***GAPoTNumLib*** MATLAB Toolbox

Francisco G. Montoya

University of Almeria (Spain)

Ahmad H. Eid

Port-Said University (Egypt)

[Introduction 2](#_Toc59615804)

[Download and install 2](#_Toc59615805)

[GAPoTNumLib Classes 2](#_Toc59615806)

[Library initialization 3](#_Toc59615807)

[Type definitions and conversion 3](#_Toc59615808)

[Basic operations 7](#_Toc59615809)

[Human readable format 8](#_Toc59615810)

[Power related functions 10](#_Toc59615811)

[Others 13](#_Toc59615812)

[Examples 14](#_Toc59615813)

# Introduction

Geometric Algebra has gained a lot of attention recently. The use of this framework in electrical and power systems has been introduced by the [GAPoT](https://www.researchgate.net/project/GAPoT-power-theory-based-on-geometric-algebra) theory. However, a powerful and flexible tool is needed in order to manage the large number of dimensions potentially required to deal with systems where a high number of harmonics are present. [GAPoTNumLib](https://github.com/ga-explorer/GAPoTNumLib/tree/master/GAPoT%20MATLAB%20Toolbox) is such a tool. Thanks to its design and architecture it is able to provide the necessary capabilities to perform electrical circuit analysis as well as to evaluate power and energy flow computations. Typically, geometric algebra software libraries are designed for manipulating general multivectors. This constraint is not necessary for GAPoT computations, which only need to deal with grade 0,1,2 sparse multivectors. This allows for efficient storage and processing of multivector coefficients.

This document presents a detailed description of the functions that the GAPoTNumLib library implements in MATLAB, as well as a short guide to its configuration and installation.

The source code is available at [Github](https://github.com/ga-explorer/GAPoTNumLib). The creator is Dr. Ahmad Hosny Eid, Software Engineer and Assistant Professor of Computer Engineering at the Faculty of Engineering, Port-Said University, Egypt. Visit [https://ga-explorer.netlify.app](https://ga-explorer.netlify.app/) for more information.

# Download and install

Note: Right now, this tool is only available for Windows.

1. Download the latest version of the C# (.NET) code *GAPoTNumLib* from GitHub:

https://github.com/ga- explorer/GAPoTNumLib

2. To perform the compilation process (using Visual Studio, for example), open and Build the *GAPoTNumLib* solution using “x64 Debug” configuration. Make sure the *GAPoTNumLib.Framework*project is the default project of the solution.

3. In the MATLAB Toolbox, open the file gapotInit.m and edit the variable gapotAssemblyPath to be the path containing the *GAPoTNumLib.Framework.exe* file.

4. In MATLAB add the toolbox folder to the MATLAB path.

# GAPoTNumLib Classes

In the .NET solution, the user can find several classes to represent GAPoT multivectors under the GAPoTNumLib.GAPoT namespace as follows:

* **GaPoTNumVector** is the class used to represent GAPoT vectors, typically holding currents and voltages. The main operations this class provides include setting, getting, and adding terms, polar phasors, and rectangular phasors. The user can also add and subtract two GAPoT vectors, compute their geometric product, find the negative, norm, squared norm, reverse, and inverse of a GAPoT vector. In addition, several methods for displaying the GAPoT vector in various text and LaTeX formats exist.
* **GaPoTNumBiversor** is used to represent GAPoT biversors; sparse multivectors which only contains elements of grades 0 and 2 typically representing power and impedance. The class provides methods for setting, getting, and adding individual terms of grade 0 and 2. The user can also compute the negative, norm, squared norm, reverse, and inverse of biversors. There are several methods for extracting power quantities from the biversor such as active, non-active, reactive, fundamental reactive, and harmful power parts. In addition, several methods for displaying the GAPoT biversor in various text and LaTeX formats exist.

For the elements of a GAPoT vector, there are three classes that can be used to hold information on terms and phasors:

• GaPoTNumVectorTerm holds a single term vector.

