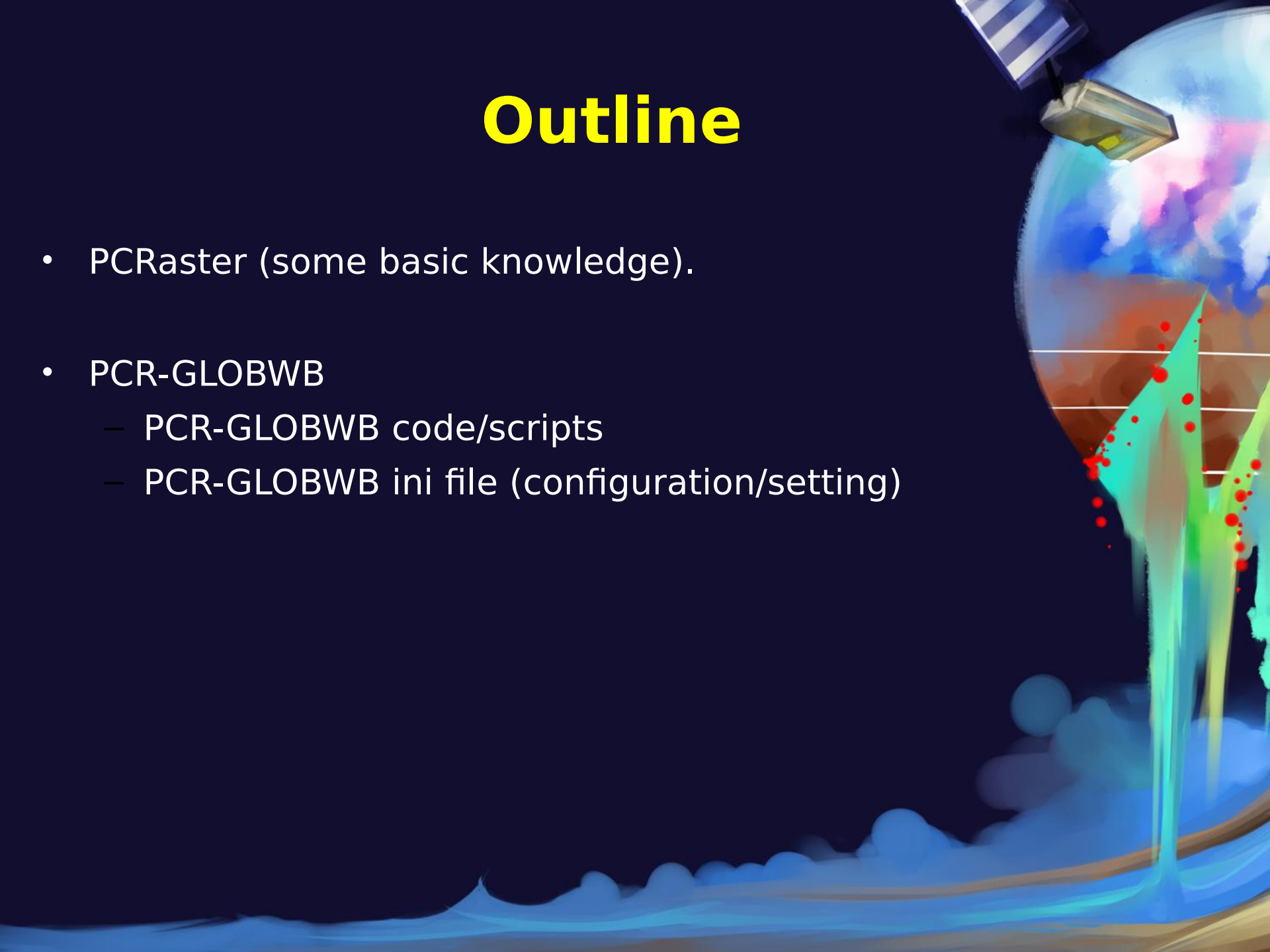
Introduction to PCR-GLOBWB code

E.H.Sutanudjaja@uu.nl ; +31 30 253 2988 ; Room: Zon104

Universiteit Utrecht

Outline

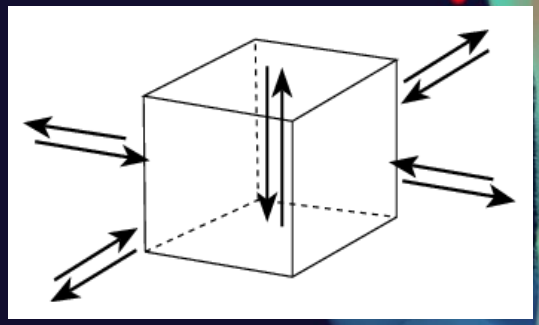
- PCRaster (some basic knowledge).
- PCR-GLOBWB
  - PCR-GLOBWB code/scripts
  - PCR-GLOBWB ini file (configuration/setting)



PCRaster

- Programming language used in PCR-GLOBWB.
  - PCR-GLOBWB also uses its Python framework.

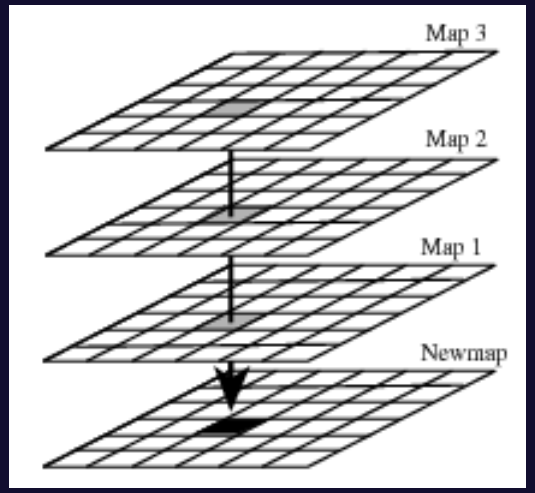
- PCRaster:
  - a set of computer tools for storing, manipulating, analyzing and retrieving geographic information
  - RASTER based
  - POINT and NEIGHBOURHOOD operators



- PCRaster documentation:
  http://pcraster.geo.uu.nl/pcraster/4.1.0/doc/manual/index.html

PCRaster: Local/point operation

- Point operation: A new map is generated on a cell-by-cell basis. No lateral/neighborhood relations between cells are included.
- Example: plus (+) operations




```
Expr1 = readmap("Expr1.map")
Expr2 = readmap("Expr2.map")
Result = Expr1 + Expr2
```

Result.map

MV	8	2
2	MV	-6
100	-7	16

Expr1.map

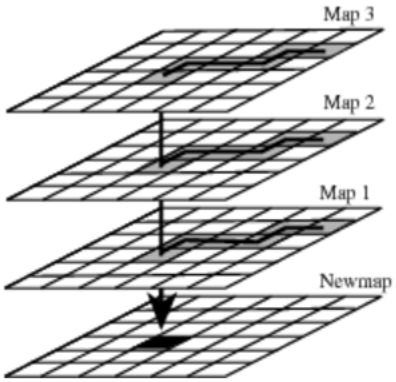
2	6.2	-3
1	MV	7
86	-1	12

Expr2.map

MV	1.8	5
1	3	-13
14	-6	4

PCRaster: Neighbourhood operations

- Neighbourhood operations relate the cell to its neighbours.
- Example:

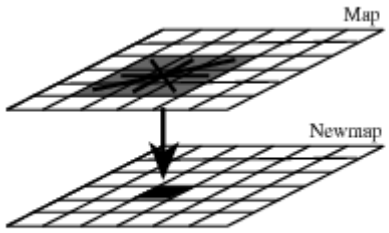


Neighbourhood operations over a path. For each cell the NewMap value is calculated on the basis of Map1, Map2, Map3,... values on a path from a source cell.

- Several categories:
  - Window operations (e.g. 3x3 averaging)
  - LDD operations (following a drainage network, upstream-downstream connection)
  - Area (zonal) operations (e.g. to calculate country statistics).
  - Map operations (e.g. statistics over an entire map).

PCRaster: Window operations

- Example: windowaverage



Neighbourhood operations within a window.

- Other functions: windowtotal, windowmaximum, windowminimum, etc.

• python

```
Expr = readmap("Expr.map")
Result1 = windowaverage( Expr, 6)
```

Result1.map

0.333	0.6	-5.4	-4.83	-7.75
1.2	1.38	-2.25	-2.22	-4.17
1.4	1.62	6.75	5.89	7.83
0.667	1.44	6.67	7.22	9.33
-0.25	0.833	8.5	9.33	12.5

Expr.map

0	-1	1	-30	0
2	MV	1	2	-3
3	2	3	4	2
0	0	2	40	2
1	-2	4	7	1

PCRaster: LDD operations

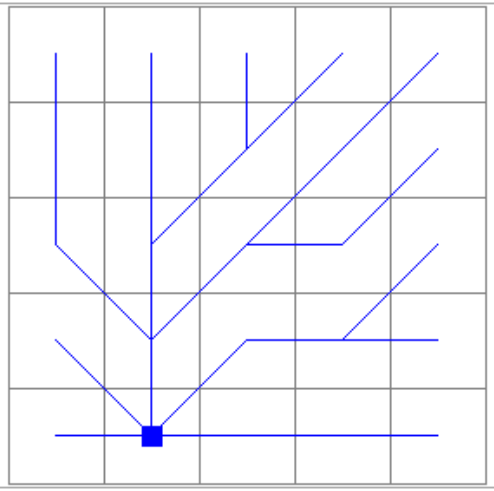
- Example: upstream: sum of the cell values of its first upstream cells(s)

```
• python
    Ldd = readmap("Ldd.map")
    Expr = readmap("Expr.map")
    Result = upstream(Ldd, Expr)
```

Result.map

0	0	0	0	0
1	1	4	4	0
1	3	MV	4	0
0	MV	4	8	0
0	14	5	5	0

Ldd.map



Expr.map

1	1	2	2	4
1	1	2	2	4
2	2	MV	4	4
2	2	2	4	4
3	7	5	5	5

- Some other functions:
  - accuflux: Accumulated material flowing into downstream cell
  - accutraveltime: Transport material downstream on a given velocity
  - kinematic, etc.

