The luanumint package in LaTeX

Chetan Shirore* and Ajit Kumar

November 17, 2023

1 Introduction

The `luanumint` package is developed using Lua to find the numerical integration of real-valued functions of a real variable over 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 `loadstring` command is used to load and evaluate functions at different points in the mathematics environment of Lua. The package also provides commands to find numerical integration with step-by-step calculations. The package's commands have an optional argument to round off the numbers to the desired number of decimal places. 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. 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 doing computations involving numerical integration in external software and copying them inside LaTeX documents.

2 Installation and License

The installation of `luanumint` package is similar to plain latex package, where the `.sty` file is in LaTeX directory of texmf tree. The package can be included with `\usepackage{luanumint}` command in the preamble of the LaTeX document. A TeX file is to be compiled using the LuaLaTeX engine.

The `luanumint` package is released under the LaTeX Project Public License v1.3c or later. The complete license text is available at http://www.latex-project.org/lppl.txt. It is developed in Lua. Lua is available as a certified open-source software. Its license is simple and liberal, which is compatible with GPL.

3 Rules for numerical integration

Let f be a real-valued function of a real variable which is Riemann-integrable over the closed and bounded interval $[a, b]$. The rules of numerical integration are used to calculate an approximate value of the integral $\int_a^b f(x)dx$.

Let n denote a number of subintervals of equal length of the closed and bounded interval $[a, b]$ corresponding to a partition $P = \{a = x_0, x_1, x_2, x_n = b\}$ of $[a, b]$. The points of a partition P are equally spaced with distance $h = \frac{b-a}{n}$. The following are some commonly used rules for numerical integration.

- **The Mid-point rule:**

$$\int_a^b f(x)dx \approx h[f(m_1) + f(m_2) + \cdots + f(m_{n-1}) + f(m_n)]$$

*Email id: mathsbeauty@gmail.com

where $m_i = \frac{x_{i-1}+x_i}{2}$ for $i = 1, 2, \ldots, n$.

- **The Trapezoidal rule:**

$$\int_a^b f(x)dx \approx \frac{h}{2}[f(x_0) + 2\,f(x_1) + 2\,f(x_2) + \cdots + 2\,f(x_{n-1}) + f(x_n)]$$

- **The Simpson's one-third rule:**

$$\int_a^b f(x)dx \approx \frac{h}{3}[f(x_0) + 4\,f(x_1) + 2\,f(x_2) + 4\,f(x_3) + 2\,f(x_4) + \cdots + 2\,f(x_{n-2}) + 4\,f(x_{n-1}) + f(x_n)]$$

  The number of subintervals n should be an even number.

- **The Simpson's three-eighth rule:**

$$\begin{aligned}\int_a^b f(x)dx \approx \frac{3h}{8}[&f(x_0) + 3\,f(x_1) + 3\,f(x_2) + 2\,f(x_3) + 3\,f(x_4) + 3\,f(x_5)\\ &+ 2\,f(x_6) + \cdots + 2\,f(x_{n-3}) + 3\,f(x_{n-2}) + 3\,f(x_{n-1}) + f(x_n)]\end{aligned}$$

  The number of subintervals n should be a multiple of 3.

4 Commands in the luanumint package

Table 1 lists commands in the `luanumint` package with their description.

Command	Description
`\luaMidpt`	Evaluates the integral using the Mid-point rule.
`\luaMidptSteps`	Evaluates the integral using the Mid-point rule and provides steps involved in calculations.
`\luaTrapz`	Evaluates the integral using the Trapezoidal rule.
`\luaTrapzSteps`	Evaluates the integral using the Trapezoidal rule and provides steps involved in calculations.
`\luaSimpsonOneThird`	Evaluates the integral using the Simpson's one-third rule.
`\luaSimpsonOneThirdSteps`	Evaluates the integral using the Simpson's rule and provides steps involved in calculations.
`\luaSimpsonThreeEighth`	Evaluates the integral using the Simpson's three-eighth rule.
`\luaSimpsonThreeEighthSteps`	Evaluates the integral using the Simpson's three-eighth rule and provides steps involved in calculations.

Table 1: Commands in the luanumint package

Each command has one compulsory argument: a function which is to be integrated. Also, each command has a common set of optional parameters. Table 2 lists these optional parameters with their description.

Optional Parameter	Description
`a`	Specifies the left end-point of the interval over which the integration is performed. The default value of `a` is 0.
`b`	Specifies the right-point of the interval over which the integration is performed. The default value of `b` is 1.
`n`	Specifies the number of subintervals used in calculating sums using different rules of numerical integration. The default value of `n` is 6.
`func`	Specifies the label for a function which is to be integrated. The default value of `func` is f.
`trun`	Specifies the number of decimal places to which the computations are to be truncated. The default value of `trun` is 4.

