MapGenerator: a wrapper for GraDs images around the Matplotlib plotting engine

Luca Telloli

Contents

1 Introduction

Historically, a collection of independent GRaDs scripts have been using inside the Caliope workflow to produce images of air-quality, emissions and meteorology from the output of simulations. As the number of simulated domains has grown and evolved into a larger collection of domain and subdomains, the usual copy-and-paste for the GraDs code rendered a collection of similar scripts with small differences and costly maintenance.

MapGenerator is an attempt to resolve this issue programmatically by producing a reusable code driven by a simple, human-readable, text-only configuration file.

1.1 General description of the software

MapGenerator is a Python software for generating images. The software reads data from one or more data sources (in the form of a data file, typically in netCDF format), and visualizes them onto a .gif image for one or more time steps. Additionally, the software produces an animated image by aggregating all the static images in the output directory into an animated .gif.

Inside the Caliope workflow, the software is currently used every day, after the CMAQ simulation of each domain, to produce air-quality and emissions images. There are currently 27 "packs" of images; one for the EU domain, one for the Iberian Peninsula + one for each of its 15 regions, one for Andalusia and one for Canary Island + one for each of its 7 islands. A total of 11 contaminants (5 for Air Quality + 6 for Emissions) is run for each pack, and for each contaminant 49 images are generated, plus a .gif animation, for a total of 14553 images and 297 animations.

2 Installation & Configuration

The source code of MapGenerator can be downloaded from the BSC svn server, at the following path: `https://svn.bsc.es/repos/earth/sds-was/mapgenerator/branches/caliope`.

2.1 Pre-requisites

The libraries have some prerequisites that should be met:

- a version of python 2.6.x or higher
- Matplotlib (v. 1.0.1), an open-source plotting library in python (`http://matplotlib.sourceforge.net/`) and the Basemap toolkit
- GaLab (v. 1.0.1), an open-source python interface to GraDs
- Grads (v. 2.0.a9), an interactive desktop tool that provides access, manipulation, and visualization capabilities of earth science data (`http://www.iges.org/grads/`)
- the ImageMagick software suite (v.6.3.5 or higher, `http://www.imagemagick.org/`)

In particular, the current `PATH` environment variable should be up to date to include the Grads executable. On MareNostrum, the `run_mn.sh` script takes care of this issue.

2.2 Architecture

Figure 1 shows the software stack of the application.

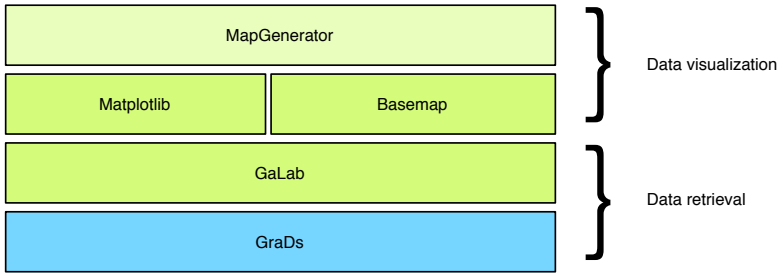

Figure 1: Architecture of MapGenerator

At the bottom level, the application uses GraDs, through the GaLab wrapper, to retrieve data from the input files. GRaDs can not only handle simple variables, but also resolve more complex mathematical expressions.

Once the data have been retrieved, Basemap is taking care of the drawing part; finally, some additional layers of information to display use Matplotlib APIs.

2.3 General workflow

Figure 2 shows the general workflow of the application.

