (\boldsymbol{A_1} \pm \boldsymbol{B_1})\boldsymbol{\varepsilon} + (\boldsymbol{A_2} \pm \boldsymbol{B_2})\boldsymbol{\omega}, \\ &\boldsymbol{A}{\cdot}\boldsymbol{B} = \boldsymbol{A_0}{\cdot}\boldsymbol{B_0} + (\boldsymbol{A_0}{\cdot}\boldsymbol{B_1} + \boldsymbol{A_1}{\cdot}\boldsymbol{B_0})\boldsymbol{\varepsilon} + (\boldsymbol{A_0}{\cdot}\boldsymbol{B_2} + 2\boldsymbol{A_1}{\cdot}\boldsymbol{B_1} + \boldsymbol{A_2}{\cdot}\boldsymbol{B_0})\boldsymbol{\omega}, \\ &\boldsymbol{A^n} = \boldsymbol{A}{\cdot}\boldsymbol{A^{n-1}}; \ n = 1, 2, 3, \dots, \end{aligned} \tag{1.3}$$

где элементы тензоров $\boldsymbol{A_0}, \boldsymbol{A_1}, \boldsymbol{A_2}$ и $\boldsymbol{B_0}, \boldsymbol{B_1}, \boldsymbol{B_2} \in \mathbb{R}_{3\times3}$, а тензоры $\boldsymbol{A_1}, \boldsymbol{A_2}$ и $\boldsymbol{B_1}, \boldsymbol{B_2}$ можно трактовать как тензоры возмущений 1-го и 2-го рода соответственно.

Гипер-дуалный базис всех гипер-дуальных тензоров – суть $(1, \boldsymbol{\varepsilon}, \boldsymbol{\omega})$. Первые слагаемые гипер-дуальных тензоров называются их *главной* частью, а второе и третье слагаемые – *мнимыми* частями. Наприме, $X.Re = x,\ \ X.Im1 = x_1,\ \ X.Im2 = x_2.$

2. Характеристическое уравнение

Характеристическое уравнение или полином [2] в терминах гипер-дуальных тензоров запишется в виде

$$\Upsilon(\Lambda) = \det(\boldsymbol{A} - \Lambda I) = \Lambda^3 - I_1\Lambda^2 + I_2\Lambda - I_3 = 0, \tag{2.1}$$

где: $\boldsymbol{A} = \boldsymbol{A_0} + \boldsymbol{A_1\varepsilon} + \boldsymbol{A_2\omega},\ \ \Lambda = \lambda + \lambda_1\boldsymbol{\varepsilon} + \lambda_2\boldsymbol{\omega};\ \ I_1,\ I_2,\ I_3 -$ гипер-дуальные инварианты тензора $\boldsymbol{A}$, которые вычисляются по формулам ($I_1,\ I_2,\ I_3 \in$ HDN):

$$I_1 = \mathrm{tr}(\boldsymbol{A}), \qquad I_2 = \left(I_1^2 - \mathrm{tr}(\boldsymbol{A^2})\right)/2, \qquad I_3 = I_1^3/6 - I_1\,\mathrm{tr}(\boldsymbol{A^2})/2 + \mathrm{tr}(\boldsymbol{A^3})/3, \tag{2.2}$$

с $\mathrm{tr}(\boldsymbol{A}) = \mathrm{tr}(\boldsymbol{A_0}) + \mathrm{tr}(\boldsymbol{A_1})\boldsymbol{\varepsilon} + \mathrm{tr}(\boldsymbol{A_2})\boldsymbol{\omega},\ \ \mathrm{tr}(.) -$ след соответствующего тензора.

