Advanced Methods of Phasor Calculation and Visualization in AC Circuits Using OOP in MATLAB

Dr. Jai P. Agrawal, Purdue University Northwest

Jai P. Agrawal is a professor in electrical and computer engineering technology at Purdue University, Calumet. He received his Ph.D. in electrical engineering from University of Illinois, Chicago, in 1991, dissertation in power electronics. He also received M.S. and B.S. degrees in electrical engineering from Indian Institute of Technology, Kanpur, India, in 1970 and 1968, respectively. His expertise includes analog and digital electronics design, power electronics, nanophotonics, and optical/wireless networking systems. He has designed several models of high frequency oscilloscopes and other electronic test and measuring instruments as an entrepreneur. He has delivered invited short courses in Penang, Malaysia and Singapore. He is also the author of a textbook in power electronics, published by Prentice-Hall, Inc. His professional career is equally divided in academia and industry. He has authored several research papers in IEEE journals and conferences. His current research is focused on renewable energy technology, smart energy grid.

Prof. Omer Farook, Purdue University Northwest

Omer Farook is a member of the faculty of Electrical and Computer Engineering Technology at Purdue University, Nothwest. Farook received the diploma of licentiate in mechanical engineering and B.S.M.E. in 1970 and 1972, respectively. He further received B.S.E.E. and M.S.E.E. in 1978 and 1983, respectively, from Illinois Institute of Technology. Farook's current interests are in the areas of embedded system design, hardware-software interfacing, digital communication, networking, image processing, and biometrics, C++, Python, PHP and Java languages. He has a keen interest in pedagogy and instruction delivery methods related to distance learning. He has a deep commitment to social justice and in achieving economic and educational equity.

ADVANCED METHODS OF PHASOR CALCULATION AND VISUALIZATION IN AC CIRCUITS USING OOP IN MATLAB

Abstract

This paper presents advanced methods of calculation and visualization in AC circuit analysis with the objective of replacing the computational drudgery by the focus on conceptual learning. These methods are designed using Object Oriented Programming (OOP) on the MATLAB platform. OOP methodology is used to specify phasors, perform mathematical and graphical algebraic operations, and visualize in phasor and time domain. Phasors are used as objects under a class definition. Using several methods and a set of user defined functions, the paper presents examples for solving a large gamut of problems like doing mesh and nodal analyses, Thevenin equivalents and power calculations and visualization of power in single and three phase circuits. The examples illustrate the simplicity and power of these methods. It is hoped that both the educators and learners will find them very useful in ac sinusoidal circuit analysis in the engineering education.

1. INTRODUCTION

Conventional method of ac circuit analysis use phasors and phasor based methods. Some educators use MATLAB as the calculation engine[5]…[14]. However, the MATLAB, even though is primarily designed to work with complex numbers and methods but it does not simplify the calculationrs involving phasors significantly. Furthermore, the phasor analysis is essentially a graphical method. Conventional MATLAB does not help in the visualization of graphical analysis. This paper presents a unique Phasor Tool Box which facilitates most of the phasor calculations and enable visualization in the phasor diagrams. The tool box is designed in MATLAB but requires students to have minimal scripting background, keeping in mind that these tools will be used by the beginner students in Electrical engineering/Technology programs. This tool box is designed by using object oriented MATLAB programming methods but are transparent to users. Using these tools is as easy as using conventional algebra for adding, subtraction, multiplication, division of phasors, and additionally visualize these operations in the complex plane. It is hoped that introduction of the phasor methods would help in fortifying the conceptual learning.

