Linear Algebra Computational Tool for LaTeX

Ajit Kumar*[1] and Chetan Shirore[2]

[1]Department of Mathematics, Institute of Chemical Technology, Mumbai, India

[2]Department of Mathematics, Institute of Chemical Technology, Mumbai, India

Abstract

Linear algebra is used in different branches of science, engineering, and data science. There are many tools for doing computations on vectors and matrices. LaTeX is one of the most widely used typesetting systems for scientific publications, and there is often a need to type vectors and matrices inside LaTeX documents and perform different operations on them. We have developed a computational tool for linear algebra to deal with standard operations on vectors and matrices inside LaTeX. The standard practice of LaTeX users is to export computational results from other software and compile them inside LaTeX. This may be cumbersome when there are vast computations. The exported output from other software may need some editing before importing it into LaTeX as it may not be in LaTeX-compatible format or in the format that the user expects. The main aim of this paper is to give a brief introduction to the computational tool of linear algebra developed by us. This paper extends the series of basic computational tools that we developed [2, 3, 4, 5, 8, 10, 7]. It will reduce the dependence of LaTeX users on external software and can also be deployed for pedagogical uses.

1 Background and Introduction

Lua programming language is a scripting language which can be embedded across platforms. With `LuaTeX` [9] and `luacode` [1] packages, it is possible to use Lua in LaTeX. TeX or LaTeX has scope for programming [12]. However, with the weird internals of TeX, there are several limitations, especially for performing calculations on numbers in LaTeX documents.

There is a good scope to perform basic mathematical computations in LaTeX using Lua. This approach is suitable for performing standard operations on real and complex numbers, sets, integers, vectors, and matrices. It involves some complex intermingling of Lua and TeX. We have developed the `luamaths` [4], `luacomplex` [2], `luaset` [8], `luagcd` [3], `luamodulartables` [5], `luanumint` [7] and `luatruthtable` [10] packages based on this approach. These packages are designed from a pedagogical perspective and enhance the computational aspect of LaTeX.

*Corresponding Author, Email: a.kumar@ictmumbai.edu.in

We are extending the series by introducing a Lua-based linear algebra computational tool. It covers most of the numerical computations on vectors and matrices. No particular environment in LaTeX is required to use this tool. The time required for performing operations on vectors and matrices using the tool is not an issue due to dynamic memory management and garbage collection facility of Lua.

The article is organized in different sections. The methodology used for the development of the tool is discussed in the second section. The license and instructions for installation are given in the third section. The fourth section summarizes the key features of the tool. The pedagogical applications with a few illustrations are given in the fifth section. The resources used in the development of the tool are covered in the sixth session. The seventh section provides some limitations of the tool. The future plans and prospects of the research work are in the eighth section.

2 Development of the tool

The development of the linear algebra computational tool can be described in the following steps.

- The resources available in Lua are used for writing small blocks or chunks in Lua IDE. The abstraction of chunks is done by writing functions in Lua. These functions are put into a single Lua module.
- In the next step, the module is thoroughly tested in Lua independently of LaTeX. The necessary changes are made and restructuring of algorithms is done wherever required. The error handling mechanism is added to ensure the appropriate input from users.
- After testing stand alone Lua modules, the `luacode` package [1] is used to integrate the Lua module with LaTeX. This integration often gets complex as the intermingling of TeX and Lua is not straightforward. The customized environments and customized commands are written in the LaTeX package file to execute functions in the Lua module through LuaTeX. The `xkeyval` [14] package is also used in the development of the tool.
- Lua based LaTeX package is again thoroughly tested with its output in PDF file. TeX commands are modified as required.

3 Licensing and deployment of the tool

Lua is certified open-source software available. Its license is simple and liberal, which is compatible with GPL. This license allows users to freely copy, modify and distribute the file for any purpose and without restrictions. The tool is released under the LaTeX Project Public License v1.3c or later.

The installation of the linear algebra computational tool is similar to the plain latex package. It can be loaded with `\usepackage{lualinalg}` command in the preamble of the LaTeX document. The TeX file is to be compiled using the LuaLaTeX engine.

