Re-engineering and optimization of GEOtop 3.0 integrated hydrological model

Elisa Bortoli[1] Giacomo Bertoldi[1] Alberto Sartori[2] Stefano Cozzini[3]

[1]EURAC Research, Institute for Alpine Environment, Bolzano, Italy

[2]SISSA, Trieste, Italy

[3]CNR-IOM and Exact-lab Trieste, Italy

EGU General Assembly, Vienna, 7-12 April 2019

Outline

Motivation

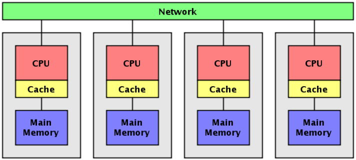
Scientific softwares

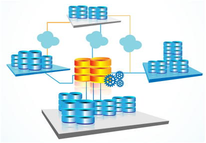
- solve complex scientific problems → IHMs
  - solve states (SWC) and fluxes (ET) in terrestrial compartments
  - computationally expensive
    - large domains → spatial high-resolution
    - long simulation periods → climatic simulations
- issue: low emphasis on code quality [1]

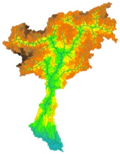
Challenges

- changes in computer architectures
- more data availability

Productivity → Need of software refactoring!





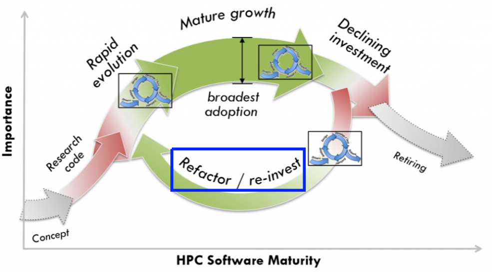


[1] Heaton et al. (2015)

Aims

GEOtop model [2]: 20 years of development

- scientific and applied problems
- increased complexity [3]



GOAL: software reengineering and refactoring

- robust and stable
- easily usable for operational applications
- optimized for modern architectures

[2] Rigon et al. (2006) [3] https://ideas-productivity.org/ideas-classic/how-to/

GEOtop model

The GEOtop model simulates

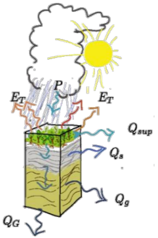
- water flow in the soil $\rightarrow$ Richards' eq (sub) + Kinematic eq (sur)
- energy exchange with the atmosphere $\rightarrow$ full integration of equation

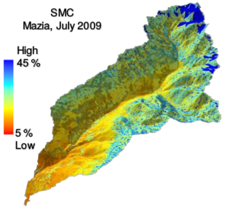
Water and energy budgets can be activated

- one or the other $\rightarrow$ simplification
- both them together $\rightarrow$ realistic

Two setup configurations

- 1D: only vertical fluxes $\rightarrow$ balances at local scale
- 3D: vertical and lateral fluxes $\rightarrow$ balances at basin scale



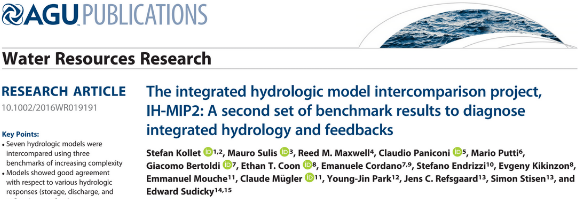


GEOtop packages

The core components of the package were presented in the **2.0 version** [4]

- written in C and released in 2014 as free open-source project
- scientifically tested and published [5]
- documented on GitHub repository

`http://geotopmodel.github.io/geotop/`



AGU PUBLICATIONS

Water Resources Research

RESEARCH ARTICLE

10.1002/2016WR019191

Key Points:

- Seven hydrologic models were intercompared using three benchmarks of increasing complexity
- Models showed good agreement with respect to various hydrologic responses (storage, discharge, and

The integrated hydrologic model intercomparison project, IH-MIP2: A second set of benchmark results to diagnose integrated hydrology and feedbacks

