



# 仮想環境の構築方法 Larchのインストールの仕方



# 1.はじめに

<https://xraypy.github.io/xraylarch/installation.html#install-lin>

Larchとは、XAFS解析をpythonのコードで行う方法である。  
デフォルトではlarchはインストールされていないため、  
自分でインストールをする必要がある。

上記のサイトに、Larchのインストールの仕方(English.ver)が載っているため、  
参考にしながら各操作を行う。

## 2. 仮想環境の構築の仕方

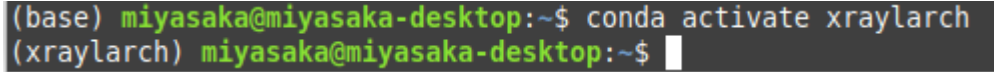
1. Linux上でターミナルを開く
2. **pyenv install -l** を入力し、利用できるversionの一覧を表示する。
3. **pyenv install <version>** で、希望するversionをインストールする。  
(今回は、3.9.7を使用)
4. **pyenv versions** でインストールされているversionを表示させる。  
\*とあるものが、現在利用している環境(version)である。
5. <https://xraypy.github.io/xraylarch/installation.html#install-lin>  
このサイトの"GetLarch.sh"スクリプトを、Text Editorにコピー&ペーストする。  
Downloadsフォルダに作成し、ファイル名は"GetLarch.sh"とする。  
※自動的にダウンロードできないため、注意が必要。
6. **cd Downloads** と入力する。(操作がDownloads上で行われる)
7. **sh GetLarch.sh** と入力し、処理を待つ。

```
(base) miyasaka@miyasaka-desktop:~$ pyenv install -l
Available versions:
  2.1.3
  2.2.3
  2.3.7
  2.4.0
  2.4.1
  2.4.2
  2.4.3
  2.4.4
  2.4.5
  2.4.6
  2.5.0
  2.5.1
  2.5.2
```

```
(base) miyasaka@miyasaka-desktop:~$ pyenv versions
system
* 3.9.7 (set by /home/miyasaka/.pyenv/version)
  anaconda3-2023.03
  miniconda3-4.7.12
```

## 2. 仮想環境の構築の仕方

以下、<https://xraypy.github.io/xraylarch/installation.html#install-lin>の1.3 Installing into an existing Anaconda Python environmentに書いてある操作を行う。

8. **conda activate** と入力する。(base)と表示されたら正しい。
9. **conda update -y conda python pip** と入力する。処理が終わるまで待つ。
10. **conda create -y -name xraylarch python=>3.9.10** と入力する。
11. **conda activate xraylarch**と入力する。  
(xraylarch)と表示されたら正しい。  

12. **conda install -y "numpy=>1.20" "scipy=>1.6" "matplotlib=>3.0" scikit-learn pandas** と入力する。
13. **conda install -y -c conda-forge wxpython pymatgen tomopy pycifrw** と入力する。  
処理が終わるまでに少し時間がかかる場合もある。  
Errorが起こる場合もあるため、その場合は最初からやり直す。
14. **pip install xraylarch** と入力する。

## 2. 仮想環境の構築の仕方

### 15. **larch -m** で最終確認する。

#### 1.3. Installing into an existing Anaconda Python environment

The following procedure is recommended for those who are familiar with [Anaconda Python](#) / [Conda](#) and have already installed it in their system.

##### Note

Some packages that Larch uses are not currently (January 2022) handled by the standard Python package manager [Pip](#). For this reason, we use a [Conda](#) environment and “conda forge” for installing them. These packages include:

- *pymatgen*: needed for handling CIF files and running FEFF calculations.
- *wxpython*: needed for all plotting, graphics and GUI applications.
- *tomopy*: needed for reconstructing X-ray fluorescence tomography.
- *python.app*: needed (from conda-forge) for Anaconda-based Python on MacOS.
- *epicsapps*: applications using the Epics control system.

Most of Larch functionality can be used as a library without these packages installed.

Within a shell:

1. activate your conda environment (called *base* by default) and update it:

```
conda activate
conda update -y conda python pip
```

2. **(optional/expert)** create a dedicated environment for Larch and activate it:

```
conda create -y --name xraylarch python=>3.9.10
conda activate xraylarch
```

3. install main dependencies:

```
conda install -y "numpy=>1.20" "scipy=>1.6" "matplotlib=>3.0" scikit-learn pandas
conda install -y -c conda-forge wxpython pymatgen tomopy pycifrw
```

4. install Larch (latest release):

```
pip install xraylarch
```

5. if anything of the above fails, report it to the [Larch Github Issues](#)

Putting that all together:

```
conda create -y --name xraylarch python=>3.9.10
conda activate xraylarch
conda install -y "numpy=>1.20" "scipy=>1.6" "matplotlib=>3.0" scikit-learn pandas
conda install -y -c conda-forge wxpython pymatgen tomopy pycifrw
pip install xraylarch
larch -m
```

## 2. 仮想環境の構築の仕方

### 15. **larch -m** で最終確認する。

#### 1.3. Installing into an existing Anaconda Python environment

The following procedure is recommended for those who are familiar with [Anaconda Python](#) / [Conda](#) and have already installed it in their system.

##### Note

Some packages that Larch uses are not currently (January 2022) handled by the standard Python package manager [Pip](#). For this reason, we use a [Conda](#) environment and “conda forge” for installing them. These packages include:

- *pymatgen*: needed for handling CIF files and running FEFF calculations.
- *wxpython*: needed for all plotting, graphics and GUI applications.
- *tomopy*: needed for reconstructing X-ray fluorescence tomography.
- *python.app*: needed (from conda-forge) for Anaconda-based Python on MacOS.
- *epicsapps*: applications using the Epics control system.

