Documentation Kernel Calculations – From Model Output to Kernel-Derived Radiative Fluxes

Kai-Uwe Eiselt, University of Tromsø

19.05.2023

Nr.	script name
1	Concat_CMIP5_Files.py
2	Concat_CMIP6_Files.py
3	Calc_and_Store_TOA_Imbalance.py
4	Extend_piControl_2d.py
5	Extend_piControl_3d.py
6	Split_3d_Files.py
7	Calc_ta_hus_RunningMean.py
8	Calc_and_Store_dQdT.py
9	Calc_and_Store_CRE.py
10	Calc_SurfaceAlbedo_Resp_KernelMethod.py
11	Calc_SurfaceAlbedo_Resp_KernelMethod_CSDiff.py
12	Calc_Temp_Resp_KernelMethod.py
13	Calc_Temp_Resp_KernelMethod_CSDiff.py
14	Calc_WaterVapour_Resp_KernelMethod.py
15	Calc_WaterVapour_Resp_KernelMethod_CSDiff.py
16	Calc_CloudResp_KernelMethod.py
17	Calc_and_Store_siarea.py
18	Calc_and_Store_LTS.py
19	Calc_and_Store_dTOA_Imbalance.py
20	Calc_and_Store_tas_Change_Loc_and_Glob_Run_piC.py
21	Gregory_LinReg_from_MonMean_and_piCRun.py
22	ClearSky_LinearityTest_GlobMean_CMIP5_6.py

These scripts have been tested on an Ubuntu system with 16 GB memory and are written to work in the Conda environment *kern_calc* which is attached (as the text file "kern_calc.txt"). Note that, apparently, due to the use of functions from the *geocat* module some of these scripts do not work with Python version 3.8.0 or later.

If Anaconda is installed on your machine, the attached environment can be installed with the following commands:

```
conda create --name kern_calc --file kern_calc.txt
```

```
conda install --name kern_calc --file kern_calc.txt
```

Note that the kern_calc.txt file was produced via the following command:

```
conda activate kern_calc  # activates the environment
```

```
conda list --explicit > kern_calc.txt  # stores the spec list
```

You can check if the above-mentioned version of Python is running by entering (after you have activated the environment):

```
python -V
```

Kernel Preparations		
Nr.	data/input dependencies	short description
1	CMIP5 model output 2d variables [time, lat, lon]; e.g., *ps, rlut, rsut, rsdt, rsus, rsds, tas, ts* etc.	concatenates model output 2d variable files into one file *along the time coordinate* and stores output files
2	same as for [1] but for CMIP6	same as for [1]
3	model output all-sky and clear-sky top-of-atmosphere (TOA) radiative flux variables: *rlut, rsut, rsdt plus* for clear-sky *rlutcs, rsutcs*; best use scripts [1] and [2] before this	calculates the all-sky and clear-sky TOA *net* fluxes and stores output files
4	piControl model output 2d variables [time, lat, lon]; or these data as pre-processed by [1] and [2]	extends the piControl data by ten years (as required either at the start or at the end or both) to make possible the calculation of a 21-year running mean in case the piControl run does not start at least 10 years before and/or does not end at least 10 years after the abrupt4xCO2 run (see branch year) and stores the output files
5	same as for [4] but for 3d variables [time, plev, lat, lon]	same as for [4]
6	temperature (*ta*) and specific humidity (*hus*) model output data	splits up the input file into smaller files of given time period length
7	piControl model output of the variables *ta* and *hus* as well as (if required) the output of [5]	calculates the 21-year running mean of piControl *ta* and *hus* and stores the output files (this is a workaround if files are too large for the memory)
8	model output of abrupt4xCO2 *ta* and *hus* as well as the 21-year running mean piControl *ta* and *hus* as calculated via [6]	calculates the $\frac{\partial Q}{\partial T}$ needed for the specific humidity kernel under the assumption of constant relative humidity and stores the output files
9	model output all-sky and clear-sky TOA radiative fluxes (*rsut[cs], rlut[cs]*); for the models for which the 2d and 3d variables are on different lat-lon grids, e.g., ACCESS1.0 etc., also some 3d variable is needed to get the 3 lat-lon grid; currently this is done by loading a file produced by [8], = *statement 1*	calculates the cloud radiative effect (CRE) and stores the output files; note that since the variables used for the calculation have units *positive-up*, CRE is calculated as clear-sky minus all-sky; note further, that for models for which 2d and 3d variables are on different lat-lon grids, the CRE is remapped to the 3d lat-lon grid (e.g., ACCESS1.0), = *statement 2*

