The Mathematica® Journal

Computational Aspects of Quaternionic Polynomials

Part I: Manipulating, Evaluating and Factoring

M. Irene Falcão
Fernando Miranda
Ricardo Severino
M. Joana Soares

This article discusses a recently developed Mathematica tool–`QPolynomial`–a collection of functions for manipulating, evaluating and factoring quaternionic polynomials. `QPolynomial` relies on the package `QuaternionAnalysis`, which is available for download at w3.math.uminho.pt/QuaternionAnalysis.

Introduction

Some years ago, the first two authors of this article extended the standard Mathematica package implementing Hamilton's quaternion algebra—the package `Quaternions`—endowing it with the ability, among other things, to perform numerical and symbolic operations on quaternion-valued functions [1]. Later on, the same authors, in response to the need for including new functions providing basic mathematical tools necessary for dealing with quaternionic-valued functions, wrote a full new package, `QuaternionAnalysis`. Since 2014, the package and complete support files have been available for download at the Wolfram Library Archive (see also [2] for updated versions).

Over time, this package has become an important tool, especially in the work that has been developed by the authors in the area of quaternionic polynomials ([3–5]). While this work progressed, new Mathematica functions were written to appropriately deal with problems in the ring of quaternionic polynomials. The main purpose of the present article is to describe these Mathematica functions. There are two parts.

In this first part, we discuss the `QPolynomial` tool, containing several functions for treating the usual problems in the ring of quaternionic polynomials: evaluation, Euclidean division, greatest common divisor and so on. A first version of `QPolynomial` was already introduced in [4], having in mind the user's point of view. Here, we take another perspective, giving some implementation details and describing some of the experiments performed.

The second part of the article (forthcoming) is entirely dedicated to root-finding methods.

The QuaternionAnalysis Package and the Algebra of Real Quaternions

In 1843, the Irish mathematician William Rowan Hamilton introduced the quaternions, which are numbers of the form

$$q = q_0 + q_1\, i + q_2\, j + q_3\, k,\ q_i \in \mathbb{R},$$

where the imaginary units i, j and k satisfy the multiplication rules

$$i^2 = j^2 = k^2 = i\, j\, k = -1.$$

This noncommutative product generates the well-known algebra of real quaternions, usually denoted by $\mathbb{H}$.

Definition 1

In analogy with the complex case, we define:

1. *Real part of* q,

$$\mathrm{Re}\,(q) = q_0;$$

2. *Vector part of* q,

$$\mathrm{Vec}\,(q) = q_1\, i + q_2\, j + q_3\, k;$$

3. *Conjugate of* q,

$$\overline{q} = q_0 - q_1\, i - q_2\, j - q_3\, k;$$

4. *Norm of* q,

$$\| q \| = \sqrt{q\,\overline{q}} = \sqrt{\overline{q}\, q} = \sqrt{q_0{}^2 + q_1{}^2 + q_2{}^2 + q_3{}^2}\,.$$

The standard package `Quaternions` adds rules to `Plus`, `Minus`, `Times`, `Divide` and the fundamental `NonCommutativeMultiply`. Among others, the following quaternion functions are included: `Re`, `Conjugate`, `AbsIJK`, `Sign`, `AdjustedSignIJK`, `ToQuaternion`, `FromQuaternion` and `QuaternionQ`. In `Quaternions`, a quaternion is an object of the form `Quaternion[x0, x1, x2, x3]` and must have real numeric valued entries; that is, applying the function `NumericQ` to an argument gives `True`.

The extended version `QuaternionAnalysis` allows the use of symbolic entries, assuming that all symbols represent real numbers. The `QuaternionAnalysis` package adds functionality to the following functions: `Plus`, `Times`, `Divide`, `Power`, `Re`, `Conjugate`, `Dot`, `Abs`, `Norm`, `Sign` and `Derivative`. We briefly illustrate some of the quaternion functions needed in the sequel. In what follows, we assume that the package `QuaternionAnalysis` has been installed.

```
Needs["QuaternionAnalysis`"]
```

```
SetCoordinates: The coordinates system is set to {X0, X1, X2, X3}.
```

These are the imaginary units.

```
qi = Quaternion[0, 1, 0, 0];
qj = Quaternion[0, 0, 1, 0];
qk = Quaternion[0, 0, 0, 1];
```

These are the multiplication rules.

```
qj ** qj
```

```
Quaternion[-1, 0, 0, 0]
```

```
qi ** qj ** qk
```

```
Quaternion[-1, 0, 0, 0]
```

Here are two quaternions with symbolic entries and their product.

```
p = Quaternion[p0, p1, p2, p3];
q = Quaternion[q0, q1, q2, q3];
p ** q
```

```
Quaternion[p0 q0 - p1 q1 - p2 q2 - p3 q3, p1 q0 + p0 q1 - p3 q2 + p2 q3,
 p2 q0 + p3 q1 + p0 q2 - p1 q3, p3 q0 - p2 q1 + p1 q2 + p0 q3]
```

The product is noncommutative.

```
q ** p
```

```
Quaternion[p0 q0 - p1 q1 - p2 q2 - p3 q3, p1 q0 + p0 q1 + p3 q2 - p2 q3,
 p2 q0 - p3 q1 + p0 q2 + p1 q3, p3 q0 + p2 q1 - p1 q2 + p0 q3]
```

```
p ** q - q ** p
```

```
Quaternion[0, -2 p3 q2 + 2 p2 q3,
 2 p3 q1 - 2 p1 q3, -2 p2 q1 + 2 p1 q2]
```

Here are some basic functions.

```
Re[p]
```

```
p0
```

```
Vec[p]
```

```
Quaternion[0, p1, p2, p3]
```

```
Re[p] + Vec[p]
```

```
Quaternion[p0, p1, p2, p3]
```

```
Conjugate[p] // TraditionalForm
```

```
p0 - p1 𝕚 - p2 𝕛 - p3 𝕜
```

The function `Power`, which was extended in `Quaternions` through the use of de Moivre's formula for quaternions, works quite well for quaternions with numeric entries.

```
Power[Quaternion[1, 1, 0, 1], 2]
```

```
Quaternion[-1, 2, 0, 2]
```

`QuaternionAnalysis` contains a different implementation of the power function, `QPower`, which we recommend whenever a quaternion has symbolic entries.

```
QPower[p, 2]
```

$$\text{Quaternion}\left[p0^2 - p1^2 - p2^2 - p3^2,\ 2\ p0\ p1,\ 2\ p0\ p2,\ 2\ p0\ p3\right]$$

We refer the reader to the package documentation for more details on the new functions included in the package.

```
? QuaternionAnalysis`*
```

▼ QuaternionAnalysis`

AbsVec	Pk	Tks
CauchyRiemannL	PolarForm	ToComplexLike
CauchyRiemannR	PureQuaternionQ	Vec
Ck	QPower	W
ComplexLike	QuaternionToComplex	X
ComplexToQuaternion	QuaternionToComplexMatrix	X0
DiracL	QuaternionToMatrixL	X1
DiracR	QuaternionToMatrixR	X2
Laplace	R	X3
LeftMonogenicQ	RightMonogenicQ	$CoordinatesList
MonogenicQ	SetCoordinates	$Dim
Paravector	SymmetricProduct	

■ Manipulating Quaternionic Polynomials

We focus now on the polynomial P in one formal variable x of the form

$$P(x) = a_n\, x^n + a_{n-1}\, x^{n-1} + \ldots + a_1\, x + a_0,\; a_n \neq 0, \tag{1}$$

where the coefficients $a_k \in \mathbb{H}$ are to the left of the powers. Denote by $\mathbb{H}[x]$ the set of polynomials of the form (1), defining addition and multiplication as in the commutative case and assuming the variable x commutes with the coefficients. This is a ring, referred to as the ring of left one-sided (or unilateral) polynomials.

