Application Demonstration

Quaternion Algebras

Miriam Ciavarella
Università degli Studi di Torino
Italy
miriam.ciavarella@unito.it

Marina Marchisio
Università degli Studi di Torino
Italy
marina.marchisio@unito.it

Introduction

The aim of this worksheet is to define some procedures in order to make computations in a quaternion algebra over the field of rational number .
Quaternion algebra means something more general than the algebra of Hamilton's quaternions (for which there exists already a Maple package) .
Let we recall some definitions. A quaternion algebra over is a central simple algebra of dimension 4 over . To give a quaternion algebra is equivalent to give a pair of non-zero rational numbers so that is defined as the -algebra of basis where the elements verify the relations and .
The conjugation is the -endomorphism of which extends the non-trivial -automorphism of , defined by ConjQ. This is an involutive anti-automorphism of and we denote by ConjQ the quaternion conjugate of the element .
Let ; let we recall the definitions of two rational numbers associated to h: the reduced trace of is TrQConjQ and the reduced norm of is NormQConjQ. The invertible elements of are those elements with non-zero reduced norm.
Let *v* be a place of with completition (so it is either the *p*-adic numbers field for some prime *p* or the real number field **R**) . We say that *B* is unramified at *v* if (defined as the tensor product over) is isomorphic to the matrices over . We say that *B* is ramified at *v* if is the quaternion division algebra over . The set of places of where a quaternion algebra over ramifies is always finite and even, and their product is called the reduced discriminant of *B*. By the Classification Theorem, the places where *B* is ramified determine *B* up to isomorphism as an algebra; thus it can be very useful to give a procedure which computes the reduced discriminant of a given quaternion algebra *B=(a,b).*
We recall that if *B* is ramified at infinity then *B* is a definite quaternion algebra over ***Q***; in this case the reduced discriminant is the product of an odd number of prime numbers and if we write *B=(a,b)* then both *a* and *b* are negative numbers.
We define the following procedures for computing some operations over a quaternion algebra *B=(a, b)*:

1)The procedure **ProdQuat** returns the product of two elements of *B*;

2)The procedure **InvQ** returns the inverse of a non-zero-element of *B*;

3)The procedure **TrQ** returns the reduced trace of an element of *B*;

4)The procedure **NormQ** returns the reduced norm of an element of *B*;

5)The procedure **Discriminant** returns the reduced discriminant of *B*.

Product of two Quaternion Numbers

Initialization

```
> restart:
> with(LinearAlgebra):
```

Procedure Definition

Let be a quaternion algebra over ***Q***, where the elements *i*, *j* verify and . The new procedure **ProdQuat** in Maple computes the product of two quaternions in *B*. The algorithm is based on an isomorphism between *B* and **GL**(2,***Q***).
Input data are the pair *(a,b)* of non zero rational numbers and two elements .
The parameters *a* and *b* can be rational numbers or symbols. The output data is the quaternion product of *h* and *g* in *B*.

```
> ProdQuat:=proc(a,b,h,g)
  local terminenoto1,terminenoto2, m1,mi,mj,mk,hm1,hm2,
  prodquatm,prodquat,coef;
  terminenoto1:=h-coeff(h,i)*i-coeff(h,j)*j-coeff(h,k)*k:
  terminenoto2:=g-coeff(g,i)*i-coeff(g,j)*j-coeff(g,k)*k:
  m1:=Matrix([[1,0],[0,1]]):
  mi:=Matrix([[sqrt(a),0],[0,-sqrt(a)]]):
  mj:=Matrix([[0,1],[b,0]]):
  mk:=mi.mj:
  hm1:=MatrixScalarMultiply(m1,terminenoto1)+
  MatrixScalarMultiply(mi,coeff(h,i))+MatrixScalarMultiply(mj,
  coeff(h,j))+MatrixScalarMultiply(mk,coeff(h,k)):
  hm2:=MatrixScalarMultiply(m1,terminenoto2)+
  MatrixScalarMultiply(mi,coeff(g,i))+MatrixScalarMultiply(mj,
  coeff(g,j))+MatrixScalarMultiply(mk,coeff(g,k)):
  prodquatm:=MatrixMatrixMultiply(hm1,hm2):
  coef:=solve({x+y*sqrt(a)=prodquatm[1,1], z+w*sqrt(a)=
  prodquatm[1,2], z*b-b*w*sqrt(a)=prodquatm[2,1], x-y*sqrt(a)=
  prodquatm[2,2]}, {x,y,z,w});
```

```
assign(coef);
prodquat:=x+y*i+z*j+k*w;
end:
```

