Processing of the $\mathbb{H}$ -holomorphic Functions

Michael Parfenov[1]

26.10.2020

Bashkortostan Branch of Russian Academy of Engineering, Ufa, Russia

Abstract To automate cumbersome, error-prone and tedious manual procedures of calculations with quaternionic holomorphic ($\mathbb{H}$ -holomorphic) functions we have developed and present here a special programmes pack in the Wolfram Mathematica® programming language. By using this pack a lot of examples of $\mathbb{H}$ -holomorphic functions is processed. They give conclusive evidence that the so-called essentially adequate theory of quaternionic holomorphy is true. All considered $\mathbb{H}$ -holomorphic functions are built from complex holomorphic ones in accordance with the general constructing rule defined earlier. At that combinations of $\mathbb{H}$ -holomorphic functions are built from the simplest basis (irreducible) $\mathbb{H}$ -holomorphic functions by using the usual rules of quaternionic algebraic operations.

Keywords: *quaternionic holomorphic functions, quaternionic Cauchy-Riemann's equations, quaternionic potential, processing of quaternionic holomorphic functions, Mathematica programming pack*

1. Introduction

This article is performed in addition to article [1]. As in [1] we denote an independent quaternionic variable by

$$p = x + yi + zj + uk = a + b \cdot j \in \mathbb{H},$$

where x, y, z, u are real values, the variables

$$a = x + yi,\ b = z + ui$$

are complex constituents of the quaternion representation in the so-called Cayley–Dickson doubling form $p = a + b \cdot j$ [1, 2], and i, j, k are quaternionic basis vectors in quaternion space $\mathbb{H}$.

The quaternionic functions are, respectively, denoted by

$$\psi(p) = \psi_1(x, y, z, u) + \psi_2(x, y, z, u)i \quad (1)$$
$$+\psi_3(x, y, z, u)j + \psi_4(x, y, z, u)k \in \mathbb{H},$$

where $\psi_1(x, y, z, u)$, $\psi_2(x, y, z, u)$, $\psi_3(x, y, z, u)$, $\psi_4(x, y, z, u)$ are real-valued functions of real variables x, y, z, u in the so-called $\mathbb{R}^4$ –representation [1, 2].

In the Cayley–Dickson doubling form (so-called $\mathbb{C}^2$ – representation [1, 2]) we have

$$\psi(p) = \phi_1(a, b, \overline{a}, \overline{b}) + \phi_2(a, b, \overline{a}, \overline{b}) \cdot j,$$

where

$$\phi_1(a, b, \overline{a}, \overline{b}) = \psi_1 + \psi_2 i,$$
$$\phi_2(a, b, \overline{a}, \overline{b}) = \psi_3 + \psi_4 i$$

are complex constituents.

For simplicity, we use the following notation:

$$\phi_1(a, b, \overline{a}, \overline{b},) = \phi_1,\ \phi_2(a, b, \overline{a}, \overline{b},) = \phi_2,$$
$$\psi_1(x, y, z, u) = \psi_1,\ \psi_2(x, y, z, u) = \psi_2,$$
$$\psi_3(x, y, z, u) = \psi_3,\ \psi_4(x, y, z, u) = \psi_4.$$

Further in developed programmes we use for variables $a, b, \overline{a}, \overline{b}$ the designations a, b, ac, bc, respectively. The functions ϕ_1, ϕ_2, $\overline{\phi_1}$, $\overline{\phi_2}$ are denoted in programmes by f1, f2, f1c, f2c, respectively.

The aim of this article is to present a special pack of the computing programmes written in Wolfram Mathematica® programming language , which allows us to automate processing of the $\mathbb{H}$ -holomorphic functions to avoid a cumbersome, error-prone and tedious manual procedure of calculations with $\mathbb{H}$ -holomorphic functions. At the same time we want, using these programmes, to show that the developed essentially adequate concept of holomorphic functions [1] *in quaternionic analysis is wholly true.*

2. Processing Principles

In accordance with the essentially adequate concept of quaternionic holomorphic ($\mathbb{H}$-holomorphic) functions [1] the $\mathbb{H}-$holomorphic functions (beeing created from the complex holomorphic counterparts by replacing an independent complex variable as a whole by a quaternionic one in expressions for complex holomorphic functions - rule of constructing 2.1 in [1]), must satisfy the so-called essentially adequate quaternionic generalization of Cauchy-Riemann's equations:

$$\begin{cases} 1)\ (\partial_a \phi_1| = (\partial_{\overline{b}} \overline{\phi}_2|, & 2)\ (\partial_a \phi_2| = -(\partial_{\overline{b}} \overline{\phi}_1|, \\ 3)\ (\partial_a \phi_1| = (\partial_b \phi_2|, & 4)\ (\partial_{\overline{a}} \phi_2| = -(\partial_{\overline{b}} \phi_1|. \end{cases} \quad (2)$$

Here ∂_i, $i = a, \overline{a}, b, \overline{b}$, denotes the partial derivative with respect to i. The overbars designate the complex (also quaternionic if needed) conjugation. The brackets $(..|$ with the closing vertical bar indicate that the transition $a = \overline{a} = x$ has been already performed in expressions enclosed in brackets.

Thus, $\mathbb{H}$ -holomorphy conditions (2) are defined so that during the check of the quaternionic holomorphy of any quaternionic function we have to do the transition $a = \overline{a} =$

[1] e-mail: parfenov.48@bk.ru , mwparfenov@gmail.com

x in already calculated expressions for the partial derivatives of the functions $\phi_1$and ϕ_2.

However, this doesn't mean that we deal with triplets in general, since the transition $a = \overline{a} = x$ (or $y = 0$) cannot be initially done for quaternionic variables and functions [1, 2]. Any quaternionic function remains the same 4-dimensional quaternionic function regardless of whether we check its holomorphy or not. This transition is needed to check the $\mathbb{H}$-holomorphy of any quaternionic function. It is also used when solving the 3D physical tasks.

Program 4.2 "$\mathbb{H}$-holomorphy testing" checks whether equations (2) are satisfied for the quaternionic function $\psi(p) = \phi_1 + \phi_2 \cdot j = f1 + f2 \cdot j$ in question, i.e. whether this function is $\mathbb{H}$ -holomorphic.

According to the developed theory of $\mathbb{H}$ -holomorphic functions [1, 2], the quaternionic multiplication of the $\mathbb{H}$ -holomorphic functions behaves as commutative. This means that two $\mathbb{H}$ -holomorphic functions $f_H(p)$ and $g_H(p)$ satisfy the equality $f_H \cdot g_H = g_H \cdot f_H$, where the dot "$\cdot$" denotes the operation of quaternionic multiplication.

We recall the rule of quaternionic multiplication in the so-called Cayley–Dickson doubling form [3, 1]:

$$p \cdot q = (a_1 + b_1 \cdot j) \cdot (a_2 + b_2 \cdot j) \quad (3)$$
$$= \left(a_1 a_2 - b_1 \overline{b_2}\right) + (a_1 b_2 + \overline{a_2} b_1) \cdot j,$$

where quaternions p and q are $p = a_1 + b_1 \cdot j$, $q = a_2 + b_2 \cdot j$; components $a_1 = x_1 + y_1 i$, $b_1 = z_1 + u_1 i$, $a_2 = x_2 + y_2 i$, $b_2 = z_2 + u_2 i$ are complex variables, and x_1, y_1, z_1, u_1, and x_2, y_2, z_2, u_2 are real variables.

Considering two $\mathbb{H}$ -holomorphic functions: $f_H(p) = inia1 + inia2 \cdot j$ and $g_H(p) = inib1 + inib2 \cdot j$, where the notation inia1, inia2, inib1, inib2 is introduced to be used in Program 4.1, we in accordance with (3) have the rule of quaternionic multiplication for $\mathbb{H}$ -holomorphic functions as follows:

$$f_H(p) \cdot g_H(p) = (inia1 + inia2 \cdot j) \cdot (inib1 + inib2 \cdot j)$$
$$= \left(inia1\ inib1 - inia2\ \overline{inib2}\ \right)$$
$$+\left(inia1\ inib2 + \overline{inib1}\ inia2\right) \cdot j.$$

Interchanging the order of multiplication and using (3), we also obtain the following expression for quaternionic multiplying $g_H(p)$ by $f_H(p)$:

$$g_H(p) \cdot f_H(p) = (inib1 + inib2 \cdot j) \cdot (inia1 + inia2 \cdot j)$$
$$= (\ inib1\ inia1 - \ inib2\ \overline{inia2})$$
$$+(inib1\ inia2 + \ \overline{inia1}\ inib2) \cdot j.$$

The commutativity of the quaternionic product $f_H(p) \cdot g_H(p)$ takes place if the following equalities are satisfied:

$$inia1\ inib1 - inia2\ \overline{inib2} \quad (4)$$
$$= inib1\ inia1 - \ inib2\ \overline{inia2}\ ,$$
$$inia1\ inib2 + \ \overline{inib1}\ inia2 \quad (5)$$
$$= inib1\ inia2 + \ \overline{inia1}\ inib2.$$

Program 4.1 "Commutativity Testing" checks whether equalities (4) and (5) are satisfied for the quaternionic functions $f_H(p)$ and $g_H(p)$, i.e. whether the quaternionic product $f_H(p) \cdot g_H(p)$ behaves as commutative, if we deal with multiplication of $\mathbb{H}$ -holomorphic functions. This program tests also whether the right quotient of two quaternionic functions in question equals the left one, if we deal with division of one $\mathbb{H}$ -holomorphic function by the other. At the output of this program we get also the expressions for components f1 and f2 of the quaternionic product $f_H(p) \cdot g_H(p) = f1 + f2 \cdot j$, which are input data for Programes 4.2, 4.3, 4.4 and 4.5.

Program 4.3 "Calculating quaternionic expressions" computes all needed expressions for $\mathbb{H}$ -holomorphic functions (quaternionic potentials) in $\mathbb{R}^4$ –representation (see (1)), which can be used in different applications, for example, in calculations of 3D potential fields. These expressions are the following [1,2]:

1) The components $\psi_1, \psi_2, \psi_3, \psi_4$ of $\mathbb{H}$ -holomorphic functions (quaternionic potentials) $\psi_H(p)$:

$$\psi_H(p) = \psi_1 + \psi_2 i + \psi_3 j + \psi_4 k,$$

2) the quaternionic potentials after transition to 3D space (after $y = 0$):

$$(\psi_H(p)| = \varphi_1(x, z, u) + \varphi_3(x, z, u) j + \varphi_4(x, z, u) k,$$

3) the first quaternionic derivative $\psi'_H(p)$:

$$\psi'_H(p) = \psi'_1 + \psi'_2 i + \psi'_3 j + \psi'_4 k,$$

where $\psi'_1, \psi'_2, \psi'_3, \psi'_4$ denote the components of the first quaternionic derivative in $\mathbb{R}^4$ –representation;

4) the quaternionic vector fields:

$$F(p) = f_1 + f_2 i + f_3 j + f_4 k,$$

where

$$f_1 = \psi'_1, f_2 = -\psi'_2, f_3 = -\psi'_3, f_4 = -\psi'_4,$$

5) the 3D vector potential fields (after y = 0):

$$(F(p)| = (f_1| + (f_2|i + (f_3|j + (f_4|k,$$

where

$$(f_1| = (\psi'_1|, \ \ (f_2| = -(\psi'_2| = 0,$$
$$(f_3| = -(\psi'_3|, \ (f_4| = -(\psi'_4|,$$

6) the vortex dencities of 3D vector fields:

$$curl\ (F(p)| = \left(\frac{\partial(f_4|}{\partial z} - \frac{\partial(f_3|}{\partial u}\right)$$
$$+\left(\frac{\partial(f_1|}{\partial u} - \frac{\partial(f_4|}{\partial x}\right) j + \left(\frac{\partial(f_3|}{\partial x} - \frac{\partial(f_1|}{\partial z}\right) k,$$

7) the divergences of 3D vector fields:

$$div\ (F(p)| = \frac{\partial(f_1|}{\partial x} + \frac{\partial(f_3|}{\partial z} + \frac{\partial(f_4|}{\partial u}.$$

Program 4.3 can be also used if we want to calculate analogous expressions for non- $\mathbb{H}$ -holomorphic functions, represented in the form (1). In these cases Program 4.2 indicates that the function in question is non- $\mathbb{H}$ -holomorphic.

When calculating full quaternionic derivatives (uniting the left and right ones) of higher orders we use in accordance with theory [1, 2] the following general expression in the Cayley–Dickson doubling form ($\mathbb{C}^2$ – representation):

$$\psi_H^{(k)}(p) = \phi_1^{(k)} + \phi_2^{(k)} \cdot j, \quad (6)$$

where a k'th derivative of $\psi_H(p)$, $k = 1,2,3, \ldots$, is denoted by $\psi_H^{(k)}(p)$ and the complex constituents $\phi_1^{(k)}$ and $\phi_2^{(k)}$ are expressed by

$$\phi_1^{(k)} = \partial_a \phi_1^{(k-1)} + \partial_{\overline{a}} \phi_1^{(k-1)} = \partial_{a,\overline{a}} \phi_1^{(k-1)},$$
$$\phi_2^{(k)} = \partial_a \phi_2^{(k-1)} + \partial_{\overline{a}} \phi_2^{(k-1)} = \partial_{a,\overline{a}} \phi_2^{(k-1)}.$$

Here $\phi_1^{(k-1)}$ and $\phi_2^{(k-1)}$ are constituents of the $(k-1)'$ th derivative of a function $\psi_H(p)$, represented in the Cayley–Dickson doubling form by $\psi_H(p)^{(k-1)} = \phi_1^{(k-1)} + \phi_2^{(k-1)} \cdot j$, $\phi_1^{(0)} = \phi_1$, $\phi_2^{(0)} = \phi_2$.

