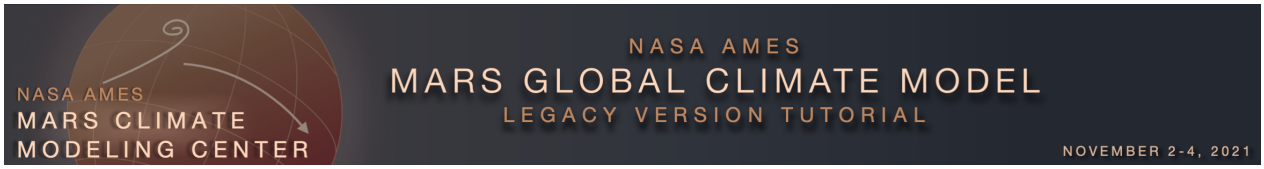

Installing the Community Analysis Pipeline (CAP)

Welcome!

This document contains the instructions for installing the NASA Ames MCMC's Community Analysis Pipeline (CAP). **We ask that you come to the MGCM Tutorial on November 2-4 with CAP installed on your machine** so that we can jump right into using it! On the second day of the tutorial (November 3rd), we will be using CAP to analyze MGCM output.

Installing CAP is fairly straightforward. We will create a Python virtual environment, download CAP, and then install CAP in the virtual environment. That's it!

A quick overview of what is covered in this installation document:

1. Creating the Virtual Environment

We begin by creating a virtual environment in which to install CAP. The virtual environment is an isolated Python environment cloned from an existing Python distribution. The virtual environment consists of the same directory trees as the original environment, but it includes activation and deactivation scripts that are used to move in and out of the virtual environment. Here's an illustration of how the two Python environments might differ:

```
      anaconda3                       virtual_env3/
      ├── bin                         ├── bin
      │   ├── pip       (copy)        │   ├── pip
      │   └── python3    >>>>         │   ├── python3
      └── lib                         │   ├── activate
                                      │   ├── activate.csh
                                      │   └── deactivate
                                      └── lib

   ORIGINAL ENVIRONMENT            VIRTUAL ENVIRONMENT
       (untouched)              (vanishes when deactivated)
```

We can install and upgrade packages in the virtual environment without breaking the main Python environment. In fact, it is safe to change or even completely delete the virtual environment without breaking the main distribution. This allows us to experiment freely in the virtual environment, making it the perfect location for installing and testing CAP.

Step 1: Identify Your Preferred Python Distribution

If you are already comfortable with Python's package management system, you are welcome to install the pipeline on top any python**3** distribution already present on your computer. Jump to Step #2 and resolve any missing package dependency.

For all other users, we highly recommend using the latest version of the Anaconda Python distribution. It ships with pre-compiled math and plotting packages such as `numpy` and `matplotlib` as well as pre-compiled libraries like `hdf5` headers for reading `netCDF` files (the preferred filetype for analysing MGCM output).

You can install the Anaconda Python distribution via the command-line or using a graphical interface (scroll to the very bottom of the page for all download options). You can install Anaconda at either the `System/` level or the `User/` level (the later does not require admin-priviledges). The instructions below are for the **command-line installation** and installs Anaconda in your **home directory**, which is the recommended location. Open a terminal and type the following:

```
(local)>$ chmod +x Anaconda3-2021.05-MacOSX-x86_64.sh   # make the .sh file executable (actual name may diffe
(local)>$ ./Anaconda3-2021.05MacOSX-x86_64.sh           # runs the executable
```

Which will return:

```
> Welcome to Anaconda3 2021.05
>
> In order to continue the installation process, please review the license agreement.
> Please, press ENTER to continue
> >>>
```

