香港天文台
HONG KONG OBSERVATORY

Practical Session on NWP (2)

WMO VCP Workshop

9 Dec 2020

香港天文台
HONG KONG OBSERVATORY

Learning Objectives

- Source and getting NWP model data (NCEP GFS) to plot forecast chart / prog chart

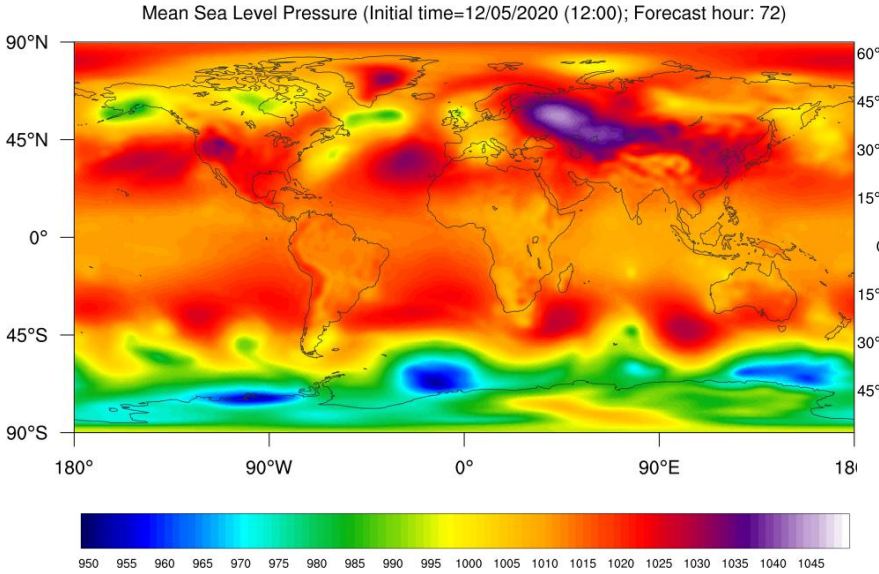
- Exercises on using plotting software (PyNGL) to visualize NWP output

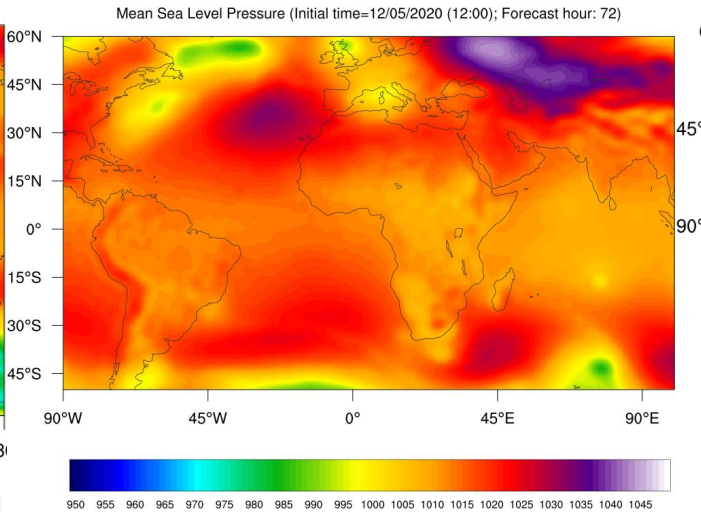
- Assumptions
  - Basic on shell scripts
  - Basic on Python language
  - Editor (vi, emacs or nano)

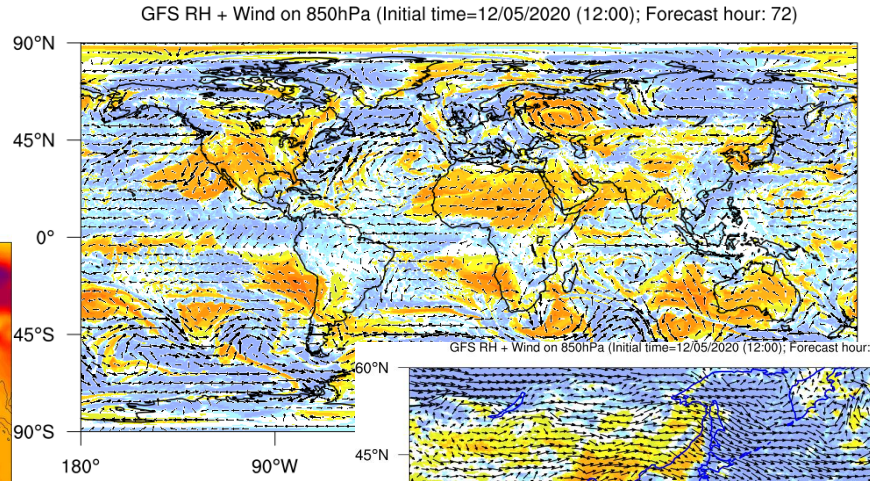
- HKO Team:
  - Wai-Kin Wong
  - Ping Cheung
  - Pluto Chui

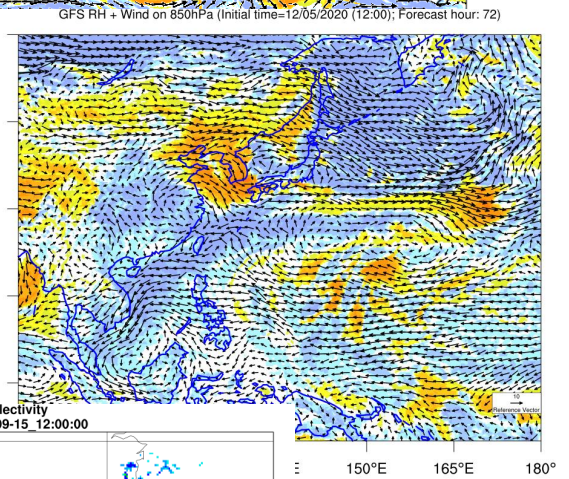
香港天文台
HONG KONG OBSERVATORY

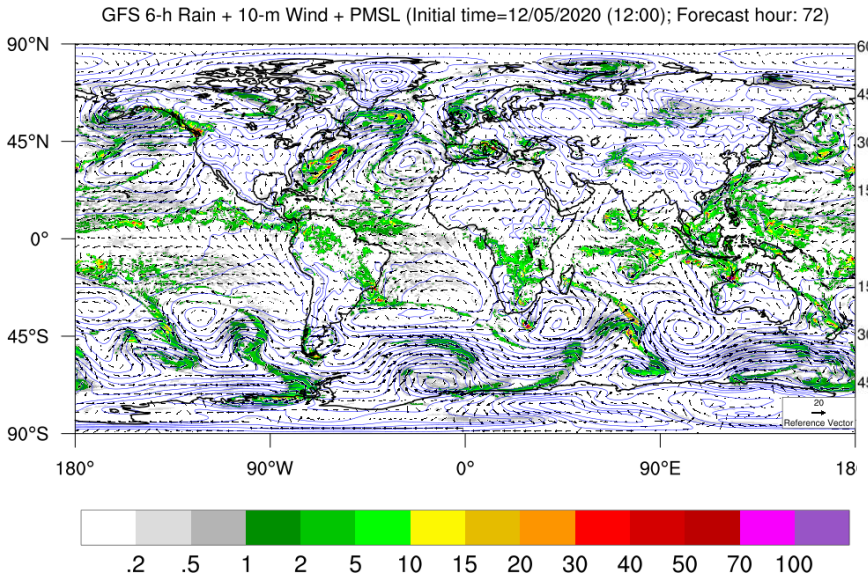
Your outcomes

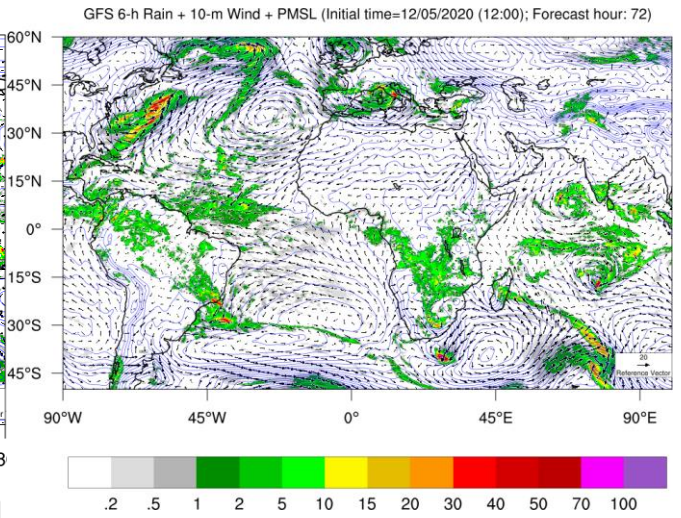


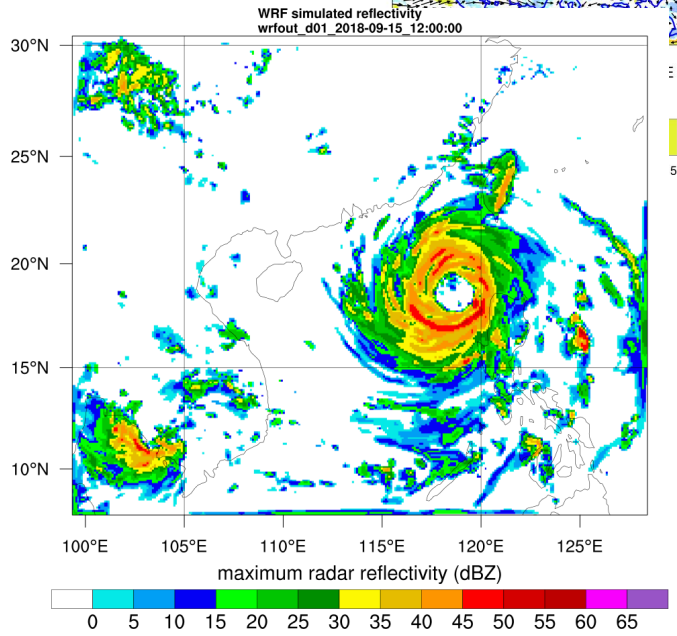












香港天文台
HONG KONG OBSERVATORY

https://nomads.ncep.noaa.gov

https://nomads.ncep.noaa.gov

Most Visited | Getting Started | Probabilistic quantit... | mountain envelop fo... | Explore the immersiv...

National Weather Service
NCEP Central Operations

Home | News

Local forecast by "City, St"
City, St Go
Search NCEP
Go

Current Hazards
Watches/Warnings
Outlooks
National
Current Conditions
Observations
Satellite Images
Radar Imagery
Lakes & Rivers
Space Weather
Unified Surface Analysis
Northern Hemisphere Surface Analysis
Product Loops
Environmental Models
Product Info
Current Status
Model Analyses & Guidance
Forecasts
Current
6 to 10 Day
Aviation
Hurricane
Marine
Tropical Marine
Fire Weather
Forecast Maps
Climate
Climate Prediction
Climate Archives
Weather Safety
Storm Ready
NOAA
Central Library
Photo Library
Public Affairs
Employment
Education Resources
Question of the Month
Image of the Day
About Us
Our Mission
The Centers
Contact Us
Website Questions

USA.gov

NOMADS

NOAA Operational Model Archive and Distribution System

Description of NOAA's NOMADS servers hosting NCEP model data

Help Desk:	Questions or problems please use the link to submit a service ticket.
Background	Background documents about the NOMADS project.
Service Description:	OCWWS Service Description Document
Archive of Model Data:	Some data sets are archived by NCEI, check availability with the link.
Terms of Data Usage:	Use this link to view the Term of Data Usage / Disclaimer.

Based on user requests NCEP will now be utilizing a subscriber list to make announcements with regards to all of our data servers. Follow this link to subscribe and review the list details.

If you have issues with any NCEP server please send your email to:

ncep.pmb.dataflow@noaa.gov

Include as much information about the error as you can - your connection information, the syntax of your request, when the problem started, the error reported, can you replicate the problem, etc.

Click on link in the Data Set field for description and availability info.

Click on the column headings for description of each data access method.

Data Set	freq	grib filter	https	gds
Global Models				
GDAS	6 hours	grib filter	https	OpenDAP
GDAS 0.25	6 hours	grib filter	https	OpenDAP
GFS 0.25 Degree	6 hours	grib filter	https	OpenDAP
GFS 0.25 Degree Hourly	6 hours	grib filter	https	OpenDAP
GFS 0.25 Degree (Secondary Parms)	6 hours	grib filter	https	-
GFS 0.50 Degree	6 hours	grib filter	https	OpenDAP
GFS 1.00 Degree	6 hours	grib filter	https	OpenDAP

Data Transfer: NCEP GFS Forecasts (0.25 degree grid)

g2sub V1.1

g2subset (grib2 subset) allows you to subset (time, field, level, or region) a GRIB2 file and sends you the result

Directory: /gfs.20201205/12

NEW Select one file only (size in bytes)

gfs.t12z.pgrb2.0p25.anl (263835232)

GRIB Filter

For GRIB data you have to option to filter the data.

Extract Levels and Variables

You may select some or all levels and variables. The selections below represent common choices which may or may not be relevant to the files that you have selected. For example choosing RH (relative humidity) would be pointless in file of sea-surface temperatures. In addition, not all possibilities are allowed. For example, suppose you only want the virtual temperature at the tropopause at 01Z. In this case you'd have to transfer the entire file.

For GRIB-2 data only.

