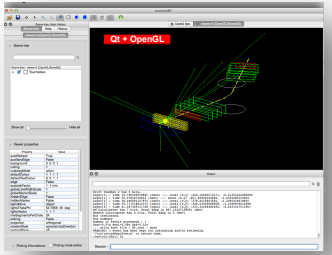
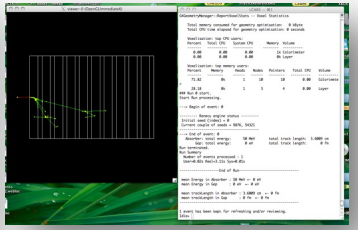
Geant4 Interface

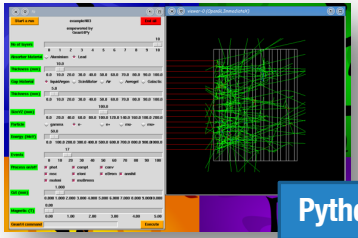
Koichi Murakami (KEK)

28th Geant4 Collaboration Meeting 2023

Geant4 UI & App.







Python App.

Python as software component bus

App Drive w/ G4-UI commands

Geant4 User Applications

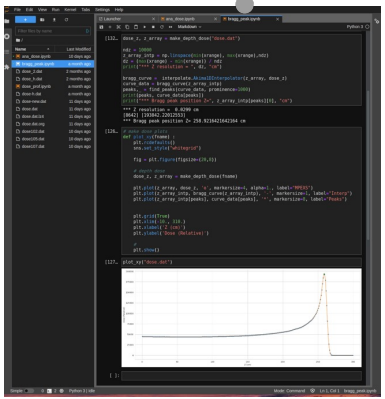
No Interface w/ C++ API

Python Front End

python™

jupyter

>>> import Geant4



UI Session

GUI | CLI | Batch

Qt | *csh, tcsh* | *macro file*

Interface

UI commands /run/beamOn, …

Intercome

Python binding

Geant4 Kernel

Notes on Python Binding

- Python2 : End of life
  - Python2 became End of Life in Apr/2020.
  - Python2 codes will be dropped in the v11 release.
  - Only support Python3 codes

- Boost.python to Pybind11
  - Change C++ binding tool
  - Wrapper approach is very similar to Boost.python (Template base)
  - Header only
  - C++11 (modern C++) support / STL container support

Pybind11

- A binding tool between C++ and Python
- https://github.com/pybind/pybind11
- Header only. Need cmake modules (pybind11)
- Installation:
  - self-install (RH-variants)
  - use apt in Ubuntu
  - use brew in Mac
    - /usr/local (Intel)
    - /opt (Apple Silicon)

How to bind from C++ to Python

```
// ========================================================================
void export_G4RunManager(py::module& m)
{
  m.def("CreateRunManager",     &::CreateRunManager);

  // ------------------------------------------------------------------
  py::class_<G4RunManager>(m, "G4RunManager")
  .def_static("GetRunManager",  &G4RunManager::GetRunManager,
                                py::return_value_policy::reference)
  .def_property("verboseLevel", &G4RunManager::GetVerboseLevel,
                                &G4RunManager::SetVerboseLevel)
  .def("SetVerboseLevel",       &G4RunManager::SetVerboseLevel)
  .def("GetVerboseLevel",       &G4RunManager::GetVerboseLevel)
  .def("GetVersionString",      &G4RunManager::GetVersionString,
                                py::return_value_policy::copy)
  .def("SetNumberOfThreads",    &G4RunManager::SetNumberOfThreads)
  .def("GetNumberOfThreads",    &G4RunManager::GetNumberOfThreads)
   // ---
   .def("Initialize",           &G4RunManager::Initialize)
  .def("BeamOn",                &G4RunManager::BeamOn,
                                py::arg("n_event"),
                                py::arg("macroFile") = nullptr,
                                py::arg("n_select") = -1)
  .def("AbortRun",              &G4RunManager::AbortRun,
                                py::arg("softAbort") = false)
  .def("AbortEvent",            &G4RunManager::AbortEvent)
  //
```

For each class, define function maps in a template meta-programming style.