• GaPoTNumPolarPhasor holds a single polar phasor vector.

• GaPoTNumRectPhasor holds a single rectangular phasor vector

# Library initialization

The library needs to be initialized once MATLAB is ready, so gapotInit command should be first executed before operations are performed.

>> gapotInit

# Type definitions and conversion

To operate with vectors and multivectors, it is necessary to define them in a particular way. For example, the definition of vectors can be carried out in several distinct ways (rectangular or polar), so a text parser is needed for an easy and flexible data entry. Once defined, they are stored as objects that can be displayed as text strings or arrays using specific functions.

* gapotBiversorToTermsArray.m

%Create a sparse MATLAB array from a GAPoT multivector terms. It is used to convert a multivector object into a readable sparse array. For example, a power multivector M with 6 terms:

>> gapotBiversorToTermsArray(M,6)

ans =

(1,1) 35951.5264416066

(1,2) 12441.9102309493

(1,3) -7248.47355839335

(1,4) -12499.6213980426

(1,5) -7248.47355839341

(1,6) 37383.4418599411

* gapotParseBiversor.m

%Parse a GAPoT Biversor expression. Creates a multivector object using term components. For example, for the multivector

>> M = gapotParseBiversor("2 <>, -5 <1,2>, 5<3,4>")

M =

GaPoTNumBiversor with no properties

* gapotParseVector.m

%Parse a GAPoT vector expression. Creates a vector object. Different strategies are possible such as polar or rectangular. For example, for a vector

Rectangular

>> v = gapotParseVector("r(0.433, -0.25) <1,2>, r(0.433, 0.25) <3,4>")

Polar (modulus and angle in degrees), with

>> v = gapotParseVector("p(0.5,0.523598775598299) <1,2>, p(0.5,-0.523598775598299) <3,4>")

Rectangular (term by term)

>> v = gapotParseVector("0.433 <1>, -0.25 <2>, 0.433 <3>, 0.25 <4>")

**Note**: Remember that GAPoT uses a slightly different criteria for phase angles in Argand diagram. Positive angle in polar form implies a rotation in the CW direction, and negative angles a rotation in CCW direction.

* gapotPartsTermsToArray.m

%Create a sparse MATLAB column array from a GAPoT vector's parts. Each vector part's terms are put in a separate colum in the array. % rowsCount indicate the max. number of allowed terms per vector part

>> mvU = gapotParseVector("2<1>, 3<2>,-2<3>, -1<4>, 1<5>");

>> gapotGetPartsTermsToArray(mv1,2,[2,2,2]);

ans =

(1,1) 2

(2,1) 3

(1,2) -2

(2,2) -1

(1,3) 1

* gapotPolarPhasorsArrayToVector.m

%Creates a GAPoT vector using MATLAB array of polar phasors. This function allows the creation of a vector using MATLAB variables and operations. One row per harmonic (modulus and phase angle).

>> u1 = 5;

>> u2 = 10;

>> mv2 = gapotPolarPhasorsArrayToVector([

100 \* sqrt(2) / u1, 30 \* pi / 180;

200 \* sqrt(3) / u2, 60 \* pi / 180;

]);

>> gapotDisplayPhasors(mv2)

ans =

p(28.2842712474619, 0.523598775598299) <1,2>, p(34.6410161513775, 1.0471975511966) <3,4>

* gapotTermsArrayToVector.m

%Creates a GAPoT vector using MATLAB array of rectangular terms. This function allows the creation of a vector using MATLAB variables and operations. One row per harmonic (2 rectangular terms).

