*American Journal of Mathematical Analysis, 2021, Vol. 9, No. 1, 6-26*
Available online at http://pubs.sciepub.com/ajma/9/1/2
Published by Science and Education Publishing
DOI:10.12691/ajma-9-1-2

# On Constructing Complicated Compositions of Quaternionic Holomorphic Functions

**Michael Parfenov***

Bashkortostan Branch of Russian Academy of Engineering, Ufa, Russia
*Corresponding author: parfenov.48@bk.ru

*Received November 21, 2021; Revised December 24, 2021; Accepted January 04, 2022*

**Abstract** The issue of constructing complicated quaternionic holomorphic ($\mathbb{H}$-holomorphic) functions in the Cayley-Dickson doubling form is considered. The way of $\mathbb{H}$-holomorphic substitutions, allowing us to construct $\mathbb{H}$-holomorphic composite functions of any degree of difficulty, is presented. The new $\mathbb{R}^4$–representation form for $\mathbb{H}$-holomorphic functions is established as a consequence of the earlier proved commutative behavior of the quaternionic multiplication in the case of $\mathbb{H}$-holomorühic functions. The specific polar form of $\mathbb{H}$-holomorphic functions with a real-valued modulus and argument similar to complex one is obtained. The $\mathbb{H}$-holomorphic generalizations of the logarithmic and inverse trigonometric and hyperbolic functions are implemented. The obtained results reaffirm that any complicated $\mathbb{H}$-holomorphic function can be constructed from its complex holomorphic analog. The processing of $\mathbb{H}$-holomorphic functions of any degree of difficulty is provided through high-speed programmes in system Wolfram Mathematica® represented in the Appendix.

***Keywords:*** *quaternionic analysis, quaternionic holomorphic functions, Cauchy-Riemann's equations, compositions of holomorphic functions, polar form, holomorphic logarithms, quaternionic inverse thigonometric and hyperbolic functions*

**Cite This Article:** Michael Parfenov, "On Constructing Complicated Compositions of Quaternionic Holomorphic Functions." *American Journal of Mathematical Analysis*, vol. 9, no. 1 (2021): 6-26. doi: 10.12691/ajma-9-1-2.

## 1. Introduction

There are known difficulties in different theories of quaternionic analysis (see, e.g., references in [5]) when constructing quaternionic holomorphic analogs of complex holomorphic functions. However, the so-called essentially adequate concept of quaternionic holomorphy overcomes successfully these difficulties [1,5].

It is now important to show that this concept enables us to construct not only simple quaternionic holomorphic functions, but also *all complicated compositions* of them that must be holomorphic as a result as well.

The main principles of this concept are shortly reviewed in the introduction to article [1]. We recall some notions of the essentially adequate concept needed in the sequel.

We denote an independent quaternionic variable by

$$p = x + yi + zj + uk = a + bj \in \mathbb{H}, \tag{1.1}$$

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

$$a = x + yi, \tag{1.2}$$

$$b = z + ui \tag{1.3}$$

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

The quaternionic functions are, respectively, denoted by

$$\begin{aligned}\psi(p) &= \psi_1(x,y,z,u) + \psi_2(x,y,z,u)i \\ &+ \psi_3(x,y,z,u)j + \psi_4(x,y,z,u)k\end{aligned} \tag{1.4}$$

where $\psi_1(x,y,z,u)$, $\psi_2(x,y,z,u)$, $\psi_3(x,y,z,u)$ and $\psi_4(x,y,z,u)$ are real-valued functions of real variables $x, y, z, u$ (the so-called $\mathbb{R}^4$ -representation [3]: an explicit dependence on only $x, y, z, u$).

In the Cayley-Dickson doubling form we use the notation

$$\psi(p) = \phi_1(a, b, \bar{a}, \bar{b}) + \phi_2(a, b, \bar{a}, \bar{b}) \cdot j, \tag{1.5}$$

where

$$\phi_1(a, b, \bar{a}, \bar{b}) = \psi_1(x, y, z, u) + \psi_2(x, y, z, u)i \tag{1.6}$$

$$\phi_2(a, b, \bar{a}, \bar{b}) = \psi_3(x, y, z, u) + \psi_4(x, y, z, u)i,$$

and by "·" is denoted the quaternionic multiplication.

For simplicity, we will also denote (in $\mathbb{C}^2$ -representation [3]: an explicit dependence only on $a, b, \bar{a}, \bar{b}$) the functions $\phi_1(a, b, \bar{a}, \bar{b})$ and $\phi_2(a, b, \bar{a}, \bar{b})$ by $\phi_1(a, b)$ and $\phi_2(a, b)$ or simpler by $\phi_1$ and $\phi_2$. We can also denote the functions $\psi_i(x, y, z, u)$, $i = 1,2.3,4$, by $\psi_i$ ($\mathbb{R}^4$ –representation).

Upon the transition to the complex plane $x+zj$ denoted by $\xi$ the Cayley-Dickson doubling form becomes the following complex expression with imaginary unit $j$:

$$\psi_C(\xi)=\psi_1(x,z)+\psi_3(x,z)\cdot j.$$

The holomorphic functions in the essentially adequate concept of $\mathbb{H}$-holomorphy are defined as the quaternionic functions, whose quaternionic derivatives are *independent of the so-called "way of their computation"* [1], i.e. independent not only of directions how $\Delta p$ approaches zero when defining a quaternionic derivative as a limit of a quaternionic difference quotient $\Delta\psi/\Delta p$, but also of the manner of quaternionic division of $\Delta\psi$ by $\Delta p$: on the left or on the right.

Strictly speaking, we have the following

**Definition 1.1** *A quaternionic function $\psi(p)$ is said to be an essentially adequate quaternionic holomorphic function ($\mathbb{EAH}$- holomorphic or, briefly, $\mathbb{H}$-holomorphic) at a point $\in\mathbb{H}$, if it has a quaternionic derivative independent of a way of its computation in some open connected neighborhood $G_4\in\mathbb{H}$ of a point $p$.*

It was established in [5] that the component $\phi_2(a,b,\overline{a},\overline{b})$ of any $\mathbb{H}$-holomorphic function can be expressed as follows:

$$\phi_2(a,b,\overline{a},\overline{b})=B(a,b,\overline{a},\overline{b})\cdot b, \qquad (1.7)$$

where $B(a,b,\overline{a},\overline{b})$ has a symmetric (in variables $a$, $\overline{a}$, and $b,\overline{b}$) form that is invariant under complex conjugation:

$$B(a,b,\overline{a},\overline{b})=\overline{B(a,b,\overline{a},\overline{b})}.$$

In other words, the function $B(a,b,\overline{a},\overline{b})$ is real-valued.

Definition 1.1 leads to the following definition of the necessary and sufficient conditions for $\psi(p)$ to be $\mathbb{H}$-holomorphic [1].

**Definition 1.2** *Suppose that the constituents $\phi_1(a,b,\overline{a},\overline{b},)$ and $\phi_2(a,b,\overline{a},\overline{b})$ of a quaternionic function $\psi(p)=\psi(a,b)=\phi_1(a,b,\overline{a},\overline{b},)+\phi_2(a,b,\overline{a},\overline{b})\cdot j$ possess continuous first-order partial derivatives with respect to $a,\overline{a},b$, and $\overline{b}$ in some open connected neighborhood $G_4\in\mathbb{H}$ of a point $p\in\mathbb{H}$. Then a function $\psi(p)$ is $\mathbb{H}$-holomorphic (further is denoted by $\psi_H(p)$) at a point $p$ if and only if the functions $\phi_1(a,b,\overline{a},\overline{b})$ and $\phi_2(a,b,\overline{a},\overline{b})$ satisfy in $G_4$ the following quaternionic generalization of complex Cauchy-Riemann's equations:*

$$\begin{cases} 1)(\partial_a\phi_1|=(\partial_{\overline{b}}\overline{\phi}_2|,\ 2)(\partial_a\phi_2|=-(\partial_{\overline{b}}\overline{\phi}_1|, \\ 3)(\partial_a\phi_1|=(\partial_b\phi_2|,\ 4)(\partial_{\overline{a}}\phi_2|=-(\partial_{\overline{b}}\phi_1|. \end{cases} \qquad (1.8)$$

Here $\partial_i$, $i=a,\overline{a},b,\overline{b}$, 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, dealing with the brackets $(..|$ in (1.8), we, first, calculate the partial derivatives of the functions $\phi_1,\overline{\phi}_1,\phi_2,\overline{\phi}_2$ with respect to variables $a,\overline{a},b,\overline{b}$; second, we carry out the transition $a=\overline{a}=x$ in them; third, we test equations (1.8).

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. *Any quaternionic function remains the same 4-dimensional quaternionic function regardless of whether we check its holomorphy or not.* This transition is needed only to check $\mathbb{H}$- holomorphy of any quaternionic function. It is also used when solving the 3D tasks [4]. For details we refer to [1,4].

We recall the following theorem, which allows us to get a $\mathbb{H}$-holomorphic function directly from its $\mathbb{C}$-holomorphic counterpart.

**Theorem 1.3.** *Let a complex function $\psi_C(\xi):G_2\to\mathbb{C}$ be complex holomorphic everywhere in a connected open set $G_2\subseteq\mathbb{C}$, except, possibly, at certain singularities. Then a $\mathbb{H}$-holomorphic function $\psi_H(p)$ of the same kind as $\psi_C(\xi)$ can be constructed (without change of a functional dependence form) from $\psi_C(\xi)$ by replacing a complex variable $\xi\in G_2$ (as a single whole) in an expression for $\psi_C(\xi)$ by a quaternionic variable $p\in G_4\subseteq\mathbb{H}$, where $G_4$ is defined (except, possibly, at certain singularities) by the relation $G_4\supset G_2$ in the sense that $G_2$ exactly follows from $G_4$ upon the transition from $p$ to $\xi$.*

This theorem allows us to construct each $\mathbb{H}$-holomorphic functions $\psi_H(p)$ from its complex holomorphic ($\mathbb{C}$-holomorphic) analog $\psi_C(\xi)$ by simple replacing a complex variable $\boldsymbol{\xi}$ by a quaternionic variable $p$ *without violating of a functional dependence form $\psi$ (function kind).*

Technically, to obtain any $\mathbb{H}$-holomorphic function, it is sufficient to represent its $\mathbb{C}$-holomorphic analog as a function depending only on complex variable $\xi$ (as a single whole), and then to replace $\xi$ by $p$ in this analog.

So, for example, we can obtain the corresponding $\mathbb{H}$-holomorphic function $\psi_H(p)=\mathrm{Log}\sqrt{\sin(\exp(p))}$ from its $\mathbb{C}$-holomorphic analog $\psi_C(\xi)=\mathrm{Log}\sqrt{\sin(\exp(\xi))}$. However, in such general expressions there are no components $\phi_1$ and $\phi_2$ in an explicit form. Therefore we further need to represent such a general expression in the Cayley-Dickson doubling form $\psi_H(p)=\phi_1(a,b)+\phi_2(a,b)j$, whence we can extract the components $\phi_1$ and $\phi_2$, required to use equations (1.8) and the theory as a whole.

In cases of uncomplicated functions such as $p$, $\frac{1}{p}$, $\sqrt{p}$, $e^p$, $\log(p)$, $\sin(p)$, $\cos(p)$ etc. and simple compositions of them [6] we can easy do it just after replacing $\xi$ by $p$ (see "Constructing rule 2.1" in [1]). However, it is very difficult for many complicated $\mathbb{H}$-holomorphic compositions of these functions to extract with confidence the functions $\phi_1$ and $\phi_2$, which must immediately satisfy equations (1.8).