Same way as boost-python

Notes on Geant4Py (1)

- There are some tips for running Geant4Py
- LD_PRELOAD (Linux)
  - For TLS memory allocation, we have to preload a Geant4 library.
    - # export LD_PRELOAD=libG4run.so (bash/zsh)
    - # setenv LD_PRELOAD libG4run.so (csh/tcsh)
  - In macOS, this is not necessary.
- The multi-threading feature is off as a 1st step.
  - In the current version of Geant4Py, we limit Geant4 in sequential mode forcibly by setting G4FORCE_RUN_MANAGER_TYPE inside __init__.py script.
  - In the future release, multi-threading mode can be activated.

Notes on Geant4Py (2)

- Qt5 conflict
  - We recommend building Geant4 without the Qt5 feature to avoid the conflict.
  - If you use the Anaconda version of Ptyhon3, there might be a conflict between the Qt5 libraries. When Geant4Py detects the conflict, it shows the following warning message.

```
######################################################################
!!! Warning !!!
A non-system python (e.g., Anaconda version of Python) is detected.
If you have a problem with Qt5 library version,
set the environment variables, "G4PY_QT5_PRELOAD = 1"
to preload the system Qt5 library as a temporal solution.
Please consider installing a Geant4 library for Geant4Py
without the Qt feature.
######################################################################
```

  - Geant4Py will preload the system Qt5 when this environment variable is set.
    - # export G4PY_QT5_PRELOAD=1 (bash/zsh)
    - # setenv G4PY_QT5_PRELOAD 1 (csh/tcsh)
  - Currently, we cannot run Geant4Py on the Anaconda version of Python on Mac. Use the system Python and install the additional packages (Jupyter/numpy/matplotlibt/...) using pip.

>>> import geant4

```
# python3
Python 3.8.8 (default, Apr 13 2021, 19:58:26)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import geant4
================================================================
  _____                  __  ____ ___
 / ___/__ ___ _____  / /_/ / // _ \__ __  Geant4-Python Interface
/ (_ / -_) _ `/ _ \/ __/_  _/ ___/ // /  Version: 1100
\___/\__/\_,_/_//_/\__/ /_//_/   \_, /   Date: (31-October-2021)
                                /___/
================================================================

Environment variable "G4FORCE_RUN_MANAGER_TYPE" enabled with value == Serial. Forcing G4RunManage


          ################################
          !!! G4Backtrace is activated !!!
          ################################


*************************************************************
 Geant4 version Name: geant4-10-07-ref-09 [MT]   (31-October-2021)
                      Copyright : Geant4 Collaboration
                     References : NIM A 506 (2003), 250-303
                                : IEEE-TNS 53 (2006), 270-278
                                : NIM A 835 (2016), 186-225
                           WWW : http://geant4.org/
*************************************************************
```

Examples with Jupyter (1)

- There are 3 examples with Jupyter .ipynb files:
- exampleB1
  - This example has the same capability as Geant4 basic example B1.
  - The geometry is implemented in C++ and exported to a Python module, which shows how to export your C++ component to Python. (Thin wrapping approach)
- phantom_dose
  - This example shows a practical application. It contains a complete chain of simulation and analysis processes.
  - We calculate dose distributions in a water phantom for electron and proton beams.
  - Voxel doses are scored with the command-line scoring capability and stored into CSV files.
  - This data is analyzed with Pandas and Matplotlib Python tools. Finally, dose maps and depth dose curves are obtained.

Examples with Jupyter (2)

- emplot
  - This example shows how to retrieve the photon cross-sections and stopping powers of charged particles.
  - It prepares a mockup (geom/pl/primary), then changes the target materials.
  - The EM calculator can calculate a cross-section for each process and stopping powers.
  - For stopping power, the ionization and bremsstrahlung components can be calculated for electrons.
  - The example includes plots by Matplotlib.

