Creating Extended Design Structure Matrix Diagrams

Computational Design Laboratory

Department of Aerospace Engineering
Iowa State University

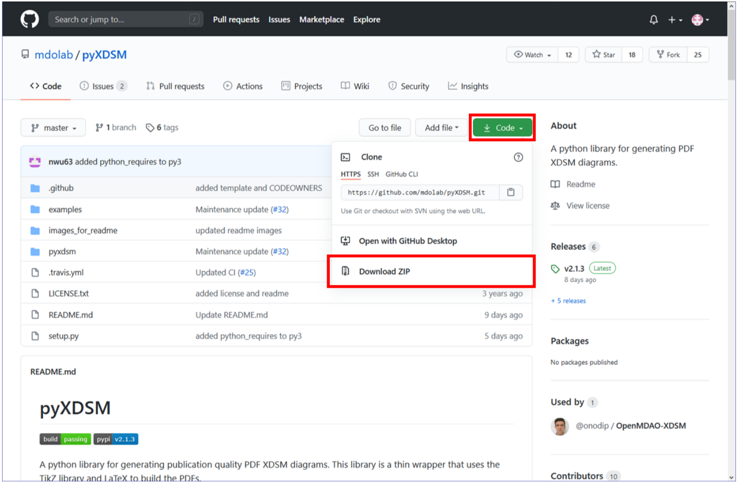
September 21, 2020

Outline

- Installing pyXDSM
- Installing LaTeX
- XDSM example 1: Multidisciplinary feasible
- XDSM example 2: The Sellar problem
- Other examples
- Further reading

Installing pyXDSM

- Download pyXDSM codes from Github: ▸ Link



nwu63 added python_requires to py3

.github	added template and CODEOWNERS	
examples	Maintenance update (#32)	
images_for_readme	updated readme images	
pyxdsm	Maintenance update (#32)	
.travis.yml	Updated CI (#25)	
LICENSE.txt	added license and readme	3 years ago
README.md	Update README.md	9 days ago
setup.py	added python_requires to py3	5 days ago

```
https://github.com/mdolab/pyXDSM.git
```

Use Git or checkout with SVN using the web URL.

About

A python library for generating PDF XDSM diagrams.

No packages published

@onodip / OpenMDAO-XDSM

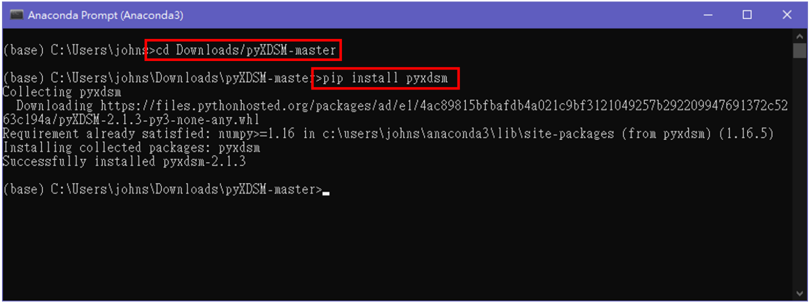
README.md

pyXDSM

A python library for generating publication quality PDF XDSM diagrams. This library is a thin wrapper that uses the TikZ library and LaTeX to build the PDFs.