*The aim of this article is to develop some general ways of constructing complicated $\mathbb{H}$-holomorphic functions in the Cayley-Dickson doubling form in order to determine their components $\phi_1$ and $\phi_2$.*

Throughout this article we assume that $\mathbb{H}$-holomorphic functions and their derivatives are defined in the simply-connected open domains $G_4\subset\mathbb{H}$. We do not point concrete domains of functions definition in this article, so that we do not clutter this article and thus further

complicate our task. In each case in question they can be specified individually.

*As complicated functions we here regard such functions whose expressions are so large and give such a high risk of errors that it is better to employ the computer software*.

Given a practical impossibility of manual processing of complicated $\mathbb{H}$ -holomorphic functions without using computer programmes, we have to represent such programmes in the necessary quantities in the Appendix.

It should be mentioned that we do not study here singularities of $\mathbb{H}$-holomorphic functions, since they can be studied as singularities of complex components $\phi_1$ and $\phi_2$. It should be also noted that the "·" sign of quaternionic multiplication is often omitted if it is clear from the context that values multiplied are quaternionic.

## 2. Substitutions Way for Complicated $\mathbb{H}$ -holomorphic Compositions

We present here the natural method of $\mathbb{H}$-holomorphic substitutions that allows us to construct composite $\mathbb{H}$-holomorphic functions of any degree of difficulty.

**Assertion 2.1.** *A $\mathbb{H}$ -holomorphic function remains $\mathbb{H}$-holomorphic when replacing its argument by another $\mathbb{H}$-holomorphic function.*

**Proof.** Consider two $\mathbb{H}$- holomorphic functions:

$$\begin{aligned} f(p) &= f(a+b\cdot j) \\ &= f_1(a,b,\bar{a},\bar{b}) + f_2(a,b,\bar{a},\bar{b})\cdot j \end{aligned} \qquad (2.1)$$

$$\begin{aligned} g(p) &= g(a+b\cdot j) \\ &= g_1(a,b,\bar{a},\bar{b}) + g_2(a,b,\bar{a},\bar{b})\cdot j \end{aligned}$$

A replacement of the argument $a+b\cdot j$ of the function $f(p)=f(a+b\cdot j)$ by the function $g(p)$ can be only exercised as replacement of the "real" part $a$ of the argument of $f(p)$ by the "real" part $g_1(a,b,\bar{a},\bar{b})$ of the function $g(p)$ and the "imaginery" part $b$ of the argument of $f(p)$ by the "imaginery" part $g_2(a,b,\bar{a},\bar{b})$ of the function $g(p)$. With this in mind, we now consider the composition $f(g(p))$ of these two $\mathbb{H}$ - holomorphic functions.

As a base function we define the function $f(p)$, in which we carry out substitutions. Such functions examples are considered in the subsection A.1 of the Appendix to this article. We further assume that $f(p)$ is $\mathbb{H}$-holomorphic in a domain that contains a range of $g(p)$.

Introducing the following substitutions:

$$\begin{aligned} ta &= g_1(a,b,\bar{a},\bar{b}),\ tb = g_2(a,b,\bar{a},\bar{b}) \\ t\bar{a} &= \overline{g_1(a,b,\bar{a},\bar{b})},\ t\bar{b} = \overline{g_2(a,b,\bar{a},\bar{b})} \end{aligned} \qquad (2.2)$$

into base function (2.1): $ta$ instead of $a$ and $tb$ instead of $b$ (correspondingly, $t\bar{a}$ instead of $\bar{a}$ and $t\bar{b}$ instead of $\bar{b}$), we obtain the expression for the composite function $f(g(p))$ as follows:

$$\begin{aligned} f(p) &= f(ta+tb\cdot j) \\ &= f_1(ta,tb,t\bar{a},t\bar{b}) + f_2(ta,tb,t\bar{a},t\bar{b})\cdot j, \end{aligned}$$

where $f_1(ta,tb,t\bar{a},t\bar{b})$ and $f_2(ta,tb,t\bar{a},t\bar{b})$ are desired components of the composite function $f(g(p))$ got by the substitutions (2.2).

In complex analysis ([9], p.85) we get a $\mathbb{C}$-holomorphic composite function $f(g(\xi))$ if complex functions $f(\xi)$ and $g(\xi)$ are both $\mathbb{C}$-holomorphic. Then by virtue of Theorem 1.3 we get the $\mathbb{H}$-holomorphic composite function $f(g(p))$ from $f(g(\xi))$ by replacing the complex variable $\xi$ by the quaternionic variable $p$ without change of a functional dependence forms ($f$ and $g$). At that, by virtue of the same theorem, the function $g(p)$ is $\mathbb{H}$-holomorphic as well. Thus, if we will use the $\mathbb{H}$ - holomorphic function $g(p)$ as a substitution into $f(p)$, then we shall get immediately the $\mathbb{H}$-holomorphic function $f(g(p))$. Q.e.d.

Note that the substitution of the functions $g_1(a,b,\bar{a},\bar{b})$, $\overline{g_1(a,b,\bar{a},\bar{b})}$, $g_2(a,b,\bar{a},\bar{b})$, $\overline{g_2(a,b,\bar{a},\bar{b})}$ instead of the variables $a,\bar{a},b,\bar{b}$, respectively, does not change the function kind $f$. In other words, the character of actions performed by the functions $f_1$ and $f_2$ with their variables is independent of variables.

Consistently, step by step using this assertion, we can construct composite functions of any degree of difficulty.

Assertion 2.1 does not prohibit using previously obtained complicated $\mathbb{H}$-holomorphic composite functions as the functions $g(p)$. In this case constructing $\mathbb{H}$-holomorphic functions can be significantly simplified. A variety of different composite functions, which can be used as the functions $g(p)$, is represented in preprint [6].

**Example 2.2.** We consider the function $f(g(p))=p+\sqrt{p^2-1}$. In this case we have the base $\mathbb{H}$-holomorphic function $f(p)=\sqrt{p}$.

Then the base function $\sqrt{p}$ is the following:

$$f(p)=\sqrt{p}=f_1(a,b,\bar{a},\bar{b})+f_2(a,b,\bar{a},\bar{b})\cdot j,$$

where

$$\begin{aligned} &f_1(a,b,\bar{a},\bar{b}) \\ &= \frac{a-\bar{a}}{2\sqrt{2}\sqrt{\frac{a+\bar{a}}{2}+\sqrt{a\bar{a}+b\bar{b}}}} + \frac{\sqrt{\frac{a+\bar{a}}{2}+\sqrt{a\bar{a}+b\bar{b}}}}{\sqrt{2}} \end{aligned} \qquad (2.3)$$

$$f_2(a,b,\bar{a},\bar{b}) = \frac{b}{\sqrt{2}\sqrt{\frac{a+\bar{a}}{2}+\sqrt{a\bar{a}+b\bar{b}}}}. \qquad (2.4)$$

Since we represent all employed base functions in the subsection A1 "Base functions examples" of the Appendix to this article, we can directly obtain these expressions by launching (or evaluating [7]) the data cell A1.9. At that, as indicated in Appendix, we need to copy the data cell A1.9 into an empty file .nb (into separate cell) of the computing system Wolfram Mathematica® [7] and then launch it there. Speaking of launching any data cell or processing program, we always mean these actions.

In this example the function $g(p)$ is the following:

$$g(p)=p^2-1=a^2-b\bar{b}-1+(a+\bar{a})b\cdot j.$$

Using the corresponding substitutions into $\sqrt{p}$ :

$$ta = a^2 - b\bar{b} - 1, tb = (a+\bar{a})b,$$
$$t\bar{a} = \bar{a}^2 - b\bar{b} - 1, t\bar{b} = (a+\bar{a})\bar{b}$$

we obtain the expressions for the complicated function $f(p) = p + \sqrt{p^2 - 1}$ as follows:

$$f_1\left(ta, tb, \overline{ta}, \overline{tb}\right) = a + \frac{ta - t\bar{a}}{2\sqrt{2}\sqrt{\frac{ta + t\bar{a}}{2} + \sqrt{tat\bar{a} + tbt\bar{b}}}}$$
$$+ \frac{\sqrt{\frac{ta + \overline{ta}}{2} + \sqrt{tat\bar{a} + tbt\bar{b}}}}{\sqrt{2}}$$

$$f_2\left(ta, tb, t\bar{a}, \overline{tb}\right) = b + \frac{tb}{\sqrt{2}\sqrt{\frac{ta + t\bar{a}}{2} + \sqrt{tat\bar{a} + tbt\bar{b}}}}.$$

We can get the long deployed expressions for these functions by launching the data cell A2.1.

By the next launching of Program in the cell A3.1 we confirm the $\mathbb{H}$- holomorphy of the function $f(g(p)) = p + \sqrt{p^2 - 1}$.

**Example 2.3.** Now we consider the following complicated composition of four $\mathbb{H}$-holomorphic functions:

$$f\left(g_3(g_2(g_1(p)))\right) = \sin(\frac{1}{e^{p^3}}),$$

where

$$g_1(p) = p^3,\ g_2(p) = e^{p^3},\ g_3(p) = \frac{1}{g_2},\ f(p) = \sin(g_3).$$

We have to use Assertion 2.1 three times, performing the substitutions step by step, to get the functions: $g_2(p)$, $g_3(p)$, $f(p)$.

1) *The function* $g_2(p)$. By launching the data cell A1.4. from the Appendix (further we will write shortly A1.4.), we get the function $g_1(p) = p^3$ as follows:

$$g_1(p) = p^3 = g_{1(1)}\left(a, b, \bar{a}, \bar{b}\right) + g_{2(1)}\left(a, b, \bar{a}, \bar{b}\right) \cdot j,$$

where

$$g_{1(1)}\left(a, b, \bar{a}, \bar{b}\right) = a^3 - (2a + \bar{a})b\bar{b},$$

$$g_{2(1)}\left(a, b, \bar{a}, \bar{b}\right) = (a^2 + a\bar{a} + \bar{a}^2)b - b^2\bar{b}.$$

We also have the following conjugate functions:

$$\overline{g_{1(1)}\left(a, b, \bar{a}, \bar{b}\right)} = \bar{a}^3 - (2\bar{a} + a)b\bar{b},$$

$$\overline{g_{2(1)}\left(a, b, \bar{a}, \bar{b}\right)} = \left(\bar{a}^2 + a\bar{a} + a^2\right)\bar{b} - \bar{b}^2 b$$

The first substitutions are the following:

$$ta = g_{1(1)} = a^3 - (2a + \bar{a})b\bar{b} \qquad (2.5)$$

$$t\bar{a} = \overline{g_{1(1)}} = \bar{a}^3 - \left(2\bar{a} + a\right)b\bar{b} \qquad (2.6)$$

$$tb = g_{2(1)} = (a^2 + a\bar{a} + \bar{a}^2)b - b^2\bar{b}, \qquad (2.7)$$

$$t\bar{b} = \overline{g_{2(1)}} = \left(\bar{a}^2 + a\bar{a} + a^2\right)\bar{b} - \bar{b}^2 b \qquad (2.8)$$

By launching the data cell A1.8 we get the base function $g_2(p) = e^p$ as follows:

$$g_{2(\text{base})}(p) = e^p = g_{1(\text{base})} + g_{2(\text{base})} \cdot j,$$

where

$$g_{1(\text{base})} = 2\beta\cos(v) + \frac{(a - \bar{a})\beta\sin(v)}{v}, \qquad (2.9)$$

$$g_{2(\text{base})} = \frac{2b\beta\sin(v)}{v} \qquad (2.10)$$

as well as (see [1] or data cell A1.1.)

$$\beta = \frac{1}{2}e^{\frac{a+\bar{a}}{2}} = \bar{\beta}, \qquad (2.11)$$

$$v = \sqrt{y^2 + z^2 + u^2} = \frac{\sqrt{4\left(a\bar{a} + b\bar{b}\right) - (a + \bar{a})^2}}{2} = \bar{v}. \qquad (2.12)$$