PCRaster: Area operation

- Example: areamaximum: maximum cell value within a class

```
• python
    Class = readmap("Class.map")
    Expr = readmap("Expr.map")
    Result = areamaximum( Expr, Class)
```

Result.map

-6	8	-6	-6	MV
8	8	-6	-6	-6
8	8	8	8	8
8	8	8	8	8
8	MV	MV	2.5	2.5

Class.map

2	6	2	2	MV
6	6	2	2	2
6	6	0	0	0
6	6	0	0	0
6	3	3	4	4

Expr.map

-9	0	-6	-6	-6
1	1	-6	-6	MV
1	1	-1	7	2
1	1	3	5	8
8	MV	MV	2.5	1.4

- Some other functions: areaaverage, areaminimum, etc.

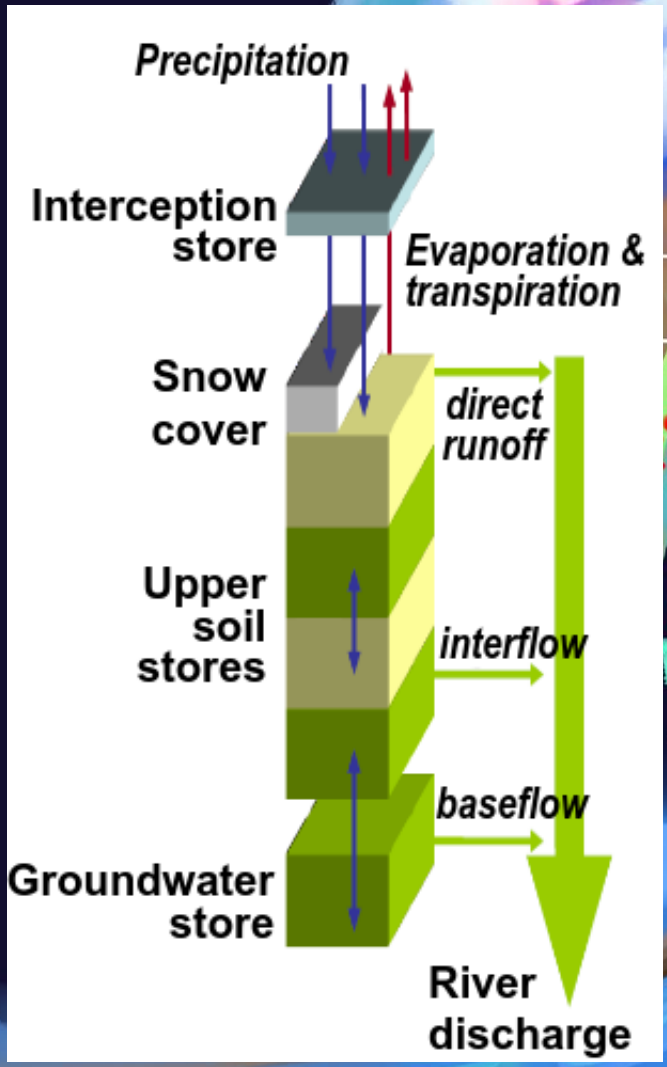
PCR-GLOBWB code (a cooking book about how to run the model)

PCR-GLOBWB 1.0 (structure)

- Original PCR-GLOBWB 1.0 (Van Beek et al, 2011)
- Resolution: 30 arc min (~50 km)
- Daily time step

- **Vertical** flow through each cell (POINT operations).
  - Sub-grid variabilities within cells are considered: topography, soil and land cover variations.

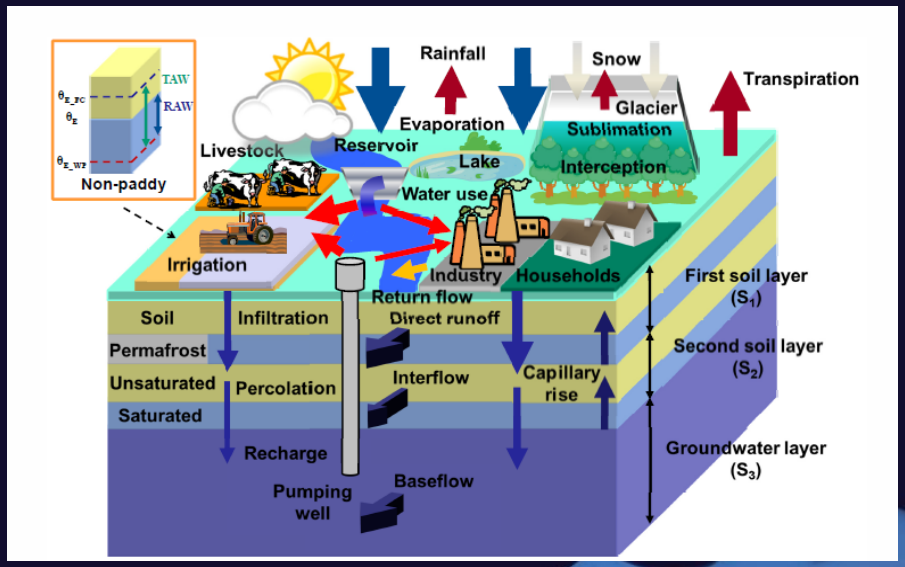
- Interaction between cells:
  - Channel routing along drainage network (LDD).
  - Surface water bodies represent either streams or lakes/reservoirs that buffer stream-flow.
  - 600 largest reservoirs.
  - Lakes and wetlands (including evaporation).

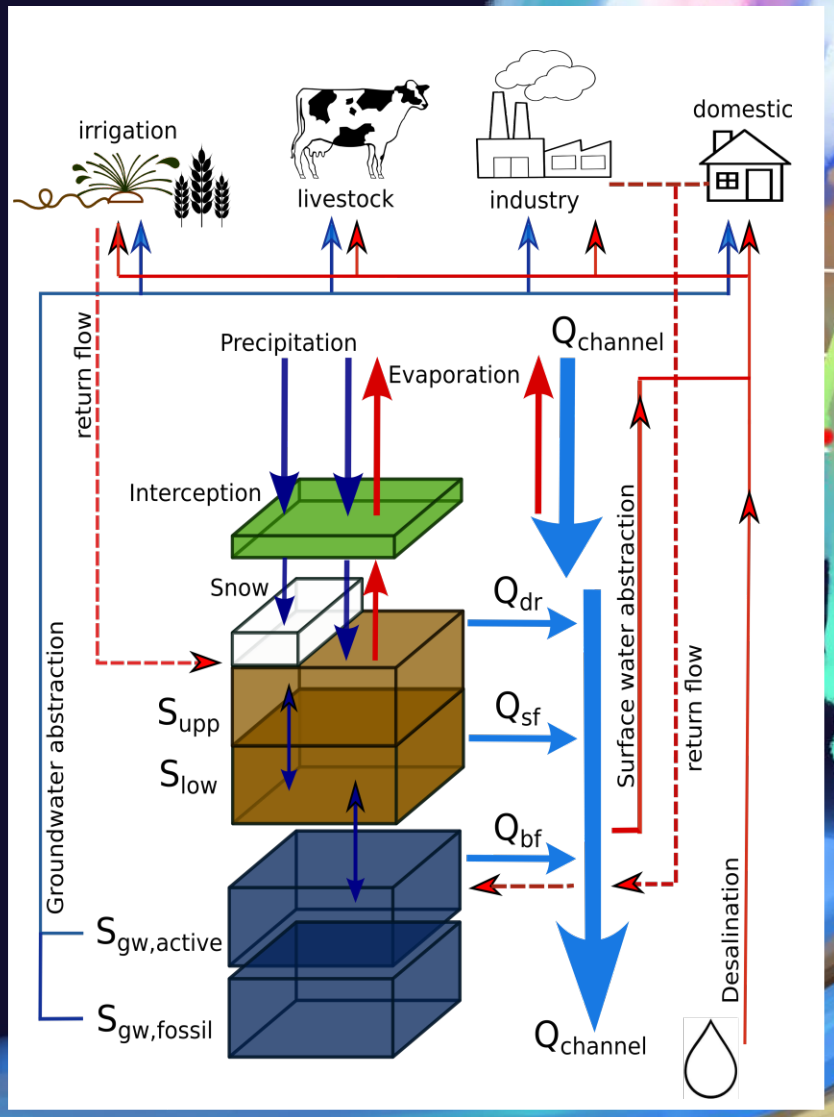


PCR-GLOBWB 2.0

- **Integrated hydrology & water resources model**, i.e. including online/interactive simulation of water demand and abstraction. Features added (examples):
  - Water demand & irrigation.
  - > 6000 reservoirs (GranD).
  - Integrated flood inundation module.

The model code is rewritten in Python and open source. Also, many refactoring actions have been done to speed up the computation (particularly for 5 arcmin ~10 km resolution)