Select the levels desired:

☐ all ☐ 0-0.1 m below ground ☐ 0.1-0.4 m below ground ☐ 0.33-1 sigma layer ☐ 0.4-1 m below ground ☐ 0.44-0.72 sigma layer ☐ 0.44-1 sigma layer ☐ 0.4 mb ☐ 0.72-0.94 sigma layer ☐ 0.995 sigma level ☐ 0C isotherm ☐ 1000 mb ☐ 100 m above ground ☐ 100 mb ☐ 10 m above ground ☐ 10 m above mean sea level ☐ 10 mb ☐ 1-2 m below ground ☐ 150 mb ☐ 15 mb ☐ 180-0 mb above ground ☐ 1829 m above mean sea level ☐ 1 hybrid level ☐ 1 mb ☐ 200 mb ☐ 20 m above ground ☐ 20 mb ☐ 250 mb ☐ 255-0 mb above ground ☐ 2743 m above mean sea level ☐ 2 m above ground ☐ 2 mb ☐ 3000-0 m above ground ☐ 300 mb ☐ 30-0 mb above ground ☐ 30 m above ground ☐ 30 mb ☐ 350 mb ☐ 3658 m above mean sea level ☐ 3 mb ☐ 400 mb ☐ 40 m above ground ☐ 40 mb ☐ 450 mb ☐ 500 mb ☐ 50 m above ground ☐ 50 mb ☐ 550 mb ☐ 5 mb ☐ 6000-0 m above ground ☐ 600 mb ☐ 650 mb ☐ 700 mb ☐ 70 mb ☐ 750 mb ☐ 7 mb ☐ 800 mb ☐ 80 m above ground ☐ 850 mb ☐ 900 mb ☐ 925 mb ☐ 950 mb ☐ 975 mb ☐ boundary layer cloud layer ☐ convective cloud bottom level ☐ convective cloud layer ☐ convective cloud top level ☐ entire atmosphere ☐ entire atmosphere (considered as a single layer) ☐ high cloud bottom level ☐ high cloud layer ☐ high cloud top level ☐ highest tropospheric freezing level ☐ low cloud bottom level ☐ low cloud layer ☐ low cloud top level ☐ max wind ☐ mean sea level ☐ middle cloud bottom level ☐ middle cloud layer ☐ middle cloud top level ☐ planetary boundary layer ☐ PV=-2e-06 (Km^2/kg/s) surface ☐ PV=2e-06 (Km^2/kg/s) surface ☐ surface ☐ top of atmosphere ☐ tropopause

Select the variables desired:

☐ all ☐ 4LFTX ☐ 5WAVH ☐ ABSV ☐ ACPCP ☐ ALBDO ☐ APCP ☐ APTMP ☐ CAPE ☐ CFRZR ☐ CICEP ☐ CIN ☐ CLWMR ☐ CPOFP ☐ CPRAT ☐ CRAIN ☐ CSNOW ☐ CWAT ☐ CWORK ☐ DLWRF ☐ DPT ☐ DSWRF ☐ DZDT ☐ FLDCP ☐ GFLUX ☐ GRLE ☐ GUST ☐ HGT ☐ HINDEX ☐ HLCY ☐ HPBL ☐ ICAHT ☐ ICEC ☐ ICEG ☐ ICMR ☐ ICSEV ☐ LAND ☐ LANDN ☐ LFTX ☐ LHTFL ☐ MSLET ☐ O3MR ☐ PEVPR ☐ PLPL ☐ POT ☐ PRATE ☐ PRES ☐ PRMSL ☐ PWAT ☐ REFC ☐ RH ☐ RWMR ☐ SHTFL ☐ SNMR ☐ SNOD ☐ SOILW ☐ SPFH ☐ SUNSD ☐ TCDC ☐ TMAX ☐ TMIN ☐ TMP ☐ TOZNE ☐ TSOIL ☐ UFLX ☐ UGRD ☐ U-GWD ☐ ULWRF ☐ USTM ☐ USWRF ☐ VFLX ☐ VGRD ☐ V-GWD ☐ VIS ☐ VRATE ☐ VSTM ☐ VVEL ☐ VWSH ☐ WATR ☐ WEASD ☐ WILT

Extract Subregion

File transfer times can be reduced by only transferring a subregion. You can use this section to extract a geographic subsection from a most GRIB files. Use negative numbers for south and west.

make subregion ☐ left longitude 0 right longitude 360
top latitude 90 bottom latitude -90

View the URL

☐ Show the URL only for web programming

Don't forget to pause before resubmitting requests.

Start download Reset

香港天文台
HONG KONG OBSERVATORY

Selection of analysis / forecast time levels

Vertical levels

Variables

Specify Region

Generate URL for web programming instead of download

Data Transfer: NCEP GFS Forecasts (0.25 degree grid)

g2sub V1.1

g2subset (grib2 subset) allows you to subset (time, field, level, or region) a GRIB2 file and sends you the result

Directory: /gfs.20201205/12

NEW Select one file only (size in bytes)

gfs.t12z.pgrb2.0p25.anl (263835232)

GRIB Filter

For GRIB data you have to option to filter the data.

Extract Levels and Variables

You may select some or all levels and variables. The selections below represent common choices which may or may not be relevant to the files that you have selected. For example choosing RH (relative humidity) would be pointless in file of sea-surface temperatures. In addition, not all possibilities are allowed. For example, suppose you only want the virtual temperature at the tropopause at 01Z. In this case you'd have to transfer the entire file.

For GRIB-2 data only.

Select the levels desired:

☐ all ☐ 0-0.1 m below ground ☐ 0.1-0.4 m below ground ☐ 0.33-1 sigma layer ☐ 0.4-1 m below ground ☐ 0.44-0.72 sigma layer ☐ 0.44-1 sigma layer ☐ 0.4 mb ☐ 0.72-0.94 sigma layer ☐ 0.995 sigma level ☐ 0C isotherm ☐ 1000 mb ☐ 100 m above ground ☐ 100 mb ☐ 10 m above ground ☐ 10 m above mean sea level ☐ 10 mb ☐ 1-2 m below ground ☐ 150 mb ☐ 15 mb ☐ 180-0 mb above ground ☐ 1829 m above mean sea level ☐ 1 hybrid level ☐ 1 mb ☐ 200 mb ☐ 20 m above ground ☐ 20 mb ☐ 250 mb ☐ 255-0 mb above ground ☐ 2743 m above mean sea level ☐ 2 m above ground ☐ 2 mb ☐ 3000-0 m above ground ☐ 300 mb ☐ 30-0 mb above ground ☐ 30 m above ground ☐ 30 mb ☐ 350 mb ☐ 3658 m above mean sea level ☐ 3 mb ☐ 400 mb ☐ 40 m above ground ☐ 40 mb ☐ 450 mb ☐ 500 mb ☐ 50 m above ground ☐ 50 mb ☐ 550 mb ☐ 5 mb ☐ 6000-0 m above ground ☐ 600 mb ☐ 650 mb ☐ 700 mb ☐ 70 mb ☐ 750 mb ☐ 7 mb ☐ 800 mb ☐ 80 m above ground ☐ 850 mb ☐ 900 mb ☐ 925 mb ☐ 950 mb ☐ 975 mb ☐ boundary layer cloud layer ☐ convective cloud bottom level ☐ convective cloud layer ☐ convective cloud top level ☐ entire atmosphere ☐ entire atmosphere (considered as a single layer) ☐ high cloud bottom level ☐ high cloud layer ☐ high cloud top level ☐ highest tropospheric freezing level ☐ low cloud bottom level ☐ low cloud layer ☐ low cloud top level ☐ max wind ☐ mean sea level ☐ middle cloud bottom level ☐ middle cloud layer ☐ middle cloud top level ☐ planetary boundary layer ☐ PV=-2e-06 (Km^2/kg/s) surface ☐ PV=2e-06 (Km^2/kg/s) surface ☐ surface ☐ top of atmosphere ☐ tropopause

Select the variables desired:

☐ all ☐ 4LFTX ☐ 5WAVH ☐ ABSV ☐ ACPCP ☐ ALBDO ☐ APCP ☐ APTMP ☐ CAPE ☐ CFRZR ☐ CICEP ☐ CIN ☐ CLWMR ☐ CPOFP ☐ CPRAT ☐ CRAIN ☐ CSNOW ☐ CWAT ☐ CWORK ☐ DLWRF ☐ DPT ☐ DSWRF ☐ DZDT ☐ FLDCP ☐ GFLUX ☐ GRLE ☐ GUST ☐ HGT ☐ HINDEX ☐ HLCY ☐ HPBL ☐ ICAHT ☐ ICEC ☐ ICEG ☐ ICMR ☐ ICSEV ☐ LAND ☐ LANDN ☐ LFTX ☐ LHTFL ☐ MSLET ☐ O3MR ☐ PEVPR ☐ PLPL ☐ POT ☐ PRATE ☐ PRES ☐ PRMSL ☐ PWAT ☐ REFC ☐ RH ☐ RWMR ☐ SHTFL ☐ SNMR ☐ SNOD ☐ SOILW ☐ SPFH ☐ SUNSD ☐ TCDC ☐ TMAX ☐ TMIN ☐ TMP ☐ TOZNE ☐ TSOIL ☐ UFLX ☐ UGRD ☐ U-GWD ☐ ULWRF ☐ USTM ☐ USWRF ☐ VFLX ☐ VGRD ☐ V-GWD ☐ VIS ☐ VRATE ☐ VSTM ☐ VVEL ☐ VWSH ☐ WATR ☐ WEASD ☐ WILT

Extract Subregion

File transfer times can be reduced by only transferring a subregion. You can use this section to extract a geographic subsection from a most GRIB files. Use negative numbers for south and west.

make subregion ☐ left longitude 0 right longitude 360

top latitude 90 bottom latitude -90

View the URL

☐ Show the URL only for web programming

Don't forget to pause before resubmitting requests.

Start download Reset

香港天文台
HONG KONG OBSERVATORY

- Suppose “all” levels and “all” variables are selected, then this URL will be generated:

URL=
https://nomads.ncep.noaa.gov/cgi-bin/filter_gfs_0p25.pl?file=gfs.t12z.pgrb2.0p25.anl&all_lev=on&all_var=on&leftlon=0&rightlon=360&toplat=90&bottomlat=-90&dir=%2Fgfs.20201205%2F12

- Copy the above URL and use “curl” command to download on Linux box:

curl https://nomads.ncep.noaa.gov/cgi-bin/filter_gfs_0p25.pl?file=gfs.t12z.pgrb2.0p25.anl&all_lev=on&all_var=on&leftlon=0&rightlon=360&toplat=90&bottomlat=-90&dir=%2Fgfs.20201205%2F12 -o output.grib2

- Select area and check “make subregion” will modify the generated URL:

URL=
https://nomads.ncep.noaa.gov/cgi-bin/filter_gfs_0p25.pl?file=gfs.t12z.pgrb2.0p25.anl&all_lev=on&all_var=on&subregion=&leftlon=40&rightlon=160&toplat=90&bottomlat=-10&dir=%2Fgfs.20201205%2F12

香港天文台
HONG KONG OBSERVATORY

NCEP Products Inventory – Global Products

https://www.nco.ncep.noaa.gov/pmb/products/gfs/

National Weather Service
NCEP Central Operations

Home | News | Organization

DOC NOAA NWS | NCEP Centers: AWC CPC EMC NCO NHC OPC SPC SWPC WPC

Local forecast by "City, St"
City, St Go

Search NCEP
Go

NCEP Quarterly Newsletter

Current Hazards
Watches/Warnings
Outlooks
National
Current Conditions
Observations
Satellite Images
Radar Imagery
Lakes & Rivers
Space Weather
Unified Surface Analysis
Northern Hemisphere Surface Analysis
Product Loops
Environmental Models
Product Info
Current Status
Model Analyses & Guidance
Forecasts
Current
6 to 10 Day
Aviation
Hurricane
Marine
Tropical Marine
Fire Weather
Forecast Maps
Climate
Climate Prediction
Climate Archives
Weather Safety
Storm Ready
NOAA
Central Library
Photo Library
Public Affairs
Employment
Education Resources
Image of the Day
About Us
Our Mission
The Centers
Contact Us
Data Questions
Website Questions

NCEP Home > NCO Home > IDSB > NCEP Product Inventory - Global Products

NCEP Products Inventory

Global Products

Updated: 06/23/2019

Global Forecast System (GFS) Model
Global Data Assimilation System (GDAS) Model

Information about the GFS

CC is the model cycle runtime (i.e. 00, 06, 12, 18)
FFF is the forecast hour of product from 000 - 384
YYYYMMDD is the Year, Month and Day

Click on the links under **Inventory** to see the file's contents.
Use as a guide when selecting specific parameters from NOMADS. Click to view what is available.

DESCRIPTION	NCEP FTP/HTTP SERVER File Name	NWS FTP SERVER File Name
GFS		
GFS - Global longitude-latitude grid Most commonly used parameters WCOSS File Name / Inventory: 0.25 degree resolution: gfs.tCCz.pgrb2.0p25.fFFF — ANL FH000 FH001-384 0.5 degree resolution: gfs.tCCz.pgrb2.0p50.fFFF — ANL FH000 FH003-384 1.0 degree resolution: gfs.tCCz.pgrb2.1p00.fFFF — ANL FH000 FH003-384	Available in GRIB2 via ftp Available in GRIB2 via https gfs.tCCz.pgrb2.0p25.fFFF gfs.tCCz.pgrb2.0p50.fFFF gfs.tCCz.pgrb2.1p00.fFFF	Not Available
GFS - Global longitude-latitude grid Least commonly used parameters WCOSS File Name / Inventory: 0.25 degree resolution: gfs.tCCz.pgrb2b.0p25.fFFF — ANL FH000 FH001-384 0.5 degree resolution: gfs.tCCz.pgrb2b.0p50.fFFF — ANL FH000 FH003-384 1.0 degree resolution: gfs.tCCz.pgrb2b.1p00.fFFF — ANL FH000 FH003-384	Available in GRIB2 via ftp Available in GRIB2 via https gfs.tCCz.pgrb2b.0p25.fFFF gfs.tCCz.pgrb2b.0p50.fFFF gfs.tCCz.pgrb2b.1p00.fFFF	Not Available

Inventory of File *gfs.t00z.pgrb2.0p25.f003*

Model: GFS
Cycle: 00 UTC
Forecast: 03
Number of Records: 586

Number	Level/Layer	Parameter	Forecast Valid	Description
001	1 hybrid level	CLWMR	3 hour fcst	Cloud Mixing Ratio [kg/kg]
002	1 hybrid level	ICMR	3 hour fcst	Ice Water Mixing Ratio [kg/kg]
003	1 hybrid level	RWMR	3 hour fcst	Rain Mixing Ratio [kg/kg]
004	1 hybrid level	SNMR	3 hour fcst	Snow Mixing Ratio [kg/kg]
005	1 hybrid level	GRLE	3 hour fcst	Graupel [kg/kg]
006	entire atmosphere	REFC	3 hour fcst	Composite reflectivity [dB]
007	surface	VIS	3 hour fcst	Visibility [m]
008	planetary boundary layer	UGRD	3 hour fcst	U-Component of Wind [m/s]
009	planetary boundary layer	VGRD	3 hour fcst	V-Component of Wind [m/s]
010	planetary boundary layer	VRATE	3 hour fcst	Ventilation Rate [m^2/s]
011	surface	GUST	3 hour fcst	Wind Speed (Gust) [m/s]
012	0.4 mb	HGT	3 hour fcst	Geopotential Height [gpm]
013	0.4 mb	TMP	3 hour fcst	Temperature [K]
014	0.4 mb	ABSV	3 hour fcst	Absolute Vorticity [1/s]
015	0.4 mb	O3MR	3 hour fcst	Ozone Mixing Ratio [kg/kg]
016	1 mb	HGT	3 hour fcst	Geopotential Height [gpm]
017	1 mb	TMP	3 hour fcst	Temperature [K]
018	1 mb	RH	3 hour fcst	Relative Humidity [%]
019	1 mb	UGRD	3 hour fcst	U-Component of Wind [m/s]
020	1 mb	VGRD	3 hour fcst	V-Component of Wind [m/s]
021	1 mb	O3MR	3 hour fcst	Ozone Mixing Ratio [kg/kg]
022	2 mb	HGT	3 hour fcst	Geopotential Height [gpm]
023	2 mb	TMP	3 hour fcst	Temperature [K]
024	2 mb	RH	3 hour fcst	Relative Humidity [%]
025	2 mb	UGRD	3 hour fcst	U-Component of Wind [m/s]

