Using Python in climate data analysis (and plotting using NCL)

Baird Langenbrunner
AOS 218
Oct. 24th, 2013

Outline

1. Python – what is it, why use it?

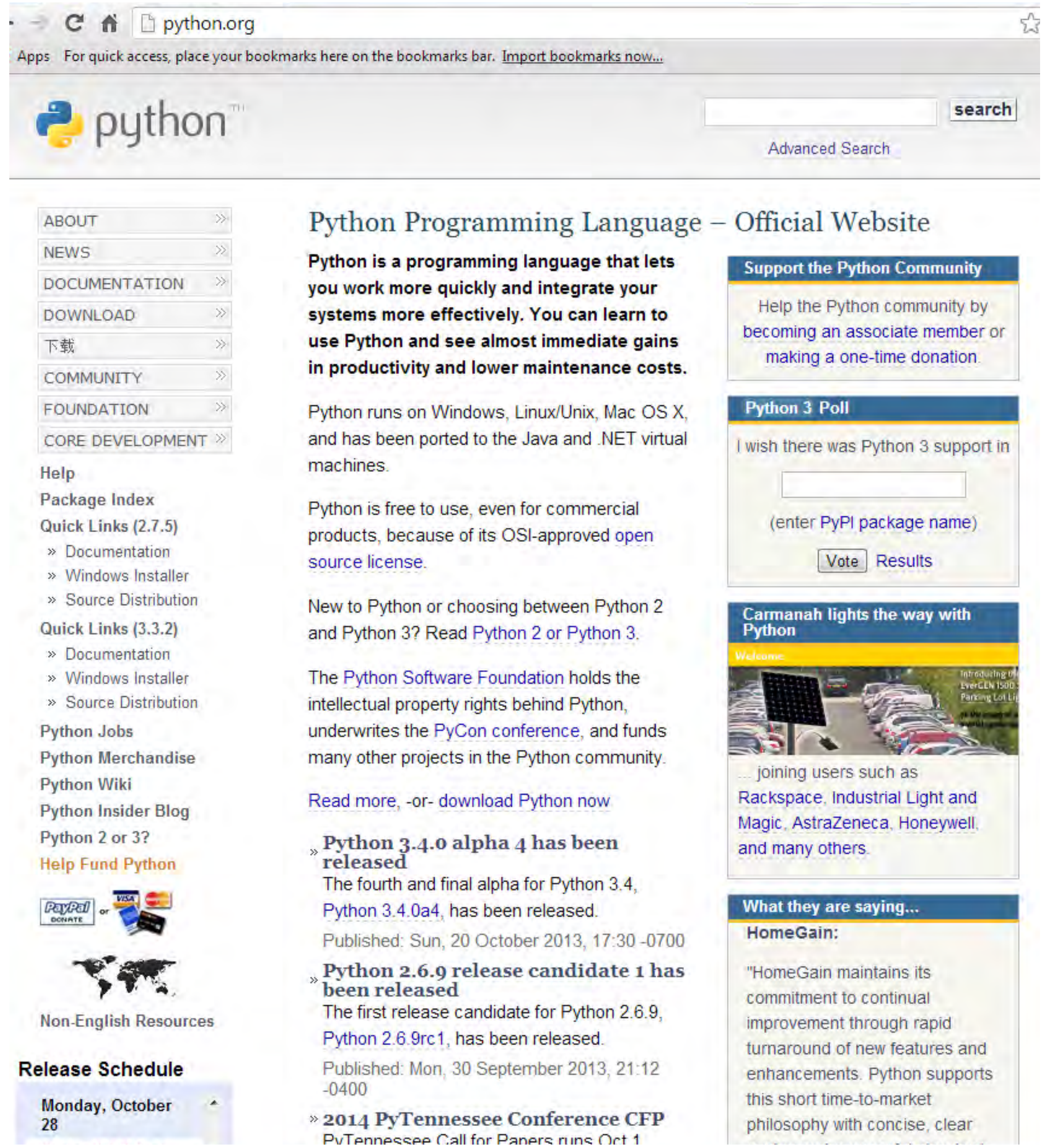
a) Intro to Python and the SciPy "ecosystem" (SciPy *library*, NumPy, Matplotlib, etc.)
b) UVCDAT, pyngl, pyclim, geopy - extra stuff
c) NCL for plotting NetCDF files

2. Model uncertainties in climate change projections

a) Intermodel disagreement/uncertainty on future projections; Knutti and Sedláček, 2012
b) US West Coast precipitation change as a hotbed of model disagreement; storm tracks as possible cause

Python

- Programming language that began in the 1980s
- Conceived by Guido van Rossum, first official released was in 1989
- Philosophy emphasizes code readability
- Free, open source, runs on Windows, Linux/Unix, Mac
- Comes preinstalled on Mac OS X



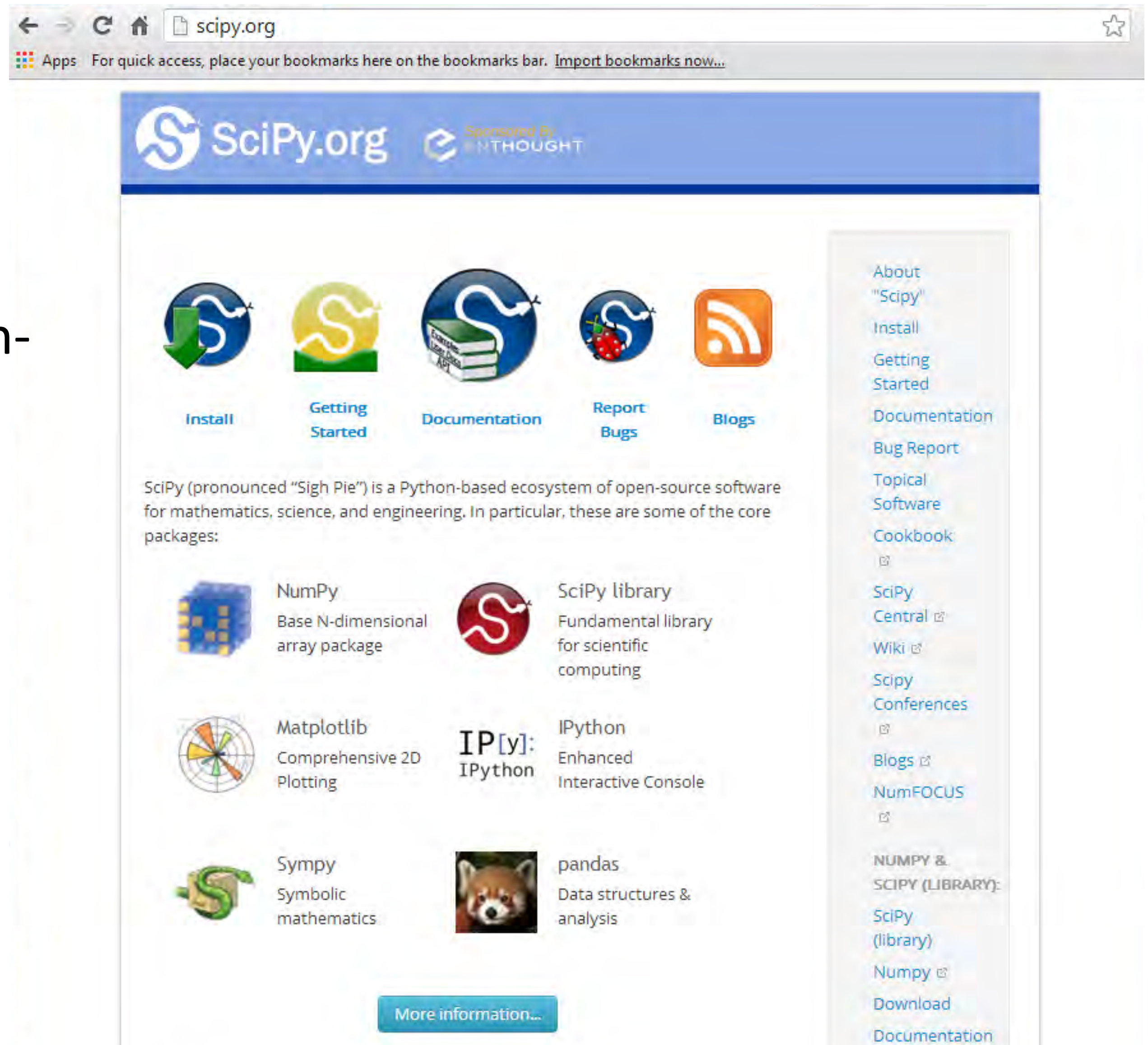
The “Zen of Python”