2. PHASOR TOOL BOX

We propose the use of a phasor tool box and associated functions on the MATLAB platform for ac circuit analysis. The phasor tool box uses a class variable $\boldsymbol{phasor}(magnitude, phase)$, to model voltage, current, impedance, admittance and the apparent power vectors. Following list gives the Phasor Tool Box codes for different vector types:

Variable	**Vector type**	**Math expression**	**Phasor Tool Box code**
Voltage V	phasor	$V = V_{rms}\ \angle\theta$	phasor(Vrms, th)
Current I	phasor	$I = I_{rms}\ \angle\theta$	phasor(Irms, th)
Impedance Z	polar	$Z = Z_m\ \angle\theta$	phasor(Zm, th)
Admittance Y	polar	$Y = Y_m\ \angle\theta$	phasor(Ym, th)
Power S	polar	$S = S_m\ \angle\theta$	phasor(Sm, th)

All above vectors are modeled as **Objects** under a **phasor class.** These objects interact like ordinary mathematical variables. Phasor objects can be added, multiplied and divided using same operators "+, -, *, / and left divide \ matrix solution" as are used in traditional mathematical operations involving constants and variables.

2.1 Algebraic Operations and Visualization

Following example shows three phasors V1, V2 and V3 are added, and visualized and added graphically in the complex plane, The two phasors V1 and V2 are specified by magnitudes and phases while the third phasor V3 is specified as a complex vector.

Three phasors: $V1=10\angle 30$ $V2=5\angle 60$ $V3=10-j16$
Phasor addition $Vadd=V1+V2+V3$

MATLAB Code

```
V1=phasor(10, 30);   V2=phasor(5, 60);        %defining two phasor objects
V3=phasor(10, -16, 'x2ph') ;                  %defining two phasor objects
Vadd=V1+V2+V3;                                %Algebraic addition of phasor objects
phplot([V1, V2, V3, Vadd])                    %Fig. 1(a) plotting phasors on the complex plane
Vadd=add_graph(V1, V2, V3)                    %Fig. 1(b) Graphical Addition of Phasors
Vadd =
  phasor with properties:
      Mag: 18.3389
    phase: -6.2094
```

```
%Representing phasors V1, V2 and Vadd can be visualized in the triangular graphical format
>>triplot([V1, V2, Vadd])                %Fig. 1 (c) visualization in the triangle format
>>f=60; T=1/f;                           %Fig. 1 (d) Time domain plot of phasors, f is the frequency
>>phplot_signal( [V1, V2, Vadd], f, 0, T );
```

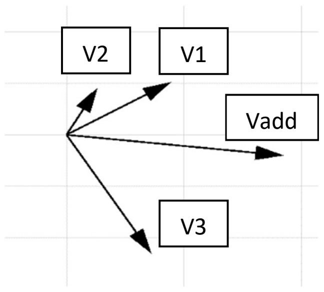

Fig. 1 (a)

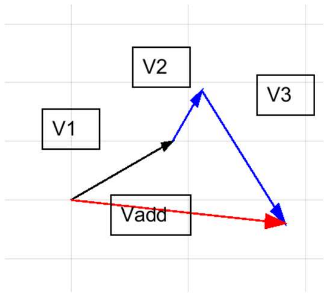

Fig. 1 (b)

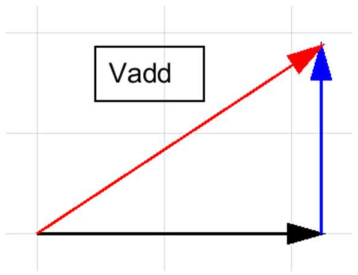

Fig. 1 (c)

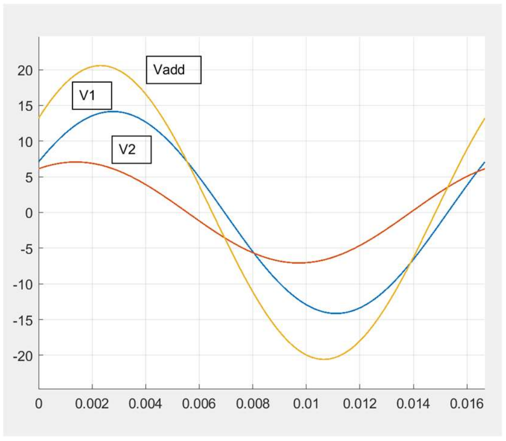

Fig. 1 (d)