The `lualinalg` package is available on the CTAN repository and bundled with standard TeX distributions such as MikTeX and TeXLive. It can easily be used in the Overleaf online LaTeX editor. The source files of the tool need to be placed in the working directory of Overleaf.

4 Features of the tool

The tool mainly consists of two parts: operations on vectors and operations on matrices. This section summarizes key features of each of the part.

4.1 Operations on vectors

The vectors of reasonable size can be handled with ease. It supports vectors defined over the field of real and complex numbers. The meta-operations are used to evaluate complex expressions involving vectors. The following are some features in the vector part of the tool.

- Defining vectors : vectors are defined with the `\vectornew` command. The standard vector of dimension n with ith coordinate 1 and other co-ordinates 0 can be produced with additional specifications in the same command.
- Operations on vectors: the following operations on vectors can be performed by using different commands available in the tool.
  - ☐ Define vectors with specified coordinates.
  - ☐ Create vectors with random coordinates.
  - ☐ Print vectors with optional arguments.
  - ☐ Parse and change coordinates of vectors.
  - ☐ Create multiple copies of vectors.
  - ☐ Perform addition, subtraction and scalar multiplication.
  - ☐ Find dot and cross product of vectors.
  - ☐ Calculate the Euclidean norm, p-norm, sum norm, and sup-norm of a vector.
  - ☐ Obtain angle between two vectors.
  - ☐ Evaluate complex expressions involving vectors using meta-operators.
  - ☐ Parse coordinates of vectors for plotting.
  - ☐ Perform Gram-Schmidt orthogonalization process with the option of producing computations step-by-step.

4.2 Operations on Matrices

The matrices of reasonable size can be handled with ease. It supports matrices with real and complex number entries. The meta-operations are used to evaluate complex expressions involving matrices. The following are some features in the matrix part of the tool.

- Defining Matrices: Matrices are defined with the `\matrixNew` command. The identity matrix can be defined by adding specific argument in the same command.

- Operations on matrices: The following operations on matrices can be performed by using different commands in the tool.
  - ☐ Define matrices with specified entries.
  - ☐ Create matrices with random entries.
  - ☐ Obtain number of rows and columns of a matrix.
  - ☐ Fetch and change elements of matrices from specified rows and columns.
  - ☐ Obtain a submatrix of the matrix.
  - ☐ Augment matrix horizontally and vertically.
  - ☐ Perform Gauss-Jordan elimination with option of producing computations step-by-step.
  - ☐ Create copies of matrices.
  - ☐ Calculate the 1-norm, infinity-norm, max-norm and Frobenius-norm of a matrix.
  - ☐ Perform addition, subtraction, multiplication and scalar multiplication.
  - ☐ Determine inverse, transpose, conjugate, complex conjugate, rank and trace of a matrix.
  - ☐ Perform elementary row and column operations on matrices.
  - ☐ Obtain Reduced Row Echelon Form (RREF) of a matrix with the option of producing computations step-by-step.

5 Pedagogical applications

- The tool can assist in creation of interactive teaching modules in LaTeX when computations on vectors and matrices are involved. A variety of problems on vectors and matrices can be generated for assignments of students by using the tool. The final computational results of problems can also be provided by using available commands in the tool.
- The beamer is a document class in LaTeX to create presentations. The tool can be used in beamer to illustrate various concepts of linear algebra while teaching.
- The tool can also be used to illustrate step-by-step procedures to obtain the rref form of a matrix.