Examples

Example 1.

```
> h[1]:=i+k;
```

$$h_1 := i + k \tag{2.3.1}$$

```
> g[1]:=-1/12*i-1/12*k;
```

$$g_1 := -\frac{i}{12} - \frac{k}{12} \tag{2.3.2}$$

```
> ProdQuat(3,5,h[1],g[1]);
```

$$1 \tag{2.3.3}$$

Example 2.

```
> h[2]:=2*i-3*j+k;
```

$$h_2 := 2\,i - 3\,j + k \tag{2.3.4}$$

```
> g[2]:=i+k;
```

$$g_2 := i + k \tag{2.3.5}$$

```
> ProdQuat(3,1,h[2],g[2]);
```

$$3 + 3\,i + 3\,j + 3\,k \tag{2.3.6}$$

Inverse of a Quaternion Number

Initialization

```
> restart:
> with(LinearAlgebra):
> with(linalg):
```

Procedure Definition

We give a new procedure **InvQ** in Maple for computing the inverse of a non-zero element h of $B=(a,b)$.
Input data are are the pair (a,b) of non zero rational numbers which define B and a quaternion number h. The parameters a and b can be rational numbers or symbols.
The output data is an element **InvQ**(h) of B such that h***InvQ**$(h)=1$.

```
> InvQ:=proc(a,b,h)
local InvQ,terminenoto,m1,mi,mj,mk,hm,coef,mh;
terminenoto:=h-coeff(h,i)*i-coeff(h,j)*j-coeff(h,k)*k:
```

```
m1:=Matrix([[1,0],[0,1]]):
mi:=Matrix([[sqrt(a),0],[0,-sqrt(a)]]):
mj:=Matrix([[0,1],[b,0]]):
mk:=mi.mj:
hm:=MatrixScalarMultiply(m1,terminenoto)+
MatrixScalarMultiply(mi,coeff(h,i))+MatrixScalarMultiply(mj,
coeff(h,j))+MatrixScalarMultiply(mk,coeff(h,k)):
mh:=inverse(hm):
coef:=solve({x+y*sqrt(a)=mh[1,1], z+w*sqrt(a)=mh[1,2], z*b-
b*w*sqrt(a)=mh[2,1], x-y*sqrt(a)=mh[2,2]}, {x,y,z,w});
assign(coef);
InvQ:=x+y*i+z*j+k*w;
end:
```

Examples

Example 1.

```
> h[1]:=i+k;
```

$$h_1 := i + k \qquad (3.3.1)$$

```
> InvQ(3,5,h[1]);
```

$$-\frac{i}{12} - \frac{k}{12} \qquad (3.3.2)$$

Example 2.

```
> h[2]:=1-i+2*j+1/2*k;
```

$$h_2 := 1 - i + 2\,j + \frac{k}{2} \qquad (3.3.3)$$

```
> InvQ(2,3,h[2]);
```

$$-\frac{2}{23} - \frac{2\,i}{23} + \frac{4\,j}{23} + \frac{k}{23} \qquad (3.3.4)$$

Reduced Trace of a Quaternion

Initialization

```
> restart:
> with(LineaeAlgebra):
```

Procedure Definition

The new procedure **TrQ** in Maple computes the reduced trace of a quaternion number.
Input data is quaternion number *h*. The output data is the reduced trace of *h*, which is a rational number.

```
> TrQ:=proc(h) local TrQ,terminenoto;
  terminenoto:=h-coeff(h,i)*i-coeff(h,j)*j-coeff(h,k)*k:
  TrQ:=2*terminenoto:
  end:
```