Installing pyXDSM

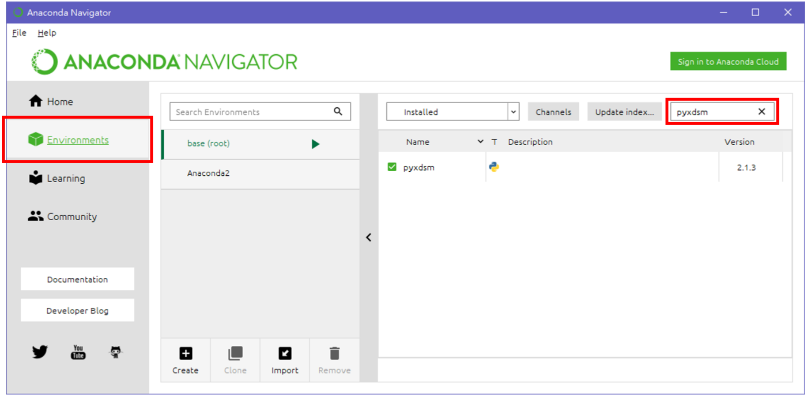
- Unzip pyXDSM from where it saved
- Open Anaconda Prompt
- Use cd command to the pyXDSM directory
- Install pyXDSM with the command: pip install pyxdsm



```
(base) C:\Users\johns>cd Downloads/pyXDSM-master

(base) C:\Users\johns\Downloads\pyXDSM-master>pip install pyxdsm
Collecting pyxdsm
  Downloading https://files.pythonhosted.org/packages/ad/e1/4ac89815bfbafdb4a021c9bf3121049257b292209947691372c52
63c194a/pyXDSM-2.1.3-py3-none-any.whl
Requirement already satisfied: numpy>=1.16 in c:\users\johns\anaconda3\lib\site-packages (from pyxdsm) (1.16.5)
Installing collected packages: pyxdsm
Successfully installed pyxdsm-2.1.3

(base) C:\Users\johns\Downloads\pyXDSM-master>_
```

Installing pyXDSM

- Check for successful installation
- Open anaconda-navigator, and search pyxdsm in Environments

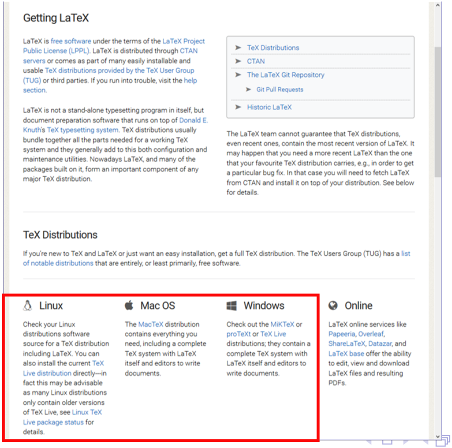


pyxdsm

Name	T	Description	Version
☑ pyxdsm			2.1.3

Installing LaTeX

- Download LaTeX installer from here: ▸ Link
- Follow the instructions to install LaTeX
- For Windows MiKTex is preferred



Getting LaTeX

LaTeX is free software under the terms of the LaTeX Project Public License (LPPL). LaTeX is distributed through CTAN servers or comes as part of many easily installable and usable TeX distributions provided by the TeX User Group (TUG) or third parties. If you run into trouble, visit the help section.

LaTeX is not a stand-alone typesetting program in itself, but document preparation software that runs on top of Donald E. Knuth's TeX typesetting system. TeX distributions usually bundle together all the parts needed for a working TeX system and they generally add to this both configuration and maintenance utilities. Nowadays LaTeX, and many of the packages built on it, form an important component of any major TeX distribution.

- TeX Distributions
- CTAN
- The LaTeX Git Repository
  - Git Pull Requests
- Historic LaTeX

The LaTeX team cannot guarantee that TeX distributions, even recent ones, contain the most recent version of LaTeX. It may happen that you need a more recent LaTeX than the one that your favourite TeX distribution carries, e.g., in order to get a particular bug fix. In that case you will need to fetch LaTeX from CTAN and install it on top of your distribution. See below for details.

TeX Distributions

If you're new to TeX and LaTeX or just want an easy installation, get a full TeX distribution. The TeX Users Group (TUG) has a list of notable distributions that are entirely, or least primarily, free software.

Linux

Check your Linux distributions software source for a TeX distribution including LaTeX. You can also install the current TeX Live distribution directly—in fact this may be advisable as many Linux distributions only contain older versions of TeX Live, see Linux TeX Live package status for details.

Mac OS

The MacTeX distribution contains everything you need, including a complete TeX system with LaTeX itself and editors to write documents.

Windows

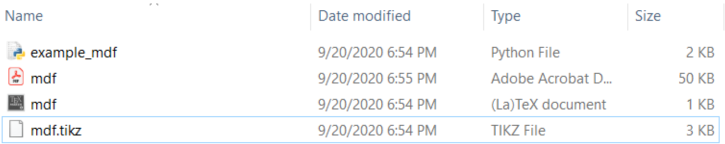
Check out the MiKTeX or proTeXt or TeX Live distributions; they contain a complete TeX system with LaTeX itself and editors to write documents.