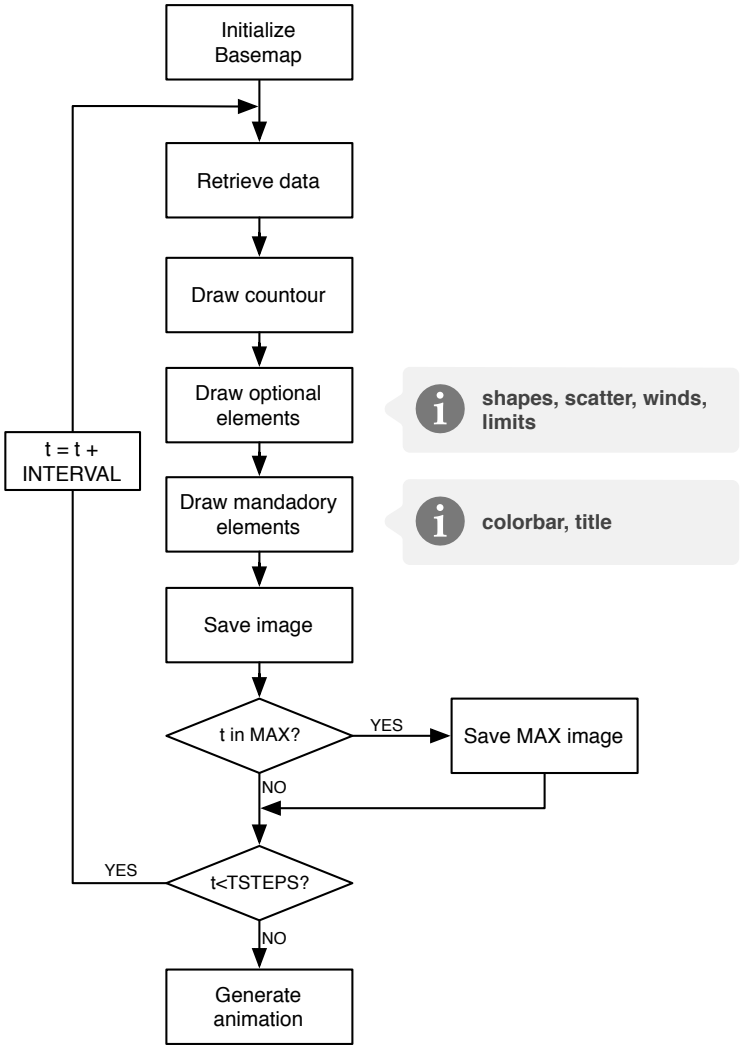

Figure 2: Architecture of MapGenerator

3 Usage

The main executable for image generation is the `map_main.py` file, located inside the `bin` directory of the main distribution; this file expects at least one parameter which identifies the section to generate inside a configuration. Additionally, a different configuration file from the default one can be specified with the `-c` option.

For instance, the command:

```
python map_main.py -c example.conf Section1
```

will look for `[Section1]` inside the file `example.conf`, read the configuration and generate the corresponding images.

3.1 Configuration files

Configuration files contain all the information needed for a run; here the user can specify information such as input directories, output directories, output map coordinates (latitude and longitude), total number of output images and their dpi, some image's attributes such as image resolution, wind's density, etc.

Appendix B shows an example of a configuration file, which is a simple text-only file. Each file is divided into sections, starting with a label specified between square brackets, plus a `General` section specifying values common to all sections. Each value is specified in the form `key = value`. A list of keys and expected values is provided in appendix A.

Each section can override values from the `General` section by simply providing a new value for the same keyword. Finally, most values inside a configuration file can be overridden directly from the command line.

3.2 Data sources and descriptors

The keyword **srcfile** specifies a list of one or more data sources; these files will be processed by the GRaDs engine by evaluating the expression of the corresponding entry of the **var** keyword.

The data source should be a self-contained source, which specifies all the data GRaDs need to draw it directly. Most of the time, this won't be true so instead of specifying a NetCDF file, the user can specify a descriptor file which contains directives that help GRaDs in the correct interpretation of the input data. Appendix C provides an example of such a descriptor file; for further details refer to `http://www.iges.org/grads/gadoc/descriptorfile.html`

4 Running on MareNostrum

4.1 Generating images

The `scripts` subdirectory contains a script called `run_mn.sh` which facilitates most of the tasks for a run on MareNostrum.