3. Определение собственных значений

Подстановка следующих выражений

$I_i = I_i.Re + I_i.Im1\boldsymbol{\varepsilon} + I_i.Im2\boldsymbol{\omega},\ \ \Lambda^i = \lambda^i + i\lambda^{i-1}\lambda_1\boldsymbol{\varepsilon} + i\left[\lambda^{i-1}\lambda_2 + (i-1)\lambda^{i-2}\lambda_1^i\right]\boldsymbol{\omega},\ \ i = 1,2,3$

в (2.1) даёт гипер-дуальную функцию вида $\Upsilon(\Lambda) = \mu(\lambda) + \mu_1(\lambda,\lambda_1)\boldsymbol{\varepsilon} + \mu_2(\lambda,\lambda_1,\lambda_2)\boldsymbol{\omega}$, равенство нулю которой определяется системой алгебраических уравнений относительно $\lambda, \lambda_1, \lambda_2$:

$$\begin{cases} \mu(\lambda) &= \lambda^3 - \lambda^2 I_1.Re + \lambda\, I_2.Re - I_3.Re = 0 \\ \mu_1(\lambda,\lambda_1) &= \lambda_1\gamma - (\lambda^2 I_1.Im1 - \lambda I_2.Im1 + I_3.Im1) = 0 \\ \mu_2(\lambda,\lambda_1,\lambda_2) &= \lambda_2\gamma - [\lambda^2 I_1.Im2 - \lambda I_2.Im2 - 6\lambda\,\lambda_1^2 + 4\lambda\lambda_1 I_1.Im1 + I_3.Im2 - \eta] = 0 \end{cases} \tag{3.1}$$

где: $\gamma(\lambda) = 3\lambda^2 - 2\lambda I_1.Re + I_2.Re,\ \ \eta(\lambda_1) = 2\lambda_1(I_2.Im1 - \lambda_1 I_1.Re)$

Особенность полученной системы (3.1) состоит в следующем: $\mu(\lambda)$ – вещественный полином 3-й степени относительно λ, а $\mu_1(\lambda,\lambda_1)$ и $\mu_2(\lambda,\lambda_1,\lambda_2)$ линейны относительно λ_1 и λ_2 соответственно, что даёт

$$\begin{cases} \mu(\lambda) = \lambda^3 - \lambda^2 I_1.Re + \lambda\, I_2.Re - I_3.Re = 0 \\ \lambda_1 = (\lambda^2 I_1.Im1 - \lambda I_2.Im1 + I_3.Im1)/\gamma = 0 \\ \lambda_2 = [\lambda^2 I_1.Im2 - \lambda I_2.Im2 - 6\lambda\,\lambda_1^2 + 4\lambda\lambda_1 I_1.Im1 + I_3.Im2 - \eta]/\gamma = 0 \end{cases} \tag{3.2}$$

Таким образом, в общем случае, полином $\mu(\lambda) = 0$ имеет три вещественных корня (комплексные корни здесь не рассматриваются) λ_{11}, λ_{12} и λ_{13} по которым последовательно вычисляются $(\lambda_{12}, \lambda_{13})$, $(\lambda_{22}, \lambda_{23})$ и $(\lambda_{32}, \lambda_{33})$ соответственно, и тогда решение гипер-дуального уравнения (2.1) можно записать в виде $\Lambda_i = \lambda_{i1} + \lambda_{i2}\boldsymbol{\varepsilon} + \lambda_{i3}\boldsymbol{\omega}$.

Графическая схема описанного процесса представлена на следующем рисунке.

$\downarrow$	λ_{11}	$\rightarrow$	λ_{12}	$\rightarrow$	λ_{13}
$\downarrow$	λ_{21}	$\rightarrow$	λ_{22}	$\rightarrow$	λ_{23}
$\downarrow$	λ_{31}	$\rightarrow$	λ_{32}	$\rightarrow$	λ_{33}

