Diagnostic Plotting

JJ Guerrette and Junmei Ban

18 APR 2019

Post processing and diagnostics

https://github.com/JCSDA/DAdiagnostics

So far we have 5 python plotting scripts that have been used to analyze mpas-bundle applications:

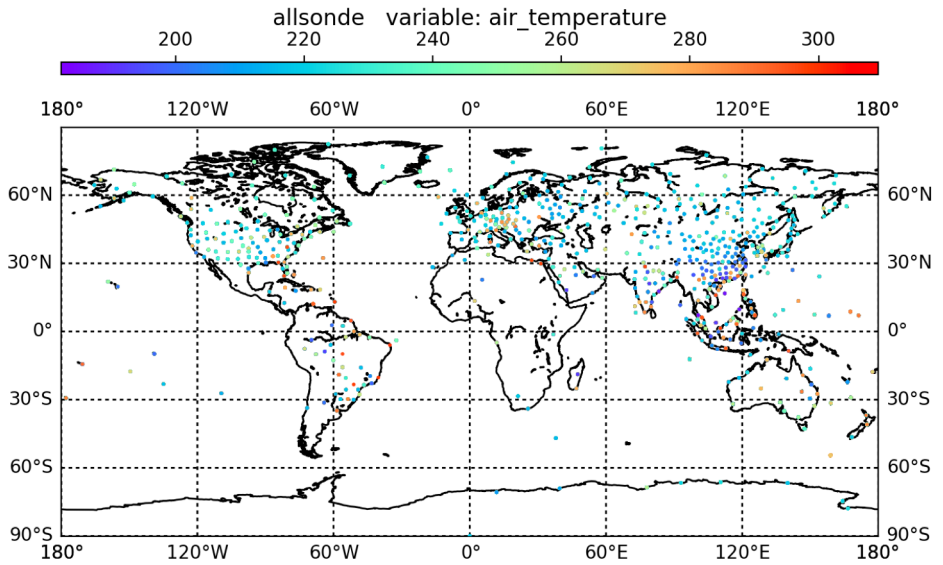
- plot_obs_nc_loc.py: spatial distribution and values of observations
- plot_BUMP_diag.py: variance and length scale plotting (needs fixes for latest BUMP changes)
- plot_cost_grad.py: quadratic cost function
- plot_diag_omaomb.py: vertical profile (radiosonde, aircraft, gnssro) and scatter plots (radiances) of OMA and OMB with statistics
- plot_diag_omaomb_timeserial.py: similar to above; used for cycling experiments (needs to be generalized for different obstypes)

Example: plot_obs_nc_loc.py

```
23      obsfiles = []
24      for files in os.listdir('../Data/'):
25          if fnmatch.fnmatch(files, '*_obs_*_m.nc4'):
26              obsfiles.append('../Data/'+files)
27      obsfiles.append('../Data/gnssro_obs_2018041500_s.nc4')
28      print 'File name list=', obsfiles
29      for file_name in obsfiles:
30          nc = Dataset(file_name, 'r')
31          print 'Plotting:', file_name
32          obstype = str(file_name[8:].split("_")[:1])
33          if obstype == "['gnssro']":
34              latnc = nc.variables['latitude']
35              lonnc = nc.variables['longitude']
36          else:
37              latnc = nc.variables['latitude@MetaData']
38              lonnc = nc.variables['longitude@MetaData']
39
40          lonnc = numpy.asarray(lonnc)
41          for i in range(len(lonnc)):
42              if lonnc[i] > 180:
43                  lonnc[i] = lonnc[i]-360
44
45          varlist = nc.variables.keys()
46          #select variables with the suffix 'ObsValue'
47          obslist = [obs for obs in varlist if (obs[-8:] == 'ObsValue')]
```

Directory structure and obs file template
Currently using “medium” obs files in our ctests, except gnssro