Stefan Kollet[1,2], Mauro Sulis[3], Reed M. Maxwell[4], Claudio Paniconi[5], Mario Putti[6], Giacomo Bertoldi[7], Ethan T. Coon[8], Emanuele Cordano[7,9], Stefano Endrizzi[10], Evgeny Kikinzon[8], Emmanuel Mouche[11], Claude Mügler[11], Young-Jin Park[12], Jens C. Refsgaard[13], Simon Stisen[13], and Edward Sudicky[14,15]

Scientific quality of the project but still missing
a **modern software engineering approach!**

[4] Endrizzi et al. (2014) [5] Kollet et al. (2016)

GEOtop 2.0: issues

Data Structures → **Definition**

- Code repetitions → time consuming maintenance
- Pointers of pointers → difficult debug

```c
typedef struct {

        short isdynamic;

        const char * name;

        long nrl,nrh,ncl,nch;

        int **co;

} INTMATRIX;

typedef struct {

        short isdynamic;

        const char * name;

        long nrl,nrh,ncl,nch;

        double **co;

} DOUBLEMATRIX;
```

```c
#define NL 1

/*-----------------------------------------------------------------

DOUBLEMATRIX *new_doublematrix0_(long nrh,long nch)
{
  DOUBLEMATRIX *m=(DOUBLEMATRIX *)malloc(sizeof(DOUBLEMATRIX));

  if (!m) t_error("allocation failure in new_doublematrix()");
  m->isdynamic=isDynamic;

  m->nrl=0;
  m->nrh=nrh;

  m->ncl=NL;
  m->nch=nch;

  m->co=dmatrix(m->nrl,m->nrh,m->ncl,m->nch);

  return m;
}
```

Data Structures → **Allocation**

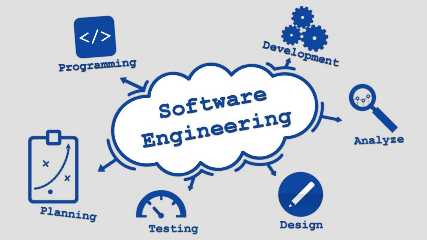
- Lower bound definition → confusing and error-prone
- Allocation functions → not easily understandable

GEOtop 3.0: requirements

New version needed $\rightarrow$ **GEOtop 3.0**

Software engineering needs

- scientific validated
- collaboratively developed
- easy to document with track changes
- modular and flexible
- extensively tested
- computationally efficient



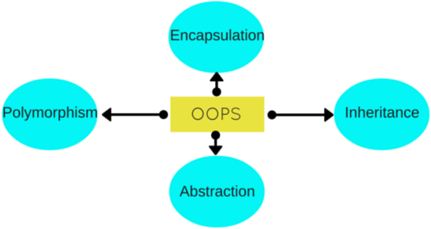
Software productivity tools

- pre/post processing for I/O preparation and visualization
- sensitivity analysis and calibration

GEOtop 3.0: modularity and flexibility

C++ programming language:

- object-oriented approach (OOPS) [6]
- presence of *templates*
- simplicity of code translation



Objectives

- uniform interface for data structure $\rightarrow$ understanding + optimization
- code reusage $\rightarrow$ maintainance
- memory management $\rightarrow$ avoiding memory leaks

[6] Stroustrup (2013)

GEOtop 3.0: modularity and flexibility

NEW Data Structures → **Definition + Allocation**

```
template <class T> class Matrix {

public:
    /** lower and upper bounds */
    std::size_t nrh, nrl; // rows
    std::size_t nch, ncl; // columns

    std::size_t n_row, n_col;

    /** the actual data */
    std::unique_ptr<T[]> co;

    /** destructor. default is fine */
    ~Matrix() = default;

    /** default constructor is deleted */
    Matrix() = delete;

    /** constructor */
    Matrix(const std::size_t _nrh, const std::size_t _nrl,
           const std::size_t _nch, const std::size_t _ncl):
           nrh{_nrh}, nrl{_nrl}, nch{_nch}, ncl{_ncl},
           n_row{nrh-nrl+1}, n_col{nch-ncl+1},
           co { new T[n_row*n_col]{} } {}

    Matrix(const std::size_t r, const std::size_t c):
           Matrix{r,1,c,1} {}
```