Online

LaTeX online services like Papeeria, Overleaf, ShareLaTeX, Datazar, and LaTeX base offer the ability to edit, view and download LaTeX files and resulting PDFs.

XDSM example 1: Multidisciplinary feasible

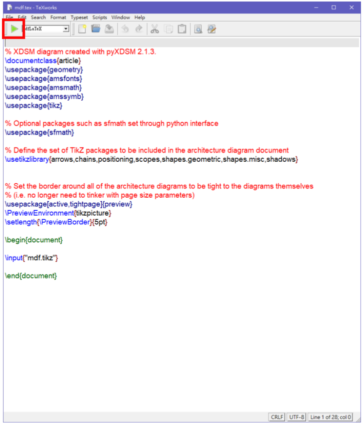
- Download XDSM examples from Github: ▸ Link
- Open example_mdf.py in Spyder and run the file
- The run will generate three files mdf.tex, mdf.tikz and mdf.pdf at code location
- If PDF file is not generated, see next slide



Name	Date modified	Type	Size
example_mdf	9/20/2020 6:54 PM	Python File	2 KB
mdf	9/20/2020 6:55 PM	Adobe Acrobat D...	50 KB
mdf	9/20/2020 6:54 PM	(La)TeX document	1 KB
mdf.tikz	9/20/2020 6:54 PM	TIKZ File	3 KB

XDSM example 1: Multidisciplinary feasible

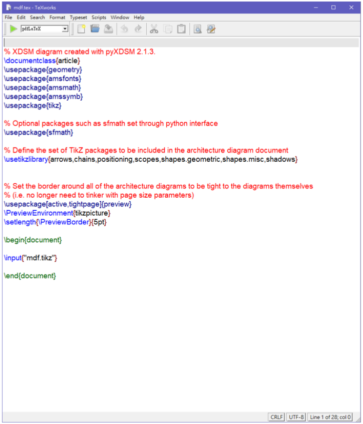
- If PDF file is not generated
- Open mdf.tex with the installed LaTeX editor and hit the run button



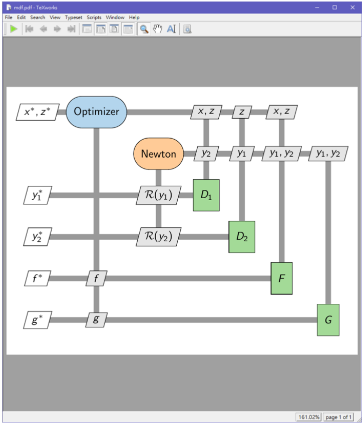
```
% XDSM diagram created with pyXDSM 2.1.3.
\documentclass{article}
\usepackage{geometry}
\usepackage{amsfonts}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{tikz}

% Optional packages such as sfmath set through python interface
\usepackage{sfmath}

% Define the set of TikZ packages to be included in the architecture diagram document
\usetikzlibrary{arrows,chains,positioning,scopes,shapes.geometric,shapes.misc,shadows}


% Set the border around all of the architecture diagrams to be tight to the diagrams themselves
% (i.e. no longer need to tinker with page size parameters)
\usepackage[active,tightpage]{preview}
\PreviewEnvironment{tikzpicture}
\setlength{\PreviewBorder}{5pt}

\begin{document}

\input{"mdf.tikz"}

\end{document}
```

XDSM example 1: Multidisciplinary feasible

- The preview of XDSM diagram will be generated and saved as mdf.pdf in the save folder



```

% XDSM diagram created with pyXDSM 2.1.3.
\documentclass{article}
\usepackage{geometry}
\usepackage{amsfonts}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{tikz}

% Optional packages such as sfmath set through python interface
\usepackage{sfmath}

% Define the set of TikZ packages to be included in the architecture diagram document
\usetikzlibrary{arrows,chains,positioning,scopes,shapes.geometric,shapes.misc,shadows}


% Set the border around all of the architecture diagrams to be tight to the diagrams themselves
% (i.e. no longer need to tinker with page size parameters)
\usepackage[active,tightpage]{preview}
\PreviewEnvironment{tikzpicture}
\setlength{\PreviewBorder}{5pt}

\begin{document}

\input{"mdf.tikz"}

\end{document}
```