2.2 Capabilities of the Phasor Tool Box

The proposed phasor toolbox enables users to perform direct algebraic and graphical manipulations on phasors, conversion from non-phasor quantities to phasor quantities, use of a mix of phasor and complex quantities in ac circuit analysis, visualize phasors on the complex plane, calculate input and out impedances and use of important theorems like the maximum power transfer and Thevenin theorems. The capabilities are listed below:

Conversion functions:	complex ↔ polar ↔ phasor
Math operations:	
Unary operations:	Amplification, rotation, negation, inversion and conjugation
Binary operations:	Addition, subtraction, multiplication, division.
Phasor diagrams:	Graphical addition of phasors, phasor triangle
Time domain diagrams of phasors	
In/Output impedance in network	
Thevenin and Norton equivalent	
Power calculations:	single and three phase circuits

Parentheses () can be used in phasor math like a regular math operation for prioritization of operations on a MATLAB line code. Phasors can also be organized as arrays and matrices using [ ] brackets.

2.3 Example of Evaluation of a Phasor Expression:

$$V_n = \frac{120\angle 60\ \ (90\angle 30\ - 50\angle{-4}\ \ )}{90\angle 30 + 50\angle 45 + 100\angle 0}$$

```
%PTB2_Ex_11.m
%A complex phasor math expression
V1=phasor(120, 60);
V2=phasor(90, 30);
V3=phasor(50, -45);
V4=phasor(100, 0);
Vn=V1 *(V2-V3) / (V2 + conj(V3) + V4)
phplot([V1, V2, V3, V4, Vn])

Vn =
  phasor with properties:

      Mag: 47.8791
    phase: 101.4343
```

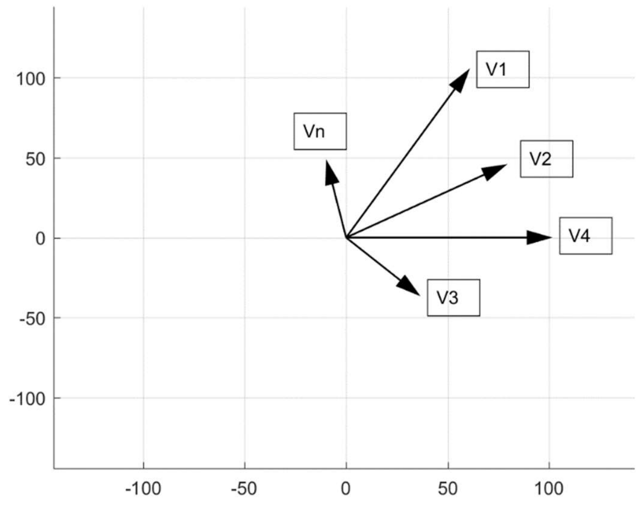

Fig. 2

2.4 AC Circuit Analysis

Following example shows the calculation of phasor current and the apparent, reactive and real power, and leading or lagging power factor:

```
E=phasor(100, 0);
ZR=phasor(6, 0);
ZL=phasor(7, 90);
ZC=phasor(15, -90);
ZT=ZR+ZL+ZC;        %total impedance phasor
ZT=add_graph(ZR, ZL, ZC, ZT)
I=E/ZT;%calculate the current phasor using ohm's law
phplot([ I, ZT])
% calculations of apparent power S,
%reactive power Q, real power
% P, and power factor
[S, P, Q, Fp, ph]=PWR(E, I)
S =
  phasor with properties:
      Mag: 1.0000e+03
    phase: -53.1301
P =
  phasor with properties:
      Mag: 600.0000
    phase: 0
Q =
  phasor with properties:
      Mag: 800.0000
    phase: -90
Fp = 0.6000
ph = 'leading'
```

