Introduction to the ClimValDiagTool

K. Gottschaldt & V. Eyring, 13. 2. 2013

1. General Info
2. Access miklip.dkrz.de
3. Get the code
4. Prepare data

5. Walk through an example
6. Modify the example
7. Try other diagnostics

8. Create a new variable and a new diagnostic
1. General Info: ClimValDiagTool

- Designed for **comparing and plotting** climate parameters from model, reanalysis and observational data, given in NetCDF format.

- A **wealth of analysis routines** is inherited from previous and current **multi-model intercomparison/verification projects**. The ClimValDiagTool is an extension of the CCMVal Diagnostic Tool (http://www.pa.op.dlr.de/CCMVal/CCMVal_DiagnosticTool.html).

- Please **cite** and refer to:

- Current **license** for use: CCMValDiagTool_license.txt (in the root folder of the tool). The option to output references and acknowledgements for the parts used is in preparation.

- **Goal for MiKlip**: compile namelists with **standard diagnostics** for the MPI-ESM decadal simulations → model skill assessment at the push of a button
1. General Info: This Tutorial

Goals:

- Enable you to use the ClimValDiagTool on the MiKlip server
- Encourage you to contribute your own diagnostics to the ClimValDiagTool

Practical matters:

- The handout is supposed to be self-explanatory. We might go through 1 – 5 together, then you are encouraged to work through the tutorial individually.
- Tasks in sections 6, 7, 8 are almost independent of each other: Choose what you would like to do (but note that your plots might look different if not sticking to the proposed order)
- Selected lines in the code have been disabled for tasks 5, 6, 7. You need to enable them, which forces you to touch some control points without much editing. The yellow boxes show how it should look like in order to work.
- Task 8 is more advanced, with plenty of freedom on how to do it. One solution will be provided, and might serve as a template for your own diagnostics.
2. Access miklip.dkrz.de

• connect to WLAN

• login at miklip.dkrz.de

```bash
$ ssh -X miklip.dkrz.de -l <your_account>
```

• make sure you can get graphics windows from the remote machine (e.g. from MS Windows: enable X11 forwarding in putty, use Exceed, MobaXterm …)

• set up environment (e.g. in .bashrc)

```bash
~]$ module load ncl/6.1.0-gccsys
alias gv='/usr/bin/ghostscript'
```

… we do need NCL

… this is just for your convenience
3. Get the code

• Create a directory (e.g. “TOOL”), where you have 300 MB of free disk space (e.g. in your $HOME). You may choose any other name & location.

```bash
~]$ mkdir TOOL
~]$ export TOOL=~/TOOL
```

• $TOOL will refer to this directory from now on

• Copy the code into $TOOL

```bash
~]$ cd $TOOL
```

• Unpack

```
TOOL]$ tar xvf ClimValDiagTool_20130211.tar
```

• Check

```
[b309056@miklip01 TOOL]$ ls
ClimValDiagTool_20130211.tar  source
```

```csh
~]$ setenv TOOL ~/TOOL
```

```bash
~]$ setenv TOOL ~/TOOL
```

(csh)
4. Prepare data

We need model data and observations/reanalysis to compare them to.

This version of the ClimValDiagTool (still) expects a certain input file name structure, which may be realized by soft links to the original data. A shell script does this for you.

• Check out the script, then execute it

```
TOOL]$ cd source
source]$ vi prepare_tutorial.sh
```

You may use any other editor.

```
# link to observational data (maintained centrally for the tutorial)
# Note: At some stage these data may be moved to /miklip/integration/data4miklip/
OBSPATH="../obs"

# path for processed model data
OUTPATH="../models"

# process example model output (renaming via soft link)
INPATH="/miklip/global/prod/archive/baseline1/output/MPI-M"
ESMin=( 'MPI-ESM-LR' )
ESMout=( 'LR' )
ENSin=( 'r1i1p1' 'r2i1p1' 'r3i1p1' )
ENSout=( '1' '2' '3' )
DECin=( 'decs4e2000' 'decs4e2001' )
DECout=( 'd2000' 'd2001' )

source]$ prepare_tutorial.sh
```
4. Prepare data