香港天文台
HONG KONG OBSERVATORY

To save time

- A NCEP GFS Forecast GRIB2 file (on surface and selected pressure levels) is available at:

  /r2/tutorial/data/2020120512/ gfs_20201205_12z_0p25_f072.grib2

- A script to download GFS GRIB2 file with user's input model initial time and forecast hour is available (let's come back later if have time)

香港天文台
HONG KONG OBSERVATORY

Using PyNGL and PyNIO

香港天文台
HONG KONG OBSERVATORY

Python version of NCL (NCAR Command Language)

Of course including text, csv, Shapefiles;

Interface with database through other Python libraries

PyNIO and PyNGL

Python packages for file input / output and visualization

NCAR

NCAR is sponsored by NSF

Search

Examples

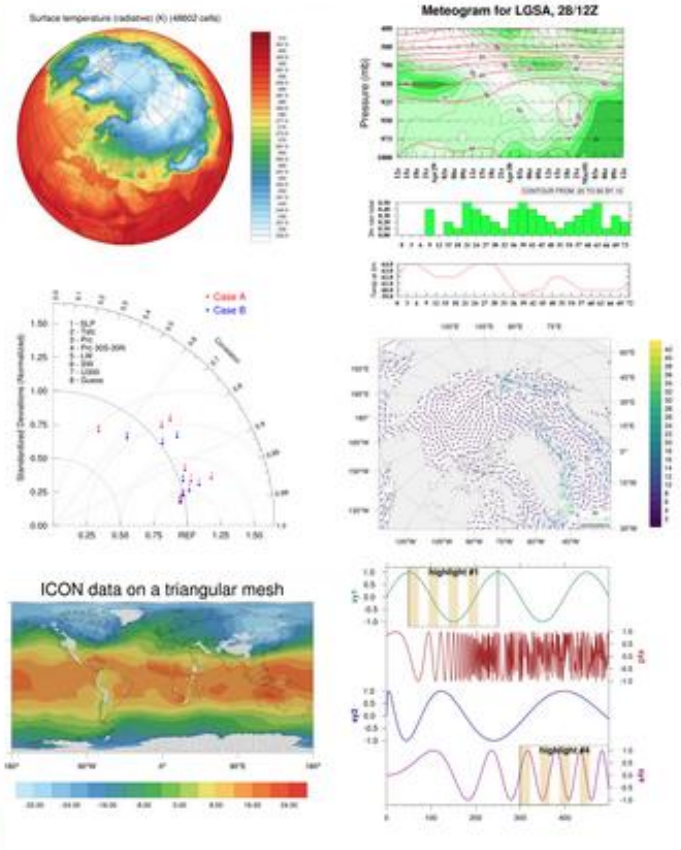
- overview
- gallery
- alphabetical
- by category
- by function
- by resource

Links

- Letter to NCL users
- Home
- PyNGL GitHub
- PyNIO GitHub
- What's new
- Functions
- Resources
- PyNIO
- Colors
- FAQ
- User forum
- Quick links
- Reporting bugs
- Glossary
- Download
- Contributors

PyNGL ("pingle") is a Python language module for generating high-quality, 2D visualizations of scientific data. It is built on top of the same "resource" model used in the NCAR Command Language.

PyNIO ("pie nee oh") is a Python module used for reading and writing NetCDF, GRIB, HDF, and HDF-EOS.



NCAR

NCAR is sponsored by the National Science Foundation.

Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the National Science Foundation.

香港天文台
HONG KONG OBSERVATORY

Installation on Linux box

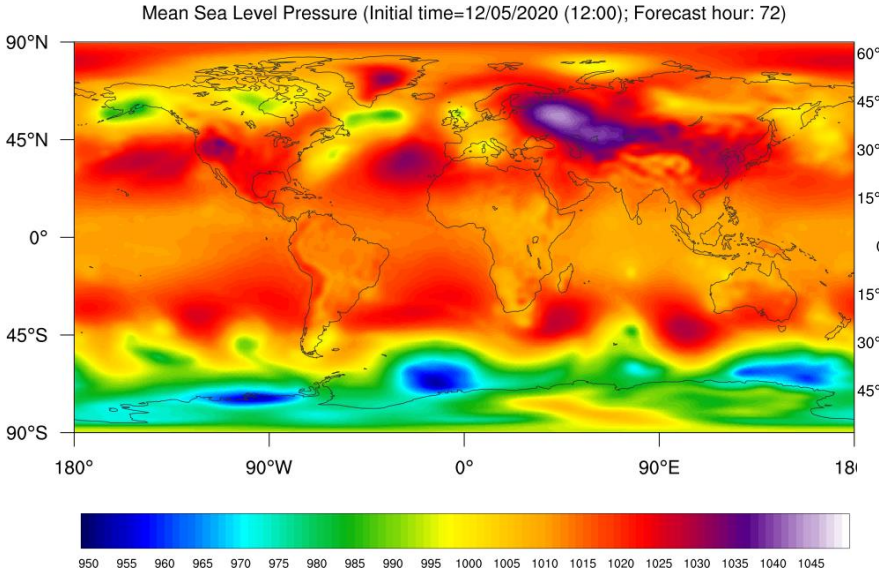
1. Anaconda:
   - wget https://repo.anaconda.com/archive/Anaconda3-2020.11-Linux-x86_64.sh
   - bash [path of download file]/Anaconda3-2020.11-Linux-x86_64.sh
   - set up Anaconda according to default path ~/anaconda3 or input under another path

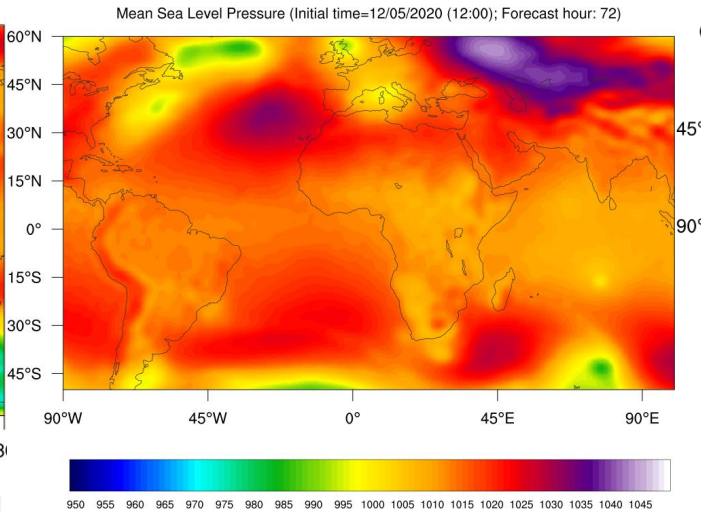
2. Install PyNGL, PyNIO and wrf-python
   - conda create --name **pyn_env** --channel conda-forge pynio pyngl
   - conda activate pyn_env
   - conda install -c conda-forge wrf-python (* install wrf-python under PyNGL environment is preferable)

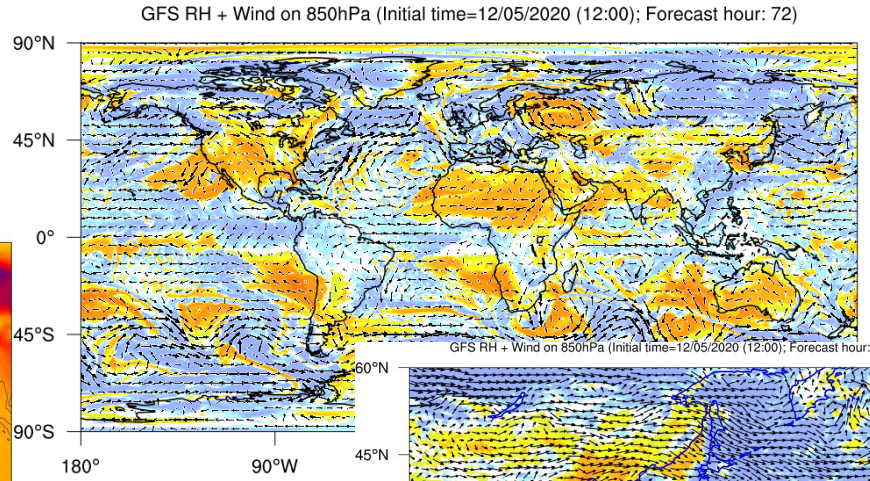
3. Optional tools:
   - conda install -c conda-forge wgrib
   - conda install -c conda-forge wgrib2
   - conda install -c conda-forge netcdf4

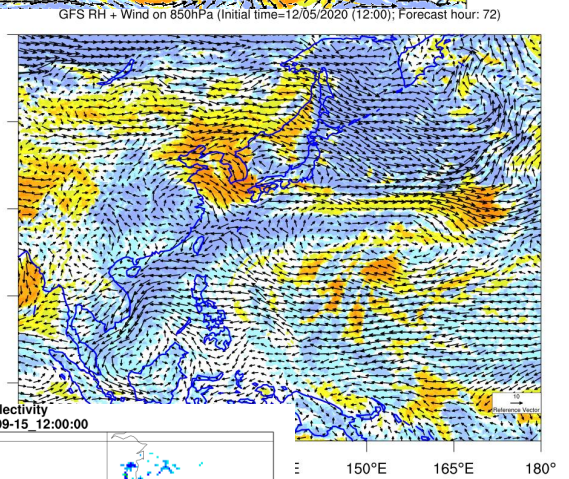
香港天文台
HONG KONG OBSERVATORY

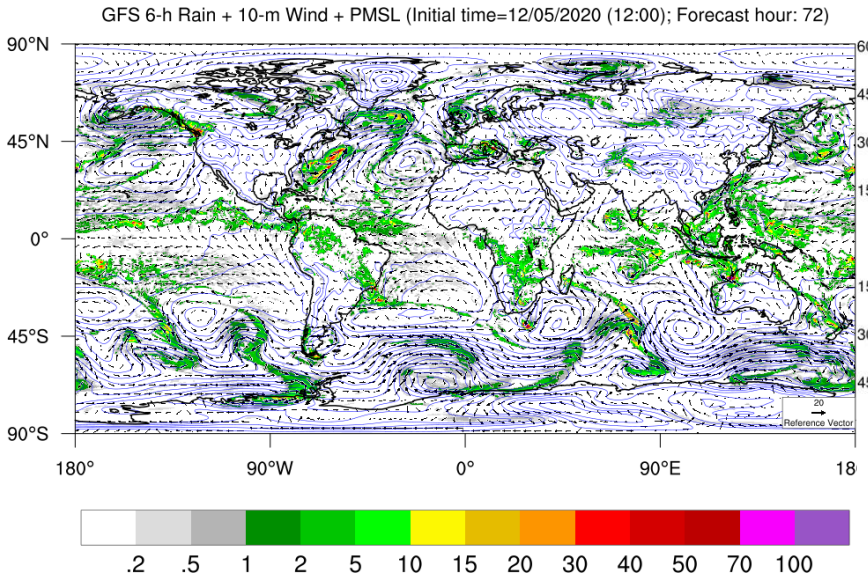
Your outcomes

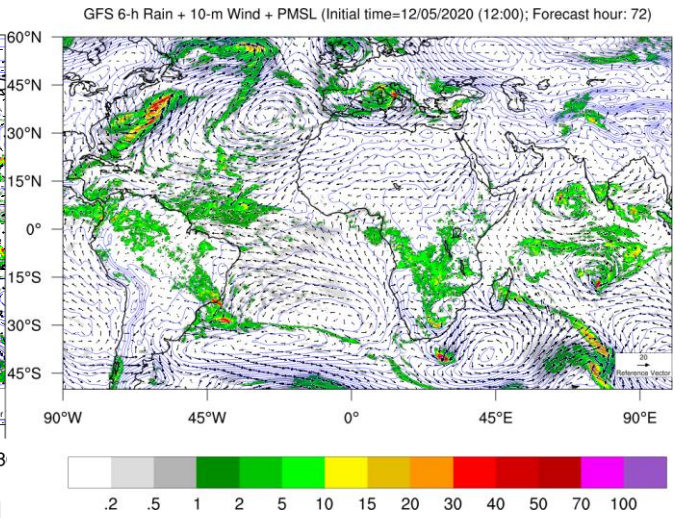


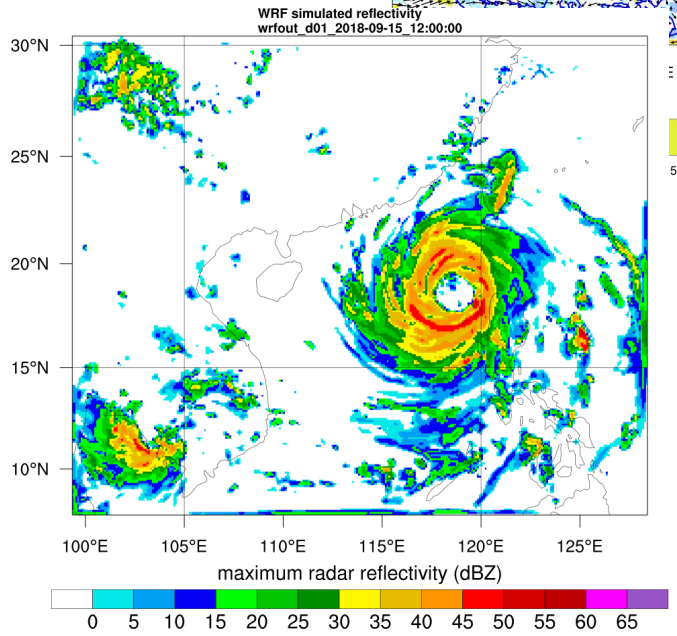












香港天文台
HONG KONG OBSERVATORY

Let's start …

香港天文台
HONG KONG OBSERVATORY

(1) Copy required Python scripts

- Source: /r2/tutorial/python

```
GFS_ex1.py
GFS_ex2.py
GFS_ex3.py
GFS_ex4.py
GFS_ex5.py
GFS_ex6.py
GFS_ex7.py
WRF_ex8.py
exlib.py
```

- Create a directory under your environment, then type:

```
cp  /r2/tutorial/python/*.py   ./
```