Последовательность вычислительного процесса

4. Определение собственных векторов

Гипер-дуальный вектор $\vec{V}$ называтся собственным ветором гипер-дуально тензора $\boldsymbol{A}$, если при заданном Λ выполняется соотношение $(\boldsymbol{A} - \Lambda I)\vec{V} = 0$. Пусть Λ один из корней гипер-дуального полинома (2.1), а $\vec{V} = \vec{v}_0 + \vec{v}_1\,\boldsymbol{\varepsilon} + \vec{v}_2\boldsymbol{\omega}$, тогда

$$
\begin{aligned}
&(\boldsymbol{A} - \lambda I)\vec{V} = (\boldsymbol{A}_0 - \lambda_0 I)\vec{v}_0 + \\
&+[(\boldsymbol{A}_0 - \lambda_0 I)\vec{v}_1 + (\boldsymbol{A}_1 - \lambda_1 I)\vec{v}_0]\boldsymbol{\varepsilon} + \\
&+ [(\boldsymbol{A}_0 - \lambda_0 I)\vec{v}_2 + (\boldsymbol{A}_2 - \lambda_2 I)\vec{v}_0 + 2(\boldsymbol{A}_1 - \lambda_1 I)\vec{v}_1\ ]\boldsymbol{\omega} = \\
&= \boldsymbol{B}_0\vec{v}_0 + (\boldsymbol{B}_0\vec{v}_1 + \boldsymbol{B}_1\vec{v}_0)\boldsymbol{\varepsilon} + (\boldsymbol{B}_0\vec{v}_2 + \boldsymbol{B}_2\vec{v}_0 + 2\boldsymbol{B}_1\vec{v}_1)\boldsymbol{\omega} \Longrightarrow \\
&\Longrightarrow \begin{cases} \boldsymbol{B}_0\vec{v}_0 = 0; & \text{Уровень } \mathbf{0} \\ \boldsymbol{B}_0\vec{v}_1 = -\boldsymbol{B}_1\vec{v}_0; & \text{Уровень } \boldsymbol{\varepsilon} \\ \boldsymbol{B}_0\vec{v}_2 = -(\boldsymbol{B}_2\vec{v}_0 + 2\boldsymbol{B}_1\vec{v}_1); & \text{Уровень } \boldsymbol{\omega} \end{cases}
\end{aligned}
\qquad (4.1)
$$

Так как $\boldsymbol{B}_0, \boldsymbol{B}_1, \boldsymbol{B}_1$ вещественные тензоры 2-го ранга, а векторы $\vec{v}_0, \vec{v}_1, \vec{v}_2$ вещественные тензоры 1-го ранга, которые описываются вещественными элементами, то на уровне **0** небходимо решить однородную систему линейных алгебраических уравнений, далее использовать её решение на уровне $\boldsymbol{\varepsilon}$ для решения неоднородной системы уравнений, и наконец, полученные решения уровней **0** и $\boldsymbol{\varepsilon}$, учесть в уровне $\boldsymbol{\omega}$ при решении неоднородной системы уравнений.

5. Численная (компьютерная) реализация

Программная реализация алгоритмов (3.2) и (4.1) была выполнена на языке C# (см. Приложение). При этом использовались библиотек `MathNet.Numerics.LinearAlgebra` и `MathNet.Numerics.RootFinding`. Программный код состоит из двух новых типов данных, описываемых структурами `T0` и `T2`. Структура `T0` представляет алгебру гипер-дуальных чисел с переопределёнными арифметическими операциями $\pm$, $\times$, $\div$. Вторая структура `T2` содержит алгебру гипер-дуальных тензоров 2-го ранга, переопределённые арифметические операции $\pm$, $\times$, $\div$ и методы: `TR(.)` – вычисление гипер-дуального следа тензора, `Invariants(.)` – вычисление гипер-дуальных инвариантов, `getΛ(.)` – вычисление собственных значений, `GetV(.)` – вычисление собственных векторов и

`SolveStable(.)` – решение неоднородной системы. Процедура `GetV(.)` использует `Svd(.)` и `SolveStable(.)`, причем `Svd(.)` (Singular value decomposition) является методом `MathNet.Numerics.LinearAlgebra` [4], который позволяет вычислять сингулярные векторы. Представленный программный код был разработан исключительно для проведения численных экспериментов.

