The Electronic Journal of Mathematics and Technology, Volume 18, Number 1, ISSN 1933-2823

Numerical methods in LaTeX using Lua

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

[1]Department of Mathematics, K.T.H.M. College, Nashik, India.
Email: mathsbeauty@gmail.com
[2]Department of Mathematics, Institute of Chemical Technology, Mumbai, India.
Email: a.kumar@ictmumbai.edu.in

Abstract

This article introduces ways of performing numerical methods inside LaTeX documents using a scripting programming language Lua. It mainly focuses on methods that use the load function in Lua to evaluate functions within the mathematics environment of Lua. These methods align with our work of creating computational packages for LaTeX using Lua. The article includes a few simple Luacodes which illustrates different ways of using Lua for performing numerical methods in LaTeX. The one purpose of using Lua for LaTeX documents is to reduce the dependence of LaTeX users on external software for computations. The other purpose is to develop methods and tools that can be deployed for pedagogical purposes.

1 Background and introduction

Lua [1] is a portable scripting language that we have used for developing computational packages for LaTeX. We have developed luamaths [5], luacomplex [2], luaset [8], luagcd [3], luatruthtable [9], lualinalg [4], and luamodulartables [6] packages. The installation of these packages are similar to the installation of plain latex packages. The packages can be loaded by using `\usepackage{package name}` in preamble of the LaTeX documents. LaTeX files need to be compiled using the LuaLaTeX engine. The research article "Basic Mathematical Computations inside LaTeX using Lua" [10] describes some of these packages.

The paper is organized into different sections. The use of the load function in Lua to evaluate mathematical functions inside LaTeX documents is described in the second section. The third section introduces the luanumint [7] package with a few illustrations and customized usage. The fourth section gives some limitations of the methods used. The fifth section provides conclusions and the last section describes the proposed future work.

*Corresponding Author

2 Load function in Lua

The load function takes a chunk as its input. A chunk in Lua is simply a sequence of statements. These statements are executed in order. The load function compiles the chunk and converts it into a function that can be called to execute the chunk. It has the following syntax.

```
load (strfun [, src [, mode [, env]]])
```

The `strfun` can be a string or function. If it is a string, it represents the Lua code that is to be executed. If `strfun` is a function, the load function calls it repeatedly to get the chunk pieces. Every time it returns a string that concatenates with the previous results. An empty string, nil, or no value indicates the end of the Lua code to be executed. The `src` is used as the source for error messages and debug information in the `lua_Debug` interface. The mode can be: ‘b’ (only binary chunks), ‘t’ (only text chunks), or ‘bt’ (both binary and text). The default is “bt”. The ‘env’ specifies the environment for variables in the chunk.

The load function can be used to evaluate mathematical functions. It is handy to evaluate mathematical functions in LaTeX documents as the only input that LaTeX can accept from Lua is a string. Luacode 1 illustrates the use of the load function to evaluate the real-valued functions of real variable(s). It defines LaTeX commands: `\LuaFnOne` and `\LuaFnTwo` to evaluate the real-valued functions of one and two real variables, respectively. These commands have an optional parameter `trun` to truncate the decimal places to the desired number.

Luacode 1: Load function in Lua.

```
\documentclass{article}
\usepackage{luacode,xkeyval}
\begin{luacode}
--function to round numbers.
function numrnd(num, numDecimalPlaces)
   local mult = 10^(numDecimalPlaces or 0)
   return math.floor(num * mult + 0.5) / mult
end
\end{luacode}
\makeatletter
% ========= KEY DEFINITIONS =========
\define@key{someop}{trun}{\def\mop@onex{#1}}%
% ========= KEY DEFAULTS =========
\setkeys{someop}{trun=4}%
% ========= Defining Command =========
\newcommand{\luaFnOne}[3][]{{%
\setkeys{someop}{#1}%
\directlua{%
exp = "("..\luastring{#2}..")"
local f = load("return function(x) return "..exp.."end",nil,"t",math)()
tex.print(numrnd(f(#3),\mop@onex))
}%
}%
```

```
}%
\newcommand{\luaFnTwo}[4][]{{%
\setkeys{someop}{#1}%
\directlua{%
exp = "("..\luastring{#2}..")"
local f = load("return function(x,y) return " ..exp.. "end",nil,"t",math)()
tex.print(numrnd(f(#3,#4),\mop@onex))
}%
}%
}%
\makeatother
\begin{document}
\luaFnOne[trun=4]{x^2+x^3}{0.576} \\
\luaFnTwo[trun=4]{sin(x)+cos(y)}{0.96}{0.36}
\end{document}
```

On compiling the LaTeX document (Luacode 1) with LuaLaTeX engine, it outputs the following.

