Read the Docs Template Documentation

Release 1.0

Read the Docs

Jun 11, 2019

CONTENTS

Contents:

CHAPTER

ONE

DATA ORGANIZATION

1. home: to keep your light weight & precious files (codes, scripts, figures,...)
2. workspace: for all personal data files (post-processing runs, diagnostics,...) It can be accessed at:

```
/local/data/<server>/workspace/<username>
```

   Or by using the corresponding environment variable:

```
cd $workspace
```

3. Observations: if you download, regrid, process,... observations into a dataset that can be useful to others. Please contact admin to move it to the shared space located at:

```
/local/data/<server>/observations
```

   Quick access with:

```
cd $observations
```

4. Simulations: model inputs/outputs from the group valuable (published, in prep.) experiments, as well as reference experiments from others group (NCAR, GFDL,...) that are of common interest should be moved to:

```
/local/data/<server>/simulations
```

   Quick access with:

```
cd $simulations
```

1.1 Discover data with intake:

Install intake and intake_xarray installed in your conda environment:

```
conda install -c intake intake
conda install -c conda-forge intake-xarray
```

you can explore the datasets available using intake catalogs available here:

```
/local/data/<server>/catalogs
```

For example, exploring ASTE data inside python can be done with:

```
import intake
catalog = intake.Catalog('/local/data/artemis/catalogs/ASTE_catalog.yml')
list(catalog)
```

Once you have found the dataset you're interested in, you can load as follows:

```
ds_biomes = catalog.Biomes_FayMcKinley.to_dask()

# test plot
import matplotlib.pylab as plt
ds_biomes.MeanBiomes.sel(face=4).plot()
plt.show()
```

CHAPTER

TWO

GLOBUS

2.1 Install:

Reference : globus for linux

Install the globus client:

```
wget https://downloads.globus.org/globus-connect-personal/linux/stable/
↪globusconnectpersonal-latest.tgz
tar xzf globusconnectpersonal-latest.tgz
cd globusconnectpersonal-2.3.5/
```

Contrary to what the reference says, we could not create the endpoint in command line. Instead log into the globus website. Go to "Endpoints", then "add Globus Connect Personal endpoint", enter <server_name> in the 'Display name' box and press 'Generate Setup Key". Save the key to clipboard then go back to terminal and run:

```
./globusconnectpersonal -setup <setup_key_from_clipboard>
```

And you're all set!

2.2 Usage:

Some useful globus commands:

```
./globusconnectpersonal -help
./globusconnectpersonal -start
./globusconnectpersonal -stop
./globusconnectpersonal -status
```

By default, globus access only your home directory. To allow globus to access your workspace:

```
./globusconnectpersonal -restrict-paths /local/data/<server_name>/workspace/$(whoami)␣
↪-start
```

CHAPTER

THREE

MOUNT GOOGLE DRIVE FROM CU

3.1 Install:

Install the google drive app from opam:

```
opam init
# Answer y to question then
eval `opam config env`
opam install google-drive-ocamlfuse

mkdir /home/$(whoami)/google_drive
```

The last step, requires X11 forwarding available (use ssh -Y server) within the campus. It will open firefox remotely on the server (painfully slow). Don't try this at home, connection behind vpn is even slower. . . :

```
google-drive-ocamlfuse /home/$(whoami)/google_drive
```

log in with your university email/password, authorize,. . . and you're all set!

3.2 Usage:

Your linux system will interact with your google drive like a local disk.

3.3 Troubleshooting:

If your command line freezes on ls $HOME or similar. You need to ask admin to unmount:

```
sudo umount -f /home/$(whoami)/google_drive
```

then you can remount the drive:

```
google-drive-ocamlfuse /home/$(whoami)/google_drive
```

CHAPTER

FOUR

SETTING UP YOUR CONDA ENVIRONMENT

To install your custom packages, you need to create a personal conda environment on the server. In this example, I show how I created my development environment.

Log into the server with ssh and load anaconda:

```
module load anaconda3
```

Create your personal conda environment, here I choose to name it dev (how original!) and clone the packages already installed in the default environment:

```
conda create --name dev --clone root
```

Activate your newly created environment:

```
source activate dev
```

Install the packages you want (for example Ryan's excellent xgcm) into your environment. (NB: you can't install new packages in the default environment):

```
conda install -c conda-forge xgcm
```

To use your conda environment in the server's jupyterhub, install the ipython kernel for your conda environment:

```
conda install ipykernel
python -m ipykernel install --user --name dev --display-name "Python3 (dev,raf)"
```

The argument for `--name` must correspond to the environment name (e.g. dev). Display-name is how the environment will be shown in jupyterhub. You only need to do this once for this environment and it will become visible in jupyterhub, in the rolling list under **New**. If you create a new environment, just repeat the entire process.

CHAPTER

FIVE

RUNNING JUPYTER NOTEBOOK ON CUIT MACHINES

Foreword: this is best run in a "screen" (virtual terminal) session. See https://linuxize.com/post/how-to-use-linux-screen/ for details.

