Atlas Release Env & Analysis Support

Shuwei Ye

Intro Talk for NPPS group meeting

Sept 2019

Brief about myself

- Prior to joining Atlas, I worked on

  BES, AMS and BaBar (discovered two puzzling particles, Y(4260) and Y(4350))

- In 2007, I joined Atlas group at BNL
  - 50% on USAtlas analysis support (Omega)
  - 50% on Software development & librarian (PAS=>NPPS): LXR & AtlasSetup

Atlas Source Code Cross-Reference

LXR is a general-purpose source code indexer and cross-reference,
providing web-based browsing of source code,
with links to the definition and usage of any identifier.

And Atlas code is mainly in C++ and python.

At BNL, we (Alex and me) provide 2 LXR servers:

- Nightlies releases: https://acode-browser**1**.usatlas.bnl.gov/lxr/source
- Stable releases: https://acode-browser**2**.usatlas.bnl.gov/lxr/source/

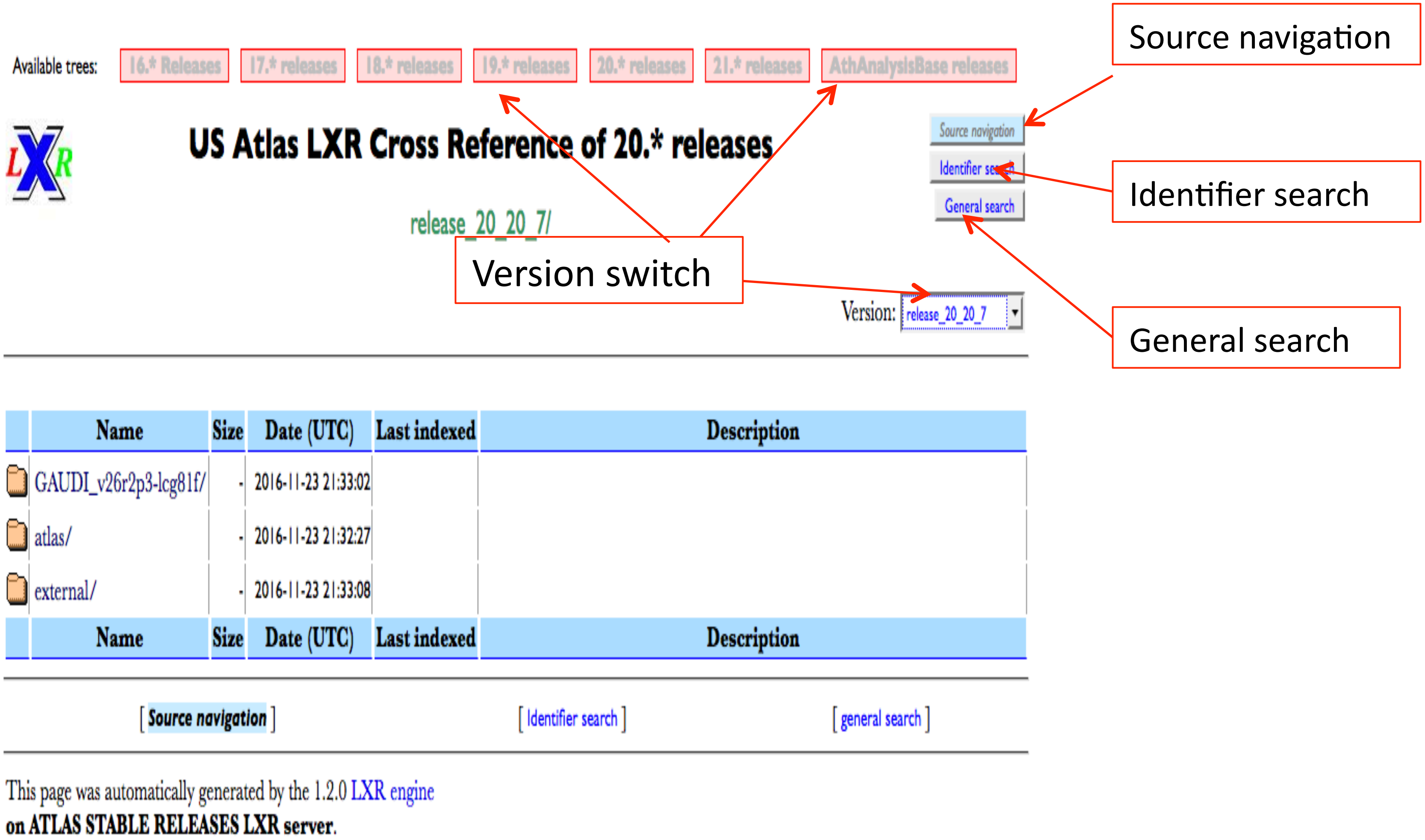
Nightlies releases are indexed everyday, the indexing takes a few hours.

To provide non-interruption service, we introduced symlink switch pending on the indexing job success.

Atlas LXR: Tree-like Version Change

Convenient version change;

Browse source, search for identifier & keyword



Available trees: 16.* Releases 17.* releases 18.* releases 19.* releases 20.* releases 21.* releases AthAnalysisBase releases

US Atlas LXR Cross Reference of 20.* releases

release_20_20_7/

Version: release_20_20_7

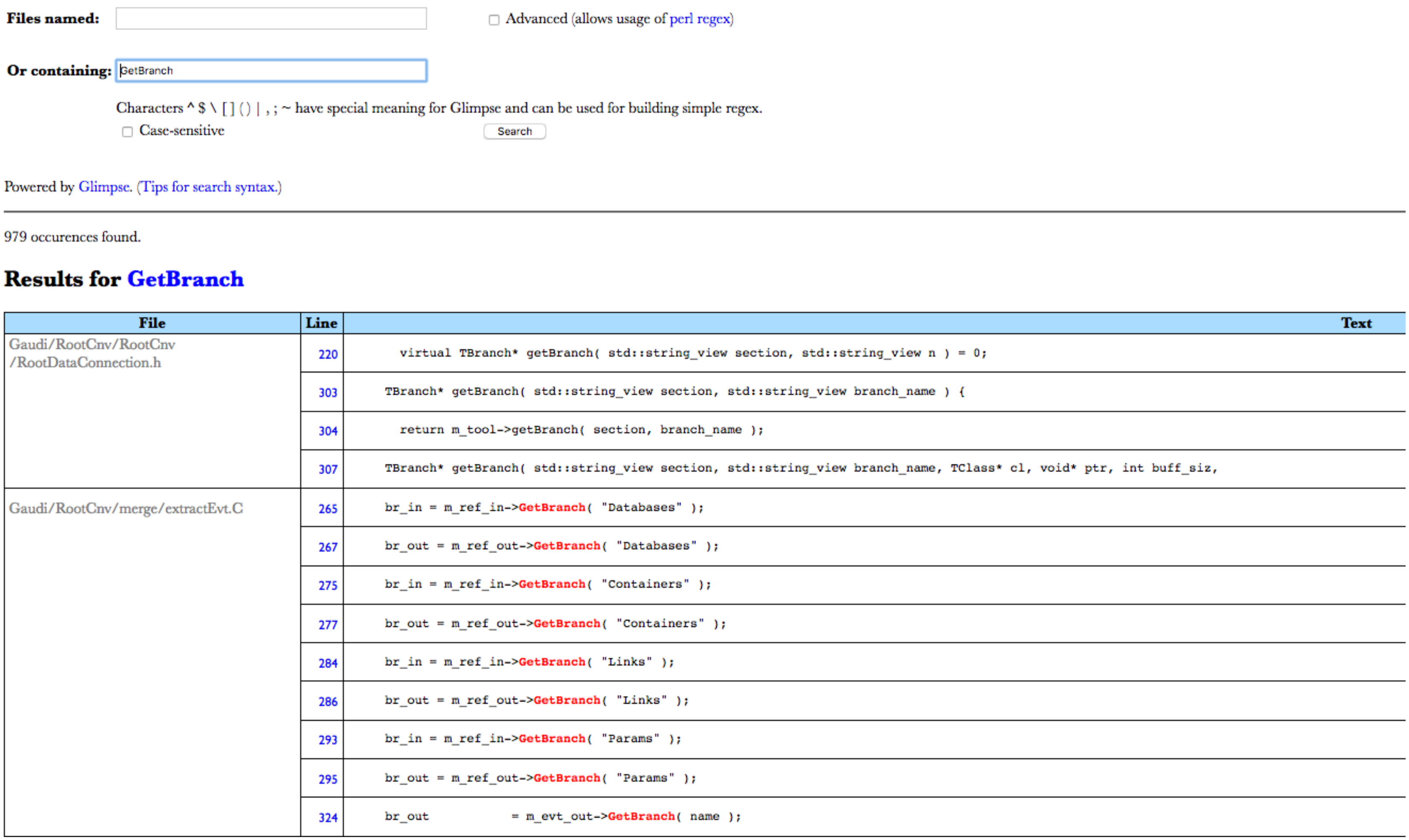
Name	Size	Date (UTC)	Last indexed	Description
GAUDI_v26r2p3-lcg81f/	-	2016-11-23 21:33:02		
atlas/	-	2016-11-23 21:32:27		
external/	-	2016-11-23 21:33:08		
Name	Size	Date (UTC)	Last indexed	Description

[Source navigation] [Identifier search] [general search]

This page was automatically generated by the 1.2.0 LXR engine
on ATLAS STABLE RELEASES LXR server.