phantom_dose: User Classes

In [3]:

```
# set detectot construction
phantom = WaterPhantom()
#phantom.phantomXY = 50.*cm  # phantom size can be changed.
#phantom.phantomZ = 30.*cm
gRunManager.SetUserInitialization(phantom)

# set physics list
physics_list = FTFP_BERT()
gRunManager.SetUserInitialization(physics_list)

# Medical Beam as PGA
medical_beam = MedicalBeam()

# User Action Initialization
class AppBuilder(G4VUserActionInitialization):
    def Build(self):
        # setup PGA
        self.SetUserAction(medical_beam)

        global runaction
        runaction = MyRunAction()
        self.SetUserAction(runaction)

        eventaction = EventCounter()
        eventaction.SetCheckCounter(10000)
        self.SetUserAction(eventaction)
```

```
<<< Geant4 Physics List simulation engine: FTFP_BERT
```

phantom_dose: Main

In [5]:

```
# initialization
app_builder = AppBuilder()
gRunManager.SetUserInitialization(app_builder)

gRunManager.Initialize()
gRunManager.BeamOn(0)
```

```
--- G4CoupledTransportation is used

 hInelastic FTFP_BERT : threshold between BERT and FTFP is over the interval
 for pions :   3 to 6 GeV
 for kaons :   3 to 6 GeV
 for proton :  3 to 6 GeV
 for neutron : 3 to 6 GeV

### Adding tracking cuts for neutron  TimeCut(ns)= 10000  KinEnergyCut(MeV)= 0
======================================================================
======                Electromagnetic Physics Parameters      ========
======================================================================
LPM effect enabled                                    1
Enable creation and use of sampling tables            0
Apply cuts on all EM processes                        0
Use general process                                   0
Enable linear polarisation for gamma                  0
Enable sampling of quantum entanglement               0
X-section factor for integral approach                0.8
Min kinetic energy for tables                         100 eV
Max kinetic energy for tables                         100 TeV
Number of bins per decade of a table                  7
Verbose level                                         1
Verbose level for worker thread                       0
Bremsstrahlung energy threshold above which
  primary e+- is added to the list of secondary       100 TeV
Bremsstrahlung energy threshold above which primary
```