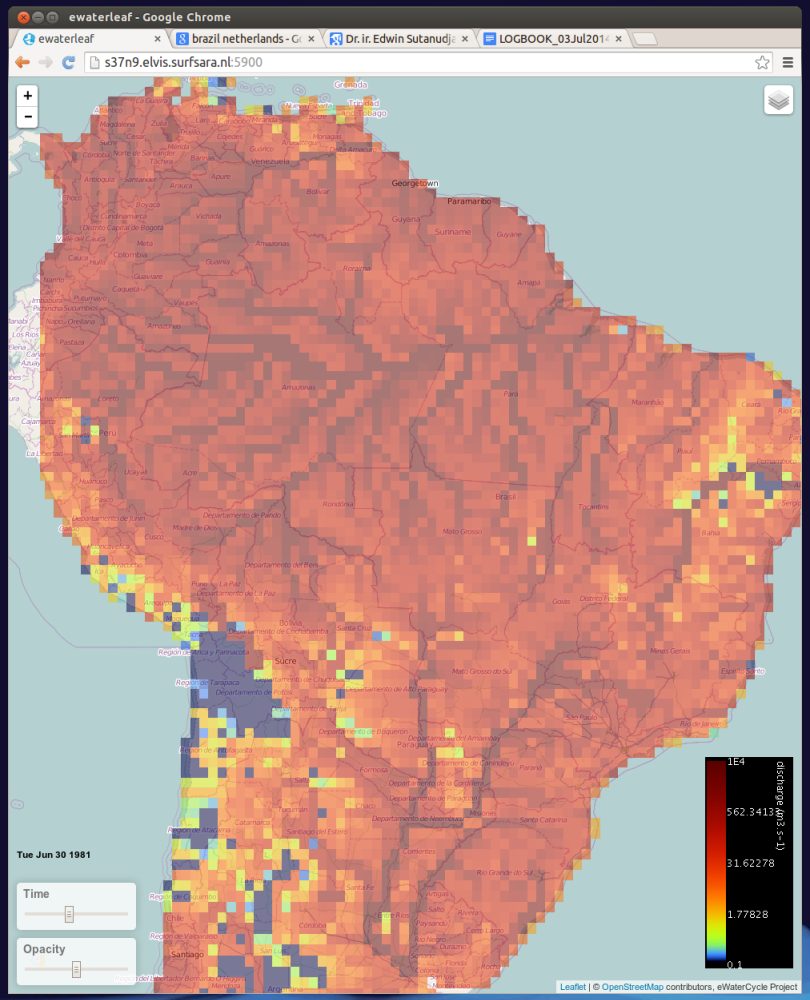


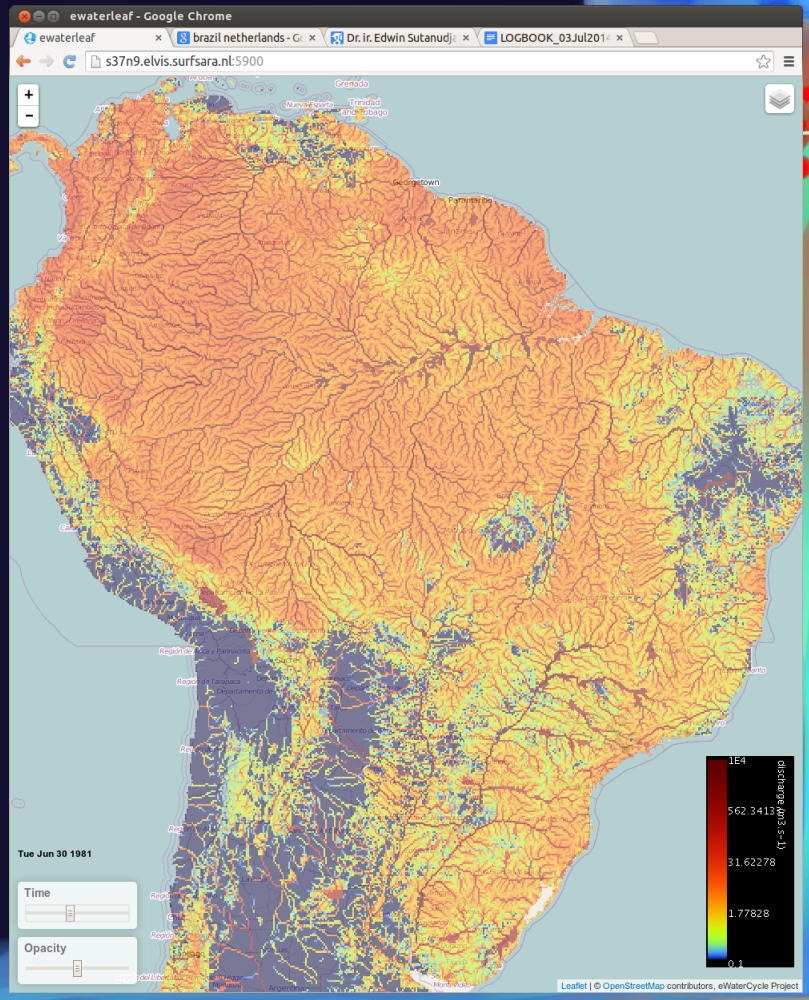


PCR-GLOBWB 2.0

River discharge at the resolutions:

30 (~55km) vs. 5 arc minutes (~10km)

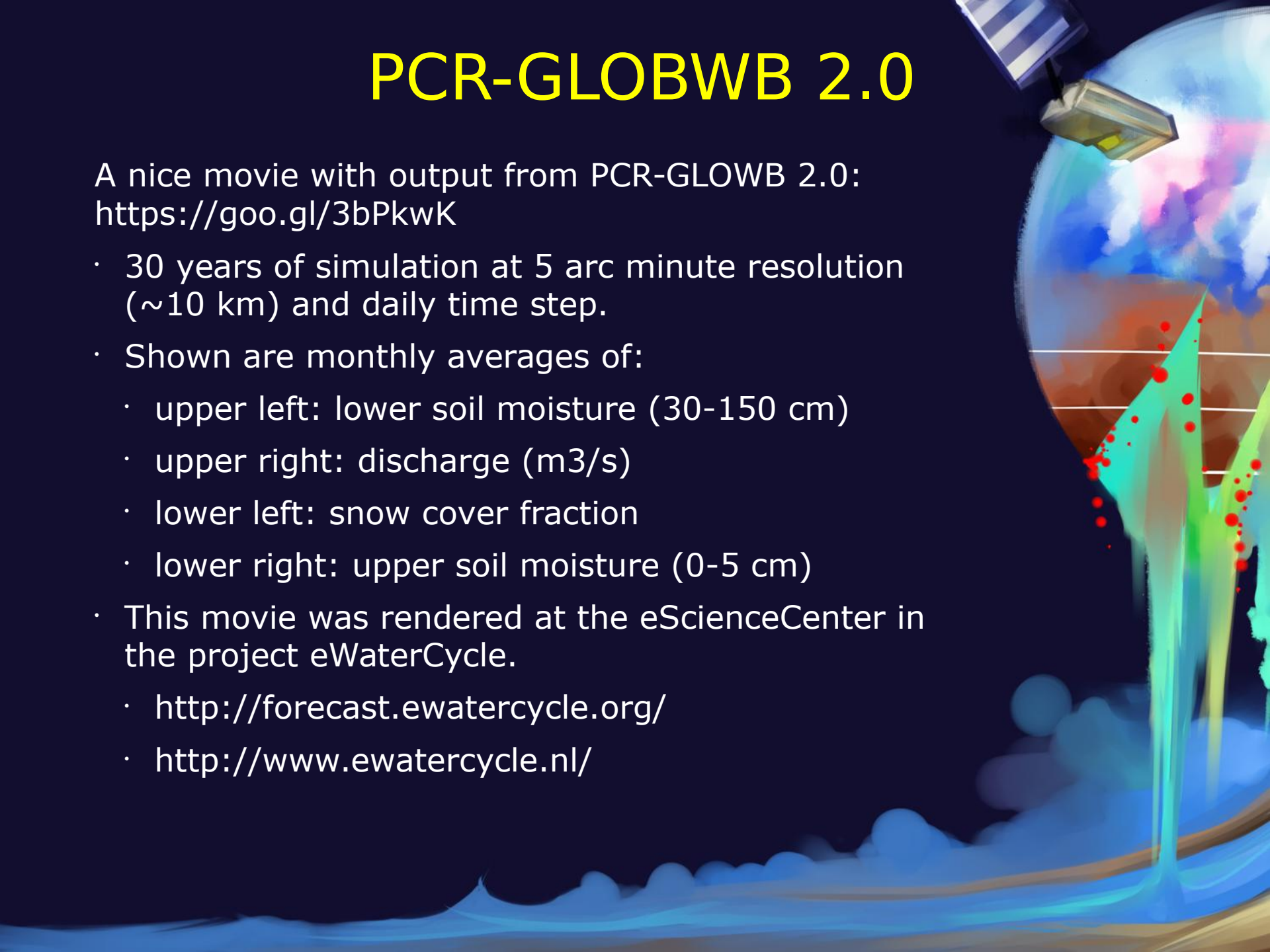




PCR-GLOBWB 2.0

A nice movie with output from PCR-GLOWB 2.0: https://goo.gl/3bPkwK

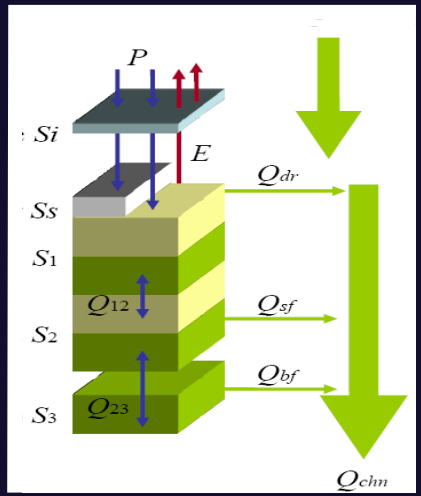
- 30 years of simulation at 5 arc minute resolution (~10 km) and daily time step.
- Shown are monthly averages of:
  - upper left: lower soil moisture (30-150 cm)
  - upper right: discharge (m3/s)
  - lower left: snow cover fraction
  - lower right: upper soil moisture (0-5 cm)
- This movie was rendered at the eScienceCenter in the project eWaterCycle.
  - http://forecast.ewatercycle.org/
  - http://www.ewatercycle.nl/



Refactoring model code

- PCR-GLOBWB 2.0 is re-written in Python (PCRaster framework).
  - Two possible resolutions: 5 arcmin and 30 arcmin
  - Suitable for regional models.

- A component based model.
- 4 main (hydrological) modules:
  - Meteo
  - Land surface
  - Groundwater
  - River / routing




```
class DeterministicRunner(DynamicModel):

    def __init__(self, configuration, model_time, initialState,

    def initial(self):

    def dynamic(self):
        #current model timestep
        self.modelTime.update(self.currentTimeStep())

        #update model
        self.model.read_forcings()
        self.model.read_parameters()
        self.model.calculate_new_state()

        #do any needed reporting for this time step
        self.reporting.report()
```

