# A very simple introduction to using Python in Latex

Nasser M. Abbasi

September 6, 2023 Compiled on September 6, 2023 at 7:55pm

## Contents



This note describes all the steps to use Python inside Latex. The directions are based on using Linux, since this is the system I tried this on.

Using texlive 2020, this package (pythontex) is already there. There is not additional installation needed. Once you install texlive itself, then you have this package.

## 1 Example using pyconsole

Here is the most simple tex file that uses python

```
\documentclass[11pt]{article}%
\usepackage{pythontex}
\begin{document}
This document uses Python
\end{document}
```

Now comes the tricky part. To compile it, we have to do the following commands

```
pdflatex foo.tex
/usr/local/texlive/2022/texmf-dist/scripts/pythontex/pythontex.py  foo.tex
```

Which prints on the screen this

```
>/usr/local/texlive/2022/texmf-dist/scripts/pythontex/pythontex.py  python_file.tex
```

```
This is PythonTeX 0.17

--------------------------------------------------
PythonTeX:  python_file - 0 error(s), 0 warning(s)
```

If you get an error `ModuleNotFoundError: No module named 'pygments'` then try the following to install pygments: `pip install pygments` first.

Now need to compile the latex file on more time

```
pdflatex foo.tex
```

We have to run the above 3 commands each time. On my installation of texlive 2022, the script `pythontex.py` was not already in the path, so instead of keep typing the above long command each time, I added an alias to my bashrc

```
vi $HOME/.bashrc #add this line below to the file
alias PYX=/usr/local/texlive/2022/texmf-dist/scripts/pythontex/pythontex.py

#after exiting vi, typed
source $HOME/.bashrc
```

So now I can write (less typing) the following

```
pdflatex foo.tex
PYX  foo.tex
pdflatex foo.tex
```

Or you can add it to the path. You must have python itself installed. Version 2.7 or better, else none of the above will work.

In the above example, we did not do anything interesting. Let us now make a python variable, raise it to the power 2, and show the result in Latex

```
\documentclass[11pt]{article}%
\usepackage{pythontex}
\usepackage{nopageno}
\begin{document}
\begin{pyconsole}
x = 987.27
x = x**2
\end{pyconsole}

The variable is $x=\pycon{x}$
\end{document}
```

Which produces the following PDF file when compiled

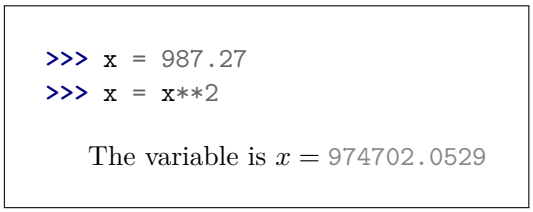



Figure 1: PDF file showing Python computation result inside it

# 2 Example using pycode

In this example we will call Python to do symbolic computation, which is integrating a function, then obtain the latex back of the result. For this **sympy** is used inside Python.

Create Latex file like this

```
\documentclass[11pt]{article}%
\usepackage{pythontex}
\usepackage{nopageno}
\begin{document}
\begin{pycode}
from sympy import *
x=symbols('x')
result = latex(integrate("(1+x)**(1/2)",x))
\end{pycode}

The result of integrating $\int \sqrt{ 1+x } \, dx$ is given by $\py{result}$

\end{document}
```

Now compile using the same commands as first example above

```
pdflatex foo.tex
/usr/local/texlive/2022/texmf-dist/scripts/pythontex/pythontex.py  foo.tex
pdflatex foo.tex
```

Which produces the following PDF file when compiled

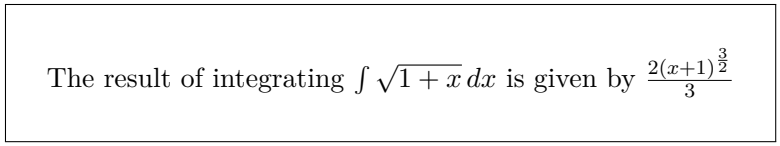



Figure 2: PDF file showing Python computation result inside it

# 3 Calling python passing it argument to process and getting the Latex back

In this example, we make a small function inside `pycode` which takes one argument which is string that represents the integrand to integrate and second argument which is the integration variable.

Create Latex file like this

```
\documentclass[11pt]{article}%
\usepackage{amsmath}
\usepackage{pythontex}
\usepackage{nopageno}
\begin{document}


\begin{pycode}

from sympy import *

def int(theIntegrand,var):
    var  = symbols(var)
    anti = integrate(theIntegrand,var)
    return latex(anti)

\end{pycode}

The result of integrating $\int \frac{1}{\sqrt{ 1+x }} \, dx$ is given by $\py{int("1/(1+x)**

Here is some list of integrations to do

\begin{align*}
\int \frac{1}{\sqrt{ 1+x }} \, dx &=  \py{int("1/(1+x)**(1/2)","x")} \\
\int \sin x \, dx &=  \py{int("sin(x)","x")} \\
\int x \sin x \, dx &=  \py{int("x*sin(x)","x")} \\
\int x^2 \sin x \, dx &=  \py{int("x**2 * sin(x)","x")} \\
\int x e^{2 x} \, dx &=  \py{int("x*exp(2*x)","x")} \\
\int \frac{1}{1+u} \, du &=  \py{int("1/(1+u)","u")} \\
\end{align*}

\end{document}
```

Compiling again using the same commands as the above examples

```
pdflatex foo.tex
/usr/local/texlive/2022/texmf-dist/scripts/pythontex/pythontex.py  foo.tex
pdflatex foo.tex
```