When working with the functions contained in `QPolynomial`, a polynomial $P(x)$ in $\mathbb{H}[x]$ is an object defined through the use of the function `Polynomial`, which returns the simplest form of $P(x)$, taking into account the following rules.

```
Polynomial[a_?ScalarQ] := a

Polynomial[a_Quaternion] := a

Polynomial[0 ..] = 0;

Polynomial[Longest[PatternSequence[0] ..], x__] :=
 Polynomial[x]
```

The function `ScalarQ` tests if an argument is a scalar in the sense that it is not a complex number, a quaternion number or a polynomial.

```
ScalarQ[x_] :=
 Apply[And, Head[x] =!= # & /@
   {Complex, Quaternion, Polynomial}]
```

For polynomials in $\mathbb{H}[x]$, the rules `Plus`, `Times`, `NonCommutativeMultiply` and `Power` have to be defined.

- *Addition*

```
Polynomial /: Plus[sc_?ScalarQ, p_Polynomial] :=
 Polynomial @@ Plus @@@ Transpose@PadLeft[{{sc}, List @@ p}]

Polynomial /: Plus[sc_Quaternion, p_Polynomial] :=
 Polynomial @@ Plus @@@ Transpose@PadLeft[{{sc}, List @@ p}]

Polynomial /: Plus[p1_Polynomial, p2_Polynomial] :=
 Polynomial @@
  Plus @@@ Transpose@PadLeft[{List @@ p1, List @@ p2}]
```

- *Product by a scalar*

```
Polynomial /: Times[sc_?ScalarQ, p_Polynomial] :=
 Map[sc # &, p]
```

- *Multiplication*

```
Polynomial /: NonCommutativeMultiply[sc_Quaternion,
  p_Polynomial] := Map[sc ** # &, p]

Polynomial /: NonCommutativeMultiply[p_Polynomial,
  sc_Quaternion] := Map[ # ** sc &, p]

Polynomial /: NonCommutativeMultiply[p1_Polynomial,
  p2_Polynomial] :=
 Module[{dim1 = Length@p1, dim2 = Length@p2},
  Polynomial @@
   Plus @@@
    Transpose[
     PadRight[MapThread[PadLeft,
        {Transpose[Outer[NonCommutativeMultiply,
           List @@ p1, List @@ p2]],
         Range[dim1, dim1 + dim2 - 1]}]]]]
```

- *Power*

```
Polynomial /: Power[p_Polynomial, n_Integer?Positive] :=
 Nest[NonCommutativeMultiply[p, #] &, p, n - 1]
```

Example 1

The polynomials $P(x) = x^2 + (1 + i - j)\,x + k$ and $Q(x) = x + (2\,i - j + k)$ can be defined using their coefficients in `Polynomial` in descending order.

```
px = Polynomial[1, Quaternion[1, 1, -1, 0],
   Quaternion[0, 0, 0, 1]];
qx = Polynomial[1, Quaternion[0, 2, -1, 1]];
```

Here is some arithmetic in $\mathbb{H}[x]$.

```
px + 2 qx
```

```
Polynomial[1, Quaternion[3, 1, -1, 0],
 Quaternion[0, 4, -2, 3]]
```

```
Quaternion[1, 1, 1, 1] ** px

Polynomial[Quaternion[1, 1, 1, 1],
 Quaternion[1, 3, 1, -1], Quaternion[-1, 1, -1, 1]]

px ** qx

Polynomial[1, Quaternion[1, 3, -2, 1],
 Quaternion[-3, 1, -2, 3], Quaternion[-1, 1, 2, 0]]

px^3

Polynomial[1, Quaternion[3, 3, -3, 0],
 Quaternion[-3, 6, -6, 3],
 Quaternion[-5, 1, -1, 6], Quaternion[-3, 0, 0, 1],
 Quaternion[-3, -1, 1, 0], Quaternion[0, 0, 0, -1]]
```

We now define three particularly important polynomials, the first two associated with a given polynomial P and the last one associated with a given quaternion q.

Definition 2

With P a polynomial as in equation (1) *and q a quaternion, define*:

1. *Conjugate of P*

$$\overline{P}(x) = \overline{a_n}\, x^n + \overline{a_{n-1}}\, x^{n-1} + \ldots + \overline{a_1}\, x + \overline{a_0};$$

2. *Companion polynomial of P*

$$C_P(x) = P(x)\, \overline{P}(x) = \overline{P}(x)\, P(x);$$

3. *Characteristic polynomial of q*

$$\Psi_q(x) = (x - q)\,(x - \overline{q}) = (x - \overline{q})\,(x - q) = x^2 + 2\,\mathrm{Re}(q)\, x + \|\, q\,\|^2.$$

The first two polynomials are constructed with the functions `Conjugate` and `CompanionPolynomial`.

```
Polynomial /: Conjugate[p_Polynomial] := Conjugate /@ p

CompanionPolynomial[p_Polynomial] := p ** Conjugate[p]
```

The built-in function `CharacteristicPolynomial` now accepts a quaternion argument.

```
Unprotect[CharacteristicPolynomial];
CharacteristicPolynomial[q_?ScalarQ] :=
  Polynomial[1, -q]^2;
Quaternion /: CharacteristicPolynomial[q_Quaternion] :=
 Polynomial[1, -q] ** Polynomial[1, -Conjugate[q]]
SyntaxInformation[CharacteristicPolynomial] =
  {"ArgumentsPattern" -> {_, _.}};
Protect[CharacteristicPolynomial];
```

Observe that C_P is a polynomial with real coefficients. For simplicity, in this context and in what follows, we assume that a quaternion with vector part zero is real.

```
Quaternion[a_, 0, 0, 0] := a
```

Example 2

Consider the polynomial $P(x)$ of Example 1 and the quaternion $t = 2\,i - j + k$.

```
Conjugate@px
```

```
Polynomial[1, Quaternion[1, -1, 1, 0],
 Quaternion[0, 0, 0, -1]]
```

```
CompanionPolynomial@px
```

```
Polynomial[1, 2, 3, 0, 1]
```

```
t = Quaternion[0, 2, -1, 1];
CharacteristicPolynomial[t]
```

```
Polynomial[1, 0, 6]
```

Evaluating Quaternionic Polynomials

The evaluation map at a given quaternion α, defined for the polynomial $P(x)$ given by (1), is

$$P(\alpha) = a_n\,\alpha^n + a_{n-1}\,\alpha^{n-1} + \ldots + a_1\,\alpha + a_0. \tag{2}$$

It is not an algebra homomorphism, as $P(x) = L(x)\,R(x)$ does not lead, in general, to $P(\alpha) = L(\alpha)\,R(\alpha)$, as the next theorem remarks.