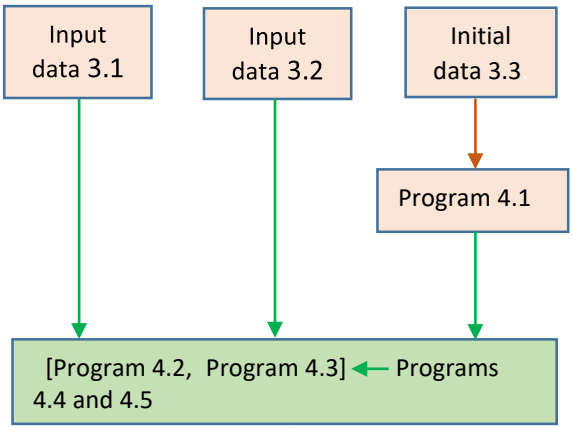
Program 4.4 calculates constituents $\phi_1^{(k)}$ and $\phi_2^{(k)}$ of the k'th derivative of $\psi_H(p)$, $k = 1,2,3, \ldots$. Since only the functions f1 and f2 can be used as input data for Programmes 4.2 and 4.3., to test $\mathbb{H}$ -holomorphy of derivatives of higher orders and calculate quaternionic expressions for them we need to mark each time the calculated components $\phi_1^{(k)}$ and $\phi_2^{(k)}$ as the functions f1 and f2, respectively.

At that we can test $\mathbb{H}$ -holomorphy of higher derivatives, launching Program 4.2 after launching Program 4.4, or calculate needed quaternionic expressions for higher

derivatives, launching Program 4.3 after launching Program 4.4.

Calculations of the 1'th, 2'th, 3'th, etc derivatives are carried out by consecutive launches of Program 4.4: the 1'th launch of Program 4.4 gives the functions f1 and f2 for the 1'th derivative, the 2'th launch of this program gives the functions f1 and f2 for the 2'th derivative, etc.

Scheme 1 illustrates how to use the presented input data and programmes.



Scheme 1.The right sequences of data and programmes launching

Only the represented sequences of launching are correct.

The green arrows show the "transmission" of functions f1 and f2 from input data 3.1, 3.2 and initial data 3.3 to Programmes 4.2, 4.3 and 4.4 as well as from Program 4.4 and 4.5 to Programmes 4.2 and 4.3.

All the below programming codes can be copied cell by cell, each only into a new separate input cell of a Notebook file .nb. All programming codes, which may be copied into file .nb., are represented below by „inherited" typeface.

To launch each programing code we need to click on it's cell in the file .nb of Wolfram Mathematica® computing system , and then holding down the Left-Shift key while press the Enter key. When saying about launching any programming code in any cell we will always mean this sequence of key-board and mouse actions.

In the Wolfram Mathematica computing system these actions are referred to as a "cell evaluation", however since a sequence of actions is here important it seems to us more convenient the wording a "cell launching".

In order to have a possibility to copy all programming codes directly from this article into empty input cells of the opened Notebook blank .nb of the computing system Wolfram Mathematica® [4] and then to launch them immediately in file .nb, we use the following designations:

1) psi(p), psi1, psi2, psi3, psi4 instead of $\psi(p)$ ψ_1, ψ_2, ψ_3, ψ_4, respectively; 2) phi1, phi2, phi3, phi4 instead of φ_1, φ_2, φ_3, φ_4, respectively; 3) d1psi, d1f1, d1f2 and d1psi1, d1psi2, d1psi3, d1psi4 instead of $\psi(p)'$, ϕ_1', ϕ_2' and ψ_1', ψ_2', ψ_3', ψ_4', respectively;
4) fn1, fn2, fn3, fn4 instead of the components f_1, f_2, f_3, f_4 of the vector field $F(p)$ and $\frac{\mathrm{dfn2}}{\mathrm{dy}}$ instead of $\frac{\partial f_2}{\partial y}$.

The functions without semicolon at the end of them must be each in a separate line after copying. For example, the fragment of the programming code from sub-section 3.3.1-2) (The second derivative of the function $\boldsymbol{e^{p^2}}$) is the following:

```
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

After copying the whole programming code from sub-section 3.3.1 - the function 2), into a separate cell of an opened Notebook file .nb, this fragment may become the following:

```
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]  inia2[a,ac,b,bc] Print["Constituents inib1, inib2 of
the quaternionic function ",quatfunctiontested,"
=(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc] inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

In this case we need, using the Enter key, to shift the function "inia2[a,ac,b,bc]" from the third line down, after that to shift the "Print["Constituents inib1, inib2…."]" down, and then, finally, to shift the function "inib2[a,ac,b,bc]" (before "mark=True;") down. Now each of the functions without semicolon at the end: inia1[a,ac,b,bc], inia2[a,ac,b,bc], inib1[a,ac,b,bc], inib2[a,ac,b,bc] is in the separate line.

The input data for Programmes 4.2, 4.3, 4.4 and 4.5 are always the functions f1, f2 (also their conjugate f1c and f2c). To illustrate the processing of $\mathbb{H}$-holomorphic (and non-$\mathbb{H}$-holomorphic) functions in accordance with Scheme 1 we consider three simplest examples.

Example 2.1. The function $\boldsymbol{p^2}$ from sub-section 3.1-2). After launching the copied cell, containing this function, we obtain in file .nb as a result the following functions f1, f2:

Constituents f1, f2 of the quaternionic function p^2:

$$a^2 - b\,bc\,,$$
$$(a + ac)\,b$$

After launching the copied cell, containing Program 4.2, we obtain in file .nb the following result:

Testing of eq1: True// of eq2: True// of eq3: True// of eq4: True
Result of full testing: The quaternionic function p^2 is H-holomorphic

After launching the copied cell, containing Program 4.3, we obtain in file .nb the following results:

RESULT 1: Quaternionic function $\mathrm{psi(p)} = \mathrm{p^2} = \mathrm{psi1} + \mathrm{psi2} * \mathrm{i} + \mathrm{psi3} * \mathrm{j} + \mathrm{psi4} * \mathrm{k}$, where $\mathrm{psi1} = -\mathrm{u^2} + \mathrm{x^2} - \mathrm{y^2} - \mathrm{z^2}$; // $\mathrm{psi2} = 2\mathrm{xy}$; // $\mathrm{psi3} = 2\mathrm{xz}$; // $\mathrm{psi4} = 2\mathrm{ux}$
RESULT 2 (correct only for H-holomorphic functions): Polar form of the function p^2 is m[cos(teta)+rsin(teta)], where $\mathrm{m}=\mathrm{u^2}+\mathrm{x^2}+\mathrm{y^2}+\mathrm{z^2}$; $\mathrm{teta}=\mathrm{Arctan}[-\frac{2\mathrm{x}\sqrt{\mathrm{u^2}+\mathrm{y^2}+\mathrm{z^2}}}{\mathrm{u^2}-\mathrm{x^2}+\mathrm{y^2}+\mathrm{z^2}}]$
RESULT 3: Quaternionic function after transition to 3D ($y = 0$): $(\mathrm{psi(p)}| = \mathrm{phi1} + \mathrm{phi2} * \mathrm{i} + \mathrm{phi3} * \mathrm{j} + \mathrm{phi4} * \mathrm{k}$, where $\mathrm{phi1} = -\mathrm{u^2} + \mathrm{x^2} - \mathrm{z^2}$; // $\mathrm{phi2} = 0$; // $\mathrm{phi3} = 2\mathrm{xz}$; // $\mathrm{phi4} = 2\mathrm{ux}$
RESULT 4: Quaternionic derivative: $\mathrm{d1psi} = \mathrm{d1f1} + \mathrm{d1f2} * \mathrm{j} = \mathrm{d1psi1} + \mathrm{d1psi2} * \mathrm{i} + \mathrm{d1psi3} * \mathrm{j} + \mathrm{d1psi4} * \mathrm{k}$, where $\mathrm{d1psi1} = 2\mathrm{x}$;// $\mathrm{d1psi2} = 2\mathrm{y}$; // $\mathrm{d1psi3} = 2\mathrm{z}$; // $\mathrm{d1psi4} = 2\mathrm{u}$
RESULT 5: Quaternionic field vector: $\mathrm{F(p)} = \mathrm{fn1} + \mathrm{fn2} * \mathrm{i} + \mathrm{fn3} * \mathrm{j} + \mathrm{fn4} * \mathrm{k}$, where $\mathrm{fn1} = \mathrm{d1psi1} = 2\mathrm{x}$; // $\mathrm{fn2} = -\mathrm{d1psi2} = -2\mathrm{y}$; // $\mathrm{fn3} = -\mathrm{d1psi3} = -2\mathrm{z}$; // $\mathrm{fn4} = -\mathrm{d1psi4} = -2\mathrm{u}$
RESULT 6: 3D field vector (after $\mathrm{y} = 0$): $(\mathrm{F(p)}| = (\mathrm{fn1}| + (\mathrm{fn2}| * \mathrm{i} + (\mathrm{fn3}| * \mathrm{j} + (\mathrm{fn4}| * \mathrm{k}$, where $(\mathrm{fn1}| = (\mathrm{d1psi1}| = 2\mathrm{x}$; // $(\mathrm{fn2}| = -(\mathrm{d1psi2}| = 0$; $(\mathrm{fn3}| = -(\mathrm{d1psi3}| = -2\mathrm{z}$; // $(\mathrm{fn4}| = -(\mathrm{d1psi4}| = -2\mathrm{u}$

RESULT 7: For quaternionic function psi(p) = p^2 the vortex density curl (F| = 0. The field (F(p)| is potential.
RESULT 8: For quaternionic function psi(p) = p^2 the divergence div (F| is equal to -2 and equal to $(\frac{\mathrm{dfn2}}{\mathrm{dy}}| = -2$. The derivative $(\frac{\mathrm{dfn2}}{\mathrm{dy}}|$ represents the dencity of sources and sinks in 3D space.

The 1'th launch of Program 4.4 gives the functions f1 and f2 for the 1'th derivative of p^2. Further, launching Program 4.2, we get the following

Testing of eq1: True// of eq2: True// of eq3: True// of eq4: True
Result of full testing: the 1'th derivative of p^2 is H-holomorphic

Analogous results we get for the 2'th, 3'th derivative, etc.
Example 2.2. The quotient $\boldsymbol{e^p/p}$ from sub-section 3.3.2-1). After launching the copied cell, containing this function, we obtain in file .nb as a result the following functions inia1, inia2 and inib1, inib2:

Constituents inia1, inia2 of the quaternionic function e^p/p=e^p·(1/p) =(inia1+inia2·j)·(inib1+inib2·j):

$$e^{\frac{a+ac}{2}}Cos\left[\frac{1}{2}\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}\right] + \frac{(a-ac)e^{\frac{a+ac}{2}}Sin\left[\frac{1}{2}\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}\right]}{\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}},$$

$$\frac{2be^{\frac{a+ac}{2}}Sin\left[\frac{1}{2}\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}\right]}{\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}}$$

Constituents inib1, inib2 of the quaternionic function e^p/p=e^p(1/p) = (inia1+inia2·j)·(inib1+inib2·j) :

$$\frac{ac}{a\,ac+b\,bc},$$

$$-\frac{b}{a\,ac+b\,bc}.$$

In accordance with Scheme 1 we further launch Program 4.1. Omitting here the obtained results of computing of the functions f1, f2 for this quotient (they will be further used in Programmes 4.2, 4.3, 4.4 and 4.5), we point here only the result of commutativity testing:

Result of testing: Quaternionic multiplication in the case of e^p/p= e^p·(1/p) behaves as commutative.
The right quaternionic quotient e^p/p= e^p·(1/p) is equal to the left one.

Further, after launching the copied cell, containing Program 4.2, we obtain in file .nb the following result:

Testing of eq1: True// of eq2: True// of eq3: True// of eq4: True
Result of full testing: The quaternionic function e^p/p=e^p·(1/p) is H-holomorphic.

Launching further the copied cell, containing Program 4.3, we get in $\mathbb{R}^4$ –representation the results for this quaternionic function analogously to above noted. We dwell here only on results 7 and 8:

RESULT 7: For quaternionic function $psi(p)$ = e^p/p=e^p·(1/p) the vortex density $curl\ (F| = 0$. The field $(F(p)|$ is potential.
RESULT 8: For quaternionic function $psi(p)$ = e^p/p=e^p·(1/p) the divergence $div\ (F|$ is equal to

$$\frac{e^x\left(\sqrt{u^2+z^2}\,(u^2+(-2+x)x+z^2)\mathrm{Cos}\sqrt{u^2+z^2}-\left((-1+x)x^2+u^2(1+x)+(1+x)z^2\right)\mathrm{Sin}\sqrt{u^2+z^2}\right)}{\sqrt{u^2+z^2}\,(u^2+x^2+z^2)^2}$$

and equal to $(\frac{dfn2}{dy}| =$

$$\frac{e^x\left(\sqrt{u^2+z^2}\,(u^2+(-2+x)x+z^2)\mathrm{Cos}\sqrt{u^2+z^2}-\left((-1+x)x^2+u^2(1+x)+(1+x)z^2\right)\mathrm{Sin}\sqrt{u^2+z^2}\right)}{\sqrt{u^2+z^2}\,(u^2+x^2+z^2)^2}.$$

The derivative $(\frac{dfn2}{dy}|$ represents the density of sources and sinks in 3D space.

The 1'th launch of Program 4.4 gives the functions f1 and f2 for the 1'th derivative of e^p/p= e^p·(1/p). Further, launching Program 4.2, we get the following:

Testing of eq1: True// of eq2: True// of eq3: True// of eq4: True
Result of full testing: the 1'th derivative of e^p/p= e^p·(1/p) is H-holomorphic.