Table 2: Optional parameters in commands of the package

5 Illustrations of commands in the luanumint package

Table 3 illustrates some commands in the `luanumint` package.

LaTeX input	Result
`$\int_{1}^{3}\sqrt{9+sin(x)}` `dx=\luaMidpt[a=1,b=3,n=4]{sqrt(9+sin(x))}$`	$\int_1^3 \sqrt{9+\sin(x)}dx = 6.2519$
`$\int_{1}^{2}\cos(x)dx=\luaTrapz[a=1,b=2,n=5]{cos(x)}$`	$\int_1^2 \cos(x)dx = 0.0676$
`$\int_{0}^{1}\sin(x+1)` `dx=\luaSimpsonOneThird[a=0,b=1,n=4,trun=6]{sin(x+1)}$`	$\int_0^1 \sin(x+1)dx = 0.95647$
`$\int_{0}^{3}\sin(x)` `dx=\luaSimpsonThreeEighth[a=0,b=3,trun=6]{sin(x)}$`	$\int_0^3 \sin(x)dx = 7.337166$

Table 3: Illustrations of commands in the luanumint package

Listing 1 generates the output shown in Table 4.

Listing 1: The luaMidptSteps command

```
\begin{dmath*}
\int_{1}^{3}\sqrt{9+sin(x)} dx
\luaMidptSteps[a=1,b=3,n=4,trun=6]{sqrt(9+sin(x))}
\end{dmath*}
```

$$\begin{aligned}\int_{1}^{3} \sqrt{9+sin(x)}dx &= 0.5\,[f(1.25)+f(1.75)+f(2.25)+f(2.75)]\\ &= 0.5\,(3.154201+3.159745+3.126991+3.06295)\\ &= 6.251943\end{aligned}$$

Table 4: The luaMidptSteps command

Listing 2 generates the output shown in Table 5.

Listing 2: The luaTrapzSteps command

```
\begin{dmath*}
\int_{0}^{1}\sqrt{1+\sin^4(x)}dx
\luaTrapzSteps[a=0,b=1,n=5,trun=6]{sqrt(1+(sin(x))^4)}
\end{dmath*}
```

$$\begin{aligned}\int_{0}^{1} \sqrt{1+\sin^4(x)}dx &= 0.1\,[f(0)+2f(0.2)+2f(0.4)+2f(0.6)+2f(0.8)+f(1.0)]\\ &= 0.1\,(1.0+2.001557+2.022866+2.099187+2.249278+1.225303)\\ &= 1.059819\end{aligned}$$

Table 5: The luaTrapzSteps command

Listing 3 generates the output shown in Table 6.

Listing 3: The luaSimpsonOnethirdSteps command

```
\begin{dmath*}
\int_{0}^{1}\sin(x+1) dx
\luaSimpsonOneThirdSteps[a=0,b=1,n=4,trun=6]{sin(x+1)}
\end{dmath*}
```

$$\begin{aligned}\int_{0}^{1} \sin(x+1)dx &= 0.083333\,[f(0)+4f(0.25)+2f(0.5)+4f(0.75)+f(1.0)]\\ &= 0.083333\,(0.841471+3.795938+1.99499+3.935944+0.909297)\\ &= 0.956466\end{aligned}$$

Table 6: The luaSimpsonOnethirdSteps command

Listing 4 generates the output shown in Table 7.

Listing 4: The luaSimpsonThreeEightSteps command

```
\begin{dmath*}
\int_{0}^{3}\sqrt{x^3+1} dx
\luaSimpsonThreeEighthSteps[a=0,b=3,n=9,trun=6,func=h]{sqrt(x^3+1)}
\end{dmath*}
```

$$\begin{aligned}\int_0^3 \sqrt{x^3+1}dx &= 0.125\,[h(0) + 3h(0.333333) + 3h(0.666667) + 2h(1.0) + 3h(1.333333) + 3h(1.666667) \\ &\qquad + 2h(2.0) + 3h(2.333333) + 3h(2.666667) + h(3.0)] \\ &= 0.125\,(1.0 + 3.05505 + 3.41565 + 2.828427 + 5.507571 + 7.118052 + 6.0 + 11.105554 \\ &\qquad + 13.40398 + 5.291503) \\ &= 7.340723\end{aligned}$$

Table 7: The luaSimpsonThreeEightSteps command

6 Known issues and limitations

The package uses no external library supporting arbitrary precision arithmetic. The `luanumint` package can handle big and small numbers within the range of Lua that it supports. Lua does not have an inbuilt function to round off numbers to the desired number of decimal places. The package uses some user-defined function to round off calculations in commands that facilitate step-by-step calculations. The round-off error may occur while dealing with big and small numbers, and it may cause slight deviations in the results.