Lualinalg code 1: Step-by-step computations of rref

```
\def\z{{{lfrac(3,2),lcomplex(3,3),5},{6,7,8},{7,0,9}}}
\matrixNew{M}{\z}
\[M = \matrixPrint{M}\]
\matrixRREFSteps{M}
```

Lualinalg code 1 generates the following output.

$$M = \begin{bmatrix} \frac{3}{2} & 3+3\mathrm{i} & 5 \\ 6 & 7 & 8 \\ 7 & 0 & 9 \end{bmatrix}$$

Step 1: Divide row 1 by $\frac{3}{2}$.

$$\begin{bmatrix} 1 & 2+2\mathrm{i} & \frac{10}{3} \\ 6 & 7 & 8 \\ 7 & 0 & 9 \end{bmatrix}$$

Step 2: Multiply row 1 by 6 and subtract it from row 2.

$$\begin{bmatrix} 1 & 2+2\mathrm{i} & \frac{10}{3} \\ 0 & -5-12\mathrm{i} & -12 \\ 7 & 0 & 9 \end{bmatrix}$$

Step 3: Multiply row 1 by 7 and subtract it from row 3.

$$\begin{bmatrix} 1 & 2+2\mathrm{i} & \frac{10}{3} \\ 0 & -5-12\mathrm{i} & -12 \\ 0 & -14-14\mathrm{i} & \frac{-43}{3} \end{bmatrix}$$

Step 4: Divide row 2 by $-5-12\mathrm{i}$.

$$\begin{bmatrix} 1 & 2+2\mathrm{i} & \frac{10}{3} \\ 0 & 1 & \frac{60}{169}+\frac{-144}{169}\mathrm{i} \\ 0 & -14-14\mathrm{i} & \frac{-43}{3} \end{bmatrix}$$

Step 5: Multiply row 2 by $2+2\mathrm{i}$ and subtract it from row 1.

$$\begin{bmatrix} 1 & 0 & \frac{466}{507}+\frac{168}{169}\mathrm{i} \\ 0 & 1 & \frac{60}{169}+\frac{-144}{169}\mathrm{i} \\ 0 & -14-14\mathrm{i} & \frac{-43}{3} \end{bmatrix}$$

Step 6: Multiply row 2 by $-14-14\mathrm{i}$ and subtract it from row 3.

$$\begin{bmatrix} 1 & 0 & \frac{466}{507}+\frac{168}{169}\mathrm{i} \\ 0 & 1 & \frac{60}{169}+\frac{-144}{169}\mathrm{i} \\ 0 & 0 & \frac{1301}{507}+\frac{-1176}{169}\mathrm{i} \end{bmatrix}$$

Step 7: Divide row 3 by $\frac{1301}{507}+\frac{-1176}{169}\mathrm{i}$.

$$\begin{bmatrix} 1 & 0 & \frac{466}{507}+\frac{168}{169}\mathrm{i} \\ 0 & 1 & \frac{60}{169}+\frac{-144}{169}\mathrm{i} \\ 0 & 0 & 1 \end{bmatrix}$$

Step 8: Multiply row 3 by $\frac{466}{507} + \frac{168}{169}$i and subtract it from row 1.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \frac{60}{169} + \frac{-144}{169}\text{i} \\ 0 & 0 & 1 \end{bmatrix}$$

Step 9: Multiply row 3 by $\frac{60}{169} + \frac{-144}{169}$i and subtract it from row 2.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Some illustrations

- The tool can be used to solve numerical problems on vectors and matrices, and to verify results. We mention a few of the computations for illustration.

Lualinalg code 2: Solving problems on vectors the linear algebra computational tool

```
\vectorNew{v1}{{3,8,5}} \vectorNew{v2}{{4,7,4}} \vectorNew{v3}{{5,4,3}}
\[v1=\left(\vectorPrint{v1}\right),\]
   \[v2=\left(\vectorPrint{v2}\right),\]
   \[v3=\left(\vectorPrint{v3}\right)\]
\vectorGramSchmidtSteps[brckt=round,truncate=3]{{'v1','v2','v3'}}
```

Lualinalg code 2 produces a step-by-step computation of Gram-Schmidt orthogonalization process on vectors $v1 = (3, 8, 5)$, $v2 = (4, 7, 4)$ and $v3 = (5, 4, 3)$. Following is the output of this procedure.