Atlas LXR: General Search

General search is conducted by glimpse.



Files named:

□ Advanced (allows usage of perl regex)

Or containing: GetBranch

Characters ^ $ \ [ ] ( ) | , ; ~ have special meaning for Glimpse and can be used for building simple regex.

□ Case-sensitive

Powered by Glimpse. (Tips for search syntax.)

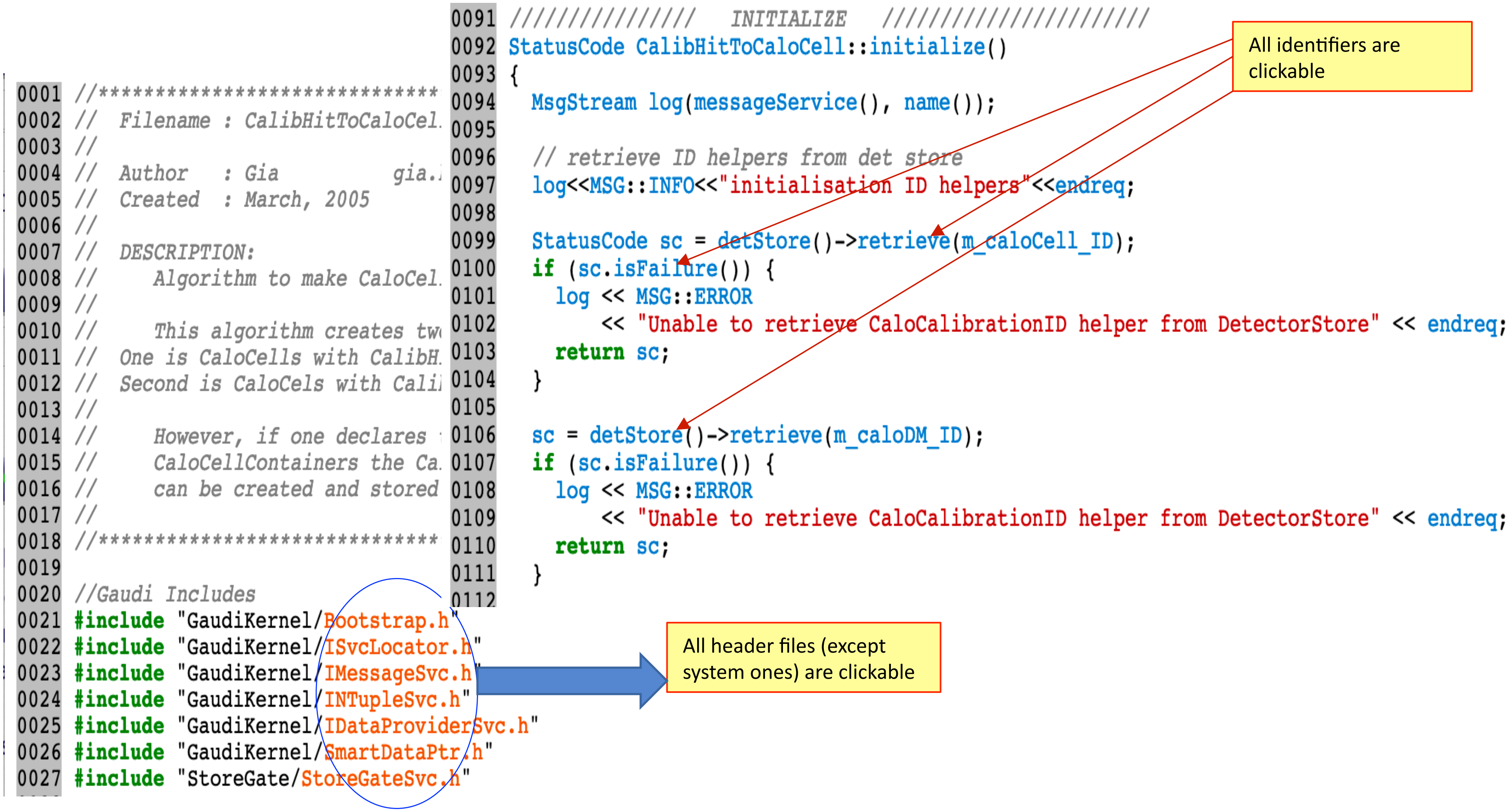
979 occurences found.

Results for GetBranch

File	Line	Text
Gaudi/RootCnv/RootCnv /RootDataConnection.h	220	`virtual TBranch* getBranch( std::string_view section, std::string_view n ) = 0;`
	303	`TBranch* getBranch( std::string_view section, std::string_view branch_name ) {`
	304	`return m_tool->getBranch( section, branch_name );`
	307	`TBranch* getBranch( std::string_view section, std::string_view branch_name, TClass* cl, void* ptr, int buff_siz,`
Gaudi/RootCnv/merge/extractEvt.C	265	`br_in = m_ref_in->GetBranch( "Databases" );`
	267	`br_out = m_ref_out->GetBranch( "Databases" );`
	275	`br_in = m_ref_in->GetBranch( "Containers" );`
	277	`br_out = m_ref_out->GetBranch( "Containers" );`
	284	`br_in = m_ref_in->GetBranch( "Links" );`
	286	`br_out = m_ref_out->GetBranch( "Links" );`
	293	`br_in = m_ref_in->GetBranch( "Params" );`
	295	`br_out = m_ref_out->GetBranch( "Params" );`
	324	`br_out           = m_evt_out->GetBranch( name );`

LXR: Clickable Identifiers & Header Files

All identifiers and header files (except system ones) are clickable.



```
//////////////////   INITIALIZE   ////////////////////////
StatusCode CalibHitToCaloCell::initialize()
{
  MsgStream log(messageService(), name());

  // retrieve ID helpers from det store
  log<<MSG::INFO<<"initialisation ID helpers"<<endreq;

  StatusCode sc = detStore()->retrieve(m_caloCell_ID);
  if (sc.isFailure()) {
    log << MSG::ERROR
        << "Unable to retrieve CaloCalibrationID helper from DetectorStore" << endreq;
    return sc;
  }

  sc = detStore()->retrieve(m_caloDM_ID);
  if (sc.isFailure()) {
    log << MSG::ERROR
        << "Unable to retrieve CaloCalibrationID helper from DetectorStore" << endreq;
    return sc;
  }
```

```
//******************************
//  Filename : CalibHitToCaloCel
//
//  Author   : Gia           gia.
//  Created  : March, 2005
//
//  DESCRIPTION:
//     Algorithm to make CaloCel
//
//     This algorithm creates tw
//  One is CaloCells with CalibH
//  Second is CaloCels with Cali
//
//     However, if one declares
//     CaloCellContainers the Ca
//     can be created and stored
//
//******************************

//Gaudi Includes
#include "GaudiKernel/Bootstrap.h"
#include "GaudiKernel/ISvcLocator.h"
#include "GaudiKernel/IMessageSvc.h"
#include "GaudiKernel/INTupleSvc.h"
#include "GaudiKernel/IDataProviderSvc.h"
#include "GaudiKernel/SmartDataPtr.h"
#include "StoreGate/StoreGateSvc.h"
```

All identifiers are clickable

All header files (except system ones) are clickable

AtlasSetup: Release Env Setup

Atlas release env is set up by the **AtlasSetup** tool (via **asetup** command).

The purpose of **asetup** is to help users to find Atlas releases and set up all the necessary env in a **convenient** and **fast** way. And the users need not know much about the release location.

It provides:

- For users, look up the location of a release, and set up the required compiler and cmake, then the release env. Something like:

  asetup Athena,22.0.0 # where project=Athena, release=22.0.0

- For release builders, set up the specified compiler (gcc/clang) and cmake. Something like:

  asetup gcc8,cmakesetup,none

The setup is fast: usually takes **4s~7s**.
It is widely used, from grid(Panda)/batch to interactive machines, from T0 to T3.

Atlas Releases

Each Atlas release is composed of hundreds of packages. For example of Athena,21.5.8, it requires 2053 (sub-)packages (internal only), contains 2568 shared libs.

Atlas releases used to be managed by CMT tool (also used by LHCb), and were moved to use cmake in 2018.

- ✧ **CMT**: Configuration Management Tool, a home grown HEP tool, developed by Christian Arnault (LAL), development stopped.
- ✧ **cmake**: open source, widely used, a tool to manage the build process of software using a compiler-independent method

And there are 2 types of releases, usually distributed on cvmfs:

- ◆ Stable releases: release name is numeric like *22.0.1* or *21.2.62.0*.
- ◆ Nightlies releases: release name is timestamp like *2019-09-04T2133* (cmake) or *rel_3* (cmt).

To locate a specific release, it requires a project name (such as *Athena* or *AtlasOffline*), a release name, a branch name (nightlies releases), and platform (x86_64-slc6-gcc8-opt).

asetup Migration from CMT to cmake

Because the old cmt asetup, designed for cmt releases, is complicated, unnecessary for cmake releases, and difficult to maintain/improve.

Thus a **new simplified asetup** was **re-designed** for cmake releases.