Read (`ENTER`) and accept (`yes`) the terms, choose your installation location, and initialize Anaconda3:

```
(local)>$ [ENTER]
> Do you accept the license terms? [yes|no]
> >>>
(local)>$ yes
> Anaconda3 will now be installed into this location:
> /Users/username/anaconda3
>
>  - Press ENTER to confirm the location
>  - Press CTRL-C to abort the installation
>  - Or specify a different location below
>
> [/Users/username/anaconda3] >>>
(local)>$ [ENTER]
> PREFIX=/Users/username/anaconda3
> Unpacking payload ...
> Collecting package metadata (current_repodata.json):
>   done
> Solving environment: done
>
> ## Package Plan ##
> ...
> Preparing transaction: done
> Executing transaction: -
> done
> installation finished.
> Do you wish the installer to initialize Anaconda3 by running conda init? [yes|no]
> [yes] >>>
(local)>$ yes
```

> For Windows users, we recommend installing the pipeline in a Linux-type environment using Cygwin. This will enable the use of CAP command line tools. Simply download the Windows version of Anaconda on the Anaconda website and follow the instructions from the installation GUI. When asked about the installation location, make sure you install Python under your emulated-Linux home directory (`/home/username`) and ***not*** in the default location (`/cygdrive/c/Users/username/anaconda3`). From the installation GUI, the path you want to select is something like:

> `C:/Program Files/cygwin64/home/username/anaconda3`. Also be sure to check **YES** when prompted to "Add Anaconda to my `PATH` environment variable."

Confirm that your path to the Anaconda Python distribution is fully actualized by closing out of the current terminal, opening a new terminal, and typing:

```
(local)>$ python[TAB]
```

If this returns multiple options (e.g. `python`, `python2`, `python 3.7`, `python.exe`), then you have more than one version of Python sitting on your system (an old `python2` executable located in `/usr/local/bin/python`, for example). You can see what these versions are by typing:

```
(local)>$ python3 --version     # Linux/MacOS
(local)>$ python.exe --version  # Cygwin/Windows
```

Check your version of `pip` the same way, then find and set your `$PATH` environment variable to point to the Anaconda Python *and* Anaconda pip distributions. If you are planning to use Python for other projects, you can update these paths like so:

```
# with bash:
(local)>$ echo 'export PATH=/Users/username/anaconda3/bin:$PATH' >> ~/.bash_profile
# with csh/tsch:
(local)>$ echo 'setenv PATH $PATH\:/Users/username/anaconda3/bin\:$HOME/bin\:.'  >> ~/.cshrc
```

Confirm these settings using the `which` command:

```
(local)>$ which python3         # Linux/MacOS
(local)>$ which python.exe      # Cygwin/Windows
```

which hopefully returns a Python executable that looks like **it was installed with Anaconda**, such as:

```
> /username/anaconda3/bin/python3     # Linux/MacOS
> /username/anaconda3/python.exe      # Cygwin/Windows
```

If `which` points to either of those locations, you are good to go and you can proceed from here using the shorthand path to your Anaconda Python distribution:

```
(local)>$ python3     # Linux/MacOS
(local)>$ python.exe  # Cygwin/Windows
```

If, however, `which` points to some other location, such as `/usr/local/bin/python`, or more than one location, proceed from here using the **full** path to the Anaconda Python distribution:

```
(local)>$ /username/anaconda3/bin/python3 # Linux/MacOS
(local)>$ /username/anaconda3/python.exe  # Cygwin/Windows
```

Step 2: Set Up the Virtual Environment:

Python virtual environments are created from the command line. Create an environment called `amesGCM3` by typing:

```
(local)>$ python3 -m venv --system-site-packages amesGCM3    # Linux/MacOS Use FULL PATH to python if needed
(local)>$ python.exe -m venv --system-site-packages amesGCM3  # Cygwin/Windows Use FULL PATH to python if nee
```

First, find out if your terminal is using *bash* or a variation of *C-shell* (*.csh*, *.tsch*…) by typing:

```
(local)>$ echo $0
> -bash
```

Depending on the answer, you can now activate the virtual environment with one of the options below:

```
(local)>$ source amesGCM3/bin/activate          # bash
(local)>$ source amesGCM3/bin/activate.csh      # csh/tcsh
(local)>$ source amesGCM3/Scripts/activate.csh  # Cygwin/Windows
```

> In Cygwin/Windows, the `/bin` directory may be named `/Scripts`.