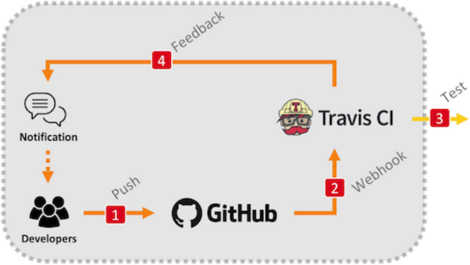
Testing: framework

To improve code reliability:

- test correctness of new code → unit tests with **google test** [7]
- access valid elements indexes → bound check with **macros**
- same results of 2.0 → continuous integration with **TravisCI** [8]

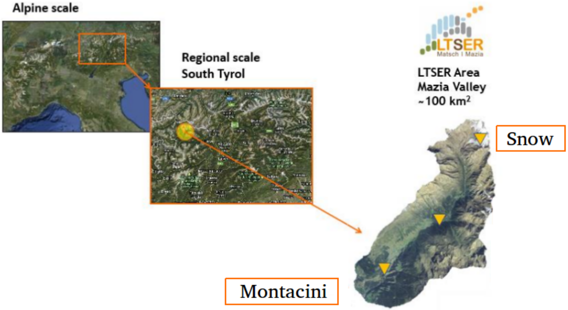
```
TEST(Matrix, constructor_2args){
  Matrix<int> m{3,5}; // 3x5 matrix
  EXPECT_EQ( std::size_t{3}, m.n_row );
  EXPECT_EQ( std::size_t{5}, m.n_col );
}
```

```
/** range-checked access operator */
T &at(const std::size_t i, const std::size_t j) {
  GEO_ERROR_IN_RANGE(i, nrl, nrh);
  GEO_ERROR_IN_RANGE(j, ncl, nch);
  return (*this)[(i-nrl)*n_col+(j-ncl)];
}
```



[7] https://github.com/google/googletest [8] https://travis-ci.org/

Testing: examples



Test case	Area $[km^2]$	Resolution $[m]$	Cells	Stations	Time
1D_WE	[-]	[-]	1	1	5 years
3D_E	62	100	10 k	7	1 month
3D_WE	2.5	20	17 k	4	1 week

Testing: hardware architectures

Local pc $\rightarrow$ profiling and testing

- Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz
- 1 socket, 4 cores/socket, 2 threads/core
- Memory: 6 MB Cache, 16 GB RAM

VSC-3 [9] $\rightarrow$ testing

- Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz
- 2 sockets, 8 cores/socket, 2 threads/core
- Memory: 20 MB Cache, 128 GB RAM

[9] http://www.vsc.ac.at/systems/vsc-3/

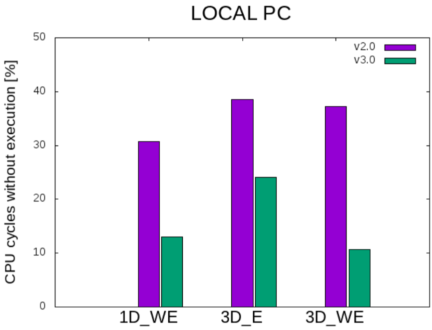
Testing: profiling

Profilers:

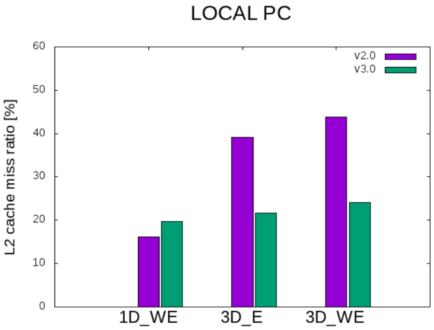
(1) **Likwid-perfctr** [10]: CPU cycles without execution + L2 cache misses