Analogous results we get for the 2'th, 3'th derivative, etc.
Example 2.3. We construct the right quaternionic quotient $\boldsymbol{p2p3/e^p}$ of the non - $\mathbb{H}$ - holomorphic function $\boldsymbol{p2p3}$ by the $\mathbb{H}$ - holomorphic function $\boldsymbol{e^p}$. The function $\boldsymbol{p2p3}$ is represented by the function f1 from the $\mathbb{H}$ - holomorphic function $\boldsymbol{p2}$ and the function f2 from the $\mathbb{H}$ - holomorphic function $\boldsymbol{p3}$ (see functions 2) and 3) in sub-section 3.1). The data input for such a non- $\mathbb{H}$ - holomorphic function $\boldsymbol{p2p3/e^p}$ is the following:

```
(*The right quotient of p2p3 by e^p: p2p3 ·(1/ e^p)= (p2p3· e^p _
conjugate) /( e^p · e^p _ conjugate) =(inia1+inia2·j) ·
(inib1+inib2·j)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="p2p3/e^p=p2p3•(1/e^p)";
inia1[a_,ac_,b_,bc_]:=a^2-b*bc;
inia1c[a_,ac_,b_,bc_]:=ac^2-b*bc;
inia2[a_,ac_,b_,bc_]:=(a^2+a*ac+ac^2)*b-b^2*bc;
inia2c[a_,ac_,b_,bc_]:=(ac^2+a*ac+a^2)*bc-bc^2*b;
vorf1i[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(a-ac)*Sin[v])/v;
vorf1ic[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(ac-a)*Sin[v])/v;
vorf2i[a_,ac_,b_,bc_]:=(2*beta*b*Sin[v])/v;
vorf2ic[a_,ac_,b_,bc_]:=(2*beta*bc*Sin[v])/v;
vorabsexp2:=(vorf1i[a,ac,b,bc]*vorf1ic[a,ac,b,bc]+vorf2i[a,ac,b,bc]*
vorf2ic[a,ac,b,bc]);
f1i[a_,ac_,b_,bc_]:=Simplify[vorf1i[a,ac,b,bc]];
f1ic[a_,ac_,b_,bc_]:=Simplify[vorf1ic[a,ac,b,bc]];
f2i[a_,ac_,b_,bc_]:=Simplify[vorf2i[a,ac,b,bc]];
f2ic[a_,ac_,b_,bc_]:=Simplify[vorf2ic[a,ac,b,bc]];
absexp2=Simplify[vorabsexp2];
inib1[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/absexp2;
inib1c[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/absexp2;
inib2[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/absexp2;
inib2c[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/absexp2;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2•j)•(inib1+inib2•j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2•j)•(inib1+inib2•j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

After launching the copied cell, containing these initial data, we obtain in file .nb the following functions inia1, inia2 and inib1, inib2:

Constituents inia1, inia2 of the quaternionic function p2p3/e^p= p2p3·(1/e^p) = (inia1+inia2·j)·(inib1+inib2·j):

$$a^2 - b\,bc,$$

$$(a^2 + a\,ac + ac^2)\,b - b^2\,bc$$

Constituents inib1, inib2 of the quaternionic function p2p3/e^p= p2p3·(1/e^p)=(inia1+inia2·j)·(inib1+inib2·j):

$$e^{-a-ac+\frac{a+ac}{2}}\left(\mathrm{Cos}\left[\frac{\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}}{2}\right] + \frac{(-a+ac)\mathrm{Sin}\left[\frac{\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}}{2}\right]}{\sqrt{-a^2+2\,a\,ac-ac^2+4\,b\,bc}}\right),$$

$$\frac{2\,b\,e^{-a-ac+\frac{a+ac}{2}}\mathrm{Sin}[\frac{\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}}{2}]}{\sqrt{-(a+ac)^2+4(a\,ac+b\,bc)}}.$$

Launching the copied cell, containing Program 4.1, and omitting results of computing functions f1 and f2, we point here only the result of commutativity testing:

Result of testing: Quaternionic multiplication in the case of p2p3/e^p=p2p3·(1/e^p) is non-commutative. The right quaternionic quotient in the case of p2p3/e^p=p2p3·(1/e^p) is not equal to the left one.

After launching the copied cell, containing Program 4.2, we obtain in file .nb the following:

Testing of eq1: False// of eq2: False// of eq3: False// of eq4: Fase
Result of full testing: The quaternionic function
p2p3/e^p=p2p3·(1/e^p) is not H-holomorphic.
The equation 1) is not satisfied
The equation 2) is not satisfied
The equation 3) is not satisfied
The equation 4) is not satisfied

Further we launch the copied cell, containing Program 4.3. Omitting the obtained results 1-5, we point here only the results 6 and 7:

RESULT 6: For quaternionic function psi(p) = p2p3/e^p = p2p3 ·(1/e^p) the vortex density curl$(F(p|! = 0$. The field $(F(p|$ is not potential.
RESULT 7: For quaternionic function psi(p) = p2p3/e^p = p2p3·(1/e^p) the divergence div$(F(p|$ is equal to

$$\frac{e^{-x}}{\sqrt{u^2+z^2}}\left(-2\sqrt{u^2+z^2}\,(-1+2u^2+(7-3x)x+2z^2)\mathrm{Cos}\sqrt{u^2+z^2}\right)$$
$$-\left((-2+x)x+u^2(-9+6x)+3(-3+2x)z^2)\mathrm{Sin}\sqrt{u^2+z^2}\right)$$

and not equal to

$$(\frac{dfn2}{dy}| = \frac{e^{-x}(2(-1+x)\sqrt{u^2+z^2}\,\mathrm{Cos}\sqrt{u^2+z^2}+(u^2-(-2+x)x+z^2)\mathrm{Sin}\sqrt{u^2+z^2})}{\sqrt{u^2+z^2}}.$$

The 1'th launch of Program 4.4 gives the functions f1 and f2 for the 1'th derivative of p2p3/e^p = p2p3 ·(1/e^p). Further, launching Program 4.2, we get the following

Testing of eq1: False// of eq2: False// of eq3: False// of eq4: Fase
Result of full testing: The quaternionic function: the first derivative of p2p3/e^p = p2p3 ·(1/e^p) is not H-holomorphic
The equation 1) is not satisfied
The equation 2) is not satisfied
The equation 3) is not satisfied
The equation 4) is not satisfied

Analogous results we get for the 2'th, 3'th derivative, etc.

The considered examples of quaternionic functions (just as all examples of $\mathbb{H}$ -holomorphic functions represented below) demonstrate the main consequences of so-called essentially adequate theory [1, 2] of quaternionic holomorphic functions: the quaternionic products of $\mathbb{H}$ -holomorphic functions behaves as commutative; the right quaternionic quotients of $\mathbb{H}$ - holomorphic functions equals the left ones; the vortex density of the potential fields, corresponding to $\mathbb{H}$ - holomorphic functions, equals zero; the divergence of such fields equals the density of sources and sinks in 3D space; and the full quaternionic derivatives of all orders of $\mathbb{H}$ - holomorphic functions are also $\mathbb{H}$ - holomorphic.

These examples show the difference between the class of $\mathbb{H}$ -holomorphic functions defined in [1] and arbitrarily defined quaternionic functions.

If a non-compliance with the noted programmes launcing sequences takes a place, then Programmes 4.1, 4.2, 4.3 and 4.4 generate error messages.

3. Data Examples for $\mathbb{H}$ -holomorphic Functions

At the beginning of each Mathematica® session on processing of $\mathbb{H}$-holomorphic functions we need to launch one time the following general standard functions:

```
beta=(1/2)*E^((a+ac)/2);
v=(1/2)*Sqrt[4*(a*ac+b*bc)-(a+ac)^2];
alpha=ArcCos[(a+ac)/(2*Sqrt[(a*ac+b*bc)])];
epl=E^v;
emin=E^(-v);
```

All the $\mathbb{H}$-holomorphic functions considered further are constructed from complex holomorphic counterparts by using constructing rule 2.1 defined in [1]. The examples of functions are selected arbitrarily. Processing of them gives in the end expressions for the functions f1 and f2, which are input data for main Programmes 4.2, 4.3, 4.4 and 4.5. We consider the functions in their domains of definition (see Theorem 1.3 in [1]), except at singularities, which can be easy specified if needed. The represented database on $\mathbb{H}$ -holomorphic functions and the processing methods will be replenished.

3.1 Examples of Irreducible $\mathbb{H}$ -holomorphic Functions

As irreducible $\mathbb{H}$-holomorphic functions we regard here the $\mathbb{H}$ -holomorphic functions that are not quaternionic products.

The above standard functions beta, v, alpha, epl, emin must be previously activated, otherwise processing of some functions can be incorrect.

It is possible only the following sequence of processing of the functions from this sub-section: launching the input data of a function in question from this sub-section, then launching Program 4.2 and/or, if needed, Programmes 4.3, 4.4 and 4.5.

1) The function $\boldsymbol{p}$

```
(*Funcion p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:= "p";
f1[a_,ac_,b_,bc_]:=a;
f2[a_,ac_,b_,bc_]:=b;
f1c[a_,ac_,b_,bc_]:=ac;
f2c[a_,ac_,b_,bc_]:=bc;
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

2) The function $\boldsymbol{p^2}$

```
(*Funcion p^2*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[p^2];
f1[a_,ac_,b_,bc_]:=a^2-b*bc
f2[a_,ac_,b_,bc_]:=(a+ac)*b
f1c[a_,ac_,b_,bc_]:=ac^2-b*bc
f2c[a_,ac_,b_,bc_]:=(a+ac)*bc
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

3) The function $\boldsymbol{p^3}$

```
(*Funcion p^3*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[p^3];
f1[a_,ac_,b_,bc_]:=a^3-(2*a+ac)*b*bc
f2[a_,ac_,b_,bc_]:=(a^2+a*ac+ac^2)*b-b^2*bc
f1c[a_,ac_,b_,bc_]:=ac^3-(2*ac+a)*b*bc
f2c[a_,ac_,b_,bc_]:=(ac^2+a*ac+a^2)*bc-bc^2*b
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

4) The function $\boldsymbol{\sin p}$

```
(*Funcion sinp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="sinp";
f1[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v)
f2[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*b)/(2*v)
f1c[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((ac-a)*(emin-
epl)*Cos[(a+ac)/2])/(4*v)
f2c[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v)
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

5) The function $\boldsymbol{\cos p}$

```
(*Funcion cosp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="cosp";
f1[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((a-ac)*(emin-
epl)*Sin[(a+ac)/2])/(4*v)
f2[a_,ac_,b_,bc_]:=((emin-epl)*b*Sin[(a+ac)/2])/(2*v)
f1c[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((ac-a)*(emin-
epl)*Sin[(a+ac)/2])/(4*v)
f2c[a_,ac_,b_,bc_]:=((emin-epl)*bc*Sin[(a+ac)/2])/(2*v)
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

6) The function $\boldsymbol{e^p}$

```
(*Funcion e^p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:=Evaluate[e^p];
f1[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(a-ac)*Sin[v])/v;
f2[a_,ac_,b_,bc_]:=(2*beta*b*Sin[v])/v;
f1c[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(ac-a)*Sin[v])/v;
f2c[a_,ac_,b_,bc_]:=(2*beta*bc*Sin[v])/v;
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

7) The function $\boldsymbol{e^{p^2}}$

```
(*Funcion e^p^2*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:=Evaluate[e^p^2];
v=(1/2)*Sqrt[4*(a*ac+b*bc)-(a+ac)^2];
beta1=E^((a^2+ac^2-2 b bc)/2);
teta=(a+ac)*v;
f1[a_,ac_,b_,bc_]:=beta1*(Cos[teta]+((a-ac)Sin[teta])/(2*v));
f2[a_,ac_,b_,bc_]:=(beta1*Sin[teta]*b)/v;
f1c[a_,ac_,b_,bc_]:=beta1*(Cos[teta]+((ac-a)Sin[teta])/(2*v));
f2c[a_,ac_,b_,bc_]:=(beta1*Sin[teta]*bc)/v;
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

8) The function $\sqrt{\boldsymbol{p}}$

```
(*Quaternionic generalization of principal branch of square root*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:= Evaluate[Sqrt[p]];
sr:=(Sqrt[((a+ac)/2+Sqrt[a*ac+b*bc])/2]);
f1[a_,ac_,b_,bc_]:=sr+(a-ac)/(4*sr);f1c[a_,ac_,b_,bc_]:=sr+(ac-
a)/(4*sr);
f2[a_,ac_,b_,bc_]:=b/(2*sr);
f2c[a_,ac_,b_,bc_]:=bc/(2*sr);
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

9) The function $\boldsymbol{r = \frac{a-ac}{2v} + \frac{b}{v} \cdot j}$

```
(*Quaternionic analog of complex imaginary unit i *)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="r";
f1[a_,ac_,b_,bc_]:=(a-ac)/(2*v)
f2[a_,ac_,b_,bc_]:=b/v
f2c[a_,ac_,b_,bc_]:=bc/v
f1c[a_,ac_,b_,bc_]:=(ac-a)/(2*v)
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

10) The function $\bar{r} = \frac{ac-a}{2v} - \frac{b}{v} \cdot j$

```
(*Quaternionic conjugate of the function r *)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="r_conjugate";
f1[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
f2[a_,ac_,b_,bc_]:=-b/v
f2c[a_,ac_,b_,bc_]:=-bc/v
f1c[a_,ac_,b_,bc_]:=(a-ac)/(2*v)
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

11) The function $\boldsymbol{Logp}$

```
(*Quaternionic generalization of principal branch of logarithmic
function*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="Logp";
f1[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
f1c[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-a)*alpha)/(2*v);
f2[a_,ac_,b_,bc_]:=(b*alpha)/v;
f2c[a_,ac_,b_,bc_]:=(bc*alpha)/v;
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

3.2 Examples of Reciprocal Functions

The above standard functions beta, v, alpha, epl, emin must be previously activated, otherwise processing of some functions can be incorrect.

In this sub-section we consider only the reciprocal $\mathbb{H}$-holomorphic functions. Although they are also irreducible functions, we write them into a separate sub-section. It is possible only the following sequence of processing of the functions from this sub-section: launching the input data of a function in question from this sub-section, then launching Program 4.2, and/or, if needed, Programmes 4.3, 4.4 and 4.5.