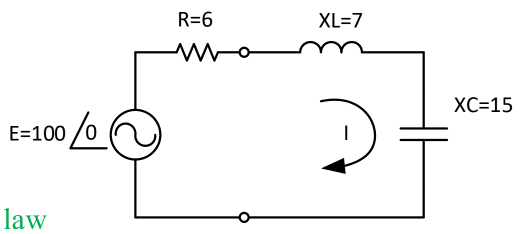

Fig. 3 (a)

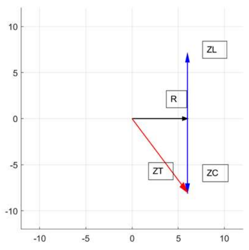

Fig. 3 (b)

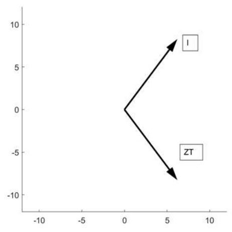

Fig. 3 (c)

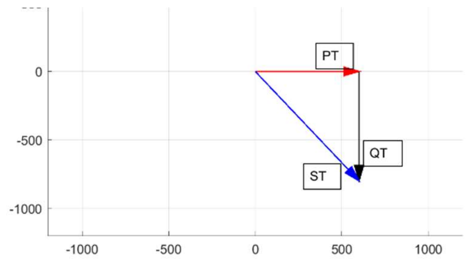

Fig. 3 (d)

2.5 Mesh Circuit Analysis using Matrices

The phasor tool box empowers the solution of simultaneous equations Z I = E in which all variables are phasors in matrix formulation:

I=Z \ E

The Phasor Tool Box has a method in the 'left divide operator' to do this job.

```
%express all E, Z and I in the phasor/polar form
Z1=phasor(1, 2, 'x2po'); Z2=phasor(4, -8, 'x2po');
Z3=phasor(0,6, 'x2po');
E1=phasor(8, 20);      E2=phasor(10, 0);
Z=[Z1+Z2,    -Z2        % Impedance matrix
    -Z2,        Z2+Z3 ];
E=[E1-E2;    E2];       % E is a column vector
%solution for loop currents
I=Z\E;                  %Solution of Z I=E equations
I(1)                    %current I1 in phasor form
ans =
  phasor with properties:
      Mag: 1.1536
```

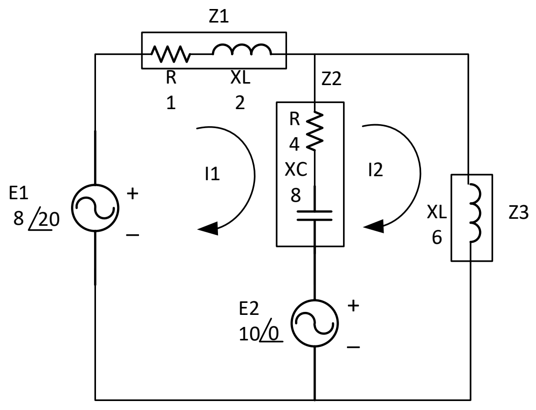

Fig. 4

```
   phase: -89.1673
I(2)                         %current I2 in phasor form
ans =
  phasor with properties:
      Mag: 1.0781
    phase: -53.4087
```

2.5 Maximum Power Transfer

The following example shows the use of Thevenin equivalent circuit to find the load impedance ZL across the terminals A-A', for maximum power transfer. All impedances are entered as polar quantities and source voltage as phasor. All output variables are in phasor/polar formats.

```
%PTB2_Ex_54B.mlx
Z1=phasor(9, 90); Z2=phasor(8, 0); E=phasor(10, 0);
zero=coeff(0); one=coeff(1);
%Write Z matrix with the terminals A-A'  shorted
Z= [ Z1,         -Z1,                zero
    -Z1,          Z1+Z1+Z1,          -Z1
     zero,        -Z1,               Z1+Z2];
% source vector for use in Thevenin calculation.
V=[E; zero; zero];
```