Substituting expressions (2.5), (2.6), (2.7), and (2.8), instead of $a$, $\bar{a}$, $b$, and $\bar{b}$, respectively, into expressions (2.9), (2.10), (2.11) and (2.12), we obtain the following expressions for the function $e^{p^3}$:

$$g_2(p) = e^{p^3} = g_{1(2)} + g_{2(2)} \cdot j,$$

where

$$g_{1(2)} = 2t\beta \cdot \cos(tv) + \frac{(ta - t\bar{a}) \cdot t\beta \cdot \sin(tv)}{tv},$$

$$g_{2(2)} = \frac{2 \cdot tb \cdot t\beta \cdot \sin(tv)}{tv},$$

$$t\beta = \frac{1}{2}e^{\frac{ta+t\bar{a}}{2}} = \overline{t\beta}, \qquad (2.13)$$

$$tv = \frac{\sqrt{4\left(ta \cdot t\bar{a} + tb \cdot t\bar{b}\right) - (ta + t\bar{a})^2}}{2} = \overline{tv}. \qquad (2.14)$$

2) *The function* $g_3(p)$. By launching the data cell A1.5 we get the base function in the case of $g_3(p)$ as follows:

$$g_{3(\text{base})}(p) = \frac{1}{p} = g_{1(\text{base})} + g_{2(\text{base})} \cdot j,$$

where

$$g_{1(\text{base})}\left(a, b, \bar{a}, \bar{b}\right) = \frac{\bar{a}}{a\bar{a} + b\bar{b}}, \qquad (2.15)$$

$$\overline{g_{1(\text{base})}}\left(a, b, \bar{a}, \bar{b}\right) = \frac{a}{a\bar{a} + b\bar{b}}, \qquad (2.16)$$

$$g_{2(\text{base})}\left(a, b, \bar{a}, \bar{b}\right) = -\frac{b}{a\bar{a} + b\bar{b}}, \qquad (2.17)$$

$$\overline{g_{2(\text{base})}}\left(a,b,\bar{a},\bar{b}\right)=-\frac{\bar{b}}{a\bar{a}+b\bar{b}}. \tag{2.18}$$

To substitute the function $g_2(p)=e^{p^3}$ into the function $g_3(p)$ we employ the following substitutions:

$$t2a=g_{1(2)}=2t\beta\cdot\cos(tv)+\frac{(ta-t\bar{a})\cdot t\beta\cdot\sin(tv)}{tv} \tag{2.19}$$

$$t2\bar{a}=\overline{t2a}=2t\beta\cdot\cos(tv)+\frac{(t\bar{a}-ta)\cdot t\beta\cdot\sin(tv)}{tv}, \tag{2.20}$$

$$t2b=g_{2(2)}=\frac{2tb\cdot t\beta\cdot\sin(tv)}{tv}, \tag{2.21}$$

$$t2\bar{b}=\overline{t2b}=\frac{2t\bar{b}\cdot t\beta\cdot\sin(tv)}{tv}, \tag{2.22}$$

where $t\beta$ and $tv$ are determined by (2.13) and (2.14).

Substituting (2.19), (2.20), (2.21), (2.22) instead of $a$, $\bar{a}$, $b$, $\bar{b}$, respectively, into (2.15, (2.16), (2.17), (2.18) we get the following expression for the function $g_3(p)=\frac{1}{e^{p^3}}$:

$$g_3(p)=g_{1(3)}\left(a,b,\bar{a},\bar{b}\right)+g_{2(3)}\left(a,b,\bar{a},\bar{b}\right)\cdot j,$$

where

$$g_{1(3)}\left(a,b,\bar{a},\bar{b}\right)=\frac{t2\bar{a}}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}},$$

$$g_{2(3)}\left(a,b,\bar{a},\bar{b}\right)=-\frac{t2b}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}},$$

$$\overline{g_{1(3)}\left(a,b,\bar{a},\bar{b}\right)}=\frac{t2a}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}},$$

$$\overline{g_{2(3)}\left(a,b,\bar{a},\bar{b}\right)}=-\frac{t2\bar{b}}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}}.$$

3) The function $f(p)=\sin\left(g_3(p)\right)=\sin(\frac{1}{e^{p^3}})$.

By launching the data cell A1.6 we get the base function $f_{(\text{base})}(p)=\sin p$ as follows:

$$f_{(\text{base})}(p)=f_{1(\text{base})}\left(a,b,\bar{a},\bar{b}\right)+f_{2(\text{base})}\left(a,b,\bar{a},\bar{b}\right)\cdot j,$$

where

$$f_{1(\text{base})}\left(a,b,\bar{a},\bar{b}\right)=-\frac{(a-\bar{a})\left(e^{-v}-e^{v}\right)\cdot\cos\left[\frac{a+\bar{a}}{2}\right]}{4v}+\frac{\left(e^{-v}+e^{v}\right)\cdot\sin\left[\frac{a+\bar{a}}{2}\right]}{2}, \tag{2.23}$$

$$f_{2(\text{base})}\left(a,b,\bar{a},\bar{b}\right)=-\frac{b\cdot\left(e^{-v}-e^{v}\right)\cdot\cos\left[\frac{a+\bar{a}}{2}\right]}{2v}, \tag{2.24}$$

The third substitution to get the $f(p)=\sin(\frac{1}{e^{p^3}})$ is the following:

$$t3a=g_{1(3)}=\frac{t2\bar{a}}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}},$$

$$t3b=g_{2(3)}=-\frac{t2b}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}},$$

$$t3\bar{a}=\overline{g_{1(3)}}=\frac{t2a}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}},$$

$$t3\bar{b}=\overline{g_{2(3)}}=-\frac{t2\bar{b}}{t2a\cdot t2\bar{a}+t2b\cdot t2\bar{b}}.$$

Substituting them into (2.23) and (2.24) as well as into (2.11) and (2.12): $t3a$ instead of $a$, $t3b$ instead of $b$, $t3\bar{a}$ instead of $\bar{a}$, and $t3\bar{b}$ instead of $\bar{b}$, we get the required composition $f(p)=\sin(\frac{1}{e^{p^3}})$ as follows:

$$\sin(\frac{1}{e^{p^3}})=f_1\left(a,b,\bar{a},\bar{b}\right)+f_2\left(a,b,\bar{a},\bar{b}\right)\cdot j,$$

where

$$f_1\left(a,b,\bar{a},\bar{b}\right)=-\frac{\left(t3a-t3\bar{a}\right)\left(e^{-t3v}-e^{t3v}\right)\cdot\cos\left[\frac{t3a+t3\bar{a}}{2}\right]}{4\cdot t3v}+\frac{\left(e^{-t3v}+e^{t3v}\right)\cdot\sin\left[\frac{t3a+t3\bar{a}}{2}\right]}{2},$$

$$f_2\left(a,b,\bar{a},\bar{b}\right)=-\frac{t3b\cdot\left(e^{-t3v}-e^{t3v}\right)\cdot\cos\left[\frac{t3a+t3\bar{a}}{2}\right]}{2\cdot t3v},$$

and

$$t3\beta=\frac{1}{2}e^{\frac{t3a+t3\bar{a}}{2}}=\overline{t3\beta},$$

$$t3v=\frac{\sqrt{4\left(t3a\cdot t3\bar{a}+t3b\cdot t3\bar{b}\right)-(t3a+t3\bar{a})^2}}{2}=\overline{t3v}.$$

If we replace the variables $ta$, $t\bar{a}$, $tb$, $t\bar{b}$, $t2a$, $t2\bar{a}$, $t2b$, $t\overline{2b}$, $t3a$, $t3\bar{a}$, $t3b$, $t\overline{3b}$ by their expressions, then we get the deployed expressions for the functions $f_1\left(a,b,\bar{a},\bar{b}\right)$ and $f_2\left(a,b,\bar{a},\bar{b}\right)$, however they are too long to write them here. It is possible to see the fully deployed expressions for $f_1$ and $f_2$ by launching the data cell A.2.2.

The deployed expressions are just those who are always processed by Programmes A3.1 and A3.2 in Appendix. In programming codes they (corresponding to the components $\phi_1$ and $\phi_2$) are denoted by $f1\left(a,b,\bar{a},\bar{b}\right)$ and $f2\left(a,b,\bar{a},\bar{b}\right)$. Further, by launching Program A3.1 we use these functions to verify the $\mathbb{H}$ - holomorphy of the

function $\sin(\frac{1}{e^{p^3}})$. Calculation of the quaternionic expressions for this function in $\mathbb{R}^4$ - representation can be performed by launching the programing cell 3.2.

# 3. Vector Parts, Polar Forms and Logarithmic Functions of Complicated $\mathbb{H}$ -holomorphic Functions

## 3.1. Vector Parts of $\mathbb{H}$-holomorphic Functions

The earlier established properties [1,5] of $\mathbb{H}$-holomorphic functions allows us to establish a specific form of $\mathbb{H}$-holomorphic functions in $\mathbb{R}^4$- representation.

**Theorem 3.1.** *A vector part $\psi_2 i+\psi_3 j+\psi_4 k$ of any $\mathbb{H}$-holomorphic function $\psi_H(p)=\psi_1+\psi_2 i+\psi_3 j+\psi_4 k$, where $\psi_1, \psi_2, \psi_3, \psi_4$ are real-valued functions of the real variables $x, y, z, u$, can be expressed as follows:*

$$yBi+zBj+uBk,$$

*where $B$ is a real-valued multiplicative factor*

$$B=\frac{\phi_2\left(a,b,\overline{a},\overline{b}\right)}{b}=B\left(x,y,z,u\right)$$

*in $\mathbb{R}^4$ – representation.*

**Proof.** The expression for $B(x,y,z,u)$ is obtained from (1.7) and then rewritten in $\mathbb{R}^4$– representation by using (1.2) and (1.3).

The proof will be divided into 2 steps. First (Step 1) we will prove the expression

$$\psi_H\left(p\right)=\psi_1+\psi_2 i+zBj+uBk, \tag{3.1}$$

Then (Step 2) we will prove the final expression

$$\psi_H\left(p\right)=\psi_1+yBi+zBj+uBk.$$

Step 1. Substituting expressions (1.6), (1.7), (1.3) into (1.5) and using the multiplication rule for the quaternionic units: $i\cdot j=k$, we immediately obtain the expression (3.1):

$$\begin{aligned}\psi_H\left(p\right)&=\psi_1+\psi_2 i+B(z+ui)\cdot j\\&=\psi_1+\psi_2 i+zBj+uBk\end{aligned}$$

where $B$ is defined from (1.7) and written in $\mathbb{R}^4$ – representation.

Step 2. It was earlier (see e.g. [4,5]) established that in case of $\mathbb{H}$-holomorphic functions the relation $\psi_2\to 0$ as $y\to 0$ is always performed, whence, without loss of generality, we can represent the function $\psi_2(x,y,z,u)$ in the form $\psi_2=y\alpha(x,y,z,u)=y\alpha$.

Now, every $\mathbb{H}$-holomorphic function can be written as

$$\psi_H\left(p\right)=\psi_1+y\alpha i+zBj+uBk, \tag{3.2}$$

Let

$$\psi_H\left(p\right)'=\psi_1'+y\beta i+zB'j+uB'k \tag{3.3}$$

be another arbitrary $\mathbb{H}$ -holomorphic function in the $\mathbb{R}^4$ –representation (the "′" sign is not to be confused with the derivative sign). The $B$ and $B'$ (as well as $\alpha$ and $\beta$) are different and do not depend on each other.