0.5229
1.7551

3 Numerical methods in LaTeX

We have effectively used the load function with the mathematics environment of Lua in the development of the luanumint package [7], which facilitates the numerical integration of the real-valued functions of a real variable over the closed and bounded intervals. The package provides commands to find numerical integration using the mid-point, trapezoidal, and Simpson's one-third and three-eighth rules. The package can assist in creating various problems on numerical integration with their solutions. The results obtained using different methods of numerical integration can be compared. It can save users' efforts of calculating numerical integrals in external software and copying them inside LaTeX documents.

Table 1 illustrates commands in the `luanumint` package.

LaTeX input	Result
`$\int_{1}^{3}\sqrt{12+\cos(x^3)} dx =\luaMidpt[a=1,b=3,n=4]{sqrt(12+cos(x^3))}$`	$\int_1^3 \sqrt{12+\cos(x^3)}dx = 6.9448$

`$\int_{1}^{2}\sin(x)dx =\luaTrapz[a=1,b=2,n=5]{sin(x)}$`	$\int_1^2 \sin(x)dx = 0.9533$
`$\int_{0}^{1}\cos(x+1) dx=\luaSimpsonOneThird [a=0,b=1,n=4,trun=6]{cos(x+1)}$`	$\int_0^1 \cos(x+1)dx = 0.067828$
`$\int_{0}^{3}\sin(x) dx =\luaSimpsonThreeEighth [a=0,b=3,trun=6]{sin(x)}$`	$\int_0^3 \sin(x)dx = 7.337166$

Table 1: Illustrations of commands in the luanumint package.

Apart from using the load function in Lua, other techniques in Lua are used to produce step-by-step calculations of the numerical integration.

Luacode 2: The luaTrapzSteps command.

```
\begin{dmath*}
\int_{0}^{1}\sqrt{1+\cos^3(x)}dx
\luaTrapzSteps[a=0,b=1,n=5,trun=6]{sqrt(1+(cos(x))^3)}
\end{dmath*}
```

Luacode 2 generates the output shown in Table 2.

$$\int_0^1 \sqrt{1+\cos^3(x)}dx = 0.1\left[f(0)+2f(0.2)+2f(0.4)+2f(0.6)+2f(0.8)+f(1.0)\right]$$
$$= 0.1\left(1.414214+2.786671+2.669371+2.499761+2.313596+1.075978\right)$$
$$= 1.275959$$

Table 2: The luaTrapzSteps command.

The `breqn` package is loaded to display and align step-by-step calculations properly. Advanced users can customize the code to achieve the desired formatting of step-by-step computations. It can also be used to illustrate the convergence of numerical integrals to the actual value.

Luacode 3: Customized use of the luanumint package.

```
\documentclass{article}
```

```
\usepackage{luanumint,longtable,booktabs}
\begin{luacode}
function iterint()
   local itbl = {}
   v = 1
   for m = 1, 6, 1 do
      itbl[v] = m .. "&\\luaTrapz[a=0,b=1,trun=6,n=" .. m .. "]{sin(x^2)}"
      v = v + 1
   end
   tex.print(table.concat(itbl, "\\\\"))
end
\end{luacode}
\newcommand{\itern}{\directlua{iterint()}}
\begin{document}
\begin{longtable}{cc}
\toprule
$n$ & Approximate value of the integrall \\ \midrule
\itern
\bottomrule
\end{longtable}
\end{document}
```

Luacode 3 generates the output shown in Table 3. It lists the values of numerical integration of the function $f(x) = \sin(x^2)$ over $[0, 1]$ using the trapezoidal rule with different number of sub-intervals.

n	Approximate value of the integral
1	0.420735
2	0.33407
3	0.320525
4	0.315975
5	0.313903
6	0.312785

Table 3: Customized use of the luanumint package.