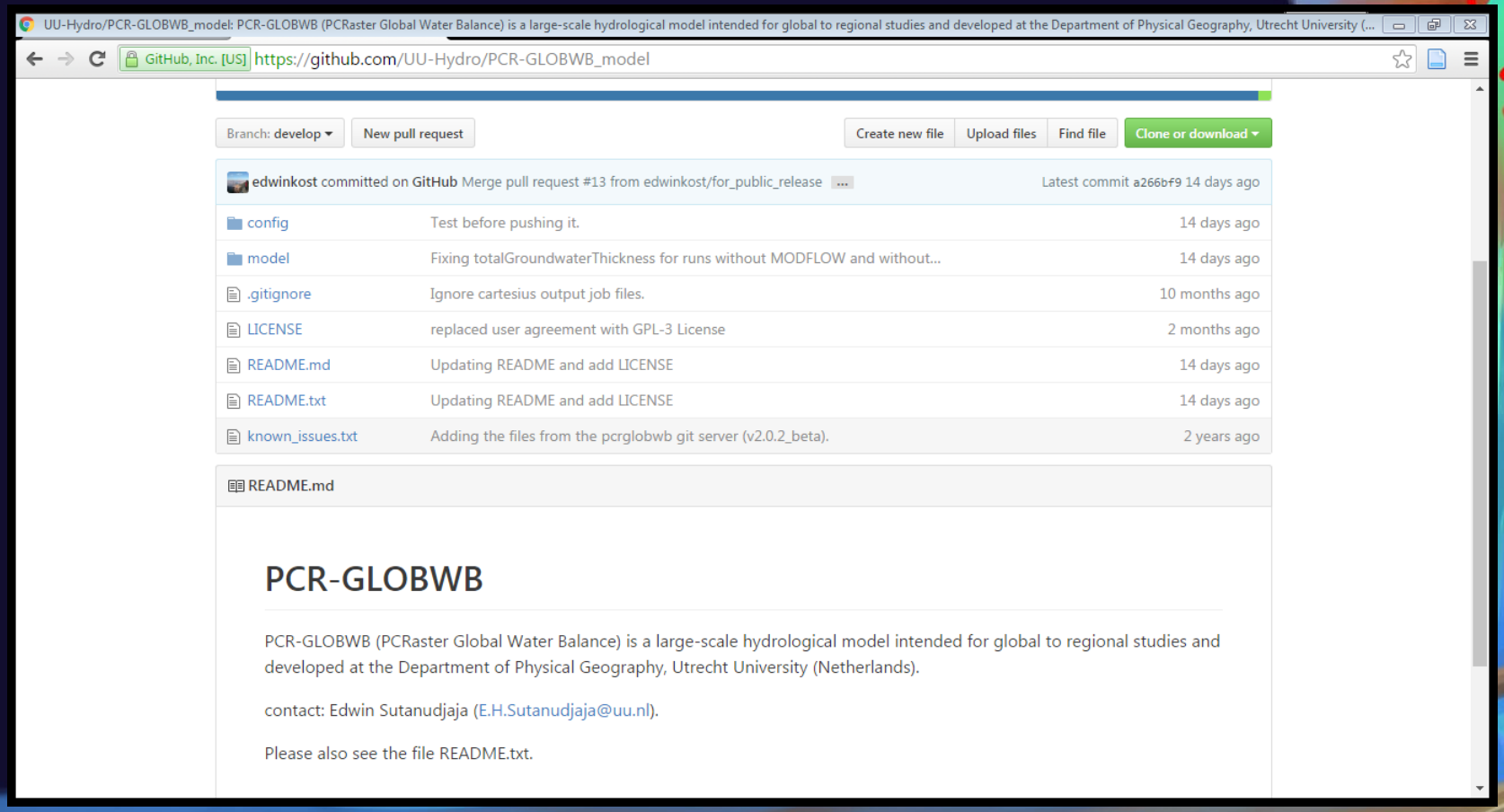
```
class PCRGlobWB(model_interface.ModelInterface):

    def __init__(self, configuration, model_time, \

    def calculate_new_state(self,report_water_balance):

        self.meteo.calculate_new_state(self._modelTime)
        self.landSurface.calculate_new_state(\
                         self.meteo,\
                         self.groundwater,\
                         self.routing,\
                         self._modelTime)
        self.groundwater.calculate_new_state(\
                         self.landSurface,\
                         self.routing,\
                         self._modelTime)
        self.routing.calculate_new_state(\
                         self.landSurface,\
                         self.groundwater,\
                         self._modelTime,\
                         self.meteo)
```

- Example: Sub-module for land surface: **Land cover**
  - Loop through 2 or 4 land covers:
    - Natural: forest + grassland
    - Irrigation: irrPaddy + irrNonPaddy
  - Sub-modules for each land cover:
    - interception, snow, impArnoScheme, soilFluxes

Output in netcdf ; most input files also in netcdf (some still in pcraster maps).

PCR-GLOBWB 2.0 (open source)

- Model code of PCR-GLOBWB 2.0 is available from the following public repository: https://github.com/UU-Hydro/PCR-GLOBWB_model
- For example input and configuration files (Rhine catchment) we refer to: https://github.com/UU-Hydro/PCR-GLOBWB_input_example
- Our global input file (300 GB): https://doi.org/10.5281/zenodo.1045338



script files and config files

- List of 'script' files and 'config' files:

```
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
ls model
ETPFunctions.py          imagemean.pyx                pcrglobwb.py
bmi.py                   landCover.py                 pcrglobwb_v1
bmiPcrglobwb.py          landSurface.py               reporting.py
configuration.py         meteo.py                     routing.py
currTimeStep.py          modflow.py                   setup.py
debug_to_version_one.sh  ncConverter.py               spinUp.py
deterministic_runner.py  oldcalc_framework.py         variable_list.py
disclaimer.py            parallel_pcrglobwb_runner.py virtualOS.py
groundwater.py           parameterSoilAndTopo.py      waterBodies.py
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
ls config/
05min_example  30min_example
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
ls config/05min_example/
setup_05min_global.ini
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
```

command to run the model:

Command to run the model:

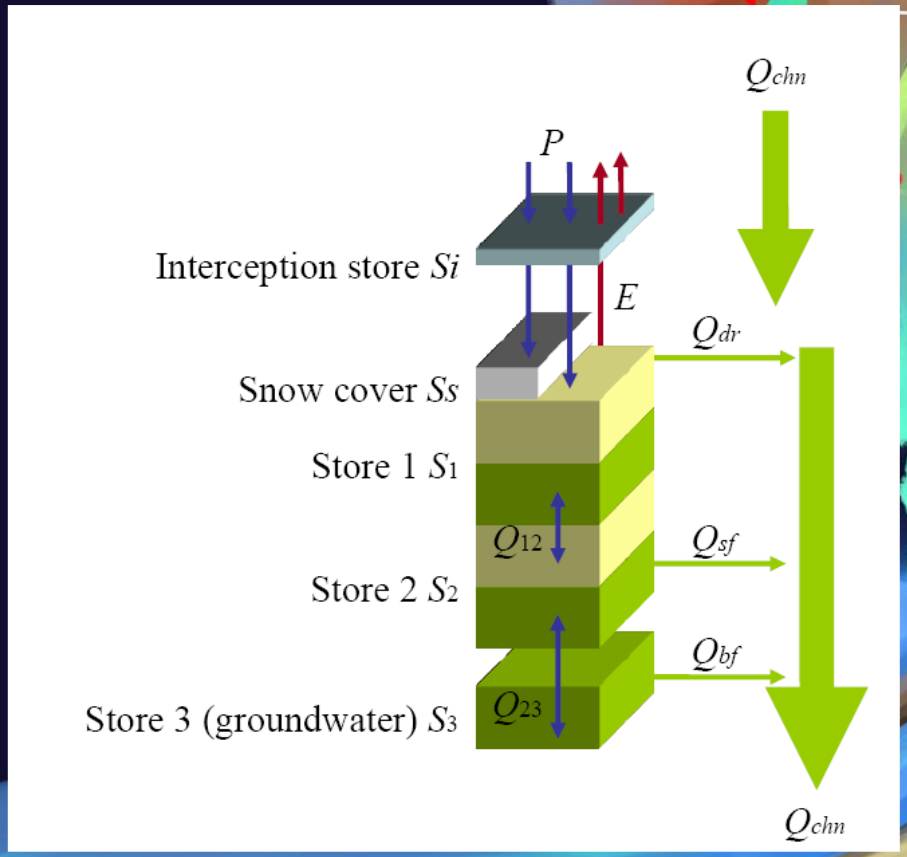
python deterministic_runner.py *<configuration_file_name>*

```
sutanudjaja@cehs-hp:~/github/UU-Hydro/PCR-GLOBWB_model$ cd model
sutanudjaja@cehs-hp:~/github/UU-Hydro/PCR-GLOBWB_model/model$ python deterministic_runner.py
../config/05min_example/setup_05min_global.ini


 PCR-GLOBWB (PCRaster Global Water Balance) Global Hydrological Model

 Copyright (C) 2016, Ludovicus P. H. (Rens) van Beek, Edwin H. Sutanudjaja, Yoshihide Wada,
 Joyce H. C. Bosmans, Niels Drost, Inge E. M. de Graaf, Kor de Jong, Patricia Lopez Lopez,
 Stefanie Pessenteiner, Oliver Schmitz, Menno W. Straatsma, Niko Wanders, Dominik Wisser,
 and Marc F. P. Bierkens,
 Faculty of Geosciences, Utrecht University, Utrecht, The Netherlands

 This program comes with ABSOLUTELY NO WARRANTY
 This is free software, and you are welcome to redistribute it under certain conditions
 See the LICENSE file for more details
```

script files

```
ETPFunctions.py          imagemean.pyx                  pcrglobwb.py
bmi.py                   landCover.py                   pcrglobwb_v1
bmiPcrglobwb.py          landSurface.py                 reporting.py
configuration.py         meteo.py                       routing.py
currTimeStep.py          modflow.py                     setup.py
debug_to_version_one.sh  ncConverter.py                 spinUp.py
deterministic_runner.py  oldcalc_framework.py           variable_list.py
disclaimer.py            parallel_pcrglobwb_runner.py   virtualOS.py
groundwater.py           parameterSoilAndTopo.py        waterBodies.py
```