XDSM example 2: The Sellar problem

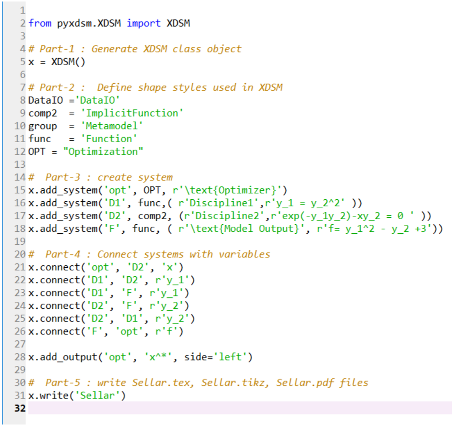
- The Sellar problem with two disciplines and one scalar input

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & f(\mathbf{x}) = y_1^2 - y_2 + 3 \\
\text{Discipline 1 :} \quad & y_1 = y_2^2 \\
\text{Discipline 1 :} \quad & exp(-y_1 y_2) - \mathbf{x} y_2
\end{aligned}
$$

- Generate XDSM diagram for above problem
- This code can also be found here : ▸ Link

XDSM example 2: The Sellar problem

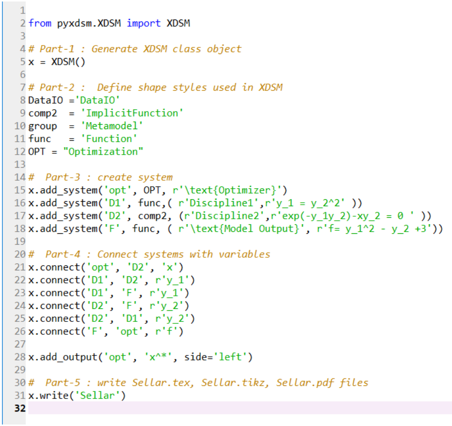
- Download 'Sellar_1.py'
- Part 1: Generates XDSM class object
- Part 2: Defines the shapes styles used in XDSM figure



```

from pyxdsm.XDSM import XDSM

# Part-1 : Generate XDSM class object
x = XDSM()

# Part-2 :  Define shape styles used in XDSM
DataIO ='DataIO'
comp2  = 'ImplicitFunction'
group  = 'Metamodel'
func   = 'Function'
OPT = "Optimization"

#  Part-3 : create system
x.add_system('opt', OPT, r'\text{Optimizer}')
x.add_system('D1', func,( r'Discipline1',r'y_1 = y_2^2' ))
x.add_system('D2', comp2, (r'Discipline2',r'exp(-y_1y_2)-xy_2 = 0 ' ))
x.add_system('F', func, ( r'\text{Model Output}', r'f= y_1^2 - y_2 +3'))

#  Part-4 : Connect systems with variables
x.connect('opt', 'D2', 'x')
x.connect('D1', 'D2', r'y_1')
x.connect('D1', 'F', r'y_1')
x.connect('D2', 'F', r'y_2')
x.connect('D2', 'D1', r'y_2')
x.connect('F', 'opt', r'f')

x.add_output('opt', 'x^*', side='left')

#  Part-5 : write Sellar.tex, Sellar.tikz, Sellar.pdf files
x.write('Sellar')