Here are some comparison numbers: cmake vs cmt

❖9 (17) files, 8 (20) classes, 94 (224) functions,

4.2K (**15K**) lines (excluding tests)

❖ Compiler setup: 1 (3) function, **129** (**850**) lines,

❖ Release locator: 1 (7) function, **208** (**1350**) lines.

❖ Full option help printout: 176 (410) lines; **210 options removed** in cmake asetup.

The above comparison demonstrates:
how complicated in the old asetup to locate a release and to set up a compiler.

asetup Deployment

Because asetup is **widely used**, we must be very careful on a new version deployment:

- StressTest (composed of >70 different tests) under native SL7,

  SL6/SL7 containers (many grid jobs run within containers).

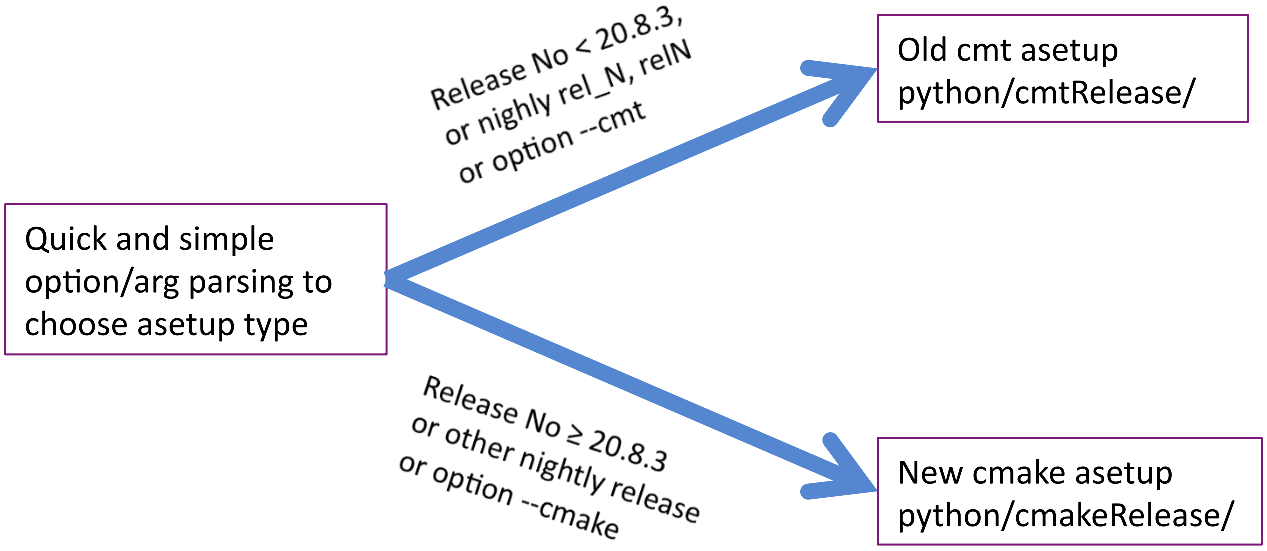
  and comparison with previous version.
- Tests on ALRB
- Also HammerCloud tests during cmt=>cmake migration

A new asetup version is usually deployed as a beta (testing) version first.

Then it will become a production version if no problem is found.

Determination of Release Type

The old cmt releases will co-exist with new cmake releases for a while, and asetup is widely used.
Make the change **transparent** to users/panda as much as possible.



Quick and simple option/arg parsing to choose asetup type

Release No < 20.8.3,
or nighly rel_N, relN
or option --cmt

Old cmt asetup
python/cmtRelease/

Release No ≥ 20.8.3
or other nightly release
or option --cmake

New cmake asetup
python/cmakeRelease/

Then pass the remaining option/arg to cmt/cmake asetup.
Issue warning msg for **258** old deprecated **tags** and **210** deprecated **long options**.

asetup Command Options

There are hundreds of options in cmt asetup, and about 60 in cmake asetup.

The help printout with all options would be very len…gthy, and not helpful.

Solution: divide the options into a few groups and print live help per group or brief help fitting on one screen.

% asetup –h

```
Usage: asetup [options] [tags]

Options:
  -h, --help            show this help message and exit
  --version             Print out the asetup version
  --printLast, --printLastSession
                        Print out the last saved session info under PWD
  --listRemovedTags     Print out the list of removed old asetup tags
  --listRemovedOpts, --listRemovedOptions
                        Print out the list of removed old asetup options
  --releasebase=<dir>, --releasepath=<dir>
                        Specify an exact path to the release bypassing the area search
  --helpMoreOn=<group>  The group name for help printout, valid groups: [CMAKE,
                        RELLOC,ASETUP,DB,PLATFORM,RELNAME,ENVVAR,COMPILER,ALL],
                        where 'All' will print all groups options

Examples: asetup Athena,22.0.0         # set up stable release 22.0.0
          asetup Athena,master,latest  # set up the latest master nightlies
          asetup Athena,master,r08     # set up the master nightlies of date 08

 or asetup source otherScript    # source additional user script

For further details, look at
  https://twiki.cern.ch/twiki/bin/view/AtlasComputing/AtlasSetupReference
```

asetup Configuration and Debug

There are multiple layers of configuration for sites/users:

- ✓ Default configuration in asetup
- ✓ $AtlasSetupSiteCMake, $AtlasSetupSite
- ✓ $AtlasSetup/../asetupSite/.asetup
- ✓ $HOME/.asetup, $PWD/.asetup
- ✓ Options --siteconfigfile=, --inputfile=

And one option --dumpconfig to print out all configuration parameters together with their source origin.

There are 2 options for debugging purpse:

- ✓ --debugprint: Indented output and with both function and caller name (for cmake asetup, automatical name lookup)
- ✓ --simulate: keep and do not source the generated temporary shell.

Convenient Ways to Use asetup

It is not very convenient to use options, like:

Asetup --project=Athena --release=22.0.0 --os=slc6 --gccversion=6.2

With the automatically translated **tags** from most long options, you can run instead:

asetup Athena,22.0.0,slc6,gcc62

Fast/convenient to setup the **latest** nightlies release, and shortcut to timestamp nightly release (using prefix-**r**, says r04 for 2019-09-04T*, otherwise it would be difficult to remember/look up, there are ~100 nightlies under branch master/)

asetup master,Athena,latest

*asetup master,Athena,**r04***

Restore the last session env under one work directory, just run without arg or option --restore

It would print out the following msg and set up the previously used release under that directory:

% asetup

```
Set up the following release as last used under the current directory
Using Athena/22.0.0 [cmake] with platform x86_64-slc6-gcc62-opt
        at /cvmfs/atlas.cern.ch/repo/sw/software/22.0
```

An option to print out the last saved release info:

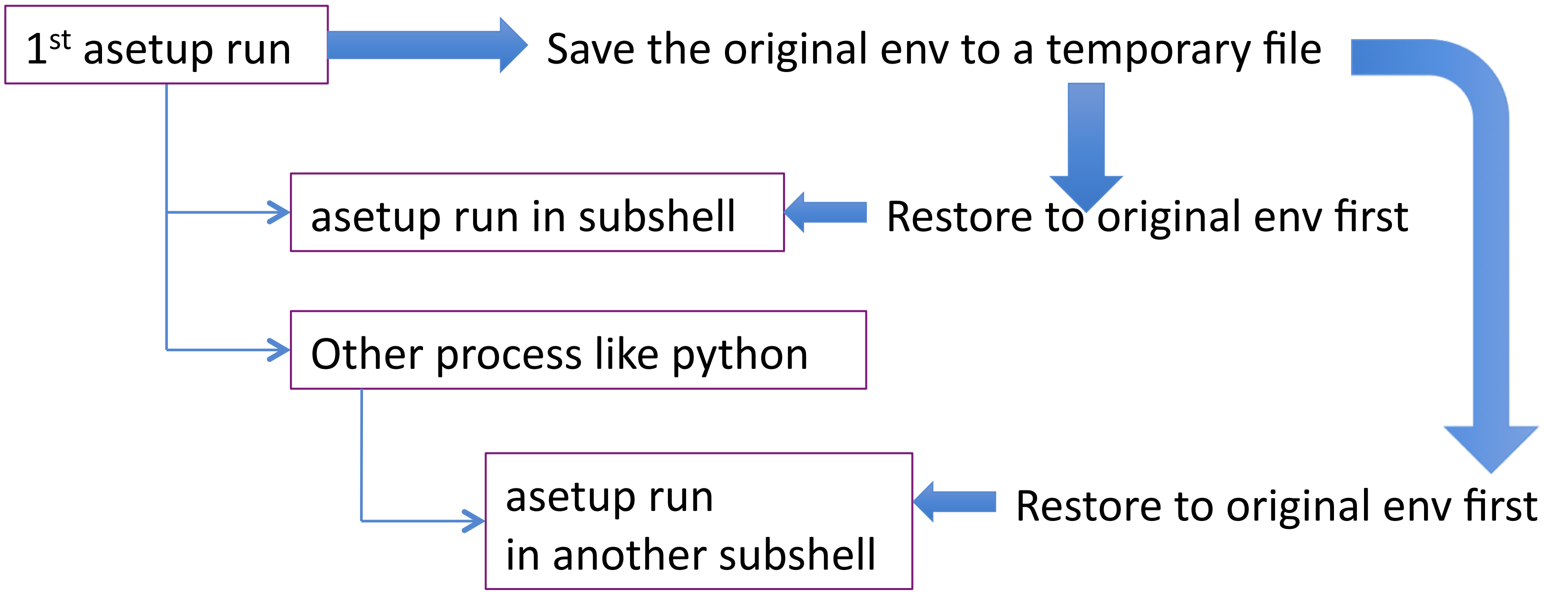
% asetup --printLast

```
The last saved release under the current directory is:
        Athena/22.0.0 [cmake] with platform x86_64-slc6-gcc62-opt
        at /cvmfs/atlas.cern.ch/repo/sw/software/22.0

Afterwards, the following epilog script(s) will also be sourced
        /cvmfs/atlas.cern.ch/repo/ATLASLocalRootBase/swConfig/asetup/asetupEpilog.sh
```

Shell Env Restoration in asetup

Sourcing the **new** release setup script **on** an **existing** release env will mess up the env.