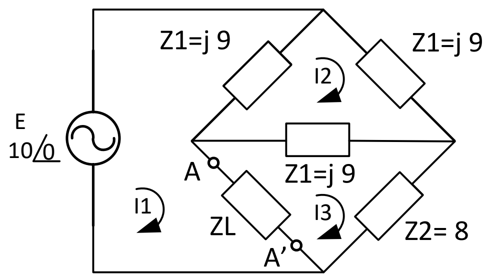

Fig. 5 (a)

```
Vex=phasor(1, 0);              %external voltage source for calculating Thevenin impedance Zth
Vx=[-Vex; zero; Vex];          %Source vector with E=0 and the external voltage Vex at the A-A'
                               %terminals
[Eth, Zth]=Thevenin(Z, V, Vx)  %the Thevenin voltage phasor and impedance in polar form
Eth =
  phasor with properties:
      Mag: 8.5440
    phase: 343.6861
Zth =
  phasor with properties:
      Mag: 5.5073
    phase: 82.4879
ZL=conj(Zth);                  %The load impedance for maximum power transfer

VL=Eth*ZL/(Zth+ZL);            %voltage across the load
IL=Eth/(Zth+ZL);               %Load current
%The apparent power Ss, real power Pp, reactive power Qq, power factor Fp and the
%leading/lagging phase
```

```
[Ss, Pp, Qq, Fp, ph]=PWR(VL, IL)
Ss =
  phasor with properties:
      Mag: 193.8805
    phase: -82.4879
Pp =
  phasor with properties:
Mag: 25.3472
    phase: 0
Qq =
  phasor with properties:
Mag: 192.2164
    phase: -90
Fp = 0.1307
ph = 'leading'
```

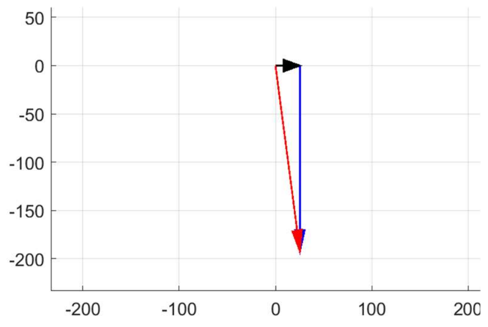

Fig. 5 (b)

2.6 3-Phase Circuits

Following example illustrates the drawing of 3-phase phasors. Three 3-phase voltage phasors each of 12 Vrms ∠45 are depicted on a complex plane. Add all three phase phasors graphically. The three phasors when added graphically, produce a zero voltage phasor.

```
%PTB2_Ex_10A.mlx
clf, clear
V(1)=phasor(12, 45);
V(2)=phasor(12, 45-120);
V(3)=phasor(12, 45-240);

phplot(V)    %plot the array of phasors, V

add_graph(V(1), V(2), V(3))
```

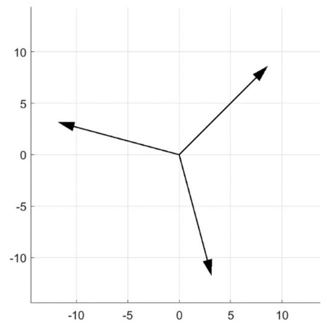

Fig. 6 (a)

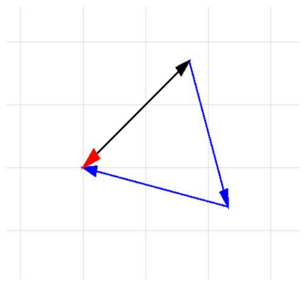

Fig. 6 (b)

2.7 Example of a 3-phase Network

This section shows the calculation of voltages, currents and powers in a 3-phase load network. The load consists of a Y-network and a Delta- network of identical impedances, see Fig. 7.