香港天文台
HONG KONG OBSERVATORY

GFS_ex1.py

```
import os, sys, types
import numpy  as np

import Ngl, Nio

f_path         = "/r2/tutorial/data/2020120512"
gfs_in         = "gfs_20201205_12z_0p25_f072.grib2"
out_img_prefix = "ex1_mslp_global"   # output image file prefix
wks_type       = "png"

# read GFS GRIB file:
f_name=f_path+"/"+gfs_in

file = Nio.open_file(f_name, "r")

fvar = file.variables["PRMSL_P0_L101_GLL0"]
lat  = file.variables["lat_0"]
lon  = file.variables["lon_0"]

# obtain value (in numpy array) from Nio object
fvar_value = fvar.get_value()
fvar_value = fvar_value * 0.01  # Pa to hPa

# create a "workstation" to draw graphics
wks = Ngl.open_wks(wks_type,out_img_prefix,None)

# instantiate contour resources:
cnres = Ngl.Resources()
cnres.nglDraw  = False
cnres.nglFrame = False

# set color map
cmap_name="BkBlAqGrYeOrReViWh200"
cmap = Ngl.read_colormap_file(cmap_name)

# set contour interval
cntr_levels = np.arange(950., 1050., 1.0)
```

(continue …)

```
cnres.mpFillOn                    = False
cnres.mpGeophysicalLineColor      = "Grey18"
cnres.mpGeophysicalLineThicknessF = 1.5
#cnres.mpGridAndLimbOn             = False

cnres.cnLinesOn                   = False
cnres.cnLineLabelsOn              = False
cnres.cnFillOn                    = True
cnres.cnFillMode                  = "RasterFill"        # speed up plotting
cnres.trGridType                  = "TriangularMesh"
cnres.cnFillPalette               = cmap[10:,:]
cnres.cnLevelSelectionMode        = "ExplicitLevels"
cnres.cnLevels                    = cntr_levels

cnres.lbOrientation               = "horizontal"
cnres.lbLabelStride               = 5
cnres.lbBoxSeparatorLinesOn       = False

cnres.tiMainString                = "Mean Sea Level Pressure (Initial time="
+ fvar.initial_time  + "; Forecast hour: " +
'{:d}'.format(fvar.forecast_time[0]) + ")"
cnres.tiMainFontHeightF           = 0.0105

cnres.sfXCStartV = float(min(lon))
cnres.sfXCEndV   = float(max(lon))
cnres.sfYCStartV = float(max(lat))
cnres.sfYCEndV   = float(min(lat))

plot = Ngl.contour_map(wks,fvar_value,cnres)

Ngl.draw(plot)
Ngl.frame(wks)
```

香港天文台
HONG KONG OBSERVATORY

Now generate your first prog chart

1. Type : "`source /r2/anaconda3/0_conda_init.sh`" ← enable conda environment
   - your command prompt will be prefixed by (base)

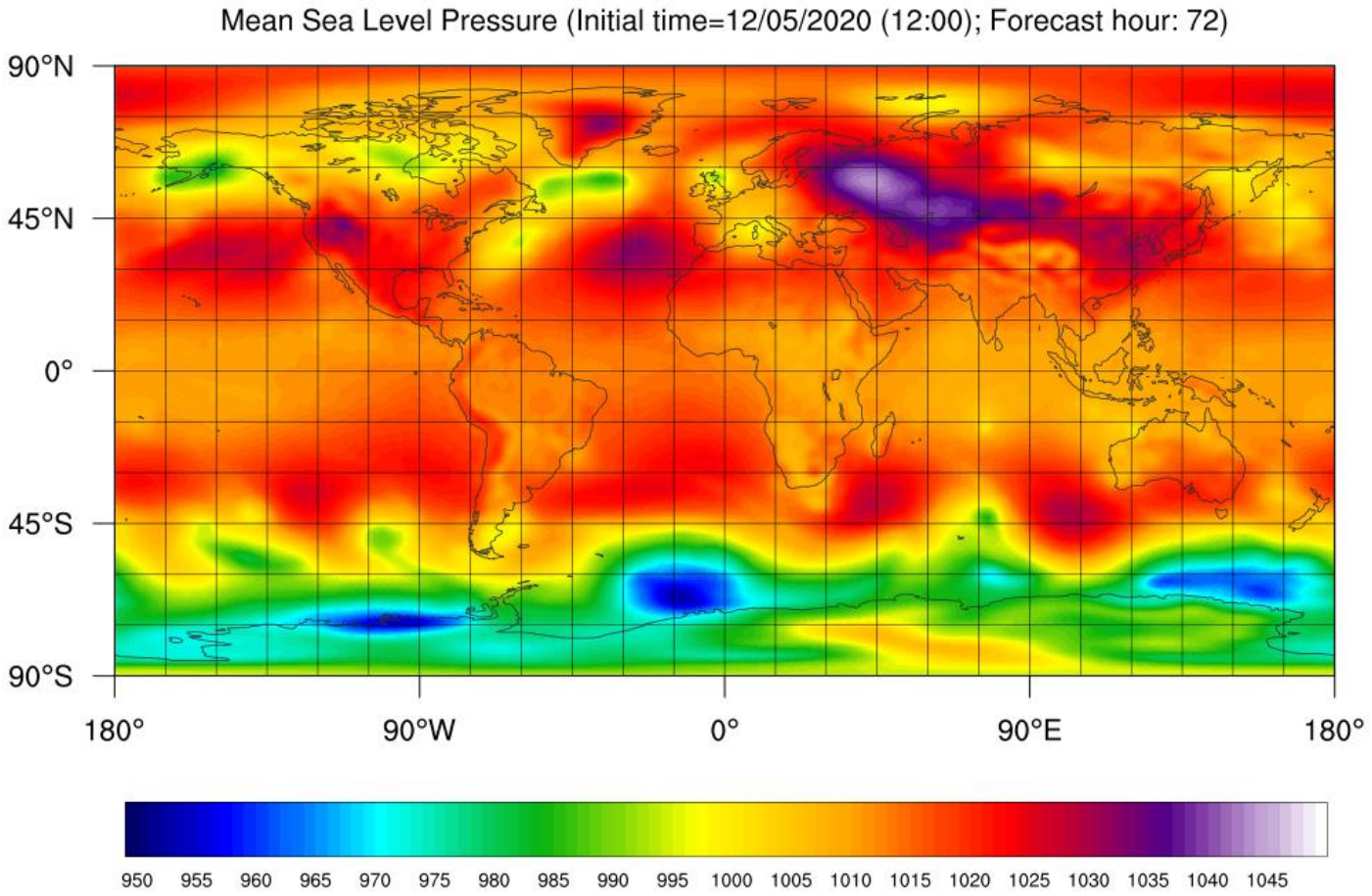
2. Type : "`conda activate pyn_env`" ← activate PyNGL, PyNIO as installed
   - Change from (base) to (pyn_env)

3. Type : "`python GFS_ex1.py`"

4. A image file "`ex1_mslp_global.png`" will be generated

香港天文台
HONG KONG OBSERVATORY

Grid lines …



香港天文台
HONG KONG OBSERVATORY

GFS_ex1.py

```
import os, sys, types
import numpy  as np

import Ngl, Nio

f_path         = "/r2/tutorial/data/2020120512"
gfs_in         = "gfs_20201205_12z_0p25_f072.grib2"
out_img_prefix = "ex1_mslp_global"   # output image file prefix
wks_type       = "png"

# read GFS GRIB file:
f_name=f_path+"/"+gfs_in

file = Nio.open_file(f_name, "r")

fvar = file.variables["PRMSL_P0_L101_GLL0"]
lat  = file.variables["lat_0"]
lon  = file.variables["lon_0"]

# obtain value (in numpy array) from Nio object
fvar_value = fvar.get_value()
fvar_value = fvar_value * 0.01  # Pa to hPa

# create a "workstation" to draw graphics
wks = Ngl.open_wks(wks_type,out_img_prefix,None)

# instantiate contour resources:
cnres = Ngl.Resources()
cnres.nglDraw  = False
cnres.nglFrame = False

# set color map
cmap_name="BkBlAqGrYeOrReViWh200"
cmap = Ngl.read_colormap_file(cmap_name)

# set contour interval
cntr_levels = np.arange(950., 1050., 1.0)
```

Disable drawing until all graphics (e.g. overlay) are completed

(continue …)

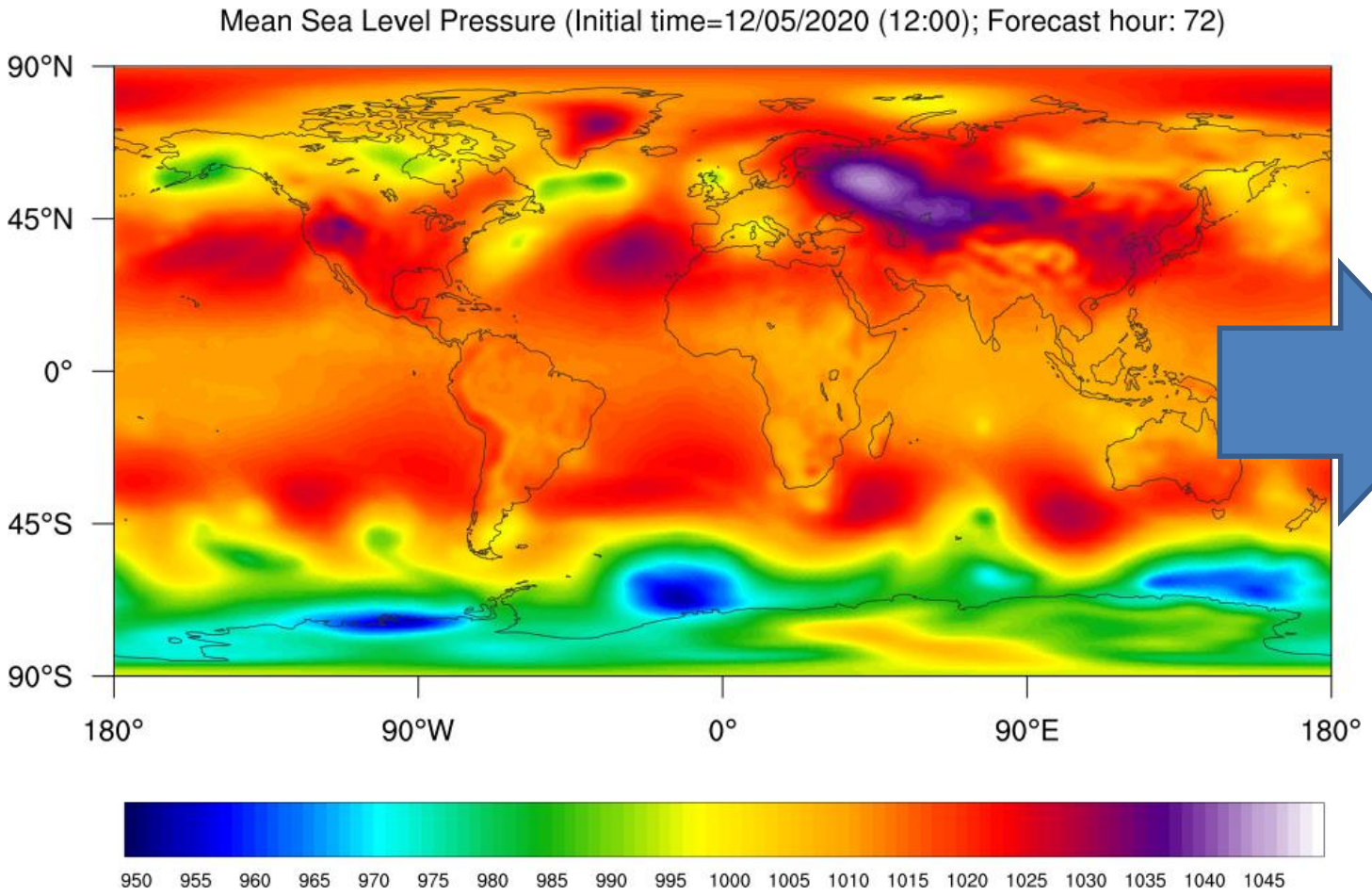
```
cnres.mpFillOn                   = False
cnres.mpGeophysicalLineColor     = "Grey18"
cnres.mpGeophysicalLineThicknessF = 1.5
#cnres.mpGridAndLimbOn            = False
```

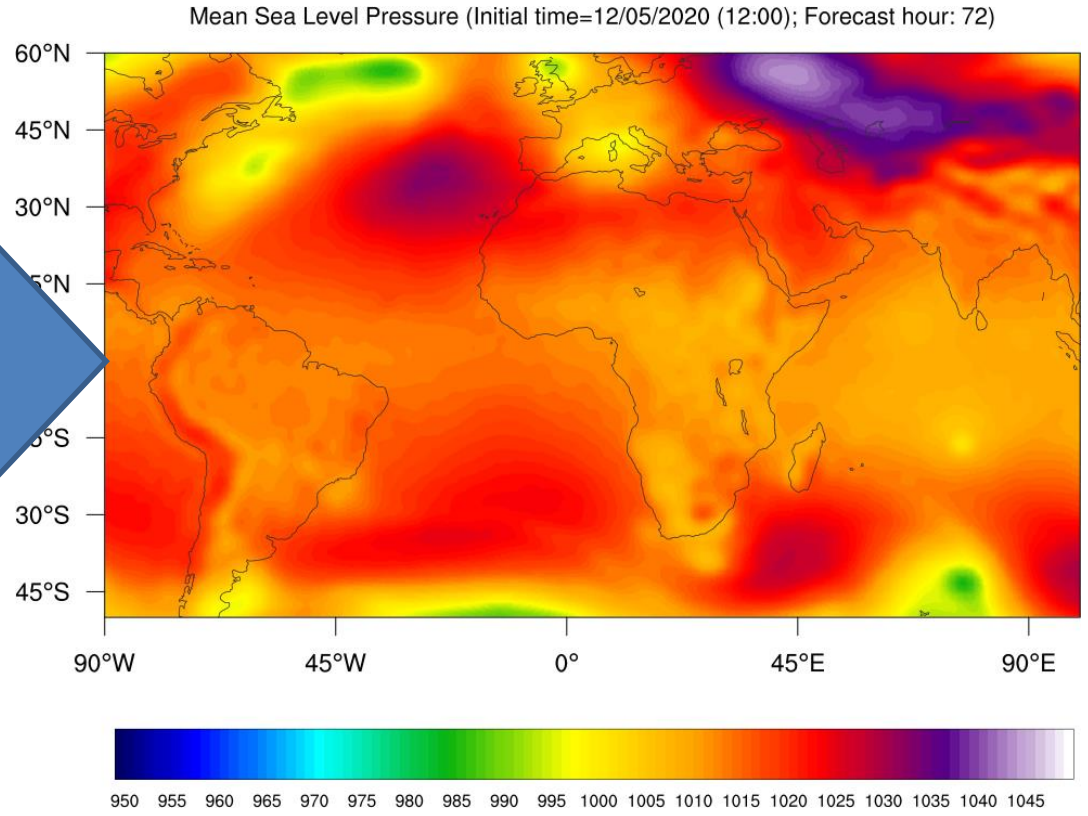
Turn off grid lines

```
cnres.cnLinesOn                  = False
cnres.cnLineLabelsOn             = False
cnres.cnFillOn                   = True
cnres.cnFillMode                 = "RasterFill"        # speed up plotting
cnres.trGridType                 = "TriangularMesh"
cnres.cnFillPalette              = cmap[10:,:]
cnres.cnLevelSelectionMode       = "ExplicitLevels"
cnres.cnLevels                   = cntr_levels

cnres.lbOrientation              = "horizontal"
cnres.lbLabelStride              = 5
cnres.lbBoxSeparatorLinesOn      = False

cnres.tiMainString               = "Mean Sea Level Pressure (Initial time="
+ fvar.initial_time  + "; Forecast hour: " +
'{:d}'.format(fvar.forecast_time[0]) + ")"
cnres.tiMainFontHeightF          = 0.0105

cnres.sfXCStartV = float(min(lon))
cnres.sfXCEndV   = float(max(lon))
cnres.sfYCStartV = float(max(lat))
cnres.sfYCEndV   = float(min(lat))

plot = Ngl.contour_map(wks,fvar_value,cnres)

Ngl.draw(plot)
Ngl.frame(wks)
```