The script assumes the following directory structure:

```
\__
   \bin__
         \current [-> symlink to releases/rXXX, where XXX is the release in use]
   \releases
   \run_mn.sh      [-> symlink to bin/current/scripts/run_mn.sh]
   \check_out.sh   [-> symlink to bin/current/scripts/check_out.sh]
```

The `run_mn.sh` script requires 4 parameters:

date : date to simulate, in the YYYYMMDD format

domain : domain to simulate, see figure 3

subdomain : the subdomain to simulate, see figure 3

class : type of images, currently 'aq', 'emis' or 'meteo'; see figure 3

For each domain/subdomain/class, the script expects a file named *class*/*domain.subdomain*.conf inside the `conf` directory and tried to execute all the sections inside the given configuration file in their order of appeareance.

The script is configured to run on MareNostrum, with some predefined execution paths. In particular, the following directories are assumed to be valid:

1. `/gpfs/apps/IMAGEMAGICK/6.7.3-5/bin/` & `/gpfs/apps/GRADS/2.0.a9/bin/`, used inside the `PATH` variable
2. `/gpfs/apps/PYTHON/2.7.1/32/bin/python`, which is the release of python in use, including the necessary required libraries (see 2.1)
3. `/gpfs/projects/bsc32/bsc32359/CMAQ-mat/FORECAST/AQ/IMG`, the initial directory where to run
4. `/gpfs/projects/bsc32/bsc32359/CMAQ-mat/FORECAST/`, the directory where input data are located

Figure 3 shows the domains subdomains and classes currently supported by the application.

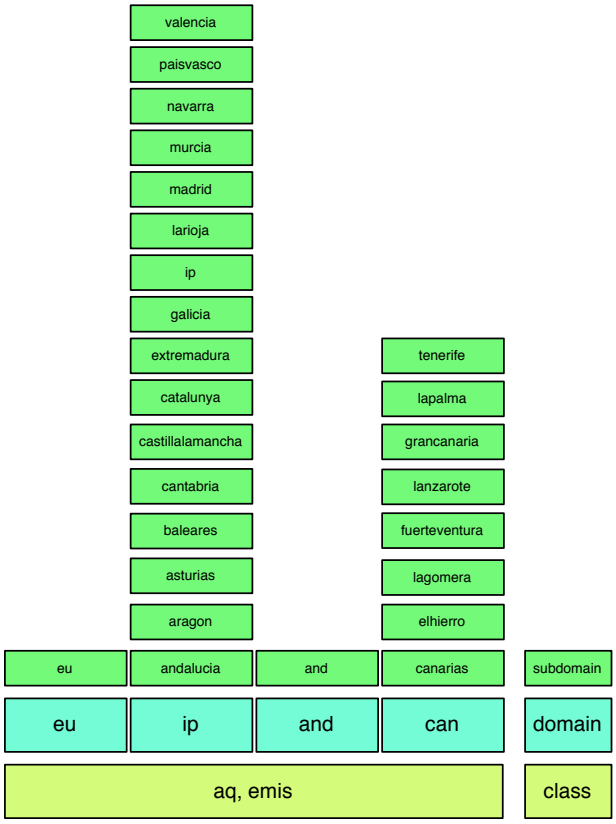

Figure 3: Domains

The currently supported options by the `run_mn.sh` script are:

```
Usage:
run_mn.sh [-abcdopr] DATE DOMAIN SUBDOMAIN CLASS
  DATE in YYYYMMDD format
  CLASS in {aq, emis, meteo}

OPTIONS:
  -a: Additional arguments to pass to the python executable
  -b: Set basedir to given value
  -c: Set configuration dir to given value
  -C: Create configuration files only (inside current directory)
  -d: Set data dir base to given value
  -o: Set out dir base to given value
  -p: Use the given python executable
  -r: Use specified release
```

```
-s: Use specified source dir
-S: Match string
```

4.1.1 Example

Invoking the following command:

```
run_mn.sh 20111210 ip ip aq
```

will run an execution for all sections inside the `aq/ip.ip.conf` file inside the `conf` directory.

4.1.2 Example

One useful option is the `-S` option which can be used to match a specific contaminant. Invoking the following command:

```
run_mn.sh -s PM10_DUST 20111210 ip ip aq
```

will only run the section inside the configuration file that matches the regular expression `*PM10_DUST`. If no section matches this value, nothing will be run.