Solution: save the shell env at the 1st time asetup run.



1st asetup run

Save the original env to a temporary file

asetup run in subshell

Restore to original env first

Other process like python

asetup run in another subshell

Restore to original env first

The subshell can be any levels below the shell of the 1st asetup run.

There is also a mechanism to clean up those saved env files during each new asetup run.

Resolving Conflicts b/w Releases & System

Due to alternate system libraries, some Atlas releases break system commands such as emacs, git or expr.

For example, new manipulation instruction set in libgmp.so is not available in some old CPUs, breaking the command expr.

asetup will check those conflict, and provides alias/wrapper with no/little performance loss:

```
alias expr=‘/lib64/ld-linux-x86-64.so.2 --library-path /usr/lib64 /bin/expr’
```

Wrapper need varies from release to release, is generated on the fly, and its path is added in $PATH for usage in subshell.

The wrapper in work directory is also cleaned up automatically.

In addition, if users have already aliased the command, the re-aliased command will carry over user option, says,

```
% alias emacs=“emacs -nw”
% alias_sys_exe  emacs
% alias emacs
   emacs='/lib64/ld-linux-x86-64.so.2 --library-path /usr/lib64 /bin/emacs -nw’
```

Where alias_sys_exe is introduced in asetup, and aliasing emacs is done automatically in release env setup if necessary.

USAtlas Shared Tier-3s at BNL/SLAC

There are 2 shared Tier-3s for USAtlas users:

- BNL (introduced first)
- SLAC (added later)

There are a lot of CPU/Space available at both shared Tier-3 sites:

	BNL	SLAC
Interactive-CPU	8 machines (8 cores each)	88 cores
Batch-CPU	~2300 slots (25K in shared pool)	~3000 cores
Private space	20GB (home), 500GB (data), 5TB (dCache)	100 GB (home), 2TB (data, could increase up to 10TB)
Shared space	Rucio RSEs (localgroupdisk)	Rucio RSEs (SLACXRD*)
Remote X-Window	NoMachine NX	FastX
Batch system	Condor	LSF

Usage at Shared Tier-3's

There are many users using the shared Tier-3 sites:

	BNL	SLAC
Users	130 (30 institutes)	50+

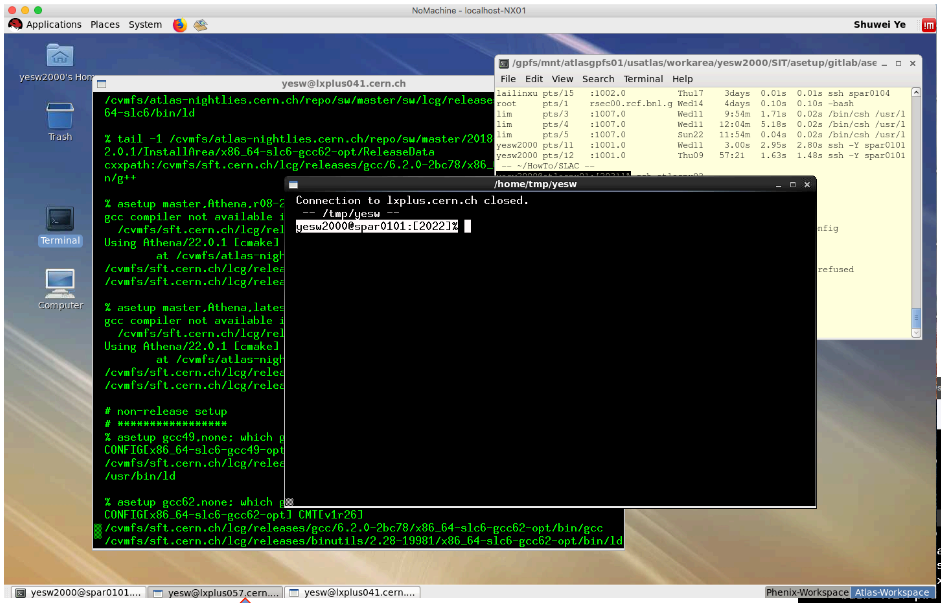
Space/batch usage at BNL:

	Users
Private space (20GB)	64 (>50% usage)
Data space (500GB)	14 (>50% usage)
dCache space (5TB)	30 (>1TB)

Batch	No (users)	No (institutes)	No (jobs)	Ave time
Last 6 months	50	15	2.84M	2800s
Last 12 months	57	16	7.96M	2137s

Remote X-Window at BNL/SLAC

BNL/SLAC provide different Remote X Window



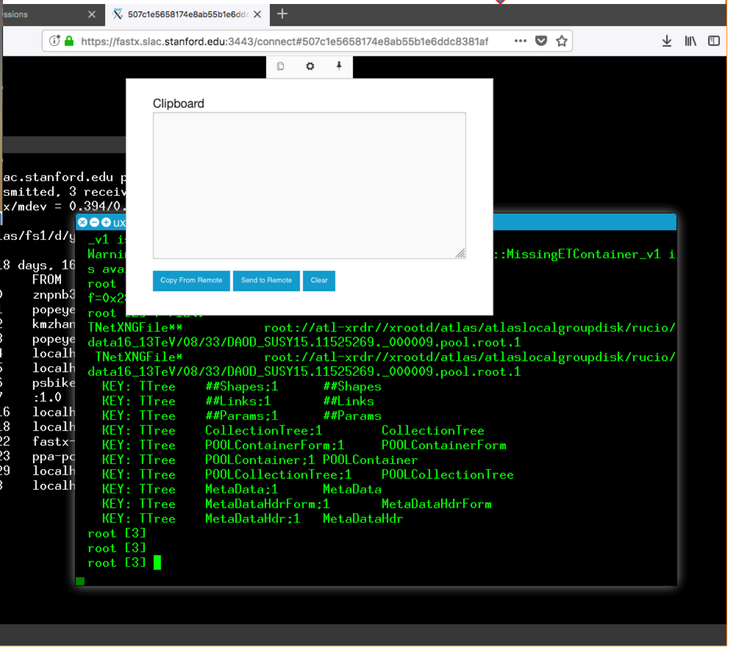
```
/cvmfs/atlas-nightlies.cern.ch/repo/sw/master/sw/lcg/releases
64-slc6/bin/ld

% tail -1 /cvmfs/atlas-nightlies.cern.ch/repo/sw/master/2018
2.0.1/InstallArea/x86_64-slc6-gcc62-opt/ReleaseData
cxxpath:/cvmfs/sft.cern.ch/lcg/releases/gcc/6.2.0-2bc78/x86_
n/g++

% asetup master,Athena,r08-2
gcc compiler not available i
  /cvmfs/sft.cern.ch/lcg/rel
Using Athena/22.0.1 [cmake]
        at /cvmfs/atlas-nigh
/cvmfs/sft.cern.ch/lcg/relea
/cvmfs/sft.cern.ch/lcg/relea

% asetup master,Athena,lates
gcc compiler not available i
  /cvmfs/sft.cern.ch/lcg/rel
Using Athena/22.0.1 [cmake]
        at /cvmfs/atlas-nigh
/cvmfs/sft.cern.ch/lcg/relea
/cvmfs/sft.cern.ch/lcg/relea

# non-release setup
# *****************
% asetup gcc49,none; which g
CONFIG[x86_64-slc6-gcc49-opt
/cvmfs/sft.cern.ch/lcg/relea
/usr/bin/ld

% asetup gcc62,none; which g
CONFIG[x86_64-slc6-gcc62-opt] CMT[v1r26]
/cvmfs/sft.cern.ch/lcg/releases/gcc/6.2.0-2bc78/x86_64-slc6-gcc62-opt/bin/gcc
/cvmfs/sft.cern.ch/lcg/releases/binutils/2.28-19981/x86_64-slc6-gcc62-opt/bin/ld
```

```
lailinxu pts/15   :1002.0          Thu17    3days  0.01s  0.01s ssh spar0104
root     pts/1    rsec00.rcf.bnl.g Wed14    4days  0.10s  0.10s -bash
lim      pts/3    :1007.0          Wed11    9:54m  1.71s  0.02s /bin/csh /usr/l
lim      pts/4    :1007.0          Wed11   12:04m  5.18s  0.02s /bin/csh /usr/l
lim      pts/5    :1007.0          Sun22   11:54m  0.04s  0.02s /bin/csh /usr/l
yesw2000 pts/11   :1001.0          Wed11    3.00s  2.95s  2.80s ssh -Y spar0101
yesw2000 pts/12   :1001.0          Thu09   57:21   1.63s  1.48s ssh -Y spar0101
 -- ~/HowTo/SLAC --
nfig
refused
```

```
Connection to lxplus.cern.ch closed.
 -- /tmp/yesw --
yesw2000@spar0101:[2022]%
```