Let

$$p=x+yi+zj+uk,$$

$$p'=x'+y'i+z'j+u'k$$

be two different arbitrary quaternions. As is known [2], the general quaternionic multiplication rule is the following:

$$\begin{aligned}&p\cdot p'=(x+yi+zj+uk)\cdot(x'+y'i+z'j+u'k)\\&=\left(xx'-yy'-zz'-uu'\right)+\left(xy'+yx'+zu'-uz'\right)i\\&+(xz'+zx'+uy'-yu')j+\left(xu'+ux'+yz'-zy'\right)k\end{aligned} \tag{3.4}$$

Substituting the $\mathbb{H}$-holomorphic functions (3.2) and (3.3) into (3.4) instead of $p$ and $p'$, relatively, we obtain the quaternionic product $\psi_H(p)\cdot\psi_H(p)'$ as follows:

$$\begin{aligned}&\psi_H\left(p\right)\cdot\psi_H\left(p\right)'\\&=(\psi_1+y\alpha i+zBj+uBk)\cdot(\psi_1'+y\beta i+zB'j+uB'k)\\&=\left(\psi_1\psi_1'-y\alpha y\beta-z^2BB'-u^2BB'\right)\\&+\left(\psi_1 y\beta+y\alpha\psi_1'+zBuB'-uBzB'\right)i\\&+(\psi_1 zB'+zB\psi_1'+uBy\beta-y\alpha uB')j\\&+(\psi_1 uB'+uB\psi_1'+y\alpha zB'-zBy\beta)k.\end{aligned} \tag{3.5}$$

By exchanging the order of functions multiplication, we get

$$\begin{aligned}&\psi_H\left(p\right)'\cdot\psi_H\left(p\right)\\&=\left(\psi_1'+y\beta i+zB'j+uB'k\right)\cdot\left(\psi_1+y\alpha i+zBj+uBk\right)\\&=(\psi_1'\psi_1-y\beta y\alpha-zB'zB-uB'uB)\\&+\left(\psi_1'y\alpha+y\beta\psi_1+zB'uB-uB'zB\right)i\\&+\left(\psi_1'zB+zB'\psi_1+uB'y\alpha-y\beta uB\right)j\\&+\left(\psi_1'uB+uB'\psi_1+y\beta zB-zB'y\alpha\right)k.\end{aligned} \tag{3.6}$$

The main property of the $\mathbb{H}$ -holomorphic functions established in [1] is that their quaternionic multiplication behaves as commutative, i.e. $\psi_H(p)\cdot\psi_H(p)'=\psi_H(p)'\cdot\psi_H(p)$. Then from (3.5) and (3.6) the following equalities must be met:

$$\begin{aligned}&\psi_1\psi_1'-y\alpha y\beta-z^2BB'-u^2BB'\\&=\psi_1'\psi_1-y\beta y\alpha-zB'zB-uB'uB,\end{aligned} \tag{3.7}$$

$$\begin{aligned}&\psi_1 y\beta+y\alpha\psi_1'+zBuB'-uBzB'\\&=\psi_1'y\alpha+y\beta\psi_1+zB'uB-uB'zB,\end{aligned} \tag{3.8}$$

$$\begin{aligned}&\psi_1 zB'+zB\psi_1'+uBy\beta-y\alpha uB'\\&=\psi_1'zB+zB'\psi_1+uB'y\alpha-y\beta uB,\end{aligned} \tag{3.9}$$

$$\begin{aligned}&\psi_1 uB^{'}+uB\psi_1^{'}+y\alpha zB^{'}-zBy\beta\\&=\psi_1^{'}uB+uB^{'}\psi_1+y\beta zB-zB^{'}y\alpha.\end{aligned}\tag{3.10}$$

The equality (3.7) is the identity, i.e. is fulfilled.

After simplifyung, equalities (3.8), (3.9), (3.10) become, respectively, the following:

$$\psi_1 y\beta+y\alpha\psi_1^{'}=\psi_1^{'}y\alpha+y\beta\psi_1,\tag{3.11}$$

$$uBy\beta=y\alpha uB^{'},\tag{3.12}$$

$$y\alpha zB^{'}=y\beta zB.\tag{3.13}$$

The equality (3.11) is the identity, i.e. is fulfilled.

Thus, one is left with two equalities (3.12) and (3.13), which must be met by choice of some $\alpha$ and $\beta$. These equalities are fulfilled, if $\alpha=B$ and $\beta=B^{'}$. Then the expressions (3.2) and (3.3) become, respectively, the following:

$$\begin{aligned}&\psi_H(p)=\psi_1+yBi+zBj+uBk,\\&\psi_H(p)^{'}=\psi_1^{'}+yB^{'}i+zB^{'}j+uB^{'}k.\end{aligned}\tag{3.14}$$

Thus, we see that a vector part of any $\mathbb{H}$-holomorphic function $\psi_H(p)$ can be expressed by $yBi+zBj+uBk$. Q.e.d

Remark. The function $\psi_1$ is arbitrary when considering this theorem. This function is determined, in principle, by equations (1.8). It can be extracted for every concrete $\mathbb{H}$-holomorphic function as a real part (see (1.3)) of the function $\phi_1(a,b,\bar{a},\bar{b})$ by launching Program A3.2.

We have established a new form of $\mathbb{R}^4$ –representation inherent only to $\mathbb{H}$-holomorphic functions.

## 3.2. Polar Form of $\mathbb{H}$-holomorphic Functions

There exist some methods to represent quaternions in a polar form. See, for example [8], polar form with a complex modulus and complex argument "inspired by the Cayley-Dickson form". We derive here the polar representation of $\mathbb{H}$-holomorphic function with a real modulus and real argument like complex analysis.

According to the definition of a quaternion modulus [2] (here in the notation as in (1.1)), we have

$$|\mathrm{p}|=\sqrt{x^2+y^2+z^2+u^2}.$$

Correspondingly, we define a modulus of a $\mathbb{H}$-holomorphic function taking into account (3.14) as follows:

$$\begin{aligned}|\psi_H(p)|&=\sqrt{{\psi_1}^2+{\psi_2}^2+{\psi_3}^2+{\psi_4}^2}\\&=\sqrt{{\psi_1}^2+y^2B^2+z^2B^2+u^2B^2}\\&=\sqrt{{\psi_1}^2+(y^2+z^2+u^2)B^2}\geq 0.\end{aligned}$$

Given (3.14), the $\mathbb{H}$-holomorphic function $\psi_H(p)$ can be represented by the following expression:

$$\psi_H(p)=\psi_1+\frac{(yi+zj+uk)}{v}Bv,\tag{3.15}$$

where the real-valued $v$ is determined by (2.12), $v\neq 0$.

Using the quaternionic analog [1]:

$$r=\frac{(yi+zj+uk)}{v},\ (r^2=-1),\tag{3.16}$$

of the complex imaginary unit $i$, we can rewrite (3.15) as follows:

$$\psi_H(p)=\psi_1+rBv,\tag{3.17}$$

Then, *a polar form* (or polar representation) of the $\mathbb{H}$-holomorphic function $\psi_H(p)$, similar to complex one [9], is the following:

$$\psi_H(p)=|\psi_H(p)|\left(\frac{\psi_1}{|\psi_H(p)|}+r\frac{Bv}{|\psi_H(p)|}\right),$$

where, given that the functions $|\psi_H(p)|$, $\psi_1$, $B$, $v$ are real-valued, we interpret the functions $\frac{\psi_1}{|\psi_H(p)|}$ and $\frac{Bv}{|\psi_H(p)|}$ as follows:

$$\cos\theta=\frac{\psi_1}{|\psi_H(p)|},\tag{3.18}$$

$$\sin\theta=\frac{Bv}{|\psi_H(p)|}.\tag{3.19}$$

Here $\theta$ plays a role of a polar angle in, so to speak, "the complex plane, with the imaginary unit $r$", which is isomorphic to usual complex plane with the imaginary unit $i$, since $|\psi_H(p)|, \psi_1, B, v$ are real-valued. We can also speak of polar coordinates $(|\psi_H(p)|,\theta)$ of the $\mathbb{H}$-holomorphic functions. Note that we must always replace the complex imaginary unit $i$ by the quaternionic imaginary unit $r$ when passing from complex case to quaternionic one.

We can finally express *polar form of a $\mathbb{H}$-holomorphic function $\psi_H(p)$* similar to complex one as follows:

$$\psi_H(p)=m(\cos\theta+r\sin\theta),\tag{3.20}$$

where

$$m=|\psi_H(p)|,$$

$\cos\theta$ and $\sin\theta$ are determined by (3.18) and (3.19).

Inequalities $\cos\theta=\frac{\psi_1}{|\psi_H(p)|}\leq 1$ and $\sin\theta=\frac{Bv}{|\psi_H(p)|}\leq 1$ must be met. When processing concrete $\mathbb{H}$-holomorphic functions it is sufficient to check the equality

$$(\cos\theta)^2+(\sin\theta)^2=\left(\frac{\psi_1}{|\psi_H(p)|}\right)^2+\left(\frac{Bv}{|\psi_H(p)|}\right)^2=1.$$

It seems useful to employ, when concrete calculating, the following expression for $\theta$:

$$\theta=\mathrm{Arg}(\psi_H(p))=\arctan\frac{Bv}{\psi_1},\ \psi_1\neq 0.\tag{3.21}$$

**Example 3.2.1.** To illustrate how we get a polar form of a $\mathbb{H}$-holomorphic function we consider the uncomplicated

function $\psi_H(p)=\sqrt{p}$ , which is defined as the quaternionic generalization of the principal branch of the $\mathbb{C}$-holomorphic function $\psi_C(\xi)=\sqrt{\xi}$.

By launching the data cell A1.9 we get the function $\sqrt{p}$ in $\mathbb{C}^2$ –representation:

$$\psi_H(p)=\sqrt{p}=f1+f2\cdot j,$$

where $f1$ and $f2$ are represented above by (2.3) and (2.4). Note that $f1$ and $f2$ are the programming designations for the functions $\phi_1$ and $\phi_2$.

The functions $f1$, $f2$ constitute the input data for Program A3.2. By launching Program A3.2, we obtain (RESULT 1) the following deployed expression for the function $\sqrt{p}$ in $\mathbb{R}^4$–representation:

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

where

$$\psi_1=\frac{\sqrt{x+\sqrt{u^2+x^2+y^2+z^2}}}{\sqrt{2}},$$

$$\psi_2=\frac{y}{\sqrt{2}\sqrt{x+\sqrt{u^2+x^2+y^2+z^2}}},$$

$$\psi_3=\frac{z}{\sqrt{2}\sqrt{x+\sqrt{u^2+x^2+y^2+z^2}}},$$

$$\psi_4=\frac{u}{\sqrt{2}\sqrt{x+\sqrt{u^2+x^2+y^2+z^2}}}.$$

Comparing the last three expressions with (3.14) and (1.4) we get $B$ in $\mathbb{R}^4$–representation as follows:

$$B=\frac{1}{\sqrt{2}\sqrt{x+\sqrt{u^2+x^2+y^2+z^2}}}.$$

The expression for $m$ is the following:

$$\begin{aligned} m&=\left|\psi_H(p)\right| \\ &=\sqrt{{\psi_1}^2+{\psi_2}^2+{\psi_3}^2+{\psi_4}^2} \\ &=\sqrt{\frac{u^2+x^2+y^2+z^2+x\sqrt{u^2+x^2+y^2+z^2}}{x+\sqrt{u^2+x^2+y^2+z^2}}}. \end{aligned} \tag{3.22}$$

According to (3.21), we obtain the following expression for real-valued $\theta$:

$$\begin{aligned} \theta&=\arg\left(\sqrt{p}\right)=\arctan\frac{Bv}{\psi_1} \\ &=\arctan\left[\frac{v}{\frac{\sqrt{2}\sqrt{x+\sqrt{u^2+x^2+y^2+z^2}}}{\sqrt{x+\sqrt{u^2+x^2+y^2+z^2}}}}\right] \\ &=\arctan\left[\frac{v}{x+\sqrt{u^2+x^2+y^2+z^2}}\right] \end{aligned} \tag{3.23}$$

where $v$ is determined by (2.12).