1) The function $\frac{1}{p}$

```
(*Function 1/p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[1/p];
f1[a_,ac_,b_,bc_]:=ac/(a*ac+b*bc)
f2[a_,ac_,b_,bc_]:=-b/(a*ac+b*bc)
f1c[a_,ac_,b_,bc_]:=a/(a*ac+b*bc)
f2c[a_,ac_,b_,bc_]:=-bc/(a*ac+b*bc)
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"]
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

2) The function $\frac{1}{p^2}$

```
(*The function 1/p^2*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[1/p^2];
sr:=(Sqrt[(a+ac)/2+Sqrt[a*ac+b*bc])/2]);
f1i[a_,ac_,b_,bc_]:=a^2-b*bc;
f1ic[a_,ac_,b_,bc_]:=ac^2-b*bc;
f2i[a_,ac_,b_,bc_]:=(a+ac)*b;
f2ic[a_,ac_,b_,bc_]:=(a+ac)*bc;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
(*for 1/Logp*)
f1vor[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1vorc[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2vor[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2vorc[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
f1[a_,ac_,b_,bc_]:=Simplify[f1vor[a,ac,b,bc]];
f1c[a_,ac_,b_,bc_]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a_,ac_,b_,bc_]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a_,ac_,b_,bc_]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"];f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

3) The function $\frac{1}{p^3}$

```
(*The function 1/p^3*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[1/p^3];
modp3square:=(a^3-(2*a+ac)*b*bc)*(ac^3-
(2*ac+a)*b*bc)+((a^2+a*ac+ac^2)*b-
b^2*bc)*((a^2+a*ac+ac^2)*bc-bc^2*b);
f1[a_,ac_,b_,bc_]:=(ac^3-(2*ac+a)*b*bc)/modp3square;
f1c[a_,ac_,b_,bc_]:=(a^3-(2*a+ac)*b*bc)/modp3square;
f2[a_,ac_,b_,bc_]:=-((a^2+a*ac+ac^2)*b-b^2*bc)/modp3square;
f2c[a_,ac_,b_,bc_]:=-((ac^2+a*ac+a^2)*bc-bc^2*b)/modp3square;
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"];f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

4) The function $\frac{1}{\sin p}$

```
(*Function 1/sinp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="1/sinp";
modsinpsquare:=(((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v))*(((emin+epl)*Sin[(a+ac)/2])/2-((ac-
a)*(emin-epl)*Cos[(a+ac)/2])/(4*v))+(-((emin-
epl)*Cos[(a+ac)/2]*b)/(2*v))*(-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v));
f1vor[a_,ac_,b_,bc_]:=(((emin+epl)*Sin[(a+ac)/2])/2-((ac-a)*(emin-
epl)*Cos[(a+ac)/2])/(4*v))/modsinpsquare;
f1vorc[a_,ac_,b_,bc_]:=(((emin+epl)*Sin[(a+ac)/2])/2-((a-
ac)*(emin-epl)*Cos[(a+ac)/2])/(4*v))/modsinpsquare;
f2vor[a_,ac_,b_,bc_]:=(((emin-
epl)*Cos[(a+ac)/2]*b)/(2*v))/modsinpsquare;
f2vorc[a_,ac_,b_,bc_]:=(((emin-
epl)*Cos[(a+ac)/2]*bc)/(2*v))/modsinpsquare;
f1[a_,ac_,b_,bc_]:=FullSimplify[f1vor[a,ac,b,bc]];
f1c[a_,ac_,b_,bc_]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a_,ac_,b_,bc_]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a_,ac_,b_,bc_]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

5) The function $\frac{1}{\cos p}$

```
(*The function 1/cosp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
```

```
quatfunctiontested:="1/cosp";
f1i[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((a-ac)*(emin-epl)*Sin[(a+ac)/2])/(4*v);
f2i[a_,ac_,b_,bc_]:=((emin-epl)*b*Sin[(a+ac)/2])/(2*v) ;
f1ic[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((ac-a)*(emin-epl)*Sin[(a+ac)/2])/(4*v);
f2ic[a_,ac_,b_,bc_]:=((emin-epl)*bc*Sin[(a+ac)/2])/(2*v) ;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
f1[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1c[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2c[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
Print["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

6) The function $\frac{1}{e^p}$

```
(*The function 1/e^p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="1/e^p";
f1i[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(a-ac)*Sin[v])/v;
f2i[a_,ac_,b_,bc_]:=(2*beta*b*Sin[v])/v;
f1ic[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(ac-a)*Sin[v])/v;
f2ic[a_,ac_,b_,bc_]:=(2*beta*bc*Sin[v])/v;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
(*for 1/e^p*)
f1[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1c[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2c[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
Print["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

7) The function $\frac{1}{e^{p^2}}$

```
(*The function 1/e^(p^2)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:=Evaluate[1/e^(p^2)];
v=(1/2)*Sqrt[4*(a*ac+b*bc)-(a+ac)^2];
beta1=E^((a^2+ac^2-2 b bc)/2);
teta=(a+ac)*v;
f1i[a_,ac_,b_,bc_]:=beta1*(Cos[teta]+((a-ac)Sin[teta])/(2*v));
f1ic[a_,ac_,b_,bc_]:=beta1*(Cos[teta]+((ac-a)Sin[teta])/(2*v));
f2i[a_,ac_,b_,bc_]:=(beta1*Sin[teta]*b)/v;
f2ic[a_,ac_,b_,bc_]:=(beta1*Sin[teta]*bc)/v;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
(*for 1/Logp*)
f1vor[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1vorc[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2vor[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2vorc[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
f1[a_,ac_,b_,bc_]:=Simplify[f1vor[a,ac,b,bc]];
f1c[a_,ac_,b_,bc_]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a_,ac_,b_,bc_]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a_,ac_,b_,bc_]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"];f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

8) The function $\frac{1}{\sqrt{p}}$

```
(*The function 1/(p^1/2)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[1/Sqrt[p]];
sr:=(Sqrt[((a+ac)/2+Sqrt[a*ac+b*bc])/2]);
f1i[a_,ac_,b_,bc_]:=sr+(a-ac)/(4*sr);
f1ic[a_,ac_,b_,bc_]:=sr+(ac-a)/(4*sr);
f2i[a_,ac_,b_,bc_]:=b/(2*sr);
f2ic[a_,ac_,b_,bc_]:=bc/(2*sr);
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
(*for 1/Logp*)
f1vor[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1vorc[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2vor[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2vorc[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
f1[a_,ac_,b_,bc_]:=Simplify[f1vor[a,ac,b,bc]];
f1c[a_,ac_,b_,bc_]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a_,ac_,b_,bc_]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a_,ac_,b_,bc_]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"];f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

9) The function $\frac{1}{r}$

```
(*The function 1/r*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="1/r";
f1i[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
f1ic[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
f2i[a_,ac_,b_,bc_]:=b/v;
f2ic[a_,ac_,b_,bc_]:=bc/v;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
(*for 1/Logp*)
f1vor[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1vorc[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2vor[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2vorc[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
f1[a_,ac_,b_,bc_]:=Simplify[f1vor[a,ac,b,bc]];
f1c[a_,ac_,b_,bc_]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a_,ac_,b_,bc_]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a_,ac_,b_,bc_]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"];f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

10) The function $\frac{1}{\bar{r}}$

```
(*The function 1/ r_conjugate *)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="1/r_conjugate";
f1i[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
f1ic[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
f2i[a_,ac_,b_,bc_]:=-b/v;
f2ic[a_,ac_,b_,bc_]:=-bc/v;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
f1vor[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1vorc[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2vor[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2vorc[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
f1[a_,ac_,b_,bc_]:=Simplify[f1vor[a,ac,b,bc]];
f1c[a_,ac_,b_,bc_]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a_,ac_,b_,bc_]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a_,ac_,b_,bc_]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"];f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

11) The function $\frac{1}{Log\ p}$

```
(*The function 1/Log p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="1/Logp";
f1i[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
f1ic[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-a)*alpha)/(2*v);
f2i[a_,ac_,b_,bc_]:=(b*alpha)/v;
f2ic[a_,ac_,b_,bc_]:=(bc*alpha)/v;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
f1[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
f1c[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
f2[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
f2c[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

12) The function $\frac{1}{1+p^2}$

```
(*Function 1/(1+p^2)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[1/(1+p^2)];
ex1:=(1+a^2-b*bc)*(1+ac^2-b*bc)+(a+ac)^2*b*bc;
f1[a_,ac_,b_,bc_]:=(1+ac^2-b*bc)/ex1;
f2[a_,ac_,b_,bc_]:=-(b*(a+ac))/ex1;
f1c[a_,ac_,b_,bc_]:=(1+a^2-b*bc)/ex1;
f2c[a_,ac_,b_,bc_]:=-(bc*(a+ac))/ex1;
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

13) The function $\frac{1}{\sin p \cdot \cos p}$

```
(*Function 1/sinp·cosp *)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[1/( sinp·cosp)];
a1[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
a1c[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(ac+a)/2])/2-((ac-a)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
b1[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*b)/(2*v);
b1c[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v);
a2[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((a-ac)*(emin-
epl)*Sin[(a+ac)/2])/(4*v);
a2c[a_,ac_,b_,bc_]:= ((emin+epl)*Cos[(a+ac)/2])/2+((ac-a)*(emin-
epl)*Sin[(a+ac)/2])/(4*v);
b2 [a_,ac_,b_,bc_]:= ((emin-epl)*b*Sin[(a+ac)/2])/(2*v);
b2c [a_,ac_,b_,bc_]:= ((emin-epl)*bc*Sin[(a+ac)/2])/(2*v);
f1sinpcosp[a_,ac_,b_,bc_]:=a1[a,ac,b,bc]*a2[a,ac,b,bc]-
b1[a,ac,b,bc]*b2c[a,ac,b,bc];
f1sinpcospc[a_,ac_,b_,bc_]:=a1c[a,ac,b,bc]*a2c[a,ac,b,bc]-
b1c[a,ac,b,bc]*b2[a,ac,b,bc];
f2sinpcosp[a_,ac_,b_,bc_]:=a1[a,ac,b,bc]*b2[a,ac,b,bc]+a2c[a,ac,
b,bc]*b1[a,ac,b,bc];
f2sinpcospc[a_,ac_,b_,bc_]:=a1c[a,ac,b,bc]*b2c[a,ac,b,bc]+a2[a,a
c,b,bc]*b1c[a,ac,b,bc];
modsinpcospsquare:=f1sinpcosp[a,ac,b,bc]*f1sinpcospc[a,ac,b,bc
]+f2sinpcosp[a,ac,b,bc]*f2sinpcospc[a,ac,b,bc];
f1vor[a_,ac_,b_,bc_]:=f1sinpcospc[a,ac,b,bc]/modsinpcospsquare;
(*components of 1/r*Logp*)
f1vorc[a_,ac_,b_,bc_]:=f1sinpcosp[a,ac,b,bc]/modsinpcospsquare;
f2vor[a_,ac_,b_,bc_]:=-f2sinpcosp[a,ac,b,bc]/modsinpcospsquare;
f2vorc[a_,ac_,b_,bc_]:=-
f2sinpcospc[a,ac,b,bc]/modsinpcospsquare;
f1[a,ac,b,bc]:=Simplify[f1vor[a,ac,b,bc]];
f1c[a,ac,b,bc]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a,ac,b,bc]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a,ac,b,bc]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

14) The function $\frac{1}{r\cdot \mathrm{Log}\, p}$

```
(*Function 1/(r·Logp) *)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:=Evaluate[1/(r·Logp)];
a1[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
a1c[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
b1[a_,ac_,b_,bc_]:=b/v;
b1c[a_,ac_,b_,bc_]:=bc/v;
a2[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
a2c[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-a)*alpha)/(2*v);
b2[a_,ac_,b_,bc_]:=(b*alpha)/v;
b2c[a_,ac_,b_,bc_]:=(bc*alpha)/v;
f1rlogp[a_,ac_,b_,bc_]:=a1[a,ac,b,bc]*a2[a,ac,b,bc]-
b1[a,ac,b,bc]*b2c[a,ac,b,bc];
f1rlogpc[a_,ac_,b_,bc_]:=a1c[a,ac,b,bc]*a2c[a,ac,b,bc]-
b1c[a,ac,b,bc]*b2[a,ac,b,bc];
f2rlogp[a_,ac_,b_,bc_]:=a1[a,ac,b,bc]*b2[a,ac,b,bc]+a2c[a,ac,b,bc]
*b1[a,ac,b,bc];
f2rlogpc[a_,ac_,b_,bc_]:=a1c[a,ac,b,bc]*b2c[a,ac,b,bc]+a2[a,ac,b,
bc]*b1c[a,ac,b,bc];
modrlogpsquare:=f1rlogp[a,ac,b,bc]*f1rlogpc[a,ac,b,bc]+f2rlogp[a,
ac,b,bc]*f2rlogpc[a,ac,b,bc];
f1vor[a_,ac_,b_,bc_]:=f1rlogpc[a,ac,b,bc]/modrlogpsquare;
f1vorc[a_,ac_,b_,bc_]:=f1rlogp[a,ac,b,bc]/modrlogpsquare;
f2vor[a_,ac_,b_,bc_]:=-f2rlogp[a,ac,b,bc]/modrlogpsquare;
f2vorc[a_,ac_,b_,bc_]:=-f2rlogpc[a,ac,b,bc]/modrlogpsquare;
f1[a,ac,b,bc]:=Simplify[f1vor[a,ac,b,bc]];
f1c[a,ac,b,bc]:=Simplify[f1vorc[a,ac,b,bc]];
f2[a,ac,b,bc]:=Simplify[f2vor[a,ac,b,bc]];
f2c[a,ac,b,bc]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents f1,f2 of the quaternionic function
",quatfunctiontested,":"];f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;
markquot=False;
s=0;
```

3.3 Examples with Commutativity Testing

In this sub-section we consider the $\mathbb{H}$ - holomorphic functions that are explicitly represented as quaternionic products.

3.3.1 Examples of $\mathbb{H}$-holomorphic Functions that are Quaternionic Products

The above standard functions beta, v, alpha, epl, emin must be previously activated, otherwise processing of some functions can be incorrect

It is possible only the following sequence of processing of the functions from this sub-section: launching the input data of a function in question from this sub-section, then launching Program 4.1, and then launching Program 4.2 and, if needed, Program 4.3 and/or 4.4, 4.5.

1) The first derivative of the function e^{p^2}:

```
(*The first derivative (e^(p^2)) [1]= (e^(p^2))*2p *)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
```

```
v=(1/2)*Sqrt[4*(a*ac+b*bc)-(a+ac)^2];
teta=(a+ac)*v;
mu:=Cos[teta]+((a-ac)*Sin[teta])/(2*v);
muc:=Cos[teta]+((ac-a)*Sin[teta])/(2*v);
gamma:=E^((a^2+ac^2-2 b bc)/2);
quatfunctiontested:=Evaluate[e^(p^2)*2*p];
inia1[a_,ac_,b_,bc_]:=gamma*mu;
inia1c[a_,ac_,b_,bc_]:=gamma*muc;
inia2[a_,ac_,b_,bc_]:=(gamma*Sin[teta]*b)/v;
inia2c[a_,ac_,b_,bc_]:=(gamma*Sin[teta]*bc)/v;
inib1[a_,ac_,b_,bc_]:=2*a;
inib1c[a_,ac_,b_,bc_]:=2*ac;
inib2[a_,ac_,b_,bc_]:=2*b;
inib2c[a_,ac_,b_,bc_]:=2*bc;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

