Sustainable scientific software: experiences of the PCRaster research and development team

Oliver Schmitz, Kor de Jong and Derek Karssenberg
Department of Physical Geography, Utrecht University, The Netherlands (o.schmitz@uu.nl)

Universiteit Utrecht
Faculty of Geosciences

Motivation and approach

The PCRaster modelling framework

- Developed since 1989, open source since 2007
- Is targeted at the development of spatio-temporal simulation models
- Fast model development and execution
- Scripting environments: PCRcalc and Python
- Rich set of model building blocks for manipulating raster maps
- Framework for stochastic spatio-temporal model building and data assimilation
- Tool for visualisation of spatio-temporal stochastic data

Core functionality

Solution scheme for stochastic modelling

for each n in Monte Carlo samples:
for each t in time steps:

$$\mathbf{z}_t^{(n)} = f(\mathbf{z}_{t-1}^{(n)}, \mathbf{i}_t^{(n)}, \mathbf{p}^{(n)})$$

state variables — transition function — inputs — parameters

Solution framework (Python)

```
from pcraster import *
from pcraster.framework import *

class SnowModel(DynamicModel, MonteCarloModel):
  def __init__(self):
    ...
  def premcloop(self):                          # sets constant variables
    dem = self.readmap("dem")                   # and parameters
    self.ldd = lddcreate(dem, ...)
    ...
  def initial(self):                            # is run at t = 0 for each
    self.snow = scalar(0)                       # Monte Carlo sample
    ...
  def dynamic(self):                            # is run for each Monte Carlo sample
    runoff = accuflux(self.ldd, rain)           # and for each time step
    self.report(runoff, "q")
    ...
  def postmcloop(self):                         # is run at end calculating sampling
    mcpercentiles("q",percentiles,..)           # statistics over Monte Carlo samples
```

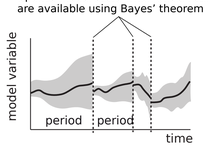
Solution scheme for data assimilation

for each period in periods:
for each n in Monte Carlo Samples:
for each t in period:

$$\mathbf{z}_t^{(n)} = f(\mathbf{z}_{t-1}^{(n)}, \mathbf{i}_t^{(n)}, \mathbf{p}^{(n)})$$

evaluate Bayes' theorem

Update model state when observations are available using Bayes' theorem



Solution framework (Python)

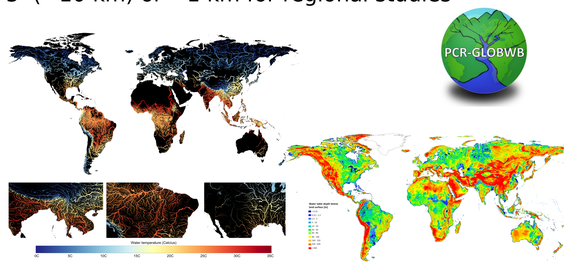
```
def suspend(self):                              # store model state at end of period
  self.report(self.snow, "s")
  ...
def updateWeight(self):                         # calculate weight of Monte Carlo sample required for
  sum = exp(maptotal(((obs - mod)**2)/          # solution of Bayes' equation and return to framework
       (2.0 * (observedStd ** 2))))
  weight = exp(sum)
  return weight
  ...
def resume(self):                               # read model state at start of next period
  self.read("s")
  ...
```

Application examples

Hydrological modelling

PCR-GLOBWB 2, a 5 arc-minute global hydrological & water resources model

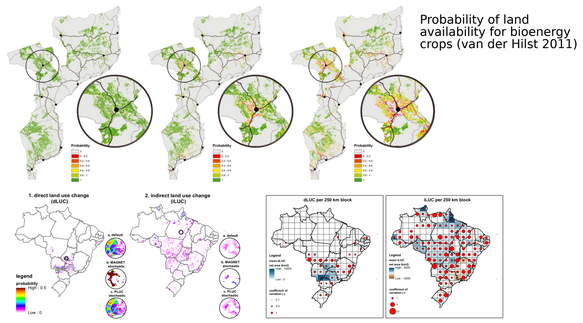
Daily time step, spatial resolution of 0.5° (~50 km), 5' (~10 km) or ~1 km for regional studies