香港天文台
HONG KONG OBSERVATORY

GFS_ex2.py





香港天文台
HONG KONG OBSERVATORY

GFS_ex2.py

See "PyNIO Extended Selection" :

http://www.pynio.ucar.edu/NioExtendedSelection.shtml

```
import os, sys, types
import numpy  as np

import Ngl, Nio

f_path         = "/r2/tutorial/data/2020120512"
gfs_in         = "gfs_20201205_12z_0p25_f072.grib2"
out_img_prefix = "ex2_mslp_region"   # output image file prefix
wks_type       = "png"

# selected area (lat:90...-90N, lon: -180..180E)
lat_s = -50
lat_n = 60
lon_w = -90.0
lon_e = 105.0

# read GFS GRIB file:
f_name=f_path+"/"+gfs_in

file = Nio.open_file(f_name, "r")

fvar = file.variables["PRMSL_P0_L101_GLL0"]
lat  = file.variables["lat_0"]
lon  = file.variables["lon_0"]

# extract area of data
fvar_area = fvar['lat_0|%f:%f lon_0|%f:%f' % (lat_n,lat_s,lon_w,lon_e)]
fvar_value = fvar_area  * 0.01
lat_area = lat['lat_0|%f:%f' % (lat_n, lat_s)]
lon_area = lon['lon_0|%f:%f' % (lon_w, lon_e)]

# create 2D array for lat/lon used in regional plot
lat2d = np.zeros((len(lat_area),len(lon_area)))
lon2d = np.zeros((len(lat_area),len(lon_area)))

for i in range(0, len(lon_area)):
   lat2d[:,i] = lat_area

for i in range(0, len(lat_area)):
   lon2d[i,:] = lon_area

# open a workstation to draw graphics
wks = Ngl.open_wks(wks_type,out_img_prefix,None)

cnres = Ngl.Resources()
cnres.nglDraw  = False
cnres.nglFrame = False
```

香港天文台
HONG KONG OBSERVATORY

Need to specify 2D array of lat/lon in regional plot of contour

Specify area of map display

```
cmap_name="BkBlAqGrYeOrReViWh200"
cmap = Ngl.read_colormap_file(cmap_name)

# set contour levels
cntr_levels = np.arange(950., 1050., 1.0)


cnres.mpFillOn                    = False
cnres.mpGeophysicalLineColor      = "Grey18"
cnres.mpGeophysicalLineThicknessF = 1.5
cnres.mpGridAndLimbOn             = False

cnres.cnLinesOn                   = False
cnres.cnLineLabelsOn              = False
cnres.cnFillOn                    = True
cnres.cnFillMode                  = "RasterFill"        # speed up plotting
cnres.trGridType                  = "TriangularMesh"
cnres.cnFillPalette               = cmap[10:,:]
cnres.cnLevelSelectionMode        = "ExplicitLevels"
cnres.cnLevels                    = cntr_levels

cnres.lbOrientation               = "horizontal"
cnres.lbLabelStride               = 5
cnres.lbBoxSeparatorLinesOn       = False

cnres.tiMainString                = "Mean Sea Level Pressure(Initial time=" + fvar.initial_time  +  ";
Forecast hour: "  + '{:d}'.format(fvar.forecast_time[0]) + ")"
cnres.tiMainFontHeightF           = 0.0105

# need to specify an explict 2D array for lat/lon in regional plot:
cnres.sfXArray = lon2d
cnres.sfYArray = lat2d

# need to align with map boundary:
cnres.mpLimitMode       = "LatLon"              #-- must be set using minLatF/maxLatF/minLonF/maxLonF
cnres.mpMinLatF         = lat_s                 #-- sub-region minimum latitude
cnres.mpMaxLatF         = lat_n                 #-- sub-region maximum latitude
cnres.mpMinLonF         = lon_w                 #-- sub-region minimum longitude
cnres.mpMaxLonF         = lon_e                 #-- sub-region maximum longitude


plot = Ngl.contour_map(wks,fvar_value,cnres)

Ngl.draw(plot)
Ngl.frame(wks)
```

香港天文台
HONG KONG OBSERVATORY

To generate prog chart:

1. Type : "`source /r2/anaconda3/0_conda_init.sh`" ← enable conda environment
   - your command prompt will be prefixed by (base)

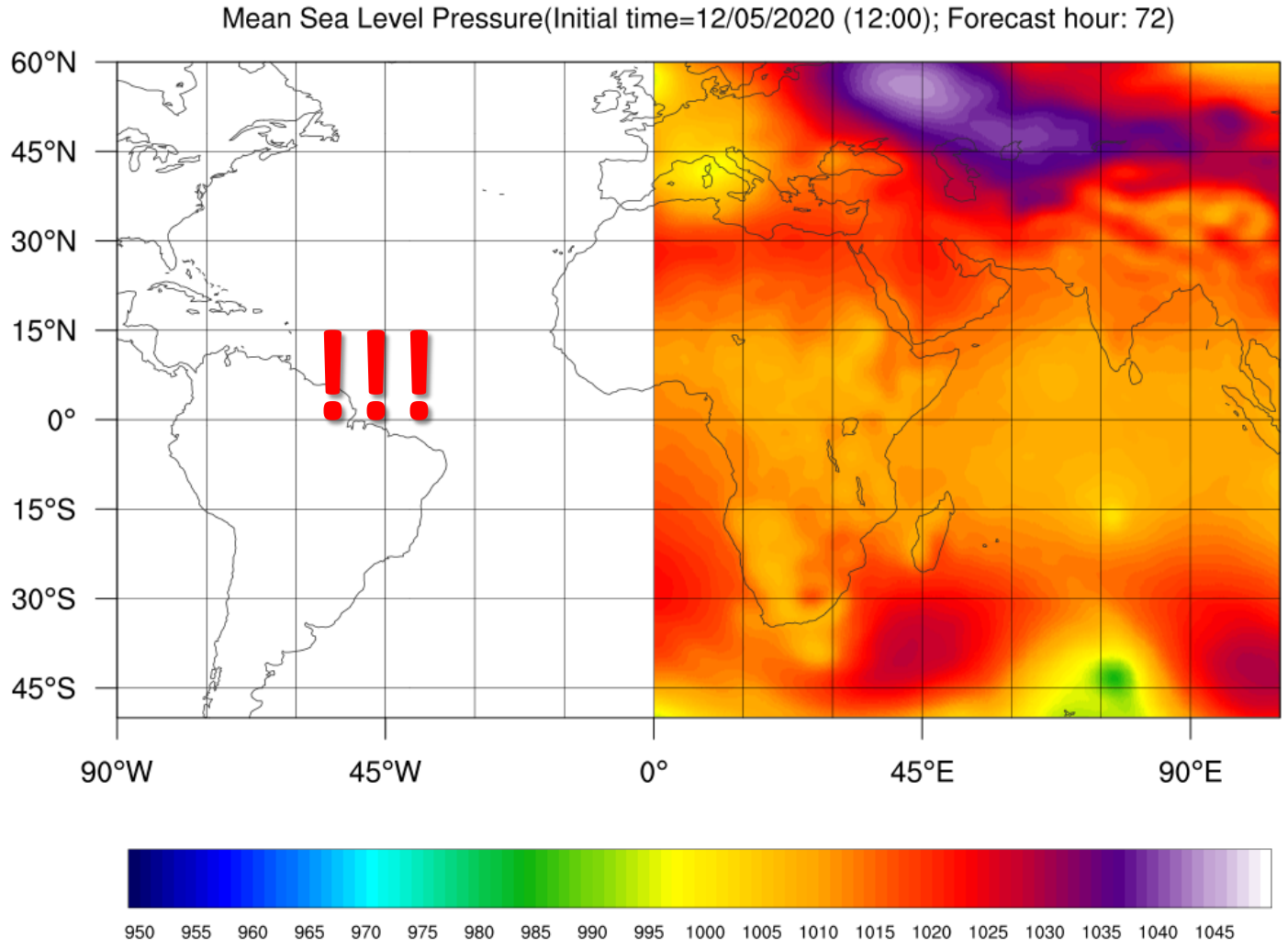
2. Type : "`conda activate pyn_env`" ← activate PyNGL, PyNIO as installed
   - Change from (base) to (pyn_env)

3. Type : "`python GFS_ex2.py`"

4. A image file "`ex2_mslp_region.png`" will be generated

香港天文台
HONG KONG OBSERVATORY

But ...



香港天文台
HONG KONG OBSERVATORY

GFS_ex3.py

import exlib.py

Get selected area using function “extract_area” in exlib.py

```
import os, sys, types
import numpy  as np

import Ngl, Nio

# import extra function
import exlib  as ex

f_path         = "/r2/tutorial/data/2020120512"
gfs_in         = "gfs_20201205_12z_0p25_f072.grib2"
out_img_prefix = "ex3_mslp_region"   # output image file prefix
wks_type       = "png"

# selected area (lat:90...-90N, lon: -180..180E)
lat_s = -50
lat_n = 60
lon_w = -90.0
lon_e = 100.0

# read GFS GRIB file:
f_name=f_path+"/"+gfs_in

file = Nio.open_file(f_name, "r")

fvar = file.variables["PRMSL_P0_L101_GLL0"]
lat  = file.variables["lat_0"]
lon  = file.variables["lon_0"]


# magnitude of lon_w must be smaller than lon_e
# for handling array extraction and map plotting
if (lon_w > 0 and lon_e < 0) :
   lon_e = lon_e + 360.0

fvar_area,lat2d,lon2d = ex.extract_area(fvar,lat,lon,lat_s,lat_n,lon_w,lon_e)

# Pa -> hPa
fvar_value = fvar_area * 0.01

# open workstation to draw graphics:
wks = Ngl.open_wks(wks_type,out_img_prefix,None)

cnres = Ngl.Resources()
cnres.nglDraw  = False
cnres.nglFrame = False

                                    (… SAME as GFS_ex2.py …)
```

香港天文台
HONG KONG OBSERVATORY

exlib.py

```
import os, sys, types
import numpy  as np

def extract_area(fvar,flat,flon,lat_s,lat_n,lon_w,lon_e):

   lat_temp = flat[:]
   lat_s_idx = (np.abs(lat_temp-lat_s)).argmin()
   lat_n_idx = (np.abs(lat_temp-lat_n)).argmin()

   if lat_s_idx == lat_n_idx:
      sys.exit('lat values are too close, need to be on different grid points')
   elif lat_s_idx > 1 and lat_n_idx < len(lat_temp)-2:
      lat_bound1 = lat_n_idx-2
      lat_bound2 = lat_s_idx+2
      lat = lat_temp[lat_bound1:lat_bound2]
   else:
      lat_bound1 = lat_n_idx
      lat_bound2 = lat_s_idx
      lat = lat_temp[lat_bound1:lat_bound2]

                                        ... code skipped ...

   if (np.sign(lon_w) + np.sign(lon_e)) >= -1 and (np.sign(lon_w) + np.sign(lon_e)) <= 1:

      fvar_temp1 = fvar[lat_bound1:lat_bound2,0:lon_bound1]
      fvar_temp2 = fvar[lat_bound1:lat_bound2,lon_bound2:lon_bound3]
      fvar_area  = np.concatenate((fvar_temp2,fvar_temp1),axis=1)
      del fvar_temp1
      del fvar_temp2

   else:

      fvar_area = fvar[lat_bound1:lat_bound2,lon_bound1:lon_bound2]


   # create 2d lat and lon
   lat2d = np.zeros((len(lat),len(lon)))
   lon2d = np.zeros((len(lat),len(lon)))
   for i in range(0, len(lon)):
      lat2d[:,i] = lat
   for i in range(0, len(lat)):
      lon2d[i,:] = lon


   return fvar_area, lat2d, lon2d
```

香港天文台
HONG KONG OBSERVATORY

To generate prog chart:

1. Type : "`source /r2/anaconda3/0_conda_init.sh`" ← enable conda environment
   - your command prompt will be prefixed by (base)