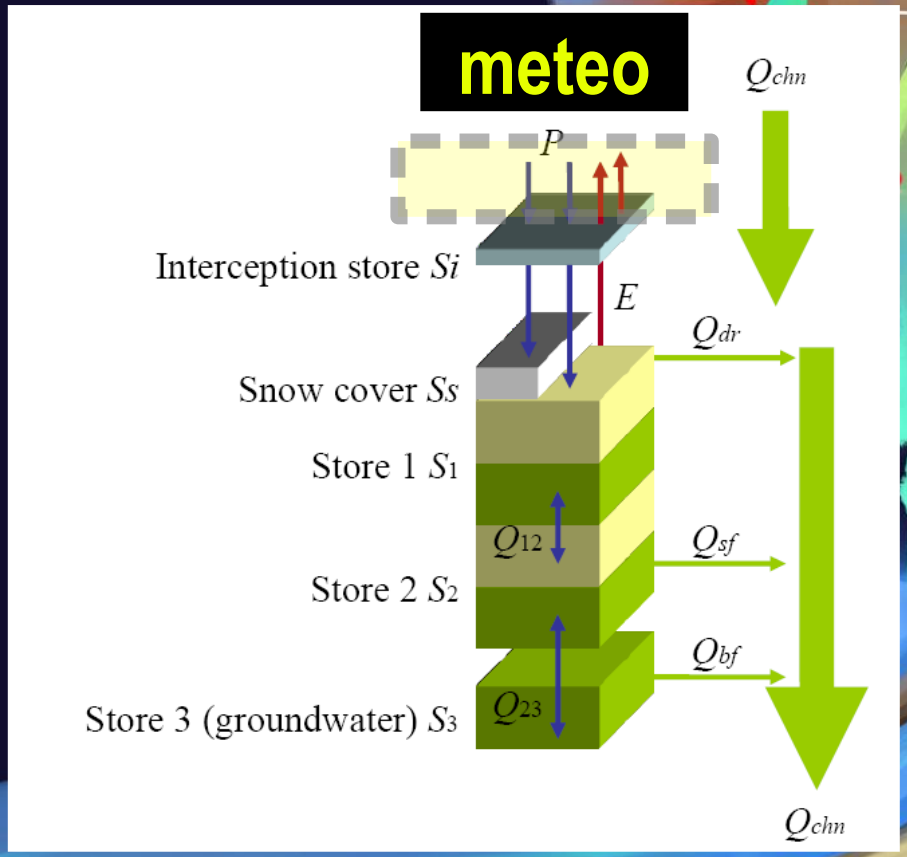
- **4 main hydrological modules:**
  - meteo.py
    - ETPFunctios.py
  - landSurface.py
    - parameterSoilAndTopo.py
    - landCover.py
  - groundwater.py
  - routing.py
    - waterBodies.py



script files

```
ETPFunctions.py          imagemean.pyx                  pcrglobwb.py
bmi.py                   landCover.py                   pcrglobwb_v1
bmiPcrglobwb.py          landSurface.py                 reporting.py
configuration.py         meteo.py                       routing.py
currTimeStep.py          modflow.py                     setup.py
debug_to_version_one.sh  ncConverter.py                 spinUp.py
deterministic_runner.py  oldcalc_framework.py           variable_list.py
disclaimer.py            parallel_pcrglobwb_runner.py   virtualOS.py
groundwater.py           parameterSoilAndTopo.py        waterBodies.py
```

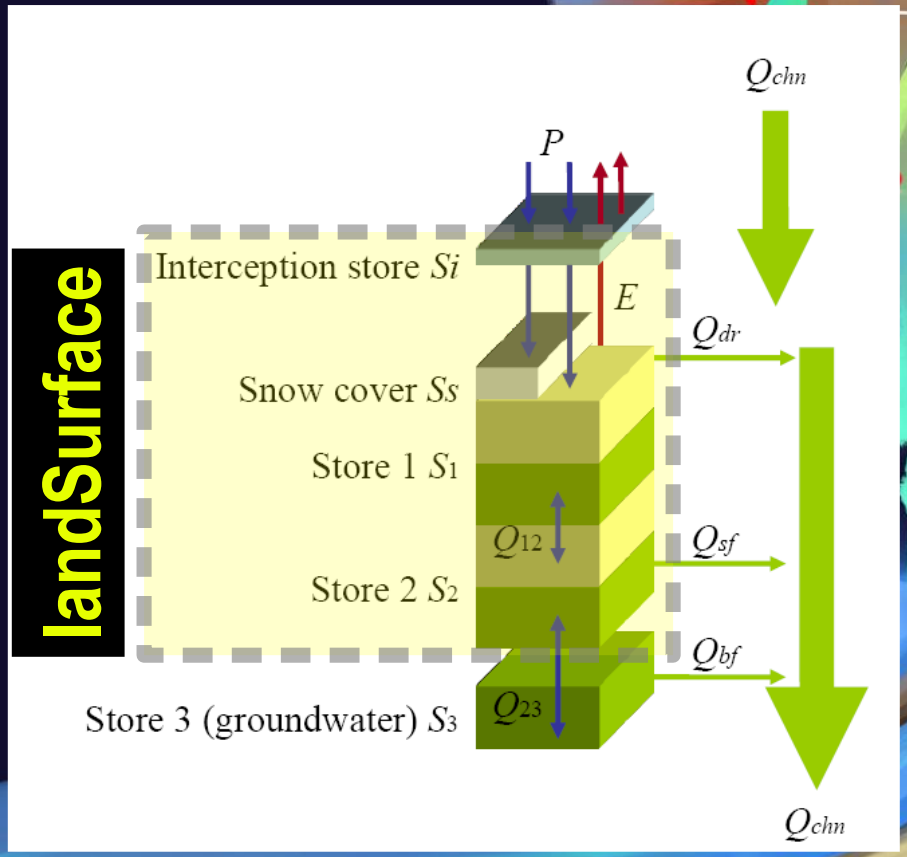
- **4 main hydrological modules:**
  - **meteo.py**
    - ETPFunctios.py
  - landSurface.py
    - parameterSoilAndTopo.py
    - landCover.py
  - groundwater.py
  - routing.py
    - waterBodies.py



script files

```
ETPFunctions.py              imagemean.pyx                pcrglobwb.py
bmi.py                       landCover.py                 pcrglobwb_v1
bmiPcrglobwb.py              landSurface.py               reporting.py
configuration.py             meteo.py                     routing.py
currTimeStep.py              modflow.py                   setup.py
debug_to_version_one.sh      ncConverter.py               spinUp.py
deterministic_runner.py      oldcalc_framework.py         variable_list.py
disclaimer.py                parallel_pcrglobwb_runner.py virtualOS.py
groundwater.py               parameterSoilAndTopo.py      waterBodies.py
```

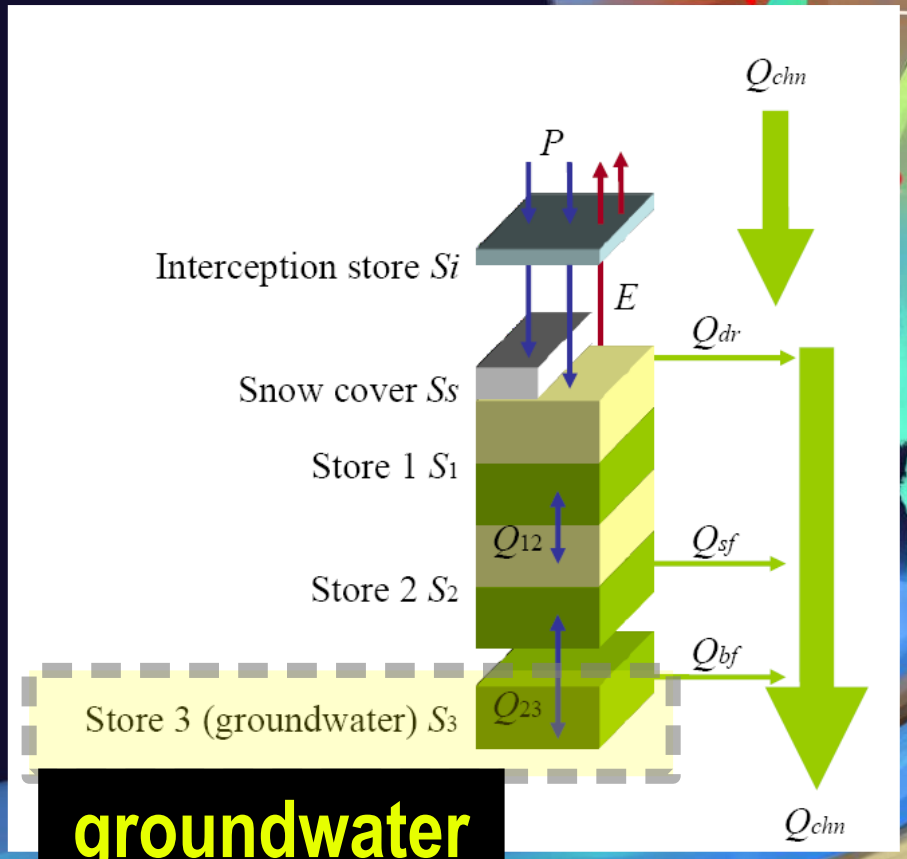
- **4 main hydrological modules:**
  - meteo.py
    - ETPFunctios.py
  - **landSurface.py**
    - parameterSoilAndTopo.py
    - landCover.py *-called 4 times*
  - groundwater.py
  - routing.py
    - waterBodies.py



script files

```
ETPFunctions.py           imagemean.pyx                  pcrglobwb.py
bmi.py                    landCover.py                   pcrglobwb_v1
bmiPcrglobwb.py           landSurface.py                 reporting.py
configuration.py          meteo.py                       routing.py
currTimeStep.py           modflow.py                     setup.py
debug_to_version_one.sh   ncConverter.py                 spinUp.py
deterministic_runner.py   oldcalc_framework.py           variable_list.py
disclaimer.py             parallel_pcrglobwb_runner.py   virtualOS.py
groundwater.py            parameterSoilAndTopo.py        waterBodies.py
```

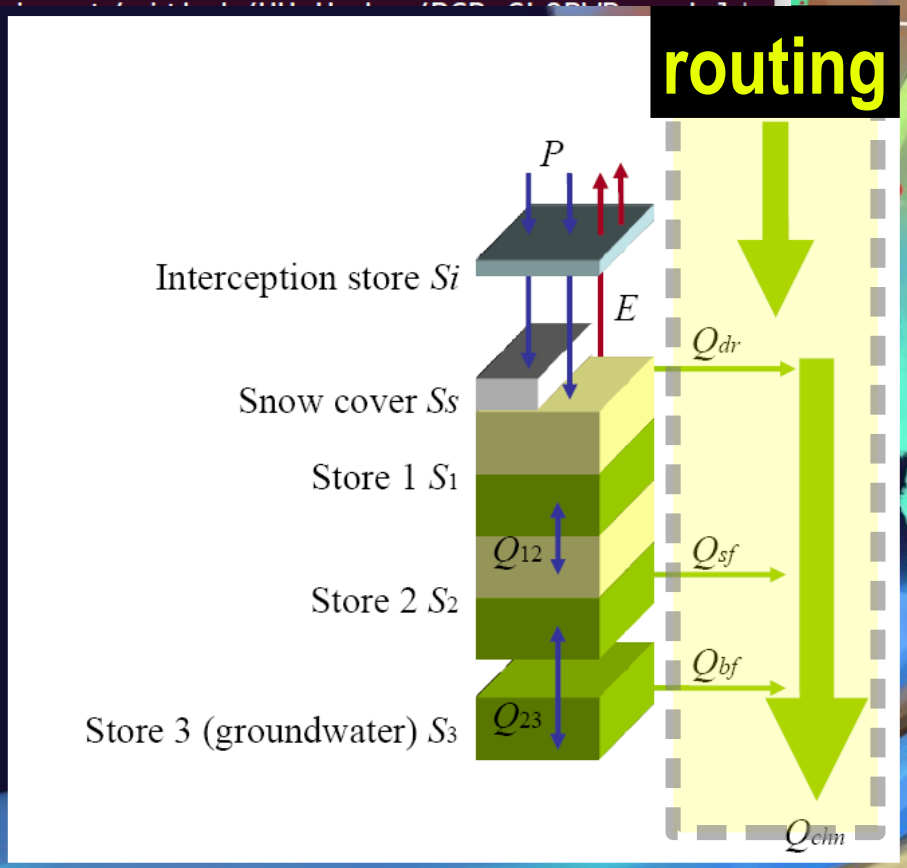
- **4 main hydrological modules:**
  - meteo.py
    - ETPFunctios.py
  - landSurface.py
    - parameterSoilAndTopo.py
    - landCover.py
  - **groundwater.py**
  - routing.py
    - waterBodies.py



script files

```
ETPFunctions.py           imagemean.pyx                  pcrglobwb.py
bmi.py                    landCover.py                   pcrglobwb_v1
bmiPcrglobwb.py           landSurface.py                 reporting.py
configuration.py          meteo.py                       routing.py
currTimeStep.py           modflow.py                     setup.py
debug_to_version_one.sh   ncConverter.py                 spinUp.py
deterministic_runner.py   oldcalc_framework.py           variable_list.py
disclaimer.py             parallel_pcrglobwb_runner.py   virtualOS.py
groundwater.py            parameterSoilAndTopo.py        waterBodies.py
```