2) The second derivative of the function e^{p^2}

```
(*The second derivative (e^p^2)[2]=2e^p^2*(2p^2+1) *)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
v=(1/2)*Sqrt[4*(a*ac+b*bc)-(a+ac)^2];teta=(a+ac)*v;
mu:=Cos[teta]+((a-ac)*Sin[teta])/(2*v);muc:=Cos[teta]+((ac-
a)*Sin[teta])/(2*v);
gamma:=E^((a^2+ac^2-2 b bc)/2);
quatfunctiontested:=Evaluate[2*e^(p^2)*(2*p^2+1)];
inia1[a_,ac_,b_,bc_]:=2*gamma*mu;
inia1c[a_,ac_,b_,bc_]:=2*gamma*muc;
inia2[a_,ac_,b_,bc_]:=(2*gamma*Sin[teta]*b)/v;
inia2c[a_,ac_,b_,bc_]:=(2*gamma*Sin[teta]*bc)/v;
inib1[a_,ac_,b_,bc_]:=1+2*(a^2-b*bc);
inib1c[a_,ac_,b_,bc_]:=1+2*(ac^2-b*bc);
inib2[a_,ac_,b_,bc_]:=2*b*(a+ac);
inib2c[a_,ac_,b_,bc_]:=2*bc*(a+ac);
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

3) The function $r \cdot Log\ p$

```
(*The function r·Logp *)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="r·Logp";
inia1[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
inia1c[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
inia2[a_,ac_,b_,bc_]:=b/v;
inia2c[a_,ac_,b_,bc_]:=bc/v;
inib1[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
inib1c[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-
a)*alpha)/(2*v);
inib2[a_,ac_,b_,bc_]:=(b*alpha)/v;
inib2c[a_,ac_,b_,bc_]:=(bc*alpha)/v;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

4) The function $p^2 \cdot Log\ p$

```
(*The function p^2·Log p *)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:=Evaluate[p^2•Logp];
inia1[a_,ac_,b_,bc_]:=a^2-b*bc;
inia1c[a_,ac_,b_,bc_]:=ac^2-b*bc;
inia2[a_,ac_,b_,bc_]:=(a+ac)*b;
inia2c[a_,ac_,b_,bc_]:=(a+ac)*bc;
inib1[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
inib1c[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-
a)*alpha)/(2*v);
inib2[a_,ac_,b_,bc_]:=(b*alpha)/v;
inib2c[a_,ac_,b_,bc_]:=(bc*alpha)/v;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2•j)•(inib1+inib2•j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2•j)•(inib1+inib2•j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

5) The function $p^2 \cdot \left(\frac{1}{r}\right)$

```
(*The function p^2•(1/r)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
v=(1/2)*Sqrt[4*(a*ac+b*bc)-(a+ac)^2];
quatfunctiontested:="p^2•(1/r)";
inia1[a_,ac_,b_,bc_]:= a^2-b*bc;
inia1c[a_,ac_,b_,bc_]:= ac^2-b*bc;
inia2 [a_,ac_,b_,bc_]:= (a+ac)*b;
inia2c [a_,ac_,b_,bc_]:= (a+ac)*bc;
inib1[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
inib1c[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
inib2[a_,ac_,b_,bc_]:=-b/v ;
inib2c[a_,ac_,b_,bc_]:=-bc/v;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

6) The function $Log\ p \cdot \left(\frac{1}{r}\right)$

```
(*The function Logp·(1/r)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
teta=(a+ac)*v;
quatfunctiontested:="Logp·(1/r)";
inia1[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
inia1c[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-
a)*alpha)/(2*v);
inia2[a_,ac_,b_,bc_]:=(b*alpha)/v;
inia2c[a_,ac_,b_,bc_]:=(bc*alpha)/v;
inib1[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
inib1c[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
inib2[a_,ac_,b_,bc_]:=-b/v;
inib2c[a_,ac_,b_,bc_]:=-bc/v;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
```

```
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

7) The function $(\boldsymbol{r}\cdot\boldsymbol{Log}\ \boldsymbol{p})'=\boldsymbol{r}\cdot\left(\frac{\boldsymbol{1}}{\boldsymbol{p}}\right)$

```
(*The function r·(1/p)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="r•1/p";
inia1[a_,ac_,b_,bc_]:=(a-ac)/(2*v)
inia1c[a_,ac_,b_,bc_]:=(ac-a)/(2*v)
inia2[a_,ac_,b_,bc_]:=b/v
inia2c[a_,ac_,b_,bc_]:=bc/v
inib1[a_,ac_,b_,bc_]:=ac/(a*ac+b*bc)
inib1c[a_,ac_,b_,bc_]:=a/(a*ac+b*bc)
inib2[a_,ac_,b_,bc_]:=-b/(a*ac+b*bc)
inib2c[a_,ac_,b_,bc_]:=-bc/(a*ac+b*bc)
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2•j)•(inib1+inib2•j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2•j)•(inib1+inib2•j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

8) The function $\boldsymbol{\sin p}\cdot\boldsymbol{\cos p}$

```
(*The function sinp·cosp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="sinp·cosp";
inia1[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
inia1c[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((ac-a)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
inia2[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*b)/(2*v);
inia2c[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v);
inib1[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((a-ac)*(emin-
epl)*Sin[(a+ac)/2])/(4*v);
inib1c[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((ac-
a)*(emin-epl)*Sin[(a+ac)/2])/(4*v);
inib2 [a_,ac_,b_,bc_]:= ((emin-epl)*b*Sin[(a+ac)/2])/(2*v);
inib2c [a_,ac_,b_,bc_]:=((emin-epl)*bc*Sin[(a+ac)/2])/(2*v);
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

9) The function $\boldsymbol{Log\ p}\cdot\boldsymbol{\sin p}$

```
(*The function Logp·sinp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="Logp·sinp";
inia1[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
inia1c[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-
a)*alpha)/(2*v);
inia2[a_,ac_,b_,bc_]:=(b*alpha)/v;
inia2c[a_,ac_,b_,bc_]:=(bc*alpha)/v;
inib1[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
inib1c[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((ac-a)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
inib2[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*b)/(2*v);
inib2c[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v);
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=False;
s=0;
```

3.3.2 Examples of $\mathbb{H}$-holomorphic Functions that are Quaternionic Quotients

The above standard functions beta, v, alpha, epl, emin must be previously activated, otherwise processing of some functions can be incorrect.

Since the quaternionic quotients can be represented as quaternionic products, this sub-section can be considered as an extension of the previous sub-section. We will represent the right quotient of $f(p)$ by $g(p)$ as a quaternionic product $\frac{f(p)}{g(p)}=f(p)\cdot\frac{1}{g(p)}=f(p)\cdot\frac{\overline{g(p)}}{g(p)\cdot\overline{g(p)}}$ and show that this product in the case of $\mathbb{H}$-holomorphic functions is commutative, i.e. the right quotient equals the left one [1].

1) The function $\boldsymbol{e^p/p}$

```
(*The right quotient of e^p by p: e^p/p=e^p·(1/p)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="e^p/p=e^p·(1/p)";
inia1[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(a-ac)*Sin[v])/v;
inia1c[a_,ac_,b_,bc_]:=2*beta*Cos[v]+(beta*(ac-a)*Sin[v])/v;
inia2[a_,ac_,b_,bc_]:=(2*beta*b*Sin[v])/v;
inia2c[a_,ac_,b_,bc_]:=(2*beta*bc*Sin[v])/v;
f1i[a_,ac_,b_,bc_]:=a;
f1ic[a_,ac_,b_,bc_]:=ac;
f2i[a_,ac_,b_,bc_]:=b;
f2ic[a_,ac_,b_,bc_]:=bc;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
inib1[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
inib1c[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
inib2[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
inib2c[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

2) The function $\frac{\boldsymbol{p^2}}{\boldsymbol{r}}=\boldsymbol{p^2}\cdot\left(\frac{\boldsymbol{1}}{\boldsymbol{r}}\right)$

```
(*The right quotient of p^2 by r: p^2·(1/r) = (inia1+inia2·j) ·
(inib1+inib2·j)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
v=(1/2)*Sqrt[4*(a*ac+b*bc)-(a+ac)^2];
quatfunctiontested:="p^2·(1/r)";
inia1[a_,ac_,b_,bc_]:= a^2-b*bc;
inia1c[a_,ac_,b_,bc_]:= ac^2-b*bc;
inia2 [a_,ac_,b_,bc_]:= (a+ac)*b;
inia2c [a_,ac_,b_,bc_]:= (a+ac)*bc;
inib1[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
inib1c[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
inib2[a_,ac_,b_,bc_]:=-b/v ;
```

```
inib2c[a_,ac_,b_,bc_]:=-bc/v;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

3) The function $\cdot \frac{r}{\sqrt{p}}$

```
(*The right quotient of r by p^(1/2): r·1/(p^(1/2))*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="r/p^(1/2)= r·1/(p^(1/2))";
sr:=(Sqrt[((a+ac)/2+Sqrt[a*ac+b*bc])/2]);
vorinia1[a_,ac_,b_,bc_]:=(a-ac)/(2*v);
vorinia1c[a_,ac_,b_,bc_]:=(ac-a)/(2*v);
vorinia2[a_,ac_,b_,bc_]:=b/v;
vorinia2c[a_,ac_,b_,bc_]:=bc/v;
inia1[a_,ac_,b_,bc_]:=Simplify[vorinia1[a,ac,b,bc]];
inia1c[a_,ac_,b_,bc_]:=Simplify[vorinia1c[a,ac,b,bc]];
inia2[a_,ac_,b_,bc_]:=Simplify[vorinia2[a,ac,b,bc]];
inia2c[a_,ac_,b_,bc_]:=Simplify[vorinia2c[a,ac,b,bc]];
vorf1i[a_,ac_,b_,bc_]:=sr+(a-ac)/(4*sr);
vorf1ic[a_,ac_,b_,bc_]:=sr+(ac-a)/(4*sr);
vorf2i[a_,ac_,b_,bc_]:=b/(2*sr);
vorf2ic[a_,ac_,b_,bc_]:=bc/(2*sr);
f1i[a_,ac_,b_,bc_]:=Simplify[vorf1i[a,ac,b,bc]];
f1ic[a_,ac_,b_,bc_]:=Simplify[vorf1ic[a,ac,b,bc]];
f2i[a_,ac_,b_,bc_]:=Simplify[vorf2i[a,ac,b,bc]];
f2ic[a_,ac_,b_,bc_]:=Simplify[vorf2ic[a,ac,b,bc]];
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
inib1[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
inib1c[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
inib2[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
inib2c[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

4) The function $\sin p \,/ Log\, p$

```
(*The right quotient of sinp by Logp:  sinp/Logp= sinp·(1/Logp)" *)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="sinp/Logp= sinp·(1/Logp)";
inia1[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
inia1c[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((ac-a)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
inia2[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*b)/(2*v);
inia2c[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v);
f1i[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
f1ic[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-a)*alpha)/(2*v);
f2i[a_,ac_,b_,bc_]:=(b*alpha)/v;
f2ic[a_,ac_,b_,bc_]:=(bc*alpha)/v;
abs2:=(f1i[a,ac,b,bc]*f1ic[a,ac,b,bc]+f2i[a,ac,b,bc]*f2ic[a,ac,b,bc]);
inib1[a_,ac_,b_,bc_]:=f1ic[a,ac,b,bc]/abs2;
inib1c[a_,ac_,b_,bc_]:=f1i[a,ac,b,bc]/abs2;
inib2[a_,ac_,b_,bc_]:=-f2i[a,ac,b,bc]/abs2;
inib2c[a_,ac_,b_,bc_]:=-f2ic[a,ac,b,bc]/abs2;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

5) The function $Log\, p / \sin p$

```
(*The right quotient of Logp by sinp: = Logp/sinp = Logp ·(1/
sinp);=(inia1+inia2·j) · (inib1+inib2·j)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="Logp/sinp =Logp·(1/sinp)";
modsinpsquare:=(((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v))*(((emin+epl)*Sin[(a+ac)/2])/2-((ac-
a)*(emin-epl)*Cos[(a+ac)/2])/(4*v))+(-((emin-
epl)*Cos[(a+ac)/2]*b)/(2*v))*(-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v));
modsinp2=modsinpsquare;
vorinia1[a_,ac_,b_,bc_]:=(((emin+epl)*Sin[(a+ac)/2])/2-((ac-
a)*(emin-epl)*Cos[(a+ac)/2])/(4*v))/modsinp2;
vorinia1c[a_,ac_,b_,bc_]:=(((emin+epl)*Sin[(a+ac)/2])/2-((a-
ac)*(emin-epl)*Cos[(a+ac)/2])/(4*v))/modsinp2;
vorinia2[a_,ac_,b_,bc_]:=(((emin-
epl)*Cos[(a+ac)/2]*b)/(2*v))/modsinp2;
vorinia2c[a_,ac_,b_,bc_]:=(((emin-
epl)*Cos[(a+ac)/2]*bc)/(2*v))/modsinp2;
inia1[a_,ac_,b_,bc_]:=Simplify[vorinia1[a,ac,b,bc]];
inia1c[a_,ac_,b_,bc_]:=Simplify[vorinia1c[a,ac,b,bc]];
inia2[a_,ac_,b_,bc_]:=Simplify[vorinia2[a,ac,b,bc]];
inia2c[a_,ac_,b_,bc_]:=Simplify[vorinia2c[a,ac,b,bc]];
inib1[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((a-ac)*alpha)/(2*v);
inib1c[a_,ac_,b_,bc_]:=Log[Sqrt[(a*ac+b*bc)]]+((ac-
a)*alpha)/(2*v);
inib2[a_,ac_,b_,bc_]:=(b*alpha)/v;
inib2c[a_,ac_,b_,bc_]:=(bc*alpha)/v;
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

