Regional Cooperation for
Limited Area Modeling in Central Europe

LACE
nwp central europe

Minimization practicals (Alaro)

Antonín Bučánek

ALADIN/HIRLAM common DA training week, 10-15 February 2019

DHMZ

SHMU

ARSO METEO
Slovenia

OMSZ

ZAMG

ROMANIA
ANM

Part A

Checking the convergence of cost function

A) Checking the convergence of cost function

- This exercise should show how to extract components of cost function out of listing (NODE.001_01)
- How to visualize the extracted values
- It shows an example when the minimizer was not converging in default **50 iterations**
- In this case we will work with files **listing.e131.*** which were just renamed **NODE.001_01**

A) Setup

Login to cca/ccb

```
-> ssh -Y cca
```

Copy the training sample

```
-> cd $PERM   # or $SCRATCH
-> cp -rp /perm/ms/spsehlam/hlam/daTraining/Day_3/costfc_conver .
-> cd costfc_conver/
```

A) Exercise 1: plot the cost function convergence

1.1) To extract Jo, Jb and J values of cost function from listing (NODE.001_01 = listing.e131.good)

GREPCOST – extracts Jo, Jb per iteration

GREPGRAD – extracts full cost function J per iteration

```
-> grep "\<GREPCOST\>" listing.e131.good | sed -e '$s/^ /# /' >cost.good
-> grep "\<GREPGRAD\>" listing.e131.good | sed -e '$s/^ /# /' >grad.good
```

"sed" command is used to comment out last line of grep, to not plot that line in gnuplot

1.2) Look to cost.good and grad.good and see number of iteration used during the minimization, remember that default setting of quasi-Newton method is NITER=50, NSIMU=60. Does this case converged before the hard stop was reached?
You should find that yes it converged after 42 iteration

1.3) Plot the convergence of J, Jb, Jo

you can choose your way of plotting, if you are not familiar with plotting tools you can use gnuplot interactively (use "exit" to leave gnuplot)

```
-> gnuplot
gnuplot> plot "cost.good" using 4:6 with line title "Jo.good", "" using 4:7 with line title "Jb.good", "grad.good"  u 4:7 w l t "J.good"
```

A) Exercise 2: examine case with slow convergence of cost function

You can imagine a case where the cost function is not converging, e.g. the minimizer stop criteria is not fulfilled for many iterations.

Here is shown example from operational run where the minimizer was stopped on default number of iteration and than rerun with NITER=200

2.1) Extract Jo, Jb and J values of cost function from listing of minimization (listing.e131.slow), again

```
-> grep "\<GREPCOST\>" listing.e131.slow | sed -e '$s/^ /# /' > cost.slow
-> grep "\<GREPGRAD\>" listing.e131.slow | sed -e '$s/^ /# /'> grad.slow
```

2.2) Now look to cost.slow and grad.slow and see number of iteration used during the minimization. Now you should see that the hard limit of number of iteration was used in minimization.

A) Exercise 2: examine case with slow convergence of cost function

2.3) Extract Jo, Jb and J values from listing.e131.slow_niter200 where the number of iteration was increased from default (50) to 200.

```
-> grep "\<GREPCOST\>" listing.e131.slow_niter200 | sed -e '$s/^ /# /' > cost.slow_n200
-> grep "\<GREPGRAD\>" listing.e131.slow_niter200 | sed -e '$s/^ /# /'> grad.slow_n200
```

You can compare these extracted Jo, Jb, J with values from point 2.1) either by plot or vimdiff/tkdiff. You should see that only difference is continuation of minimization algorithm compared to point 2.1). This exercise is not intended to give you advice how to solve the problem when the minimization converge slowly but just to point out that it is recommended to monitor the convergence of cost function.

```
->gnuplot
gnuplot> plot "cost.slow" using 4:6 with point title "Jo.slow", "" using 4:7 with point title "Jb.slow", "grad.slow" u 4:7 w p t "J.slow", "cost.slow_n200" using 4:6 with line title "Jo.slow.n200", "" using 4:7 with line title "Jb.slow.n200", "grad.slow_n200" u 4:7 w l t "J.slow.n200"
```

Part B

Single observation experiments

B) Single obs experiment (bator, screening, minimization)

- This exercise should show all steps before the analysis is produced, i.e. bator, screening, minimization.
- Only single observation is used for demonstration
- We will use single temperature obs at 500hPa over Prague (1K difference to FG)
- We will look to ODB on first guess departure and analysis departure
- We will plot maps and cross section of analysis increment (R scripts).

B) Setup

Login to cca/ccb

-> ssh -Y cca

Copy the training sample

-> cd $PERM   # or $SCRATCH

-> cp -rp /perm/ms/spsehlam/hlam/daTraining/Day_3/sample_alaro .