6. Результаты расчета

Численные результаты последовательного решения гипер-дуальных уравнений (3.1) и (4.1) для гипер-дуального тензора

$$\mathbf{T} = \begin{bmatrix} 5 & 1 & 4 \\ 3 & 3 & 2 \\ 6 & 2 & 10 \end{bmatrix} + \begin{bmatrix} -5 & -1 & -4 \\ 3 & 3 & 2 \\ 6 & 2 & 10 \end{bmatrix} \boldsymbol{\varepsilon} + \begin{bmatrix} -5 & 1 & 4 \\ -3 & 3 & 2 \\ -6 & 2 & 10 \end{bmatrix} \boldsymbol{\omega},$$

представлены в следующих таблицах.

I	**1**	$\boldsymbol{\varepsilon}$	$\boldsymbol{\omega}$
I_1	18	8	8
I_2	64	52	28
I_3	64	64	-64

Табл. 1. Гипер-дуальные инварианты тензора ***T***

Λ	**1**	$\boldsymbol{\varepsilon}$	$\boldsymbol{\omega}$
Λ_1	13.65685424949238	6.4142135623730985	9.7937860531805043
Λ_2	2.00000000000000	-2.0000000000000000	22.000000000002025
Λ_3	2.34314575050762	3.5857864376268984	-23.793786053180231

Табл. 2. Корни гипер-дуального уравнения (3.1)

$\vec{V}$		x	y	z
$\vec{V}_0$	**1**	0.4290834926044791	0.28184531550428449	0.85816698520895829
	$\boldsymbol{\varepsilon}$	-0.8092339594808842	0.03737381342479135	0.39234240565653727
	$\boldsymbol{\omega}$	1.1917301639904823	-0.23365384116828214	-0.51912682275443922
$\vec{V}_1$	**1**	0.3162277660168362	-0.94868329805051455	10^{-15}
	$\boldsymbol{\varepsilon}$	2.2768399153212697	0.75894663844041577	-1.89736659610105800
	$\boldsymbol{\omega}$	-21.288453208254932	-7.09615106941823810	20.1120859186721110
$\vec{V}_2$	**1**	-0.0918364863508703	0.97868804982672186	-0.18367297270174082
	$\boldsymbol{\varepsilon}$	0.8951836055244460	-0.05075343954533695	-0.71802780670976185
	$\boldsymbol{\omega}$	-20.558871321876559	1.09852301320589250	16.1328354816624610

Табл. 3. Корни гипер-дуального уравнения (4.1)

Численный эксперимент проводился на компьютере с операционной системой Microsoft Windows 10 Pro, CPU 3.40 GHz, RAM 16 MB в среде MS VS 2021. Максимальная невязка полученных решений для Λ и $\vec{V}$ составила не более 10^{-13}, а время полного расчёта – порядка 9 мс.

Заключение

Представленный метод вычисления собственных значений и векторов гипер-дуальных тензоров 2-го ранга позволяет корректно учитывать нильпотентную структуру алгебры и обеспечивает устойчивость численных процедур. Программный код получился достаточно компактным и легко читаемым с впечатляющим быстродействием.