NoMachine at BNL

FastX at SLAC


ssions
507c1e5658174e8ab55b1e6dd
https://fastx.slac.stanford.edu:3443/connect#507c1e5658174e8ab55b1e6ddc8381af
Clipboard
Copy From Remote
Send to Remote
Clear
Phenix-Workspace
Atlas-Workspace
ux
uxterm
uxterm

```
ac.stanford.edu p
smitted, 3 receiv
x/mdev = 0.394/0.
/gpfs/slac/atlas/fs1/d/y
-bash-4.1$ w
 07:56:35 up 18 days, 16
USER     TTY      FROM
jian     pts/0    znpnb3
jonl     pts/1    popeye
kmzhang  pts/2    kmzhan
jonl     pts/3    popeye
colocho  pts/4    localh
colocho  pts/5    localh
wilko    pts/6    psbike
traorei  pts/7    :1.0
colocho  pts/16   localh
iwhite   pts/18   localh
yesw     pts/22   fastx-
russell  pts/23   ppa-pc
yesw     pts/29   localh
loney    pts/8    localh
-bash-4.1$
```

```
_v1 i
Warni                                                ::MissingETContainer_v1 i
s ava
root
f=0x2
root
TNetXNGFile**           root://atl-xrdr//xrootd/atlas/atlaslocalgroupdisk/rucio/
data16_13TeV/08/33/DAOD_SUSY15.11525269._000009.pool.root.1
 TNetXNGFile*           root://atl-xrdr//xrootd/atlas/atlaslocalgroupdisk/rucio/
data16_13TeV/08/33/DAOD_SUSY15.11525269._000009.pool.root.1
  KEY: TTree    ##Shapes;1      ##Shapes
  KEY: TTree    ##Links;1       ##Links
  KEY: TTree    ##Params;1      ##Params
  KEY: TTree    CollectionTree;1        CollectionTree
  KEY: TTree    POOLContainerForm;1     POOLContainerForm
  KEY: TTree    POOLContainer;1 POOLContainer
  KEY: TTree    POOLCollectionTree;1    POOLCollectionTree
  KEY: TTree    MetaData;1      MetaData
  KEY: TTree    MetaDataHdrForm;1       MetaDataHdrForm
  KEY: TTree    MetaDataHdr;1   MetaDataHdr
root [3]
root [3]
root [3]
```

dCache at BNL: Dataset Listing

Developed a tool to generate clist file for both official/private datasets on dCache.

% **pnfs_ls.py –h**

```
Usage:
     pnfs_ls.py [options] dsetListFile
  or
     pnfs_ls.py [options] dsetNamePattern[,dsetNamePattern2[,more namePatterns]]
  or
     pnfs_ls.py -o clistFilename /pnfs/FilePathPattern [morePaths]
  or
     pnfs_ls.py -p -o clistFilename [pnfsFilePath | pnfsDirPath] [morePaths]

  This script generates pfn (physical file name), pnfs-path,
or xrootd-path of files on BNL dcache for given datasets or files on PNFS,
where wildcard and symlink are supported in pnfsFilePath and pnfsDirPath

  Options:
  -h, --help            show this help message and exit
  -v                    Verbose
  -V, --version         print my version
  -p, --privateFiles    List private non-dataset files on dCache
  -i, --incomplete      Use incomplete sites if complete not available
  -u, --usersDCache     Use datasets under users private dCache
  -l, --listOnly        list only matched datasets under users dCache, no pfn output
  -o OUTPFNFILE, --outPfnFile=OUTPFNFILE
                        write pfn list into a file instead of printing to the screen
  -d OUTPFNDIR, --dirForPfn=OUTPFNDIR
                        write pfn list into a directory with a file per dataset
  -N, --usePNFS         using pNFS access, default is xrootd within BNL
  --useXRootdOutside    using xroot from outside BNL: access, default is
                        xrootd within BNL
  -L LOCALBNLSITE, --localBNLSite=LOCALBNLSITE
                        specify a BNL site, overriding the one choosen by the script
```

Hiro developed a tool rucio-bnl-get for fast dataset replication onto private dCache.

Users had problem with this tool and dCache access from to time.

Copying Files/Dirs to dCache at BNL

Developed a tool to help copy files to BNL dCache in multiple threads.

% **pnfs-copy.py –h**

```
Usage: pnfs-copy.py [options] files pnfs_directory
   pnfs-copy.py [options] dirs pnfs_directory

    or using quote-enclosed wildcard such as
       pnfs-copy.py [options] "*.root" pnfs_directory

   This script uses gfal-copy to conduct the file copy, thus requires the
  setup of RucioClients and a valid grid proxy.

Options:
  -h, --help  show this help message and exit
  --verbose   Print verbose info
  --version   Print the script version then exit
```

On default it starts 3 threads with 3 retries, and will report the copy status: numbers of successfully copied files and failed files after 4 tries.

Survey on SLAC T3 Users & Improvements

Since SLAC T3 was just added, we conducted a 19-question survey.

- There are 8 responses in total
- User support is very good
- Enough space to most users
- 6/8 could use SLAC as local T3 alternative, and would recommend to collaborators
- Quick batch jobs start
- Documentation needs improvement
- Most users have never used FastX
- Batch job failure rate is not very low, but many of them should be able to be fixed easily

We (Wei & I) have improved the documentation (easy to follow). SLAC would think about FastX and possible replacement.

I plan to development a tool similar to pnfs_ls.py for SLAC.

Xcache at SLAC/BNL

Xcache enables to access data remotely and also to cache them (**fully** or **partially**) locally for faster access in future.

SLAC/BNL provide the Xcache server to try:

- SLAC Xcache server: **root://atlfax.slac.stanford.edu/**
- BNL Xcache server: **root://xrootd03.usatlas.bnl.gov:1094/**

There are two ways of Xcache:

1. Specify both Xcache server and remote xrootd server, hence contain **TWO** root:/

root://atlfax.slac.stanford.edu//**root:**//dcgftp.usatlas.bnl.gov:1094/pnfs/usatlas.bnl.gov/LOCALGROUPDISK/rucio/data18_13TeV/da/ea/DAOD_EXOT12.14278917._000001.pool.root.1

root://xrootd03.usatlas.bnl.gov:1094//**root:**//griddev03.slac.stanford.edu:2094//xrootd/atlas/atlaslocalgroupdisk/rucio/data16_13TeV/f9/bd/DAOD_SUSY15.11525262._000003.pool.root.1

2. gLFN (global Logical File Name) access, **without knowing file location**

root://atlfax.slac.stanford.edu/**/atlas/rucio/***data18_13TeV:DAOD_EXOT12.14278917._000001.pool.root.1*

root://xrootd03.usatlas.bnl.gov:1094/**/atlas/rucio/***data16_13TeV:DAOD_SUSY15.11525262._000003.pool.root.1*

We need test and study the performance.

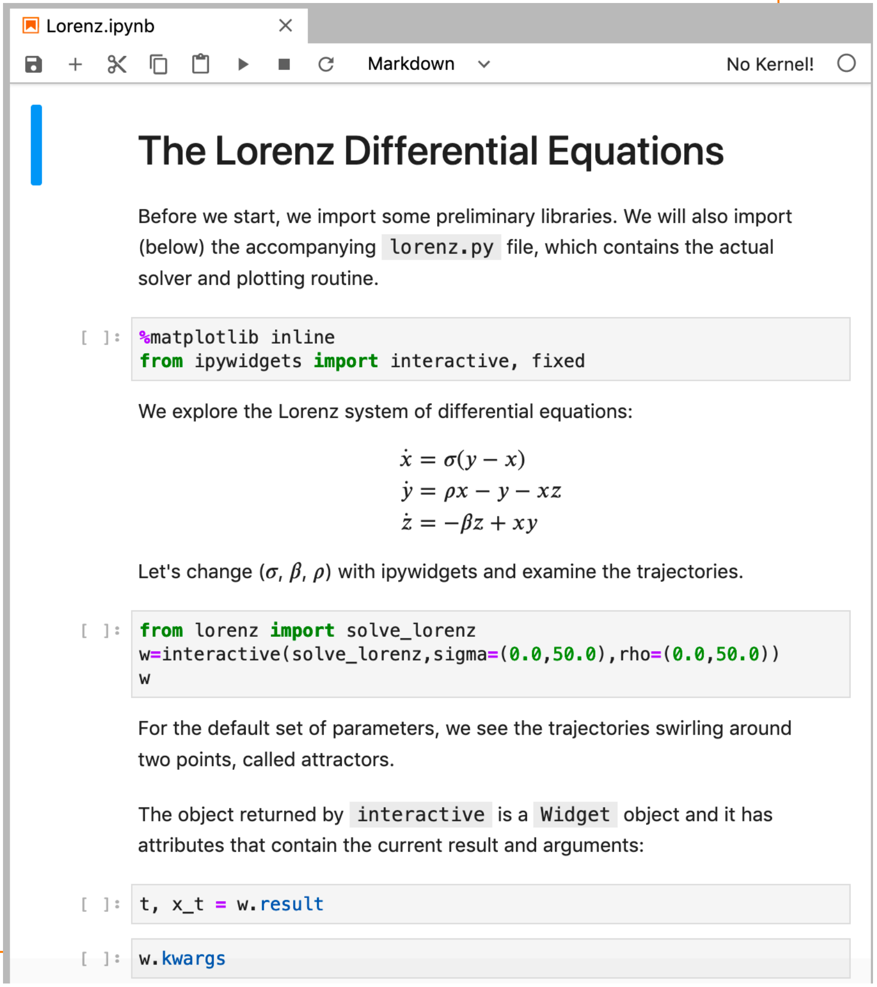
Jupyter at BNL/SLAC

BNL/SLAC provide experimental Jupyter Notebook service (powerful w/ GPUs).