Theorem 1

Let $L(x) = \sum_{i=0}^{n} a_i\,x^i$ and $R(x) = \sum_{j=0}^{m} b_j\,x^j$ be two polynomials in $\mathbb{H}[x]$ and consider the polynomial $P(x) = L(x)\,R(x)$ and $\alpha \in \mathbb{H}$. Then:

1. $P(\alpha) = \sum_{i=0}^{n} a_i\,R(\alpha)\,\alpha^i$.

2. *If* $R(\alpha) = 0$, *then* $P(\alpha) = 0$.

3. *If* $R(\alpha) \neq 0$, *then* $P(\alpha) = L(\tilde{\alpha})\,R(\alpha)$, *where* $\tilde{\alpha} = R(\alpha)\,\alpha\,(R(\alpha))^{-1}$.

4. *If* $L(x)$ *is a real polynomial*, *then* $P(\alpha) = R(\alpha)\,L(\alpha)$.

5. *If* $\alpha \in \mathbb{R}$, *then* $P(\alpha) = L(\alpha)\,R(\alpha)$.

As usual, we say that α is a zero (or root) of $P(x)$ if $P(\alpha) = 0$. An immediate consequence of Theorem 1 is that if $R(\alpha) \neq 0$, then α is a zero of $P(x)$ if and only if $R(\alpha)\,\alpha\,(R(\alpha))^{-1}$ is a zero of $L(x)$.

A straightforward implementation of equation (2) can be obtained through `Eval`.

```
Eval[p_Polynomial] :=
 Plus @@ MapThread[NonCommutativeMultiply,
    {List @@ p,
     Reverse@
      Function[x, NestList[(# ** x) &, 1, Length@p - 1]][#]}] &

Eval[p_Polynomial, x_] := Eval[p][x]
```

As in the classical (real or complex) case, the evaluation of a polynomial can also be obtained by the use of Horner's rule [3]. The nested form of equation (2) is

$$P(\alpha) = ((\ldots(a_n\,\alpha + a_{n-1})\,\alpha + \ldots)\,\alpha + a_1)\,\alpha + a_0,$$

and the quaternionic version of Horner's rule can be implemented as `HornerEval`.

```
HornerEval[p_Polynomial, x_] :=
 Fold[NonCommutativeMultiply[#1, x] + #2 &, 0, p]
```

Example 3

Consider again the polynomial $P(x) = x^2 + (1 + i - j)\,x + k$. The problem of evaluating $P(x)$ at $t = 2\,i - j + k$ can be solved through one of the following (formally) equivalent expressions.

```
Eval[px, t]

Quaternion[-9, 1, -2, 3]

HornerEval[px, t]

Quaternion[-9, 1, -2, 3]
```

Example 4

We now illustrate some of the conclusions of Theorem 1 by considering the polynomials $L(x) = x - i + k$, $R(x) = x - 1 - j + k$ and $S(x) = x - 2$ and the quaternion $u = i - k$.

```
lx = Polynomial[1, Quaternion[0, -1, 0, 1]];
rx = Polynomial[1, Quaternion[-1, 0, -1, 1]];
sx = Polynomial[1, -2];
u = Quaternion[0, 1, 0, -1];

px1 = lx ** rx;
Eval[px1][u] === Eval[lx][u] ** Eval[rx][u]

False

Eval[px1][Abs@u] === Eval[lx][Abs@u] ** Eval[rx][Abs@u]

True

px2 = sx ** rx;
Eval[px2][u] === Eval[rx][u] ** Eval[sx][u]

True
```

The Euclidean Algorithm

For the theoretical background of this section, we refer the reader to [6] (see also [7] where basic division algorithms in $\mathbb{H}[x]$ are presented). Since $\mathbb{H}[x]$ is a principal ideal domain, left and right division algorithms can be defined. The following theorem gives more details.

Theorem 2—Euclidean division

If $P_1(x)$ and $P_2(x)$ are polynomials in $\mathbb{H}[x]$ (with $0 < \deg P_2 \leq \deg P_1$), then there exist unique $Q_{\text{left}}(x)$, $R_{\text{left}}(x)$, $Q_{\text{right}}(x)$ and $R_{\text{right}}(x)$ such that

$$P_1(x) = Q_{\text{left}}(x)\, P_2(x) + R_{\text{left}}(x) \tag{3}$$

and

$$P_1(x) = P_2(x)\, Q_{\text{right}}(x) + R_{\text{right}}(x), \tag{4}$$

with $\deg R_{\text{left}} \leq \deg P_2$ *and* $\deg R_{\text{right}} \leq \deg P_2$.

If in equation (3), $R_{\text{left}}(x) = 0$, then $P_2(x)$ is called a right divisor of $P_1(x)$, and if in equation (4), $R_{\text{right}}(x) = 0$, $P_2(x)$ is called a left divisor of $P_1(x)$. This article only presents right versions of the division functions; in `QPolynomial` both the left and right versions are implemented. The function `PolynomialDivisionR` performs the right division of two quaternionic polynomials, returning a list with the quotient and remainder of the division.

```
PolynomialDivisionR[p1_Polynomial, sc_?ScalarQ] :=
 {p1 / sc, 0}

PolynomialDivisionR[p1_Polynomial, sc_Quaternion] :=
 {p1 ** (1 / sc), 0}

PolynomialDivisionR[p1_Polynomial, p2_Polynomial] :=
 Module[{tt, qq = 0, rr = p1, degree = Length@p1 - Length@p2},
  While[
   Head@rr === Polynomial && degree ≥ 0,
   tt = Polynomial @@
     (PadRight[{First@rr ** (1 / First@p2)}, degree + 1]);
   qq = qq + tt;
   rr = rr - tt ** p2;
   degree = Length[rr] - Length[p2]
  ];
  {qq, rr}]
```

Example 5

Consider the polynomials $P_1(x) = x^2 + (-1 + i - k)\, x + 2 + 2\, j + 2\, k$ and $P_2(x) = x - 2\, k$.

```
px1 = Polynomial[1, Quaternion[-1, 1, 0, -1],
   Quaternion[2, 0, 2, 2]];
px2 = Polynomial[1, Quaternion[0, 0, 0, -2]];
PolynomialDivisionR[px1, px2]
```

```
{Polynomial[1, Quaternion[-1, 1, 0, 1]], 0}
```

Since $R(x) = 0$, $P_2(x)$ is a right divisor of $P_1(x)$ and $P_1(x) = (x - 1 + i + k)(x - 2k)$. On the other hand, $P_3(x) = x - 1 + i + k$ does not right-divide $P_1(x)$ (but it is a left divisor).

```
px3 = Polynomial[1, Quaternion[-1, 1, 0, 1]];
PolynomialDivisionR[px1, px3]
```

```
{Polynomial[1, Quaternion[0, 0, 0, -2]],
 Quaternion[0, 0, 4, 0]}
```

The greatest common (right or left) divisor polynomial of two polynomials can now be computed using the Euclidean algorithm by a basic procedure similar to the one used in the complex setting. The function `GCDR` implements this procedure for the case of the greatest common right divisor.

```
GCDR[a_, 0] = a;

GCDR[a_, b_] := GCDR[b, Last[PolynomialDivisionR[a, b]]]

GCDR[a_, b_, c__] := GCDR[GCDR[a, b], c]

PolynomialGCDR[a_, b_, c___] := PNormalizeL[GCDR[a, b, c]]
```

Here `PNormalizeL` is defined as follows.

```
PNormalizeL[0] = 0;

PNormalizeL[sc_?ScalarQ] := (1 / sc) ** sc

PNormalizeL[sc_Quaternion] := 1

PNormalizeL[p_Polynomial] := (1 / First@p) ** p
```

Example 5 (continued)

$\mathrm{GCDR}(P_1, P_2) = P_2$ and $\mathrm{GCDR}(P_1, P_2, P_3) = 1$.

```
PolynomialGCDR[px1, px2]
```

```
Polynomial[1, Quaternion[0, 0, 0, -2]]
```

The Zero Structure in $\mathbb{H}(x)$

Before describing the zero set $\mathbb{Z}_P$ of a quaternionic polynomial P, we need to introduce more concepts.