- **4 main hydrological modules:**
  - meteo.py
    - ETPFunctios.py
  - landSurface.py
    - parameterSoilAndTopo.py
    - landCover.py
  - groundwater.py
  - **routing.py**
    - waterBodies.py



ini files

```
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
ls config/
05min_example  30min_example
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
ls config/05min_example/
setup_05min_global.ini
edwinsut@int1.cartesius.surfsara.nl:/home/edwinsut/github/UU-Hydro/PCR-GLOBWB_model$
```

- What should be defined in a configuration file (*.ini)?
  - Input file locations or input values
    - Forcing: precipitation, temperature, etc.
    - Water demand files
    - Model parameters
    - Initial conditions.
  - Other configuration: startTime, endTime, cloneMap, landmask, etc.
  - Which output variables do you want to report?

ini files

- Structure of a configuration file

```
  1
  2 [globalOptions]
 36
 37 [meteoOptions]
 49 [meteoDownscalingOptions] # only for 5 arcmin runs
 71
 72 [landSurfaceOptions]
119 [forestOptions]
163 [grasslandOptions]
207 [irrPaddyOptions]
250 [irrNonPaddyOptions]
293
294 [groundwaterOptions]
337
338 [routingOptions]
407
408 [reportingOptions]
432
433 [mergingOutputOptions] # only for parallel runs
434
```

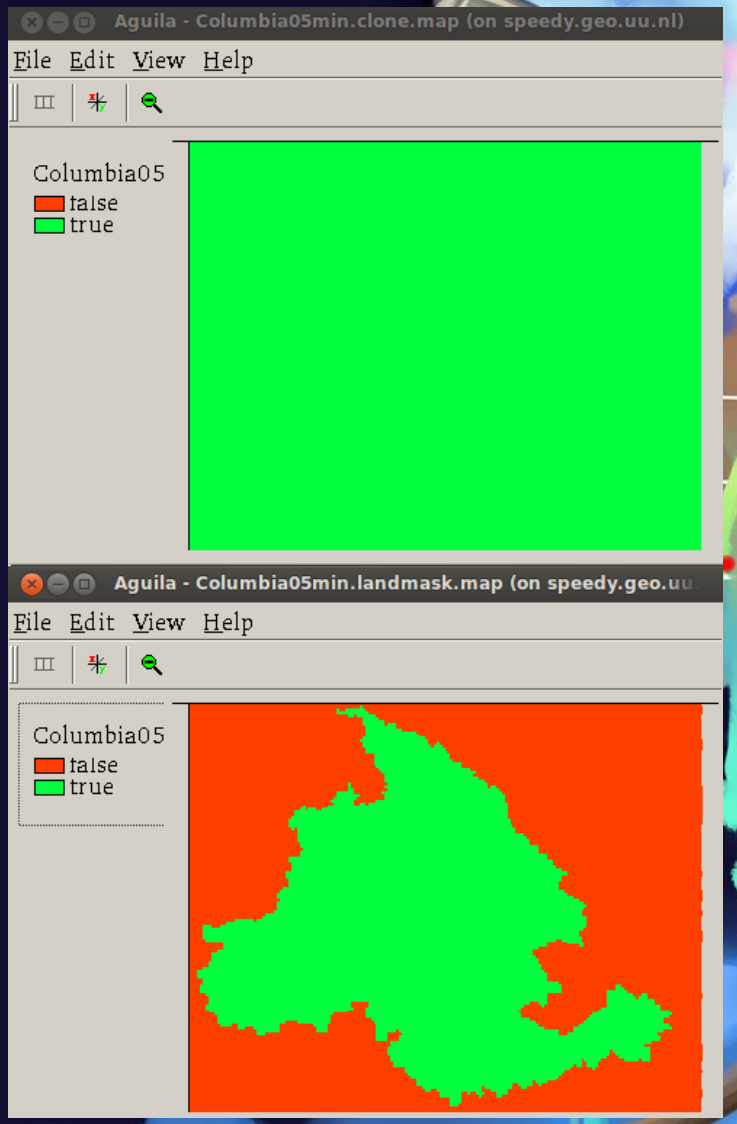
Location of PCR-GLOBWB input data

- All input data are provided in the global coverage.
- On speedy/rapid (our servers): /data/hydroworld/
- On Cartesius: /projects/0/dfguu/data/hydroworld/
- The fixed/stable/release version is available on https://doi.org/10.5281/zenodo.1045338 (300 GB)
- A small input example (Rhine-Meuse) is also provided on https://github.com/UU-Hydro/PCR-GLOBWB_input_example

```
sutan101@speedy:/data/hydroworld$
sutan101@speedy:/data/hydroworld$ ls
basedata  forcing  others  parameterizationscripts  PCRGLOBWB10  PCRGLOBWB20  synchronize.sh
sutan101@speedy:/data/hydroworld$
sutan101@speedy:/data/hydroworld$ cd PCRGLOBWB20/
sutan101@speedy:/data/hydroworld/PCRGLOBWB20$
sutan101@speedy:/data/hydroworld/PCRGLOBWB20$ ls -l
total 12
drwxrwsr-x 8 sutan101 hydroworld 4096 Mar 12 14:28 input30min
drwxrwsr-x 9 straa005 hydroworld 4096 Feb  4 14:24 input5min
drwxrwsr-x 4 straa005 hydroworld 4096 Oct  4  2013 scripts
sutan101@speedy:/data/hydroworld/PCRGLOBWB20$ cd input5min/
sutan101@speedy:/data/hydroworld/PCRGLOBWB20/input5min$ ls -l
total 28
drwxrwsr-x 2 straa005 hydroworld 4096 Aug 27  2013 global
drwxrwsr-x 3 straa005 hydroworld 4096 Mar  5 17:32 groundwater
drwxr-sr-x 3 beek0120 hydroworld 4096 Dec 24 16:54 hydeLandCover
drwxrwsr-x 6 straa005 hydroworld 4096 Mar 11 11:14 landCover
drwxrwsr-x 5 straa005 hydroworld 4096 Mar 27 00:38 landSurface
drwxrwsr-x 3 straa005 hydroworld 4096 Mar 11 10:18 meteo
drwxrwsr-x 4 straa005 hydroworld 4096 Mar  7 14:26 routing
sutan101@speedy:/data/hydroworld/PCRGLOBWB20/input5min$
```

globalOptions in the ini file

```
[globalOptions]

# Set the input directory map in an absolute path.
# - The input forcing and parameter directories will be relative to this.
inputDir = /data/hydroworld/

# Set the output directory in an absolute path.
outputDir = /scratch/edwin/test_3_layers/

# Map of clone (must be provided in PCRaster maps)
# - Spatial resolution and coverage are based on this map:
cloneMap = others/Columbia/Columbia05min.clone.map

# The area/landmask of interest:
landmask = others/Columbia/Columbia05min.landmask.map
# If None, area/landmask is limited for cells with ldd value.

# start and end simulation period
startTime = 1979-01-01
endTime   = 2010-12-31
# Format: YYYY-MM-DD
timeStep = 1.0
timeStepUnit = day
# PS: The current model still runs of the daily time step.

# spinning up options:
# Note: for the purpose of DA, there should be no SpinUp
maxSpinUpsInYears = 5
minConvForTotlSto = 0.5
minConvForSoilSto = 0.5
minConvForGwatSto = 0.5
minConvForChanSto = 0.5
```



- cloneMap: all cells have TRUE values
  - Model area and model resolution.
  - Border coordinates must be in integer.
- landmask: only TRUE at your area of interest

Fig. CloneMap & landmask maps at 5 arc-min for Columbia river basin.

meteoOptions in the ini file

```
[meteoOptions]

# Set the forcing temperature and precipitation files (relative to inputDir)
temperatureNC    = forcing/ERA-Interim-GPCPCorrected/temperature_ERA_Interim_1979to2010.nc
precipitationNC = forcing/ERA-Interim-GPCPCorrected/precipitation_ERA_Interim_GPCPCorrected_1979to2010.nc

# Method to calculate referencePotETP (reference potential evaporation+transpiration)
referenceETPotMethod = Hamon
# options are "Hamon" and "Input" ; If "Input", the netcdf input file must be given:
refETPotFileNC = None

...

[meteoDownscalingOptions]
# These options are only used and tested for 5 min runs which are using 30 min forcing

downscalePrecipitation  = True
downscaleTemperature    = True
downscaleReferenceETPot = True
...

# lapse rates:
temperLapseRateNC = PCRGLOBWB20/input5min/meteo/downscalingFrom30ArcMin/temperature_slope.nc
precipLapseRateNC = PCRGLOBWB20/input5min/meteo/downscalingFrom30ArcMin/precipitation_slope.nc
```

- Precipitation and temperature files must be in the netcdf format.
- You can also use “absolute path” (while defining your own forcing).
- Downscaling options are only used (and tested) for 5 min runs.