Thus, we have specified polar form of the function $\sqrt{p}$:

$$\sqrt{p}=m(\cos\theta+r\sin\theta),$$

where $m$ and $\theta$ are defined by (3.22) and (3.23). The values of $m$ and $\theta$ for any $\mathbb{H}$-holomorphic function can be calculated by launching Program A3.2 (RESULT 2).

The conformity of polar forms (3.20) of $\mathbb{H}$ -holomorphic functions and polar forms $\psi_C(p)=m(\cos\theta+i\sin\theta)$ of complex holomorphic functions [9] reflects the similarity between concepts of essentially adequate quaternionic holomorphic functions and holomorphic functions in complex analysis, which is already identified earlier in [1].

## 3.3. General Formula for Logarithms of Complicated $\mathbb{H}$-holomorphic Functions

The isomorphy of the algebras of complex numbers with imaginary unit $i$ and "complex numbers with imaginary unit $r$" allows us to use the following quaternionic generalization of complex Euler's formula:

$$e^{r\theta}=\cos\theta+r\sin\theta.$$

Using this formula we rewrite (3.20) as follows:

$$\psi_H(p)=me^{r\theta}. \tag{3.24}$$

Substituting the identity $m=e^{\log(m)}$ into (3.24), we obtain the following expression:

$$\psi_H(p)=me^{r\theta}=e^{\log(m)}e^{r\theta},$$

whence, taking into account the above mentioned isomorphy of complex algebras with imaginary units $r$ and $i$, we have

$$\psi_H(p)=e^{\log(m)+r\theta}. \tag{3.25}$$

According to the general definition of the logarithm [9], we now get from (3.25) the following expression for the quaternionic logarithm of any complicated $\mathbb{H}$-holomorphic function in $\mathbb{R}^4$–representation:

$$\log\left(\psi_H(p)\right)=\log(m)+r\theta. \tag{3.26}$$

It must be emphasized that we use the designation "log" for natural logarithms, as is customary in system Wolfram Mathematica®, instead of the usual designation "ln".

To get this expression in $\mathbb{C}^2$ –representation needed for further processing by Programmes A3.1 and A3.2, we must represent $r$ as a quaternionic function in the Cayley-Dickson doubling form. It follows from (1.2) and (1.3) that

$$yi=\frac{(a-\bar{a})}{2},\ zj+uk=b\cdot j.$$

Substituting these into (3.16) we get $r$ in the Cayley-Dickson doubling form as hollows:

$$r=\frac{(yi+zj+uk)}{v}=\frac{(a-\bar{a})}{2v}+\frac{b}{v}\cdot j. \tag{3.27}$$

Further, substituting (3.27) into (3.26), we obtain the following expression:

$$\log\left(\psi_H\left(p\right)\right)=\log(m)+(\frac{\left(a-\overline{a}\right)}{2v}+\frac{b}{v}\cdot j)\theta,$$

whence, given that $\theta$ is real-valued, we have

$$\log\left(\psi_H\left(p\right)\right)=\log\left(m\right)+\frac{\left(a-\overline{a}\right)\theta}{2v}+\frac{b\theta}{v}\cdot j.$$

Given (3.21), we represent this expression in the Cayley-Dickson doubling form, so to speak, in the mixed $\mathbb{R}^4$ and $\mathbb{C}^2$ – representation as follows:

$$\log\left(\psi_H\left(p\right)\right)=\phi_1+\phi_2\cdot j,$$

where

$$\phi_1=\log\left(m\right)+\frac{\left(a-\overline{a}\right)\arctan\frac{Bv}{\psi_1}}{2v},\ \phi_2=\frac{b\arctan\frac{Bv}{\psi_1}}{v}.$$

Considering the function $\psi_H(p)$ in $\mathbb{C}^2$ –representation, we could use Program A3.2 to represent $\psi_H(p)$ in the form (1.4), whence we could extract the expressions for $\psi_1$ and $B$ in $\mathbb{R}^4$ –representation. However, using all programmes of Appendix provides for data input in $\mathbb{C}^2$ –representation, and therefore we must rewrite the expressions for $\psi_1$ and $B$ in $\mathbb{C}^2$ –representation.

Using expression (1.6) and its complex conjugation, we get $\psi_1$ directly in $\mathbb{C}^2$ –representation as follows:

$$\psi_1\left(a,b,\overline{a},\overline{b}\right)=\frac{\phi_1\left(a,b,\overline{a},\overline{b}\right)+\overline{\phi_1\left(a,b,\overline{a},\overline{b}\right)}}{2}.\quad(3.28)$$

Further, using (1.7), we have in $\mathbb{C}^2$ –representation:

$$B\left(a,b,\overline{a},\overline{b}\right)=\frac{\phi_2\left(a,b,\overline{a},\overline{b}\right)}{b}.\quad(3.29)$$

According to usual [9] definition of the modulus of a comlex function, we also get the following expression for $m$ in $\mathbb{C}^2$ –representation:

$$m\left(a,b,\overline{a},\overline{b}\right)=\sqrt{\phi_1\overline{\phi_1}+\phi_2\overline{\phi_2}}.\quad(3.30)$$

Finally, we obtain the following general formula for the logarithm of any complicated function $\psi_H(p)$ in $\mathbb{C}^2$ –representation:

$$\log\left(\psi_H\left(p\right)\right)=\phi_1\left(a,b,\overline{a},\overline{b}\right)+\phi_2\left(a,b,\overline{a},\overline{b}\right)\cdot j,\quad(3.31)$$

where

$$\phi_1\left(a,b,\overline{a},\overline{b}\right)=\log\left(m\right)+\frac{\left(a-\overline{a}\right)\arctan\frac{Bv}{\psi_1}}{2v},$$

$$\phi_2\left(a,b,\overline{a},\overline{b}\right)=\frac{b\arctan\frac{Bv}{\psi_1}}{v},$$

and $\psi_1$, $B$, $m$ and $v$ are defined by (3.28), (3.29), (3.30), and (2.12), respectively, in $\mathbb{C}^2$ –representation.

Given (3.26), we can interpret logarithms of $\mathbb{H}$-holomorphic functions as holomorphic logarithms in "the complex plane with imaginary unit r". Then we can speak of the principal values domains [9]: $m>0$ and $-\pi<\theta=\arctan\frac{Bv}{\psi_1}\le\pi$.

Calculation of the logarithm of any complicated $\mathbb{H}$ -holomorphic function $\psi_H(p)$ is carried out by the special program represented in the cell A2.5.

Recall that in all data cells and Programmes A3.1, A3.2 we employ the designations of components $f1[a,ac,b,bc]$ and $f2[a,ac,b,bc]$ of the function $\psi_H(p)$ in the Cayley-Dickson doubling form instead of designations $\phi_1$ and $\phi_2$, respectively.

## 4. Inverse Trigonometric and Hyperbolic $\mathbb{H}$-holomorphic Functions

Now, knowing how to build logarithms of complicated $\mathbb{H}$ -holomorphic functions, we can construct quaternionic inverse trigonometric and hyperbolic functions.

The principal branches of the basic inverse trigonometric and hyperbolic $\mathbb{C}$-holomorphic functions are given [9] by the formulas:

$$\mathrm{Arccos}\xi=-i\mathrm{Log}[\xi+\sqrt{\xi^2-1}]$$

$$\mathrm{Arcsin}\xi=-i\mathrm{Log}[i\xi+\sqrt{1-\xi^2}]$$

$$\mathrm{Arctan}\xi=\frac{i}{2}\mathrm{Log}[\frac{i+\xi}{i-\xi}]$$

$$\mathrm{Arccosh}\xi=\mathrm{Log}[\xi+\sqrt{\xi^2-1}],$$

$$\mathrm{Arcsinh}\xi=\mathrm{Log}[\xi+\sqrt{\xi^2+1}],$$

$$\mathrm{Arctanh}\xi=\frac{1}{2}\mathrm{Log}[\frac{1+\xi}{1-\xi}],$$

where $\xi$ designates an independent complex variable.

It is sufficient to note that the domains of the definitions of logarithms are here such that the inverse functions are single-valued, i.e. if we have $\mathrm{Log}[\psi_H(p)]$, then the domain of inverse function with this logarithm can be considered as $|\psi_H(p)|>0$ and $-\pi<\mathrm{Arg}(\psi_H(p))\le\pi$. For example, the principal value of the inverse function $\mathrm{Arccos}\,\xi$ is given in $\left|\xi+\sqrt{\xi^2-1}\right|>0$ and $-\pi<\mathrm{Arg}(\xi+\sqrt{\xi^2-1})\le\pi$. In order not to overburden this article we do not here detail.

By replacing the complex variable $\xi$ by the quaternionic variable $p$ (here it is taken into account that complex imaginary unit $i$ must be also replaced by quaternionic $r$) we get by virtue of Theorem 1.3 the following quaternionic generalizations of the above principal branches of inverse complex trigonometric and hyperbolic functions:

$$\mathrm{Arccos}p=-r\cdot\mathrm{Log}[p+\sqrt{p^2-1}]$$

$$\mathrm{Arcsin}p=-r\cdot\mathrm{Log}[r\cdot p+\sqrt{1-p^2}],$$

$$\mathrm{Arctan}p=\frac{r}{2}\cdot\mathrm{Log}[\frac{r+p}{r-p}],$$

$$\mathrm{Arccosh}p=\mathrm{Log}[p+\sqrt{p^2-1}],$$

$$\text{Arcsinh}p = \text{Log}[\,p + \sqrt{p^2+1}\,],$$
$$\text{Arctanh}p = \frac{1}{2}\text{Log}[\frac{1+p}{1-p}],$$

where "·" sign indicates the quaternionic multiplication.

Since we perform the quaternionic generalizations of the principal values of the complex inverse functions, i.e. single-valued functions, it is evident that the above inverse quaternionic trigonometric and hyperbolic functions are single-valued.

To calculate an inverse quaternionic function in question it is necessary 1) to calculate its complicated argument (i.e. the functions $f1[a,b,\bar{a},\bar{b}]$ and $f2[a,b,\bar{a},\bar{b}]$ of it), 2) then the logarithmic function of this argument, and then, finally, 3) the inverse quaternionic function on the whole. Next we can verify the $\mathbb{H}$-holomorphy of the obtained inverse quaternionic function by launching Program A3.1.

The given sequence of launching is important, since the calculated needed functions $f1$ and $f2$ will be transmitted from one data cell or program to another.