The Zen of Python

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

SciPy and the "SciPy Stack"

- SciPy = "scientific python"
- The stack is a collection of open-source software and scientific computing tools for Python



SciPy and the "SciPy Stack"

- the SciPy Stack:
  - **(Python)**
  - **SciPy *library***
  - **NumPy**
  - **Matplotlib**
  - *IPython, pandas, SymPy, nose*

- Python – the basic language on which the rest of the stack is built
- SciPy *library* – the package that provides high-end statistics and linear algebra functions; uses NumPy
- NumPy – "numerical python" – the package that gives Python the capability of handling large arrays and performing quick/meaty calculations
- Matplotlib – "mathematical plotting library" – what you use if you want to plot figures within Python

Python/SciPy packages for data analysis

Data input/output

- NumPy input/output module ("import numpy.io")
  - "npz", text, binary
- SciPy input/output module ("import scipy.io")
  - Read/write/save MATLAB, IDL, and NetCDF files

Analyzing data

- Statistics module ("import scipy.stats")
  - Mean, standard deviation, correlation and covariance, etc.
- Linear algebra module ("import scipy.linalg")
  - SVD, CCA, EOF analysis
  - Other matrix decompositions
- Other SciPy modules for:
  - Fourier transforms, interpolation, optimization, integration, signal processing, etc.

Python/SciPy packages specific to NetCDF files

NetCDF input/output

- "UV-CDAT" (separate download)
- "scipy.io.netcdf" (SciPy input/output module [as on previous slide])
- "netcdf4-python" (separate download)
- "PyNIO" (separate download, from NCAR)

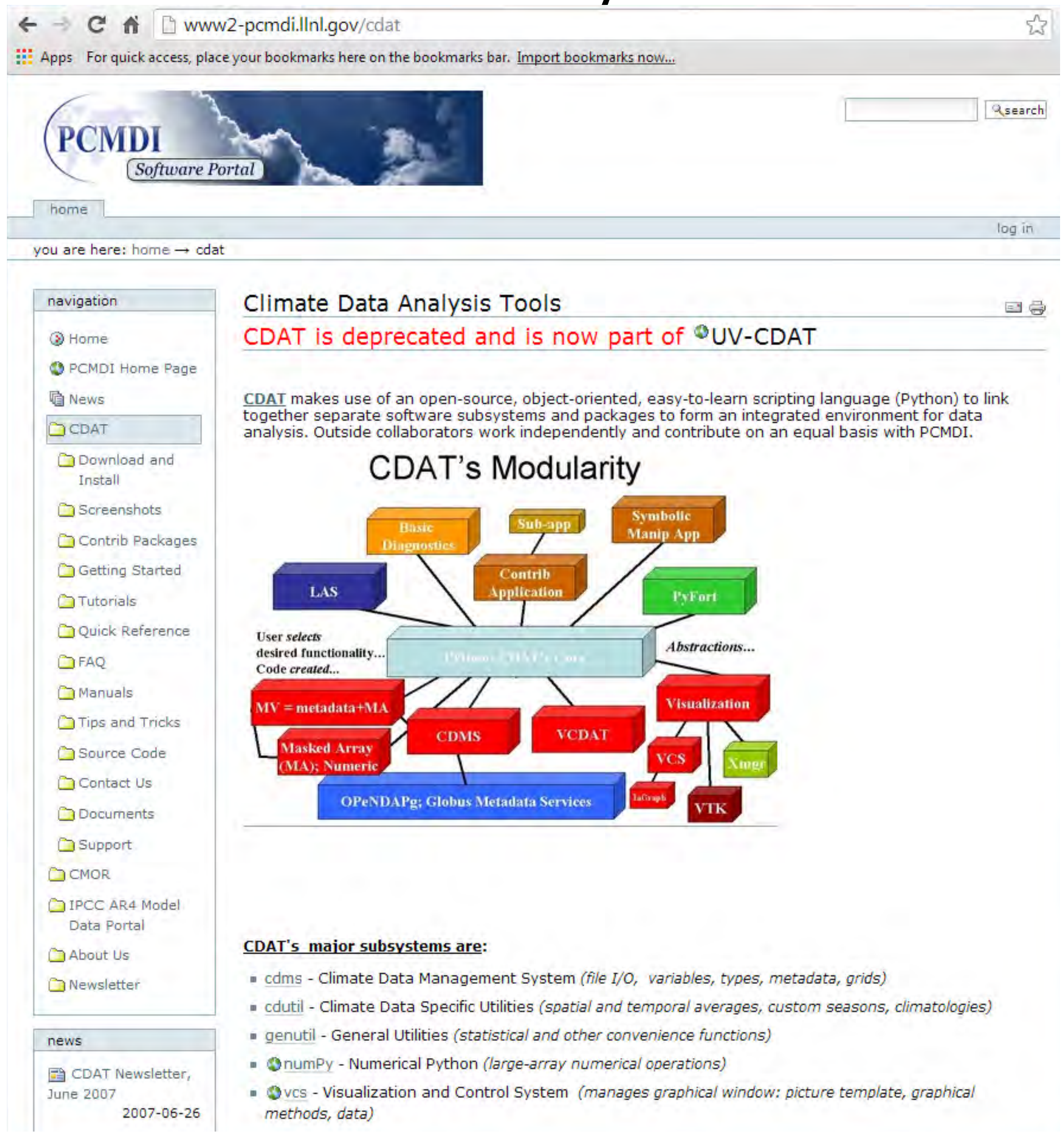
NetCDF plotting

- "UV-CDAT" (separate download)
- "PyNGL" (Python interface to the NCL Graphics Library, from NCAR)
- Matplotlib "basemap" toolkit (requires netcdf4-python)

UV-CDAT

"Ultrascale Visualization – Climate Data Analysis Tools"

- Package for Python that is great for data processing
- Developed by PCMDI at LLNL



UV-CDAT

"Ultrascale Visualization – Climate Data Analysis Tools"

- CDAT is the most important part of UV-CDAT
  - Has 4 pieces:
  - cdms – climate data management system (file i/o, variables, types, etc.)
  - cdutil – climate data specific utilities (spatial/temporal averages, custom seasons, climatologies)
  - genutil – general utilities (some rudimentary statistics)
  - vcs – visualization and control system (graphics/plotting) [deprecated in UV-CDAT…]