Jupyter Notebook provides the following features:

- Interactive on a web client (remotely or locally)
- Explanatory rich text (shown on the right)
- Intersperse code with its result (charts/figures)
- Instant result after code change
- Work stored on a notebook document
- Easy to share/reproduce
- Able to convert to other formats (script, PDF, latex, html, …)

Available kernels are python (default), R, Ruby, Julia, etc.



The Lorenz Differential Equations

Before we start, we import some preliminary libraries. We will also import (below) the accompanying `lorenz.py` file, which contains the actual solver and plotting routine.

```
[ ]: %matplotlib inline
     from ipywidgets import interactive, fixed
```

We explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

Let's change (σ, β, ρ) with ipywidgets and examine the trajectories.

```
[ ]: from lorenz import solve_lorenz
     w=interactive(solve_lorenz,sigma=(0.0,50.0),rho=(0.0,50.0))
     w
```

For the default set of parameters, we see the trajectories swirling around two points, called attractors.

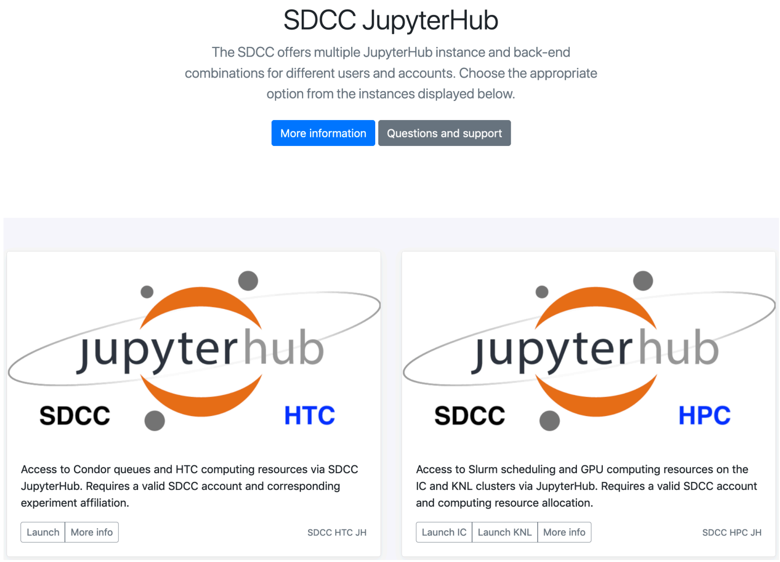
The object returned by `interactive` is a `Widget` object and it has attributes that contain the current result and arguments:

```
[ ]: t, x_t = w.result
```

```
[ ]: w.kwargs
```

Logging In to BNL Jupyter

Visit Webpage: https://jupyter.sdcc.bnl.gov/
providing 2 types: HTC and HPC (GPU)



SDCC JupyterHub

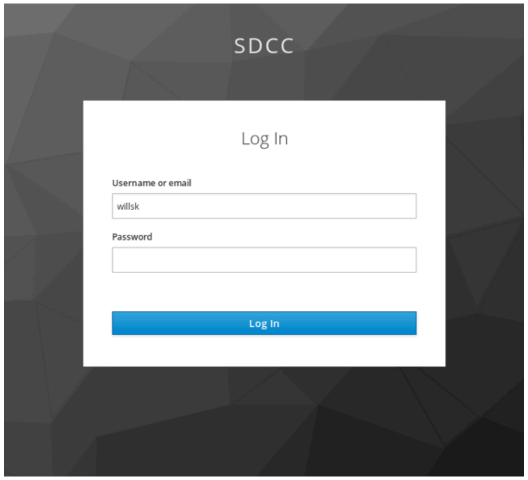
The SDCC offers multiple JupyterHub instance and back-end combinations for different users and accounts. Choose the appropriate option from the instances displayed below.

Access to Condor queues and HTC computing resources via SDCC JupyterHub. Requires a valid SDCC account and corresponding experiment affiliation.

Access to Slurm scheduling and GPU computing resources on the IC and KNL clusters via JupyterHub. Requires a valid SDCC account and computing resource allocation.

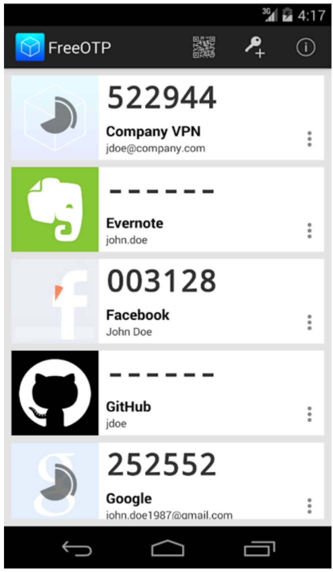
Two-step authentication:

1. SDCC account login
2. QR Code → Google Authenticator or FreeOTP app
   Easy setup by scanning QR code first time



Username or email: willsk

Password:



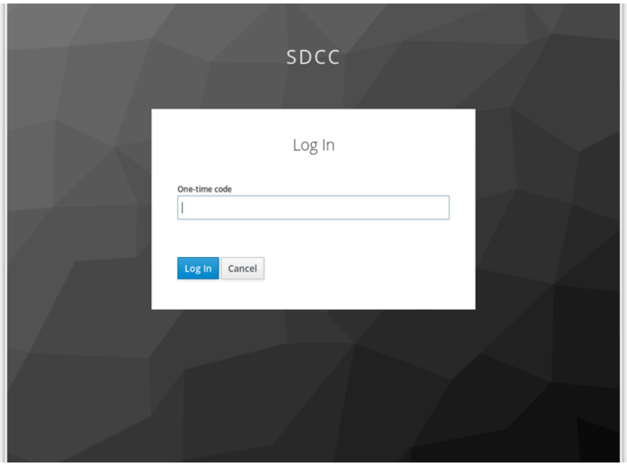
522944
Company VPN
jdoe@company.com

- - - - - -
Evernote
john.doe

003128
Facebook
John Doe

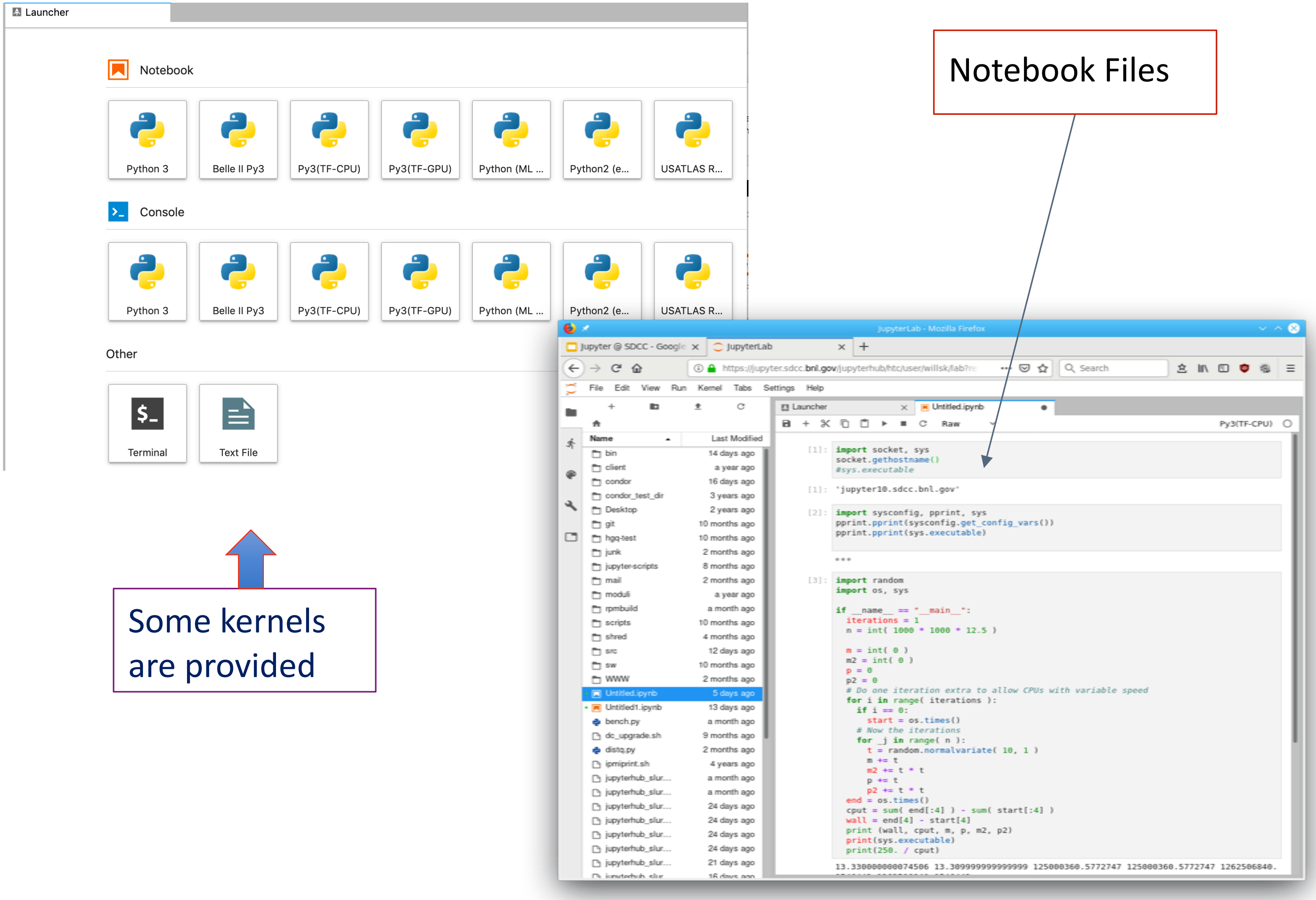
- - - - - -
GitHub
jdoe

252552
Google
john.doe1987@gmail.com



One-time code:

Jupyter at BNL



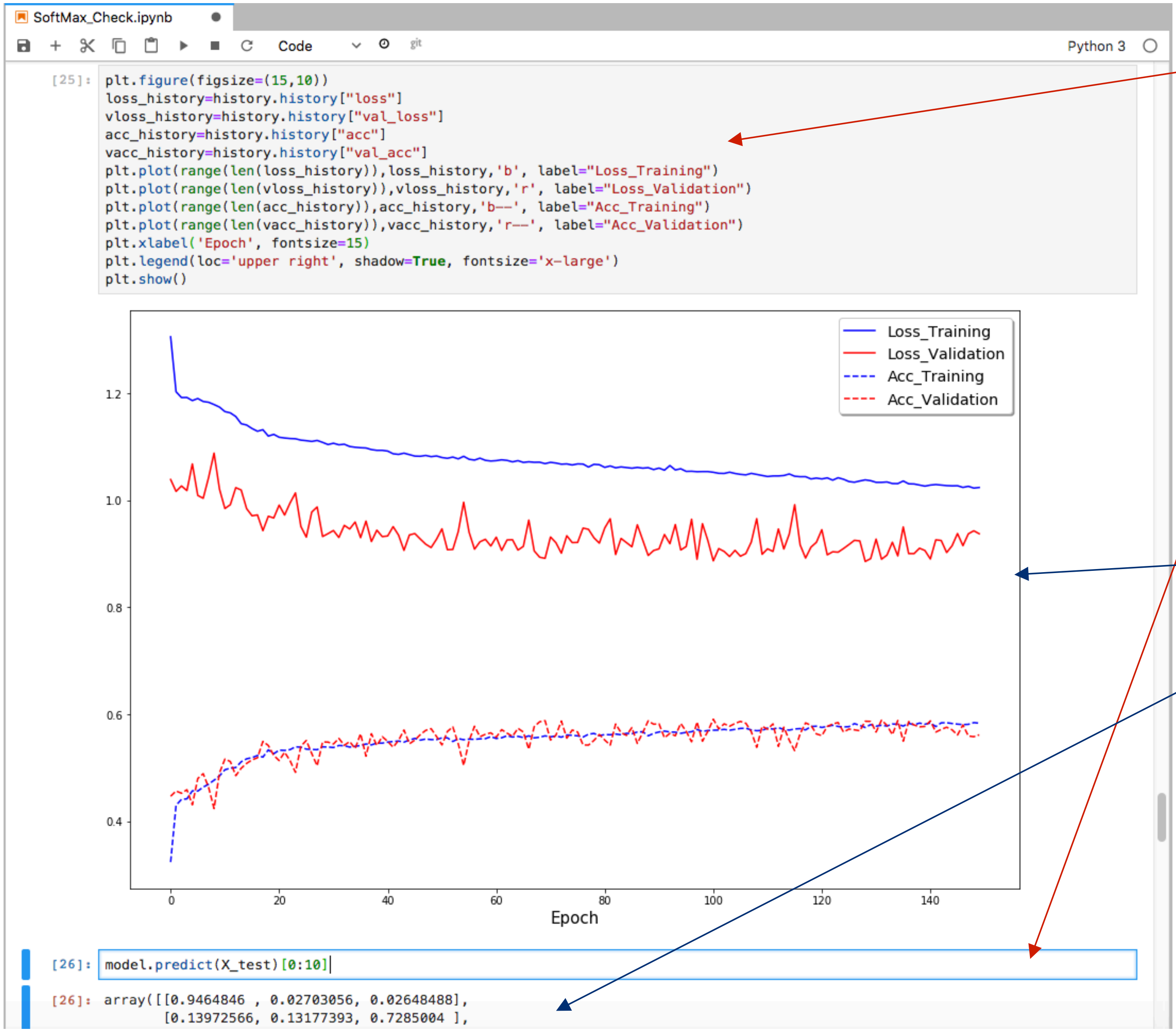
Name	Last Modified
bin	14 days ago
client	a year ago
condor	16 days ago
condor_test_dir	3 years ago
Desktop	2 years ago
git	10 months ago
hgq-test	10 months ago
junk	2 months ago
jupyter-scripts	8 months ago
mail	2 months ago
moduli	a year ago
rpmbuild	a month ago
scripts	10 months ago
shred	4 months ago
src	12 days ago
sw	10 months ago
WWW	2 months ago
Untitled.ipynb	5 days ago
Untitled1.ipynb	13 days ago
bench.py	a month ago
dc_upgrade.sh	9 months ago
distq.py	2 months ago
ipmiprint.sh	4 years ago
jupyterhub_slur....	a month ago
jupyterhub_slur....	a month ago
jupyterhub_slur....	24 days ago
jupyterhub_slur....	24 days ago
jupyterhub_slur....	24 days ago
jupyterhub_slur....	24 days ago
jupyterhub_slur....	21 days ago
jupyterhub_slur	16 days ago

```
[1]: import socket, sys
     socket.gethostname()
     #sys.executable

[1]: 'jupyter10.sdcc.bnl.gov'

[2]: import sysconfig, pprint, sys
     pprint.pprint(sysconfig.get_config_vars())
     pprint.pprint(sys.executable)

     ...

[3]: import random
     import os, sys

     if __name__ == "__main__":
       iterations = 1
       n = int( 1000 * 1000 * 12.5 )

       m = int( 0 )
       m2 = int( 0 )
       p = 0
       p2 = 0
       # Do one iteration extra to allow CPUs with variable speed
       for i in range( iterations ):
         if i == 0:
           start = os.times()
         # Now the iterations
         for _j in range( n ):
           t = random.normalvariate( 10, 1 )
           m += t
           m2 += t * t
           p += t
           p2 += t * t
       end = os.times()
       cput = sum( end[:4] ) - sum( start[:4] )
       wall = end[4] - start[4]
       print (wall, cput, m, p, m2, p2)
       print(sys.executable)
       print(250. / cput)

     13.330000000074506 13.309999999999999 125000360.5772747 125000360.5772747 1262506840.
```

Notebook Files

Some kernels are provided

Jupyter Example at BNL



```
[25]: plt.figure(figsize=(15,10))
      loss_history=history.history["loss"]
      vloss_history=history.history["val_loss"]
      acc_history=history.history["acc"]
      vacc_history=history.history["val_acc"]
      plt.plot(range(len(loss_history)),loss_history,'b', label="Loss_Training")
      plt.plot(range(len(vloss_history)),vloss_history,'r', label="Loss_Validation")
      plt.plot(range(len(acc_history)),acc_history,'b--', label="Acc_Training")
      plt.plot(range(len(vacc_history)),vacc_history,'r--', label="Acc_Validation")
      plt.xlabel('Epoch', fontsize=15)
      plt.legend(loc='upper right', shadow=True, fontsize='x-large')
      plt.show()
```

```
[26]: model.predict(X_test)[0:10]
```