Examples

Example 1.

```
> h[1]:=2+4*i;
```

$$h_1 := 2 + 4\,i \qquad (4.3.1)$$

```
> TrQ(h[1]);
```

$$4 \qquad (4.3.2)$$

Example 2.

```
> h[2]:=1/2+4*i-11*j+k;
```

$$h_2 := \frac{1}{2} + 4\,i - 11\,j + k \qquad (4.3.3)$$

```
> TrQ(h[2]);
```

$$1 \qquad (4.3.4)$$

Reduced Norme of a Quaternion

Initialization

```
> restart:
> with(LinearAlgebra):
```

Procedure Definition

We give a new procedure **NormQ** in Maple for computing the reduced norm of a quaternion number.
Input data are are the pair *(a,b)* of non zero rational numbers which define *B* and a quaternion number *h*. The parameters *a* and *b* can be rational numbers or symbols.
The output data is the reduced norm **NormQ***(h)* of *h*, which is a rational number.

```
> NormQ:=proc(a,b,h) local NormQ,terminenoto;
                         terminenoto:=h-coeff(h,i)*i-coeff
  (h,j)*j-coeff(h,k)*k: NormQ:=terminenoto^2-a*coeff(h,i)^2-b*
  coeff(h,j)^2+a*b*coeff(h,k)^2: end:
```

Examples

Example 1.

```
> h[1]:=i;
```

$$h_1 := i \quad (5.3.1)$$

```
> NormQ(3,5,h[1]);
```

$$-3 \quad (5.3.2)$$

Example 2.

```
> h[2]:=-2+3*i-1/3*k;
```

$$h_2 := -2 + 3\,i - \frac{k}{3} \quad (5.3.3)$$

```
> NormQ(3,4,h[2]);
```

$$-\frac{65}{3} \quad (5.3.4)$$

Discriminant of a Quaternion Algebra B

Initialization

```
> restart:
> with(padic):
```

Procedure Definition

We give a new procedure (**Discriminant**) in Maple for computing the discriminant of a quaternion algebra *B=(a,b).*
Input data are the rational numbers *a*, *b* defining the quaternion algebra where and . The output data is the product of the finite places of ***Q*** which ramify in *B*.

```
> Discriminant:=proc(a,b)
  local a2,b2,u2,v2,eu,ev,wu,wv,alpha, beta, u, v, d ,e, p,Lu,
  Lv,Discriminant;
  a2:=ordp(a,2):
  b2:=ordp(b,2):
  u2:=a/(2^a2):
  v2:=b/(2^b2):
  eu:=modp((u2-1)/2,2):
  ev:=modp((v2-1)/2,2):
  wu:=modp((u2^2-1)/8,2):
  wv:=modp((v2^2-1)/8,2):
  if (-1)^(eu*ev+a2*wv+b2*wu)=-1 then
            d:=2
  else
            d:=1
  end if:
```

```
for p from 3 to max(abs(a),abs(b)) do
          if type(p,prime) and gcd(a*b,p)<>1 then
                    alpha:=ordp(a,p):
                    beta:=ordp(b,p):
                    u:=a/(p^alpha):
                    v:=b/(p^beta):
                    e:=modp((p-1)/2,2):
                    Lu:=modp(u^((p-1)/2),p):
                    Lv:=modp(v^((p-1)/2),p) :
                    if Lv=p-1 then
                              Lv:=-1
                    end if:
                    if Lu=p-1 then
                              Lu:=-1
                    end if:
                    if (-1)^(alpha*beta *e)*(Lu)^beta*(Lv)
^alpha=-1 then
                              d:=d*p
                    end if:
          end if:
end do:
Discriminant:=d;
end:
```

Examples

Example 1.

```
> Discriminant(-2,-5);
```

$$5 \tag{6.3.1}$$

Example 2.

```
> Discriminant(3,5);
```

$$15 \tag{6.3.2}$$

References

Roger C. Alperin, *Free Subgroups of Quaternion Algebras*, Proceedings of the American Mathematical Society, Volume 118, Number 1, May 1993.

Marie-France Vigneras, *Arithmétique des Algèbres de Quaternions*, Lecture Notes in Mathematics, 800, Springer-Verlag, 1980.