6) The function $e^{p^2}/(\sin p \cdot \cos p)$

```
(*The right quotient of e^(p^2) by sinp·cosp: e^(p^2)/(sinp·cosp)=
e^(p^2)·(1/(sinp·cosp)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,
quatfunctiontested];
quatfunctiontested:="e^(p^2)/(sinp·cosp)= e^(p^2)·(1/(sinp·cosp))";
beta1=E^((a^2+ac^2-2 b bc)/2);
teta=(a+ac)*v;
inia1[a_,ac_,b_,bc_]:=beta1*(Cos[teta]+((a-ac)Sin[teta])/(2*v));
inia1c[a_,ac_,b_,bc_]:=beta1*(Cos[teta]+((ac-a)Sin[teta])/(2*v));
inia2[a_,ac_,b_,bc_]:=(beta1*Sin[teta]*b)/v;
inia2c[a_,ac_,b_,bc_]:=(beta1*Sin[teta]*bc)/v;
a1[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
a1c[a_,ac_,b_,bc_]:=((emin+epl)*Sin[(ac+a)/2])/2-((ac-a)*(emin-
epl)*Cos[(a+ac)/2])/(4*v);
b1[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*b)/(2*v);
b1c[a_,ac_,b_,bc_]:=-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v);
a2[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((a-ac)*(emin-
epl)*Sin[(a+ac)/2])/(4*v);
a2c[a_,ac_,b_,bc_]:=((emin+epl)*Cos[(a+ac)/2])/2+((ac-a)*(emin-
epl)*Sin[(a+ac)/2])/(4*v);
b2[a_,ac_,b_,bc_]:=((emin-epl)*b*Sin[(a+ac)/2])/(2*v);
b2c[a_,ac_,b_,bc_]:=((emin-epl)*bc*Sin[(a+ac)/2])/(2*v);
f1sinpcosp[a_,ac_,b_,bc_]:=a1[a,ac,b,bc]*a2[a,ac,b,bc]-
b1[a,ac,b,bc]*b2c[a,ac,b,bc];
f1sinpcospc[a_,ac_,b_,bc_]:=a1c[a,ac,b,bc]*a2c[a,ac,b,bc]-
b1c[a,ac,b,bc]*b2[a,ac,b,bc];
f2sinpcosp[a_,ac_,b_,bc_]:=a1[a,ac,b,bc]*b2[a,ac,b,bc]+a2c[a,ac,
b,bc]*b1[a,ac,b,bc];
f2sinpcospc[a_,ac_,b_,bc_]:=a1c[a,ac,b,bc]*b2c[a,ac,b,bc]+a2[a,a
c,b,bc]*b1c[a,ac,b,bc];
```

```
modsinpcospsquare:=f1sinpcosp[a,ac,b,bc]*f1sinpcospc[a,ac,b,bc
]+f2sinpcosp[a,ac,b,bc]*f2sinpcospc[a,ac,b,bc];
f1vor[a_,ac_,b_,bc_]:=f1sinpcospc[a,ac,b,bc]/modsinpcospsquare;
f1vorc[a_,ac_,b_,bc_]:=f1sinpcosp[a,ac,b,bc]/modsinpcospsquare;
f2vor[a_,ac_,b_,bc_]:=-f2sinpcosp[a,ac,b,bc]/modsinpcospsquare;
f2vorc[a_,ac_,b_,bc_]:=-
f2sinpcospc[a,ac,b,bc]/modsinpcospsquare;
inib1[a_,ac_,b_,bc_]:=Simplify[f1vor[a,ac,b,bc]];
inib1c[a_,ac_,b_,bc_]:=Simplify[f1vorc[a,ac,b,bc]];
inib2[a_,ac_,b_,bc_]:=Simplify[f2vor[a,ac,b,bc]];
inib2c[a_,ac_,b_,bc_]:=Simplify[f2vorc[a,ac,b,bc]];
Print["Constituents inia1, inia2 of the quaternionic quotient
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic quotient
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

7) The function $\tan p = \sin p / \cos p$

```
(*The right quotient of sinp by cosp: tanp= sinp/cosp = sinp ·(1/
cosp) = (inia1+inia2·j) · (inib1+inib2·j)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="tanp=sinp/cosp = sinp·(1/cosp)";
modcospsquare:=(((emin+epl)*Cos[(a+ac)/2])/2+((a-ac)*(emin-
epl)*Sin[(a+ac)/2])/(4*v))*(((emin+epl)*Cos[(a+ac)/2])/2+((ac-
a)*(emin-epl)*Sin[(a+ac)/2])/(4*v))+(((emin-
epl)*b*Sin[(a+ac)/2])/(2*v))*(((emin-epl)*bc*Sin[(a+ac)/2])/(2*v));
modcosp2=modcospsquare;
vorinia1[a_,ac_,b_,bc_]:=(((emin+epl)*Cos[(a+ac)/2])/2+((ac-
a)*(emin-epl)*Sin[(a+ac)/2])/(4*v))/modcosp2;
vorinia1c[a_,ac_,b_,bc_]:=(((emin+epl)*Cos[(a+ac)/2])/2+((a-
ac)*(emin-
epl)*Sin[(a+ac)/2])/(4*v))/modcosp2;vorinia2[a_,ac_,b_,bc_]:=(-
((emin-epl)*b*Sin[(a+ac)/2])/(2*v))/modcosp2;
vorinia2c[a_,ac_,b_,bc_]:=(-((emin-
epl)*bc*Sin[(a+ac)/2])/(2*v))/modcosp2;
inia1[a_,ac_,b_,bc_]:=Simplify[vorinia1[a,ac,b,bc]];
inia1c[a_,ac_,b_,bc_]:=Simplify[vorinia1c[a,ac,b,bc]];
inia2[a_,ac_,b_,bc_]:=Simplify[vorinia2[a,ac,b,bc]];
inia2c[a_,ac_,b_,bc_]:=Simplify[vorinia2c[a,ac,b,bc]];
vorinib1[a_,ac_,b_,bc_]:= ((emin+epl)*Sin[(a+ac)/2])/2-((a-
ac)*(emin-epl)*Cos[(a+ac)/2])/(4*v);
vorinib1c[a_,ac_,b_,bc_]:= ((emin+epl)*Sin[(a+ac)/2])/2-((ac-
a)*(emin-epl)*Cos[(a+ac)/2])/(4*v);
vorinib2 [a_,ac_,b_,bc_]:= -((emin-epl)*Cos[(a+ac)/2]*b)/(2*v);
vorinib2c [a_,ac_,b_,bc_]:= -((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v);
inib1[a_,ac_,b_,bc_]:=Simplify[vorinib1[a,ac,b,bc]];
inib1c[a_,ac_,b_,bc_]:=Simplify[vorinib1c[a,ac,b,bc]];
inib2[a_,ac_,b_,bc_]:=Simplify[vorinib2[a,ac,b,bc]];
inib2c[a_,ac_,b_,bc_]:=Simplify[vorinib2c[a,ac,b,bc]];
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

8) The function $\cot p = \cos p / \sin p$

```
(*The right quotient of cosp by sinp: cotp=cosp/sinp = cosp·(1/
sinp) = (inia1+inia2·j)·(inib1+inib2·j)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c,
inib2,inib2c, quatfunctiontested];
quatfunctiontested:="cotp=cosp/sinp = cosp·(1/sinp)";
modsinpsquare:=(((emin+epl)*Sin[(a+ac)/2])/2-((a-ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v))*(((emin+epl)*Sin[(a+ac)/2])/2-((ac-
a)*(emin-epl)*Cos[(a+ac)/2])/(4*v))+(-((emin-
epl)*Cos[(a+ac)/2]*b)/(2*v))*(-((emin-epl)*Cos[(a+ac)/2]*bc)/(2*v));
modsinp2=modsinpsquare;
vorinia1[a_,ac_,b_,bc_]:=(((emin+epl)*Sin[(a+ac)/2])/2-((ac-
a)*(emin-epl)*Cos[(a+ac)/2])/(4*v))/modsinp2;
vorinia1c[a_,ac_,b_,bc_]:=(((emin+epl)*Sin[(a+ac)/2])/2-((a-
ac)*(emin-
epl)*Cos[(a+ac)/2])/(4*v))/modsinp2;vorinia2[a_,ac_,b_,bc_]:=(((e
min-epl)*Cos[(a+ac)/2]*b)/(2*v))/modsinp2;
vorinia2c[a_,ac_,b_,bc_]:=(((emin-
epl)*Cos[(a+ac)/2]*bc)/(2*v))/modsinp2;
inia1[a_,ac_,b_,bc_]:=Simplify[vorinia1[a,ac,b,bc]];
inia1c[a_,ac_,b_,bc_]:=Simplify[vorinia1c[a,ac,b,bc]];
inia2[a_,ac_,b_,bc_]:=Simplify[vorinia2[a,ac,b,bc]];
inia2c[a_,ac_,b_,bc_]:=Simplify[vorinia2c[a,ac,b,bc]];
vorinib1[a_,ac_,b_,bc_]:= ((emin+epl)*Cos[(a+ac)/2])/2+((a-
ac)*(emin-epl)*Sin[(a+ac)/2])/(4*v);
vorinib1c[a_,ac_,b_,bc_]:= ((emin+epl)*Cos[(a+ac)/2])/2+((ac-
a)*(emin-epl)*Sin[(a+ac)/2])/(4*v);
vorinib2 [a_,ac_,b_,bc_]:= ((emin-epl)*b*Sin[(a+ac)/2])/(2*v);
vorinib2c [a_,ac_,b_,bc_]:= ((emin-epl)*bc*Sin[(a+ac)/2])/(2*v);
inib1[a_,ac_,b_,bc_]:=Simplify[vorinib1[a,ac,b,bc]];
inib1c[a_,ac_,b_,bc_]:=Simplify[vorinib1c[a,ac,b,bc]];
inib2[a_,ac_,b_,bc_]:=Simplify[vorinib2[a,ac,b,bc]];
inib2c[a_,ac_,b_,bc_]:=Simplify[vorinib2c[a,ac,b,bc]];
Print["Constituents inia1, inia2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inia1[a,ac,b,bc]
inia2[a,ac,b,bc]
Print["Constituents inib1, inib2 of the quaternionic function
",quatfunctiontested," =(inia1+inia2·j)·(inib1+inib2·j):"];
inib1[a,ac,b,bc]
inib2[a,ac,b,bc]
mark=True;
markquot=True;
s=0;
```

Function 5) from sub-section 3.3.1 and function 2) from sub-section 3.3.2 are identical, however the first function is processed as a quaternionic product and the second as a quaternionic quotient. The difference is in setting the variable markquot. We can also process functions 6) and 7) from sub-section 3.3.1 as quaternionic quotients by setting markquot =True in input data for these functions. To process the higher derivatives we must add setting s=0.

4. Main Processing Programmes

Programmes are adapted for the direct transfer of these programmes (copying) into empty file .nb of the computing system Wolfram Mathematica®. To obtain the functions f1, f2 as input data for Programmes 4.2, 4.3, 4.4 and 4.5 it is needed to launch data of a function in question from sub-sections 3.1 and 3.2. After that we can launch Programmes 4.2, 4.3 as well as 4.4 or 4.5 for processing of this function. The data from sub-section 3.3 cannot be directly used in Programmes 4.2, 4.3, 4.4 and 4.5. We need first to launch Program 4.1 for these data, which generates the functions f1 and f2.

We do not consider in this article programmes of plotting equipotential and stream surfaces of fields in 3D space.

4.1 Commutativity Testing

The above standard functions beta, v, alpha, epl, emin must previously be activated, otherwise processing of some functions can be incorrect.

This programm must be used only when a quaternionic function in question is represented directly as a quaternionic product as well as if we want to verify whether the quaternionic mulniplication of the included functions behaves as commutative. Since a quotient of two $\mathbb{H}$ -

holomorphic functions can be represented as a quaternionic product, this program tests also whether the right quotient of two $\mathbb{H}$ - holomorphic functions is equal to the left one.

Finally, this program, using the input functions inia1, inia2, inib1, inib2, calculates the functions f1 and f2 of the quaternionic product in question to use them further as input functions for the programes 4.2, 4.3 and 4.4.