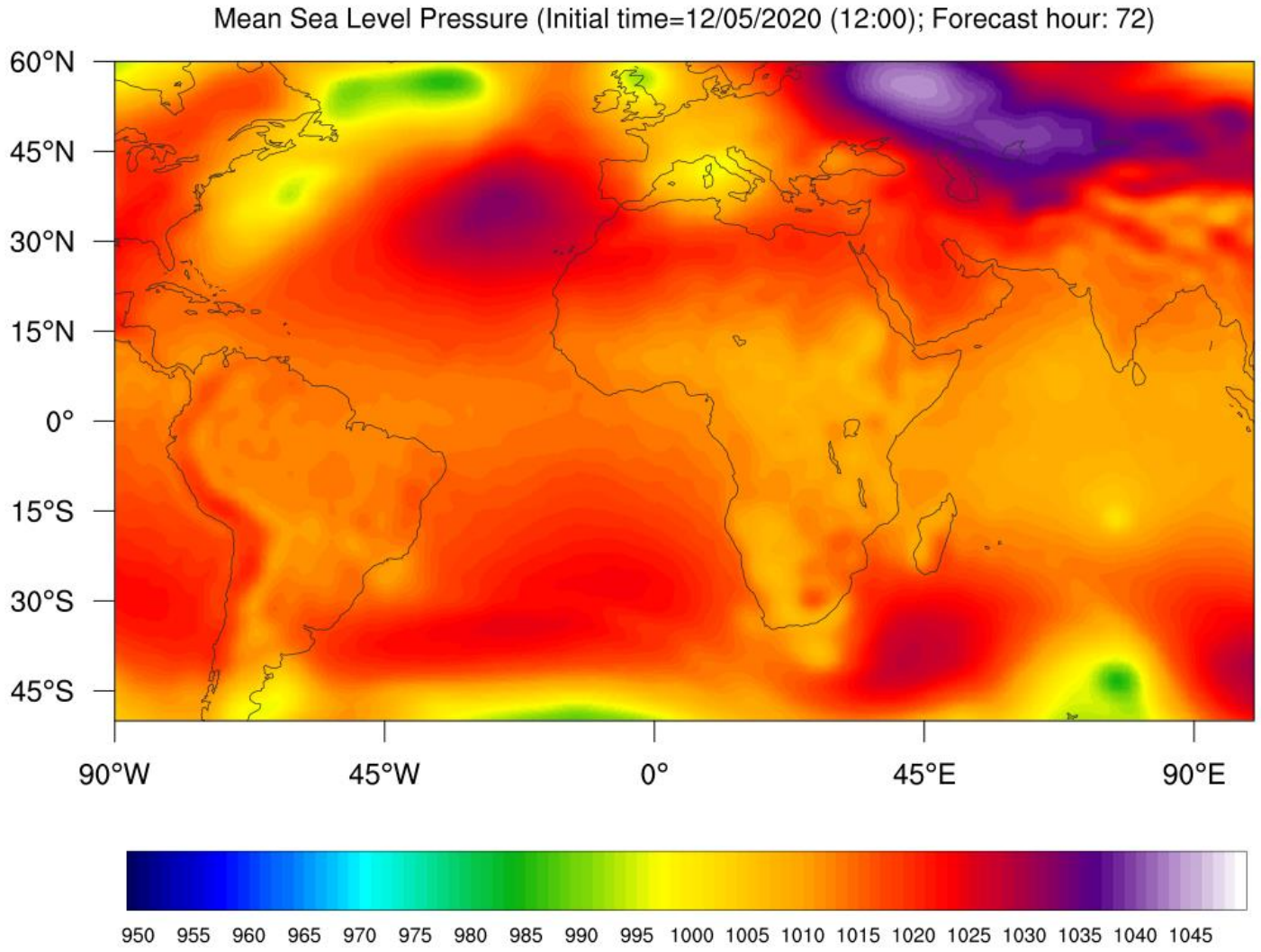
2. Type : "`conda activate pyn_env`" ← activate PyNGL, PyNIO as installed
   - Change from (base) to (pyn_env)

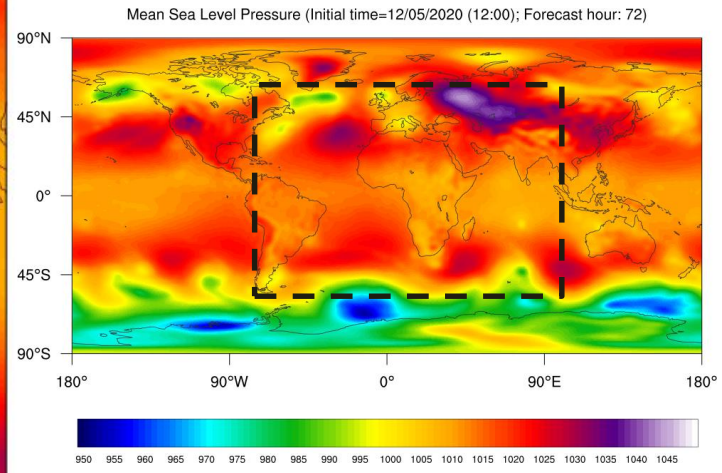
3. Type : "`python GFS_ex3.py`"

4. A image file "`ex3_mslp_region.png`" will be generated

香港天文台
HONG KONG OBSERVATORY

Done !



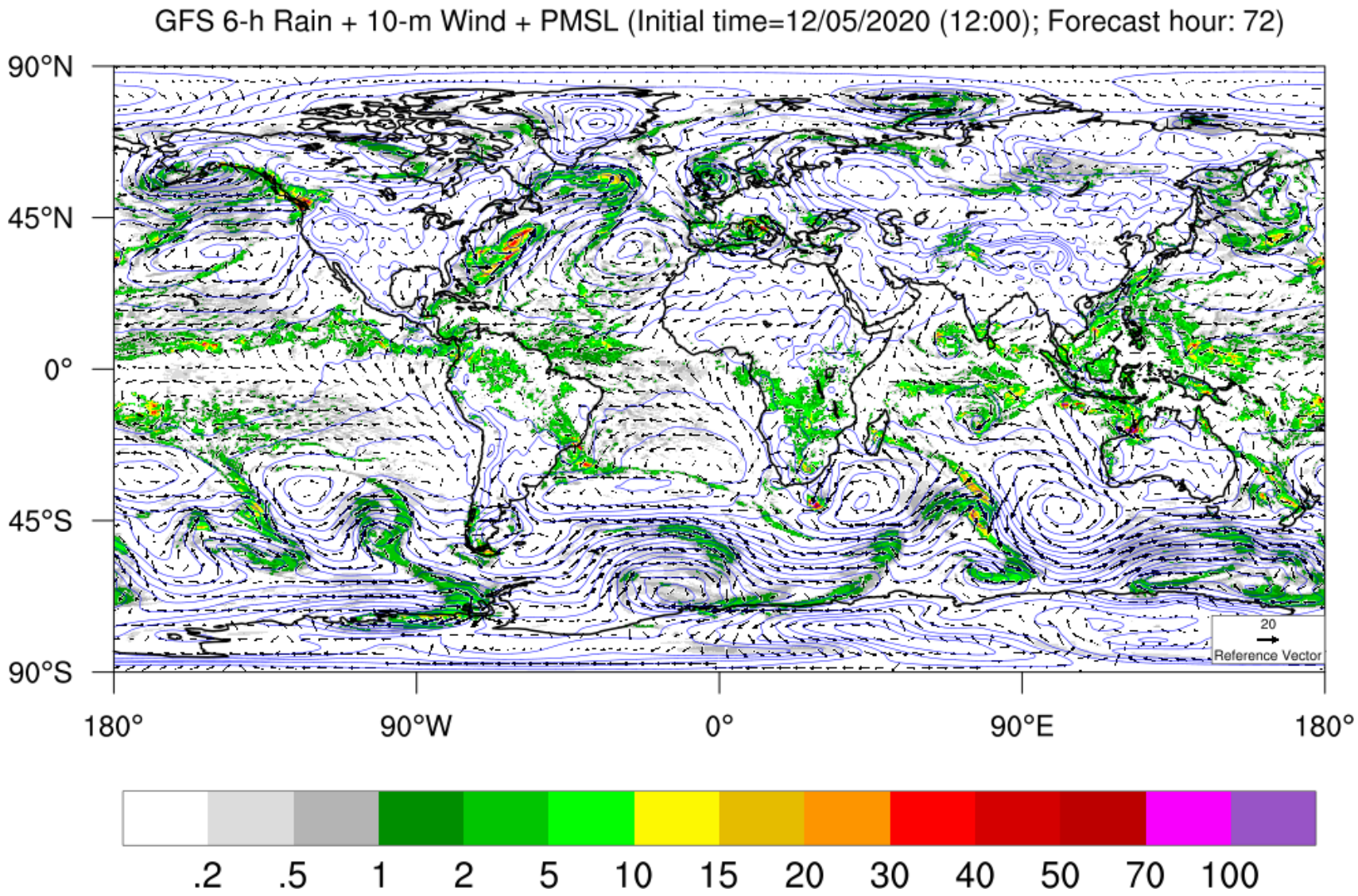


"ex3_mslp_region.png"

香港天文台
HONG KONG OBSERVATORY

More complicated composite plots

GFS_ex4.py



ex4_sfc_global.png

香港天文台
HONG KONG OBSERVATORY

1. Need more variables:

```
import os, sys, types
import numpy  as np

import Ngl, Nio

f_path         = "/r2/tutorial/data/2020120512"
gfs_in         = "gfs_20201205_12z_0p25_f072.grib2"
out_img_prefix = "ex4_sfc_global"    # output image file prefix
wks_type       = "png"

# read GFS GRIB file:
f_name=f_path+"/"+gfs_in

file = Nio.open_file(f_name, "r")

pmsl = file.variables["PRMSL_P0_L101_GLL0"]
u10m = file.variables["UGRD_P0_L103_GLL0"]
v10m = file.variables["VGRD_P0_L103_GLL0"]
rf6h = file.variables["APCP_P8_L1_GLL0_acc6h"]
lat  = file.variables["lat_0"]
lon  = file.variables["lon_0"]


plot_title = "GFS 6-h Rain + 10-m Wind + PMSL (Initial time=" + pmsl.initial_time
+ "; Forecast hour: " + '{:d}'.format(pmsl.forecast_time[0]) + ")"

# create a "workstation" to draw graphics
wks = Ngl.open_wks(wks_type,out_img_prefix,None)
```

香港天文台
HONG KONG OBSERVATORY

2. Define and set up "resources" for each field

```
# resources for rainfall
rain_res = Ngl.Resources()
rain_res.nglDraw  = False
rain_res.nglFrame = False

rain_rgb = np.array([[255,255,255],[220,220,220],[180,180,180],
                     [  0,142,  0],[  1,197,  1],[  2,253,  2],[253,248,  2],
                     [229,188,  0],[253,149,  0],[253,  0,  0],[212,  0,  0],
                     [188,  0,  0],[248,  0,253],[152, 84,198]], np.float32) / 255.0

rain_levels = [ 0.2, 0.5, 1.0, 2.0, 5.0, 10.0, 15.0, 20.0, 30.0, 40.0, 50.0, 70.0, 100.0]

rain_res.mpFillOn                    = False
rain_res.mpGeophysicalLineColor      = "Black" #"Grey18"
rain_res.mpGeophysicalLineThicknessF = 3.0
rain_res.mpGridAndLimbOn             = False    # turn off map grid line

rain_res.cnLinesOn                   = False
rain_res.cnLineLabelsOn              = False
rain_res.cnFillOn                    = True
rain_res.cnFillMode                  = "RasterFill"        # speed up plotting
rain_res.trGridType                  = "TriangularMesh"
rain_res.cnFillPalette               = rain_rgb
rain_res.cnLevelSelectionMode        = "ExplicitLevels"
rain_res.cnLevels                    = rain_levels

rain_res.lbOrientation               = "horizontal"
rain_res.lbBoxSeparatorLinesOn       = False

rain_res.tiMainString                = plot_title
rain_res.tiMainFontHeightF           = 0.0105

rain_res.sfXCStartV = float(min(lon))
rain_res.sfXCEndV   = float(max(lon))
rain_res.sfYCStartV = float(max(lat))
rain_res.sfYCEndV   = float(min(lat))
```

香港天文台
HONG KONG OBSERVATORY

2. Define and set up "resources" for each field

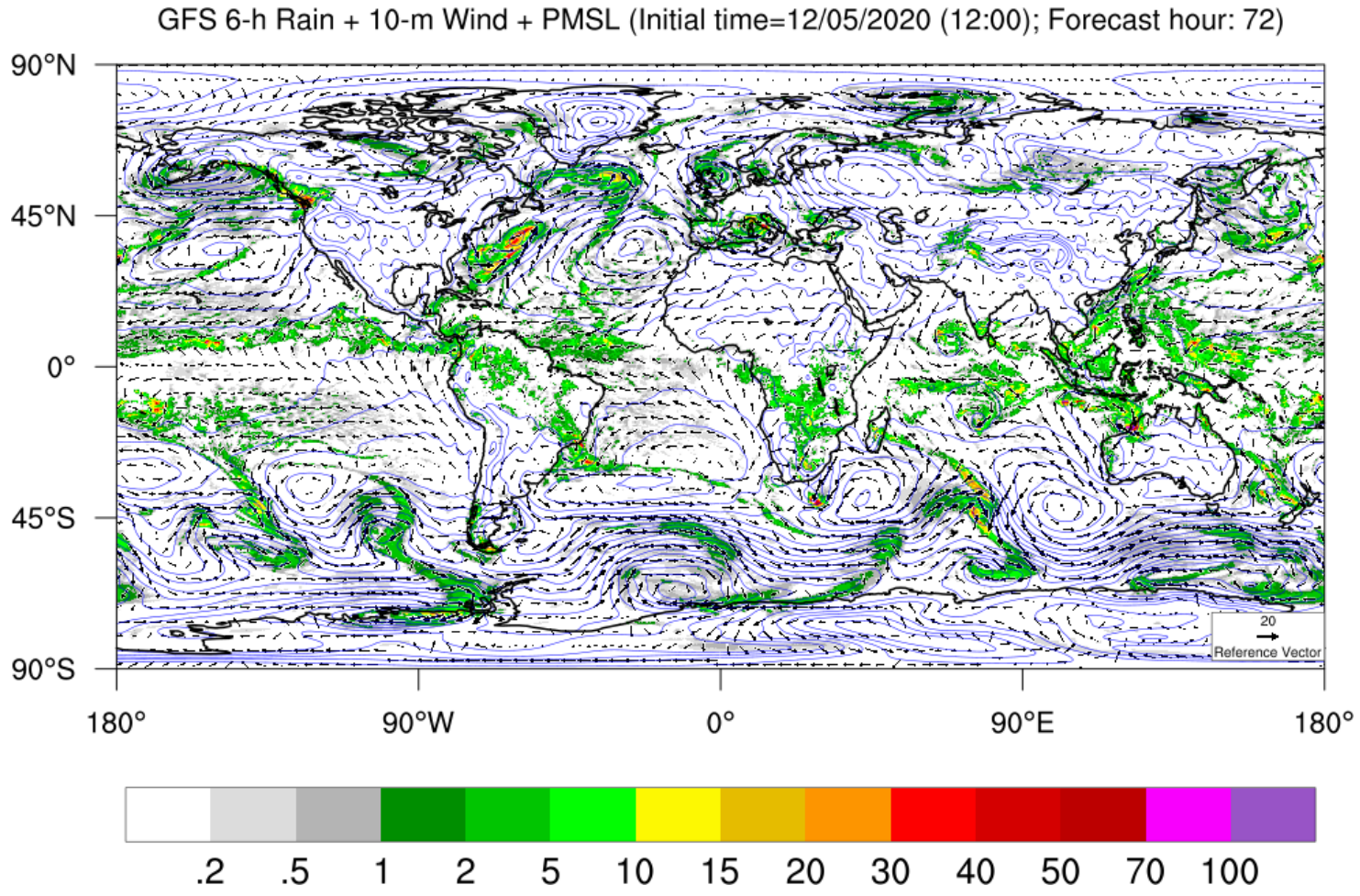
```
# vector resources:
wind_res                          = Ngl.Resources()
wind_res.nglDraw                  = False
wind_res.nglFrame                 = False

wind_res.vfXCStartV = float(min(lon))
wind_res.vfXCEndV   = float(max(lon))
wind_res.vfYCStartV = float(max(lat))
wind_res.vfYCEndV   = float(min(lat))

wind_style = "LineArrow" # "WindBarb" / "LineArrow" / "CurlyVector"

if ( wind_style == "WindBarb" ) :
   wind_res.vcGlyphStyle              = wind_style
   wind_res.vcRefLengthF              = 0.01            # define length of vec ref
   wind_res.vcMinDistanceF            = 0.010
   wind_res.vcWindBarbLineThicknessF = 1.5
   wind_res.vcRefAnnoOn               = False

else:
   # for vector
   wind_res.vcFillArrowsOn           = True
   wind_res.vcRefMagnitudeF          = 20.0            # define vector ref mag
   wind_res.vcRefLengthF             = 0.015           # define length of vec ref
   wind_res.vcMinFracLengthF         = 0.3
   wind_res.vcMinDistanceF           = 0.010
   wind_res.vcRefAnnoOrthogonalPosF = -0.20
   wind_res.vcRefAnnoFontHeightF     = 0.005
   wind_res.vcLineArrowThicknessF    = 3.0
```

```
# contour for MSLP:
pmsl_res = Ngl.Resources()
pmsl_res.nglDraw  = False
pmsl_res.nglFrame = False

pmsl_res.sfXCStartV = float(min(lon))
pmsl_res.sfXCEndV   = float(max(lon))
pmsl_res.sfYCStartV = float(max(lat))
pmsl_res.sfYCEndV   = float(min(lat))

pmsl_value = pmsl.get_value()
pmsl_value = pmsl_value * 0.01
pmsl_levels = np.arange(900., 1080., 4.0)

pmsl_res.cnLinesOn                    = True
pmsl_res.cnLineLabelsOn               = False
pmsl_res.cnFillOn                     = False
pmsl_res.cnLevelSelectionMode         = "ExplicitLevels"
pmsl_res.cnLevels                     = pmsl_levels
pmsl_res.cnLineColor                  = "Blue"
pmsl_res.cnInfoLabelOn                = False
```