$$v1 = (3, 8, 5)\,,$$
$$v2 = (4, 7, 4)\,,$$
$$v3 = (5, 4, 3)$$

Take given vectors as $v_1, \ldots, v_3$ in order.
Step 1:

$$u_1 = v_1 = (3, 8, 5)$$
$$e_1 = \frac{u_1}{||u_1||} = (0.303, 0.808, 0.505)$$

Step 2:

$$u_2 = v_2 - \sum_{j=1}^{1} proj_{u_j}(v_2) = (1.306, -0.184, -0.49)$$

$$e_2 = \frac{u_2}{||u_2||} = (0.928, -0.131, -0.348)$$

Step 3:

$$u_3 = v_3 - \sum_{j=1}^{2} proj_{u_j}(v_3) = (0.247, -0.66, 0.907)$$

$$e_3 = \frac{u_3}{||u_3||} = (0.215, -0.574, 0.79)$$

By using `\vectorEuclidNorm` and `\vectorDot` commands in the tool, it can be verified that e_1, e_2 and e_3 are pair-wise orthogonal unit vectors.

- The tool can be used to solve examples on matrices, and to verify results numerically. As an example, consider the following system of linear equations.

$$\frac{1}{4}x + y + z = 6, 2x - 3y - z = 9, 4x - y + z = 10 + \mathrm{i}$$

With the command `\matrixGaussJordanSteps` in tool, it is possible to produce step-by-step computation of Gauss-Jordan elimination of an augmented matrix.

Lualinalg code 3: Solving linear equations using the linear algebra computational tool.

```
\def\a{{{\lfrac(1,4),1,1},{2,-3,-1},{4,-1,1}}}
\def\b{{{6},{9},{\lcomplex(10,1)}}}
\matrixNew{S}{\a}
\matrixNew{T}{\b}
\matrixConcatH{W}{S}{T}
\[W = \matrixPrint{W}\]
\matrixGaussJordanSteps{S}{T}
```

Lualinalg code 3 generates the following output.

$$W = \begin{bmatrix} \frac{1}{4} & 1 & 1 & 6 \\ 2 & -3 & -1 & 9 \\ 4 & -1 & 1 & 10 + \mathrm{i} \end{bmatrix}$$

Step 1: Divide row 1 by $\frac{1}{4}$.

$$\begin{bmatrix} 1 & 4 & 4 & 24 \\ 2 & -3 & -1 & 9 \\ 4 & -1 & 1 & 10 + \mathrm{i} \end{bmatrix}$$

Step 2: Multiply row 1 by 2 and subtract it from row 2.

$$\begin{bmatrix} 1 & 4 & 4 & 24 \\ 0 & -11 & -9 & -39 \\ 4 & -1 & 1 & 10+\mathrm{i} \end{bmatrix}$$

Step 3: Multiply row 1 by 4 and subtract it from row 3.

$$\begin{bmatrix} 1 & 4 & 4 & 24 \\ 0 & -11 & -9 & -39 \\ 0 & -17 & -15 & -86+\mathrm{i} \end{bmatrix}$$

Step 4: Divide row 2 by -11.

$$\begin{bmatrix} 1 & 4 & 4 & 24 \\ 0 & 1 & \frac{9}{11} & \frac{39}{11} \\ 0 & -17 & -15 & -86+\mathrm{i} \end{bmatrix}$$

Step 5: Multiply row 2 by 4 and subtract it from row 1.

$$\begin{bmatrix} 1 & 0 & \frac{8}{11} & \frac{108}{11} \\ 0 & 1 & \frac{9}{11} & \frac{39}{11} \\ 0 & -17 & -15 & -86+\mathrm{i} \end{bmatrix}$$

Step 6: Multiply row 2 by -17 and subtract it from row 3.

$$\begin{bmatrix} 1 & 0 & \frac{8}{11} & \frac{108}{11} \\ 0 & 1 & \frac{9}{11} & \frac{39}{11} \\ 0 & 0 & \frac{-12}{11} & \frac{-283}{11}+\mathrm{i} \end{bmatrix}$$

Step 7: Divide row 3 by $\frac{-12}{11}$.