*Consider in detail the function* $Arccos\,p$. The needed components $f1[a,b,\bar{a},\bar{b}]$ and $f2[a,b,\bar{a},\bar{b}]$ of the complicated variable in the logarithm:

$$p+\sqrt{p^2-1} = f1[a,b,\bar{a},\bar{b}] + f2[a,b,\bar{a},\bar{b}]\cdot j$$

are already got in Example 2.2. They are calculated by launching the data cell A2.1. Next, we launch the data cell A2.5 to obtain the function $\text{Log}[p+\sqrt{p^2-1}]$, whereafter we launch the cell A2.6 to calculate, finally, the components $f1[a,b,\bar{a},\bar{b}]$ and $f2[a,b,\bar{a},\bar{b}]$ of the inverse function

$$\text{Arccos}p = -r\text{Log}\left[\,p+\sqrt{p^2-1}\,\right]$$
$$= f1[a,b,\bar{a},\bar{b}] + f2[a,b,\bar{a},\bar{b}]\cdot j.$$

They are the following:

$$f1[a,b,\bar{a},\bar{b}] =$$
$$-\frac{1}{e1}\{-e1\,\text{Arctan}\left[\frac{2v}{\sqrt{e3}}\right] + (\bar{a}-a)\sqrt{-e1}$$
$$\cdot\,\text{Log}[\sqrt{\frac{(-1+a\bar{a}+b\bar{b}+\sqrt{e2})(-1+a^2+a\bar{a}+\bar{a}^2-b\bar{b}+\sqrt{e2}+(a+\bar{a})\sqrt{e3})}{e3}}\,]\}$$

and

$$f2[a,b,\bar{a},\bar{b}] = -\frac{2b}{\sqrt{-e1}}$$
$$\cdot\,\text{Log}\left[\sqrt{\frac{(-1+a\bar{a}+b\bar{b}+\sqrt{e2})(-1+a^2+\bar{a}^2-b\bar{b}+\sqrt{e2}+(a+\bar{a})\sqrt{e3})}{e3}}\right],$$

where

$$e1 = a^2 - 2a\bar{a} + \bar{a}^2 - 4b\bar{b},$$
$$e2 = -\bar{a}^2 + a^2\left(-1+\bar{a}^2\right) + 2a\bar{a}b\bar{b} + (1+b\bar{b})^2,$$
$$e3 = -2 + a^2 + \bar{a}^2 - 2b\bar{b} + 2\sqrt{e2}.$$

*Consider the inverse hyperbolic function* $Arcsinh\,p$. By launching the data cell A2.4 we obtain the components $f1[a,b,\bar{a},\bar{b}]$, $f2[a,b,\bar{a},\bar{b}]$ of the complicated argument $p+\sqrt{p^2+1} = f1[a,b,\bar{a},\bar{b}] + f2[a,b,\bar{a},\bar{b}]\cdot j$. Then we launch the data cell A2.5 to calculate the components $f1[a,b,\bar{a},\bar{b}]$ and $f2[a,b,\bar{a},\bar{b}]$ of the logarithm of $p+\sqrt{p^2+1}$. Further, we launch the data cell A2.9 to get, finally, the components $f1[a,b,\bar{a},\bar{b}]$ and $f2[a,b,\bar{a},\bar{b}]$ of the function $\text{Arcsinh}\,p$. Next, by launching Program A3.1 we can verify whether this function is $\mathbb{H}$-holomorphic.

The substitutions into the base function $\sqrt{p}$ (see Example 2.2) when calculating $p+\sqrt{p^2+1}$ are the following:

$$ta = a^2 - b\bar{b} + 1,\ tb = (a+\bar{a})b,$$
$$t\bar{a} = \bar{a}^2 - b\bar{b} + 1,\ t\bar{b} = (a+\bar{a})\bar{b}.$$

The $\mathbb{H}$-holomorphic function $\text{Arcsin}\,p$ is calculated by the following sequence of launching: 1) launching the data cell A2.3 to calculate $rp+\sqrt{1-p^2}$, 2) launching the data cell A2.5 to calculate $\text{Log}[rp+\sqrt{1-p^2}]$, and 3) launching the data cell A2.7.

The $\mathbb{H}$-holomorphic function $\text{Arccosh}\,p$ is calculated by the following sequence of launching: 1) launching the data cell A2.1, 2) launching the data cell A2.5 to calculate $\text{Log}[p+\sqrt{p^2-1}]$, and 3) launching the data cell A2.8.

We shall not dwell further on the remaining inverse functions $\text{Arctan}\,p$, $\text{Arctanh}\,p$. These can be calculated analogously. They are also $\mathbb{H}$-holomorphic. We notice only that calculation of quaternionic arguments in the expressions for these functions is performed by using the following formulas:

$$\frac{r+p}{r-p} = \frac{(r+p)\overline{(r-p)}}{(r-p)\overline{(r-p)}} \text{ and } \frac{1+p}{1-p} = \frac{(1+p)\overline{(1-p)}}{(1-p)\overline{(1-p)}}.$$

Instead of these, it is also possible to use the formulas:

$$\text{Log}\left[\frac{r+p}{r-p}\right] = \text{Log}[r+p] - \text{Log}[r-p]$$
$$\text{Log}\left[\frac{1+p}{1-p}\right] = \text{Log}[1+p] - \text{Log}[1-p]$$

We have shown that within the essentially adequate concept of the quaternionic holomorphy it is possible to construct successfully the complicated $\mathbb{H}$-holomorphic functions and their $\mathbb{H}$-holomorphic compositions by using the presented substitutions way and the general formula (3.31) for quaternionic logarithmic function, based on deduced expressions for vector parts and polar forms.

We have reaffirmed the main principle of the essentially adequate concept of $\mathbb{H}$-holomorphy: *all* $\mathbb{H}$*-holomorphic functions of any degree of difficulty* can be constructed from their $\mathbb{C}$-holomorphic analogs. The established $\mathbb{R}^4$ –representations of vector parts and polar forms inherent only to $\mathbb{H}$-holomorphic functions are similar to complex ones that demonstrates continuity in the development of the essentially adequate theory of the quaternionic holomorphy.

## References


[1] Parfenov, M., "Essentially Adequate Concept of Holomorphic functions in Quaternionic Analysis". *American Journal of Mathematical Analysis*, vol. 8, no. 1, 14-30. Jun 2020.

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

[3] Rönn, S., "*Bicomplex algebra and function theory*", arXiv: math.CV/ 0101200v1, Oct 2018. Available: arxiv.org/pdf/math/0101200.pdf.

[4] Parfenov, M., "A Quaternionic Potential Conception with Applying to 3D Potential Fields". *American Journal of Mathematical Analysis*, Vol. 7, No. 1, 1-10. Apr 2019.

[5] Parfenov, M., "On Properties of Holomorphic Functions in Quaternionic Analysis." *American Journal of Mathematical Analysis*, Vol. 5, No. 1, 17-24. Jul 2017.

[6] Parfenov, M., *Processing of the H-holomorphic Functions*, preprint, [viXra: Functions and Analysis], May 2021. Available: https://vixra.org/abs/2010.0210.

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

[8] Stephen J. Sangwine, Nicolas Le Bihan. "*Quaternion polar representation with a complex modulus and complex argument inspired by the Cayley-Dickson form*", arXiv:0802.0852v2 [math.RA], available: https://arxiv.org/abs/0802.0852.

[9] Mathews, J. H., Howell, R. W., *Complex Analysis for Mathematics and Engineering*, 3rd ed, Jones and Bartlett Publishers, Boston-Toronto-London-Singapore, 1997.


## Appendix

The data base and programming codes, without wich no work with complicated $\mathbb{H}$-holomorphic functions is practically possible, are represented here.

To work with represented below data and processing programmes we need to open an empty Notebook file .nb in the installed computing system Wolfram Mathematica® [7]. Then we need to copy (for instance, by keys "Strg"+"C" and "Strg"+"V") all functions examples and processing programmes, cell by cell, from PDF file of this text into input cells (only one function or one program into one new cell!) of the opened file .nb. *Copying starts always with the line (*...*).*

*Checking the correctness of cells copying, we need to take into account that the functions without semicolon at the end of them must after copying be each in a separate line in the cell of file .nb. All data cells and programmes shall be left-aligned. If it is not a case, then it must be corrected by the Enter key in copied file .nb.* For more details, refer to [1,6].

Further, to launch each function data or programing code we need to click on it's cell in the file .nb of Wolfram Mathematica® computing system, and then holding down the Shift key while press the Enter key. When speaking of launching (evaluating [7]) any programming code in any cell we will always mean this sequence of key-board and mouse actions.

The task of each data cell is to calculate the functions $f1[a,b,ac,\mathrm{bc}]$, $f2[a,b,ac,\mathrm{bc}]$,which are, respectively, the programming designations for the components $\phi_1$ and $\phi_2$ of a quaternionic function in question. The designations $a,b,ac,bc$ and $f1c$, $f2c$ are used, respectively instead of $a,b,\overline{a},\overline{b}$ and $\overline{f1}$, $\overline{f2}$ in programming codes. In the case of quaternionic derivatives the functions $f1$ and $f2$ can designate, respectively, the components $\phi_1^{(k)}$ and $\phi_2^{(k)}$ of the $k'$th derivative in its Cayley-Dickson doubling form [1].

These components are calculated by launching the data cells. The lastest calculated expressions for $f1$ and $f2$ remain in Mathematica® computing system memory as long as they will be replaced by the others. By launching the functions processing, it is important to remember what the functions $f1$ and $f2$ are in computing system memory at the moment.

The functions $f1$ and $f2$ are the input data for data cells A2.5–A2.9 and main Programmes A3.1 and A3.2. Program A3.1 verifies whether a quaternionic function in question is $\mathbb{H}$-holomorphic. Program A3.2 calculates the needed quaternionic expressions in $\mathbb{R}^4$ –representation for $\mathbb{H}$-holomorphic functions in question, including their polar form as well as expressions for the first derivatives, expressions for 3D field vectors and their vortex density and divergence.

The cell A2.5 calculates the logarithm of $\mathbb{H}$-holomorphic function given by any of the cells A1.2-A1.9, A2.1-A2.4 as well as by A2.6-A2.9 or another $\mathbb{H}$-holomorphic function.

Calculation of the inverse trigonometric or hyperbolic function is proceeded in the following sequence: 1) by launching one of the cells: A2.1, A2.3 or A2.4, according to an argument of a logarithm in an inverse function in question, then 2) by launching the cell A2.5 to calculate the Log of this argument, and then 3) by launching one of the cells A2.6 – A2.9, corresponding the inverse function in question. In the case of violation of the correct sequence the programmes generate error messages.

In order to have a possibility to copy all programming codes directly from PDF file of this article into empty input cells of the opened Notebook blank .nb of the computing system Wolfram Mathematica®, and then to launch them immediately, without corrections, in the file .nb, the following designations are used in programing codes:

1) psi(p), psi1, psi2, psi3, psi4 instead of $\psi(p), \psi_1, \psi_2, \psi_3, \psi_4$, respectively; 2) phi1, phi2, phi3, phi4 instead of $\varphi_1$, $\varphi_2$ , $\varphi_3$, $\varphi_4$, respectively; 3) d1psi, d1f1, d1f2 and d1psi1, d1psi2, d1psi3, d1psi4 instead of $\psi(p)'$, $\phi_1'$, $\phi_2'$ and $\psi_1'$, $\psi_2'$, $\psi_3'$, $\psi_4'$, 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 $\frac{\mathrm{dfn}\,2}{\mathrm{dy}}$ instead of $\frac{\partial f_2}{\partial y}$ [4].

At the beginning of each Mathematica® session on processing of the $\mathbb{H}$-holomorphic functions we necessarily need to launch the data cell A1.1 one time, otherwise data processing shall be incorrect. When having difficulty the more details could be find in [1,6].