(2) **Callgrind** [11]: CPU cycles in each function

(3) **Class** `Timer<T>`: function calls + CPU time



-30%



-20%

[10] https://github.com/RRZE-HPC/likwid

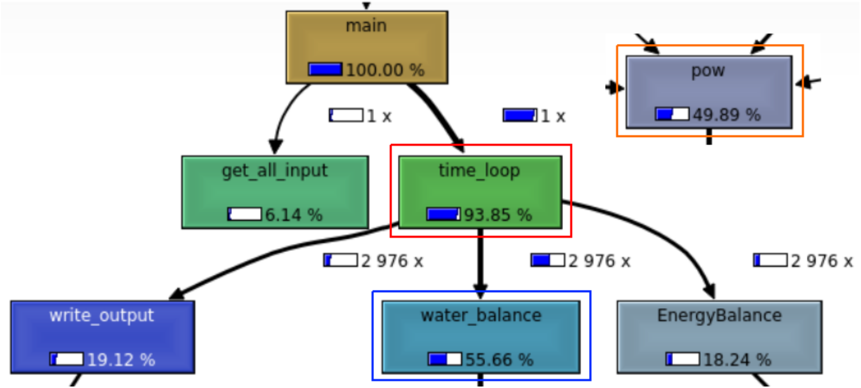
[11] http://valgrind.org

Testing: profiling

CPU time: most expensive functions

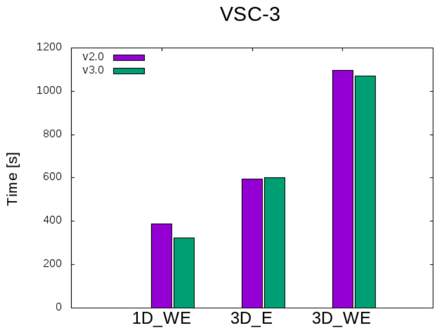
- water balance → B2 and Montacini (35% and 73%)
- input reading → snow (42%)

CPU cycles: `pow()` → B2 and Montacini (50% and 31%)



Optimization strategy

Check: run time comparison between GEOtop 3.0 and to 2.0



Optimization

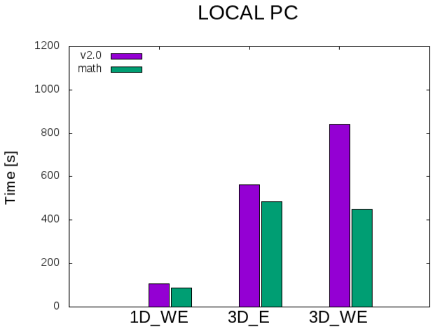
- Maths optimization
- OpenMP parallelization
- integration with MeteoIO library

Math optimizations

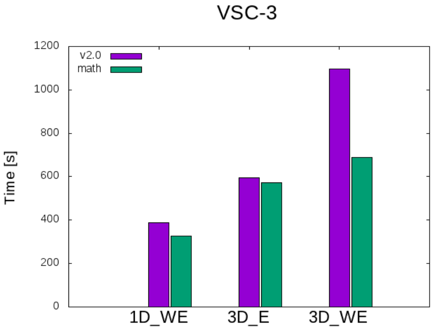
The function `pow()`: very much used

- `pow(a,2)` $\rightarrow$ `#define pow_2(a)((a)(a))`
- applied a property of logarithms: $a^b = e^{b*log(a)}$

Results: CPU time decrease for all test cases



-50%



-40%

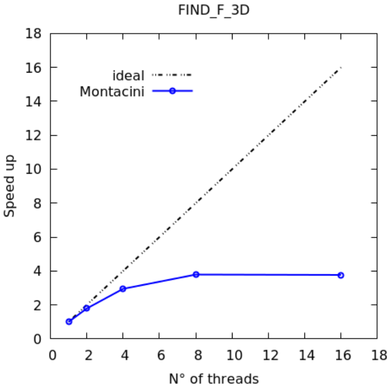
OpenMP parallelization

Parallelization of expensive functions:

- processes $\rightarrow$ input reading (snow) + water balance (B2, Montacini)
- used the same data $\rightarrow$ OpenMP (shared memory)

Results:

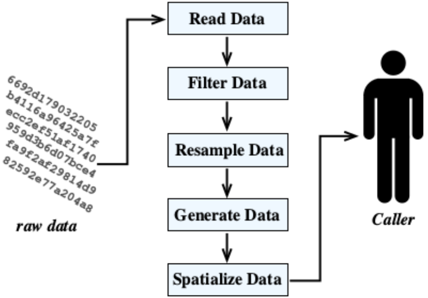
- water balance: sub-linear speed up $\rightarrow$ threads competing for cores
- input reading: no scaling $\rightarrow$ need better I/O!