• Check the results

[b309056@miklip01 source]$ ls ..../obs/ClimVal*
..../obs/ClimVal_obs_ERA40_1_T2Mz_ta_197901_200112.nc
..../obs/ClimVal_obs_ERA40_1_T3M_ta_197901_200112.nc
..../obs/ClimVal_obs_ERAI_1_T2Mz_ta_199501_200512.nc
..../obs/ClimVal_obs_ERAI_1_T3M_ta_199501_200512.nc
..../obs/ClimVal_obs_ERAI_T255_T2Ms_tas_200501_200512.nc
..../obs/ClimVal_obs_NCEP_1_T2Mz_ta_197901-200812.nc
..../obs/ClimVal_obs_NCEP_1_T2Mz_ta_197901-200812.nc
..../obs/ClimVal_obs_NCEP_1_T3M_ta_197901-200812.nc
..../obs/ClimVal_obs_NCEP2_1_T2Mz_ta_197901-200812.nc
..../obs/ClimVal_obs_NCEP2_1_T2Mz_ta_197901-200812.nc
..../obs/ClimVal_obs_NCEP2_1_T3M_ta_197901-200812.nc
..../obs/ClimVal_obs_NCEP2_1_T3M_ta_197901-200812.nc

[b309056@miklip01 source]$ ls ..../models
ClimVal_d2000_LR_1_T2Ms_tas_200101-201012.nc  ClimVal_d2001_LR_1_T2Ms_tas_200201-201112.nc
ClimVal_d2000_LR_1_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_1_T3M_ta_200201-200912.nc
ClimVal_d2000_LR_1_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_1_T3M_ta_200201-200912.nc
ClimVal_d2000_LR_2_T2Ms_tas_200101-201012.nc  ClimVal_d2001_LR_2_T2Ms_tas_200201-201112.nc
ClimVal_d2000_LR_2_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_2_T3M_ta_200201-200912.nc
ClimVal_d2000_LR_2_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_2_T3M_ta_200201-200912.nc
ClimVal_d2000_LR_3_T2Ms_tas_200101-201012.nc  ClimVal_d2001_LR_3_T2Ms_tas_200201-201112.nc
ClimVal_d2000_LR_3_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_3_T3M_ta_200201-200912.nc
ClimVal_d2000_LR_3_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_3_T3M_ta_200201-200912.nc
ClimVal_d2000_LR_3_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_3_T3M_ta_200201-200912.nc
ClimVal_d2000_LR_3_T3M_ta_200101-200912.nc  ClimVal_d2001_LR_3_T3M_ta_200201-200912.nc
5. Walk through an example: Primer

Model Output
- specific processing
  - internal
    - reformat
  - external
    - shell scripts, cdo ...

Observations
- plot_type/input_data
  - internal
  - external
    - like another model

Basic control
- namelist_*
  - Set global flags
  - Specify model / obs names, years and paths
  - Specify diagnostic set

Diagnostics
- diag_att/*.att
  - Plot type
  - List of variables
  - Field type

Variable attributes
- var_att/<for each var>_att.ncl
  - set info@ ... parameters for each plot type
  - if derived variable: need a calculate function

Output
- work
  - Plots, NetCDF files

Paths on this slide are relative to $TOOL/source/
5. Walk through an example: Primer

- Check the reformat branch:

```bash
source]$ cd reformat
reformat]$ cd fix_ClimVal/
fix_ClimVal]$ vi ClimVal_LR.ncl
```

```ncl
;op_kg_20130211: set calendar to "standard" and Pa -> hPa

undef("fix_data")
function fix_data(dataX)
begin
  if (dataX&time@calendar.eq."proleptic_gregorian") then
    dataX&time@calendar = "standard"
  end if
  if isdim(dataX,"plev") then
    dataX&plev = dataX&plev /100.
    dataX&plev@units = "hPa"
  end if
  return(dataX)
end
```

This slide shall just make you aware of this intermediate step.
No action needed.