Produces the following PDF file

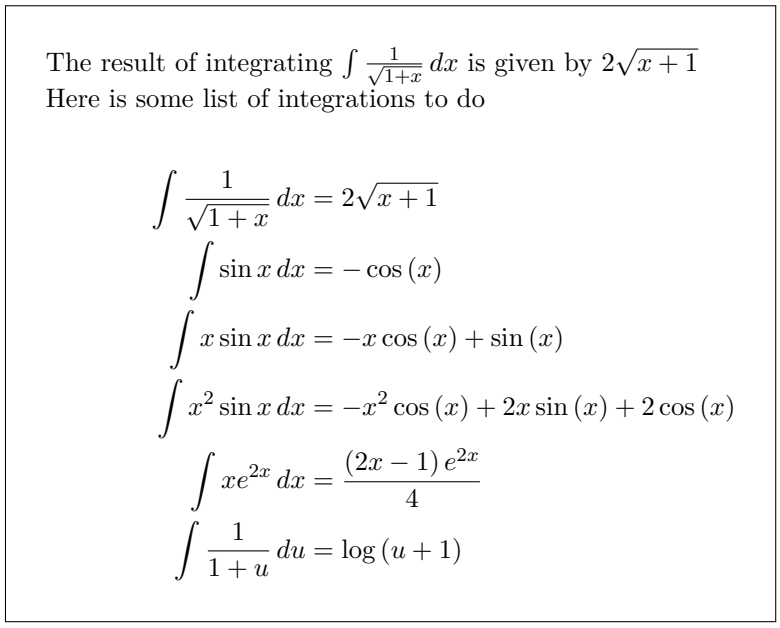



Figure 3: PDF file showing Python computation result inside it

# 4 Calling python from latex to solve differential equations and show its solution

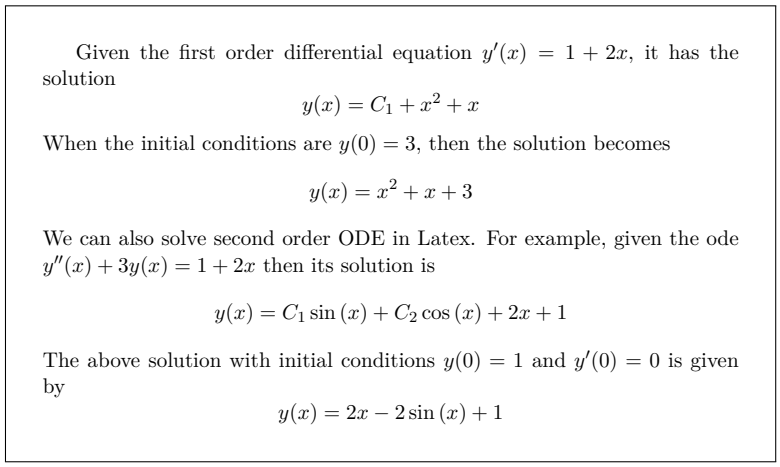
In this example, we make a small function inside `pycode` which takes and ode to solve, using different initial conditions.

Create Latex file like this

```
\documentclass[11pt]{article}%
\usepackage{amsmath}
\usepackage{pythontex}
\usepackage{nopageno}
\begin{document}


\begin{pycode}

from sympy import *

def solveODE(ode,y,x):
    x = Symbol(x)
    y = Function(y)
    lhs,rhs = ode.split('=')
    ode = Eq(S(lhs),S(rhs))
```

```
    sol = dsolve(ode,y(x))
    return latex(sol)

def solveODEwithIC(ode,y,x,ic):
    x   = Symbol(x)
    y   = Function(y)
    lhs,rhs = ode.split('=')
    ode = Eq(S(lhs),S(rhs))
    sol = dsolve(ode,y(x),ics= S(ic) )
    return latex(sol)


\end{pycode}

Given the first order differential equation $y'(x)=1+2 x$, it has the solution
\[
   \py{solveODE("Derivative(y(x),x)=1+2*x","y","x")}
\]
%
When the initial conditions are $y(0)=3$, then the solution becomes
\[
    \py{solveODEwithIC("Derivative(y(x),x)=1+2*x","y","x","{y(0):3}")}
\]
We can also solve second order ODE in Latex. For example, given the ode $y''(x)+3 y(x)=1+2 x$
then its solution is
\[
    \py{solveODE("Derivative(y(x),x,x)+y(x)=1+2*x","y","x")}
\]
The above solution with initial conditions $y(0)=1$ and $y'(0)=0$ is given by
\[
    \py{solveODEwithIC("Derivative(y(x),x,x)+y(x)=1+2*x","y","x","{y(0):1,y(x).diff(x).subs(x
\]


\end{document}
```

Compiling again using the same commands as the above examples produces the following PDF file



Given the first order differential equation $y'(x) = 1 + 2x$, it has the solution

$$y(x) = C_1 + x^2 + x$$

When the initial conditions are $y(0) = 3$, then the solution becomes

$$y(x) = x^2 + x + 3$$

We can also solve second order ODE in Latex. For example, given the ode $y''(x) + 3y(x) = 1 + 2x$ then its solution is

$$y(x) = C_1 \sin(x) + C_2 \cos(x) + 2x + 1$$

The above solution with initial conditions $y(0) = 1$ and $y'(0) = 0$ is given by

$$y(x) = 2x - 2\sin(x) + 1$$

Figure 4: PDF file showing Python computation result inside it

References:


1. Pythontex main documenation http://www.ctan.org/pkg/pythontex
2. pythontex quickstart PDF
3. Pythontex GitHub
4. A Gentle Introduction to PythonTeX