-> cd sample_alaro/

-> ls

You should see directories (**bin, data, etc, namelist, scr_3dvar, scr_singleobs**)

bin directory where scripts for plotting are stored

data directory contains input files (first guess, observations) and output files (listings, ECMAs, CCMAs, analyses)

etc directory contains B matrix, constant files, climatological files

namelist directory contains all namelist which are static

scr_3dvar directory contains script to run full observation analysis

scr_singleobs directory contains scripts to run single observation analyses

B) Setup 2

Enter the single observation directory

-> cd scr_singleobs

-> ls

You should see files: test_bator_singleobs, test_screening_singleobs, test_minim_singleobs, test_fullpos_singleobs, run.sh

test_bator_singleobs is script to run Bator

test_screening_singleobs is script to run Screening

test_minim_singleobs is script to run Minimization

test_fullpos_singleobs is script to run Fullpos (post processing for plotting)

run.sh is script which submits all previously named scripts to cca/ccb

the run.sh is prepared in such a way that it submits bator, screening and minim to cca/ccb at once but the tasks are waiting in the queue to be executed in the correct order (bator, screening, minim).

If you open **run.sh** you will see that there is exported **environmental variable** which sets which **kind of single observation** test will be executed.

B) Setup 3

Look to the minimization script to check what we have learned in the morning
-> vi **test_minim_singleobs**

- At the top of the file you should see #PBS the submit directives
- Below setup of MPI/OpenMP paralelization
- Then launching directives for MASTERODB
- Working directory is $TMPWORKDIR
- General ODB setting
- Copy of Bmatrix (stabal96.cv, stabal96.bal), amv_p_tracking_error
- Copy of first guess (${YYYY}${MM}${DD}${NT}_guess)
- Copy of CCMA after screening
- Setup of NAMELIST (minim.namel) -> fort.4
- Minimization ODB setting
- Run MASTERODB (ALADIN executable)
- “cat” of listing
- Storing output analysis, CCMA, listing

B) Execution of 3DVAR (run.sh)

Run in the first single observation experiment

-> **./run.sh**

To submits bator, screening, minimization and fullpos to the cca/ccb queue.

If the script are not putted to the queue due to billing account and you are from country which has its own billing units please remove this line from all 4 scripts (bator, screen,minim, fullpos)

#PBS -l EC_billing_account=hlamharm

Check the tasks are finished before you submit the same experiment again: ---> **qstat –u <username>**

If you need to run just one task again (after crash), e.g. test_minim_singleobs

-> **(export type=<obs_type> ; qsub –v type test_fullpos_singleobs)**

B) Exercise 1: run single TEMP observation assimilation at 500hPa with 1K difference to guess

1.1) Run the script run.sh

```
-> ./run.sh
```

1.2) Check that runs did not failed (bator, screening, minim, fullpos)

```
-> grep "exit 0" <jobout>
```

1.3) Check the size of increment in observation location

```
-> module load odb
-> mkdir tmp
-> cd tmp
-> tar xvf ../../data/output_singleobs/S00_CCMA_minim_2019010700.tar
-> cd CCMA
-> odbsql -q "SELECT fg_depar,an_depar,obsvalue FROM hdr,body"
fg_depar@body an_depar@body obsvalue@body
1 0.48032651083824 248.68521878147
```

You should see that obs-analysis is almost half of the obs-firstGuess, this means that the expected errors in model and in the observation are comparable.

B) Exercise 1: run single TEMP observation assimilation at 500hPa with 1K difference to guess

1.4) Plot the temperature analysis increment by R scripts (horizontal map, vertical cross section)

The R script are in **bin/R_plots/incr/** and **bin/R_plots/incr_cross**

- **incr.sh** produces analysis increment map of temperature at 500 hPa (increment.png)
- **incr_wind.sh** produces wind vectors on top of temperature increments (increment_wind.png)
- **cross.sh** produces cross section at specified lat,lon (cross_increment.meridional.png, cross_increment.zonal.png)
- **crossm.sh** produces cross section at specified lat,lon but Y-axis contains height above sea level (crossm_increment.meridional.png, crossm_increment.zonal.png)

The R scripts need 2 input files ANAL (FANAL) and INIT (FINIT).

B) Exercise 1: run single TEMP observation assimilation at 500hPa with 1K difference to guess

1.4) Plot the temperature analysis increment by R scripts (horizontal map, vertical cross section)

```
  -> mkdir -p tmp
  -> cd tmp
  -> ln -sf ../../data/output_singleobs/S00_analysis_2019010700 ANAL
  -> ln -sf ../../data/input/2019010700_guess INIT
  -> cp -f ../../bin/R_plots/incr/incr.sh .
  -> cp -f ../../bin/R_plots/incr/incr_wind.sh .
  -> cp -f ../../bin/R_plots/incr_cross/cross.sh .
  -> ./incr.sh
  -> ./incr_wind.sh
  -> ./cross.sh
```

open the output files and look on the increments

```
  -> display <outputFile>
```

You should see that the temperature increments are "symmetrical" -- consequence of homogeneity and isotropy of auto-covariances.

