GAwxM and about how it works

User defined infix operators

User defined infix operators allow geometric algebraic expressions to be developed, simplified and evaluated. During initialization, the functions called by the operators are allocated to the binary infix operators. The binary blade product operators are...

"~*" , the geometric blade product
"~^" , the blade outer product
"~." , the blade left inner product

The word “blade” is used loosely and refers to any member of the canonical basis below.

The list of bases is generated from the pseudoscalar, e1~e2~e3. Although the symbol 'tilda' is defined as an infix operator, it does not have an associated binary infix function; it is used to form the k-vectors of the canonical basis of the algebra for k>1.

e.g. 1

e1, e2, e3,

e1~e2, e1~e3, e2~e3,

e1~e2~e3

The operator 'tilda' itself has the highest precedence; it is a left associative binary operator and serves only to bind together the ascending basis vectors into higher grade members. Thus, expressions such as e1~e1 and e3~e2 should not occur, but expressions like e1~*e1 and e3~^e2 are valid and will be evaluated when used.

All of the blade product operators above have exactly the same precedence (less than 'tilda'). The precedence of the intrinsic maxima infix parentheses cannot be redefined by the user. This means that the parentheses are retained when maxima tries to resolve the meaning of ‘tilda’. So e1~e2~e3 becomes (e1~e2)~e3. This may seem awkward at first but the feature actually helps to locate the bases when long expressions occur in the output.

Other dimensions are studied by editing the pseudoscalar within the Initialization section of code. In order to work in the four-dimensional space, G4 the pseudoscalar, e1~e2~e3~e4 would be used.

There is a similar syntax for the multivector product operators...

"&*" , the geometric multivector product
"&^" , the multivector outer product
"&." , the multivector left inner product

These operators allow geometric algebraic expressions resembling vector equations to be coded; their precedences are also equal to one another and all are lower than that of the blade operators. The functions called by the operators can process rational parametric coefficients in a multivector (or in any vector).

Some fundamental active documents (software)

The tutorial document named ...GAwxM/wxM_tutorial.wxm is a "how to get started" introduction to wxMaxima. Within wxMaxima, a “software program” i.e. a sequence of Maxima commands is held in an “active document” and text boxes may be used to annotate and document the code. So, where we read “document” we can understand “documented program”.

There is a document for developing algorithms named ...GAwxM/GA_syntax/left_inner_template.wxm and it is useful to examine this document to understand the initialization. In particular, note that the batchload of GAwxM specific function files from the folder ...GAwxM/GA_functions/ relies upon the file named ...maxima-init.mac.

The name ...left_inner... has been chosen to show clearly from the start that the GAwxM functions are relevant only to the formulation of Geometric Algebra using the left (contraction) inner product rather than any other inner product. Also, the initial development of GAwxM has, of necessity, concentrated upon the definition and validation of the GAwxM syntax.

The document named ...GAwxM/GA_syntax/blade_operator_syntax.wxm gives many examples of the level precedence aspect of the blade operators. The document named ...GAwxM/GA_syntax/geometric_product_syntax.wxm gives some examples of the level precedence aspect of the multivector product operators. Folder...GAwxM/Working_code/ contains documents used to develop GA product and parsing functions. Folder...GAwxM/Working_code_GC/ contains documents used to develop calculus functions.

The files in folders …GC_fundamental/ and …GC_solid_fundamental/ attempt to confirm the fundamental theorem in some specific cases. The former equates integration over a surface with integration over the boundary curves. The latter equates integration over the volume of a *solid* with integration over its boundary surfaces. They are quite heavy going but equality was found. The folder …Projective_geometry/ considers a two dimensional theorem in G4. It clarifies just how to handle the extra dimension and also shows some lengthy Maxima calculations. The documents in folder…GC_space_time/ are probably the most significant. The Maxima intrinsic imaginary, %i, has been used in order to imitate the G(1,3) space-time algebra.

The mere fact that %i is not a part of GA and yet we can perform a space/time split begins to suggest that the split is actually artificial (i.e. subjective) and may be a feature of the measurement process (as in QM). I actually prefer this approach to that of imposing the split at a lower level by assuming a geometry of G(1,3). Earlier during the development I thought that GAwxM would have a severe limitation in not being able to model indefinite metrics but there now seems to be a way forward.

Initialization of GAwxM documents

The initialization sequence is the same for every document. This initialization sequence, shown below, occurs at the beginning of every active document in GAwxM and is consequently almost set in stone. So the document named ...GAwxM/GA_syntax/left_inner_template.wxm...may be used as a template to generate any new document. It performs the following actions…

- Set intrinsic Maxima logicals
- Load intrinsic (Maxima or lisp) function files
- Batchload GA specific (Maxima) function files
- Batchload GC specific function files
- Pseudoscalar definition (specifies the space dimension)
- Calculation of the inverse pseudoscalar used to generate the dual of a multivector
- Formation of the standard basis for the specified dimension
- Fill global arrays with that basis...
- lstblds[] is an array of lists of blades and allblds[] is a list of all blades

Validation sequence of GAwxM syntax

The identity test sequence in the document named ...GawxM/GA_syntax/multivector_function_syntax.wxm...

- Full dimension multivector generation
- Associativity test for the triple geometric product
- Associativity test for the triple outer product
- Test the code for the commutator product
- Test for the Jakobi expansion of the triple commutator product
- Consistency checks with the left inner product...; using two particular pure grade multivectors (generation of multivectors containing particular grades only) ; using the dual of the dual of a pure grade multivector ;
- Check that the reverse of the geometric product equals the juxtaposed product of the reverses
- Check the scalar product function is symmetric and positive definite
- Test the norm (modulus) function

The fundamental identity test and duality test sequences in the document named ...GawxM/GA_syntax/left_inner_product_duality.wxm...

- Testing the left inner product in the fundamental identity
- Checking the first duality identity for multivectors, (A.B)* = A^B*
- Testing the other duality identity for multivectors; (A^B)* = A.B*