香港天文台
HONG KONG OBSERVATORY

3. Generate plot of each field, overlay and draw the composite

```
# plot:
pmsl_plot = Ngl.contour(wks,pmsl_value,pmsl_res)
wind_plot = Ngl.vector(wks,u10m,v10m,wind_res)
rain_plot = Ngl.contour_map(wks,rf6h,rain_res)

Ngl.overlay(rain_plot,wind_plot)
Ngl.overlay(rain_plot,pmsl_plot)
Ngl.maximize_plot(wks, rain_plot)


Ngl.draw(rain_plot)
Ngl.frame(wks)
```

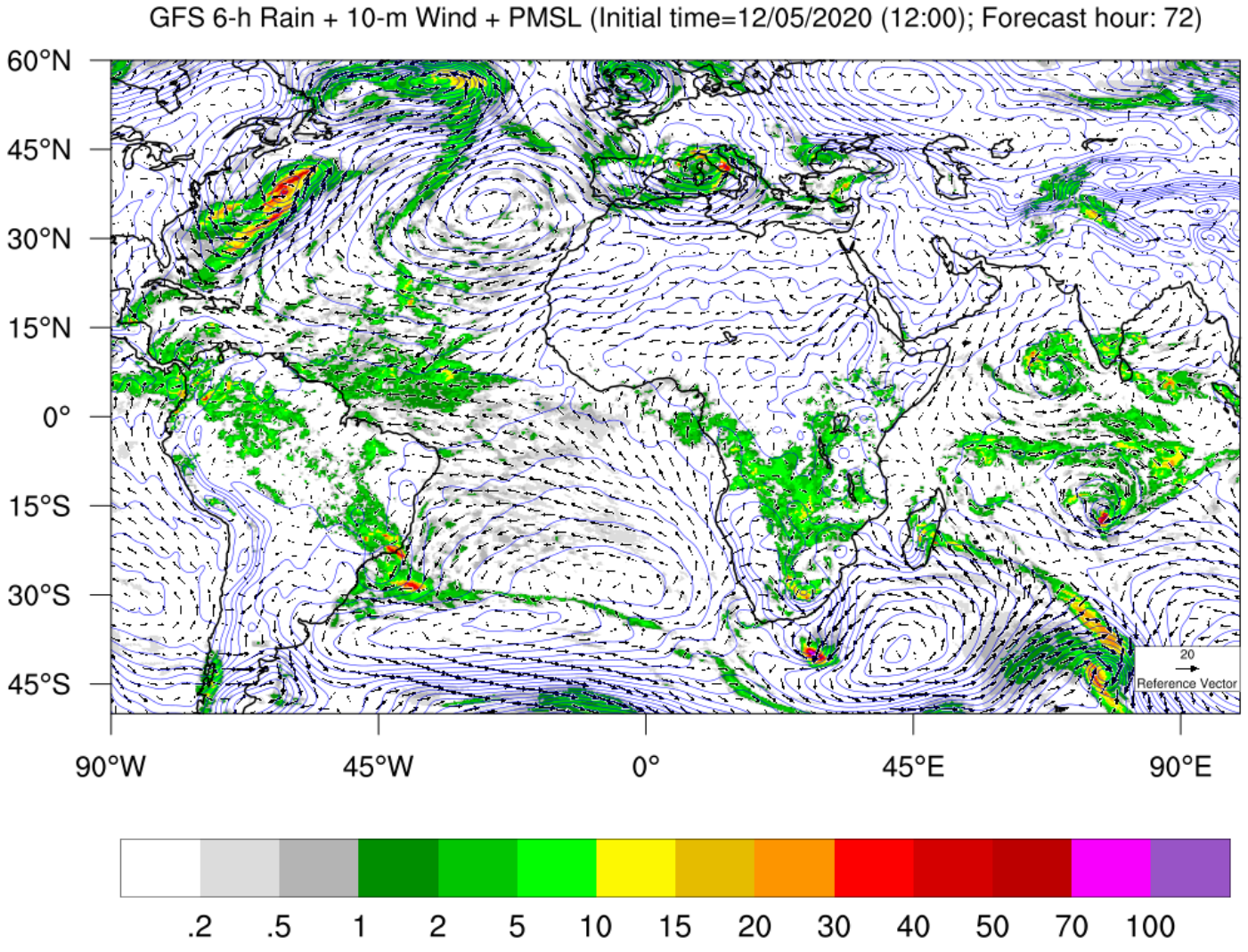


ex4_sfc_global.png

香港天文台
HONG KONG OBSERVATORY

GFS_ex5.py

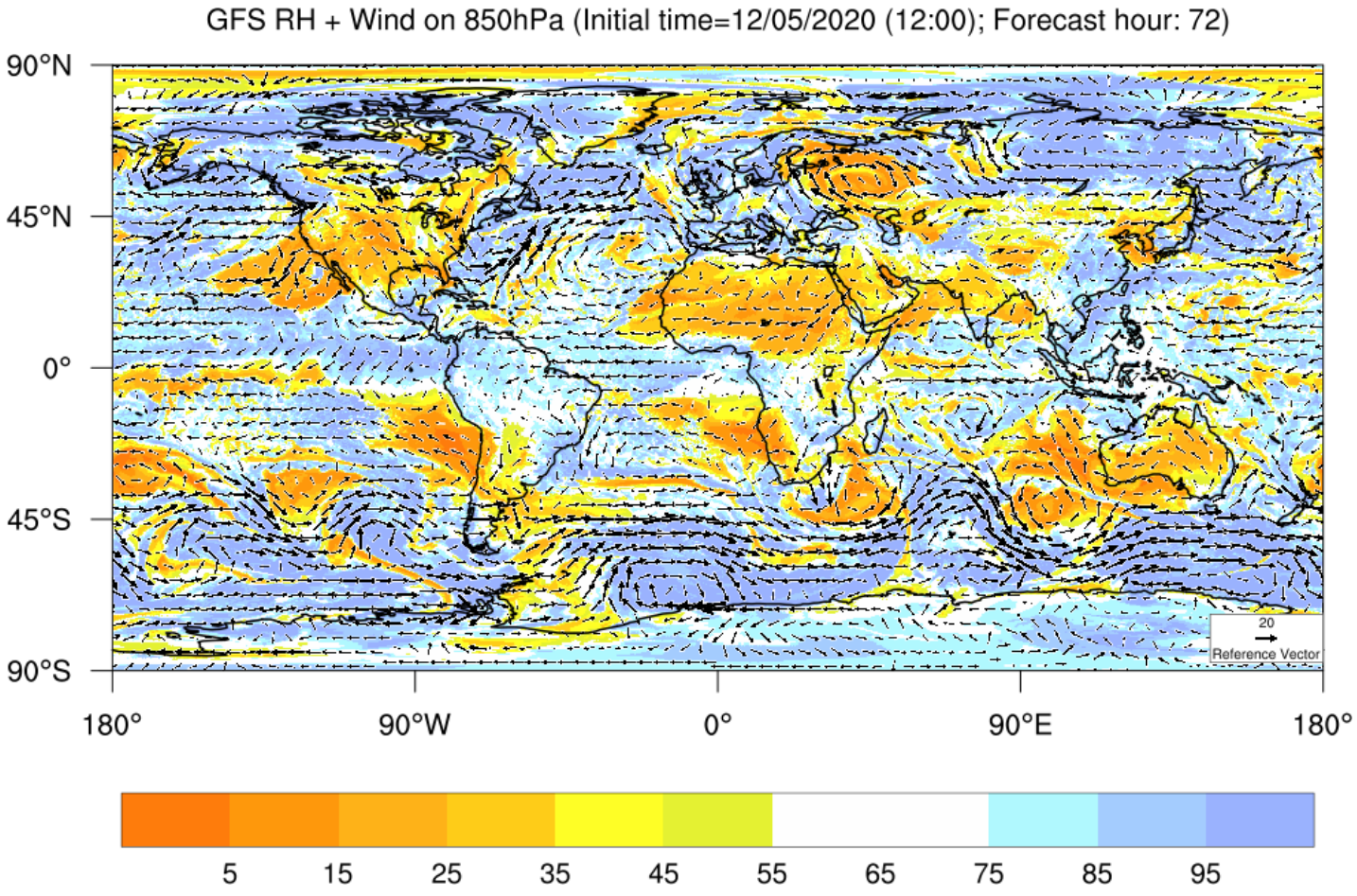
Modified from GFS_ex4.py using area extraction function in “exlib.py”



ex5_sfc_region.png

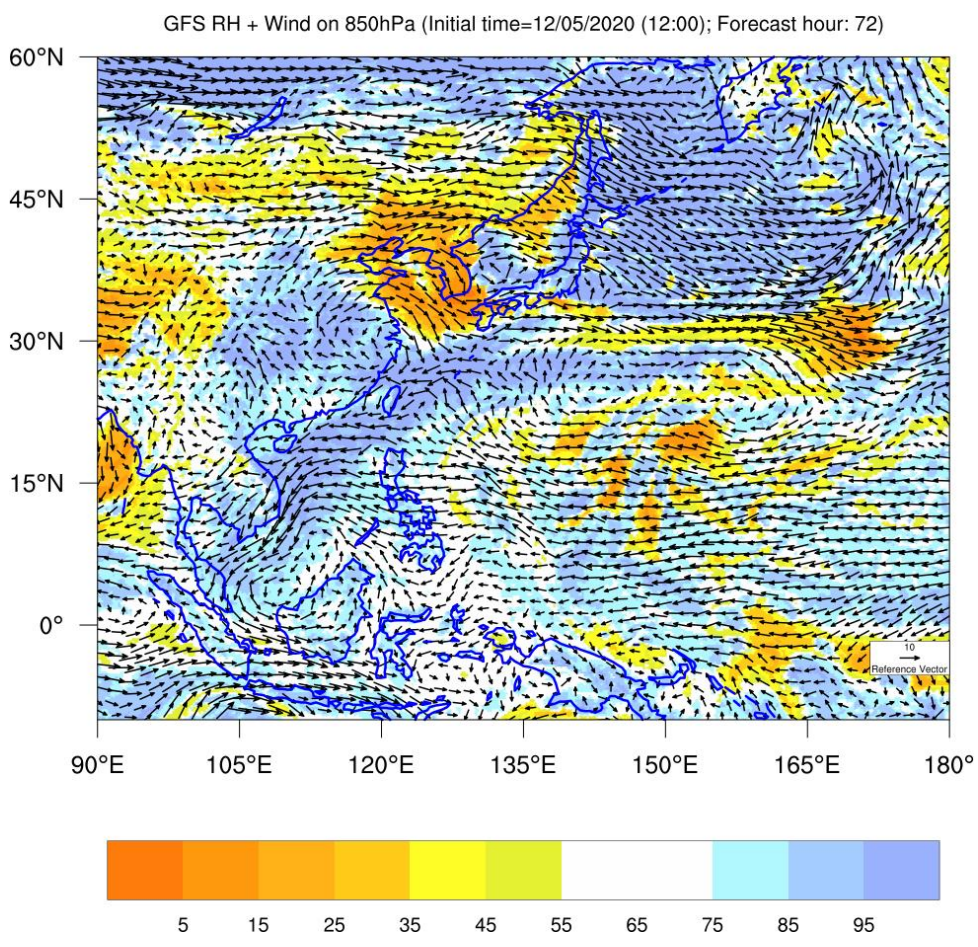
香港天文台
HONG KONG OBSERVATORY

GFS_ex6.py



ex6_rh_wind.png

GFS_ex7.py



ex7_rh_wind_region.png

香港天文台
HONG KONG OBSERVATORY

GFS_ex6.py

```
import os, sys, types
import numpy  as np

import Ngl, Nio

f_path         = "/r2/tutorial/data/2020120512"
gfs_in         = "gfs_20201205_12z_0p25_f072.grib2"
out_img_prefix = "ex6_rh_wind"   # output image file prefix
wks_type       = "png"

# read GFS GRIB file:
f_name=f_path+"/"+gfs_in

file = Nio.open_file(f_name, "r")

rh_lvl = file.variables["RH_P0_L100_GLL0"]
uu_lvl = file.variables["UGRD_P0_L100_GLL0"]
vv_lvl = file.variables["VGRD_P0_L100_GLL0"]
lat  = file.variables["lat_0"]
lon  = file.variables["lon_0"]
lvl  = file.variables["lv_ISBL0"]

plv = 850

rh = rh_lvl['lv_ISBL0|%f' % (plv*100) ]
uu = uu_lvl['lv_ISBL0|%f' % (plv*100) ]
vv = vv_lvl['lv_ISBL0|%f' % (plv*100) ]


plot_title = "GFS RH + Wind on " + '{:d}'.format(plv) + "hPa (Initial time=" + rh_lvl.initial_time  +
"; Forecast hour: " + '{:d}'.format(rh_lvl.forecast_time[0]) + ")"


# create workstation to draw graphics
wks = Ngl.open_wks(wks_type,out_img_prefix,None)
```