You will notice that after sourcing `amesGCM3`, your prompt changed indicate that you are now *inside* the virtual environment (i.e. `(local)>$` changed to `(amesGCM3)>$`).

We can verify that `which python` and `which pip` unambiguously point to `amesGCM3/bin/python3` and `amesGCM3/bin/pip`, respectively, by calling `which` within the virtual environment:

```
(amesGCM3)>$ which python3         # in bash, csh
> amesGCM3/bin/python3
(amesGCM3)>$ which pip
> amesGCM3/bin/pip

(amesGCM3)>$ which python.exe      # in Cygwin/Windows
> amesGCM3/Scripts/python.exe
(amesGCM3)>$ which pip
> amesGCM3/Scripts/pip
```

There is therefore no need to reference the full paths while **inside** the virtual environment.

---

2. Installing CAP

Now we can download and install CAP in `amesGCM3`. CAP was provided to you in the tarfile `amesgcm-master.zip` that was sent along with these instructions. Download `amesgcm-master.zip` and leave it in `Downloads/`.

Using `pip`

Open a terminal window, activate the virtual environment, and untar the file:

```
(local)>$ source ~/amesGCM3/bin/activate          # bash
(local)>$ source ~/amesGCM3/bin/activate.csh      # cshr/tsch
(local)>$ source ~/amesGCM3/Scripts/activate.csh  #  Cygwin/Windows
(amesGCM3)>$
(amesGCM3)>$ tar -xf amesgcm-master.zip
(amesGCM3)>$ cd amesgcm-master
(amesGCM3)>$ pip install .
```

> Please follow the instructions to upgrade pip if recommended during that steps.

That's it! CAP is installed in `amesGCM3` and you can see the `MarsXXXX.py` executables stored in `~/amesGCM3/bin/` :

```
(local)>$ ls ~/amesGCM3/bin/
> Activate.ps1     MarsPull.py      activate.csh             nc4tonc3         pip3
> MarsFiles.py     MarsVars.py      activate.fish            ncinfo           pip3.8
> MarsInterp.py    MarsViewer.py    easy_install             normalizer       python
> MarsPlot.py      activate         easy_install-3.8         pip              python3
```

> Shall you need to modify any code, note that when you access the `Mars` tools above, those are **not** executed from the `amesgcm-master/` folder in your `/Downloads` directory, but instead from the `amesGCM3` virtual environment where they were installed by pip. You can safely move amesgcm-master.zip and the amesgcm-master directory to a different location on your system.

Double check that the paths to the executables are correctly set in your terminal by exiting the virtual environment:

```
(amesGCM3)>$ deactivate
```

then reactivating the virtual environment:

```
(local)>$ source ~/amesGCM3/bin/activate     # bash
(local)>$ source ~/amesGCM3/bin/activate.csh # csh/tsch
(local)>$ source ~/amesGCM3/Scripts/activate.csh
```

and checking the documentation for any CAP executable using the `--help` option:

```
(amesGCM3)>$ MarsPlot.py --help
(amesGCM3)>$ MarsPlot.py -h
```

or using **full** paths:

```
(amesGCM3)>$ ~/amesGCM3/bin/MarsPlot.py -h     # Linux/MacOS
(amesGCM3)>$ ~/amesGCM3/Scripts/MarsPlot.py -h # Cygwin/Windows
```

If the pipeline is installed correctly, `--help` will display documentation and command-line arguments for `MarsPlot` in the terminal.