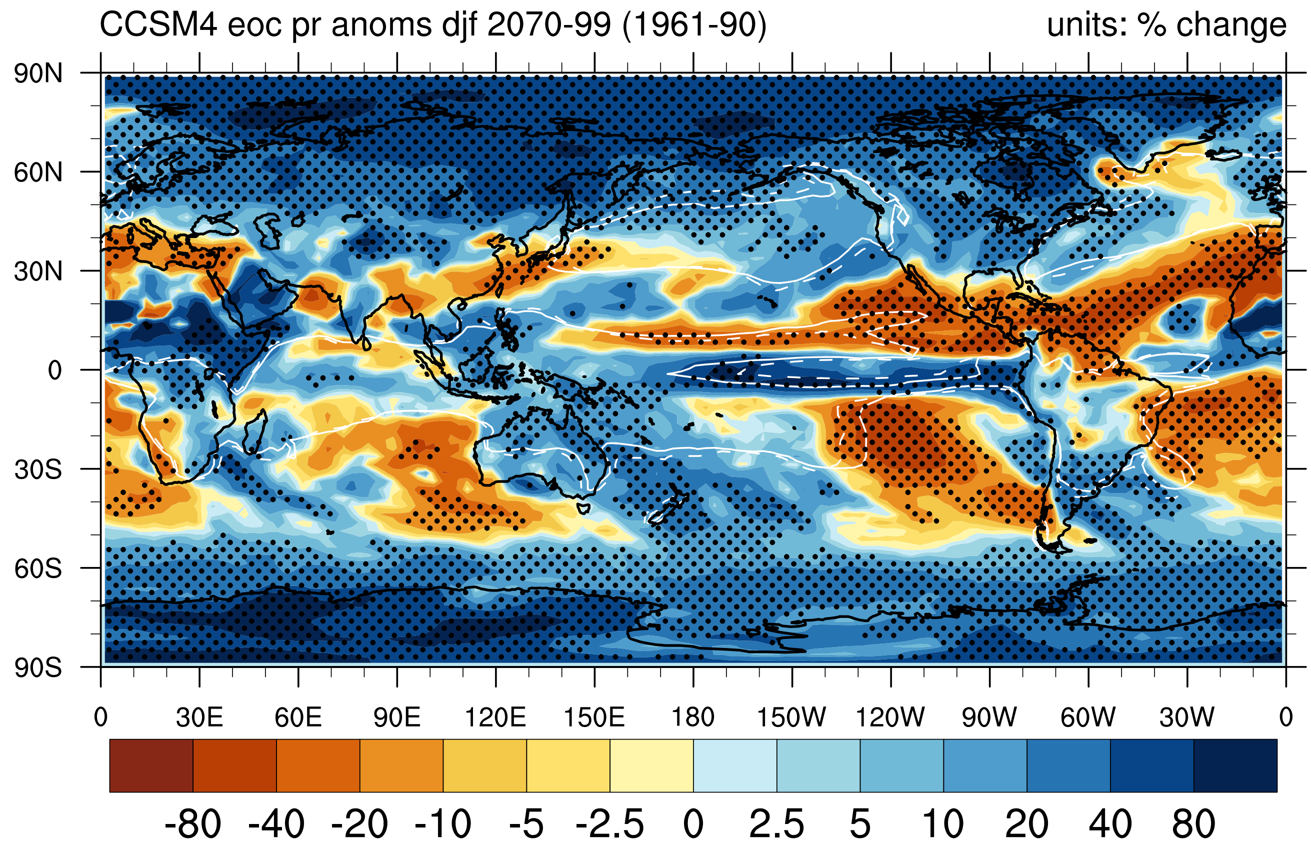
Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

- Plot the end-of-century precipitation change for a given model (CCSM4)
- Plot the rainfall change patterns with "stippling" where they are statistically significant at the 95% confidence level (based on a *t*-test)

Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this



white solid line: 1961-90 4mm/day contour
white dashed line: 2070-99 4mm/day contour

Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

Import the necessary packages into the Python script

class_example_calculating_eoc_statistics.py

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_calculating_eoc_statistics.py

class_example_calculating_eoc_statistics.py (no symbol selected)

```python
#!/usr/local/uvcdat/1.2.0/bin/python2.7

#  type the line below into the terminal shell for cdat to work fully:
#  source /usr/local/uvcdat/1.2.0/bin/setup_cdat.sh", shell=True); import vcs
import cdutil, cdtime
import cdms2 as cdms
import MV2 as MV
import numpy, numpy.linalg
import scipy.stats

########################################################################################
########################################################################################
# CREATE A SAVING FUNCTION FOR NETCDF FILES
########################################################################################
########################################################################################
def save_file(data=None, filename=None, var_id=None):
    if (var_id!=None and filename!=None and var_id!=None):
        save_file = cdms.open(filename, 'w')
        save_file.write(data, id=var_id)
        save_file.close()
        print "================================================================================"
        print "File saved as", filename
        print "================================================================================"
    else:
        print "Specify all variables (data, filename, var_id)"
cdms.setNetcdfShuffleFlag(0)
cdms.setNetcdfDeflateFlag(0)
cdms.setNetcdfDeflateLevelFlag(0)

########################################################################################
########################################################################################
# DEFINE REGIONS FOR ANALYSIS
########################################################################################
########################################################################################
global9090 = cdutil.region.domain(latitude=(-90.,90.), longitude=(0.,360.))
#tropics2525 = cdutil.region.domain(latitude=(-25.,25.), longitude=(0.,360.))
#nino34 = cdutil.region.domain(latitude=(-5.,5.), longitude=(190.,240.)) # nino34 box
#global4545 = cdutil.region.domain(latitude=(-45.,45.), longitude=(0.,360.))

########################################################################################
########################################################################################
# DEFINE SEASONS (NOTE DJF, JJA, MAM, SON ARE ALREADY DEFINED)
########################################################################################
########################################################################################
ONDJFM = cdutil.times.Seasons('ONDJFM')

########################################################################################
########################################################################################
# IMPORT DATA
########################################################################################
########################################################################################
file_example =
    cdms.open('/net/nino/ninod/cmip5/mo/bcc/rcp8.5/run1/pr/pr_185001-209912_r1i1p1_timefixed_2.5regrid.nc')('pr')

model_names = ['CCSM4']

seasons_list = ['djf','mam','jja','son','annual','ondjfm']
season_to_use = 0

# ROOT OF DIRECTORY WHERE MY NETCDF FILES ARE STORED
file_root = '/net/nino/ninod/baird/cmip5/concat_nc_files/pr_regrid_2.5x2.5/'
pr_hist = ['pr_Amon_CCSM4_historical_r1i1p1_185001-200512_2.5x2.5regrid.nc']
pr_rcp85 = ['pr_Amon_CCSM4_rcp85_r1i1p1_200601-210012_2.5x2.5regrid.nc']

########################################################################################
########################################################################################
# DEFINE TIME PERIOD FOR ANALYSIS (END-OF-CENTURY AND BASE PERIOD)
########################################################################################
########################################################################################
eoc_years = numpy.array(([2070,2100]))
base_years = numpy.array(([1961,1991]))

# CONVERT THE TIME PERIODS INTO CDAT-READABLE YEARS
eoc_start_time = cdtime.comptime(eoc_years[0])
eoc_end_time = cdtime.comptime(eoc_years[1])
base_start = cdtime.comptime(base_years[0])
base_end = cdtime.comptime(base_years[1])
```

Line 78 Col 1 | Python | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:46:47 PM | 10.623 / 788 / 211

Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

Create a function that saves data in NetCDF format

class_example_calculating_eoc_statistics.py

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_calculating_eoc_statistics.py

class_example_calculating_eoc_statistics.py (no symbol selected)

```python
#!/usr/local/uvcdat/1.2.0/bin/python2.7

#  type the line below into the terminal shell for cdat to work fully:
#  source /usr/local/uvcdat/1.2.0/bin/setup_cdat.sh", shell=True); import vcs
import cdutil, cdtime
import cdms2 as cdms
import MV2 as MV
import numpy, numpy.linalg
import scipy.stats

##################################################################################
##################################################################################
# CREATE A SAVING FUNCTION FOR NETCDF FILES
##################################################################################
##################################################################################
def save_file(data=None, filename=None, var_id=None):
    if (var_id!=None and filename!=None and var_id!=None):
        save_file = cdms.open(filename, 'w')
        save_file.write(data, id=var_id)
        save_file.close()
        print "=================================================================================="
        print "File saved as", filename
        print "=================================================================================="
    else:
        print "Specify all variables (data, filename, var_id)"
cdms.setNetcdfShuffleFlag(0)
cdms.setNetcdfDeflateFlag(0)
cdms.setNetcdfDeflateLevelFlag(0)

##################################################################################
##################################################################################
# DEFINE REGIONS FOR ANALYSIS
##################################################################################
##################################################################################
global9090 = cdutil.region.domain(latitude=(-90.,90.), longitude=(0.,360.))
#tropics2525 = cdutil.region.domain(latitude=(-25.,25.), longitude=(0.,360.))
#nino34 = cdutil.region.domain(latitude=(-5.,5.), longitude=(190.,240.)) # nino34 box
#global4545 = cdutil.region.domain(latitude=(-45.,45.), longitude=(0.,360.))

##################################################################################
##################################################################################
# DEFINE SEASONS (NOTE DJF, JJA, MAM, SON ARE ALREADY DEFINED)
##################################################################################
##################################################################################
ONDJFM = cdutil.times.Seasons('ONDJFM')

##################################################################################
##################################################################################
# IMPORT DATA
##################################################################################
##################################################################################
file_example =
    cdms.open('/net/nino/ninod/cmip5/mo/bcc/rcp8.5/run1/pr/pr_185001-209912_r1i1p1_timefixed_2.5regrid.nc')('pr')

model_names = ['CCSM4']

seasons_list = ['djf','mam','jja','son','annual','ondjfm']
season_to_use = 0

# ROOT OF DIRECTORY WHERE MY NETCDF FILES ARE STORED
file_root = '/net/nino/ninod/baird/cmip5/concat_nc_files/pr_regrid_2.5x2.5/'
pr_hist = ['pr_Amon_CCSM4_historical_r1i1p1_185001-200512_2.5x2.5regrid.nc']
pr_rcp85 = ['pr_Amon_CCSM4_rcp85_r1i1p1_200601-210012_2.5x2.5regrid.nc']

##################################################################################
##################################################################################
# DEFINE TIME PERIOD FOR ANALYSIS (END-OF-CENTURY AND BASE PERIOD)
##################################################################################
##################################################################################
eoc_years = numpy.array(([2070,2100]))
base_years = numpy.array(([1961,1991]))

# CONVERT THE TIME PERIODS INTO CDAT-READABLE YEARS
eoc_start_time = cdtime.comptime(eoc_years[0])
eoc_end_time = cdtime.comptime(eoc_years[1])
base_start = cdtime.comptime(base_years[0])
base_end = cdtime.comptime(base_years[1])
```

Line 78 Col 1 | Python | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:46:47 PM | 10,623 / 788 / 211

Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

Define seasons (regular seasons are already defined)

Insert the file names that you want to open (here, CCSM4 historical and RCP8.5 monthly data, regridded)

Define your base and end-of-century time periods

class_example_calculating_eoc_statistics.py

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_calculating_eoc_statistics.py

class_example_calculating_eoc_statistics.py (no symbol selected)

```python
#!/usr/local/uvcdat/1.2.0/bin/python2.7

#  type the line below into the terminal shell for cdat to work fully:
#  source /usr/local/uvcdat/1.2.0/bin/setup_cdat.sh", shell=True); import vcs
import cdutil, cdtime
import cdms2 as cdms
import MV2 as MV
import numpy, numpy.linalg
import scipy.stats

################################################################################
################################################################################
# CREATE A SAVING FUNCTION FOR NETCDF FILES
################################################################################
################################################################################
def save_file(data=None, filename=None, var_id=None):
    if (var_id!=None and filename!=None and var_id!=None):
        save_file = cdms.open(filename, 'w')
        save_file.write(data, id=var_id)
        save_file.close()
        print "=========================================================================="
        print "File saved as", filename
        print "=========================================================================="
    else:
        print "Specify all variables (data, filename, var_id)"
cdms.setNetcdfShuffleFlag(0)
cdms.setNetcdfDeflateFlag(0)
cdms.setNetcdfDeflateLevelFlag(0)

################################################################################
################################################################################
# DEFINE REGIONS FOR ANALYSIS
################################################################################
################################################################################
global9090 = cdutil.region.domain(latitude=(-90.,90.), longitude=(0.,360.))
#tropics2525 = cdutil.region.domain(latitude=(-25.,25.), longitude=(0.,360.))
#nino34 = cdutil.region.domain(latitude=(-5.,5.), longitude=(190.,240.)) # nino34 box
#global4545 = cdutil.region.domain(latitude=(-45.,45.), longitude=(0.,360.))

################################################################################
################################################################################
# DEFINE SEASONS (NOTE DJF, JJA, MAM, SON ARE ALREADY DEFINED)
################################################################################
################################################################################
ONDJFM = cdutil.times.Seasons('ONDJFM')

################################################################################
################################################################################
# IMPORT DATA
################################################################################
################################################################################
file_example =
    cdms.open('/net/nino/ninod/cmip5/mo/bcc/rcp8.5/run1/pr/pr_185001-209912_r1i1p1_timefixed_2.5regrid.nc')('pr')

model_names = ['CCSM4']

seasons_list = ['djf','mam','jja','son','annual','ondjfm']
season_to_use = 0

# ROOT OF DIRECTORY WHERE MY NETCDF FILES ARE STORED
file_root = '/net/nino/ninod/baird/cmip5/concat_nc_files/pr_regrid_2.5x2.5/'
pr_hist = ['pr_Amon_CCSM4_historical_r1i1p1_185001-200512_2.5x2.5regrid.nc']
pr_rcp85 = ['pr_Amon_CCSM4_rcp85_r1i1p1_200601-210012_2.5x2.5regrid.nc']

################################################################################
################################################################################
# DEFINE TIME PERIOD FOR ANALYSIS (END-OF-CENTURY AND BASE PERIOD)
################################################################################
################################################################################
eoc_years = numpy.array(([2070,2100]))
base_years = numpy.array(([1961,1991]))

# CONVERT THE TIME PERIODS INTO CDAT-READABLE YEARS
eoc_start_time = cdtime.comptime(eoc_years[0])
eoc_end_time = cdtime.comptime(eoc_years[1])
base_start = cdtime.comptime(base_years[0])
base_end = cdtime.comptime(base_years[1])
```

Line 78 Col 1 | Python | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:46:47 PM | 10.623 / 788 / 211

Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

Create lists to store climatologies, etc.