```

XDSM example 2: The Sellar problem

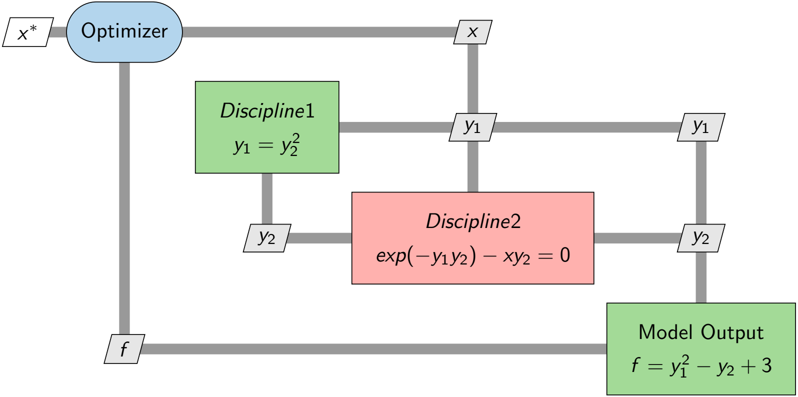
- Part 3: Add different systems to object x
- Part 4: Connects the created systems with each other
- Part 5: Writes .tex, .tikz and .pdf files at code location



```python

from pyxdsm.XDSM import XDSM

# Part-1 : Generate XDSM class object
x = XDSM()

# Part-2 :  Define shape styles used in XDSM
DataIO ='DataIO'
comp2  = 'ImplicitFunction'
group  = 'Metamodel'
func   = 'Function'
OPT = "Optimization"

#  Part-3 : create system
x.add_system('opt', OPT, r'\text{Optimizer}')
x.add_system('D1', func,( r'Discipline1',r'y_1 = y_2^2' ))
x.add_system('D2', comp2, (r'Discipline2',r'exp(-y_1y_2)-xy_2 = 0 ' ))
x.add_system('F', func, ( r'\text{Model Output}', r'f= y_1^2 - y_2 +3'))

#  Part-4 : Connect systems with variables
x.connect('opt', 'D2', 'x')
x.connect('D1', 'D2', r'y_1')
x.connect('D1', 'F', r'y_1')
x.connect('D2', 'F', r'y_2')
x.connect('D2', 'D1', r'y_2')
x.connect('F', 'opt', r'f')

x.add_output('opt', 'x^*', side='left')

#  Part-5 : write Sellar.tex, Sellar.tikz, Sellar.pdf files
x.write('Sellar')

```

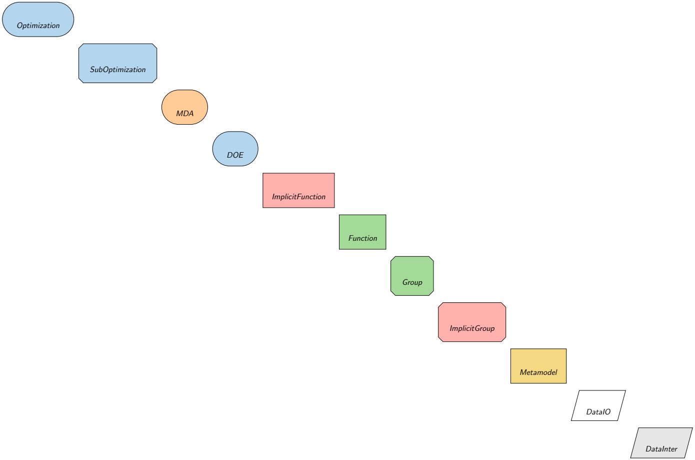
XDSM example 2: The Sellar problem

- Output of Sellar_1.py



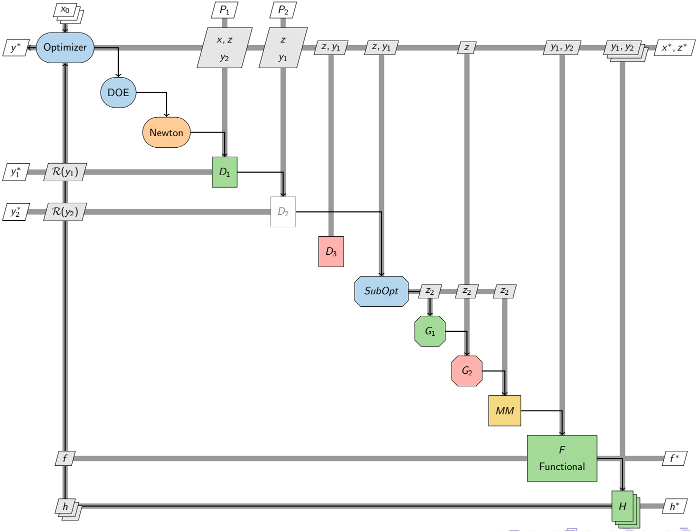
Other examples

- Output of shapes.py ▸ Link
- Shows available shapes in pyxdsm package



Other complex examples

- Output of example_kitchen_sink.py ▸ Link



Further reading

- Link for more details : ▸ Link