get latitude/longitude arrays

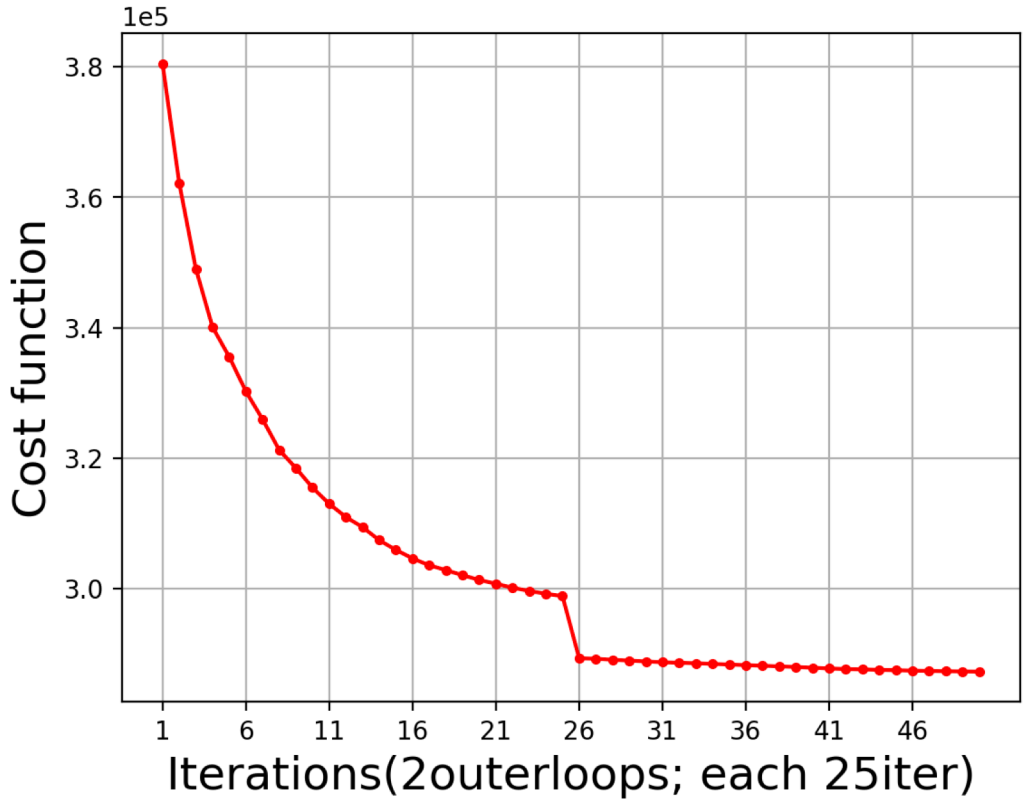


obslist contains variable names with “ObsValue” suffix

Example: plot_cost_grad.py

```
19  def readdata():
20
21      dalogfiles = []
22      for files in os.listdir('../testoutput/'):
23          if fnmatch.fnmatch(files, '?d*.test.log.out'):
24              dalogfiles.append('../testoutput/'+files)
25              #print(dalogfiles)
                    ⋮
54          plot(forx,cost,iters,VAR1,dalogfile[14:])
55          plot(forx,grad,iters,VAR2,dalogfile[14:])
56
57  def plot(forx,value,iters,VAR,expname):
58      fig, ax1 = plt.subplots()
59
60      ax1.set_xlabel('Iterations',fontsize=16)
61      ax1.set_xticks(forx[::2])
62      ax1.set_xticklabels(iters.astype(np.int)[::2]) #,rotation=45)
63      ax1.set_ylabel('%s'%VAR,fontsize=16)
```

Directory structure and log file template
MPAS: script executed from
build/mpas-bundle/mpas/test/graphics



Example: plot_diag_omaomb.py

```
24  def readdata():
25
26      print_fmt = 'png' #lower fidelity, faster
27      #print_fmt = 'pdf' #higher fidelity, slower
28
29      profile_group  = ['sonde','aircraft','satwind','gnssro']
30      radiance_group = ['amsua_n18','amsua_n19','amsua_metop-a','amsua_metop-b']
31      #dummy_group   = ['dummy_obstype1']
32
33      all_groups = []
34      all_groups.append(profile_group)
35      all_groups.append(radiance_group)
36      #all_groups.append(dummy_group)
37      #all_groups.append(['dummy_obstype2'])
38
39      # Diagnostic oma/omb files located in diagdir
40      diagdir    = '../Data/'
41      diagprefix = 'obsout_'
42      diagsuffix = '_*.nc4'
43      # * in diagsuffix is 4-digit processor rank [0000 to XXXX]
44      #npedigits = 4
45
46      # Suffixes to required nc variables
47      omb_var = 'ombg'
48      oma_var = 'oman'
49      obs_var = 'ObsValue'
50      qc_var  = 'EffectiveQC' #Only for final outer iteration currently
51
52      obsoutfiles = []
53      for files in os.listdir(diagdir):
54          #print(files)
55          if fnmatch.fnmatch(files, diagprefix+'*'+diagsuffix):   # 1tile
56              obsoutfiles.append(diagdir+files)
57      #print(obsoutfiles)
```