Open up data, extract the relevant time periods, calculate climatologies, anomalies, *t*-tests on these anomalies

class_example_calculating_eoc_statistics.py

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_calculating_eoc_statistics.py

class_example_calculating_eoc_statistics.py (no symbol selected)

```python


##########################################################################################
##########################################################################################
# THE LINES BELOW OPEN THE MODEL'S HISTORICAL AND RCP8.5 DATA,
# PLACE THE CLIMATOLOGIES, ANOMALIES, PVALS, AND STDEVS INTO LISTS,
# AND THEN SAVES EACH OF THEM AS A SEPARATE NETCDF FILE
##########################################################################################
##########################################################################################
print "Calculating departures from base period for models"

# SPECIFY WHICH SEASON YOU'RE CALCULATING FOR
# THIS STRING WILL GO INTO FILE NAMES LATER
pr_baseclim_list = []
pr_eocclim_list = []
pr_climdepartures_list = []
pr_baseseason_list = []
pr_eocseason_list = []
pr_departures_list = []
base_stdevs_list = []
eoc_stdevs_list = []
pvals_list = []

print "Calculating departures for the", seasons_list[season_to_use], "season"
# SPECIFY WHICH MODELS YOU WANT TO DO THESE CALCULATIONS FOR
which_mods = 0
i = which_mods

print "Opening mod", model_names[i]
pr_eoc = cdms.open(file_root+pr_rcp85[i])('pr')
pr_base = cdms.open(file_root+pr_hist[i])('pr')
# CONVERT TO MM/DAY (DATA ORIGINALLY IN MM/SEC)
pr_eoc = pr_eoc*86400
pr_base = pr_base*86400
pr_eoc = pr_eoc(time=(eoc_start_time, eoc_end_time, 'co'))(global9090)
pr_base = pr_base(time=(base_start, base_end, 'co'))(global9090)
nrows,ncols = pr_eoc.shape[1:3]

pr_baseseason_list.append(cdutil.DJF(pr_base))
pr_eocseason_list.append(cdutil.DJF(pr_eoc))
pr_baseclim_list.append(cdutil.DJF.climatology(pr_base))
pr_eocclim_list.append(cdutil.DJF.climatology(pr_eoc))

pr_departures_list.append(cdutil.DJF.departures(pr_eocseason_list[-1], ref=pr_baseclim_list[-1]))
pr_climdepartures_list.append(cdutil.DJF.departures(pr_eocclim_list[-1], ref=pr_baseclim_list[-1]))




##########################################################################################
# CALCULATE INTERNAL VARIABILITY (STDEVS AT EACH GRIDPOINT FOR BOTH BASE AND END-OF-CENTURY)
##########################################################################################
base_stdevs = numpy.zeros((nrows,ncols))
eoc_stdevs = numpy.zeros((nrows,ncols))
for j in range(nrows):
    for k in range(ncols):
        base_stdevs[j,k] = numpy.std(pr_baseseason_list[-1].data[:,j,k], ddof=1)
        eoc_stdevs[j,k] = numpy.std(pr_eocseason_list[-1].data[:,j,k], ddof=1)
base_stdevs_list.append(base_stdevs)
eoc_stdevs_list.append(eoc_stdevs)

##########################################################################################
# CALCULATE STATISTICAL SIGNIFICANCE OF CHANGE IN MEAN FOR BASE AND EOC PRECIP
# THIS IS A TWO-TAILED T-TEST FOR INDEPENDENT MEANS
##########################################################################################
pvals = numpy.zeros((nrows,ncols))
for j in range(nrows):
    for k in range(ncols):
        pvals[j,k] = scipy.stats.ttest_ind(pr_baseseason_list[-1].data[:,j,k], pr_eocseason_list[-1].data[:,j,k])[1]
pvals_list.append(pvals)

print pr_climdepartures_list[0][0,:,:].shape, "is the shape of the eoc pr anom field"; print "=================="
```

Line 124 Col 1 | Python | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 8:25:29 PM | 10,637 / 790 / 211

Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

Save climatologies, anomalies, and results of *t*-test as separate NetCDF files

class_example_calculating_eoc_statistics.py

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_calculating_eoc_statistics.py

class_example_calculating_eoc_statistics.py (no symbol selected)

```
########################################################################################
# SAVE END-OF-CENTURY PRECIPITATION CHANGES
########################################################################################
model_num = i
saveas = '/net/nino/ninod/baird/cmip5/eoc_calculations/pr/' + model_names[model_num] + '_pr_eoc_anoms_2070-99_1961-90_'
    + seasons_list[season_to_use] + '.nc'
save_file(data=pr_climdepartures_list[-1][0,:,:], filename=saveas, var_id='pr')

########################################################################################
# SAVE BASE PERIOD CLIMATOLOGY (FOR PLOTTING 4MM/DAY CONTOURS)
########################################################################################
model_num = i
saveas = '/net/nino/ninod/baird/cmip5/eoc_calculations/pr/' + model_names[model_num] + '_pr_1961-90_climatology_' +
    seasons_list[season_to_use] + '.nc'
save_file(data=pr_baseclim_list[-1][0,:,:], filename=saveas, var_id='pr')

########################################################################################
# SAVE END-OF-CENTURY PRECIPITATION CLIMATOLOGY
########################################################################################
model_num = i
saveas = '/net/nino/ninod/baird/cmip5/eoc_calculations/pr/' + model_names[model_num] + '_pr_2070-99_climatology_' +
    seasons_list[season_to_use] + '.nc'
save_file(data=pr_eocclim_list[-1][0,:,:], filename=saveas, var_id='pr')

########################################################################################
# SAVE BASE PERIOD STDEVS (MEASURE OF INTERNAL VARIABILITY FOR THAT SEASON) AS NETCDF FILE
########################################################################################
model_num = i
vec = base_stdevs_list[-1]
ID = 'pr'
stdevs_nc = MV.array(vec, typecode='f', id=ID)
stdevs_nc.setAxis(0, file_example.getLatitude())
stdevs_nc.setAxis(1, file_example.getLongitude())
saveas = '/net/nino/ninod/baird/cmip5/eoc_calculations/pr/' + model_names[model_num] + '_pr_1961-90_stdevs_' +
    seasons_list[season_to_use] + '.nc'
save_file(data=stdevs_nc, filename=saveas, var_id='pr')

########################################################################################
# SAVE STDEVS FOR EOC PERIOD AS NETCDF FILE
########################################################################################
model_num = i
vec = eoc_stdevs_list[-1]
ID = 'pr'
stdevs_nc = MV.array(vec, typecode='f', id=ID)
stdevs_nc.setAxis(0, file_example.getLatitude())
stdevs_nc.setAxis(1, file_example.getLongitude())
saveas = '/net/nino/ninod/baird/cmip5/eoc_calculations/pr/' + model_names[model_num] + '_pr_2070-99_stdevs_' +
    seasons_list[season_to_use] + '.nc'
save_file(data=stdevs_nc, filename=saveas, var_id='pr')

########################################################################################
# SAVE P-VALUES FROM T-TEST AS A NETCDF FILE
########################################################################################
model_num = i
vec = pvals_list[-1]
ID = 'pvals'
pvals_nc = MV.array(vec, typecode='f', id=ID)
pvals_nc.setAxis(0, file_example.getLatitude())
pvals_nc.setAxis(1, file_example.getLongitude())
#pvals_nc.mask = pvals_nc>0.05
saveas = '/net/nino/ninod/baird/cmip5/eoc_calculations/pr/' + model_names[model_num] + '_pr_eoc_change_pvals_' +
    seasons_list[season_to_use] + '.nc'
save_file(data=pvals_nc, filename=saveas, var_id='pvals')
```

