Hypercomplex Notation in APL

Bob Smith
Sudley Place Software

Originally Written
17 May 2016
Updated
9 Apr 2018

Introduction

In this paper, I present the various forms of notation for Hypercomplex numbers.

Complex Numbers

One choice for the input and output notation for Complex numbers uses infix notation with `i` or `J` as a separator. For example, `2i3` and `2J3` represent the same number.

Following Iverson's lead in J, two additional input notations (for Complex numbers only) allow you to use polar form as in `1ad90` and `1ar2` which describe the radius and angle in either degrees (`ad`) or radians (`ar`). As suggested by David Rabenhorst[6], another input notation is Unit Normalized (`au`) as in `1au0.25` (`=1ad90`) where the right hand part is a normalized number between `0` and `1` where `0` represents `0`º (or `0` radians) and `1` represents `360`º. (or 2π radians).

The display form of either input notation uses either `i` or `J` depending upon a program-wide preference (*Edit | Customize...|Hypercomplex Prefs|Use 'J' instead of 'i' as a Complex number separator*). By default, an imaginary part with value zero is omitted on output depending upon a program-wide preference (*Edit | Customize...|*

Hypercomplex Prefs|Display the imaginary parts of a Hypercomplex number even if 0).

The infix notation shown for Hypercomplex numbers is only one way to enter such numbers. Jacob Brickman[1] has suggested an alternative input and output notation for not just Complex numbers but Hypercomplex numbers as well which uses a combination of infix and postfix notation. For example, `0i1` may be written more succinctly as `1i`, and in general, Complex numbers are written as `1s2i`, Quaternions as `1s2i3j4k`, and Octonions as `1s2i3j4k5l6ij7jk8kl`. If any part is zero (e.g., the `0s` in `0s1i`), it may be elided. This form may be chosen as the output form depending upon a program-wide preference (*Edit | Customize...| Hypercomplex Prefs|Display Hypercomplex numbers using Infix instead of Postfix notation*).

These two forms of input notation may be used interchangeably even within the same strand.

The coefficients of a Hypercomplex number may be any one of four datatypes, as long as all the coefficients come from one of the following four datatypes:

- Fixed precision (64-bit) integer (a.k.a. Gaussian integers)
- Fixed precision (64-bit) floating point
- Multiple precision integer/rational
- Multiple precision floating point

For example,

`1i2`	Complex fixed precision integer
`1i2.5`	Complex fixed precision floating point
`1i5r2`	Complex multiple precision integer/rational
`1i2.5v`	Complex multiple precision floating point

Quaternion Numbers

The input and output notation for Quaternions uses infix notation with `i`, `j`, and `k` as separators just as `i` or `J` is used in complex numbers. For example. `1i2j3k4`. One or more imaginary parts may be elided on input if zero, as in `1j3` ↔ `1i0j3k0`. By default, imaginary parts with value zero are omitted on output depending upon a program-wide preference (*Edit* | *Customize...*|*Hypercomplex Prefs*|*Display the imaginary parts of a Hypercomplex number even if 0*).

Brickman's postfix notation applies to Quaternions as well as in `3j` ↔ `0j3` ↔ `0i0j3k0`.

As with Complex numbers, the coefficients of Quaternions may be any one of the same four datatypes.

Octonion Numbers

The notation for Octonions uses infix notation with `i`, `j`, `k`, `l`, `ij`, `jk`, `kl` as separators. For example `1i2j3k4l5ij6jk7kl8`. As with Quaternions, one or more imaginary parts may be elided on input if zero, as in `1j3kl4` ↔ `1i0j3k0l0ij0jk0kl4`. By default, imaginary parts with value zero are omitted in output depending upon a program-wide preference (*Edit* | *Customize...*|*Hypercomplex Prefs*| *Display the imaginary parts of a Hypercomplex number even if 0*).

Brickman's postfix notation applies to Octonions as well as in `3kl` ↔ `0kl3` ↔ `0i0j0k0l0ij0jk0kl3`.

Rabenhorst has suggested another input/output notation for Octonions which uses a single-character separator for the leading (Quaternion) part of an Octonion and a two-character separator for the trailing part as in `1i2j3k4os5oi6oj7ok8`. This notation for input/output may be selected via the program-wide preference (*Edit | Customize...|Hypercomplex Prefs|Display Octonion numbers using Octonion Digraphs*).

As with Complex numbers and Quaternions, the coefficients of Octonions may be any of the same four datatypes.

Online Version

This paper is an ongoing effort and can be out-of-date the next day. To find the most recent version, goto http://sudleyplace.com/APL/ and look for the title of this paper on that page. Related papers such as "Hypercomplex GCD in APL"[2], "Hypercomplex Quotients in APL"[3], "Hypercomplex Numbers in APL"[5] as well as "Rational & Variable-precision Floating Point Numbers"[4] may be found in the same place.

Executable Version

All of the above APL notations may be executed in NARS2000, an experimental APL interpreter available for free as Open Source software.

The latest released version of the NARS2000 software may be found in http://www.nars2000.org/download/ in either 32- or 64-bit versions. This software runs natively under Microsoft Windows XP or later as well as any Linux or Mac OS version which supports Wine (32-bit only) which acts as a translation layer.

References

1. Personal communication, 10 August 2015
2. "Hypercomplex GCD in APL"
   http://www.sudleyplace.com/APL/HyperComplex GCD in APL.pdf
3. "Hypercomplex Quotients in APL"

http://www.sudleyplace.com/APL/HyperComplex Quotients in APL.pdf

4. "Rational & Variable-precision Floating Point Numbers", http://www.sudleyplace.com/APL/Rational & Variable-Precision FP.pdf
5. "Hypercomplex Numbers in APL" http://www.sudleyplace.com/APL/HyperComplex Numbers in APL.pdf
6. NARS2000 Forum, http://forum.nars2000.org/topic507.html, 26 Jan 2017.