MeteoIO library

C++ library to make data access easy and safe for simulations [12]

- **uniform interface** to meteo data in the model
- **robust I/O**, unobtrusive and simple for the user
- **filtering**, resampling, spatial interpolation



Simplified view of the MeteoIO dataflow.

[12] Bavay et al. (2014)

Conclusions and Outlook

GEOtop 3.0

Software engineering practices

- collaboratively developed $\rightarrow$ community-based
- easy to document with track changes $\rightarrow$ git versioning system [13]
- modular and flexible $\rightarrow$ object oriented approach
- extensively tested $\rightarrow$ unit + integration tests
- computationally efficient $\rightarrow$ Maths + OpenMP

[13] https://git-scm.com/

Conclusions and Outlook

Learned lessons

- importance of refactoring before optimization
- optimization results depend on the type of test case
  - 1D vs 3D
  - water vs energy

To do

- Maths optimization $\rightarrow$ use libraries (BLAS [14], Eigen [15], ...)
- OpenMP parallelization $\rightarrow$ computationally expensive functions
- MeteoIO library $\rightarrow$ data filtering + interpolation

THANKS FOR YOUR TIME!

Acknowledgment

The research reported in this work was supported under HPC-TRES program award number 2017-20 by OGS, CINECA and EURAC Research, and by the project DPS4ESLAB (Data Platform and Sensing Technologies for Environmental Sensing LAB), financed by the EU program: FESR 2014-2020: Asse 1 Ricerca e Innovazione 3 bando.

The computational results presented have been achieved [in part] using the Vienna Scientific Cluster (VSC).

For the test cases, data from the Long Term Ecological Research Area Mazia Valley (South Tyrol, Italy) have been used.

Siegfried Höfinger, Samuel Senorer, Christian Brida and Emanuele Cordano are acknowledged for their technical support.

References I

[1] D. Heaton, Dustin and J.C. Carver
Claims about the use of software engineering practices in science: A systematic literature review
Information and Software Technology 67, 207219, https://doi.org/10.1016/j.infsof.2015.07.011, 2015.

[2] R. Rigon, G. Bertoldi, and T.M. Over
GEOtop: a distributed hydrological model with coupled water and energy budgets
J. Hydrometeorol. 7 (3),371388, https://doi.org/10.1175/JHM497.1, 2006.

[3] https://ideas-productivity.org/ideas-classic/how-to/

References II

[4] S. Endrizzi, S. Gruber, M. Dall'Amico, and R. Rigon
GEOtop 2.0: simulating the combined energy and water balance at and below the land surface accounting for soil freezing, snow cover and terrain effects
Geosci. Model Dev., 7,2831-2857,
https://doi.org/10.5194/gmd-7-2831-2014, 2014.

[5] S. Kollet, M. Sulis, R. M. Maxwell, C. Paniconi, M. Putti, G. Bertoldi, E. T. Coon, E. Cordano, S. Endrizzi, E. Kikinzon, E. Mouche, C. Mugler, Y. Park, J. C. Refsgaard, S. Stisen, E. Sudicky
The integrated hydrologic model intercomparison project, IH-MIP2: A second set of benchmark results to diagnose integrated hydrology and feedbacks
Water Resour. Res., 53, 867890,
https://doi.org/10.1002/2016WR019191, 2017.

References III

[6] B. Stroustrup(2013)
The C++ Programming Language : Fourth Edition.
ISBN : 0321154916. URL :
https://books.google.de/books?id=LgmqCAAAQBAJ.

[7] https://github.com/google/googletest

[8] https://travis-ci.org/

[9] http://www.vsc.ac.at/systems/vsc-3/

[10] https://github.com/RRZE-HPC/likwid

[11] http://valgrind.org/docs/manual/cl-manual.html

[12] M. Bavay, T. Egger
MeteoIO 2.4.2: a preprocessing library for meteorological data
Geosci. Model Dev., 7,3135-3151,
https://doi:10.5194/gmd-7-3135-2014, 2014.

[13] https://git-scm.com/

[14] http://www.netlib.org/blas/

[15] http://eigen.tuxfamily.org/index.php?title=Main_Page