landSurfaceOptions in the ini file

```
[landSurfaceOptions]
...

numberOfUpperSoilLayers = 2
# 2 soil layers: 30 cm and 120 cm; 3 soil layers: 5 cm, 25 cm and 120 cm

includeIrrigation = True
# if True, there are four land cover types defined: forest,grassland,irrPaddy,irrNonPaddy
# if False, two (natural) land cover types defined: forest,grassland
historicalIrrigationArea = PCRGLOBWB20/input30min/landSurface/waterDemand/irrigated_areas/irrigationArea30ArcMin.nc

includeDomesticWaterDemand = True
includeIndustryWaterDemand = True
domesticWaterDemandFile = PCRGLOBWB20/input30min/landSurface/waterDemand/domesticWaterDemand30ArcMin.nc
industryWaterDemandFile = PCRGLOBWB20/input30min/landSurface/waterDemand/industryWaterDemand30ArcMin.nc

limitAbstraction = False
```

- 3 soil layers are mainly used for the eWaterCycle project.
- IncludeIrrigation = True → 4 land covers = forest, grassland, irrPaddy & irrNonPaddy
- limitAbstraction = False → allowing other abstraction sources (e.g. fossil water)

*<landCover>*Options (2 layer model)

```
[forestOptions]
...
# snow module properties
snowModuleType      =  Simple
freezingT           = -0.0
degreeDayFactor     =  0.0025
...

# land cover properties:
landCoverMapsNC   = PCRGLOBWB20/input30min/landCover/forest/forestProperties.nc
cropCoefficientNC = PCRGLOBWB20/input30min/landCover/forest/Global_CropCoefficientKc-Forest_30min.nc
interceptCapNC    = PCRGLOBWB20/input30min/landCover/forest/interceptCapInputForest366days.nc
coverFractionNC   = PCRGLOBWB20/input30min/landCover/forest/coverFractionInputForest366days.nc

# initial conditions (for two layer model):
interceptStorIni = 0.0
snowCoverSWEIni  = 0.0
snowFreeWaterIni = 0.0
topWaterLayerIni = 0.0
storUppIni       = 0.0
storLowIni       = 0.0
interflowIni     = 0.0
...

[grasslandOptions]
[irrPaddyOptions]
[irrNonPaddyOptions]
```

- Each landCover has each own field.
- Initial Conditions must be defined in each land cover.
- The example above is for a **2-layer model**.
- For a 3-layer model, initial conditions are different (i.e. there are more states).

*<landCover>*Options (3 layer model)

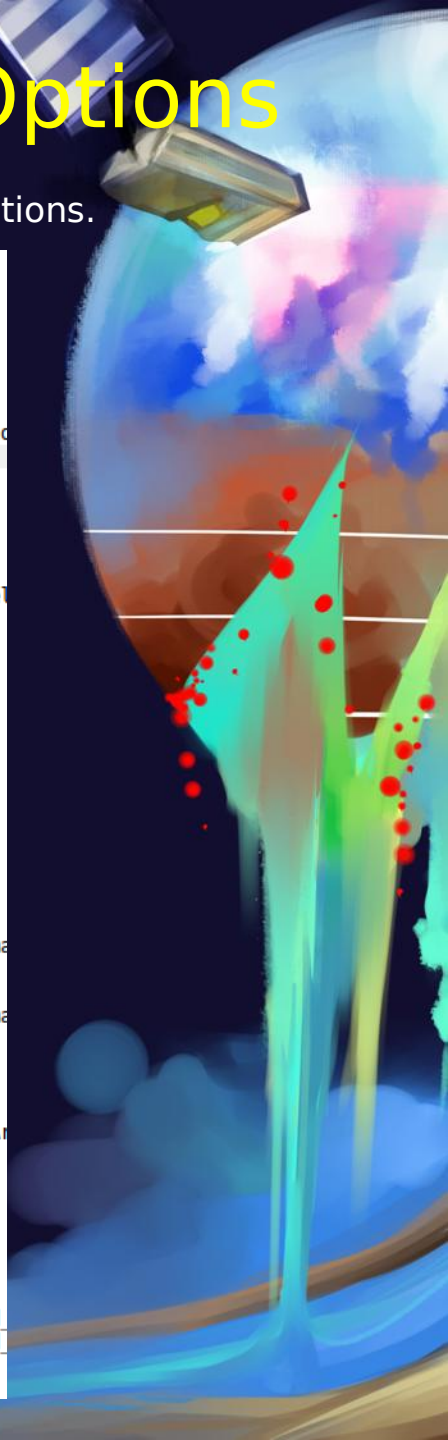
```
[forestOptions]
...
# snow module properties
snowModuleType      =  Simple
freezingT           = -0.0
degreeDayFactor     =  0.0025
...

# land cover properties:
landCoverMapsNC   = PCRGLOBWB20/input30min/landCover/forest/forestProperties.nc
cropCoefficientNC = PCRGLOBWB20/input30min/landCover/forest/Global_CropCoefficientKc-Forest_30min.nc
interceptCapNC    = PCRGLOBWB20/input30min/landCover/forest/interceptCapInputForest366days.nc
coverFractionNC   = PCRGLOBWB20/input30min/landCover/forest/coverFractionInputForest366days.nc

# initial conditions (for three layer model):
interceptStorIni = 0.0
snowCoverSWEIni  = 0.0
snowFreeWaterIni = 0.0
topWaterLayerIni = 0.0
storUpp000005Ini = 0.0
storUpp005030Ini = 0.0
storLow030150Ini = 0.0
interflowIni     = 0.0
...

[grasslandOptions]
[irrPaddyOptions]
[irrNonPaddyOptions]
```

- Each landCover has each own field.
- Initial Conditions must be defined in each land cover.
- The example above is for a **3-layer model**.
- For a 2-layer model, initial conditions are different (i.e. there are fewer states).

groundwaterOptions & routingOptions

- Both 2-layer & 3-layer models have the same groundwaterOptions & routingOptions.

```
[groundwaterOptions]

groundwaterPropertiesNC = PCRGLOBWB20/input5min/groundwater/groundwaterProperties5ArcMin.nc

...

# initial conditions:
storGroundwaterIni                      = /projects/0/aqueduct/users/edwinsut/pcrglobwb_runs_2016_oct_nov/pcrglo
...

[routingOptions]

lddMap      = PCRGLOBWB20/input5min/routing/lddsound_05min.map
cellAreaMap = PCRGLOBWB20/input5min/routing/cellsize05min.correct.map
gradient    = /projects/0/dfguu/users/edwin/data/floodplain_05arcmin_world_final/based_on_daily_runoff/map/channel

# manning coefficient
manningsN   = 0.04

# manning coefficient for floodplain
floodplainManningsN = 0.07

# routing method: (options are kinematicWave, simplifiedKinematicWave and accuTravelTime)
routingMethod = accuTravelTime

# Option for flood plain simulation
dynamicFloodPlain = True

# constant channel width (optional)
constantChannelWidth = /projects/0/dfguu/users/edwin/data/floodplain_05arcmin_world_final/based_on_daily_runoff/ma

# constant channel depth
constantChannelDepth = /projects/0/dfguu/users/edwin/data/floodplain_05arcmin_world_final/based_on_daily_runoff/ma

...

# lake and reservoir parameters
waterBodyInputNC = PCRGLOBWB20/input5min/routing/reservoirs/waterBodiesFinal_version15Sept2013/maps/waterBodies5Ar
onlyNaturalWaterBodies = False

# composite crop factors for WaterBodies:
cropCoefficientWaterNC = PCRGLOBWB20/input30min/routing/cropCoefficientForOpenWater.nc
...

# initial conditions:
waterBodyStorageIni            = /projects/0/aqueduct/users/edwinsut/pcrglobwb_runs_2016_oct_nov/pcrglobwb_4_land_
channelStorageIni              = /projects/0/aqueduct/users/edwinsut/pcrglobwb_runs_2016_oct_nov/pcrglobwb_4_land_
...
```



reportingOptions

```
[reportingOptions]

# output files that will be written in the disk in netcdf files:
# - daily resolution
outDailyTotNC = discharge
# - monthly resolution
outMonthAvgNC =
discharge,dynamicFracWat,surfaceWaterStorage,interceptStor,snowFreeWater,snowCoverSWE,topWaterLayer,storUppTotal,storLowTotal,storG
roundwater,storGroundwaterFossil,totalActiveStorageThickness,totalWaterStorageThickness,satDegUpp,satDegLow,channelStorage,waterBod
yStorage
outMonthEndNC =
storGroundwater,storGroundwaterFossil,waterBodyStorage,channelStorage,totalWaterStorageThickness,totalActiveStorageThickness
outMonthTotNC =
totalEvaporation,waterBodyActEvaporation,actualET,irrPaddyWaterWithdrawal,irrNonPaddyWaterWithdrawal,domesticWaterWithdrawal,indust
ryWaterWithdrawal,livestockWaterWithdrawal,precipitation,gwRecharge,runoff,totalRunoff,baseflow,directRunoff,interflowTotal,totalGr
oundwaterAbstraction,desalinationAbstraction,surfaceWaterAbstraction,nonFossilGroundwaterAbstraction,fossilGroundwaterAbstraction,i
rrGrossDemand,nonIrrGrossDemand,totalGrossDemand,nonIrrWaterConsumption,nonIrrReturnFlow
# - annual resolution
outAnnuaAvgNC = None
outAnnuaEndNC = None
outAnnuaTotNC = None
outMonthMaxNC = None
outAnnuaMaxNC = None

# netcdf format and zlib setup
formatNetCDF = NETCDF4
zlib = True
```

- Output variables can be reported in **Daily**, **Month**ly and **Annua**l resolution; in cumulative (**Tot**), average (**Avg**), end of a period (**End**), or maximum (**Max**) values.

list of output variables

- See the model script file “variable_list.py” (inside the “model” folder).