Add vertically-distributed obs

Add radiance instruments

Add groups that require unique plotting technique

Directory structure and obsout file template

Assuming IODA obsout files have “_”. Need to generalize for “-” and others

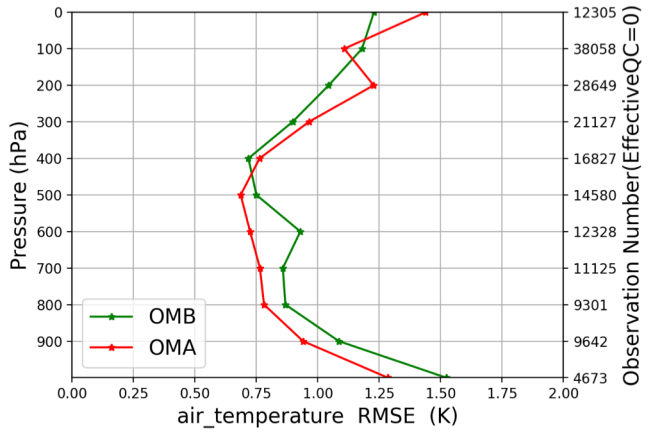
Definitions of OMB and OMA suffixes, taken from application YAML file

Works with multiple IODA output files from separate processors (not shown)

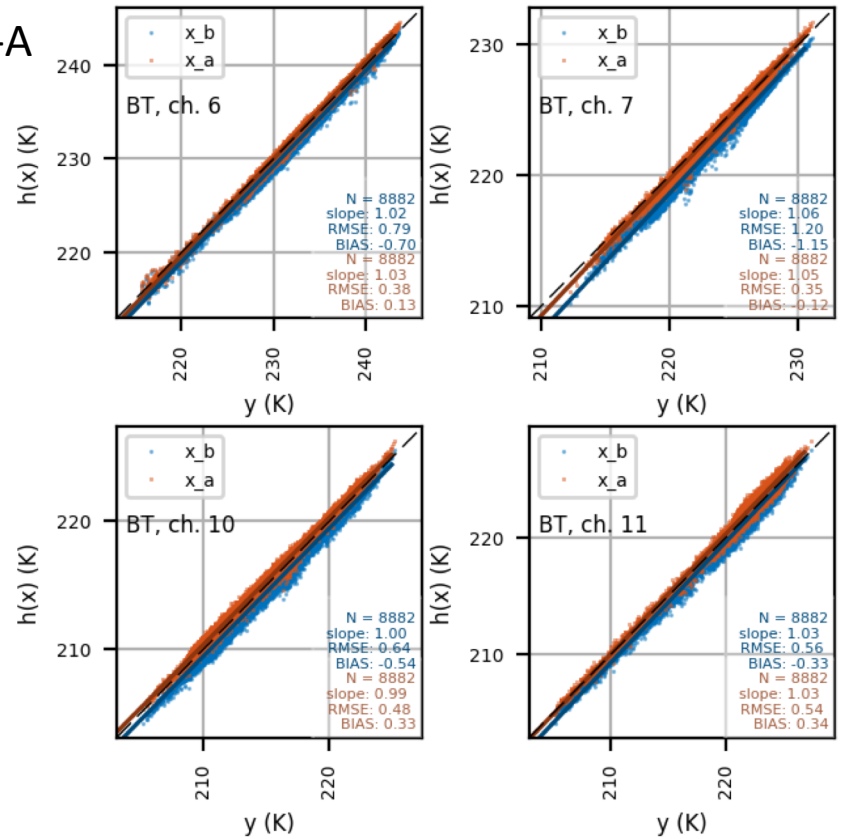
Example: plot_diag_omaomb.py

```python
def readdata():

    print_fmt = 'png' #lower fidelity, faster
    #print_fmt = 'pdf' #higher fidelity, slower

    profile_group  = ['sonde','aircraft','satwind','gnssro']
    radiance_group = ['amsua_n18','amsua_n19','amsua_metop-a','amsua_metop-b']
    #dummy_group   = ['dummy_obstype1']

    all_groups = []
    all_groups.append(profile_group)
    all_groups.append(radiance_group)
    #all_groups.append(dummy_group)
    #all_groups.append(['dummy_obstype2'])

    # Diagnostic oma/omb files located in diagdir
    diagdir    = '../Data/'
    diagprefix = 'obsout_'
    diagsuffix = '_*.nc4'
    # * in diagsuffix is 4-digit processor rank [0000 to XXXX]
    #npedigits = 4

    # Suffixes to required nc variables
    omb_var = 'ombg'
    oma_var = 'oman'
    obs_var = 'ObsValue'
    qc_var  = 'EffectiveQC' #Only for final outer iteration currently

    obsoutfiles = []
    for files in os.listdir(diagdir):
        #print(files)
        if fnmatch.fnmatch(files, diagprefix+'*'+diagsuffix):   # 1tile
            obsoutfiles.append(diagdir+files)
    #print(obsoutfiles)
```

Radiosonde, T



METOP-B, AMSU-A



Wrapping up

- So far these scripts work for mpas-bundle applications; probably changes are needed for others
- Please use these scripts however you like
- Relevant groups may want to discuss aims of this repository

- General functionalities for marine and/or atmosphere applications could be wrapped up in a library, similar to IODA convertors, or build off that existing framework