First, we need to get a compute node allocated for our notebook, either from the global pool:

```
salloc -N 1 -A ocpbgc --exclusive
```

or from our own:

```
salloc -N 1 -A ocpbgc -p ocpbgc --exclusive
```

You can ask for more than one node (-N 2 for 2 nodes,...). Once our node is allocated we can activate our conda environment:

```
source activate myenv
```

We also need to disable this environment variable:

```
unset XDG_RUNTIME_DIR
```

And find the ip of the node:

```
export ip=$( srun hostname -i )
```

Note the ip address, will be refered to <nodeip> below.

Now we can start the notebook (here I start it in my workspace, not home directory):

```
srun jupyter notebook --no-browser --ip=$ip --notebook-dir=/rigel/ocpbgc/users/$whoami
```

The notebook should be running on port 8888. If not adapt the next items to the current port. Leave this terminal alone, or detach the screen session. In a new terminal, we're gonna create a tunnel to the 8888 port of the compute node to the 8010 port of our workstation/laptop, bouncing off the login node (e.g. habanero):

```
ssh -L 8010:<nodeip>:8888 <name_of_login_node>
```

In your web browser, type:

```
localhost:8010
```

And you should be connected to your notebook.

CHAPTER

SIX

GROUP POLICIES

6.1 User accounts:

Request for new user accounts on the group shared computing must be sent to group leader (Galen McKinley) with CC to the server admin. Upon approval from group leader, account will be created. Users authenticate through LDAP system and require an active UNI to be granted access.

6.2 Data policy (contigency/disaster plan):

Users are required to keep lightweighted files (codes, scripts,...) needed to regenerate datasets (numerical simulations or other products) under version control (git, svn,...) using a cloud service of their choice.

No confidential, sensitive or personal data can be stored on the group server. All data stored on the group's server is considered public.

Users are required to keep their big files (>50 Mo) on the Google Drive storage provided by Columbia.

All data stored on the group servers is considered as disposable and reproducible in case of any event leading to the destruction of the present data.

6.3 Server maintenance:

Server maintenance is scheduled weekly (every Tuesday mornings) but can happen more often if needed.

The system administrator is required to keep an inventory of all software installed (as a batch install script) as well as any information useful to reproduce the current state of the system.

CHAPTER

SEVEN

TUTORIAL: DISCOVERING, LOADING AND PLOTTING DATA WITH INTAKE AND XARRAY

We need to have intake installed (see https://server-howto.readthedocs.io/en/latest/data_organization.html#discover-data-with-intake).

[1]:
```
import intake
```

Display figures inside notebook

[2]:
```
%matplotlib inline
```

The observation datasets interpolated on the ASTE grid are listed in the following catalog:

[3]:
```
cat = intake.Catalog('/local/data/artemis/catalogs/ASTE_catalog.yml')
```

You can see what datasets are available in the catalog with:

[4]:
```
list(cat)
```

[4]:
```
['MLD_deBoyerMontegut',
 'Biomes_FayMcKinley',
 'NPP_CBPM_MODIS',
 'NPP_CBPM_SeaWIFS',
 'NPP_CBPM_VIIRS',
 'NPP_EppleyVGPM_MODIS',
 'NPP_EppleyVGPM_SeaWIFS',
 'NPP_EppleyVGPM_VIIRS']
```

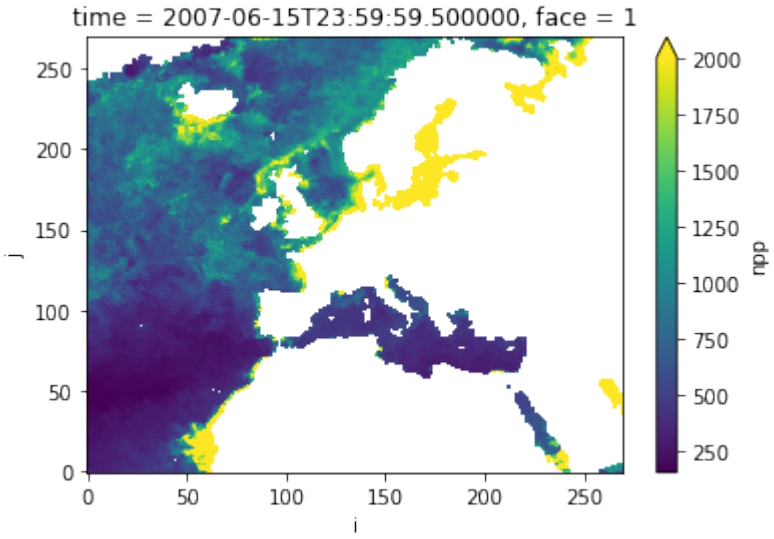
The intake catalog provides all the necessary information on the data location, type,... so the loading process can be done easily with:

[5]:
```
npp_obs = cat.NPP_EppleyVGPM_MODIS.to_dask()
```

The dataset obtained can be plotted using xarray functions.

[6]:
```
npp_obs['npp'].sel(time='2007-6', face=1).plot(vmax=2000)
```

[6]:
```
<matplotlib.collections.QuadMesh at 0x7f2fdc06a438>
```


time = 2007-06-15T23:59:59.500000, face = 1
j
i
0
50
100
150
200
250
2000
1750
1500
1250
1000
750
500
250
npp

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search