Definition 3

We say that a quaternion q is congruent (or similar) to a quaternion r (and write $q \sim r$) if there exists a nonzero quaternion h such that $r = h\,q\,h^{-1}$.

This is an equivalence relation in $\mathbb{H}[x]$ that partitions $\mathbb{H}[x]$ into congruence classes. The congruence class containing a given quaternion q is denoted by $[q]$. It can be shown (see, e.g. [8]) that

$$[q] = \{r \in \mathbb{H} : \operatorname{Re} q = \operatorname{Re} r \text{ and } \|r\| = \|q\|\}.$$

This result gives a simple way to test if two or more quaternions are similar, implemented with the function `SimilarQ`.

```
SimilarQ[q_Quaternion, r_Quaternion] :=
 ZeroQ[Re@q - Re@r] && ZeroQ[Norm@q - Norm@r]
```

For zero or equality testing, we use the `ZeroQ` test function.

```
ZeroQ[a_] := PossibleZeroQ[a, Method → "ExactAlgebraics"]
```

```
ZeroQ[q_Quaternion] := And @@ ZeroQ /@ q
```

```
q5 = Quaternion[1, 2, 3, 4];
q6 = Quaternion[1, 3, 4, 2];
q7 = Quaternion[-1, 2, 3, 4];
SimilarQ[q5, q6]
```

```
True
```

```
SimilarQ[q6, q7]
```

```
False
```

It follows that $[q] = \{q\}$ if and only if $q \in \mathbb{R}$. The congruence class of a nonreal quaternion $q = q_0 + q_1 i + q_2 j + q_3 k$ can be identified with the three-dimensional sphere in the hyperplane $\{(x_0, x_1, x_2, x_3) \in \mathbb{R}^4 : x_0 = q_0\}$ with center $(q_0, 0, 0, 0)$ and radius $\sqrt{q_1{}^2 + q_2{}^2 + q_3{}^2}$.

Definition 4

A zero q of $P \in \mathbb{H}[x]$ is called an isolated zero of P if $[q]$ contains no other zeros of P. Otherwise, q is called a spherical zero of P and $[q]$ *is referred to as a sphere of zeros.*

It can be proved that if q is a zero that is not isolated, then all quaternions in $[q]$ are in fact zeros of P (see Theorem 4); therefore the choice of the term spherical to designate this type of zero is natural. According to the definition, real zeros are always isolated zeros. Identifying zeros can be done, taking into account the following results.

Theorem 3 ([9])

Let $P \in \mathbb{H}[x]$ and $\alpha \in \mathbb{H}$. The following conditions are equivalent:

1. *There is an $\alpha' \in [\alpha]$ such that $P(\alpha') = 0$.*

2. *The characteristic polynomial of α, Ψ_α, is a divisor of the companion polynomial C_P of P.*

3. *α is a root of C_P.*

Theorem 4 ([9], [10])

A nonreal zero α is a spherical zero of $P \in \mathbb{H}[x]$ if and only if any of the following equivalent conditions hold:

1. α *and* $\overline{\alpha}$ *are both zeros of* P.

2. $[\alpha] \subset \mathbb{Z}_P$.

3. *The characteristic polynomial* Ψ_α *of* α *is a right divisor of* P; *that is*, *there exists a polynomial* $Q \in \mathbb{H}[x]$ *such that* $P(x) = Q(x)\,\Psi_\alpha(x)$.

Example 6

We are going to show that the polynomial

$$P(x) = x^3 + (1 - i + j)\,x^2 + 2\,x + 2 - 2\,i + 2\,j$$

has a spherical zero: $i + j$ and an isolated one: $-1 + i - j$.

```
px4 = Polynomial[1, Quaternion[1, -1, 1, 0], 2,
   Quaternion[2, -2, 2, 0]];
sph = Quaternion[0, 1, 1, 0];
iso = Quaternion[-1, 1, -1, 0];
```

We first observe that both `sph` and `iso` are zeros of P.

```
Eval[px4][sph]
```

```
0
```

```
Eval[px4][iso]
```

```
0
```

Now we use Theorem 4-1 to conclude that the zero `sph` is spherical, while the zero `iso` is isolated.

```
Eval[px4][Conjugate@sph]
```

```
0
```

```
Eval[px4][Conjugate@iso]
```

```
Quaternion[8, -2, 2, 0]
```

We can reach the same conclusion from Theorem 4-3.

```
PolynomialDivisionR[px4, CharacteristicPolynomial[sph]]
```

```
{Polynomial[1, Quaternion[1, -1, 1, 0]], 0}
```

```
PolynomialDivisionR[px4, CharacteristicPolynomial[iso]]
```

```
{Polynomial[1, Quaternion[-1, -1, 1, 0]],
 Polynomial[Quaternion[1, 2, -2, 0], Quaternion[5, 1, -1, 0]]}
```

Taking all this into account, the verification of the nature of a zero can be done using the function `SphericalQ`.

```
SphericalQ[p_Polynomial, q_Quaternion] :=
 ZeroQ[Eval[p][q]] && ZeroQ[Eval[p][Conjugate@q]]
```

Consider the same polynomial and quaternions again.

```
SphericalQ[px4, sph]
```

```
True
```

```
SphericalQ[px4, iso]
```

```
False
```

We now list other results needed in the next section.

Theorem 5—Factor theorem ([11], [12])

Let $P \in \mathbb{H}[x]$ *and* $\alpha \in \mathbb{H}$. *Then* α *is a zero of* P *if and only if there exists* $Q \in \mathbb{H}[x]$ *such that* $P(x) = Q(x)(x - \alpha)$.

Theorem 6—Fundamental theorem of algebra ([13])

Any nonconstant polynomial in $\mathbb{H}[x]$ *always has a zero in* $\mathbb{H}$.

Factoring

In this section, we address the problem of factoring a polynomial P. We mostly follow [4]. As in the classical case, it is always possible to write a quaternionic polynomial as a product of linear factors; however the link between these factors and the corresponding zeros is not straightforward. As an immediate consequence of Theorems 5 and 6, one has the following theorem.