Line voltage is 200∠0,
delta-connected impedance are
Zab=Zbc=Zca= 6-j 8 ohms and
the Y-connected load impedances are
Zan=Zbn=Zcn= 4+j 3 ohms.

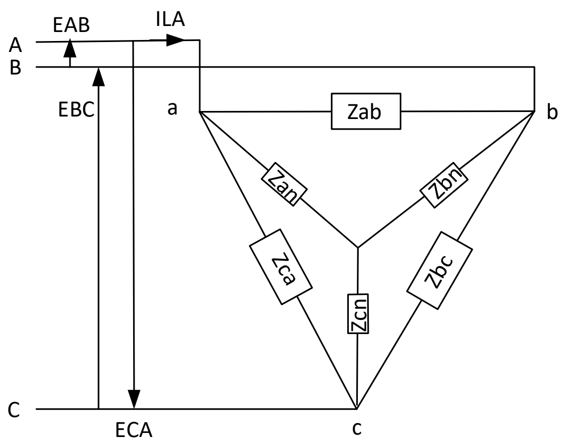

Fig. 7

```
%Line voltages
EAB=phasor(200, 0);            EBC=phasor(200, -120);        ECA=phasor(200, -240);
Zab=x2ph(6-j*8, 'po'); Zbc=Zab; Zca=Zab;      %Delta connected load
Zan=x2ph(4+j*3, 'po'); Zbn=Zan; Zcn=Zan;    %Y-connected load

%method1: first combine the Delta and Y loads then calculate powers.
%Convert delta-connected impedance load to Y-connected network combine the impedances in
%each branch
[Ean, Ebn, Ecn] = line2phase(EAB, EBC, ECA);     %conversion from Line to Phase voltages
[Zan2, Zbn2, Zcn2] = delta2wye(Zab, Zbc, Zca);     %conversion from Delta to Y-network load

%Parallel combination of two Y-connected loads
Zant=parallelZ([Zan,Zan2]);      Zbnt=parallelZ([Zbn,Zbn2]);     Zcnt=parallelZ([Zcn,Zcn2]);

%phase currents in the combined load network
Ian=Ean/Zant;       Ibn=Ebn/Zbnt;       Icn=Ecn/Zcnt;
%Ed is the array of phase voltages and Id is the array of line currents in the combined load network
Ed=[Ean, Ebn, Ecn];        Id=[Ian, Ibn, Icn];
%Calculation of powers in the load network, ST is the total apparent power, PT is the total real
%power, QT is the total reactive power, Fp is the power factor and leading/lagging information,
%each power type is a phasor object.
[ST, PT, QT, Fp, phase]=PWR_3phase(Ed, Id)
ST =
  phasor with properties:
         Mag: 1.4422e+04
         phase: -19.4400
       PT =
         phasor with properties:
         Mag: 1.3600e+04
          phase: 0
       QT =
          phasor with properties:
          Mag: 4.8000e+03
          phase: -90
       Fp = 0.9430
       phase = 'leading'
```

3. OBJECT ORIENTED PROGRAMMING IN MATLAB

This section presents the special programming features and design methodology of the phasor tool box. This section is only for designers. When this toolbox is used as the teaching tool in the class-room of students in the early circuits courses, do not need to know.