$$\begin{bmatrix} 1 & 0 & \frac{8}{11} & \frac{108}{11} \\ 0 & 1 & \frac{9}{11} & \frac{39}{11} \\ 0 & 0 & 1 & \frac{283}{12}+\frac{-11}{12}\mathrm{i} \end{bmatrix}$$

Step 8: Multiply row 3 by $\frac{8}{11}$ and subtract it from row 1.

$$\begin{bmatrix} 1 & 0 & 0 & \frac{-22}{3}+\frac{2}{3}\mathrm{i} \\ 0 & 1 & \frac{9}{11} & \frac{39}{11} \\ 0 & 0 & 1 & \frac{283}{12}+\frac{-11}{12}\mathrm{i} \end{bmatrix}$$

Step 9: Multiply row 3 by $\frac{9}{11}$ and subtract it from row 2.

$$\begin{bmatrix} 1 & 0 & 0 & \frac{-22}{3}+\frac{2}{3}\mathrm{i} \\ 0 & 1 & 0 & \frac{-63}{4}+\frac{3}{4}\mathrm{i} \\ 0 & 0 & 1 & \frac{283}{12}+\frac{-11}{12}\mathrm{i} \end{bmatrix}$$

These computations show that

$$x = \frac{-22}{3}+\frac{2}{3}\mathrm{i}, y = \frac{-63}{4}+\frac{3}{4}\mathrm{i} \text{ and } z = \frac{283}{12}+\frac{-11}{12}\mathrm{i}$$

is the solution. It also illustrates the Gauss-Jordan elimination to obtain the solution of a system of linear equations.

- Apart from computations on vectors, the tool can be used with other packages that can plot vectors defined over the field of real numbers in 2 or 3 dimensions. The tool thus facilitates graphical illustration of various concepts on vectors. As an example, the package can be used with the `tikz` package. Lualinalg code 4 illustrates the plotting of vectors in a 3-D plane using the `luavector` and `tikz` package. It produces figure 1.

Lualinalg code 4: Plotting vectors in 3-dimensions with the linear algebra computational tool

```
\documentclass{article}
\usepackage{tikz,tikz-3dplot,lualinalg}
\begin{document}
\tdplotsetmaincoords{60}{100}
\begin{tikzpicture}[scale=1,
    tdplot_main_coords,
    axis/.style={->,thick},
    vector/.style={-stealth,very thick},
    vector guide/.style={dashed,red,thick}]
\vectorNew{o}{{0,0,0}}
\vectorNew{e1}{{5,0,0}}
\vectorNew{e2}{{0,3,0}}
\vectorNew{e2n}{{0,-3,0}}
\vectorNew{e3}{{0,0,6}}
\vectorNew{f}{{1,2,0}}
\vectorNew{g}{{-2,1,2}}
% Axes
\draw [axis] \vectorParse{o}-- \vectorParse{e1} node [below left] {$x$};
\draw [axis] \vectorParse{e2n}-- \vectorParse{e2} node [right] {$y$};
\draw [axis] \vectorParse{o}-- \vectorParse{e3} node [above] {$z$};
% Plotting Vectors
\draw [vector, red] \vectorParse{o} --\vectorParse{f};
```

```
\draw [vector, blue] \vectorParse{o} --\vectorParse{g};
\vectorCross{h}{f}{g}
\draw [vector, green] \vectorParse{o} --\vectorParse{h};
% Labels
\node [below right] at \vectorParse{f} {$f$};
\node [above left] at \vectorParse{g} {$g$};
\node [left] at \vectorParse{h} {$f \times g$};
\draw[vector guide, black] \vectorParse{h} --
   (\vectorGetCoordinate{h}{1},0,0) node [below right]
   {$x=\vectorGetCoordinate{h}{1}$};
\draw[vector guide, black] \vectorParse{h} --
   (0,\vectorGetCoordinate{h}{2},0) node [below left, xshift = 0.3cm]
   {$y=\vectorGetCoordinate{h}{2}$};
\draw[vector guide, black] \vectorParse{h} --
   (0,0,\vectorGetCoordinate{h}{3}) node [right]
   {$z=\vectorGetCoordinate{h}{3}$};
\end{tikzpicture}
\end{document}
```

