**Steve Sangwine**

## *MATLAB toolbox for Clifford Algebras*

(work done in cooperation with E. Hitzer)

MATLAB is a powerful numerical computing environment oriented towards manipulation of matrices and vectors (in the linear algebra sense, that is arrays of numbers).

The author (with Nicolas le Bihan) developed a quaternion toolbox for MATLAB in 2005 (QTFM), subsequently extended, and downloaded from Sourceforge over 10,000 times. This toolbox includes Fourier transforms, matrix decompositions including LU, QR, SVD, and the eigenvalue decomposition, as well as operations for signal and image processing (one of the original aims of the toolbox).

In 2013 development started on a Clifford toolbox along similar lines. The new toolbox can be initialised to any Clifford algebra with signature $(p, q, r)$ where $p, q, r$ are the numbers of basis elements that square to $+1, -1, 0$ respectively. A very powerful feature of the toolbox is the ability to iterate over Clifford algebras by re-initialising the toolbox signature inside a loop. This does not destroy existing variables, but checks are performed to ensure that variables are not used unless their signature matches the current signature. It is thus possible to explore whether particular algorithms will work in all algebras, or only in a subset. As with the QTFM toolbox, it is designed to work just like Matlab, so that the powerful colon notation and square bracket concatenation works with multivector arrays. Overloading of MATLAB function names means that operations on multivectors are carried out with the same function names as operations on real or complex arrays.

The fundamental structure of the toolbox is now complete, and some steps have started on implemented high-level algorithms, beginning with the LU decomposition and the matrix inverse. The toolbox inherently treats matrices of multivectors, and the components of the multivectors may be complex (this follows from the fact that ordinary numbers in MATLAB are by default complex). Since the underlying numeric operations are implemented by MATLAB, calculations are vectorised (that is implemented using processor-level parallelism), giving fast performance. Care has been taken to represent multivectors internally so that zero components are stored as empty arrays $-$ avoiding needless multiplications by zero and wasted storage (important if large matrices are manipulated in a high-dimensional algebra).

The talk will include live demonstrations of the toolbox capabilities.