## A.1. Base Functions Examples

### A1.1. Initial functions and variables setting

```
(*Initial functions and variables setting *)
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);
s=0; s1=0; forarc=0;
```

### A1.2. The Function $p$

```
(*A1.2: Funcion p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c, inib2,inib2c, quatfunctiontested];
quatfunctiontested:= "p";
quatfunctiontested
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;
```

### A1.3. The Function $p^2$

```
(*A1.3: Funcion p^2*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c, inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[p^2];
quatfunctiontested
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;
```

### A1.4. The Function $p^3$

```
(*A1.4: Funcion p^3*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c, inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[p^3];
quatfunctiontested
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;
```

### A1.5. The Function $\frac{1}{p}$

```
(*A1.5: Function 1/p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c, inib2,inib2c, quatfunctiontested];
quatfunctiontested:=Evaluate[1/p];
quatfunctiontested
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;
```

### A1.6. The Function $\sin p$

```
(*A1.6: Funcion sinp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c, inib2,inib2c, quatfunctiontested];
quatfunctiontested:="sinp";
quatfunctiontested
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;
```

### A1.7. The Function $\cos p$

```
(*A1.7: Funcion cosp*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c, inib2,inib2c, quatfunctiontested];
quatfunctiontested:="cosp";
quatfunctiontested
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;
```

### A1.8. The Function $e^p$

```
(*A1.8: Funcion e^p*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,quatfunctiontested];
quatfunctiontested:=Evaluate[e^p];
quatfunctiontested
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;
```

### A1.9. The Function $\sqrt{p}$

```
(*A1.9: Quaternionic generalization of a principal branch of a complex square root*)
Clear[f1,f2,f1c,f2c,inia1,inia1c, inia2,inia2c, inib1,inib1c, inib2,inib2c, quatfunctiontested];
quatfunctiontested:= Evaluate[Sqrt[p]];
quatfunctiontested
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;
```

## A.2. Complicated Functions Examples

### A2.1. The Function $p+\sqrt{p^2-1}$

```
(*A2.1: Funcion p+Sqrt[p^2-1], argument of Log in the functions Arccos(p) and Arccosh(p) *)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,quatfunctiontested];
quatfunctiontested:=Evaluate[p+Sqrt[p^2-1]];
ta:=a^2-b*bc-1;tac:=ac^2-b*bc-1;tb:=(a+ac)*b;tbc:=(a+ac)*bc;
sr1:=Sqrt[((ta+tac)/2+Sqrt[ta*tac+tb*tbc])];
f1[a_,ac_,b_,bc_]:=a+(ta-tac)/(2*Sqrt[2]*sr1)+sr1/Sqrt[2];f1c[a_,ac_,b_,bc_]:=ac+(tac-ta)/(2*Sqrt[2]*sr1)+sr1/Sqrt[2];
f2[a_,ac_,b_,bc_]:=b+tb/(Sqrt[2]*sr1);
f2c[a_,ac_,b_,bc_]:=bc+tbc/(Sqrt[2]*sr1);
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; s1=0; forarc=1;
```

### A2.2. The Function $\sin\frac{1}{e^{p^3}}$

```
(*A2.2: Funcion sin(1/(e^p^3))*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,quatfunctiontested];
quatfunctiontested:=Evaluate[Sin[1/e^(p^3)]];
quatfunctiontested
ta:=a^3-(2*a+ac)*b*bc;
tb:=(a^2+a*ac+ac^2)*b-b^2*bc;
tac:=ac^3-(2*ac+a)*b*bc;
tbc:=(ac^2+a*ac+a^2)*bc-bc^2*b;
tbeta=(1/2)*E^((ta+tac)/2);
tv=(1/2)*Sqrt[4*(ta*tac+tb*tbc)-(ta+tac)^2];
t2a:=Simplify[2*tbeta*Cos[tv]+(tbeta*(ta-tac)*Sin[tv])/tv];
t2b:=Simplify[(2*tbeta*tb*Sin[tv])/tv];
t2ac:=Simplify[2*tbeta*Cos[tv]+(tbeta*(tac-ta)*Sin[tv])/tv];
t2bc:=Simplify[(2*tbeta*tbc*Sin[tv])/tv];
t3a:=Simplify[t2ac/(t2a*t2ac+t2b*t2bc)];
t3b:=Simplify[-t2b/(t2a*t2ac+t2b*t2bc)];
t3ac:=Simplify[t2a/(t2a*t2ac+t2b*t2bc)];
t3bc:=Simplify[-t2bc/(t2a*t2ac+t2b*t2bc)];
t3v=Simplify[(1/2)*Sqrt[4*(t3a*t3ac+t3b*t3bc)-(t3a+t3ac)^2]];
t3epl=Simplify[E^t3v];
t3emin=Simplify[E^(-t3v)];
f1[a_,ac_,b_,bc_]:=Simplify[((t3emin+t3epl)*Sin[(t3a+t3ac)/2])/2-((t3a-t3ac)*(t3emin-
t3epl)*Cos[(t3a+t3ac)/2])/(4*t3v)];
f2[a_,ac_,b_,bc_]:=Simplify[-((t3emin-t3epl)*Cos[(t3a+t3ac)/2]*t3b)/(2*t3v)];
f1c[a_,ac_,b_,bc_]:=Simplify[((t3emin+t3epl)*Sin[(t3a+t3ac)/2])/2-((t3ac-t3a)*(t3emin-
t3epl)*Cos[(t3a+t3ac)/2])/(4*t3v)];
f2c[a_,ac_,b_,bc_]:=Simplify[-((t3emin-t3epl)*Cos[(t3a+t3ac)/2]*t3bc)/(2*t3v)];
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;
```

### A2.3. The Function $rp+\sqrt{1-p^2}$

```
(*A2.3: Funcion rp+Sqrt[1- p^2], argument of Log in the function Arcsin(p)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,quatfunctiontested];
quatfunctiontested:=Evaluate[r*p+Sqrt[1-p^2]];
```

```
fa1:=(a-ac)/(2*v);fa2:=b/v;fa1c:=(ac-a)/(2*v);fa2c:=bc/v;
fb1:=a; fb2:=b; fb1c:=ac; fb2c:=bc;
p1f1:=fa1*fb1-fa2*fb2c; p1f1c:=fa1c*fb1c-fa2c*fb2;p1f2:=fa1*fb2+fa2*fb1c; p1f2c:=fa1c*fb2c+fa2c*fb1;
ta:=1-a^2+b*bc; tac:=1-ac^2+b*bc; tb:=-(a+ac)*b; tbc:=-(a+ac)*bc;
tsr:=(Sqrt[((ta+tac)/2+Sqrt[ta*tac+tb*tbc])/2]);
p2f1:=tsr+(ta-tac)/(4*tsr); p2f1c:=tsr+(tac-ta)/(4*tsr);
p2f2:=tb/(2*tsr); p2f2c:=tbc/(2*tsr);
f1[a_,ac_,b_,bc_]=Simplify[p1f1+p2f1];f1c[a_,ac_,b_,bc_]=Simplify[p1f1c+p2f1c];
f2[a_,ac_,b_,bc_]=Simplify[p1f2+p2f2];
f2c[a_,ac_,b_,bc_]=Simplify[p1f2c+p2f2c];
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; s1=0; forarc=2;
```

### A2.4. The Function $p + \sqrt{p^2 + 1}$

```
(*A2.4: Funcion p+Sqrt[1+ p^2], argument of Log in the function Arcsinh(p)*)
Clear[f1,f2,f1c,f2c,inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,quatfunctiontested];
quatfunctiontested:=Evaluate[p+Sqrt[p^2+1]];
ta:=a^2-b*bc+1; tac:=ac^2-b*bc+1; tb:=(a+ac)*b; tbc:=(a+ac)*bc;
sr1:=Sqrt[((ta+tac)/2+Sqrt[ta*tac+tb*tbc])];
f1[a_,ac_,b_,bc_]:=a+(ta-tac)/(2*Sqrt[2]*sr1)+sr1/Sqrt[2];f1c[a_,ac_,b_,bc_]:=ac+(tac-ta)/(2*Sqrt[2]*sr1)+sr1/Sqrt[2];
f2[a_,ac_,b_,bc_]:=b+tb/(Sqrt[2]*sr1);
f2c[a_,ac_,b_,bc_]:=bc+tbc/(Sqrt[2]*sr1);
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; s1=0; forarc=3;
```

### A2.5. The Function Log$[\psi_H(p)]$

First we launch one of the functions from A1 or A2 (or another $\mathbb{H}$-holomorphic function) and then launch this cell to obtain the logarithm of the function in question, i.e. the functions $f1$ and $f2$ of the function Log$[\psi_H(p)]$.

```
(*A2.5: Special program to get the Log of any ℍ-holomorphic function*)
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[s==0,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]];m=Simplify[Sqrt[argf1*argf1c+argf2*argf2c]];logm=Simplify[Log[m]];
psi1=Simplify[(argf1+argf1c)/2]; 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)]; f1c[a_,ac_,b_,bc_]=Simplify[logm+((ac-a)*ArcTan[(bbig*v)/psi1])/(2*v)];
f2[a_,ac_,b_,bc_]=Simplify[(b*ArcTan[(bbig*v)/psi1])/v];
f2c[a_,ac_,b_,bc_]=Simplify[(bc*ArcTan[(bbig*v)/psi1])/v];]];
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]
s=s+1; mark=False; markquot=False;
```

### A2.6. The Function Arccos(p) $= -r$Log$[p + \sqrt{p^2 - 1}]$

```
(*A2.6: Function Arccos(p), launching this cell should follow only after the following sequence of launching:
1) launching the data cell A2.1, then 2) launching the data cell A2.5 *)
Clear[inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,bbig,m,logm,psi1,coteta,siteta,cosikv];
nolog=StringFreeQ[ToString[quatfunctiontested, StandardForm],"Log"];
If[forarc!=1||nolog||mark,
Print[" ////////////////// ERROR://///////////////\n Incorrect data input! The imposed argument is\n the wrong for the function Arccos. The supplied function\n can not be used. Results can not be correct."];
ClearAll[f1,f1c,f2,f2c];,
```

```
If[s1==0,name=ToString[quatfunctiontested, StandardForm];lgf1=f1[a,ac,b,bc]; lgf1c=f1c[a,ac,b,bc];lgf2=f2[a,ac,b,bc];
lgf2c=f2c[a,ac,b,bc];
quatfunctiontested="Arccos(p)=-r*"<>name;
fa1=(ac-a)/(2*v);fa2=-b/v;fa1c=(a-ac)/(2*v);fa2c=-bc/v;
vf1=fa1*lgf1-fa2*lgf2c;vf1c=fa1c*lgf1c-fa2c*lgf2;
f1[a_,ac_,b_,bc_]=Simplify[vf1,TimeConstraint->4400];
f1c[a_,ac_,b_,bc_]=Simplify[vf1c,TimeConstraint->4400];
vf2=fa1*lgf2+fa2*lgf1c;vf2c=fa1c*lgf2c+fa2c*lgf1;
f2[a_,ac_,b_,bc_]=Simplify[vf2,TimeConstraint->4400];
f2c[a_,ac_,b_,bc_]=Simplify[vf2c,TimeConstraint->4400];]];
If[forarc!=1||nolog||mark,,
Print
["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"]];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
s1=s1+1; mark=False; markquot=False; s=0; forarc=0;
```

### A2.7. The Function $\mathbf{Arcsin(p) = -r Log[r p + \sqrt{1 - p^2}]}$

```
   (*A2.7: Function Arcsin(p), launching only after the following sequence: 1) launching the data cell A2.3, then
2) launching the data cell A2.5 *)
Clear[inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,bbig,m,logm,psi1,coteta,siteta,cosikv];
nolog=StringFreeQ[ToString[quatfunctiontested, StandardForm],"Log"];
If[forarc!=2||nolog||mark,Print[" ////////////////// ERROR://///////////////\n Incorrect data input! The imposed argument is\n the
wrong for the function Arcsin. The supplied function\n can not be used. Results can not be correct."];
ClearAll[f1,f1c,f2,f2c],
If[s1==0,name=ToString[quatfunctiontested, StandardForm];lgf1=f1[a,ac,b,bc];lgf1c=f1c[a,ac,b,bc];
lgf2=f2[a,ac,b,bc];lgf2c=f2c[a,ac,b,bc];
quatfunctiontested="Arcsin(p)=-r*"<>name;
fa1=(ac-a)/(2*v); fa2=-b/v; fa1c=(a-ac)/(2*v); fa2c=-bc/v;
vf1=fa1*lgf1-fa2*lgf2c; vf1c=fa1c*lgf1c-fa2c*lgf2;
f1[a_,ac_,b_,bc_]=Simplify[vf1,TimeConstraint->4400];
f1c[a_,ac_,b_,bc_]=Simplify[vf1c,TimeConstraint->4400];
vf2=fa1*lgf2+fa2*lgf1c; vf2c=fa1c*lgf2c+fa2c*lgf1;
f2[a_,ac_,b_,bc_]=Simplify[vf2,TimeConstraint->4400];
f2c[a_,ac_,b_,bc_]=Simplify[vf2c,TimeConstraint->4400];]];
If[forarc!=2||nolog||mark,,
Print
["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"]];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
s1=s1+1; mark=False; markquot=False; s=0; forarc=0;
```