The input data for this program are only the data from sub-section 3.3.

```
(*Program 4.1, Commutativity testing*)
Clear[vorf1,vorf1c,f1,f1c,q1q2f1,q1q2f1c,vorf2,vorf2c,f2,f2c,q1q2f2
,q1q2f2c,vorfif1,vorfif1c,fif1,fif1c,q2q1f1,q2q1f1c,vorfif2,vorfif2c,fif2
,fif2c,q2q1f2,q2q1f2c,coins1,coins2,coins];
vorf1[a_,ac_,b_,bc_]=inia1[a,ac,b,bc]*inib1[a,ac,b,bc]-
inia2[a,ac,b,bc]*inib2c[a,ac,b,bc];
vorf1c[a_,ac_,b_,bc_]=inia1c[a,ac,b,bc]*inib1c[a,ac,b,bc]-
inia2c[a,ac,b,bc]*inib2[a,ac,b,bc];
f1[a_,ac_,b_,bc_]=Simplify[vorf1[a,ac,b,bc],TimeConstraint-
>9400];
f1c[a_,ac_,b_,bc_]=Simplify[vorf1c[a,ac,b,bc],TimeConstraint-
>9400];
q1q2f1:=f1[a,ac,b,bc];
q1q2f1c:=f1c[a,ac,b,bc];
vorf2[a_,ac_,b_,bc_]=inia1[a,ac,b,bc]*inib2[a,ac,b,bc]+inia2[a,ac,b
,bc]*inib1c[a,ac,b,bc];
vorf2c[a_,ac_,b_,bc_]=inia1c[a,ac,b,bc]*inib2c[a,ac,b,bc]+inia2c[a,
ac,b,bc]*inib1[a,ac,b,bc];
f2[a_,ac_,b_,bc_]=Simplify[vorf2[a,ac,b,bc],TimeConstraint-
>9400];
f2c[a_,ac_,b_,bc_]=Simplify[vorf2c[a,ac,b,bc],TimeConstraint-
>9400];
Print["Result of computing: Constituents f1, f2 of
",quatfunctiontested,"= f1+f2*j :"];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
q1q2f2:=f2[a,ac,b,bc];
q1q2f2c:=f2c[a,ac,b,bc];
vorfif1[a_,ac_,b_,bc_]=inia1[a,ac,b,bc]*inib1[a,ac,b,bc]-
inia2c[a,ac,b,bc]*inib2[a,ac,b,bc];
vorfif1c[a_,ac_,b_,bc_]=inia1c[a,ac,b,bc]*inib1c[a,ac,b,bc]-
inia2[a,ac,b,bc]*inib2c[a,ac,b,bc];
fif1[a_,ac_,b_,bc_]=Simplify[vorfif1[a,ac,b,bc],TimeConstraint-
>9400];
fif1c[a_,ac_,b_,bc_]=Simplify[vorfif1c[a,ac,b,bc],TimeConstraint-
>9400];
q2q1f1:=fif1[a,ac,b,bc];
q2q1f1c:=fif1c[a,ac,b,bc];
vorfif2[a_,ac_,b_,bc_]=inia1c[a,ac,b,bc]*inib2[a,ac,b,bc]+inia2[a,ac
,b,bc]*inib1[a,ac,b,bc];
vorfif2c[a_,ac_,b_,bc_]=inia1[a,ac,b,bc]*inib2c[a,ac,b,bc]+inia2c[a,
ac,b,bc]*inib1c[a,ac,b,bc];
fif2[a_,ac_,b_,bc_]=Simplify[vorfif2[a,ac,b,bc],TimeConstraint-
>9400];
fif2c[a_,ac_,b_,bc_]=Simplify[vorfif2c[a,ac,b,bc],TimeConstraint-
>9400];
q2q1f2:=fif2[a,ac,b,bc];
q2q1f2c:=fif2c[a,ac,b,bc];
coins1:=True===Simplify[q1q2f1==q2q1f1,TimeConstraint->9400];
coins2:=True===Simplify[q1q2f2==q2q1f2,TimeConstraint->9400];
coins:=coins1&&coins2;
If[f1[a,ac,b,bc]===(inia1[a,ac,b,bc] inib1[a,ac,b,bc]-inia2[a,ac,b,bc]
inib2c[a,ac,b,bc]),Print["////////////////// ERROR://///////////////\n Incorrect
data input! Using Programmes 4.2 and/or 4.3 shall be further
incorrect. Please, launch again a function represented only as
product of two H -holomorphic functions:
(inia1+inia2j)(inib1+inib2j), then launch once more this
  program."],If[coins,Print["Result of testing: Quaternionic
multiplication in the case of ",quatfunctiontested," behaves as
commutative."],Print["Result of testing: Quaternionic multiplication
in the case of ",quatfunctiontested," is non-commutative."]]]
If[f1[a,ac,b,bc]===(inia1[a,ac,b,bc] inib1[a,ac,b,bc]-inia2[a,ac,b,bc]
inib2c[a,ac,b,bc]),,If[markquot,If[markquot&&coins,Print["The right
quaternionic quotient ",quatfunctiontested," is equal to the left
one."],Print["The right quaternionic quotient in the case of
",quatfunctiontested," is not equal to the left one."]]]]
Clear[inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c];
mark=False;
markquot=False;
```

4.2 Testing $\mathbb{H}$-holomorphy

The input data for this program are only the functions f1, f2, f1c, f2c. To get, for example, the expression $(\partial_a f1|$ for the system of equations (2) this program calculates the derivative of f1 with respect to a, and then sets $y = 0$ (the transition to 3D space) in the obtained expression for derivative.

```
(*Program 4.2, H - holomorphy testing*)
ClearAll[df1,df1c,df2,df2c,trdf1,sftrdf1,trdf1c,sftrdf1c,trdf2,sftrdf2,tr
df2c,sftrdf2c,trmdf1da,trmdf2cdbc,dif1,trmdf2da,trmdf1cdbc,dif2,tr
mdf2db,dif3,trmdf2dac,trmdf1dbc,dif4,eq1,eq2,eq3,eq4];
If[(f1[a,ac,b,bc]===inia1[a,ac,b,bc] inib1[a,ac,b,bc]-inia2[a,ac,b,bc]
inib2c[a,ac,b,bc])||mark,Print["////////////////// ERROR://///////////////\n
Incorrect data input! Repeat again launching the function in
question and then the correct sequence of launching the needed
main programmes."],
 df1=D[f1[a,ac,b,bc],{{a,bc}}];
 trdf1=df1/.{a->x,ac->x};
 sftrdf1=Simplify[trdf1,TimeConstraint->9400];
 ClearAll[trdf1,df1];
 df1c=D[f1c[a,ac,b,bc],{bc}];
 trdf1c=df1c/.{a->x,ac->x};
 sftrdf1c=Simplify[trdf1c,TimeConstraint->9400];
 ClearAll[trdf1c,df1c];
 df2=D[f2[a,ac,b,bc],{{a,ac,b}}];
 trdf2=df2/.{a->x,ac->x};
 sftrdf2=Simplify[trdf2,TimeConstraint->9400];
 ClearAll[trdf2,df2];
 df2c=D[f2c[a,ac,b,bc],{bc}];
 trdf2c=df2c/.{a->x,ac->x};
 sftrdf2c=Simplify[trdf2c,TimeConstraint->9400];
 ClearAll[trdf2c,df2c];
 trmdf1da=PowerExpand[sftrdf1[[1]]]//Expand;
 trmdf2cdbc=PowerExpand[sftrdf2c]//Expand;
 dif1=Simplify[trmdf1da-trmdf2cdbc,TimeConstraint->9400];
 ClearAll[trmdf2cdbc];
 eq1=True===Simplify[dif1==0,TimeConstraint->9400];
 ClearAll[dif1];
 trmdf2da=PowerExpand[sftrdf2[[1]]]//Expand;
 trmdf1cdbc=PowerExpand[-sftrdf1c]//Expand;
 dif2=Simplify[trmdf2da-trmdf1cdbc,TimeConstraint->9400];
 ClearAll[trmdf2da,trmdf1cdbc];
 eq2=True===Simplify[dif2==0,TimeConstraint->9400];
 ClearAll[dif2];
 trmdf2db=PowerExpand[sftrdf2[[3]]]//Expand;
 dif3=Simplify[trmdf1da-trmdf2db,TimeConstraint->9400];
 ClearAll[trmdf1da,trmdf2db];
 eq3=True===Simplify[dif3==0,TimeConstraint->9400];
 ClearAll[dif3];
 trmdf2dac=PowerExpand[sftrdf2[[2]]]//Expand;
 trmdf1dbc=PowerExpand[-sftrdf1[[2]]]//Expand;
 dif4=Simplify[trmdf2dac-trmdf1dbc,TimeConstraint->9400];
 ClearAll[trmdf2dac,trmdf1dbc];
 eq4=True===Simplify[dif4==0,TimeConstraint->9400];
 ClearAll[dif4];
 Print["Testing of eq1: ",eq1,"// of eq2: ",eq2,"// of eq3: ",eq3,"// of
eq4: ",eq4];
 If[eq1∧eq2∧eq3∧eq4,Print["Result of full testing: The quaternionic
function ",quatfunctiontested,"
   is H-holomorphic"],Print["Result of full testing: The quaternionic
function ",quatfunctiontested,"
   is not H-holomorphic"]];
 If[eq1,,Print["The equation 1) is not satisfied"]];
 If[eq2,,Print["The equation 2) is not satisfied"]];
 If[eq3,,Print["The equation 3) is not satisfied"]];
 If[eq4,,Print["The equation 4) is not satisfied"]];]
ClearAll[df1,df1c,df2,df2c,trdf1,sftrdf1,trdf1c,sftrdf1c,trdf2,sftrdf2,tr
df2c,sftrdf2c,trmdf1da,trmdf2cdbc,dif1,trmdf2da,trmdf1cdbc,dif2,tr
mdf2db,dif3,trmdf2dac,trmdf1dbc,dif4,eq1,eq2,eq3,eq4];
```

Remark. When considering non-$\mathbb{H}$-holomorphic functions Program 4.2 will indicate, which equations of system (2) are not satisfied.