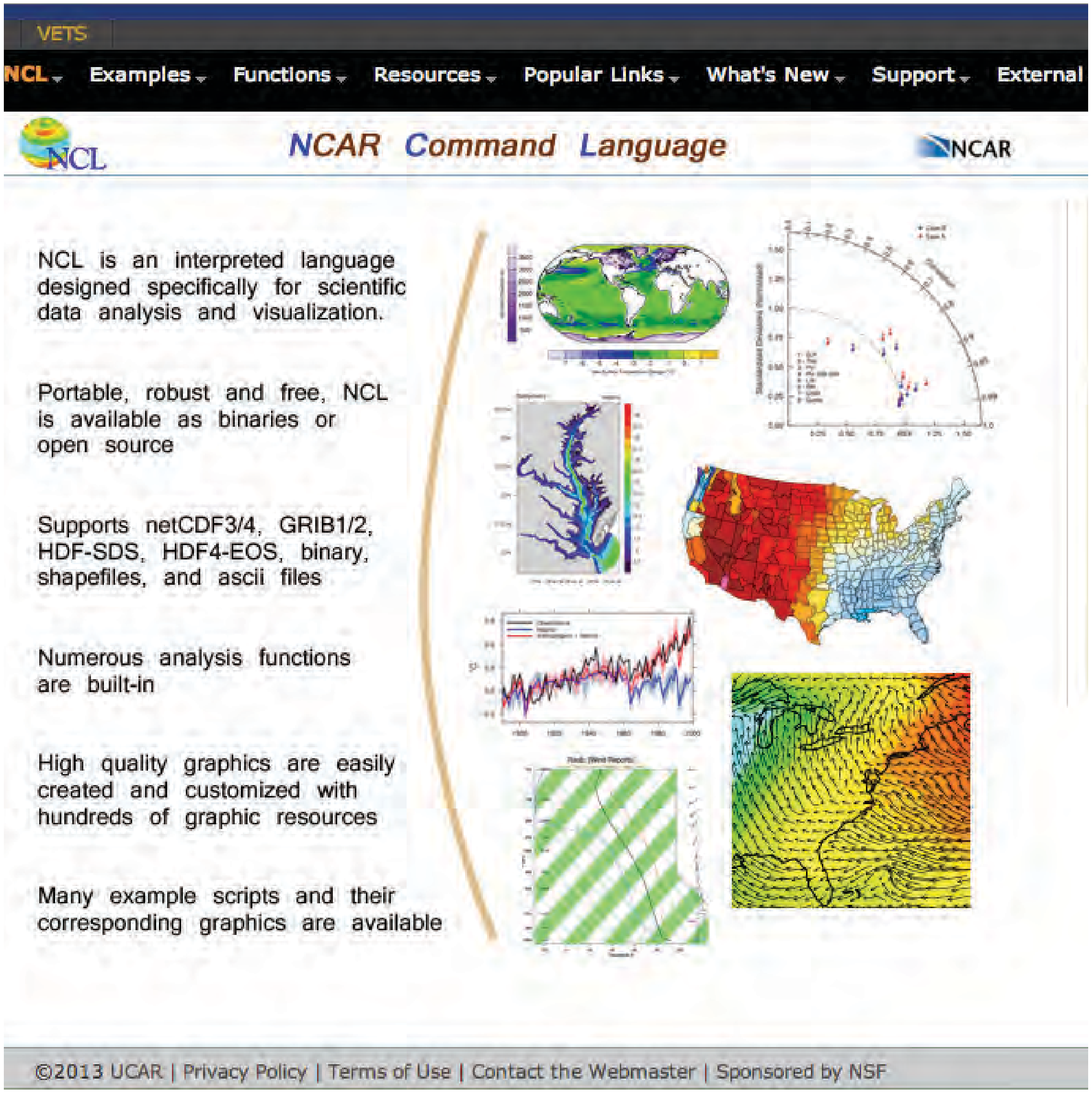
Line 78 Col 1 | Python | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:46:47 PM | 10.623 / 788 / 211

page 3

NCL

now plot your NetCDF file using NCL

- NCL – NCAR Command Language
- Really versatile language; easy/free to download and install, great support community
- Created for "scientific data analysis and visualization"
- Easily reads in NetCDF ".nc" files, handles attributes like latitude and longitude well



NCL

now plot your NetCDF file using NCL

Load NCL code necessary for plotting

Define the file names for what you will be plotting

Open the anomalies, the *t*-test results…

class_example_plotting_pr_change_with_pvals.ncl

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_plotting_pr_change_with_pvals.ncl

```
;-------------------------------------------------------------------------
;   Script for plotting*:
;   (1) End-of-century precipitation changes for a specific model, with an overlay of
;   (2) A dot pattern where these changes are statistically signif. at the 95% confidence interval,
;   (3) The base period climatological 4mm/day precip contour as a solid line (storm tracks in base period), and
;   (4) The end-of-century climatological 4mm/day contour as a dotted line (to see changes in storm tracks)
;
;   * Note all of these are plotted for the DJF season
;
;   Created by Baird Langenbrunner for NCL version 6.0.0
;   October 23, 2013
;   AOS 218 Presentation
;-------------------------------------------------------------------------

load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_csm.ncl"

begin
; OPEN FILE AND READ IN DATA

season = "djf"

model_names = (/"ACCESS1-0","ACCESS1-3","bcc-csm1-1","bcc-csm1-1-m","BNU-ESM",
    "CanESM2","CCSM4","CESM1-BGC","CESM1-CAM5","CMCC-CESM","CMCC-CM","CMCC-CMS","CNRM-CM5",
    "CSIRO-Mk3-6-0","EC-EARTH","FGOALS-g2","GFDL-CM3","GFDL-ESM2G","GFDL-ESM2M","GISS-E2-H",
    "GISS-E2-R","HadGEM2-AO","HadGEM2-CC","HadGEM2-ES","inmcm4","IPSL-CM5A-LR","IPSL-CM5A-MR",
    "IPSL-CM5B-LR","MIROC5","MIROC-ESM","MIROC-ESM-CHEM","MPI-ESM-LR","MPI-ESM-MR","MRI-CGCM3",
    "NorESM1-M","NorESM1-ME"/)

do i=0,35
;-----------------------------------------------------------------------------------------
; SPECIFY THE FILES YOU WANT TO OPEN (NETCDF FILES HERE, PREVIOUSLY SAVED IN PYTHON)
;-----------------------------------------------------------------------------------------
model_num = i
model = model_names(model_num)

file_root = "~/Dropbox/cmip5_data/temp_nc_files/"
filename_pr_base_stdevs = file_root + model + "_pr_1961-90_stdevs_"+season+".nc"
filename_pr_eoc_anoms = file_root + model + "_pr_eoc_anoms_2070-99_1961-90_"+season+".nc"
filename_pr_eoc_pvals = file_root + model + "_pr_eoc_change_pvals_"+season+".nc"
filename_pr_base_clim = file_root + model + "_pr_1961-90_climatology_"+season+".nc"
filename_pr_eoc_clim = file_root + model + "_pr_2070-99_climatology_"+season+".nc"

;-----------------------------------------------------------------------------------------
; OPEN FILES, SPECIFY THE LATITUDES AND LONGITUDES JUST TO BE SAFE
;-----------------------------------------------------------------------------------------
setfileoption("nc", "MissingToFillValue", False)

ncfile1 = addfile(filename_pr_eoc_anoms, "r")
ncfile2 = addfile(filename_pr_eoc_pvals, "r")
ncfile3 = addfile(filename_pr_base_clim, "r")
ncfile4 = addfile(filename_pr_eoc_clim, "r")

; PR END OF CENTURY ANOMALIES
pr_eoc_anoms = ncfile1->pr(:,:)
pr_eoc_anoms@units = ""
lat = ncfile1->lat
lon = ncfile1->lon
pr_eoc_anoms!0 = "lat"
pr_eoc_anoms!1 = "lon"
lat@units = "degrees_north"
lon@units = "degrees_east"
pr_eoc_anoms&lat = lat
pr_eoc_anoms&lon = lon

; PR EOC PVALS FOR TTEST OF INDEPENDENT MEANS
pr_eoc_pvals = ncfile2->pvals(:,:)
pr_eoc_pvals@units = ""
lat = ncfile2->lat
lon = ncfile2->lon
pr_eoc_pvals!0 = "lat"
pr_eoc_pvals!1 = "lon"
lat@units = "degrees_north"
lon@units = "degrees_east"
pr_eoc_pvals&lat = lat
pr_eoc_pvals&lon = lon
```

ine 185 Col 45 | NCAR Command Language | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:50:54 PM | 9.272 / 813 / 227