4.1.3 Example

Invoking the following command:

```
run_mn.sh -o /tmp/test 20111210 ip ip aq
```

will run an execution for all sections inside the `aq/ip.ip.conf` file inside the `conf` directory and will place the output inside the `/tmp/test` directory. The path provided to the command are **expected to be absolute**.

4.2 Checking the generated output

The `scripts` directory contains another script named `check_out.sh` which purpose is to check if all packs of images were correctly produced, and relaunch any pack that was not created nor completed.

The currently supported options of the `check_out.sh` script are:

```
Usage: check_out.sh [-c CLASS] [-f] [-w] [DATE]
optional DATE parameter to specify a custom date, otherwise today
    -c: CLASS argument to choose in 'aq', 'emis' or 'meteo'
    -f: fix missing images (launches a job on MN)
    -w: use current directory as base dir
```

Note: if a class is not specified, the 'aq' class is assumed as the default class.

4.2.1 Example

Invoking the following command:

```
check_out.sh -f
```

will check if all air quality images have been produced, and will launch a job on MareNostrum for each pack that was not correctly generated.

4.2.2 Example

Invoking the following command:

```
check_out.sh -c emis
```

will check if all images of the 'Emissions' class have been generated. At the end it will display a report for each pack that was generated or missing. Appending the `-f` flag would also force the regeneration of any missing pack.

4.3 Control script

The generation of images for CALIOPE is started from the control script, a C-shell script located in `/gpfs/projects/bsc32/bsc32359/CMAQ-mat/FORECAST/AQ/scripts/` and named `control_global.sc`.

Currently, the generation of each pack of images is started right after the finalization of the simulation producing the input data. For instance, the images for the Iberian Peninsula are generated with the following commands:

```
foreach p ( ip andalucia aragon asturias baleares cantabria \
            castillalamancha castillaleon catalunya extremadura \
            galicia larioja madrid murcia navarra paisvasco valencia )
        $JOBIFY -n im_aq_${p} -w1h  -a es1 -c benchmark -i $IMG_NEW \
            -l logs/$YEARMONTHDAY ./run_mn.sh $YEARMONTHDAY IP ${p} aq
        $JOBIFY -n im_em_${p} -w90m -a es1 -c benchmark -i $IMG_NEW \
            -l logs/$YEARMONTHDAY ./run_mn.sh $YEARMONTHDAY IP ${p} emis
end
```

At the end of the script, a control is placed to check if all images were properly created, using the script in section 4.2, with the following commands:

```
$IMG_NEW/check_out.sh -f
$IMG_NEW/check_out.sh -f -c emis
```

The following environment variables are used:

JOBIFY : path to the jobify.sh script, a script used to submit any script as a job for MareNostrum. The current value of this variable is `/gpfs/projects/bsc32/share/bin/jobify.sh`

IMG_NEW : the base directory where the software distribution is installed. The current value of this variable is `/gpfs/projects/bsc32/bsc32359/CMAQ-mat/FORECAST/AQ/IMG`

YEARMONTHDAY : specifies the simulation date with a YYYYMMDD pattern

5 Improvements over previous version

The use of this software adds some relevant improvements to the original, GRaDs-only software, including:

higher quality images : the quality of images improved with more deepness (due to the increase in the number of colors) and anti-aliasing support (improves the appearance of polygon edges, especially in typographic characters)

substantial reduction of code size : instead of copies of similar code running for each domain/subdomain, there is now a single code and multiple, compact configuration files which express domain-specific characteristics. The GRaDs scripts comprised around 1500 lines of code for each pack, while the new code is about 1000 lines, plus about 80 to 100 lines for each pack's configuration file, implying a reduction of more than 90% in the total number of lines.

rationalization of the process :