Kernel Calculations (based on the Pendergrass et al., 2018 scripts)		
Nr.	data/input dependencies	short description
10	model output surface radiative fluxes (*rsds, rsus*) for both abrupt4xCO2 and piControl as well as the (if required) the corresponding files produced by [4]; radiative kernels for surface albedo as downloaded from their providers; also: see *statement 1*	calculates the (either all-sky or clear-sky) radiative kernel derived radiative flux change due to surface albedo change in the forced experiment (here abrupt4xCO2) compared to the base experiment (here piControl) and stores the output files; NB: this script also calculates the 21-year running mean over the piControl data; also: see *statement 2*
11	same as for [10]	same as for [10] but for the difference of clear-sky minus all-sky kernel, i.e. this script calculate the surface albedo cloud adjustment (see below) and stores the output files
12	output of [8]; surface (skin) temperature (currently *ts*) and surface pressure (*ps*) for both abrupt4xCO2 and piControl as well as the corresponding files produced by [4];	same as for [10] but for temperature
13	same as for [12]	same as for [11] but for temperature
14	same as for [12] but with water vapour kernels and without surface temperature (*ts*)	same as for [11] but for water vapour mixing ratio
15	same as for [14]	same as for [11] but for water vapour mixing ratio
16	outputs of [8] (i.e., CRE), [11], [13], [15] (i.e., the adjustments to CRE due to the individual feedbacks)	calculates the adjusted CRE as described in Soden et al. (2008) and stores the output files; note that the CO_2 forcing adjustment is **not included** in the output because it is (mostly) unavailable (see the comment at the beginning of the script)

Other Scripts		
Nr.	data/input dependencies	short description
17	model output sea ice concentration, i.e., *sic* (CMIP5) and *siconc* (CMIP6) as well as the grid cell area, i.e., *areacello* (or *areacelli* for NESM3)	calculates the NH and SH sea ice area and stores the output files; for the sea ice area, the *sic* or *siconc* is simply multiplied on *areacello* and then integrated over the NH and SH, separately
18	model output air temperature (*ta*), surface (air) temperature (*tas* or *ts*), surface pressure (*ps*); all variables are needed for abrupt4xCO2 and piControl; note that for *ta* the 21-year running mean piControl is needed, i.e., the output of [6]; some file that contains the CanESM2 lat-lon grid	calculates LTS for abrupt4xCO2 and the 21-year piControl running mean as well as their difference (i.e., the LTS anomaly); furthermore, calculates the linear regression of the LTS anomaly on the global mean surface (air) temperature; also, re-grids the latter on a specified model grid (the CanESM2 grid); stores all these products in an output file
19	output of [3] for abrupt4xCO2 and piControl	calculates the change of all-sky or clear-sky TOA imbalance from piControl to abrupt4xCO2 and stores the output file; the 21-year running mean over the piControl TOA imbalance is also stored in this file
20	model output surface (air) temperature (i.e., *ts* or *tas*) for abrupt4xCO2 and piControl; some file that contains the CanESM2 lat-lon grid	calculates the change of *ts* or *tas* from piControl to abrupt4xCO2 and also re-grids the product on a specified model grid (the CanESM2 grid); both products (original grid and re-gridded) are stored in an output file; note that the change is produced with respect to (1) a 21-year running mean of the piControl run and also (2) a linear trend over the piControl run
21	outputs of [3], [19], and [20]	performs the Gregory linear regression and stores the results in an output file which includes its own extensive documentation
22	clear-sky radiative fluxes for temperature, water vapour, and surface albedo; global mean tas, global mean TOA clear-sky imbalance	“performs” the clear-sky linearity test (CSLT; see our paper for details); generates two lists in the form of two .csv files, the one file contains the models that pass the CSLT and the other contains the models that fail the CSLT; the script also generates the clear-sky Gregory plot for each model that is considered (plots are not stored); finally produces and stores a plot that shows which models pass and which fail the CSLT