This completes the one-time installation of CAP in your virtual environment, `amesGCM3` , which now looks like:

```
amesGCM3/
├── bin
│   ├── MarsFiles.py
│   ├── MarsInterp.py
│   ├── MarsPlot.py
│   ├── MarsPull.py
│   ├── MarsVars.py
│   ├── activate
│   ├── activate.csh
│   ├── deactivate
│   ├── pip
```

```
│   └── python3
├── lib
│   └── python3.7
│       └── site-packages
│           ├── netCDF4
│           └── amesgcm
│               ├── FV3_utils.py
│               ├── Ncdf_wrapper.py
│               └── Script_utils.py
├── mars_data
│   └── Legacy.fixed.nc
└── mars_templates
    ├──amesgcm_profile
    └── legacy.in
```

Using `conda`

If you prefer using the `conda` package manager for setting up your virtual environment instead of `pip`, you may use the following commands to install CAP.

First, verify (using `conda info` or `which conda`) that you are using the intented `conda` executable (two or more versions of `conda` might be present if both Python2 and Python3 are installed on your system). Then, create the virtual environment with:

```
(local)>$ conda create -n amesGCM3
```

Activate the virtual environment, then install CAP:

```
(local)>$ conda activate amesGCM3
(amesGCM3)>$ conda install pip
(amesGCM3)>$ cd ~/Downloads
(amesGCM3)>$ tar -xf CAP_tarball.zip
(amesGCM3)>$ cd amesgcm-master
(amesGCM3)>$ pip install .
```

The source code will be installed in:

```
/path/to/anaconda3/envs/amesGCM3/
```

and the virtual environment may be activated and deactivated with `conda`:

```
(local)>$ conda activate amesGCM3
(amesGCM3)>$ conda deactivate
(local)>$
```

> **Note:** CAP requires the following Python packages, which were automatically installed with CAP:
>
> ```
> matplotlib        # the MatPlotLib plotting library
> numpy             # math library
> scipy             # math library and input/output for fortran binaries
> netCDF4 Python    # handling netCDF files
> requests          # downloading GCM output from the MCMC Data Portal
> ```

Removing CAP

To permanently remove CAP, activate the virtual environment and run the `uninstall` command:

```
(local)>$ source amesGCM3/bin/activate          # bash
(local)>$ source amesGCM3/bin/activate.csh      # csh/tcsh
(local)>$ source amesGCM3/Scripts/activate.csh  # Cygwin/Windows
(amesGCM3)>$ pip uninstall amesgcm
```

You may also delete the `amesGCM3` virtual environment directory at any time. This will uninstall CAP, remove the virtual environment from your machine, and will not affect your main Python distribution.

---

3. Testing & Using CAP

Whenever you want to use CAP, simply activate the virtual environment and all of CAP's executables will be accessible from the command line:

```
(local)>$ source amesGCM3/bin/activate          #   bash
(local)>$ source amesGCM3/bin/activate.csh      #   csh/tcsh
(local)>$ source amesGCM3/Scripts/activate.csh  #   Cygwin/Windows
```

You can check that the tools are installed properly by typing `Mars` and then pressing the **TAB** key. No matter where you are on your system, you should see the following pop up:

```
(amesGCM3)>$ Mars[TAB]
> MarsFiles.py   MarsInterp.py  MarsPlot.py    MarsPull.py    MarsVars.py
```

If no executables show up then the paths have not been properly set in the virtual environment. You can either use the full paths to the executables:

```
(amesGCM3)>$ ~/amesGCM3/bin/MarsPlot.py
```

Or set up aliases in your `./bashrc` or `.cshrc` :

```
# with bash:
(local)>$ echo alias MarsPlot='/Users/username/amesGCM3/bin/MarsPlot.py' >> ~/.bashrc
(local)>$ source ~/.bashrc

# with csh/tsch
(local)>$ echo alias MarsPlot /username/amesGCM3/bin/MarsPlot >> ~/.cshrc
(local)>$ source ~/.cshrc
```

---

4. Practical Tips for Later Use During the Tutorial

Install `ghostscript` to Create Multiple-Page PDFs When Using `MarsPlot`

Installing `ghostscript` on your local machine allows CAP to generate a multiple-page PDF file instead of several individual PNGs when creating several plots. Without `ghostcript`, CAP defaults to generating multiple `.png` files instead of a single PDF file, and we therefore strongly recommend installing `ghostscript` to streamline the plotting process.