Here you have the chance to make model specific adjustments to the input data …
or just apply some q&d fixes that should go into the code later.
5. Walk through an example

Adjust the main namelist:

```
$ cd $TOOL/source
$ vi namelist_ClimVal
```

```
# specify the project (CCMVal2, CCMVal1)
project ClimVal

# specify the work directory
wrk_dir ./work

# specify the plot directory
plot_dir $wrk_dir/plots_ClimVal/

# specify the directory where the climo files are stored
climo_dir $wrk_dir/climo_ClimVal/
```

<table>
<thead>
<tr>
<th>key</th>
<th>value</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_plots</td>
<td>[true,false]</td>
<td>Currently not used</td>
</tr>
<tr>
<td>write_netcdf</td>
<td>[true,false]</td>
<td>Currently not used</td>
</tr>
<tr>
<td>force_processing</td>
<td>[true,false]</td>
<td>Force rewriting of certain intermediate netCDF files</td>
</tr>
<tr>
<td>project</td>
<td>e.g., EMBRACE</td>
<td>Specify project</td>
</tr>
<tr>
<td>wrk_dir</td>
<td>[path]</td>
<td>Specify output path</td>
</tr>
<tr>
<td>plot_dir</td>
<td>$wrk_dir/plots</td>
<td>Specify the output plot directory</td>
</tr>
<tr>
<td>climo_dir</td>
<td>$wrk_dir/climo</td>
<td>Specify the output directory for intermediate and climatology netCDF files</td>
</tr>
<tr>
<td>write_plot_vars</td>
<td>[true,false]</td>
<td>Currently not used</td>
</tr>
</tbody>
</table>

### MODELS

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MODEL</td>
<td>year</td>
<td>year</td>
<td>directory</td>
</tr>
<tr>
<td></td>
<td>LR</td>
<td>2000</td>
<td>1</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td></td>
<td>LR</td>
<td>2000</td>
<td>2</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td>#</td>
<td>LR</td>
<td>2001</td>
<td>1</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td>#</td>
<td>LR</td>
<td>2001</td>
<td>2</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td></td>
<td>ERAI</td>
<td>obs</td>
<td>1</td>
<td>$TOOL/obs</td>
</tr>
<tr>
<td></td>
<td>NCEP</td>
<td>obs</td>
<td>1</td>
<td>$TOOL/obs</td>
</tr>
<tr>
<td>#</td>
<td>LR</td>
<td>2000</td>
<td>1</td>
<td>$TOOL/models</td>
</tr>
</tbody>
</table>

### DIAGNOSTICS

```
# specify the program
# e.g., './diagnosis' diag_ClimVal
```

... just remove the appropriate `#`
to make your namelist_ClimVal look like here
5. Walk through an example

Adjust the diagnostic namelist:

```
#ta  T3M  E06FIG01
ta   T3M  E06FIG07
#ta  T3M  vertconplot
#ta  T3M  vertconplot_pair
#tas T2Ms surfconplot
```

```
variable  type  diagnostic
```

• Execute the main python script

```
diag_att]$ cd ..
source]$ main.py namelist_ClimVal
```

• Check the plot

```
source]$ gv work/plots_ClimVal/E06FIG07_ta.ps
```

![100 hPa Temperature at Equator](image)
6. Modify the example
6.1. Specify a reference model

• Check which parameters are evaluated by the diagnostic routine
  ```bash
  source]$ cd plot_type/
  plot_type]$ vi E06FIG07.ncl
  ```

  ; info attributes required:
  ; fig07_lat_min   arr
  ; fig07_lat_max   arr
  ; fig07_lev_sel   arr
  ;                       mus
  ;                       fig
  ; fig07_refModel   nam

  • Those parameters are expected from the variable namelist, which is specified by "ta"
  in $TOOL/source/diag_att/diag_ClimVal.att.

  • Find corresponding section in $TOOL/source/var_att/ta_att.ncl & enable info@fig07_refModel

  ```bash
  plot_type]$ cd ..
  source]$ cd var_att
  var_att]$ vi ta_att.ncl
  ```

  info@fig07_lat_min = (/0./) ;
  info@fig07_lat_max = (/0./) ;
  info@fig07_lev_sel = (/100./) ;
  info@fig07_refModel = ("ERAI")

  • Redo the plot and see what has changed

  ```bash
  source]$ main.py namelist_ClimVal
  source]$ gv work/plots_ClimVal/E06FIG07_ta.ps
  ```
6. Modify the example
6.2. Add another simulation