NCL

now plot your NetCDF file using NCL

… and the base/end-of-century climatologies

Define what you want to save the plot as

Open a work station

Declare the resources for each component of the plot individually (anomalies, the base map, the significance test results, and the end-of-century and base period climatologies)

class_example_plotting_pr_change_with_pvals.ncl

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_plotting_pr_change_with_pvals.ncl

class_example_plotting_pr_change_with_pvals.ncl

```
; PR BASE PERIOD CLIMATOLOGY
pr_base_clim = ncfile3->pr(:,:)
pr_base_clim@units = ""
lat = ncfile3->lat
lon = ncfile3->lon
pr_base_clim!0 = "lat"
pr_base_clim!1 = "lon"
lat@units = "degrees_north"
lon@units = "degrees_east"
pr_base_clim&lat = lat
pr_base_clim&lon = lon

; PR END OF CENTURY CLIMATOLOGY
pr_eoc_clim = ncfile4->pr(:,:)
pr_eoc_clim@units = ""
lat = ncfile4->lat
lon = ncfile4->lon
pr_eoc_clim!0 = "lat"
pr_eoc_clim!1 = "lon"
lat@units = "degrees_north"
lon@units = "degrees_east"
pr_eoc_clim&lat = lat
pr_eoc_clim&lon = lon

; CONVERT EOC ANOMALIES INTO A PERCENT CHANGE BY DIVIDING THEM BY THE MODEL'S BASE PERIOD CLIMATOLOGY
pr_eoc_anoms = (pr_eoc_anoms/pr_base_clim)*100.

;----------------------------------------------------------------------------------------
; SET THE DIRECTORY AND FILE NAME FOR THE FINISHED FIGURE
;----------------------------------------------------------------------------------------
root = "~/Dropbox/plots_raw/eoc_plots/pr/perc_change_with_pvals/"
saveas = root + "pr_eoc_anoms_"+model+"_"+season+"_1961-90_2070-99"

;----------------------------------------------------------------------------------------
; CREATE COLOR BAR AND A "WORK STATION"
;----------------------------------------------------------------------------------------
wks = gsn_open_wks("x11", saveas)
gsn_define_colormap(wks, "BlueYellowRed")
gsn_reverse_colormap(wks)

;----------------------------------------------------------------------------------------
; CREATE THE PLOTTING RESOURCES FOR THE ANOMALIES
;----------------------------------------------------------------------------------------
res_anoms                               = True
res_anoms@gsnDraw                       = False
res_anoms@gsnFrame                      = False
res_anoms@lbLabelBarOn                  = True
res_anoms@cnFillOn                      = False
res_anoms@cnLinesOn                     = False
res_anoms@cnLineLabelsOn                = False
res_anoms@cnFillOn                      = True
res_anoms@cnLinesOn                     = False
res_anoms@cnLineLabelsOn                = False
res_anoms@gsnSpreadColors               = True
res_anoms@lbLabelAutoStride             = True
res_anoms@lbAutoManage                  = False
res_anoms@lbLabelFontHeightF            = 0.015
res_anoms@pmLabelBarOrthogonalPosF      = 0.055

; SET CONTOUR LEVELS FOR PLOT EXPLICITLY
res_anoms@cnLevelSelectionMode          = "ExplicitLevels"
res_anoms@cnLevels                      = (/ -80.,-40.,-20.,-10.,-5.,-2.5,0,2.5,5.,10.,20.,40.,80. /)

;----------------------------------------------------------------------------------------
; CREATE BASE MAP RESOURCES
;----------------------------------------------------------------------------------------
res_map                                 = True
res_map@mpFillOn                        = True
res_map@gsnMaximize                     = True
res_map@gsnPaperOrientation             = "auto"
res_map@mpOutlineOn                     = True
res_map@gsnDraw                         = False
res_map@gsnFrame                        = False
res_map@mpGeophysicalLineThicknessF     = 3.
;; SPECIFY GLOBAL PLOTTING DOMAIN
res_map@mpMinLatF                       = -90.0
res_map@mpMaxLatF                       = 90.0
```

Line 185 Col 45 | NCAR Command Language | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:50:54 PM | 9,272 / 813 / 227

NCL

now plot your NetCDF file using NCL

continued: Declare the resources for each component of the plot individually (anomalies, the base map, the significance test results, and the end-of-century and base period climatologies)

class_example_plotting_pr_change_with_pvals.ncl

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_plotting_pr_change_with_pvals.ncl

```
res_map@mpMaxLatF                     = 90.0
res_map@mpMinLonF                     = 0.
res_map@mpMaxLonF                     = 360.
;; FOR GLOBAL PLOTS
res_map@mpLimitMode                   = "LatLon"
res_map@mpRelativeCenterLat           = True
res_map@mpRelativeCenterLon           = True

;---------------------------------------------------------------------------
; CREATE 4mm/day CLIMATOLOGY LINE FOR BASE PERIOD
;---------------------------------------------------------------------------
res_base_4mm                              = True
res_base_4mm@gsnDraw                      = False
res_base_4mm@gsnFrame                     = False
res_base_4mm@cnLevelSelectionMode         = "ExplicitLevels"
res_base_4mm@cnLevels                     = 4.0
res_base_4mm@cnLineColor                  = "White"
res_base_4mm@cnLineThicknessF             = 2.
res_base_4mm@lbLabelBarOn                 = False
res_base_4mm@cnLineLabelsOn               = False
res_base_4mm@cnInfoLabelOn                = False
res_base_4mm@gsnContourPosLineDashPattern = 0

;---------------------------------------------------------------------------
; CREATE 4mm/day CLIMATOLOGY LINE FOR EOC PERIOD (dotted)
;---------------------------------------------------------------------------
res_eoc_4mm                               = True
res_eoc_4mm@gsnDraw                       = False
res_eoc_4mm@gsnFrame                      = False
res_eoc_4mm@cnLevelSelectionMode          = "ExplicitLevels"
res_eoc_4mm@cnLevels                      = 4.0
res_eoc_4mm@cnLineColor                   = "White"
res_eoc_4mm@cnLineThicknessF              = 2.
res_eoc_4mm@lbLabelBarOn                  = False
res_eoc_4mm@cnLineLabelsOn                = False
res_eoc_4mm@cnInfoLabelOn                 = False
res_eoc_4mm@gsnContourPosLineDashPattern  = 14

;---------------------------------------------------------------------------
; CREATE RESOURCES FOR T-TEST
;---------------------------------------------------------------------------
res_pvals                         = True
res_pvals@gsnDraw                 = False
res_pvals@gsnFrame                = False
res_pvals@cnLinesOn               = False
res_pvals@lbLabelBarOn            = False
res_pvals@cnLineLabelsOn          = False
res_pvals@cnInfoLabelOn           = False
res_pvals@cnLevelSpacingF         = 0.05
res_pvals@cnFillDotSizeF          = 0.004
eoc_pvals_plot                    = gsn_csm_contour(wks,pr_eoc_pvals,res_pvals)
opt_pvals                         = True
opt_pvals@gsnShadeFillType        = "pattern"
opt_pvals@gsnShadeLow             = 17

;---------------------------------------------------------------------------
; INSTANTIATE THE PLOT
;---------------------------------------------------------------------------
datestr = systemfunc("date")

title_leftalign = model+ " eoc pr anoms " +season+ " 2070-99 (1961-90)"
res_map@gsnLeftString           = title_leftalign
res_map@gsnRightString          = "units: % change"

plot_map                    = gsn_csm_map_ce(wks,res_map)
plot_anoms                  = gsn_csm_contour(wks,pr_eoc_anoms,res_anoms)
plot_eoc_pvals              = gsn_contour_shade(eoc_pvals_plot,0.05,999.,opt_pvals)
plot_base_4mm               = gsn_csm_contour(wks,pr_base_clim,res_base_4mm)
plot_eoc_4mm                = gsn_csm_contour(wks,pr_eoc_clim,res_eoc_4mm)
overlay(plot_map, plot_anoms)
overlay(plot_map, plot_base_4mm)
overlay(plot_map, plot_eoc_4mm)
overlay(plot_map, plot_eoc_pvals)
draw(plot_map)
frame(wks)
end do
end
```