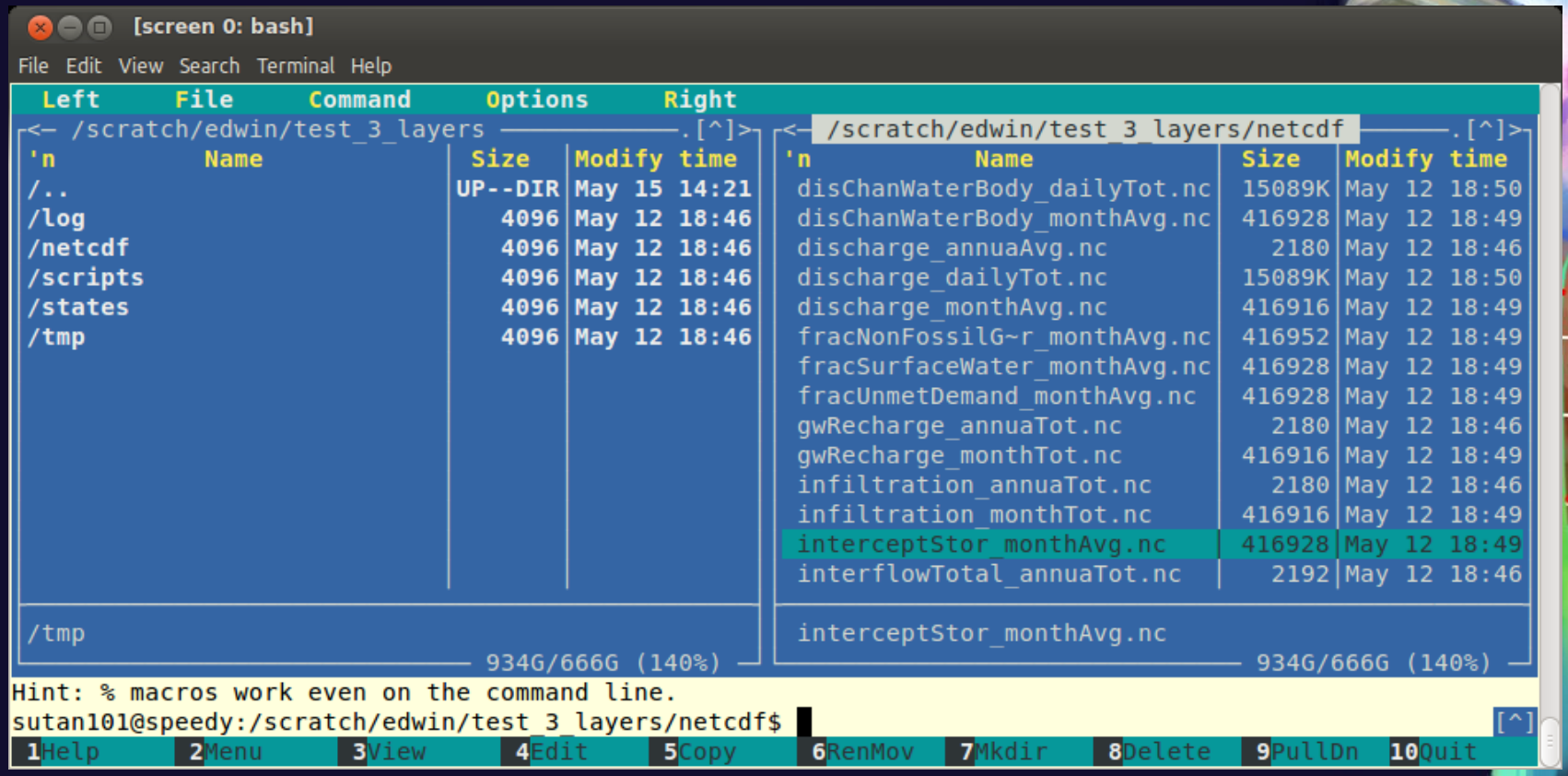
```
# totLandSurfaceActuaET
pcrglobwb_variable_name = 'totLandSurfaceActuaET'
netcdf_short_name[pcrglobwb_variable_name] = 'land_surface_actual_evaporation'
netcdf_unit[pcrglobwb_variable_name]       = 'm.day-1'
netcdf_monthly_total_unit[pcrglobwb_variable_name] = 'm.month-1'
netcdf_yearly_total_unit[pcrglobwb_variable_name]  = 'm.year-1'
netcdf_long_name[pcrglobwb_variable_name]  = 'total_actual_evaporation_and_transpiration_at_land_surface'
description[pcrglobwb_variable_name]       = None
comment[pcrglobwb_variable_name]           = 'Not including water bodies. Values given are over the entire cell
area.'
latex_symbol[pcrglobwb_variable_name]      = None

# fractionLandSurfaceET
pcrglobwb_variable_name = 'fractionLandSurfaceET'
netcdf_short_name[pcrglobwb_variable_name] = 'land_surface_evaporation_fraction'
netcdf_unit[pcrglobwb_variable_name]       = '1'
netcdf_monthly_total_unit[pcrglobwb_variable_name] = None
netcdf_yearly_total_unit[pcrglobwb_variable_name]  = None
netcdf_long_name[pcrglobwb_variable_name]  =
'ratio_between_actual_and_potential_values_of_evaporation_and_transpiration_at_land_surface'
description[pcrglobwb_variable_name]       = None
comment[pcrglobwb_variable_name]           = 'Not including water bodies.'
latex_symbol[pcrglobwb_variable_name]      = None

# interceptStor
pcrglobwb_variable_name = 'interceptStor'
netcdf_short_name[pcrglobwb_variable_name] = 'interception_storage'
netcdf_unit[pcrglobwb_variable_name]       = 'm'
netcdf_monthly_total_unit[pcrglobwb_variable_name] = None
```

mergingOutputOptions

- Only used for global 5 arcmin runs that are parallelized.
- List of output variables that are reported and will be merged.

```
[reportingOptions]
# output files that will be written in the disk in netcdf files:
# - daily resolution
outDailyTotNC = discharge
# - monthly resolution
outMonthAvgNC = totalEvaporation,gwRecharge,totalRunoff,totalGroundwaterAbstraction
outMonthEndNC =
storGroundwater,storGroundwaterFossil,waterBodyStorage,channelStorage,totalWaterStorageThickness,totalActiveStorageThickness
outMonthTotNC = None
# - annual resolution
outAnnuaAvgNC = None
outAnnuaEndNC = None
outAnnuaTotNC = None
outMonthMaxNC = None
outAnnuaMaxNC = None

[mergingOutputOptions]
# output variables/files that will be merged:
outDailyTotNC = None
outMonthAvgNC = totalEvaporation,gwRecharge
outMonthEndNC = None
outMonthTotNC = None
outAnnuaAvgNC = None
outAnnuaEndNC = None
outAnnuaTotNC = None
outMonthMaxNC = None
outAnnuaMaxNC = None
```

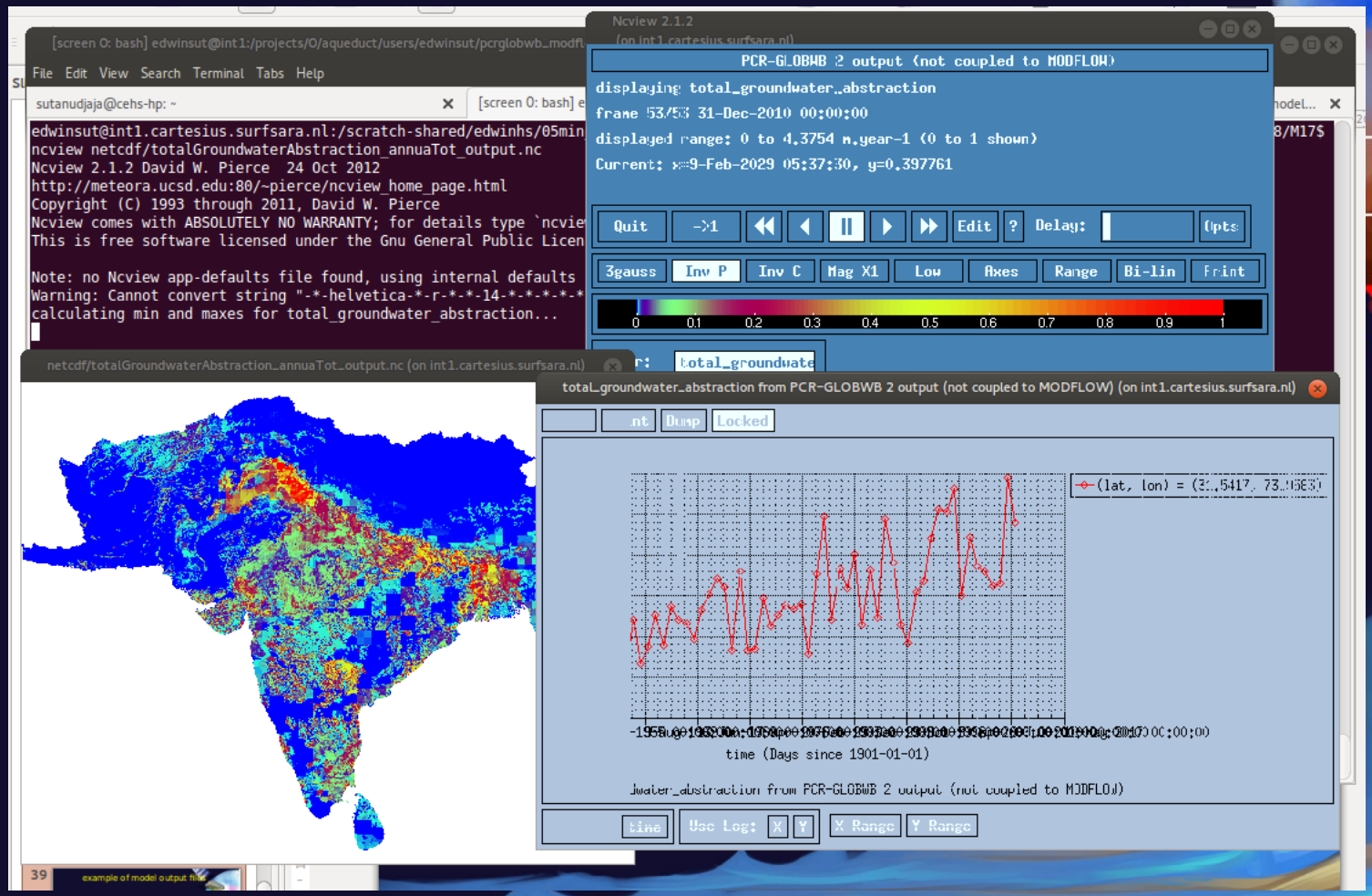
check your output / model results



- Contents of the outputDir: **log** (including backup of the ini file); **netcdf** output ; backup **scripts** used to run the model ; model **states** at the end of each year ; **tmp** directory (used during the resampling process)
- We will explore these during the exercises.
- To visualize a certain netcdf output variable: **ncview *<name_of_netcdf_file>***

example of model output files

- Output files are mainly in netdf files (some are in pcraster maps)
- To visualize a netcdf file: **ncview** ***<name_of_netcdf_file>***



Questions?

E.H.Sutanudjaja@uu.nl ; +31 30 253 2988 ; Room: Zon104

Universiteit Utrecht