- Modify the main namelist
  ```
  source]$ vi namelist_ClimVal
  ```

<table>
<thead>
<tr>
<th>MODELS</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>d2000</td>
<td>1</td>
<td>2002</td>
<td>2005</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td>LR</td>
<td>d2000</td>
<td>2</td>
<td>2002</td>
<td>2005</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td>LR</td>
<td>d2001</td>
<td>1</td>
<td>2002</td>
<td>2005</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td>LR</td>
<td>d2001</td>
<td>2</td>
<td>2002</td>
<td>2005</td>
<td>$TOOL/models</td>
</tr>
<tr>
<td>ERAI</td>
<td>obs</td>
<td>1</td>
<td>2002</td>
<td>2005</td>
<td>$TOOL/obs</td>
</tr>
<tr>
<td>NCEP</td>
<td>obs</td>
<td>1</td>
<td>2002</td>
<td>2005</td>
<td>$TOOL/obs</td>
</tr>
</tbody>
</table>

- Redo the plot and see what has changed
  ```
  source]$ main.py namelist_ClimVal
  source]$ gv work/plots_ClimVal/E06FIG07_ta.ps
  ```

100 hPa Temperature at Equator
6. Modify the example
6.3. Explicitely specify colors

- CCMVal assigned a specific color to each model. ClimVal default: Unspecifically assigns a different color to each Model-Case-Ensemble combination. Suppose you want identical colors for each group of decadals …

- Check which color table is called in E06FIG07.ncl

```
source]$ vi plot_type/E06FIG07.ncl
```

- Colors are defined in ~/TOOL/source/plot_type/CCMVal_FUNCTION/misc_function.ncl

```
function ESMVal_ColorTable(model:string,case:string,imod)
begin
    return(ESMVal_Colors1(model,case,imod))
end
```

- Enable lines with “LR“ (& “end if”) in ESMVal_Colors1

```
else if (model.eq."LR" .and. case.eq."d2000") then
    colori(i) = "darkorange"
else if (model.eq."LR" .and. case.eq."d2001") then
    colori(i) = "cyan"
```

- Redo the plot and see what has changed
6. Modify the example
6.4. Panelling

- E06FIG07.ncl allows multiple plots on a page.

- The number of plots is determined by the number of latitude-level combinations given in the variable attributes namelist.

```
source]$ vi var_att/ta_att.ncl
```

- Extend the appropriate vectors:

```ncl
info@fig07_lat_min = (/0., -30., 0./)
info@fig07_lat_max = (/0., 30., 0./)
info@fig07_lev_sel = (/100., 100., 850./)
info@fig07_refModel = ("ERAI")
```

- Redo the plot and see what has changed
7. Other diagnostics
7.1. Add a plot type with internal observations

• Modify the diagnostic namelist

```bash
source]$ vi diag_att/diag_ClimVal.att
```

```
ta T3M E06FIG01
#ta T3M E06FIG07
#ta T3M vertcomplot
```

• Check required info attributes in ~/TOOL/source/plot_type/E06FIG01.ncl and adjust ~/TOOL/source/var_att/ta_att.ncl

```python
info@fig01_refModel = ("ERAI")
info@fig01_climObs = ("UKMO") ;Climatological Observation file
info@fig01_climObs_file = ("./plot_type/input_data/OBS/CCMVal2_1992-2001_UKMO_Obs_C2Mz_ta.nc")
```

Note “info@fig01_climObs_file”
- UKMO from internal observations
- ERAI treated like a model

• Create and check the additional plot

```bash
source]$ main.py namelist_ClimVal
source]$ gv work/plots_ClimVal/E06FIG01.ps
```
7. Other diagnostics
7.2. Just try more …

• Adjust the following lines in $TOOL/source/diag_att/diag_ClimVal.att

```
#ta    T3M  E06FIG01
#ta    T3M  E06FIG07
ta     T3M  vertconplot
#tas   T2Ms  surfconplot
```

$TOOL/source/var_att/ta_att.ncl contains
```
info@refModel = "ERAI"
```

• Difference to that reference is plotted by `vertconplot`

```
source]$  gv  work/plots_ClimVal/vertconplot/vertconplot_ref_ANN_ta_c.ps
source]$  gv  work/plots_ClimVal/vertconplot_pair/vertconplot_pair_ANN_ta_c.ps
```

• `vertconplot_pair` compares first two "models" of
$TOOL/source/namelist_ClimVal
7. Other diagnostics
7.3. Try a different data type (the new ERAI data on the MiKlip server)