Приложение

```
using MathNet.Numerics.LinearAlgebra;
using MathNet.Numerics.LinearAlgebra.Factorization;
using MathNet.Numerics.LinearAlgebra.Double;
using MathNet.Numerics.RootFinding;
using System.Collections.Generic;
using System.Linq;
using System;

public struct T0
{  public double Re, Im1, Im2;
   public T0(double x, double x1, double x2) { Re = x; Im1 = x1; Im2 = x2; }
   public static T0 operator *(double x, T0 X)
      { return new T0(x * X.Re, x * X.Im1, x * X.Im2);}
   public static T0 operator /(T0 X, double x)
      { return new T0(X.Re / x, X.Im1 / x, X.Im2 / x);}
   public static T0 operator *(T0 X, T0 Y)
      { return new T0(X.Re * Y.Re,
        X.Re * Y.Im1 + Y.Re * X.Im1,
        X.Re * Y.Im2 + 2.0 * X.Im1 * Y.Im1 + Y.Re * X.Im2);
      }
   public static T0 operator ^(T0 X, int n)
      { return new T0(Math.Pow(X.Re, n),
        n * Math.Pow(X.Re, n-1) * X.Im1,
        n * (Math.Pow(X.Re, n-1) * X.Im2 + (n-1) * Math.Pow(X.Re, n-2) * X.Im1 * X.Im1));
      }
   public static T0 operator +(T0 X, T0 Y)
      { return new T0(X.Re + Y.Re, X.Im1 + Y.Im1, X.Im2 + Y.Im2);}
   public static T0 operator -(T0 X, T0 Y)
      { return new T0(X.Re - Y.Re, X.Im1 - Y.Im1, X.Im2 - Y.Im2); }
   public static T0 operator -(T0 X) { return new T0(-X.Re, -X.Im1, -X.Im2);}
}

public struct T2
{  public Matrix<double> Re, Im1, Im2;
   public T2(double[,] M, double[,] M1, double[,] M2)
      { Re = Matrix<double>.Build.DenseOfArray(M);
        Im1 = Matrix<double>.Build.DenseOfArray(M1);
        Im2 = Matrix<double>.Build.DenseOfArray(M2);
      }
   public T2(Matrix<double> M, Matrix<double> M1, Matrix<double> M2)
```

```
      { Re = M; Im1 = M1; Im2 = M2; }
   public static T2 operator *(double x, T2 X)
      { return new T2(x * X.Re, x * X.Im1, x * X.Im2);}
   public static T2 operator *(T2 X, T2 Y)
      { return new T2(X.Re * Y.Re, X.Re * Y.Im1 + X.Im1 * Y.Re, X.Re * Y.Im2 + 2.0 * X.Im1 *
               Y.Im1 + X.Im2 * Y.Re);}
   public static T2 operator +(T2 X, T2 Y)
      { return new T2(X.Re + Y.Re, X.Im1 + Y.Im1, X.Im2 + Y.Im2); }
   public static T2 operator -(T2 X, T2 Y)
      { return new T2(X.Re - Y.Re, X.Im1 - Y.Im1, X.Im2 - Y.Im2); }
   public static T2 operator ^(T2 X, int n)
      { var Y = X;
          for (int i = 2; i <= n; i++) { Y = Y * X; }
          return Y;
      }
//====== Методы
   public static T0 TR(T2 A)
      { return new T0(A.Re.Trace(), A.Im1.Trace(), A.Im2.Trace());}
   public static (T0 I1, T0 I2, T0 I3) Invariants(T2 A)
      { T2 A2 = A * A,   A3 = A * A2;
        T0 I1 = TR(A), I2 = (I1*I1 - TR(A2))/2.0,
           I3 = (I1 * I1 * I1)/6.0 - I1 * TR(A2)/2.0 +  TR(A3)/3.0;
         return (I1, I2, I3);
      }
   public static double[,] getΛ((T0 I1, T0 I2, T0 I3) I)
   {(double, double, double) roots;
    roots = Cubic.RealRoots(-I.I3.Re, I.I2.Re, -I.I1.Re);
    double λ, λ1, λ2, γ;
    double[] Rt = { roots.Item1, roots.Item2, roots.Item3 };
    double[,] R = new double[3, 3];
    for (int i = 0; i <= 2; i++)
    { λ = Rt[i];
      γ = 3 *  λ * λ - 2 * I.I1.Re * λ + I.I2.Re;
      λ1 = ( I.I1.Im1 * λ * λ - I.I2.Im1 * λ + I.I3.Im1) / γ;
      λ2 = (I.I1.Im2 * λ * λ - I.I2.Im2 * λ - 6 * λ1 * λ1 * λ +
            4 * I.I1.Im1 * λ * λ1 - 2 * I.I2.Im1 * λ1 +
            2 * I.I1.Re * λ1 * λ1 + I.I3.Im2) / γ;
      R[i, 0] = λ; R[i, 1] = λ1; R[i, 2] = λ2;
     }
     return R;
   }
   public static Vector<double>[,] GetV(T2 A, double[,] Λ)
   {Svd<double> svd;
    Vector<double> v, v1, v2, Vb;
    Matrix<double> A0 = A.Re, A1 = A.Im1, A2 = A.Im2;
    Matrix<double> B0, B1, B2;
    Vector<double>[,] M = new Vector<double>[3, 3];
    var I = Matrix<double>.Build.DenseIdentity(3);
    var λ = Matrix<double>.Build.DenseOfArray(Λ);
    for (int i = 0; i <= 2; i++)
    {   var λi = λ.Row(i);
        B0 = A.Re - I * λi[0]; B1 = A.Im1 - I * λi[1]; B2 = A.Im2 - I * λi[2];
        svd = B0.Svd(true);
        v = svd.VT.Row(svd.VT.RowCount - 1);  //-- решение уравнения B*v=0
        Vb = B1 * v;
        v1 = SolveStable(B0, -Vb);            //-- решение уравнения B*v=-Vb
        Vb = B2 * v + 2 * (B1 * v1);
        v2 = SolveStable(B0, -Vb);            //-- решение уравнения B*v=-Vb
        M[i, 0] = v; M[i, 1] = v1; M[i, 2] = v2;
     }
```

```
      return M;
    }
    private static Vector<double> SolveStable(Matrix<double> A, Vector<double> v,
    double tol = 1e-12)
    { var svd = A.Svd(true); var U = svd.U; var S = svd.S; var VT = svd.VT;  int n = S.Count;
      double[] Sinv = new double[n];
      for (int i = 0; i < n; i++) Sinv[i] = (S[i] > tol) ? 1.0 / S[i] : 0.0;
      var Splus = DiagonalMatrix.OfDiagonal(n, n, Sinv);
      var x = VT.TransposeThisAndMultiply(Splus.Multiply(U.TransposeThisAndMultiply(v)));
      return x;
    }
}
//=======Пример расчета=======
double[,] t1 = { { 5, 1, 4 },    { 3, 3, 2 },  { 6, 2, 10 } };
double[,] t2 = { { -5, -1, -4 }, { 3, 3, 2 },  { 6, 2, 10 } };
double[,] t3 = { { -5, 1, 4 },   { -3, 3, 2 }, { -6, 2, 10 } };
var T = new T2(t1, t2, t3);  //создание тензора
var I = T2.Invariants(A);    //вычисление инвариантов
var Λ = T2.getΛ(I);          //вычисление собственных чисел
var V = T2.GetV( A, Λ);      //вычисление собственных векторов
```