Theorem 7—Factorization into linear terms

Any monic polynomial P *of degree* $n \geq 1$ *in* $\mathbb{H}[x]$ *factors into linear factors*; *that is*, *there exist* $x_1, x_2, \ldots, x_n \in \mathbb{H}$ *such that*

$$P(x) = (x - x_n)(x - x_{n-1}) \ldots (x - x_1). \tag{5}$$

Definition 5

In a factorization of P *of the form* (5), *the quaternions* $x_1, x_2, \ldots, x_n$ *are called factor terms of* P *and the n-tuple* $(x_1, x_2, \ldots, x_n)$ *is called a factor terms chain associated with* P *or simply a chain of* P.

If $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$ are chains associated with the same polynomial P, then we say that the chains are similar and write $(x_1, x_2, \ldots, x_n) \sim (y_1, y_2, \ldots, y_n)$.

The function `PolynomialFromChain` constructs a polynomial with a given chain, and the function `SimilarChainQ` checks if two given chains are similar.

```
PolynomialFromChain[list_List] :=
 NonCommutativeMultiply @@
  (Polynomial[1, -#1] &) /@ Reverse[list]

ZeroQ[p_Polynomial] := And @@ (ZeroQ /@ p)

SimilarChainQ[c1_List, c2_List] :=
 ZeroQ[PolynomialFromChain[c1] - PolynomialFromChain[c2]]
```

The repeated use of the next result allows the constructions of similar chains, if any.

Theorem 8

Let $(x_1, x_2, \ldots, x_{l-1}, x_l, \ldots, x_n)$ *be a chain of a polynomial* P. *If* $h = \overline{x_l} - x_{l-1} \neq 0$, *then*

$(x_1, \ldots, h^{-1} x_l h, h^{-1} x_{l-1} h, \ldots, x_n) \sim (x_1, x_2, \ldots, x_{l-1}, x_l, \ldots \mathrm{x}_n)$.

Theorem 8 can be implemented using the function `FactorShift`.

```
FactorShift[{q1_, q2_}] := Module[{
    h = (Conjugate@q2 - q1)},
   If[ZeroQ@h, {q2, q1}, {(1 / h) ** q2 ** h, (1 / h) ** q1 ** h}]
  ]

FactorShift[l_List, m_Integer?Positive,
   n_Integer?Positive] := Module[
   {lista = l, dim = Length@l, seq},
   If[m > dim || n > dim, Message[FactorShift::args],
    seq = Sort /@ Partition[Range[m, n, Sign[n - m]], 2, 1];
    Map[(Part[lista, #] = FactorShift[Part[lista, #]]) &,
     seq]];
   lista
  ]

FactorShift::args = "Chain too short.";
```

Example 7

This constructs chains similar to the chain $(1 + i + k, -1 + j, i + j + k)$.

```
c1 = {Quaternion[1, 1, 0, 1], Quaternion[-1, 0, 1, 0],
    Quaternion[0, 1, 1, 1]};

c2 = FactorShift[c1, 3, 1]
```

$\left\{\text{Quaternion}\left[0, \frac{43}{27}, \frac{7}{27}, \frac{17}{27}\right],\right.$
$\left.\text{Quaternion}\left[1, \frac{23}{189}, \frac{248}{189}, \frac{97}{189}\right], \text{Quaternion}\left[-1, \frac{2}{7}, \frac{3}{7}, \frac{6}{7}\right]\right\}$

```
c3 = FactorShift[c1, 1, 3]
```

$$\left\{\text{Quaternion}\left[-1, \frac{6}{7}, \frac{3}{7}, -\frac{2}{7}\right], \text{Quaternion}\left[0, \frac{17}{35}, \frac{19}{35}, \frac{11}{7}\right], \text{Quaternion}\left[1, \frac{23}{35}, \frac{36}{35}, \frac{5}{7}\right]\right\}$$

Observe that `c1`, `c2` and `c3` are similar chains.

```
SimilarChainQ[c1, c2]

True

SimilarChainQ[c2, c3]

True
```

We emphasize that there are polynomials with just one chain. This issue is addressed in Theorem 12. For the moment, we just give an example of such a polynomial.

```
c4 = {x4 = Quaternion[0, 1, 0, 0], x5 = Quaternion[0, 0, 1, 0],
   x6 = Quaternion[0, 0, 0, 1]};
FactorShift[c4, 3, 2]

{Quaternion[0, 1, 0, 0],
 Quaternion[0, 0, 1, 0], Quaternion[0, 0, 0, 1]}

FactorShift[c4, 1, 3]

{Quaternion[0, 1, 0, 0],
 Quaternion[0, 0, 1, 0], Quaternion[0, 0, 0, 1]}

PolynomialFromChain[c4]

Polynomial[1, Quaternion[0, -1, -1, -1],
 Quaternion[0, -1, 1, -1], -1]

Polynomial[1, -x6] ** Polynomial[1, -x5] ** Polynomial[1, -x4]

Polynomial[1, Quaternion[0, -1, -1, -1],
 Quaternion[0, -1, 1, -1], -1]
```

These computations lead us to the conclusion that the polynomial $x^3 - (i + j + k)\,x^2 - (i - j + k)\,x - 1$ factors uniquely as $(x - k)\,(x - j)\,(x - i)$.

The next fundamental results shed light on the relation between factor terms and zeros of a quaternionic polynomial.

Theorem 9 ([12–14])

Let $(x_1, x_2, \ldots, x_n)$ be a chain of the polynomial P. Then every zero of P is similar to some factor term x_k in the chain and conversely, every factor term x_k is similar to some zero of P.

Theorem 10—Zeros from factors ([12])

Consider a chain $(x_1, x_2, \ldots, x_n)$ *of the polynomial* P. *If the similarity classes* $[x_k]$ *are distinct, then* P *has exactly* n *zeros* ζ_k, *which are given by*:

$$\zeta_k = \overline{\mathcal{P}_k}(x_k)\, x_k \big(\overline{\mathcal{P}_k}(x_k)\big)^{-1};\ k = 1, \ldots, n, k = 1, \ldots, n,$$

where

$$\mathcal{P}_k(x) = \begin{cases} 1 & \text{if } k = 1, \\ (x - x_{k-1}) \ldots (x - x_1) & \text{otherwise}. \end{cases} \tag{6}$$

The function `ZerosFromChain` determines the zeros of a polynomial with a prescribed chain in the case where no two factors in the chain are similar quaternions, giving a warning if this condition does not hold.

```
ZerosFromChain[fact_List] := Module[
  {n = Length@fact, factors = fact, roots = {}, RPoli, RPoliz},
  If[n > Length@Union@Map[{Re@#, Norm@#} &, fact],
   Message[ZerosFromChain::args],
   For[i = 1, i ≤ n, i++,
    RPoli = RPol[factors, i];
    RPoliz = Eval[RPoli][factors⟦i⟧];
    AppendTo[roots, RPoliz ** factors⟦i⟧ ** (1 / RPoliz)];
   ];
   roots
  ]]

ZerosFromChain::args =
  "Arguments in the same similarity class.";

RPol[fact_, i_] := Module[
  {n = Length@fact},
  Which[
    i == 1, Polynomial[1],
    i == 2, Polynomial[1, -Conjugate@#] & @@
     (Drop[fact, i - n - 1]),
    i ≤ n,
    (NonCommutativeMultiply @@
      (Polynomial[1, -Conjugate@#] & /@
        (Drop[fact, i - n - 1])))
   ];
 ]
```