Note: same year

```
source]$ vi namelist_ClimVal
```

```
<table>
<thead>
<tr>
<th>MODELS</th>
</tr>
</thead>
<tbody>
<tr>
<td># LR</td>
</tr>
<tr>
<td># LR</td>
</tr>
<tr>
<td># LR</td>
</tr>
<tr>
<td># LR</td>
</tr>
<tr>
<td># ERAI</td>
</tr>
<tr>
<td># NCEP</td>
</tr>
<tr>
<td>LR</td>
</tr>
<tr>
<td>ERAI</td>
</tr>
</tbody>
</table>
```

```
source]$ vi diag_att.diag_ClimVal.att
```

```
#ta T3M E06FIG01
#ta T3M E06FIG07
#ta T3M vertconplot
#ta T3M vertconplot_pair
tas T2Ms surfconplot
```

```
source]$ main.py namelist_ClimVal
source]$ gv work/plots_ClimVal/surfconplot/surfconplot_ann_tas_c.ps
source]$ vi var_att/tas_att.ncl
```

```
;info@rgb_file = "amwg.rgb"
info@rgb_file = "red-blue.rgb"
info@refModel = "ERAI"
```

```
source]$ main.py namelist_ClimVal
source]$ gv work/plots_ClimVal/surfconplot/surfconplot_ref_ann_tas_c.ps
```
8. Create a new variable and a new diagnostic

8.1. Create a derived variable “MyVar” that contains ta @ 200 hPa

```bash
source]$ cd var_att
var_att]$ vi MyVar_att.ncl
```

The string “200 hPa” shall be passed on via an info attribute.

• Take another variable as template and consider $TOOL/source/ncl_code/extract_data.ncl
• Note: The ‘Comment’ header of MyVar_att.ncl is actually evaluated!

8.2. Adjust main and diagnostic namelists to contain only these entries:

```
MODELS
  LR  d2000  1  2002 2005 $TOOL/models
MyVar  T3M  MyDiag
```

8.3. Create a diagnostic that calculates the time mean & plots a Mollweide projection of MyVar

```bash
var_att]$ cd ../plot_type/plot_type]$ vi MyDiag.ncl
```

Hints (consult www.ncl.ucar.edu):

• `list_vars()` lists all currently defined variables
• Average time with `dim_avg_n_Wrap`
• Open ps file: `gsn_open_wks`
• Colors: `gsn_define_colormap`
• Plot resources: `mpProjection`, `cnFillOn`, `tiMainString`, `gsnSpreadColors`, `gsnLeftString`, `gsnRightString`
• Plot with `gsn_csm_contour_map`
8. Create a new variable and a new diagnostic, Further information: NCL

- Please take existing code from the ESMValTool as template and consult the NCL website (http://www.ncl.ucar.edu/)

Good to know about NCL …

- Parameters are global by default and available in all routines, even if not explicitly passed
- Parameters need to be deleted explicitly before changing dimensions or type
- Parameter exchange with Python is via environment variables
- Index count starts from 0
8. Create a new variable and a new diagnostic, Further information: Code components

- See $TOOL/source/doc/tutorial.pdf
  $TOOL/source/doc/README_20120719.txt

Paths on this slide are relative to ~/TOOL/source/
8. Create a new variable and a new diagnostic, Further information: Control flow

- See $TOOL/source/doc/control_flow.txt

\[
\text{main.py} \ \text{nml}
\]

---read_namelist.py reads 'nml' with GENERAL settings: paths, what to do, etc
  MODEL settings: five variables per line: model name, case name, ensemble no, years, path
  DIAGNOSTICS settings: list of diagnostics to perform (what to plot). These are defined in the diag_att/ folder.

--loop over all diagnostics defined above

---read_diag_att(type) from diag_att/
  Read the diagnostics, each defined as a combination of a variable a field, & a plot.

--loop over variables

  --loop over models

    --create case folder if necessary
    ---ccsm.py (e.g.): if necessary, rewrite data to time series
    ---cf_convert.py
      write time series into specified time chunks, standardize variable names (lat/lon/time/plev)
    ---climat.py: compute climatology, annual, seasonal, monthly, from the chosen field
      (see reformat/attribute.ncl for a full list of supported fields)

  --main.ncl
    --load variable attributes (var_att/variable_att.ncl)

    --loop over models

      --if it is a derived variable, calculate dependent variables (unless it is precomputed)
      --loop over plot_types and plot