Литература

1. *Олифер В.И.* Усеченные гипер-дуальные числа в автоматическом дифференцировании. – URL: http://viosolutions.amerihomesrealty.com/pdf/усеченные_гипер-дуальные_числа_в_автоматическом_дифференцировании.pdf, – 2020.

2. *Олифер В.И.* Характеристические многочлены матриц над гипер-дуальной алгеброй с соотношениями $\varepsilon^2 = 2\omega, \varepsilon\omega = \omega^2 = 0$. – URL: https://viosolutions.amerihomesrealty.com/pdf/Характеристические многочлены матриц.pdf, – 2026.

3. *Олифер В.И.* Вычисление собственных векторов гипер-дуальных матриц. – URL: http://viosolutions.amerihomesrealty.com/pdf/Вычисление собственных гипер-дульных векторов.pdf, – 2026.

4. Math.NET Numerics – https://www.nuget.org/packages/MathNet.Numerics

Абстракт

В статье рассматривается метод определения собственных значений и собственных векторов, соответствующих спектральному уравнению над усечённой гипер-дуальной алгеброй, определяемой отношениями $\varepsilon^2 = 2\omega, \varepsilon\omega = \omega^2 = 0$. *Приведен код программного обеспечения и рассмотрены численные примеры, полученные на его основе.*

This article examines a method for determining eigenvalues and eigenvectors corresponding to a spectral equation over a truncated hyper-dual algebra defined by the relations $\varepsilon^2 = 2\omega, \varepsilon\omega = \omega^2 = 0$. *Software code is provided, along with numerical examples generated using it.*

Ключевые слова: *спектральное уравнение, собственные значения, собственные векторы, гипер-дуальные числа и матрицы, spectral equation, eigenvalues, eigenvectors, hyper-dual numbers and matrices.*

20 май 2026