3.1 Variable Types of Arguments in Methods in an Object

This section shows how the object under the class phasor is designed to accept polar/phasor or complex vector as arguments. Output type of the object is only polar/phasor.

```
classdef phasor
   properties
      Mag
      phase
   end
methods
function obj = phasor(val1, val2, varargin)
if nargin > 0                    %if number of arguments is greater than zero
         if isnumeric(val1)      %if the arguments are numeric type
          obj.Mag = val1;        %assign val1 to the first property of the object, Magnitude
          obj.phase = val2;      %assign val1 to the second property of the object, phase
                 if nargin==2    %if the number of arguments is equal to 2
              type=' ';          %type is blank character
          else
              type=varargin{1}; %if the number of arguments is greater than 2, the type is assigned a
                                 %value or string of the third argument
          end
          %if the type is 'x2po' or 'x2ph', to indicating to convert the first two arguments from
          %complex to polar or phasor format
          if ((type=='x2po') |( type=='x2ph'))
              x=val1+j*val2;     %x is the complex value from the first two arguments
              obj.Mag=abs(x);    %the Mag property of the object is assigned to be the magnitude of
                                 % the complex value x
              if (type=='x2ph')  %if the type is 'x2ph' indicating the complex value is converted to a
                                 %phasor in which the Mag is a rms value
                 obj.Mag=obj.Mag/sqrt(2);
              end
              obj.phase=atand(val2/val1);        %the second property of the object, phase, is assigned
                                                 %the phase angle of the complex variable x in degrees
              if(val1<0)                         %make sure that the phase angle is in the proper
                                                 %quadrangle
                 obj.phase=180+obj.phase;
              end
          end
         else
           error('Value must be numeric')        %if the arguments are not numerical, output the error
                                                 % message.
         end
      end
     end
    end
```

3.2 Overloading Mathematical Operator for Phasor Algebra

This section shows how two objects under the class phasor are 'added' by overloaded symbol '+'. The following method is included in the class definition of phasor. Output type of the method is only polar/phasor.

```
function r = plus(o1,o2)
%o1 and o2 are two objects under the class phasor and r is the resulting output phasor
        %real part of the resulting phasor r is obtained from the o1 and o2 objects
        r1=[o1.Mag]*cosd([o1.phase]) + [o2.Mag]*cosd([o2.phase]);
        %imaginary parts of the resulting phasor r is obtained from o1 and o2 objects
        i2=[o1.Mag]*sind([o1.phase])+[o2.Mag]*sind([o2.phase]);
       rm= sqrt( r1^2 + i2^2);          %rm is the magnitude of the resulting phasor r
       rp=atand(i2/r1);                 %rp is the phase angle in degrees of the resulting phasor r
       if(r1<0)                         %determine the correct quadrant of the phase angle rp
         rp=180+atand(i2/r1);
       end
       r=phasor(rm, rp);                %r is the resulting output phasor
     end
```

4. PHASOR TOOL BOX AS THE TEACHING TOOL

The tool box is designed to be used as the teaching tool in the classroom of beginner students in the early electrical circuit course in the Electrical Engineering/Technology programs.. It requires students to have minimal scripting background. Using these tools is as easy as using conventional algebra for adding, subtraction, multiplication, division of phasors, and additionally visualize these operations in the complex plane.

Students must have the prerequisite knowledge of basic components of ac circuits such as resistor, inductor, capacitor, impedance, phase angle, rms value of magnitude, voltage, current and ohm's law, series and parallel combinations of impedances etc. These components are generally provided in the beginning few weeks of the ac circuit analysis course. To elaborate further, let us consider again a part of the example of ac circuit in sec. 2.4, Fig. 3(a), re-listed below.

```
% list the components and the voltage source as phasors
Line 1:   E=phasor(100, 0);
Line 2: ZR=phasor(6, 0);
Line 3: ZL=phasor(7, 90);
Line 4: ZC=phasor(15, -90);
Line 5: ZT=ZR+ZL+ZC;
Line 6: ZT=add_graph(ZR, ZL, ZC, ZT)
Line 7: I=E/ZT;
Line 8: phplot([E, I, ZT])
```

Line by line explanation of the code: The explanation specifies what the student is required to know/learn in using this tool for ac circuit analysis.