>> mv1 = gapotTermsArrayToVector([

100 \* sqrt(2) / u1;

0;

200 \* sqrt(3) / u2 \* cos(60 \* pi / 180);

-200 \* sqrt(3) / u2 \* sin(60 \* pi / 180)

]);

>> gapotDisplayTerms(mv1)

ans =

28.2842712474619 <1>, 17.3205080756888 <3>, -30 <4>

>> gapotDisplayPhasors(mv1)

ans =

p(28.2842712474619, 0) <1,2>, p(34.6410161513775, 1.0471975511966) <3,4>

* gapotVectorToPolarPhasorsArray.m

%Create a sparse MATLAB array from a GAPoT vector polar phasors. It is used to convert a vector object into a readable sparse array of modulus and phase angle for each harmonic. Using the above vector mv1, it reads as

>> gapotVectorToPolarPhasorsArray(mv1,3)

ans =

(1,1) 28.2843

(2,1) 34.6410

(2,2) 1.0472

* gapotVectorToTermsArray.m

%Create a sparse MATLAB array from a GAPoT vector rectangular terms. It is used to convert a vector object into a readable sparse array with rectangular coordinates for each harmonic. Using the above vector mv1, it reads as

>> gapotVectorToTermsArray(mv1,4)

ans =

(1,1) 28.2843

(3,1) 17.3205

(4,1) -30.0000

# Basic operations

A few simple operations have been also implemented to facilitate basic computations.

* gapotAdd.m

%Add two GAPoT vectors.

* gapotGp.m

%The geometric product of two GAPoT multivectors. When one is a vector and another is a multivector (scalar + bivectors) the result is the vector part of the actual geometric product (the scalar and trivector parts are omitted). Because of GAPoT formulation, the product of a vector and a multivector can only yield a vector, so this operation is optimized for this condition.

>> R=gapotGp(mv1,mv2)

R =

GaPoTNumBiversor with no properties.

>> R.TermsToText

ans =

1892.82032302755 <>, -400 <1,2>, 65.6338798447071 <1,3>, -113.681214588903 <1,4>,

244.948974278318 <2,3>, -424.264068711928 <2,4>

* gapotInverse.m

%Compute the inverse of a GAPoT vector or Biversor

>> inv=gapotInverse(mv1)

inv =

GaPoTNumVector with properties:

Count: 3

>> inv.TermsToText

ans =

0.014142135623731 <1>, 0.00866025403784439 <3>, -0.015 <4>

* gapotNegative.m

%Compute the negative of a GAPoT vector or Biversor

* gapotNorm.m

%Compute the norm of a GAPoT vector or Biversor

>> gapotNorm(mv1)

ans =

44.7214

* gapotNorm2.m

%Compute the squared norm of a GAPoT vector or Biversor

>> gapotNorm2(mv1)

ans =

2000

* gapotPower.m

%See gapotGp

* gapotReverse.m

%Compute the reverse of a GAPoT vector or Biversor

* gapotSubtract.m

%Subtract two GAPoT vectors

# Human readable format

All vectors and multivectors are stored as objetcs in MATLAB. Some functions needs to be used to display the content in a human readable format.

* gapotDisplayPhasors.m

%Display the polar phasors of a GAPoT vector

>> gapotDisplayPhasors(mv1)

ans =

p(28.2842712474619, 0) <1,2>, p(34.6410161513775, 1.0471975511966) <3,4>

* gapotDisplayTerms.m

%Display the terms of a GAPoT vector or Biversor

>> gapotDisplayTerms(mv1)

ans =

28.2842712474619 <1>, 17.3205080756888 <3>, -30 <4>

* gapotGetTermPart.m

%Get the term part of a GAPoT power multivector. Mainly used to extract a specific Biversor term in the geometric apparent power

>> part=gapotGetTermPart(M,3,4)

part =

GaPoTNumBiversor with no properties.

>> part.TermsToText

ans =

5 <3,4>

* gapotGetTermValue.m

%Get the coefficient of a specific term of a GAPoT power multivector.

>> part=gapotGetTermValue(M,3,4)

part =

5

# Power related functions

* gapotGetActivePart.m

%Get the scalar part of a GAPoT power multivector,i.e., the active Power.

>> P=gapotGetActivePart(M)

P =

GaPoTNumBiversor with no properties.