Example 8

Consider the polynomial $P(x) = (x - i + j - k)\,(x + 2\,k)\,(x - 1)\,(x + 1 - i + j)$. One of its chains is $(-1 + i - j, 1, -2\,k, i - j + k)$, and it follows at once that the similarity classes of the factor terms are all distinct. Therefore, we conclude from Theorem 10 that P has four distinct isolated roots, which can be obtained with the following code.

```
ZerosFromChain[{Quaternion[-1, 1, -1, 0], 1,
  Quaternion[0, 0, 0, -2], Quaternion[0, 1, -1, 1]}]

{Quaternion[-1, 1, -1, 0], 1,
 Quaternion[0, 0, 0, -2], Quaternion[0, 1, -1, 1]}
```

On the other hand, the polynomial $P(x) = (x - i)(x + 2k)(x - j)$ has $(j, -2k, i)$ as one of its chains. Since $[i] = [j]$, one cannot apply Theorem 10 to find the roots of P.

```
ZerosFromChain[
 c5 = {Quaternion[0, 0, 1, 0], Quaternion[0, 0, 0, -2],
   Quaternion[0, 1, 0, 0]}]
```

```
ZerosFromChain: Arguments in the same similarity class.
```

Observe that this does not mean that the roots of P are spherical.

```
px5 = PolynomialFromChain[c5]
```

```
Polynomial[1, Quaternion[0, -1, -1, 2],
 Quaternion[0, 2, 2, 1], 2]
```

```
Eval[px5][First@c5]
```

```
0
```

```
SphericalQ[px5, First@c5]
```

```
False
```

This issue will be resumed later in connection with the notion of the multiplicity of a zero. The following theorem indicates how, under certain conditions, one can construct a polynomial having prescribed zeros.

Theorem 11—Factors from zeros ([9])

If $\zeta_1, \ldots, \zeta_n$ are quaternions such that the similarity classes $[\zeta_k]$ are distinct, then there is a unique polynomial P of degree n with zeros $\zeta_1, \ldots, \zeta_n$ that can be constructed from the chain $(x_1, x_2, \ldots, x_n)$, where

$$x_k = \mathcal{P}_k(\zeta_k)\, \zeta_k (\mathcal{P}_k(\zeta_k))^{-1},\ k = 1, \ldots, n,$$

where $\mathcal{P}_k$ is the polynomial (6).

The function `ChainFromZeros` implements the procedure described in Theorem 11.

```
ChainFromZeros[root_List] := Module[
  {n = Length@root, factor, factors, QPoli = 1, QPoliz},
  If[n > Length@Union@Map[{Re@#, Norm@#} &, root],
   Message[ChainFromZeros::args1],
   factor = First@root; factors = {factor};
   For[i = 2, i ≤ n, i++,
    QPoli = Polynomial[1, -factor] ** QPoli;
    QPoliz = Eval[QPoli][root[[i]]];
    factor = QPoliz ** root[[i]] ** (1 / QPoliz);
    AppendTo[factors, factor];
   ];
   factors
  ]]
```

```
ChainFromZeros::args1 =
  "Arguments in the same similarity class. Use an
   alternative syntax.";
```

Example 9

Consider the problem of constructing a polynomial having the isolated roots $i,\ 1+i+k,\ -1+3\,j$. We first determine one chain associated with these zeros.

```
chain = ChainFromZeros[
  roots = {Quaternion[0, 1, 0, 0], Quaternion[1, 1, 0, 1],
    Quaternion[-1, 0, 3, 0]}]
```

$$\left\{\text{Quaternion}[0,\ 1,\ 0,\ 0],\ \text{Quaternion}[1,\ 0,\ 1,\ 1],\ \text{Quaternion}\left[-1,\ -\frac{94}{33},\ \frac{31}{33},\ \frac{2}{33}\right]\right\}$$

Now we determine the polynomial associated with this chain.

```
px6 = PolynomialFromChain[chain]
```

$$\text{Polynomial}\left[1,\ \text{Quaternion}\left[0,\ \frac{61}{33},\ -\frac{64}{33},\ -\frac{35}{33}\right],\ \text{Quaternion}\left[\frac{28}{33},\ -\frac{65}{33},\ \frac{127}{33},\ -\frac{63}{11}\right],\ \text{Quaternion}\left[-\frac{65}{33},\ 2,\ \frac{125}{33},\ \frac{92}{33}\right]\right]$$

Check the solution.

```
Eval[px6][roots]
```

```
{0, 0, 0}
```

Theorem 12 ([9–15])

Let P be a quaternionic polynomial of degree n. Then $x_1 \in \mathbb{H}\setminus\mathbb{R}$ is the unique zero of P if and only if P admits a unique chain $(x_1, x_2, \ldots, x_n)$ with the property

$$x_l \in [x_1] \text{ and } x_l \neq \overline{x}_{l-1}, \tag{7}$$

for all $l = 2, \ldots, n$.

Moreover, if a chain $(x_1, x_2, \ldots, x_n)$ associated with a polynomial P has property (7), Q is a polynomial of degree m such that $y_1 \in \mathbb{H}\setminus\mathbb{R}$ is its unique zero and $y_1 \notin [x_1]$, then the polynomial $P\,Q$ (of degree $m+n$) has only two zeros, namely x_1 and $\overline{P}(y_1)\,y_1\big(\overline{P}(y_1)\big)^{-1}$.

We can now introduce the concept of the multiplicity of a zero and a new *kind* of zero. In this context, we have to note that several notions of multiplicity are available in the literature (see [9], [15–17]).

Definition 6

The multiplicity of a zero q of P is defined as the maximum degree of the right factors of P with q as their unique zero and is denoted by $m_P(q)$. The multiplicity of a sphere of zeros $[q]$ of P, denoted by $m_p([q])$, is the largest $k \in \mathbb{N}_0$ for which $\Psi_q^{\,k}$ divides P.

A zero q of P is called a mixed zero if $m_p([q]) > 0$ and $m_P(q) > m_P(q')$ for all $q' \in [q]$.

Example 10

The polynomial $P(x) = (x - k)(x - j)(x - 1 + i)$ has an isolated root $q_1 = 1 - i$ with multiplicity $m_P(q_1) = 1$ and an isolated root $q_2 = j$ with multiplicity $m_P(q_2) = 1$.

The polynomial $P(x) = (x + j)(x - j)(x - 1 + i)$ has an isolated root $q_1 = 1 - i$ with multiplicity $m_P(q_1) = 1$ and a sphere of zeros $[q_2] = [j]$ with multiplicity $m_P([q_2]) = 1$.

The polynomial $P(x) = (x + j)(x - j)^2$ has a mixed root $q = j$ with multiplicity $m_P(q) = 2$ and $m_P([q]) = 1$.

Finally, one can construct a polynomial with assigned zeros by the repeated use of the following result.