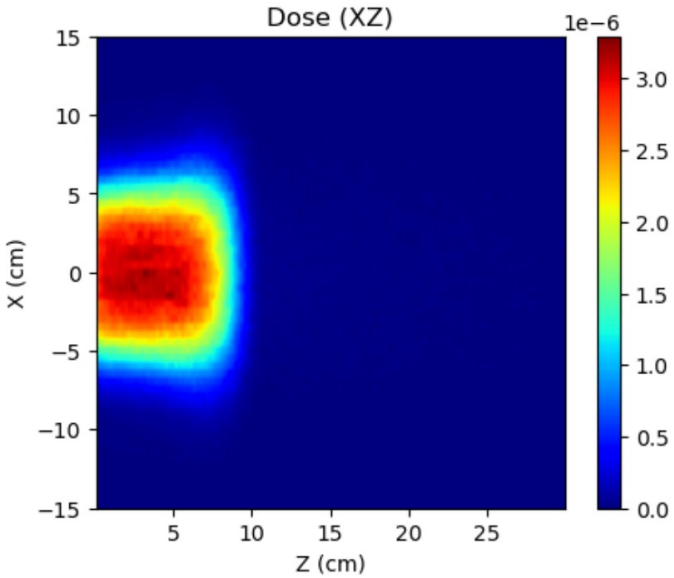
phantom_dose: Electron dose

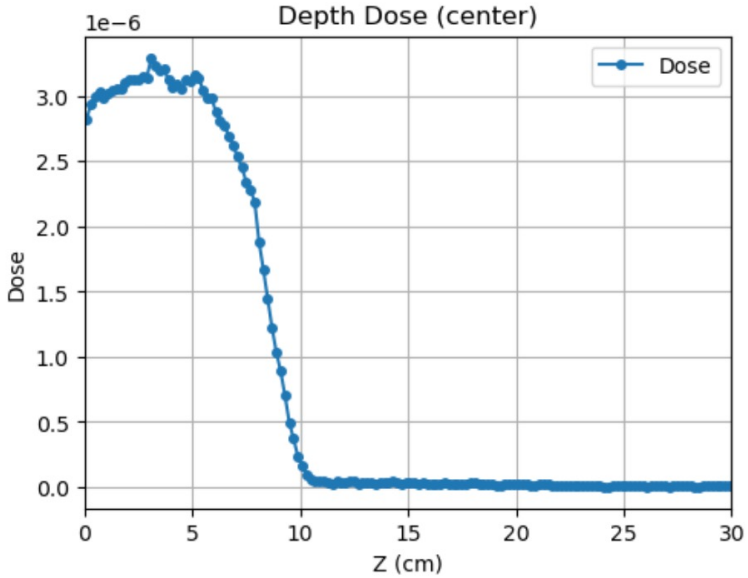
Electron 20 MeV

Command-line scoring

In [16]:
```
# 10M events
plot_dose("dose_e20.csv")
```

```
Sum dose/k =  1.303546326741707e-08
```



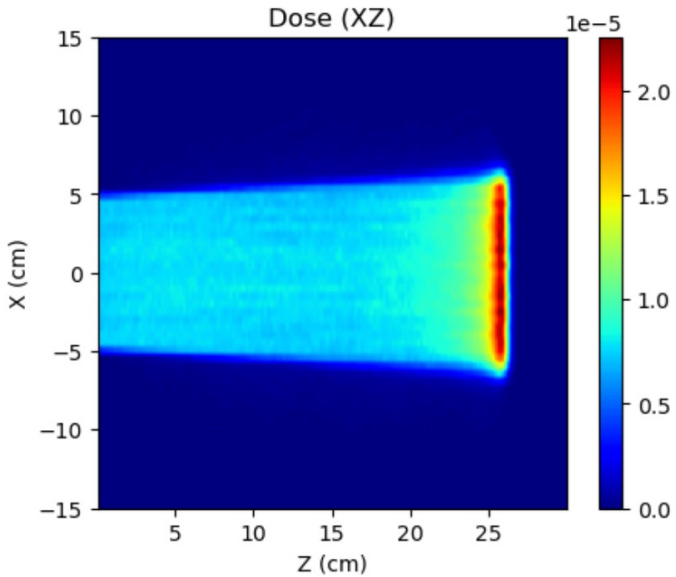


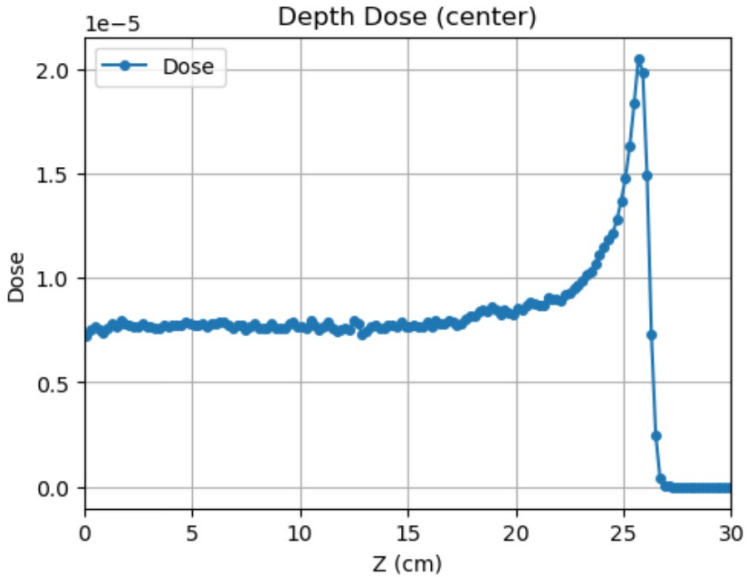
phantom_dose: Proton dose

Proton 200 MeV

In [17]:
```
# 10M events
plot_dose("dose_p200.csv")
```

```
Sum dose/k =  1.1409669100868195e-07
```