### A2.8. The Function $\mathbf{Arccosh(p) = Log[p + \sqrt{p^2 - 1}]}$

```
   (*A2.8: Function Arccosh(p), launching only after the following sequence: 1) launching the data cell A2.1, then
2) launching the data cell A2.5 *)
Clear[inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,bbig,m,logm,psi1,coteta,siteta,cosikv];
nolog=StringFreeQ[ToString[quatfunctiontested, StandardForm],"Log"];
If[forarc!=1||nolog||mark,
  Print[" ////////////////// ERROR://///////////////\n Incorrect data input! The imposed argument is\n the wrong for the function
Arccosh. The supplied function\n can not be used. Results can not be correct."];
  ClearAll[f1,f1c,f2,f2c];,
If[s1==0,name=ToString[quatfunctiontested, StandardForm];lgf1=f1[a,ac,b,bc];
lgf1c=f1c[a,ac,b,bc];lgf2=f2[a,ac,b,bc];lgf2c=f2c[a,ac,b,bc];
quatfunctiontested="Arccosh(p)="<>name;
f1[a_,ac_,b_,bc_]=Simplify[lgf1,TimeConstraint->4400];
f1c[a_,ac_,b_,bc_]=Simplify[lgf1c,TimeConstraint->4400];
f2[a_,ac_,b_,bc_]=Simplify[lgf2,TimeConstraint->4400];
f2c[a_,ac_,b_,bc_]=Simplify[lgf2c,TimeConstraint->4400];]];
If[forarc!=1||nolog||mark,,
  Print
   ["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"]];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
```

```
s1=s1+1; mark=False; markquot=False; s=0; forarc=0;
```

### A2.9. The Function $\mathbf{Arcsinh(p) = Log[p + \sqrt{p^2 + 1}]}$

```
(*A2.9: Function Arcsinh(p), launching only after the following sequence: 1) launching the data cell A2.4, then
2) launching the data cell A2.5 *)
Clear[inia1,inia1c,inia2,inia2c,inib1,inib1c,inib2,inib2c,bbig,m,logm,psi1,coteta,siteta,cosikv];
nolog=StringFreeQ[ToString[quatfunctiontested, StandardForm],"Log"];
If[forarc!=3||nolog||mark,
Print[" ////////////////// ERROR://///////////////\n Incorrect data input! The imposed argument is\n the wrong for the function Arcsinh. The supplied function\n can not be used. Results can not be correct."];
ClearAll[f1,f1c,f2,f2c];,
If[s1==0,name=ToString[quatfunctiontested, StandardForm];lgf1=f1[a,ac,b,bc];
lgf1c=f1c[a,ac,b,bc];lgf2=f2[a,ac,b,bc];lgf2c=f2c[a,ac,b,bc];
quatfunctiontested="Arcsinh(p)="<>name;
f1[a_,ac_,b_,bc_]=lgf1;
f1c[a_,ac_,b_,bc_]=lgf1c;
f2[a_,ac_,b_,bc_]=lgf2;
f2c[a_,ac_,b_,bc_]=lgf2c;]];
If[forarc!=3||nolog||mark,,
Print
["Constituents f1,f2 of the quaternionic function ",quatfunctiontested,":"]];
f1[a,ac,b,bc]
f2[a,ac,b,bc]
s1=s1+1; mark=False; markquot=False; s=0; forarc=0;
```

## A.3. Main Processing Programmes

The programmes are adapted for the direct transfer of these programmes (copying) into empty file .nb of the computing system Wolfram Mathematica® to work with them (launch) in this system. To obtain the functions $f1, f2$ as input data for Programmes A3.1, A3.2 it is needed to launch the copied data cell of a function in question from subsections A1 and A2. After that we can launch copied Programmes A3.1, A3.2 for processing of this function.

### A3.1. Testing of $\mathbb{H}$-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 (1.8) 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 the derivative $\partial_a f1$.

```
(*Program A3.1, H - holomorphy testing*)
ClearAll[df1,df1c,df2,df2c,trdf1,sftrdf1,trdf1c,sftrdf1c,trdf2,sftrdf2,trdf2c,sftrdf2c,trmdf1da,trmdf2cdbc,dif1,trmdf2da,tr
mdf1cdbc,dif2,trmdf2db,dif3,trmdf2dac,trmdf1dbc,dif4,eq1,eq2,eq3,eq4];
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};
f1s=Simplify[inif1,{_Symbol∈Reals},TimeConstraint-
>4400];f1sexp=Expand[f1s];ref1=Simplify[selectpart[f1sexp,True],{_Symbol∈Reals},TimeConstraint->4400];
imf1=Simplify[selectpart[f1sexp,False],{_Symbol∈Reals},TimeConstraint->4400];
inif1c=f1c[a,ac,b,bc]/.{a->x+I*y,ac->x-I*y,b->z+I*u,bc->z-I*u};
f1cs=Simplify[inif1c,{_Symbol∈Reals},TimeConstraint->4400];f1csexp=Expand[f1cs];
ref1c=Simplify[selectpart[f1csexp,True],{_Symbol∈Reals},TimeConstraint->4400];
imf1c=Simplify[selectpart[f1csexp,False],{_Symbol∈Reals},TimeConstraint->4400];
f1recoinc=True===Simplify[ref1==ref1c,TimeConstraint->2400];
nonrightf1c=True===Simplify[imf1==imf1c,TimeConstraint->2400];
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||nonrightf1c,Print["////////////////// ERROR://///////////////\n  Incorrect data input (including also the improper function f1c). Repeat again launching the correct 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->4400];
ClearAll[trdf1,df1];
df1c=D[f1c[a,ac,b,bc],{bc}];
trdf1c=df1c/.{a->x,ac->x};
sftrdf1c=Simplify[trdf1c,TimeConstraint->4400];
ClearAll[trdf1c,df1c];
```

```
df2=D[f2[a,ac,b,bc],{{a,ac,b}}];
trdf2=df2/.{a->x,ac->x};
sftrdf2=Simplify[trdf2,TimeConstraint->4400];
ClearAll[trdf2,df2];
df2c=D[f2c[a,ac,b,bc],{bc}];
trdf2c=df2c/.{a->x,ac->x};
sftrdf2c=Simplify[trdf2c,TimeConstraint->4400];
ClearAll[trdf2c,df2c];
trmdf1da=PowerExpand[sftrdf1[[ 1 ]]]//Expand;
trmdf2cdbc=PowerExpand[sftrdf2c]//Expand;
dif1=Simplify[trmdf1da-trmdf2cdbc,TimeConstraint->4400];
ClearAll[trmdf2cdbc];
eq1=True===Simplify[dif1==0,TimeConstraint->4400];
If[eq1,,Print["The equation 1) is not satisfied"]];
ClearAll[dif1];
trmdf2da=PowerExpand[sftrdf2[[ 1 ]]]//Expand;
trmdf1cdbc=PowerExpand[-sftrdf1c]//Expand;
dif2=Simplify[trmdf2da-trmdf1cdbc,TimeConstraint->4400];
ClearAll[trmdf2da,trmdf1cdbc];
eq2=True===Simplify[dif2==0,TimeConstraint->4400];
ClearAll[dif2];
If[eq2,,Print["The equation 2) is not satisfied"]];
trmdf2db=PowerExpand[sftrdf2[[ 3 ]]]//Expand;
dif3=Simplify[trmdf1da-trmdf2db,TimeConstraint->4400];
ClearAll[trmdf1da,trmdf2db];
eq3=True===Simplify[dif3==0,TimeConstraint->4400];
ClearAll[dif3];
If[eq3,,Print["The equation 3) is not satisfied"]];
trmdf2dac=PowerExpand[sftrdf2[[ 2 ]]]//Expand;
trmdf1dbc=PowerExpand[-sftrdf1[[ 2 ]]]//Expand;
dif4=Simplify[trmdf2dac-trmdf1dbc,TimeConstraint->4400];
ClearAll[trmdf2dac,trmdf1dbc];
eq4=True===Simplify[dif4==0,TimeConstraint->4400];
ClearAll[dif4];
If[eq4,,Print["The equation 4) is not satisfied"]];
Print["Testing eq1: ",eq1,"// eq2: ",eq2,"// eq3: ",eq3,"// eq4: ",eq4];
If[eq1∧eq2∧eq3∧eq4,Print["General result: The quaternionic function ",quatfunctiontested,"
is H-holomorphic."],Print["General result: The quaternionic function ",quatfunctiontested,"
is not H-holomorphic."]];]
ClearAll[df1,df1c,df2,df2c,trdf1,sftrdf1,trdf1c,sftrdf1c,trdf2,sftrdf2,trdf2c,sftrdf2c,trmdf1da,trmdf2cdbc,dif1,trmdf2da,tr
mdf1cdbc,dif2,trmdf2db,dif3,trmdf2dac,trmdf1dbc,dif4,eq1,eq2,eq3,eq4];
```

Remark. When processing of non-$\mathbb{H}$-holomorphic functions, this program will indicate those $\mathbb{H}$-holomorphy equations from (1.8), which is not satisfied. Such a resultat appears immediately and we can immediately abort the program to avoid further unnecessary work.

### A3.2. Calculating Quaternionic Expressions

The input data for this program are only the functions $f1$, $f2$, $f1c$, $f2c$. The calculated quaternionic expressions are displayed in $\mathbb{R}^4$ –representation. In the cases of some complicated $\mathbb{H}$-holomorphic functions operating of this program can take a very long time to get all planned expressions for them. However, the calculation of the representation (1.4) and polar form (3.20) takes relatively little time and the results (RESULT 1 and RESULT 2) appear very quickly, whereafter operating of this program, if needed, can be aborted.

```
(*Program A3.2, Calculating quaternionic expressions*)
ClearAll[f1s,f2s,f1sexp,f2sexp,ref1,imf1,ref2,imf2,ref1tr,imf1tr,ref2tr,imf2tr,iniderf1,iniderf2,derf1r4,derf2r4,derf1s,derf
2s,derf1sexp,derf2sexp,rederf1,imderf1,rederf2,imderf2,redf1tr,imdf1tr,redf2tr,imdf2tr,trmf1y0sf,trmf3y0sf,trmf4y0sf,tr
mcurl,const1,const2,const3,curl1,curl2,curl3,curl1together,curl2together,curl3together,curl1togpowexp,curl2togpowexp,c
url3togpowexp,comp1,comp2,comp3,df1dx,df2dy,df3dz,df4du,df1dxtr,df2dytr,df3dztr,df4dutr,df1dxtrsf,trmdf2dytrfullsf,
df3dztrsf,df4dutrsf,df1dxtrsfpowexex,df2dytrsfpowexex,df3dztrsfpowexex,df4dutrsfpowexex,vordivfield,divfieldfullsf,di
vsumminusdf2dy,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:=Match
Q[vr,Complex[xx_/;NumberQ[xx]&&Re[xx]!=0,yy/;NumberQ[yy]]];bg:=bg1||bg2||bg3;If[bg,revr:=Simplify[Re[vr],{_Sy
mbol∈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,a
c,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[divsum
minusdf2dy==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]];]