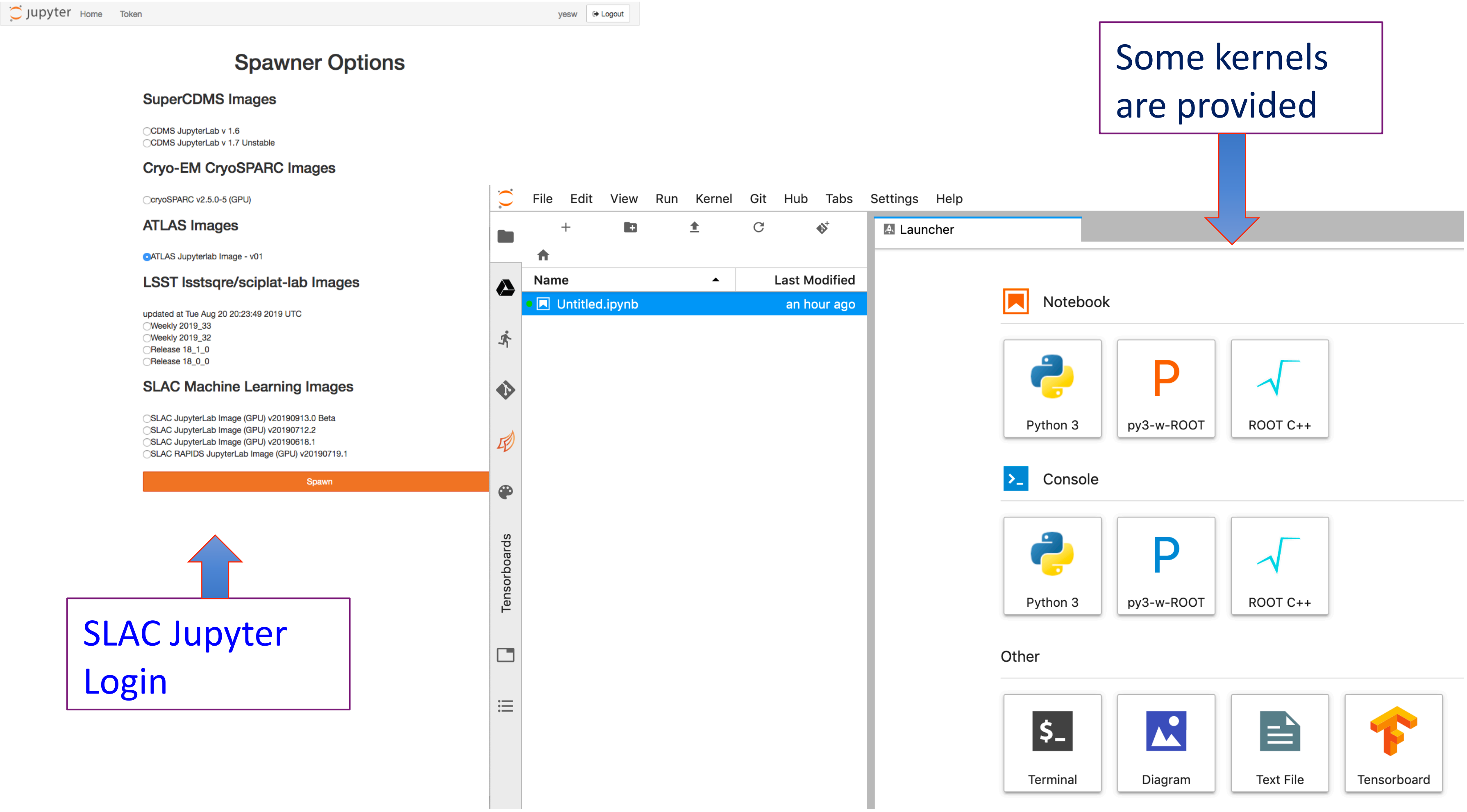
```
[26]: array([[0.9464846 , 0.02703056, 0.02648488],
             [0.13972566, 0.13177393, 0.7285004 ],
```

Code input

Result right after code

Jupyter at SLAC

SLAC Jupyter Login: https://jupyter.slac.stanford.edu



Spawner Options

SuperCDMS Images

○ CDMS JupyterLab v 1.6
○ CDMS JupyterLab v 1.7 Unstable

Cryo-EM CryoSPARC Images

○ cryoSPARC v2.5.0-5 (GPU)

ATLAS Images

● ATLAS Jupyterlab Image - v01

LSST lsstsqre/sciplat-lab Images

updated at Tue Aug 20 20:23:49 2019 UTC
○ Weekly 2019_33
○ Weekly 2019_32
○ Release 18_1_0
○ Release 18_0_0

SLAC Machine Learning Images

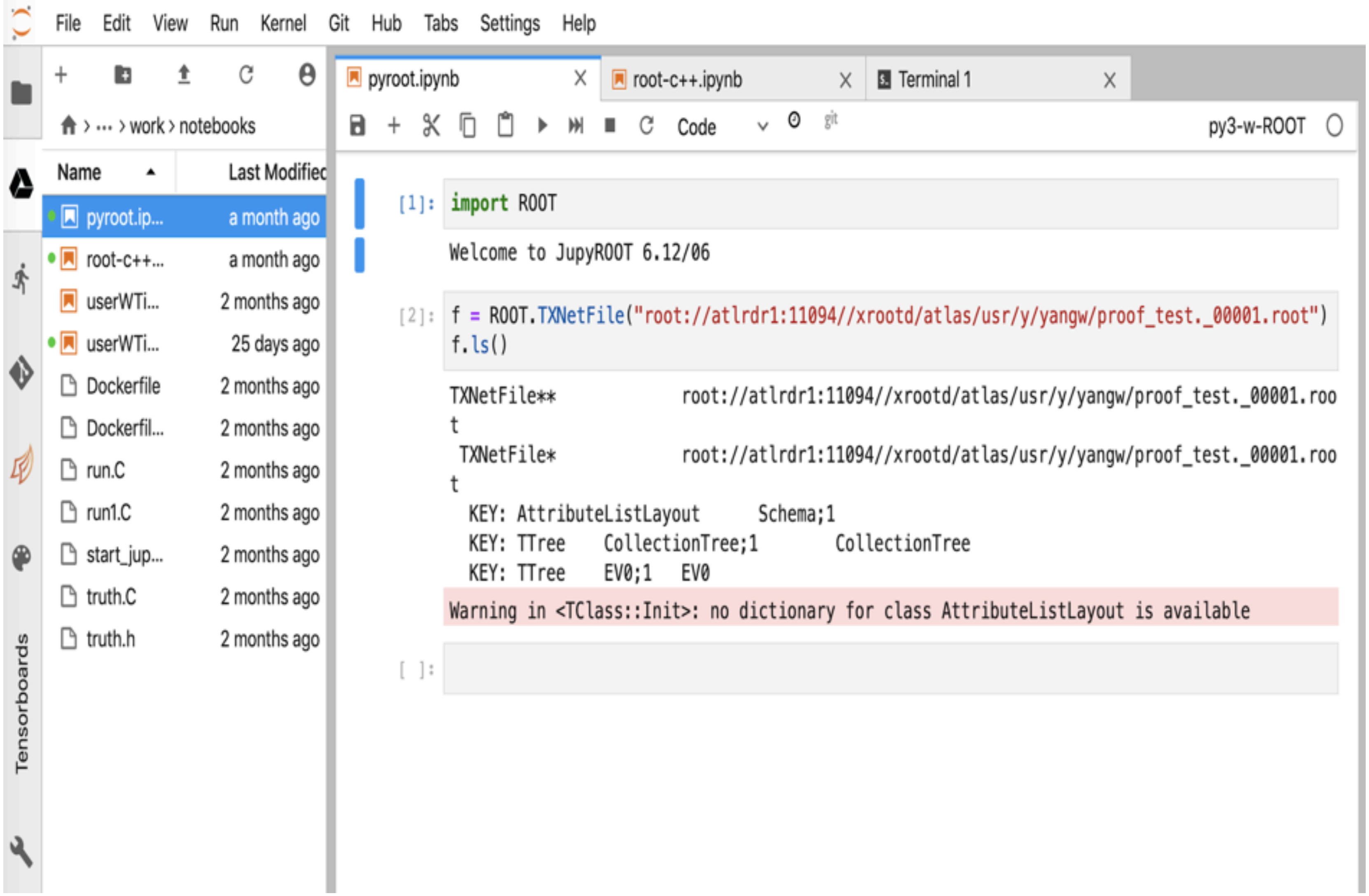
○ SLAC JupyterLab Image (GPU) v20190913.0 Beta
○ SLAC JupyterLab Image (GPU) v20190712.2
○ SLAC JupyterLab Image (GPU) v20190618.1
○ SLAC RAPIDS JupyterLab Image (GPU) v20190719.1

Name	Last Modified
Untitled.ipynb	an hour ago

Some kernels are provided

SLAC Jupyter Login

Jupyter Example at SLAC



Name ▲	Last Modified
pyroot.ip...	a month ago
root-c++...	a month ago
userWTi...	2 months ago
userWTi...	25 days ago
Dockerfile	2 months ago
Dockerfil...	2 months ago
run.C	2 months ago
run1.C	2 months ago
start_jup...	2 months ago
truth.C	2 months ago
truth.h	2 months ago

```
[1]: import ROOT
```

```
Welcome to JupyROOT 6.12/06
```

```
[2]: f = ROOT.TXNetFile("root://atlrdr1:11094//xrootd/atlas/usr/y/yangw/proof_test._00001.root")
     f.ls()
```

```
TXNetFile**          root://atlrdr1:11094//xrootd/atlas/usr/y/yangw/proof_test._00001.roo
t
 TXNetFile*          root://atlrdr1:11094//xrootd/atlas/usr/y/yangw/proof_test._00001.roo
t
  KEY: AttributeListLayout     Schema;1
  KEY: TTree     CollectionTree;1       CollectionTree
  KEY: TTree     EV0;1   EV0
Warning in <TClass::Init>: no dictionary for class AttributeListLayout is available
```

```
[ ]:
```