Simulating water temperature (Wanders 2019) Simulating groundwater head (de Graaf 2017)

Land use change

Potential for bioenergy production with PLUC, the PCRaster Land Use Change model

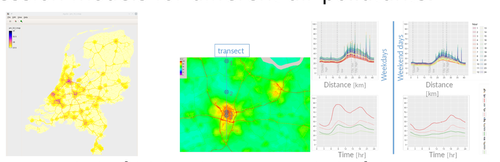


Probability of land availability for bioenergy crops (van der Hilst 2011)

Uncertainty estimates in direct or indirect land use change (Verstegen 2015)

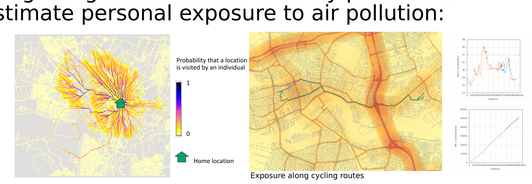
Environmental health

Calculation of high resolution (5x5 m) land use regression models for different air pollutants.



Average PM_{10} concentrations (µg/m^3) (Schmitz 2019) Hourly NO_2 concentrations (µg/m^3) along a transect in Utrecht

Integrating activities and mobility patterns to estimate personal exposure to air pollution:



Exposure along cycling routes

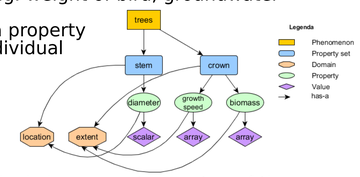
Research & Development

Our current work focusses on two topics, the development of one modelling environment for the construction of integrated field-based and agent-based models, and the development of operations making advantage of multi-core systems and compute clusters.

Integrating fields and agents

We develop a unifying conceptutal data model to store continuous fields and discrete agents:

Phenomenon: e.g. birds, groundwater
Property-set: Collection of properties sharing the same spatial and temporal domain
Domain: Information on space and time (e.g. location of birds, subsoil volume)
Property: Attribute (e.g. weight of bird, groundwater pollution level)
Value: Magnitude of a property
Item: Identifies an individual



Our physical data model is implemented on top of the HDF5 data model. A modelling framework allows for map algebra like operations both on fields or agents:

```
import numpy
from pcraster.framework import *
from pcraster.fame import *

class LueDynamic(DynamicModel):
  def __init__(self):
    DynamicModel.__init__(self)
    setclone(1,1,1,0,0)

    self.countries = Phenomenon()
    self.countries.catchments = PropertySet()

    # Read a set of spatial extents (clone maps)
    areas = Areas()
    areas.read('areas.csv')
    self.countries.catchments.domain = areas
    self.countries.catchments.clones = Property()

  def initial(self):
    self.countries.catchments.alives = uniform(self.countries.catchments.clones) < 0.15
    report(self.countries.catchments.alives, "output", self.currentTimeStep())

  def dynamic(self):
    numberOfAliveNeighbours = windowtotal(self.countries.catchments.alives, 15) -
        self.countries.catchments.alives
    threeAliveNeighbours = numberOfAliveNeighbours == 3
    birth = threeAliveNeighbours & !self.countries.catchments.alives
    survivalA = (numberOfAliveNeighbours == 2) & self.countries.catchments.alives
    survivalB = (numberOfAliveNeighbours == 3) & self.countries.catchments.alives
    survival = survivalA | survivalB
    self.countries.catchments.alives = birth | survival
    report(self.countries.catchments.alives, "output", self.currentTimeStep())

myModel = LueDynamic()
dynModel = DynamicFramework(myModel, 150)
dynModel.run()
```

Parallel and distributed computing

We enhance the PCRaster model building framework with built-in capabilities to run models on various hardware platforms, resulting in hardware scalable models that can be constructed by environmental modellers.