- configuration files are organized in directories with meaningful names
- images are organized in *packs* (one pack for each domain/subdomain)
- output directory is organized in a hierarchical tree by date, class, domain and subdomain
- image elements (maps, colorbars, titles) and their positioning are now consistent thorough all packs

smaller animations : with the use of ImageMagick operators, the size of animations has been reduced from 30% up to 70%

higher granularity : the code allows for the creation of any item of a sequence of images, or a specific pack, or all images of a domain. All values are overridable from command line

other graphical improvements , which include:

- configurable support for winds and wind arrows densities
- configurable support for image proportions
- configurable support for air quality limits

5.1 Performance

Table 5.1 provides a view of the current running times for Air Quality image generation, while 5.1 provides information for Emissions image generation. Previous running times for Air Quality have been quantified as "more or less" 30 to 40 minutes for each pack, including all contaminants; no quantification was previously provided for Emissions' images. The new application allows a better granularity, providing precise times for each contaminant.

From tables 5.1, we can deduce that the general level of performance on MareNostrum (where each pack runs in parallel) has been maintained, while the single pack performance has been maintained or improved.

A List of configurable attributes

General values			
Name	Type	Description	Example
`indir`	string	input directory, where data files are located	indir = .
`outdir`	string	output directory, where images will be placed	outdir = out
lat	list of floats	lower latitude, upper latitude, distance between lines	lat = 41.6, 44.0, 0.2
lon	list of floats	lower longitude, upper longitude, distance between lines	lon = -9.7, -6.3, 0.3
max	list of ints	time steps where image of max should be printed	max = 23, 47
total	int	number of images to draw	total = 49
start	int	starting time step	start = 0
interval	int	time steps between two consecutive images	interval = 1
srcfile	list of strings	list of input data files (or data descriptor files)	srcfile = pollutants.sdf,
var	list of strings	list of variables to draw. Accepts any formula supported by GraDs	var = 'o3*1962'
gap	list of ints	list of the first time step to take from each specified source. Useful when multiple input data have different initial times. Size of this list and srcfile should be equal	gap = 1, 13
bounds	list of float	list of values for value intervals in the image drawing	bounds = 0.1, 0.15, 0.2, 0.3, 0.4, 0.5, 0.65, 0.8, 1, 1.5, 2, 3, 4, 5
Drawing values			
Name	Type	Description	Example
resolution	value in: {'l', 'i', 'h', 'f'}	map resolution. Higher resolutions require higher initialization times	resolution = 'l'
limits	list of int	drawing of black triangles at the configured levels	limits = 120, 180, 240
under	HTML color string	color for values below first in bounds	under = '#004BFF'

over	HTML color string	color for values above last in bounds	over = '#000000'
colors	list of HTML color strings	list of colors to apply over the bounds list. The two lists should have the same size	colors = '#075aff', '#1f9df0', '#2bffe5', ...
area_thresh	int	threshold for drawing surfaces. Any surface with an area smaller than the specified value won't be represented. Value is in km^2	area_thresh = 100
colorbar	boolean	boolean toggle for colorbar. Default: true	colorbar = False

Optional drawing values

Name	Type	Description	Example
shapes	list of strings	list of additional shapefiles to draw on the picture. The files should be placed into the INDIR/data directory	shapes = 'ESP_adm1', 'ESP_adm2',
drawopts	any subset of strings from {'coastlines', 'countries'}	draw additional graphical elements on the map	drawopts = 'coastlines'
wind	string	winds data file (or data descriptor file)	wind = 'winds.sdf'
windopt	3 or 4 strings	variable name for u & v component, number of values to skip between two consecutive arrows; a fourth optional value specifies the relative arrow length	windopts = uwind, vwind, 4
contours	string	variable or expression for non-filled contours	contours = 'pblh'
contours_interval	integer	interval for values in non-filled contours	contours_interval = 4
windopt	3 strings	variable name for u & v component, number of values to skip between two consecutive arrows	windopts = uwind, vwind, 4

B Example of configuration file

```
[General]
title = 'Generic title'
#dirs
indir = '.'
outdir = 'out/eu/'
#coords
lat = 16, 65, 5
lon = -40, 54, 10
#time (1=every instant, 2=each 2 instants, etc...)
wind = 'winds.eu.sdf'
windopts = uwind, vwind, 13, 60 # ucomp, vcomp, skip
max = 23, 47
resolution = 'l'
total = 49
interval = 1
freq = 1
aspect = False
srcfile = pollutants.eu.sdf,
gap = 1,
area_thresh = 500
```