Most of Larch functionality can be used as a library without these packages installed.

Within a shell:

1. activate your conda environment (called *base* by default) and update it:

```
conda activate
conda update -y conda python pip
```

2. **(optional/expert)** create a dedicated environment for Larch and activate it:

```
conda create -y --name xraylarch python=>3.9.10
conda activate xraylarch
```

3. install main dependencies:

```
conda install -y "numpy=>1.20" "scipy=>1.6" "matplotlib=>3.0" scikit-learn pandas
conda install -y -c conda-forge wxpython pymatgen tomopy pycifrw
```

4. install Larch (latest release):

```
pip install xraylarch
```

5. if anything of the above fails, report it to the [Larch Github Issues](#)

Putting that all together:

```
conda create -y --name xraylarch python=>3.9.10
conda activate xraylarch
conda install -y "numpy=>1.20" "scipy=>1.6" "matplotlib=>3.0" scikit-learn pandas
conda install -y -c conda-forge wxpython pymatgen tomopy pycifrw
pip install xraylarch
larch -m
```

# 3.Jupyter Notebookとの連携方法

16. **pip install notebook** と入力し、仮想環境とJupyter notebookの環境を同期させる。

17. `from larch import Interpreter` をJupyter Notebookに入力し、正しくImportできるか確かめる。

(17. でErrorが出た場合...)

**python** と入力し、

>>>の後ろに、**from larch import Interpreter**と入力する。

正しくImportされていたら、>>>と出力される。→**pip install notebook** を改めて入力。  
Errorとなったら正しくインストールできていないため、最初からやり直す。

※やり直す場合は、xraylarchを削除してから各操作を行う。

```
In [2]: from larch import Interpreter  
        session = Interpreter()
```

```
(base) miyasaka@miyasaka-desktop:~$ python  
Python 3.10.10 | packaged by conda-forge | (main, Mar 24 2023, 20:08:06) [GCC 11  
.3.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> from larch import Interpreter  
>>> 
```



# 4. Larchについて

<https://qiita.com/inashiro/items/f1c10440618cf73a2d81>

は、Larchに関する日本語サイトです。

このサイトが書かれたときとは、Larchのバージョンが異なるので注意が必要です。

## Old.ver

### larch ライブラリのimport <sup>1</sup>

公式ドキュメントだとlarch\_plugins.hoge から必要なメソッドをいちいちimport するようになっているが、面倒かつ煩雑です。ここではlarch\_plugins をいったん丸ごとimport し、そこから必要に応じてメソッドを呼び出していく方法をとります。

また、Larch のメソッドの多くは、larch.Interpreter インスタンスを引数に指定する必要があります。ここでは、session という名前のインスタンスを作ります。

```
import larch_plugins as lp
from larch import Interpreter
session = Interpreter()
```

import larch\_plugins as lp は、動きません。  
(この部分だけ変更されているみたいです)



## New.ver



@inashiro

2020-02-06 20:33 ...

@tatsuya0618

実は、最近のバージョンのLarch ではライブラリの構成が若干変わったので、上記のコードのままでは動きません。  
(修正しようと思っていたのですが、手が回らず...)

さて、具体的にどこが変わったかと言いますと、larch\_plugins は無くなったようです。xafs, io, xray 等のlarch\_plugins に入っていたモジュールは、larch ライブラリ直下に移動しています。

従来、

```
import larch_plugins as lp
dat = lp.io.read_ascii("./K_Cu12.5K_Si111_20101108.txt", labels="energy mu", _larch=session)
```

などとしていたのを、最新のバージョンでは

```
from larch import io
dat = io.read_ascii("./K_Cu12.5K_Si111_20101108.txt", labels="energy mu", _larch=session)
```

とすると動きます。xafs, xray 等についても同様です。

また、from パッケージ import モジュール という書き方についてはPython の機能ですので別途調べて頂くと良いかと思いますが、以下のようにして一度に複数のモジュールを読み込むことが可能です。

```
from larch import io, xafs, xray # io, xafs, xray を同時に読み込む
```

# 4. Larchについて

利用できるモジュールは以下のとおりです。

<https://xraypy.github.io/xraylarch/python.html>

## 8.5. Larch submodules

The *larch* module is broken up into a number of submodules, as described in the table and sections below. To be clear, this list is incomplete and subject to change. The rest of this document should be used for details of the functionality of the various modules and functions.

**Table of the Larch Python Modules** This is an incomplete list of all the Python modules available from *larch*, but should cover those that are most useful to Python programmers.

module name	Description
<i>larch</i>	top-level module, with Interpreter and applications
<i>larch.io</i>	input/output routines
<i>larch.xafs</i>	XAFS analysis and data processing
<i>larch.xray</i>	X-ray properties (with <a href="#">xraydb</a> )
<i>larch.xrf</i>	X-ray fluorescence processing and analysis
<i>larch.xrd</i>	X-ray diffraction processing
<i>larch.xrmap</i>	working with XRF/XRD microprobe maps
<i>larch.fitting</i>	Data fitting (with <a href="#">lmfit</a> )
<i>larch.math</i>	General-purpose mathematical functions
<i>larch.utils</i>	General-purpose utility functions
<i>larch.epics</i>	using with Epics control system (with <a href="#">pyepics</a> )
<i>larch.xmlrpc_server</i>	running a Larch server for other processes
<i>larch.shell</i>	command-line shell
<i>larch.wxlib</i>	Wx-python utilities (including <a href="#">wxmplot</a> plotting)

Note that several of these modules (*xray*, *fitting*, *epics*, the plotting functionality in *wxlib*) are fairly thin wrappers around other python libraries that are generally well-documented and do not need to be used only from with larch. If you find yourself using these submodules, it might be easier to just use the more general-purpose library.