PCRaster on shared memory systems

A binding between PCRaster and Fern provides about 50 parallel local and focal operations. Fern is a highly generic C++ software library for raster processing that can be tailored to the configuration of a modelling framework.

PCRaster on distributed memory systems

Algorithms that operate on an irregular topology, such as material transport over a local drainage network, require a decomposition into fine grained sets of concurrent tasks for efficient execution. These tasks will be connected with other tasks from multiple algorithms into a task-graph, and an external HPX runtime library executes all tasks both on shared and distributed memory systems.

Sustainability challenges

Sustaining the development

- Acquiring funding is major issue, software development is often a minuscule part of research proposals
- Providing tailored consultancy or maintenance contracts
- Full time software development not always guaranteed

Maintaining our user base

- High initial effort to create documentation or courses
- Unpredictable moments and amount of user support
- Open source is not automatically the silver bullet: reported issues yes, but no pull requests so far

Maintaining the code base

- Version control and unit testing less familiar to environmental scientists
- Aligning with 3rd party dependencies (Qt, Boost, Gdal, ...)
- Packaging or building for different operating systems
- Dealing with legacy code, oldest C code from early 1990s
- Programming languages advance: C11, C++20; Python 2, Python 3
- Support of new hardware (e.g. GPU) requires entirely new code

Attribution of research software

- Traditional by publications, but these were often on environmental research topics
- Nowadays changing: DOI assignment, 'Software availability' sections, RSE communities, Computer Science oriented journals
- Still, software development is not part of a measurement, e.g. the h-index

Additional information

http://www.pcraster.eu/

https://github.com/pcraster/pcraster

https://github.com/pcraster/lue

de Bakker, de Jong, Schmitz, Karssenberg, Design and demonstration of a data model to integrate agent-based and field-based modelling. Environmental Modelling & Software (2019), https://doi.org/10.1016/j.env-soft.2016.11.016

de Graaf, van Beek, Gleeson, Moosdorf, Schmitz, Sutanudjaja, Bierkens, A Global-Scale Two-Layer Transient Groundwater Model: Development and Application to Groundwater Depletion, Advances in Water Resources (2017), https://doi.org/10.1016/j.advwatres.2017.01.011

van der Hilst, Verstegen, Karssenberg, Faaij, Spatiotemporal land use modelling to assess land availability for energy crops – illustrated for Mozambique, GCB Bioenergy (2011), https://doi.org/10.1111/j.1757-1707.2011.01147.x

Sutanudjaja, van Beek, Wanders, Wada, Bosmans, Drost, van der Ent, de Graaf, Hoch, de Jong, Karssenberg, Lopez Lopez, Pessenteiner, Schmitz, Straatsma, Vannametee, Wisser, Bierkens, PCR-GLOBWB–2: a 5 arcmin global hydrological and water resources model, Geoscientific Model Development (2018), https://doi.org/10.5194/gmd-11-2429-2018

Schmitz, Beelen, Strak, Hoek, Soenario, Brunekreef, Vaartjes, Dijst, Grobbee, Karssenberg, High resolution annual average air pollution concentration maps for the Netherlands, Scientific Data (2019) https://doi.org/10.1016/10.1038/sdata.2019.35

Verstegen, van der Hilst, Woltjer, Karssenberg, de Jong, Faaij, What can and can't we say about indirect land-use change in Brazil using an integrated economic – land-use change model?, GCB Bioenergy (2015), https://doi.org/10.1111/gcbb.12270

Wanders, van Vliet, Wada, Bierkens, van Beek, High-resolution global water temperature modelling, Water Resources Research (2019), https://doi.org/10.1029/2018WR023250

Our website | Our source code | Publications on applications | Publications on development | Download this poster

Photography encouraged