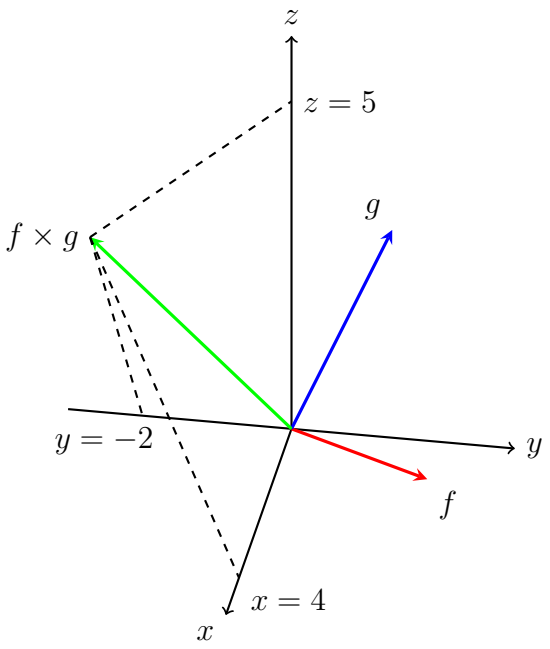

Figure 1: Plotting of 3-D Vectors with the luavector and tikz package

Customized usage

The commands available in the package can be used for performing further operations on matrices and vectors. These commands can be modified or extended as well. The tool has no functions to check whether the matrix is involutory and to check the orthogonality of vectors. It is easily possible to construct these functions using the tool. Lualinalg code 5 illustrates this.

Lualinalg code 5: Customized usage of the tool

```
\documentclass{article}
\usepackage{lualinalg}
\begin{document}
\begin{luacode}
function matrix.checkInvolutory(m)
 if #m ~= #m[1] then error( "Matrix is not square.") end
 idn = matrix(#m, 'I')
return matrix.chqeql(m^2,idn)
end

function vector.checkOrthoVecs(v,w)
return lnumChqEql(vector.dot(v,w),0)
end
\end{luacode}

\newcommand\matrixChkInvolutory[1]{%
    \directlua{%
        tex.print(tostring(matrix.checkInvolutory(matrices['#1'])))
    }%
}%

\newcommand\vectorChkOrtho[2]{%
    \directlua{%
        tex.print(tostring(vector.checkOrthoVecs(vectors['#1'],vectors['#2']
            )))%
    }%
}%

\def\s{{{-5,-8,0},{3,5,0},{1,2,-1}}}
\matrixNew{m}{\s}
\(m = \matrixPrint{m}\)

It is \matrixChkInvolutory{m} that matrix \(m\) is Involutory.

\vectorNew{v1}{{lcomplex(0,1),lcomplex(-1,-1), 1}}
\vectorNew{v2}{{lcomplex(-1,1),lcomplex(0,2), lcomplex(1,1)}}
\(v1=(\vectorPrint{v1})\) and \(v2=(\vectorPrint{v2})\).

It is \vectorChkOrtho{v1}{v2} that vectors \(v1\) and \(v2\) are orthogonal.

\end{document}
```

Lualinalg code 5 generates the following output.

$$m = \begin{bmatrix} -5 & -8 & 0 \\ 3 & 5 & 0 \\ 1 & 2 & -1 \end{bmatrix}$$

It is true that matrix m is involutory.
$v1 = (3, 8, 5)$ and $v2 = (-1 + \mathrm{i}, 2\mathrm{i}, 1 + \mathrm{i})$.
It is true that vectors vectors $v1$ and $v2$ are orthogonal.

6 Resources used

- Techniques and methods used in development of the tool do not need any special resources. All the techniques are platform-independent and work on the standard operating systems. The technique mainly uses Lua and LaTeX, which are platform-independent and do not need strong hardware resources.

- No proprietary software and tools are used in the development of the tool. The tool is made available as freeware and open source. This is important as the research work can further be extended without any restrictions. This is within the philosophy of open source and freeware tools that are available for the mathematics community.