Apart from numerical integration, it is also possible to perform other methods, such as the bisection method, to find roots of equations. Luacode 4 provides this code. Further, it is possible to customize the code to produce a table of iterations.

Luacode 4: Customized use of the luanumint package.

```
\documentclass{article}
\usepackage{luacode,xkeyval}
\begin{luacode}
function numrnd(num, numDecimalPlaces)
```

```
    local mult = 10^(numDecimalPlaces or 0)
    return math.floor(num * mult + 0.5) / mult
end
function bisect(f, a, b, e)
    if f(a) * f(b) >= 0 then
        error("The values of a and b are not selected properly.")
    end
    local k = 1
    local test = true
    while test do
        c = (a + b) / 2
        if f(a) * f(c) < 0 then
            b = c
        else
            a = c
        end
        k = k + 1
        test = (math.abs(f(c)) > e)
    end
    return c
end

\end{luacode}
\makeatletter
% ========= KEY DEFINITIONS =========
\define@key{someop}{trun}{\def\mop@onex{#1}}%
% ========= KEY DEFAULTS =========
\setkeys{someop}{trun=4}%
% ========= Defining Command =========
\newcommand{\luaBisect}[5][]{{%
\setkeys{someop}{#1}%
\directlua{%
exp = "("..\luastring{#2}..")"
local f = load("return function(x) return "..exp.."end",nil,"t",math)()
tex.print(numrnd(bisect(f,#3,#4,#5),\mop@onex))
}%
}%
}%
\makeatother
\begin{document}
The root of function $\luaBisect[trun=6]{x^3-2.2369}{1}{3}{0.001}$ by the
    bisection method is $1.307861$, when the specified error is $0.001$.
\end{document}
```

Luacode 4 generates the following output.

The root of function $f(x) = x^3 - 2.2369$ by the bisection method is 1.307861, when the specified error is 0.001.

4 Limitations and known issues

The computational packages that we developed use double precision for floating-point numbers. It represents each number with 64 bits, 11 of which are used for the exponent. Double-precision floating-point numbers can represent numbers with roughly 16 significant decimal digits. This representation of numbers is inherited from Lua. The handling of small and big numbers inside packages depends entirely on Lua. 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`.

5 Conclusions

The load function in Lua can effectively be used for evaluation of mathematical functions inside LaTeX documents. This paper included a few simple Luacodes to illustrate numerical methods inside LaTeX documents. A variety of different applications are possible. The load function in Lua can also be used for plotting graphs inside LaTeX documents. Luacodes can be constructed to execute other numerical methods inside LaTeX documents.

6 Future work

After developing computational packages for LaTeX, we aim to combine these packages into a single module. This would help to achieve the uniform syntax of commands in different packages and strengthen the error-handling mechanism of different packages. In order to add better support for handling small and big numbers, we plan to use some external number-precision library.

An attempt will also be made to support symbolic computations and develop a portable computer algebra system inside LaTeX. Lua supports procedural programming, object-oriented programming, functional programming, data-driven programming, and data description. It is thus possible to develop a tool with Graphical User Interface (GUI) to facilitate interactive computations. The tool can also have a facility to import and export computations in LaTeX-compatible format. Another proposed work is developing a Lua-based package for 3-D plotting inside LaTeX documents. The package would serve the purpose of illustrating various concepts graphically inside LaTeX documents.

References

[1] *Lua Programming Language*. URL: https://www.lua.org (visited on 03/26/2022).

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

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

[4] *lualinalg package page*. URL: https://ctan.org/pkg/lualinalg (visited on 01/01/2024).

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

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

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

[10] Chetan Shirore and Ajit Kumar. "Basic Mathematical Computations inside LaTeX using Lua". In: *The Electronic journal of Mathematics and Technology* 17.1 (2023). ISSN: 1933-2807. URL: https://php.radford.edu/~ejmt/deliverAbstract.php?paperID=eJMT_v17n1n1.