DOMAIN	O3	NOx	SO2	CO	PM10	TOTAL
eu	4.8	4.7	4.3	4.2	18.0	35.9
ip	6.2	6.9	6.7	6.5	14.2	40.4
andalucia	5.6	5.9	5.9	5.7	7.6	30.7
aragon	5.6	5.7	5.8	5.9	7.6	30.5
asturias	5.5	5.6	5.8	5.7	6.9	29.4
baleares	2.5	2.5	2.4	2.3	4.5	14.1
cantabria	5.5	5.7	5.8	5.6	6.7	29.2
castillalamancha	5.6	6.2	6.0	5.9	7.6	31.2
castillaleon	5.8	6.2	6.1	6.0	8.0	32.0
catalunya	5.5	5.9	5.8	5.7	7.5	30.2
extremadura	5.5	5.8	5.7	5.5	7.3	29.8
galicia	5.5	5.9	5.7	5.5	7.2	29.8
larioja	5.6	5.6	5.5	5.6	6.8	29.1
madrid	5.4	5.6	5.5	5.6	6.8	28.9
murcia	5.4	5.8	5.6	5.5	7.3	29.6
navarra	5.1	5.4	5.3	5.2	6.6	27.5
paisvasco	5.5	6.0	5.7	5.8	7.2	30.2
valencia	5.4	5.9	5.7	5.8	7.2	30.0
canarias	3.2	3.1	3.2	3.2	9.4	22.1
tenerife	2.3	2.5	2.4	2.2	3.5	12.9
grancanaria	2.3	2.4	2.2	2.2	3.6	12.6
fuerteventura	2.3	2.4	2.2	2.2	3.6	12.6
lanzarote	3.8	3.7	3.7	3.7	4.9	19.7
lagomera	3.6	3.6	3.7	3.6	4.7	19.3
lapalma	3.6	3.6	3.5	3.5	4.5	18.6
elhierro	3.7	3.5	3.5	3.5	4.5	18.7
bcn	4.2	4.6	4.2	4.3	9.8	27.1

Table 1: Running times (in minutes) for Air Quality images (data from Jan, 8th 2012 run)

DOMAIN	NO	NO2	SO2	PM	CO	VOCs	TOTAL
eu	11.4	10.4	12.8	31.9	11.2	27.4	104.9
ip	9.9	10.0	10.3	22.3	10.8	30.9	94.1
andalucia	6.2	6.3	6.8	8.9	6.9	11.9	47.0
aragon	5.9	6.0	6.2	8.4	6.5	9.8	42.6
asturias	5.3	5.3	5.4	7.1	5.7	8.1	36.9
baleares	5.3	5.2	5.5	7.1	5.4	8.5	36.9
cantabria	5.6	5.6	5.7	7.3	5.6	8.4	38.1
castillalamancha	6.2	6.4	6.3	9.1	6.6	11.0	45.6
castillaleon	6.5	6.8	6.7	9.6	7.0	11.2	47.7
catalunya	5.8	6.1	6.0	7.5	6.1	9.7	41.1
extremadura	5.3	5.5	5.6	7.9	5.9	10.1	40.1
galicia	5.9	5.9	6.0	8.1	6.3	8.9	40.9
larioja	5.3	5.4	5.4	7.2	5.7	8.3	37.2
madrid	5.4	5.7	5.7	7.1	5.7	9.1	38.8
murcia	5.2	5.2	5.2	7.2	5.4	8.9	37.1
navarra	5.6	5.7	5.7	7.7	5.9	8.9	39.5
paisvasco	5.8	5.9	5.9	7.5	6.1	8.1	39.3
valencia	5.6	5.8	5.7	7.9	5.8	9.3	40.1
canarias	4.6	4.5	4.4	13.6	5.0	20.4	52.5
tenerife	3.2	3.1	2.5	3.8	3.3	4.7	20.5
grancanaria	3.0	2.8	2.4	3.6	3.1	4.3	19.0
fuerteventura	3.9	3.8	3.6	4.7	4.2	5.2	25.4
lanzarote	3.8	3.8	3.6	4.2	4.1	4.8	24.3
lagomera	3.7	3.6	3.5	4.2	4.0	5.0	23.9
lapalma	3.6	3.5	3.4	4.1	3.8	4.9	23.2
elhierro	3.5	3.2	3.2	4.0	3.5	4.6	21.9
bcn	5.6	5.5	5.5	14.6	5.9	21.6	58.6