7 Known issues and limitations

- The tool uses double precision for floating-point numbers. It represents each number with 64 bits, 11 of which are used for the exponent. The double-precision floating-point numbers can represent numbers with roughly 16 significant decimal digits. The tool supports small and big numbers. They can be input in the usual scientific notation. The math library in Lua defines constants with the maximum `math.maxinteger` and the minimum `math.mininteger` values for an integer. The result wraps around when there is a computational operation on integers that would result in a value smaller than the `mininteger` or larger than the `maxinteger`. It means that the computed result is the only number between the `miniinteger` and `maxinteger`. The bits after 64 bits are not taken into account.

- The symbolic computations on vectors and matrices are not supported at present.

- The error handling mechanism in the tool is not robust. There are some custom errors included in the package. However the package mostly depends on error handling mechanism of Lua. The error handling can be strengthened in future updates of the tool.

8 Future Plans and Prospects of the Research Work

- The tool currently supports only numerical computations on vectors and matrices. The table in a Lua is a data type that implements an associative array. This feature is used in packages to define and store vectors and matrices. This approach is close to

object-oriented programming. It will allow easy conversion of algorithms in packages for symbolic computations. Future package updates will consider algorithm conversions to support symbolic calculations.

- The approach of embedding Lua in LaTeX can be extended in several other ways. With `luamplib` [6] and `metapost` [11] libraries, it is possible to plot graphs of functions in LaTeX in a native way. This sort of package can be developed in the future to enhance the graphical aspect of LaTeX using Lua.

- Lua supports procedural programming, object-oriented programming, functional programming, data-driven programming, and data description. It is thus possible to upgrade the developed tool with Graphical User Interface (GUI) to facilitate interactive computations on vectors and matrices. The tool can also have a facility to import and export computations in LaTeX-compatible format.

- Lua-based mini Computer Algebra System can be developed and integrated into LaTeX. It will consist of Lua packages designed for performing symbolic mathematical computations. There is good scope for developing such Computer Algebra System. There have already been efforts in this direction. There is a `symmath lua` project on GitHub. It is available on this link [13].

References

[1] *luacode package page*. URL: `https://ctan.org/pkg/luacode` (visited on 03/10/2022).

[2] *luacomplex package page*. URL: `https://ctan.org/pkg/luacomplex` (visited on 12/29/2022).

[3] *luagcd package page*. URL: `https://ctan.org/pkg/luatruthtable` (visited on 12/30/2022).

[4] *luamaths package page*. URL: `https://ctan.org/pkg/luamaths` (visited on 12/27/2022).

[5] *luamodulartables package page*. URL: `https://ctan.org/pkg/luamodulartables` (visited on 12/31/2022).

[6] *luamplib package*. URL: `https://mirror.kku.ac.th/CTAN/macros/luatex/generic/luamplib/luamplib.pdf` (visited on 02/22/2022).

[7] *luanumint package page*. URL: `https://ctan.org/pkg/luanumint` (visited on 08/04/2023).

[8] *luaset package page*. URL: `https://ctan.org/pkg/luaset` (visited on 12/28/2022).

[9] *LuaTeX package page*. URL: `https://ctan.org/pkg/luatex` (visited on 01/22/2022).

[10] *luatruthtable package page*. URL: `https://ctan.org/pkg/luatruthtable` (visited on 09/18/2022).

[11] *Metapost language*. URL: `https://www.tug.org/docs/metapost/mpman.pdf` (visited on 02/22/2022).

[12] William M. Richter. "TEX and scripting languages". In: *TUGboat* 25.1 (80 2004), pp. 71–88. ISSN: 0896-3207. URL: `https://tug.org/TUGboat/tb25-1/richter.pdf`.

[13] *Symbolic Mathematics in Lua*. 2022. URL: https://christopheremoore.net/symbolic-lua (visited on 05/22/2022).

[14] *Xkeyval package*. URL: https://ctan.org/pkg/xkeyval?lang=en (visited on 06/12/2021).