B) Exercise 1: run single TEMP observation assimilation at 500hPa with 1K difference to guess

1.4) Plot the temperature analysis increment by R scripts (horizontal map, vertical cross section)

# Plot the temperature analysis increment **in height instead of model levels**.

(fullpos is needed to get height/geopotential of model levels)

```
-> ln -sf ../../data/output_singleobs/S00_fullpos_2019010700 FANAL
-> ln -sf ../../data/input/2019010618_FPOSALAS+0006 FINIT
-> cp -f ../../bin/R_plots/incr_cross/crossm.sh .
-> ./crossm.sh
```

You should see increments that are not that symmetrical since we use a terrain following vertical coordinate in ARPEGE/IFS.

B) Exercise 1: run single TEMP observation assimilation at 500hPa with 1K difference to guess

1.5) Adapt the R scripts and plot humidity increments the name of the fields are: **HUMI.SPECIFI**

```
-> mkdir -p tmp
  -> cd tmp
  -> ln -sf ../../data/output_singleobs/S00_analysis_2019010700 ANAL
  -> ln -sf ../../data/input/2019010700_guess INIT
  -> sed -e 's/parameter=.*$/parameter="HUMI.SPECIFI"/g' -e
's/outname=.*$/outname="increment_humi"/g' ../../bin/R_plots/incr/incr.sh > incr_humi.sh
  -> sed -e 's/outname=.*$/outname="cross_humi"/g' -e 's/TEMPERATURE/HUMI.SPECIFI/g'
../../bin/R_plots/incr_cross/cross.sh >cross_humi.sh
  -> chmod +x incr_humi.sh
  -> chmod +x cross_humi.sh
  -> ./incr_humi.sh
  -> ./cross_humi.sh
```

1.6) you can play with different single obs, see “type” variable in run.sh

B) Exercise 2: REDNMC extreme cases

In this exercise we will verify that REDNMC is scaling the B matrix, **larger RENMC less weight is given to first guess**, lower REDNMC more weight is given to first guess

2.1) Extreme case, very large REDNMC: **REDNMC=10000**

open run.sh and change type to "**rednmc_temp1K**" and execute it
-> **./run.sh**

If everything finished OK check the size of the resulting increment. How big it is? Is it expected?

```
-> module load odb
-> mkdir -p tmp2
-> cd tmp2
-> tar xvf ../../data/output_singleobs/S01_CCMA_minim_2019010700.tar
-> cd CCMA
-> odbsql -q "SELECT fg_depar,an_depar,obsvalue FROM hdr,body"
fg_depar@body an_depar@body obsvalue@body
1 6.1063395548899e-06 248.68521878147
```

B) Exercise 2: REDNMC extreme cases

2.2) Extreme case, very small REDNMC: **REDNMC=0**

Hints:
modify the script test_minim_singleobs is such a way that REDNMC would be set 0.0 in the namelist for minimization and run minimization again.
(modify line containing REDNMC=10000.0)
-> **(export type="rednmc_temp1K“ ; qsub –v type test_minim_singleobs)**

If everything finished OK check the size of the resulting increment again as in point 2.1. How big it is? Is it expected?

B) Exercise 3: SIGMAO_COEF extreme cases

In this exercise we will verify how SIGMAO_COEF influence the analysis, larger observation error less weight to observation, smaller observation error more weight to observations.

3.1) Extreme case, very large sigmao_coef: **SIGMAO_COEF=10000**

open run.sh and change type to "**sigmao_temp1K**“, execute it

```
-> ./run.sh
```

if everything finished OK check the size of the resulting increment. How big it is? Is it expected?

```
-> module load odb
-> mkdir -p tmp3
-> cd tmp3
-> tar xvf ../../data/output_singleobs/S02_CCMA_minim_2019010700.tar
-> cd CCMA
-> odbsql -q "SELECT fg_depar,an_depar,obsvalue FROM hdr,body"
fg_depar@body an_depar@body obsvalue@body
1 1 248.68521878147
```

B) Exercise 3: SIGMAO_COEF extreme cases

3.2) Extreme case, very small sigmao_coef: **SIGMAO_COEF=0**

You need to **modify** scripts **test_bator_singleobs, test_screening_singleobs, test_minim_singleobs** in such a way that SIGMAO_COEF would be set 0.0 in the namelist used in those scripts

(modify line containing SIGMAO_COEF=10000.0)

and run again

-> **./run.sh**

How big is the result in increment now? Is it expected?