Table 2: Running times (in minutes) for Emissions images (data from Jan, 8th 2012 run)

```
drawopts = "countries", "coastlines"
anim = True
under = '#0022FF'
over = '#6e0000'

[BSC-ES_FORECAST_O3]
title = """BSC-ES/AQF ARWv3+CMAQv4.5+HERMESv2 Ozone (g/m)
%(step)sh forecast for %(simhh)sUTC %(day)s %(MONTH)s %(year)s - Europe Res: 12x12km"""
maxtitle = """BSC-ES/AQF ARWv3+CMAQv4.5+HERMESv2 Max 1-hr Ozone (g/m)
d+%(simday)s forecast for %(day)s %(MONTH)s %(year)s - Europe Res: 12x12km"""
bounds = 80, 100, 120, 140, 160, 180, 240
boundaries = 0, 500
colors = '#075aff', '#1f9df0', '#2bffe5', '#07ff66', '#91ff20', '#fff000'
var = 'o3*1962',
limits = 120, 180, 240
```

C Example of GRaDs descriptor files

```
dset /gpfs/projects/bsc32/bsc32359/CMAQ-mat/FORECAST/AQ/F-AND2/OUT/CCTM/2011350/CCTM_FORECAST_ANDCONC.FORECAST_AND
title Models-3 DATA
dtype netcdf
undef -9999.
pdef  332 178 lcc   35.72874  -7.968964   1   1   37.0   43.0    -3.0  2000.  2000.
xdef  340 linear  -7.9  0.0215
ydef  180 linear   35.9  0.017
zdef 15 LINEAR 1 1
tdef 49 LINEAR 00:00Z16dec2011 01hr
vars 26
O3=>o3 15 t,z,y,x Ozone
NO=>no 15 t,z,y,x Nitrogen_Monoxide
NO2=>no2 15 t,z,y,x Nitrogen_Dioxide
CO=>co 15 t,z,y,x Carbon_Monoxide
SO2=>so2 15 t,z,y,x Sulphur Dioxide
ASO4I=>aso4i 15 t,z,y,x ASO4I
ASO4J=>aso4j 15 t,z,y,x ASO4J
ANO3I=>ano3i 15 t,z,y,x ANO3I
ANO3J=>ano3j 15 t,z,y,x ANO3J
ANH4I=>anh4i 15 t,z,y,x ANH4I
ANH4J=>anh4j 15 t,z,y,x ANH4J
AORGAI=>aorgai 15 t,z,y,x AORGAI
AORGAJ=>aorgaj 15 t,z,y,x AORGAJ
AORGPAI=>aorgpai 15 t,z,y,x AORGPAI
AORGPAJ=>aorgpaj 15 t,z,y,x AORGPAJ
AORGBI=>aorgbi 15 t,z,y,x AORGBI
AORGBJ=>aorgbj 15 t,z,y,x AORGBJ
AECI=>aeci 15 t,z,y,x AECI
AECJ=>aecj 15 t,z,y,x AECJ
ANAJ=>anaj 15 t,z,y,x ANAJ
ACLJ=>aclj 15 t,z,y,x ACLJ
ANAK=>anak 15 t,z,y,x ANAK
ACLK=>aclk 15 t,z,y,x ACLK
ASO4K=>aso4k 15 t,z,y,x ASO4K
```

```
A25J=>a25j 15 t,z,y,x A25J
ACORS=>acors 15 t,z,y,x ACORS
endvars
```