Theorem 13 ([4])

A polynomial with ζ_1 and ζ_2 as its isolated zeros with multiplicities m and n, respectively, and a sphere of zeros $[\zeta_s]$ with multiplicity k can be constructed through the chain

$$\big(\overbrace{\zeta_1, \ldots, \zeta_1}^{m}, \overbrace{\tilde{\zeta}_2, \ldots, \tilde{\zeta}_2}^{n}, \overbrace{\zeta_s, \overline{\zeta_s}, \ldots, \zeta_s, \overline{\zeta_s}}^{2k}\big)$$

where $\tilde{\zeta}_2 = Q(\zeta_2)\,\zeta_2(Q(\zeta_2))^{-1}$ and $Q(x) = (x - \zeta_1)^m$.

An alternative syntax for the function `ChainFromZeros` addresses the problem of constructing a polynomial (in fact it constructs a chain) once one knows the nature and multiplicity of its roots.

```
ChainFromZeros[{}, {}] := {}

ChainFromZeros[isoroots : {{_, _} ..}, {}] := Module[
  {nisoroots = Length@isoroots, factor, multiplicity,
   factors, QPoli = 1, QPoliz},
  If[
   nisoroots >
    Length@Union@Map[{Re@#, Norm@#} &,
       First@Transpose@isoroots],
   Message[ChainFromZeros::args2],
   factor = isoroots⟦1, 1⟧;
   multiplicity = isoroots⟦1, 2⟧;
   factors = Table[factor, {multiplicity}];
   For[i = 2, i ≤ nisoroots, i++,
    QPoli = Power[Polynomial[1, -factor], multiplicity] **
      QPoli;
    QPoliz = Eval[QPoli][isoroots⟦i, 1⟧];
    factor = QPoliz ** isoroots⟦i, 1⟧ ** (1 / QPoliz);
    multiplicity = isoroots⟦i, 2⟧;
    factors = Join[factors, Table[factor, {multiplicity}]];
   ];
   factors
  ]
 ]
```

```
ChainFromZeros[{}, sphroots : {{_, _} ..}] := Module[
  {nsphroots = Length@sphroots},
  If[nsphroots > Count[sphroots, {_Quaternion, _}],
   Message[ChainFromZeros::args4],
   If[nsphroots >
     Length@Union@Map[{Re@#, Norm@#} &,
        First@Transpose@sphroots],
    Message[ChainFromZeros::args3], Null];
   Flatten@(Table[{#1, Conjugate@#1}, {#2}] & @@@ sphroots)
  ]
 ]

ChainFromZeros[isoroots : {{_, _} ...},
  sphroots : {{_, _} ...}] := Module[
  {ch1 = ChainFromZeros[isoroots, {}],
   ch2 = ChainFromZeros[{}, sphroots]},
  If[ch1 =!= Null && ch2 =!= Null, Join[ch1, ch2], Null]
 ]

ChainFromZeros::args2 =
  "Two or more isolated zeros are in the same
    similarity class.";
ChainFromZeros::args3 =
  "Two or more spherical zeros are in the same
    similarity class.";
ChainFromZeros::args4 = "Spherical zeros must be nonreals.";
```

Example 11

We reconsider here Example 6 of [4]. An example of a polynomial P that has $\zeta_1 = i$ as a zero of multiplicity three, $\zeta_2 = -1 + j + k$ as a zero of multiplicity two and $[2+i]$ as a sphere of zeros with multiplicity two is

$$P(x) = \Psi_{2+i}{}^2 (x - x_2)^2 \, (x - x_1)^3,$$

where $x_1 = \zeta_1 = i$, $x_2 = Q(\zeta_2)\,\zeta_2(Q(\zeta_2))^{-1}$ and $Q(x) = (x - x_1)^3$; that is,

$$P(x) = (x - 2 - i)^2 \, (x - 2 + i)^2 \left(x + 1 + \frac{7}{5} i + \frac{1}{5} j\right)^2 (x - i)^3.$$

Of course this solution is not unique. For example, the polynomial

$$Q(x) = (x - 2 - i)^2 \, (x - 2 + i)^2 \, (x + 1 - j - k) \left(x + 1 + \frac{7}{5} i + \frac{1}{5} j\right) (x - k)\,(x - j)\,(x - i)$$

solves the same problem.

We confirm this using the function `ChainFromZeros` with the new syntax.

```
ChainFromZeros[{{Quaternion[0, 1, 0, 0], 3},
   {Quaternion[-1, 0, 1, 1], 2}},
  {{Quaternion[2, 1, 0, 0], 2}}]
```

```
{Quaternion[0, 1, 0, 0],
 Quaternion[0, 1, 0, 0], Quaternion[0, 1, 0, 0],
 Quaternion[-1, -7/5, -1/5, 0], Quaternion[-1, -7/5, -1/5, 0],
 Quaternion[2, 1, 0, 0], Quaternion[2, -1, 0, 0],
 Quaternion[2, 1, 0, 0], Quaternion[2, -1, 0, 0]}
```

```
ChainFromZeros[{{Quaternion[0, 1, 0, 0], 1}},
  {{Quaternion[0, 1, 0, 0], 1}}]
```

```
{Quaternion[0, 1, 0, 0],
 Quaternion[0, 1, 0, 0], Quaternion[0, -1, 0, 0]}
```

```
PolynomialFromChain[%]
```

```
Polynomial[1, Quaternion[0, -1, 0, 0],
 1, Quaternion[0, -1, 0, 0]]
```

Here are two spherical roots corresponding to the same sphere.

```
ChainFromZeros[{},
  {{Quaternion[0, 1, 0, 0], 1}, {Quaternion[0, 0, 1, 0], 1}}]
```

ChainFromZeros: Two or more spherical zeros are in the same similarity class.

```
{Quaternion[0, 1, 0, 0], Quaternion[0, -1, 0, 0],
 Quaternion[0, 0, 1, 0], Quaternion[0, 0, -1, 0]}
```

```
PolynomialFromChain[%]
```

```
Polynomial[1, 0, 2, 0, 1]
```

Observe that the result is, of course, the same as this one.

```
PolynomialFromChain[
  ChainFromZeros[{}, {{Quaternion[0, 1, 0, 0], 2}}]]
```

```
Polynomial[1, 0, 2, 0, 1]
```

Recall that a real root is always an isolated root, and two roots in the same congruence class cannot be isolated.

```
ChainFromZeros[{{Quaternion[0, 1, 0, 0], 2},
   {Quaternion[0, 0, 1, 0], 2}},
  {{Quaternion[1, 1, 0, 0], 3}, {2, 2}}]
```

```
ChainFromZeros: Two or more isolated zeros are in the same similarity
      class.
```

```
ChainFromZeros: Spherical zeros must be nonreals.
```

Conclusion

This article has discussed implementation issues related to the manipulation, evaluation and factorization of quaternionic polynomials. We recommend that interested readers download the support file `QPolynomial.m` to get complete access to all the implemented functions. The increasing interest in the use of quaternions in areas such as number theory, robotics, virtual reality and image processing [18] makes us believe that developing a computational tool for operating in the quaternions framework will be useful for other researchers, especially taking into account the power of Mathematica as a symbolic language.

In the ring of quaternionic polynomials, new problems arise mainly because the structure of zero sets, as we have described, is very different from the complex case. In this article, we did not discuss the problem of computing the roots or the factor terms of a polynomial; all the results we have presented assumed that either the zeros or the factor terms of a given polynomial are known. Methods for computing the roots or factor terms of a quaternionic polynomial are considered in Part II.