香港天文台
HONG KONG OBSERVATORY

```
# resources for rainfall
rh_res = Ngl.Resources()
rh_res.nglDraw  = False
rh_res.nglFrame = False

rh_rgb = np.array([[254,124, 12],[254,152, 12],[254,178, 24],[254,204,
24],[254,254, 38],[228,241, 50],
[254,254,254],[254,254,254],[176,248,254],[164,204,254],[154,178,254]],
np.float32) / 255.0

rh_levels = np.arange( 5., 105., 10.0)


rh_res.mpFillOn                    = False
rh_res.mpGeophysicalLineColor      = "Black" #"Grey18"
rh_res.mpGeophysicalLineThicknessF = 3.0
rh_res.mpGridAndLimbOn             = False    # turn off map grid line

rh_res.cnLinesOn                   = False
rh_res.cnLineLabelsOn              = False
rh_res.cnFillOn                    = True
rh_res.cnFillMode                  = "RasterFill"        # speed up plotting
rh_res.trGridType                  = "TriangularMesh"
rh_res.cnFillPalette               = rh_rgb
rh_res.cnLevelSelectionMode        = "ExplicitLevels"
rh_res.cnLevels                    = rh_levels

rh_res.lbOrientation               = "horizontal"
rh_res.lbBoxSeparatorLinesOn       = False
rh_res.lbLabelFontHeightF          = 0.01

rh_res.tiMainString                = plot_title
rh_res.tiMainFontHeightF           = 0.0105

rh_res.sfXCStartV = float(min(lon))
rh_res.sfXCEndV   = float(max(lon))
rh_res.sfYCStartV = float(max(lat))
rh_res.sfYCEndV   = float(min(lat))
```

```
# vector resources:
wind_res                           = Ngl.Resources()
wind_res.nglDraw                   = False
wind_res.nglFrame                  = False

wind_res.vfXCStartV = float(min(lon))
wind_res.vfXCEndV   = float(max(lon))
wind_res.vfYCStartV = float(max(lat))
wind_res.vfYCEndV   = float(min(lat))

wind_style = "LineArrow"   #"WindBarb" / "LineArrow" / "CurlyVector"

if ( wind_style == "WindBarb" ) :
   wind_res.vcGlyphStyle             = wind_style      #"WindBarb"
   wind_res.vcRefLengthF             = 0.01            # define length of vec ref
   wind_res.vcMinDistanceF           = 0.010
   wind_res.vcWindBarbLineThicknessF = 1.5
   wind_res.vcRefAnnoOn              = False

else:
   # for vector
   wind_res.vcFillArrowsOn           = True
   wind_res.vcRefMagnitudeF          = 20.0            # define vector ref mag
   wind_res.vcRefLengthF             = 0.015           # define length of vec ref
   wind_res.vcMinFracLengthF         = 0.3
   wind_res.vcMinDistanceF           = 0.010
   wind_res.vcRefAnnoOrthogonalPosF  = -0.20
   wind_res.vcRefAnnoFontHeightF     = 0.005
   wind_res.vcLineArrowThicknessF    = 3.0


# plot:
wind_plot = Ngl.vector(wks,uu,vv,wind_res)
rh_plot = Ngl.contour_map(wks,rh,rh_res)

Ngl.overlay(rh_plot,wind_plot)
Ngl.maximize_plot(wks, rh_plot)


Ngl.draw(rh_plot)
Ngl.frame(wks)
```

香港天文台
HONG KONG OBSERVATORY

WRF_ex8.py

```
import Nio, Ngl, os, sys
import numpy as np
from wrf import getvar, get_pyngl

f_path         = "/r2/tutorial/data"
wrf_in         = "wrfout_d01_2018-09-15_12:00:00"
out_img_prefix = "ex8_wrf_dbz"   # output image file prefix
wks_type       = "png"

filename = f_path + "/" + wrf_in

#---Read data
a  = Nio.open_file(filename+".nc")  # Must add ".nc" suffix for Nio.open_file
#dbz = getvar(a,"dbz")   # 3-d reflectivity field
mdbz = getvar(a,"mdbz")  # maximum reflectivity

dbz_levels = np.arange(0., 70., 5.)
# Create the color table for radar reflectivity
dbz_rgb = np.array([[255,255,255],[4,233,231],
                    [1,159,244], [3,0,244],
                    [2,253,2], [1,197,1],
                    [0,142,0], [253,248,2],
                    [229,188,0], [253,149,0],
                    [253,0,0], [212,0,0],
                    [188,0,0],[248,0,253],
                    [152,84,198]], np.float32) / 255.0

#---Open file for graphics
wks_type = "png"
wks = Ngl.open_wks(wks_type,out_img_prefix)
```

```
# Set map options based on information in WRF output file
res = get_pyngl(mdbz)
res.tfDoNDCOverlay       = True           # required for native projection

#---Contour options
res.cnFillOn             = True           # turn on contour fill
res.cnLinesOn            = False          # turn off contour lines
res.cnLineLabelsOn       = False          # turn off line labels
res.cnFillMode           = "RasterFill"         # speed up plotting
res.cnFillPalette        = dbz_rgb
res.cnLevelSelectionMode = "ExplicitLevels"
res.cnLevels             = dbz_levels

res.lbOrientation      = "horizontal"   # default is vertical
res.pmLabelBarHeightF  = 0.08
res.pmLabelBarWidthF   = 0.65
res.lbTitleString      = "%s (%s)" % (mdbz.description, mdbz.units)
res.lbTitleFontHeightF = 0.015
res.lbLabelFontHeightF = 0.015

res.tiMainString       = "WRF simulated reflectivity~C~" + wrf_in
res.tiMainFont         = "helvetica-bold"
res.tiMainFontHeightF  = 0.01

#plot = Ngl.contour_map(wks,dbz[1,:,:],res)
plot = Ngl.contour_map(wks,mdbz,res)

Ngl.end()
```

香港天文台
HONG KONG OBSERVATORY

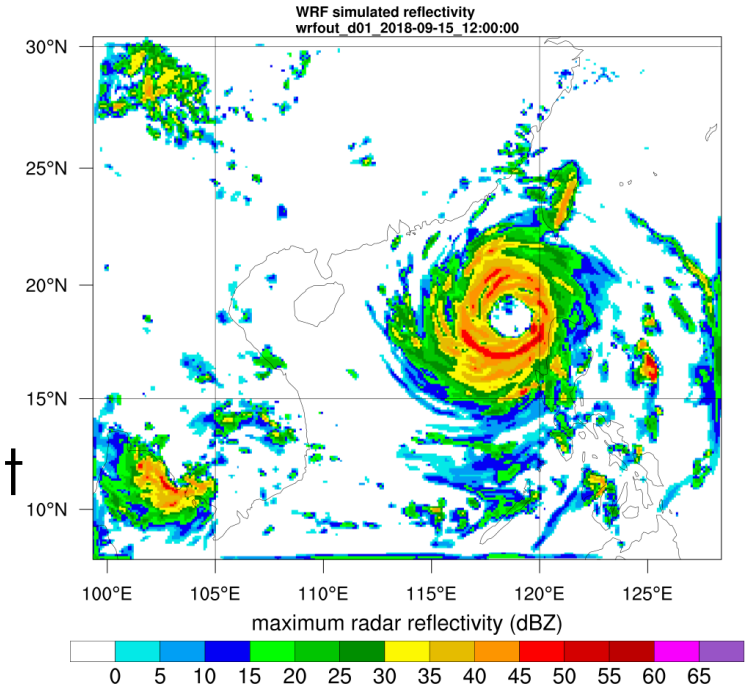
To generate prog chart:

1. Type : "`source /r2/anaconda3/0_conda_init.sh`" ← enable conda environment
   - your command prompt will be prefixed by (base)

2. Type : "`conda activate pyn_env`" ← activate PyNGL, PyNIO as installed
   - Change from (base) to (pyn_env)

3. Type : "`python WRF_ex8.py`"

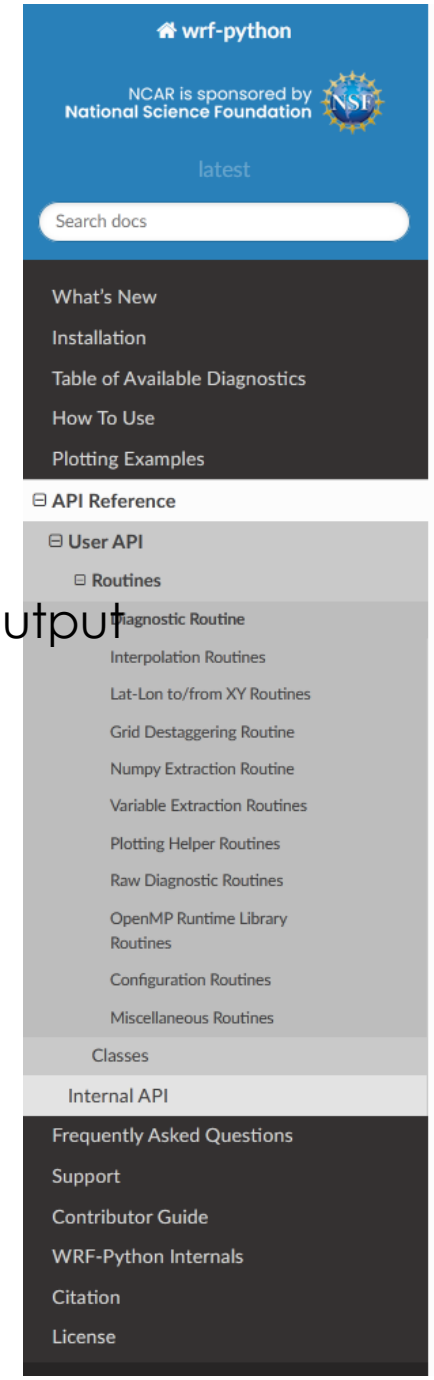
4. A image file "`ex8_wrf_dbz.png`" will be generat



香港天文台
HONG KONG OBSERVATORY

https://wrf-python.readthedocs.io/en/latest/user_api/generated/wrf.getvar.html

wrf.getvar :
return basic diagnostics from WRF model output



wrf.getvar

`wrf.getvar(wrfin, varname, timeidx=0, method='cat', squeeze=True, cache=None, meta=True, **kwargs)`

Returns basic diagnostics from the WRF ARW model output.

A table of all available diagnostics is below.

Variable Name	Description	Available Un
avo	Absolute Vorticity	10-5 s-1
eth/theta_e	Equivalent Potential Temperature	K degC degF
cape_2d	2D CAPE (MCAPE/MCIN/LCL/LFC)	J kg-1 ; J kg-
cape_3d	3D CAPE and CIN	J kg-1
ctt	Cloud Top Temperature	degC K degF
cloudfrac	Cloud Fraction	%
dbz	Reflectivity	dBZ
mdbz	Maximum Reflectivity	dBZ
geopt/geopotential	Geopotential for the Mass Grid	m2 s-2

香港天文台
HONG KONG OBSERVATORY

Any other choices of plotting software (using Python) ?

matplotlib
Version 3.3.3

Installation | Documentation | Examples | Tutorials | Contributing

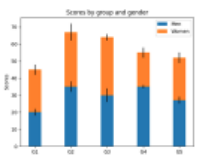
home | contents » Gallery

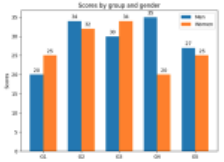
Gallery

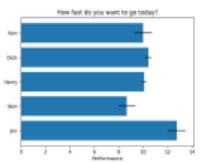
This gallery contains examples of the many things you can do with Matplotlib. Click on any image to see the full image and source code.

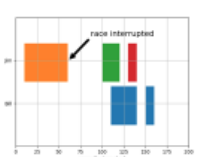
For longer tutorials, see our tutorials page. You can also find external resources and a FAQ in our user guide.

Lines, bars and markers

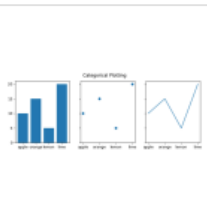

Stacked bar chart

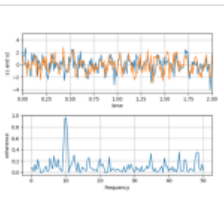

Grouped bar chart with labels

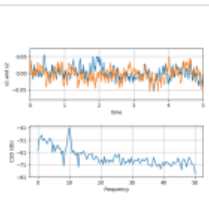

Horizontal bar chart

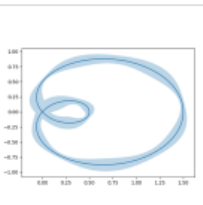


Broken Barh









Iris v2.4

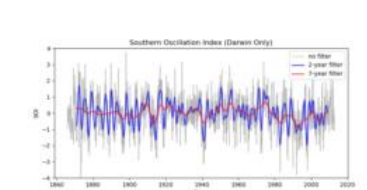
A powerful, format-agnostic, community-driven Python library for analysing and visualising Earth science data.

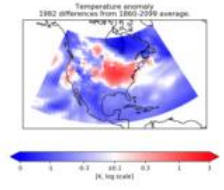
home | examples | gallery | contents |

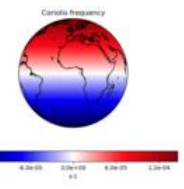
Click on any image to see full size image and source code

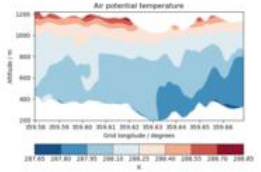
- Gallery
  - General
  - Meteorology
  - Oceanography

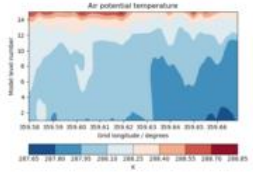
General

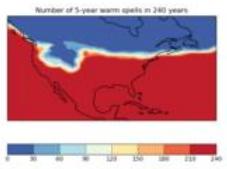


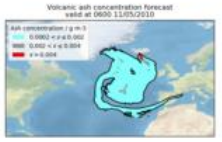


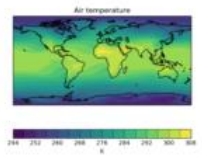


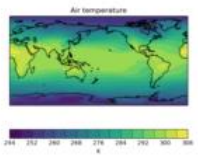


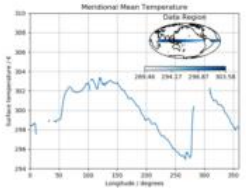


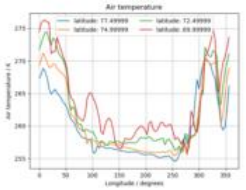












香港天文台
HONG KONG OBSERVATORY

Q&A

- Supplementary:
  - /r2/tutorial/python/supp
    - ex1_mslp/ ➔ change input of GRIB2 file to command line argument (use full path)
    - ex2_tt2m/ ➔ demonstrate plotting of global 2m temperature forecast
    - gfs/ ➔ contain a script get_gfs.sh to download the data (surface + isobaric levels) with input options of selected area, output location and output file name
    - readgrid/ ➔ demonstrate PyNIO functions on showing the names, attributes and keys of the data fieldcontained in the GFS GRIB2 data.