4.3 Calculating Quaternionic Expressions

The input data for this program are only the functions f1, f2 (also conjugate f1c, f2c). The sequences of launching of input and initial data correspond Scheme 1.

```
(*Program 4.3, Calculating quaternionic expressions*)
ClearAll[f1s,f2s,f1sexp,f2sexp,ref1,imf1,ref2,imf2,ref1tr,imf1tr,ref2tr,imf2tr,iniderf1,iniderf2,derf1r4,derf2r4,derf1s,derf2s,derf1sexp,derf2sexp,rederf1,imderf1,rederf2,imderf2,redf1tr,imdf1tr,redf2tr,imdf2tr,trmf1y0sf,trmf3y0sf,trmf4y0sf,trmcurl,const1,const2,const3,curl1,curl2,curl3,curl1together,curl2together,curl3together,curl1togpowexp,curl2togpowexp,curl3togpowexp,comp1,comp2,comp3,df1dx,df2dy,df3dz,df4du,df1dxtr,df2dytr,df3dztr,df4dutr,df1dxtrsf,trmdf2dytrfullsf,df3dztrsf,df4dutrsf,df1dxtrsfpowexex,df2dytrsfpowexex,df3dztrsfpowexex,df4dutrsfpowexex,vordivfield,divfieldfullsf,divsumminusdf2dy,keydiv];
If[(f1[a,ac,b,bc]===inia1[a,ac,b,bc] inib1[a,ac,b,bc]-inia2[a,ac,b,bc] inib2c[a,ac,b,bc])||mark,Print["////////////////// ERROR:///////////////\n Incorrect data input! Repeat again launching the function in question and then the correct sequence of launching main programmes."],selectpart[vr_,keyreorim_]:=(vrm:=vr/.Complex[0,p_]->M*p;revr:=Select[(vrm//Expand),FreeQ[#,M]&];
  imvr:=(vr-revr)/I//Expand;bg1:=MatchQ[vr,d_.*Complex[0,cc_]];bg2:=MatchQ[vr,(cc_/;NumberQ[cc]&&Im[cc]==0)];bg3:=MatchQ[vr,Complex[xx_/;NumberQ[xx]&&Re[xx]!=0,yy/;NumberQ[yy]]];bg:=bg1||bg2||bg3;If[bg,revr:=Simplify[Re[vr],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800]];If[bg,imvr:=Simplify[Im[vr],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800]];
  If[keyreorim,Return[revr],Return[imvr]]);
 ClearAll[a,ac,b,bc,x,y,z,u];
 inif1=f1[a,ac,b,bc]/.{a->x+I*y,ac->x-I*y,b->z+I*u,bc->z-I*u};
 inif2=f2[a,ac,b,bc]/.{a->x+I*y,ac->x-I*y,b->z+I*u,bc->z-I*u};f1s=Simplify[inif1,{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800];f2s=Simplify[inif2,{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800];f1sexp=Expand[f1s];ref1=Simplify[selectpart[f1sexp,True],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800];imf1=Simplify[selectpart[f1sexp,False],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800];f2sexp=Expand[f2s];ref2=Simplify[selectpart[f2sexp,True],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800];
imf2=Simplify[selectpart[f2sexp,False],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->9800];
 Print["RESULT 1: Quaternionic function: psi(p)=",quatfunctiontested,"=psi1+psi2*i+psi3*j+psi4*k, where psi1=",ref1,";//psi2=",imf1,";//psi3=",ref2,";//psi4=",imf2];
ClearAll[f1s,f2s,f1sexp,f2sexp];m:=Simplify[Sqrt[ref1^2+imf1^2+ref2^2+imf2^2],{_Symbol∈Reals},TimeConstraint->9800];
 bbig:=Simplify[Cancel[imf1/y],{_Symbol∈Reals},TimeConstraint->9800];
realv:=Sqrt[y^2+z^2+u^2];tangteta:=Simplify[(bbig*realv)/ref1,{_Symbol∈Reals},TimeConstraint->9800];
 Print["RESULT 2 (correct only for H -holomorphic functions) : Polar form of the function
  ",quatfunctiontested," is m[cos(teta)+rsin(teta)], where m=",m,";//teta=Arctan[",tangteta,"]"];
 ClearAll[m,bbig,realv,tangteta];
 ref1tr=ref1/.{y->0};
 imf1tr=imf1/.{y->0};
 ref2tr=ref2/.{y->0};
 imf2tr=imf2/.{y->0};
 ClearAll[ref1,imf1,ref2,imf2];
 Print["RESULT 3: Quaternionic function after transition to 3D (y=0): (psi(p)\[LeftBracketingBar]=phi1+phi2*i+phi3*j+phi4*k, where phi1=",ref1tr,";//phi2=",imf1tr,";//phi3=",ref2tr,";//phi4=",imf2tr];
ClearAll[ref1tr,imf1tr,ref2tr,imf2tr];iniderf1=D[f1[a,ac,b,bc],a]+D[f1[a,ac,b,bc],ac];iniderf2=D[f2[a,ac,b,bc],a]+D[f2[a,ac,b,bc],ac];
 ClearAll[a,ac,b,bc,x,y,z,u];
 derf1r4=iniderf1/.{a->x+I*y,ac->x-I*y,b->z+I*u,bc->z-I*u};
 derf2r4=iniderf2/.{a->x+I*y,ac->x-I*y,b->z+I*u,bc->z-I*u};
ClearAll[iniderf1,iniderf2];derf1s=Simplify[derf1r4,{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->59200];derf2s=Simplify[derf2r4,{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->59200];
 ClearAll[derf1r4,derf2r4];
 derf1sexp=Expand[derf1s];
ClearAll[derf1s];rederf1=Simplify[selectpart[derf1sexp,True],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->59200];imderf1=Simplify[selectpart[derf1sexp,False],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->59200];
 ClearAll[derf1sexp];
derf2sexp=Expand[derf2s];ClearAll[derf2s];rederf2=Simplify[selectpart[derf2sexp,True],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->59200];imderf2=Simplify[selectpart[derf2sexp,False],{_Symbol∈Reals,x>0,y>0,z>0,u>0},TimeConstraint->59200];
 ClearAll[derf2sexp];
 Print["RESULT 4: Quaternionic derivative: d1psi = d1f1+d1f2*j = d1psi1+d1psi2*i+d1psi3*j+d1psi4*k, where d1psi1=",rederf1,";//d1psi2=",imderf1,";//d1psi3=",rederf2,";//d1psi4=",imderf2];
 redf1tr=rederf1/.{y->0};
 imdf1tr=imderf1/.{y->0};
 redf2tr=rederf2/.{y->0};
 imdf2tr=imderf2/.{y->0};
 Print["RESULT 5: Quaternionic field vector: F(p)=fn1+fn2*i+fn3*j+fn4*k, where fn1=d1psi1 =",rederf1,";//fn2=-d1psi2=",-imderf1,";//fn3=-d1psi3=",-rederf2,";//fn4=-d1psi4=",-imderf2];
 Print["RESULT 6: 3D field vector (after y=0):(F(p)\[LeftBracketingBar]=(fn1\[LeftBracketingBar]+(fn2\[LeftBracketingBar]*i+(fn3\[LeftBracketingBar]*j+(fn4\[LeftBracketingBar]*k, where (fn1\[LeftBracketingBar]=(d1psi1\[LeftBracketingBar]=",redf1tr,";//(fn2\[LeftBracketingBar]=-(d1psi2\[LeftBracketingBar]=",-imdf1tr,";//(fn3\[LeftBracketingBar]=-(d1psi3\[LeftBracketingBar]=",-redf2tr,";//(fn4\[LeftBracketingBar]=-(d1psi4\[LeftBracketingBar]=",-imdf2tr];
 trmf1y0sf=PowerExpand[redf1tr]//Expand;
 trmf3y0sf=PowerExpand[-redf2tr]//Expand;
 trmf4y0sf=PowerExpand[-imdf2tr]//Expand;
 ClearAll[redf1tr,redf2tr,imdf2tr];
 trmcurl=Curl[{trmf1y0sf,trmf3y0sf,trmf4y0sf},{x,z,u}];
 ClearAll[trmf1y0sf,trmf3y0sf,trmf4y0sf];
 const1=trmcurl[[1]];
 curl1=Simplify[const1,TimeConstraint->59200];
 ClearAll[const1];
 const2=trmcurl[[2]];
 curl2=Simplify[const2,TimeConstraint->59200];
 ClearAll[const2];
 const3=trmcurl[[3]];
 curl3=Simplify[const3,TimeConstraint->59200];
 ClearAll[const3];
 curl1together=Together[curl1];
 curl1togpowexp=PowerExpand[curl1together];
 ClearAll[curl1,curl1together];
 curl2together=Together[curl2];
 curl2togpowexp=PowerExpand[curl2together];
 ClearAll[curl2,curl2together];
 curl3together=Together[curl3];
 curl3togpowexp=PowerExpand[curl3together];
 ClearAll[curl3,curl3together];
 comp1=True===Simplify[curl1togpowexp==0,TimeConstraint->59200];
ClearAll[curl1togpowexp];comp2=True===Simplify[curl2togpowexp==0,TimeConstraint->59200];
ClearAll[curl2togpowexp];comp3=True===Simplify[curl3togpowexp==0,TimeConstraint->59200];ClearAll[curl3togpowexp];If[comp1∧comp2∧comp3,Print["RESULT 7: For quaternionic function psi(p)= ",quatfunctiontested," the vortex density curl(F(p)\[LeftBracketingBar]=0. The field (F(p)\[LeftBracketingBar] is potential."],Print["RESULT 7: For quaternionic function psi(p)= ",quatfunctiontested," the vortex density curl(F(p)\[LeftBracketingBar]!= 0. The field (F(p)\[LeftBracketingBar] is not potential."]];
 Print["\n The following part of program calculates the divergence div(F(p)\[LeftBracketingBar] and tests the equality div(F(p)\[LeftBracketingBar]=(dfn2/dy\[LeftBracketingBar]. For some complicated H-holomorphic functions this procedure can take a long time and RESULT 8 long time does not appear. If you do not need it, you can abort this program and use the obtained RESULTS 1-7.\n"];
 ClearAll[comp1,comp2,comp3];
 df1dx=D[rederf1,x];
 df2dy=D[-imderf1,y];
 df3dz=D[-rederf2,z];
 df4du=D[-imderf2,u];
 df1dxtr=df1dx/.{y->0};
```

```
df2dytr=df2dy/.{y->0};
df3dztr=df3dz/.{y->0};
df4dutr=df4du/.{y->0};
ClearAll[df1dx,df2dy,df3dz,df4du];
trmdf2dytrfullsf=Simplify[df2dytr,TimeConstraint->59200];
df1dxtrsf=Simplify[df1dxtr,TimeConstraint->59200];
df3dztrsf=Simplify[df3dztr,TimeConstraint->59200];
df4dutrsf=Simplify[df4dutr,TimeConstraint->59200];
ClearAll[df1dxtr,df3dztr,df4dutr];
df1dxtrsfpowexex=PowerExpand[df1dxtrsf]//Expand;
ClearAll[df1dxtrsf];
df2dytrsfpowexex=PowerExpand[df2dytr]//Expand;
ClearAll[df2dytr];
df3dztrsfpowexex=PowerExpand[df3dztrsf]//Expand;
ClearAll[df3dztrsf];
df4dutrsfpowexex=PowerExpand[df4dutrsf]//Expand;
ClearAll[df4dutrsf];vordivfield=df1dxtrsfpowexex+df3dztrsfpowexex+df4dutrsfpowexex;
divfieldfullsf=Simplify[vordivfield,TimeConstraint->59200];
divsumminusdf2dy=Simplify[vordivfield-df2dytrsfpowexex,TimeConstraint->59200];ClearAll[df1dxtrsfpowexex,df3dztrsfpowexex,df4dutrsfpowexex,vordivfield];keydiv=True===Simplify[divsumminusdf2dy==0,TimeConstraint->59200];
If[keydiv,Print["RESULT 8: For quaternionic function psi(p)= ",quatfunctiontested," the divergence div(F(p)\[LeftBracketingBar] is equal to ",divfieldfullsf," and equal to (dfn2/dy\[LeftBracketingBar]= ",trmdf2dytrfullsf,". The derivative (dfn2/dy\[LeftBracketingBar] represents the density of sources and sinks in 3D space."],Print["RESULT 8: For quaternionic function psi(p)= ",quatfunctiontested," the divergence div(F(p)\[LeftBracketingBar]is equal to ",divfieldfullsf," and not equal to (dfs2/dy\[LeftBracketingBar]=",trmdf2dytrfullsf]];]
```

4.4 Calculating Higher Derivatives

The input data for this program are only the functions f1, f2 (also conjugate f1c, f2c). The sequences of launching of input and initial data correspond Scheme 1. The k'th derivative is calculated after k'th launching this program. After obtaining the components f1, f2 (also conjugate f1c, f2c) of the higher derivative in question, we launch Program 4.2 to verify the $\mathbb{H}$ - holomorphy of this derivative and/or Program 4.3 to calculate the quaternionic expressions for this derivative. The launching of Program 4.5 obtains the natural logarithm of the derivative in question.

```
(*Program 4.4, Calculating higher derivatives*)
If[(f1[a,ac,b,bc]===inia1[a,ac,b,bc] inib1[a,ac,b,bc]-inia2[a,ac,b,bc] inib2c[a,ac,b,bc])||mark,Print["////////////////// ERROR:////////////////\n Incorrect data input! Repeat again launching the function in question and then the correct sequence of launching the needed main
programmes."],If[s==0,name=ToString[quatfunctiontested,StandardForm]];
 s=s+1;
 s1=ToString[s];
 quatfunctiontested=" the "<>s1<>"'th derivative of "<>name;
 d1f1=D[f1[a,ac,b,bc],a]+D[f1[a,ac,b,bc],ac];
 chad1f1:=d1f1/.{a->kk,ac->ll,b->mm,bc->nn};
 d1f1c:=chad1f1/.{mm->bc,nn->b,ll->a,kk->ac};
 d1f2=D[f2[a,ac,b,bc],a]+D[f2[a,ac,b,bc],ac];
 chad1f2:=d1f2/.{a->kk,ac->ll,b->mm,bc->nn};
 d1f2c:=chad1f2/.{mm->bc,nn->b,ll->a,kk->ac};
 Print["Constituent f1 of",quatfunctiontested," is equal to ",d1f1];
 Clear[f1];
 f1[a_,ac_,b_,bc_]:=d1f1;
 Clear[f1c];
 f1c[a_,ac_,b_,bc_]:=d1f1c;
 Print["Constituent f2 of",quatfunctiontested," is equal to ",d1f2];
 Clear[f2];
 f2[a_,ac_,b_,bc_]:=d1f2;
 Clear[f2c];
 f2c[a_,ac_,b_,bc_]:=d1f2c;]
```

4.5 Calculating natural Logarithms

We consider here the general algorithm and program to calculate the natural logarithms of $\mathbb{H}$ - holomorphic functions. The sequences of launching of input and initial data correspond Scheme 1. To calculate the natural algorithm of some function we launch the cell with the function in question and then launch the program 4.5 below. For example, to calculate the Log $(\sqrt{p})$ we launch the cell 3.1-8) and then program 4.5 below. The repeated launch of Program 4.5 gives the function Log[Log $(\sqrt{p})$] etc. We can combine Programs 4.4 and 4.5 many times and every time launch Program 4.2 and 4.3 to verify the $\mathbb{H}$ - holomorphy and calculate the quaternionic expressions.

```
(*Program 4.5, Calculating the natural logarthms of ψH(p)*)
Clear[inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,bbig,m,logm,psi1,coteta,siteta,cosikv];
isf1=!StringFreeQ[ToString[f1[a,ac,b,bc],StandardForm],"f1"];
If[(f1[a,ac,b,bc]===inia1[a,ac,b,bc] inib1[a,ac,b,bc]-inia2[a,ac,b,bc] inib2c[a,ac,b,bc])||isf1||mark,Print[" ////////////////// ERROR:////////////////\n Incorrect data input (including also the improper function f1c). Repeat again launching the correct\n function in question and then the correct sequence of launching the needed programming codes."];
ClearAll[f1,f1c,f2,f2c];,If[mark==False,name=ToString[quatfunctiontested,StandardForm];argf1=f1[a,ac,b,bc];argf1c=f1c[a,ac,b,bc];argf2=f2[a,ac,b,bc];argf2c=f2c[a,ac,b,bc];quatfunctiontested="Log["<>name<>"]";bbig=Simplify[Cancel[argf2/b],TimeConstraint->9400];m=Simplify[Sqrt[argf1*argf1c+argf2*argf2c],TimeConstraint->9400];logm=Simplify[Log[m],TimeConstraint->9400];psi1=Simplify[(argf1+argf1c)/2,TimeConstraint->9400];coteta=psi1/m;siteta=(bbig*v)/m;cosikv=Simplify[coteta^2+siteta^2];
f1[a_,ac_,b_,bc_]=Simplify[logm+((a-ac)*ArcTan[(bbig*v)/psi1])/(2*v),TimeConstraint->9400];f1c[a_,ac_,b_,bc_]=Simplify[logm+((ac-a)*ArcTan[(bbig*v)/psi1])/(2*v),TimeConstraint->9400];
f2[a_,ac_,b_,bc_]=Simplify[(b*ArcTan[(bbig*v)/psi1])/v,TimeConstraint->9400];
f2c[a_,ac_,b_,bc_]=Simplify[(bc*ArcTan[(bbig*v)/psi1])/v];],TimeConstraint->9400];
If[(f1[a,ac,b,bc]===inia1[a,ac,b,bc] inib1[a,ac,b,bc]-inia2[a,ac,b,bc] inib2c[a,ac,b,bc])||isf1||mark,,Print["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"];];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
mark=False;markquot=False;s=0;
```

5. Conclusions

Processing of data for forty-two $\mathbb{H}$ - holomorphic functions has shown, that the main properties of the class of $\mathbb{H}$ - holomorphic functions introduced in [1] are wholly confirmed:

1) the quaternionic functions constructed from complex holomorphic functions by replacing an independent complex variable as a whole by a quaternionic one in the expressions for the complex holomorphic functions (see rule of constructing 2.1 in [1]) are $\mathbb{H}$ - holomorphic,

2) the quaternionic multiplication of $\mathbb{H}$ - holomorphic functions behaves as commutative,

3) the right quaternionic quotient of two $\mathbb{H}$ - holomorphic functions equals the left one,

4) each 3D field $(F(x,z,u)|$ represented by some $\mathbb{H}$ - holomorphic function is potential, i. e.

$$curl\ (F(x,z,u)| = 0,$$

5) for the divergence of each 3D field represented by some $\mathbb{H}$ - holomorphic function holds true

$$div\ (F(x,z,u)| = (\frac{\partial f2}{\partial y}|,$$

where $(\frac{\partial f2}{\partial y}|$ represents a density of flow (field) sources and sinks in 3D space [1, 2].

6) the full quaternionic derivatives of all orders of $\mathbb{H}$ - holomorphic functions are $\mathbb{H}$ - holomorphic as well.

$\mathbb{H}$-holomorphy testing of higher derivatives (the 4'th, 5'th and more orders) of the $\mathbb{H}$ - holomorphic functions can sometines take long time. The rules for quaternionic differentiation established in [1] are best used in these cases (see examples of the use: 1) and 2) in sub-section 3.3.1). Programs for calculating inverse trigonometric and hyperbolic $\mathbb{H}$ - holomorphic functions will be considered in the following papers.

References

[1] Michael Parfenov, "Essentially Adequate Concept of Holomorphic Functions in Quaternionic Analysis." *American Journal of Mathematical Analysis*, vol. **8**, no. 1 (2020): 14-30. DOI: 10.12691/ajma-8-1-3. Available: pubs.sciepub.com/ajma/8/1/3/

[2] Michael Parfenov, "A Quaternionic Potential Conception with Applying to 3D Potential Fields." *American Journal of Mathematical Analysis*, Vol. **7**, no. 1 (2019): 1-10, doi: 10.12691/ajma-7-1-1. Available: pubs.sciepub.com/ajma/7/1/1/

[3] Kantor, I. L., Solodovnikov, A. S.. *Hypercomplex numbers. An Elementary Introduction to Algebras*. Springer-Verlag, 1989

[4] Wellin, P.R., Gaylord, R. J., Kamin, S.N., *An Introduction to Programming with Mathematica*, 3rd ed, Cambridge University Press, New York, 2005.