First, check whether you already have `ghostscript` on your machine. Open a terminal and type:

```
(local)>$ gs -version
> GPL Ghostscript 9.54.0 (2021-03-30)
> Copyright (C) 2021 Artifex Software, Inc.  All rights reserved.
```

If `ghostscript` is not installed, follow the directions on the `ghostscript` website to install it.

> If `gs -version` returns a 'command not found error' but you are able to locate the `gs` executable on your system (e.g. /opt/local/bin/gs) you may need to add that specific directory (e.g. /opt/local/bin/) to your search $PATH as done for Python and pip in Step 1

Enable Syntax Highlighting for the Plot Template

The `MarsPlot` executable requires an input template with the `.in` file extension. We recommend using a text editor that provides language-specific (Python) syntax highlighting to make keywords more readable. A few options include: Atom and vim (compatible with MacOS, Windows, Linux), notepad++ (compatible with Windows), or gedit (compatible with Linux).

The most commonly used text editor is vim. Enabling proper syntax-highlighting for Python in **vim** can be done by adding the following lines to `~/.vimrc`:

```
syntax on
colorscheme default
au BufReadPost *.in  set syntax=python
```

---

5. Do This Before Attending the Tutorial

In order to follow along with the practical part of the MGCM Tutorial, we ask that you **download several MGCM output files beforehand**. You should save these on the machine you'll be using during the tutorial.

We'll use CAP to retrieve these files from the MGCM Data Portal. To begin, activate the virtual environment:

```
(local)>$ source amesGCM3/bin/activate      # bash
(local)>$ source amesGCM3/bin/activate.csh  # csh/tcsh
```

Choose a directory in which to store these MGCM output files on your machine. We will also create two sub- directories, one for an MGCM simulation with radiatively inert clouds (RIC) and one for an MGCM simulation with radiatively active clouds (RAC):

```
(amesGCM3)>$ mkdir CAP_tutorial
(amesGCM3)>$ cd CAP_tutorial
(amesGCM3)>$ mkdir INERTCLDS ACTIVECLDS
```

Then, download the corresponding data in each directory:

```
(amesGCM3)>$ cd INERTCLDS
(amesGCM3)>$ MarsPull.py -id INERTCLDS -ls 255 285
```

```
(amesGCM3)>$ cd ../ACTIVECLDS
(amesGCM3)>$ MarsPull.py -id ACTIVECLDS -ls 255 285
```

Finally, check for files integrity using the `disk use` command:

```
cd ..
du -h INERTCLDS/fort.11*
du -h ACTIVECLDS/fort.11*
> 433M    fort.11_0719
[...]
```

The files should be 433Mb each. That's it! `CAP_tutorial` now holds the necessary `fort.11` files from the radiatively active and inert MGCM simulations:

```
CAP_tutorial/
├── INERTCLDS/
│   └── fort.11_0719  fort.11_0720  fort.11_0721  fort.11_0722  fort.11_0723
└── ACTIVECLDS/
    └── fort.11_0719  fort.11_0720  fort.11_0721  fort.11_0722  fort.11_0723
```

You can now deactivate the virtual environment:

```
(amesGCM3)>$ deactivate
```

> If you encounter an issue during the download process or if the files are not 433Mb, please verify the files availability on the MCMC Data Portal and try again later. You can re-attempt to download specific files as follows: `MarsPull.py -id ACTIVECLDS -f fort.11_0720 fort.11_0723` (make sure to navigate to the appropriate simulation directory first), or simply download the 10 files listed above manually from the website.

and we'll see you **November 2, 2021** for the tutorial!