>> P.TermsToText

ans =

2 <>

* gapotGetActiveTotal.m

%Get the coefficient of the scalar part of a GAPoT power multivector. Actually, is nothing more than P. See gapotGetActivePart.m

* gapotGetHarmPart.m

%Get the non-scalar part of a GAPoT power multivector except for <1,2>. Basically, it gets Biversor parts related with harmonic distortion and unbalance power.

>> H=gapotGetHarmPart(M)

H =

GaPoTNumBiversor with no properties.

>> H.TermsToText

ans =

5 <3,4>

* gapotGetHarmTotal.m

%Get the non-scalar part values of a GAPoT power multivector except for <1,2>. Sum the coefficients of the extracted terms.

>> M = gapotParseBiversor("2 <>, -5 <1,2>, 5<3,4>, 7<7,8>")

M =

GaPoTNumBiversor with no properties.

>> H=gapotGetHarmTotal(M)

H = 12

* gapotGetNonActivePart.m

%Get the non-scalar part of a GAPoT power multivector.

>> Non=gapotGetNonActivePart(M)

Non =

GaPoTNumBiversor with no properties.

>> Non.TermsToText

ans =

-5 <1,2>, 5 <3,4>, 7 <7,8>

* gapotGetNonActiveTotal.m

%Get the non-scalar coefficients of a GAPoT power Biversor and sum them.

>> gapotGetNonActiveTotal(M)

ans =

7

* gapotGetReactiveFundamentalPart.m

%Get the <1,2> non-scalar part of a GAPoT power Biversor. According to GAPoT theory, the term <1,2> holds for the fundamental harmonic reactive power.

>> Q1=gapotGetReactiveFundamentalPart(M)

Q1 =

GaPoTNumBiversor with no properties.

>> Q1.TermsToText

ans =

-5 <1,2>

* gapotGetReactiveFundamentalTotal.m

%Get the coefficient of the <1,2> part of a GAPoT power multivector. Actually, is nothing more than Q1. See gapotGetReactiveFundamentalPart.m

* gapotGetReactivePart.m

%Get the reactive part of a GAPoT power Biversor. It computes the Biversor terms in the form <2k-1,2k> where k is the harmonic order.

>> Q=gapotGetReactivePart(M)

Q =

GaPoTNumBiversor with no properties.

>> Q.TermsToText

ans =

-5 <1,2>, 5 <3,4>, 7 <7,8>

* gapotGetReactiveTotal.m

%Sum of the coefficients for terms <2k-1,2k>. It is the reactive power in the Budeanu sense.

>> gapotGetReactiveTotal(M)

ans =

7

# Others

* gapotGetParts.m

%Extract rectangular parts from vectors. Using the vector mv1 defined above, we can split into 2 vectors of lengh 2 each by using

>> mvuparts=gapotGetParts(mv1,[2,2]);

>> mvuparts(1).TermsToText

ans =

28.2842712474619 <1>

>> mvuparts(2).TermsToText

ans =

17.3205080756888 <3>, -30 <4>

* gapotSparseMatrixDataToArray.m

%Function to convert data arrays into a sparse MATLAB array.

# Examples

* gapotSample1.m

%see code at https://github.com/ga-explorer/GAPoTNumLib/blob/master/GAPoT%20MATLAB%20Toolbox/gapotSample1.m

* gapotSample2.m

%see code at https://github.com/ga-explorer/GAPoTNumLib/blob/master/GAPoT%20MATLAB%20Toolbox/gapotSample2.m

* gapotSample3.m

%see code at https://github.com/ga-explorer/GAPoTNumLib/blob/master/GAPoT%20MATLAB%20Toolbox/gapotSample3.m

* gapotSample4.m

%see code at https://github.com/ga-explorer/GAPoTNumLib/blob/master/GAPoT%20MATLAB%20Toolbox/gapotSample4.m

* gapotSample5.m

%see code at https://github.com/ga-explorer/GAPoTNumLib/blob/master/GAPoT%20MATLAB%20Toolbox/gapotSample5.m