Line 1 requires the student to learn the syntax of specifying a voltage source in phasor form, using the Phasor Tool Box Class, **phasor** (rms volts, angle in degrees).
Line 2 requires the student to learn the syntax of specifying a resistor in phasor form, using the Phasor Tool Box Class, **phasor** (resistance value, 0 phase angle).
Line 3 requires the student to learn the syntax of specifying an inductor in phasor form, using the Phasor Tool Box Class, **phasor** (inductance value, 90 phase angle).
Line 4 requires the student to learn the syntax of specifying a capacitor in phasor form, using the Phasor Tool Box Class, **phasor** (capacitance value, -90 phase angle).
Line 5 requires the student to know the overall impedance of impedances in series by adding the impedances. The regular math **operator '+'** used for addition of phasors.

Line 6 requires the student to learn the syntax of Phasor Tool Box function, **add_graph** (phasor 1, phasor2,…).
Line 7 requires the student to know the ohm's law for calculating the current from given voltage E and total impedance ZT. The regular math **operator '/'** used for division of phasors.
Line 8 requires the student to learn the syntax of Phasor Tool Box function, **phplot** ([phasor 1, phasor2,…])

This tool box was developed during the teaching of the beginning course, AC Circuit Analysis, ECET 15200 at the Purdue University Northwest in 2017. After full development, this tool has not been tested in the classroom, so no data available of the students' response.

5. SUMMARY

This paper presents a MATLAB phasor toolbox for analyzing ac sinusoidal circuits in the Electrical and Computer Engineering Technology program. The toolbox has functions for conversion among complex, polar and phasor forms, for adding, subtraction, multiplication and division of phasors, plotting phasors on the complex plane and in their waveforms in the time domain. The toolbox also enables the calculation of Thevenin equivalent circuits, input impedances in networks, and calculation of apparent, reactive and real power in single and 3-phase ac circuits. The toolbox has the capability of analyzing ac circuits with sources of different frequencies. It is also capable of drawing Power Triangle diagrams and power factors in 3-phase networks.

References

[1] "Phasor Methods of AC Circuit Analysis", Jai P Agrawal (2018), www.kdp.Amazon.com, ISBN 978-1-72-0666028.
[2] "Phasor ToolBox For AC Circuit Analysis Using MATLAB", Jai P Agrawal and Omer Farook, ASEE Annual Conference 2018, Salt Lake City, June 25-27, 2018.
[3] "Introductory Circuit Analysis" Robert Boylestad, 13th ed., Pearson Education Inc.
[4] MATLAB v. 2017b, Mathworks Inc.
[5] "On Phasor Methods for A.C. Analysis", Peter C. Byrne, Volume: 27 issue: 1, page(s): 63-67 January 1, 1990, https://doi.org/10.1177/002072099002700111
[6] https://en.wikipedia.org/wiki/Phasor
[7] ″Mathematics for Engineers and Technologists ″, Huw Fox and William Bolton (2002). Butterworth-Heinemann. ISBN 978-0-08-051119-1.
[8] ″Basic AC Circuits", Clay Rawlins (2000), (2nd ed.), Newnes. ISBN 978-0-08-049398-5.
[9] "Electric Circuits and Networks", K. S. Suresh Kumar (2008).. Pearson Education India, ISBN 978-81-317-1390-7.
[10] "Pragmatic Electrical Engineering: Fundamentals", William J. Eccles (2011). Morgan & Claypool Publishers, ISBN 978-1-60845-668-0.
[11] "Introduction to Electric Circuits". Richard C. Dorf; James A. Svoboda (2010), (8th ed.). John Wiley & Sons. ISBN 978-0-470-52157-1.
[12] "Circuit Analysis: Theory and Practice", Allan H. Robbins; Wilhelm Miller (2012). (5th ed.). Cengage Learning. ISBN 1-285-40192-1.
[13] "Circuit Systems with MATLAB and PSpice", Won Y. Yang; Seung C. Lee (2008), John Wiley & Sons. ISBN 978-0-470-82240-1.
[14] "Phasor Representation of Alternating Quantities", Singh, Ravish R (2009). Mcgraw Hill Higher Education. ISBN 0070260966.