Line 223 Col 11 | NCAR Command Language | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:50:54 PM | 9,270 / 813 / 225

NCL

now plot your NetCDF file using NCL

Now plot each component (the map, the anomalies, the p-values, the base and end-of-century 4mm/day contours)

Lastly, overlay them all on top of the map itself

class_example_plotting_pr_change_with_pvals.ncl

File Path ▾ : ~/Google Drive/sy_2013_2014/aos_218/class_example_plotting_pr_change_with_pvals.ncl

```
res_map@mpMaxLatF                        = 90.0
res_map@mpMinLonF                        = 0.
res_map@mpMaxLonF                        = 360.
;; FOR GLOBAL PLOTS
res_map@mpLimitMode                      = "LatLon"
res_map@mpRelativeCenterLat              = True
res_map@mpRelativeCenterLon              = True

;-------------------------------------------------------------------------------
; CREATE 4mm/day CLIMATOLOGY LINE FOR BASE PERIOD
;-------------------------------------------------------------------------------
res_base_4mm                                = True
res_base_4mm@gsnDraw                        = False
res_base_4mm@gsnFrame                       = False
res_base_4mm@cnLevelSelectionMode           = "ExplicitLevels"
res_base_4mm@cnLevels                       = 4.0
res_base_4mm@cnLineColor                    = "White"
res_base_4mm@cnLineThicknessF               = 2.
res_base_4mm@lbLabelBarOn                   = False
res_base_4mm@cnLineLabelsOn                 = False
res_base_4mm@cnInfoLabelOn                  = False
res_base_4mm@gsnContourPosLineDashPattern   = 0

;-------------------------------------------------------------------------------
; CREATE 4mm/day CLIMATOLOGY LINE FOR EOC PERIOD (dotted)
;-------------------------------------------------------------------------------
res_eoc_4mm                                 = True
res_eoc_4mm@gsnDraw                         = False
res_eoc_4mm@gsnFrame                        = False
res_eoc_4mm@cnLevelSelectionMode            = "ExplicitLevels"
res_eoc_4mm@cnLevels                        = 4.0
res_eoc_4mm@cnLineColor                     = "White"
res_eoc_4mm@cnLineThicknessF                = 2.
res_eoc_4mm@lbLabelBarOn                    = False
res_eoc_4mm@cnLineLabelsOn                  = False
res_eoc_4mm@cnInfoLabelOn                   = False
res_eoc_4mm@gsnContourPosLineDashPattern    = 14

;-------------------------------------------------------------------------------
; CREATE RESOURCES FOR T-TEST
;-------------------------------------------------------------------------------
res_pvals                            = True
res_pvals@gsnDraw                    = False
res_pvals@gsnFrame                   = False
res_pvals@cnLinesOn                  = False
res_pvals@lbLabelBarOn               = False
res_pvals@cnLineLabelsOn             = False
res_pvals@cnInfoLabelOn              = False
res_pvals@cnLevelSpacingF            = 0.05
res_pvals@cnFillDotSizeF             = 0.004
eoc_pvals_plot                       = gsn_csm_contour(wks,pr_eoc_pvals,res_pvals)
opt_pvals                            = True
opt_pvals@gsnShadeFillType           = "pattern"
opt_pvals@gsnShadeLow                = 17

;-------------------------------------------------------------------------------
; INSTANTIATE THE PLOT
;-------------------------------------------------------------------------------
datestr = systemfunc("date")

title_leftalign = model+ " eoc pr anoms " +season+ " 2070-99 (1961-90)"
res_map@gsnLeftString             = title_leftalign
res_map@gsnRightString            = "units: % change"

plot_map                   = gsn_csm_map_ce(wks,res_map)
plot_anoms                 = gsn_csm_contour(wks,pr_eoc_anoms,res_anoms)
plot_eoc_pvals             = gsn_contour_shade(eoc_pvals_plot,0.05,999.,opt_pvals)
plot_base_4mm              = gsn_csm_contour(wks,pr_base_clim,res_base_4mm)
plot_eoc_4mm               = gsn_csm_contour(wks,pr_eoc_clim,res_eoc_4mm)
overlay(plot_map, plot_anoms)
overlay(plot_map, plot_base_4mm)
overlay(plot_map, plot_eoc_4mm)
overlay(plot_map, plot_eoc_pvals)
draw(plot_map)
frame(wks)
end do
end
```

Line 223 Col 11 | NCAR Command Language | Unicode (UTF-8) | Unix (LF) | Last saved: 10/23/13 5:50:54 PM | 9,270 / 813 / 225

Downloading and more info

Language/package	Website
Python	http://python.org/
SciPy Stack	http://scipy.org/ (has links to NumPy, Matplotlib, etc.)
UV-CDAT	http://uv-cdat.llnl.gov/
NCL	https://www.earthsystemgrid.org/dataset/ncl.html

- Some of the most useful tips for learning these languages and packages are contained within the "tutorials" on their separate websites. My own advice is always to start with a tutorial and build up from there.

Other helpful links

Python/SciPy/NCL/NCO tips and tricks	
NumPy "for MATLAB users"	http://wiki.scipy.org/NumPy_for_Matlab_Users
Python for the Atmospheric and Oceanic Sciences	http://pyaos.johnny-lin.com/
	- The PyAOS blog mentions that there are beginner, intermediate, and advanced courses on using Python in Climate and Meteorology at the 2014 AMS meeting in Atlanta
Think Python (learn the language)	http://www.greenteapress.com/thinkpython/
Python in Hydrology (learn Python with examples in hydrology)	http://www.greenteapress.com/pythonhydro/pythonhydro.html
Great website with NCO tricks	http://jisao.washington.edu/data/nco/
Other links???	

ERRRGGGHHHHH

troubleshooting

- Ask real people first!
  – Students in AOS use Python/SciPy, NCO, NCL, MATLAB, etc.
  – One of us has potentially already written a script to do what you need
- Or look within stackoverflow.com
  – "Question and answer site for professional and enthusiast programmers"
  – Big Python user base