Acknowledgments

Research at the Centre of Mathematics at the University of Minho was financed by Portuguese Funds through FCT - Fundação para a Ciência e a Tecnologia, within the Project UID/MAT/00013/2013. Research at the Economics Politics Research Unit was carried out within the funding with COMPETE reference number POCI-01-0145-FEDER-006683 (UID/ECO/03182/2013), with the FCT/MEC's (Fundação para a Ciência e a Tecnologia, I.P.) financial support through national funding and by the European Regional Development Fund through the Operational Programme on "Competitiveness and Internationalization - COMPETE 2020" under the PT2020 Partnership Agreement.

References

[1] M. I. Falcão and F. Miranda, "Quaternions: A Mathematica Package for Quaternionic Analysis," in *Computational Science and Its Applications (ICCSA 2011)*, *Lecture Notes in Computer Science*, **6784** (B. Murgante, O. Gervasi, A. Iglesias, D. Taniar and B. O. Apduhan, eds.), Berlin, Heidelberg: Springer, 2011 pp. 200–214. doi:10.1007/978-3-642-21931-3_17.

[2] F. Miranda and M. I. Falcão. "QuaternionAnalysis Mathematica Package." w3.math.uminho.pt/QuaternionAnalysis.

[3] M. I. Falcão, F. Miranda, R. Severino and M. J. Soares, "Evaluation Schemes in the Ring of Quaternionic Polynomials," *BIT Numerical Mathematics*, **58**(1), pp. 51–72. doi:10.1007/s10543-017-0667-8.

[4] M. I. Falcão, F. Miranda, R. Severino and M. J. Soares, "Mathematica Tools for Quaternionic Polynomials," in *Computational Science and Its Applications (ICCSA 2017)*, *Lecture Notes in Computer Science*, **10405**, (O. Gervasi, B. Murgante, S. Misra, G. Borruso, C. M. Torre, A. M. A. C. Rocha, D. Taniar, B. O. Apduhan, E. Stankova and A. Cuzzocrea, eds.), Berlin, Heidelberg: Springer, 2017 pp. 394–408. doi:10.1007/978-3-319-62395-5_27.

[5] M. I. Falcão, F. Miranda, R. Severino and M. J. Soares, "Weierstrass Method for Quaternionic Polynomial Root-Finding," *Mathematical Methods in the Applied Sciences*, 2017 pp. 1–15. doi:10.1002/mma.4623.

[6] N. Jacobson, *The Theory of Rings (Mathematical Surveys and Monographs)*, New York: American Mathematical Society, 1943.

[7] A. Damiano, G. Gentili and D. Struppa, "Computations in the Ring of Quaternionic Polynomials," *Journal of Symbolic Computation*, **45**(1), 2010 pp. 38–45. doi:10.1016/j.jsc.2009.06.003.

[8] F. Zhang, "Quaternions and Matrices of Quaternions," *Linear Algebra and Its Applications*, **251**, 1997 pp. 21–57. doi:10.1016/0024-3795(95)00543-9.

[9] B. Beck, "Sur les équations polynomiales dans les quaternions," *L' Enseignement Mathématique*, **25**, 1979 pp. 193–201.

[10] A. Pogorui and M. Shapiro, "On the Structure of the Set of Zeros of Quaternionic Polynomials," *Complex Variables. Theory and Application*, **49**(6), 2004 pp. 379–389. doi:10.1080/0278107042000220276.

[11] B. Gordon and T. S. Motzkin, "On the Zeros of Polynomials over Division Rings," *Transactions of the American Mathematical Society*, **116**, 1965 pp. 218–226. doi:10.1090/S0002-9947-1965-0195853-2.

[12] T.-Y. Lam, *A First Course in Noncommutative Rings*, New York: Springer-Verlag, 1991.

[13] I. Niven, "Equations in Quaternions," *The American Mathematical Monthly*, **48**(10), 1941 pp. 654–661. www.jstor.org/stable/2303304.

[14] R. Serôdio and L.-S. Siu, "Zeros of Quaternion Polynomials". *Applied Mathematics Letters*, **14**(2), 2001 pp. 237–239. doi:10.1016/S0893-9659(00)00142-7.

[15] R. Pereira, *Quaternionic Polynomials and Behavioral Systems*, Ph.D. thesis, Departamento de Matemática, Universidade de Aveiro, Portugal, 2006.

[16] G. Gentili and D. C. Struppa, "On the Multiplicity of Zeroes of Polynomials with Quaternionic Coefficients," *Milan Journal of Mathematics*, **76**(1), 2008 pp. 15–25. doi:10.1007/s00032-008-0093-0.

[17] M. I. Falcão, F. Miranda, R. Severino and M. J. Soares, "Quaternionic Polynomials with Multiple Zeros: A Numerical Point of View," in *11th International Conference on Mathematical Problems in Engineering, Aerospace and Sciences (ICNPAA 2016)*, La Rochelle, France, *AIP Conference Proceedings*, **1798**(1), 2017 p. 020099. doi:10.1063/1.4972691.

[18] H. R. Malonek, "Quaternions in Applied Sciences. A Historical Perspective of a Mathematical Concept," in *17th International Conference on the Applications of Computer Science and Mathematics in Architecture and Civil Engineering (IKM 2003)* (K. Gürlebeck and C. Könke, eds.), Weimar, Germany, 2003.

M. I. Falcão, F. Miranda, R. Severino and M. J. Soares, "Computational Aspects of Quaternionic Polynomials," *The Mathematica Journal*, 2018. dx.doi.org/doi:10.3888/tmj.20-4.

Additional Material

1. The package `QuaternionAnalysis`.

Available at: w3.math.uminho.pt/QuaternionAnalysis

2. The file `QPolynomial.m`.

Available at: www.mathematica-journal.com/data/uploads/2018/05/QPolynomial.m

About the Authors

M. Irene Falcão is an associate professor in the Department of Mathematics and Applications of the University of Minho. Her research interests are numerical analysis, hypercomplex analysis and scientific software.

Fernando Miranda is an assistant professor in the Department of Mathematics and Applications of the University of Minho. His research interests are differential equations, quaternions and related algebras and scientific software.

Ricardo Severino is an assistant professor in the Department of Mathematics and Applications of the University of Minho. His research interests are dynamical systems, quaternions and related algebras and scientific software.

M. Joana Soares is an associate professor in the Department of Mathematics and Applications of the University of Minho. Her research interests are numerical analysis, wavelets mainly in applications to economics, and quaternions and related algebras.

M. Irene Falcão
CMAT - Centre of Mathematics
DMA - Department of Mathematics and Applications
University of Minho
Campus de Gualtar, 4710-057 Braga
Portugal
mif@math.uminho.pt

Fernando Miranda
CMAT - Centre of Mathematics
DMA - Department of Mathematics and Applications
University of Minho
Campus de Gualtar, 4710-057 Braga
Portugal
fmiranda@math.uminho.pt

Ricardo Severino
DMA - Department of Mathematics and Applications
University of Minho
Campus de Gualtar, 4710-057 Braga
Portugal
ricardo@math.uminho.pt

M. Joana Soares
NIPE - Economics Politics Research Unit
DMA - Department of Mathematics and Applications
University of Minho
Campus de Gualtar, 4710-057 Braga
Portugal
jsoares@math.uminho.pt