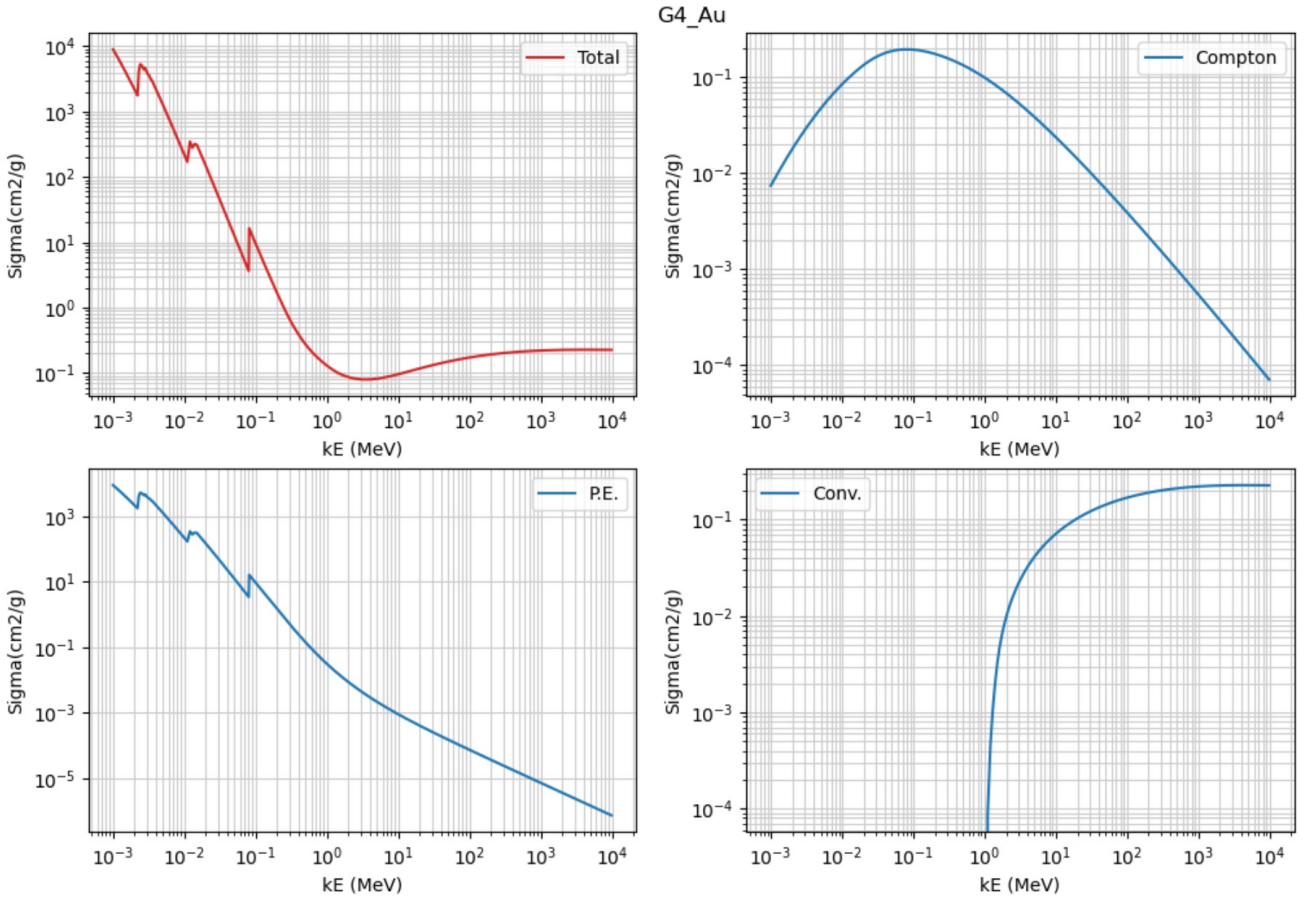




emplot: Photon cross sections

EmCalculator

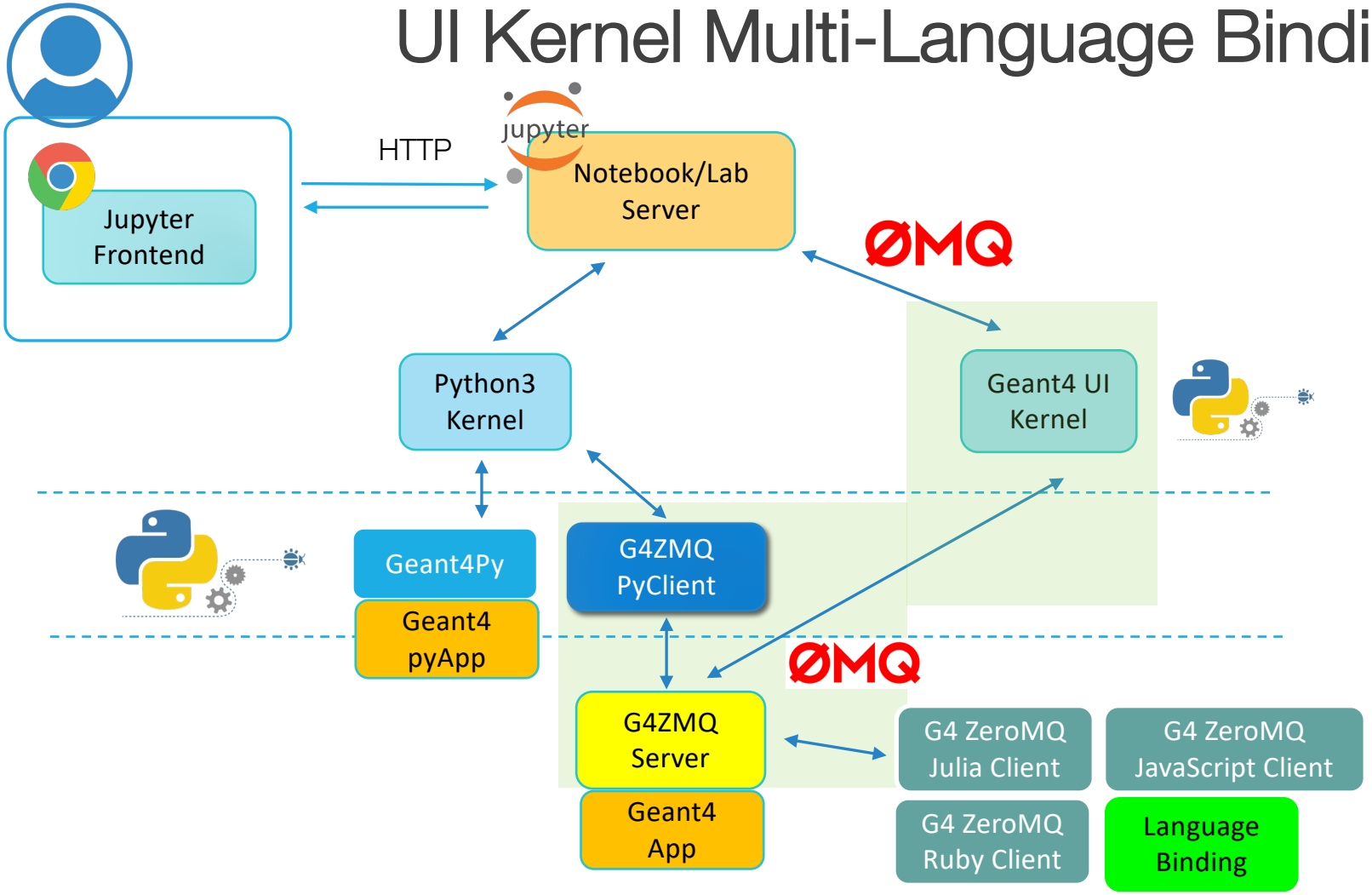
```
In [55]: plot_photon_xsec(xsection_list, title_=material_name)
```



Still some considerations are needed…

- Python Objects
  - Python variables are automatically managed, which means a local variable is automatically deleted on the Python side.
  - This mechanism is different from objects allocated in C++. Some classes are taken care of as never-deleted objects in Geant4Py, but they are still imperfect.
  - If there is a weird behavior (seg. fault), set the Python variable as global.
  - An object of a user-inherited class in Python should be set as global.

UI Kernel Multi-Language Binding



Summary

- Python binding tool was migrated to pybind11.
- There are some tricks for running Geant4 with Python.
  - Memory management issue
  - Library confliction
  - Object management
- Examples of Jupyter sessions
  - Thin wrapping approach
  - Dose calculation and analysis
  - Plots of photon cross-sections and stopping powers
- Geant4 UI : multi-language binding capability with ZeroMQ