Large Scale Simulation of Flow and Transport in Porous Media

Olaf Ippisch

Institut de Mathematiques
École Polytechnique de Clausthal

Séminaire de la Modélisation
13 Février 2018
Why do we need large area simulations?

Weather Forecasting

Source: Tagesschau, ARD

Climate Research

Source: Robert A. Rohde / Global Warming Art

Water Management

Source: V. M. Ponce: Sustainable Yield of Ground Water
Example of a Landscape simulated with 1 to $5 \times 5$ Grid Cells
Challenge: Multi-scale Nature of Geosystems

Process Description

Experiments

Applications
Reasonable Spatial Resolution

1 m Resolution

1000 m Resolution

10 m Resolution

100 m Resolution
At 10 m resolution individual fields are still resolved.
  - Soil parameters can be measured or deduced from pedotransfer functions and soil maps.
  - Crop specific parameters can be applied for evapotranspiration.

Already at 100 m resolution averaging over different land use and soil types is necessary.
  - Processes are strongly non-linear.
    ⇒ Processes can act on very different time scales.
  - Above a certain horizontal distance processes are very weakly coupled.
    ⇒ Averaging is difficult and will lead to non-unique and time-dependent parameters.
Reasonable Spatial Resolution

- At 10 m resolution individual fields are still resolved.
  - Soil parameters can be measured or deduced from pedotransfer functions and soil maps.
  - Crop specific parameters can be applied for evapotranpiration.

- Already at 100 m resolution averaging over different land use and soil types is necessary.
  - Processes are strongly non-linear.
    ⇒ Processes can act on very different time scales.
  - Above a certain horizontal distance processes are very weakly coupled.
    ⇒ Averaging is difficult and will lead to non-unique and time-dependent parameters.

⇒ A resolution of at least 10 m is desirable
Coupling to models of the atmosphere

- At the moment land-surface models and models of the atmosphere are often run at the same spatial resolution.
- For the simulation of processes in the atmosphere a high temporal resolution is needed, while acceptable results are obtained with rather coarse spatial resolutions.
- For the soil it is vice versa. Processes are slow but heterogeneity is high and complicated.
Coupling to models of the atmosphere

- At the moment land-surface models and models of the atmosphere are often run at the same spatial resolution.
- For the simulation of processes in the atmosphere a high temporal resolution is needed, while acceptable results are obtained with rather coarse spatial resolutions.
- For the soil it is vice versa. Processes are slow but heterogeneity is high and complicated.

⇒ Use larger time-steps and finer spatial resolution for soil, coupling by averaged fluxes.
Large scale simulation

Huge number of grid cells

Assumptions:

- horizontal element width 10 m
- 100 grid cells vertically

<table>
<thead>
<tr>
<th>Region</th>
<th>Area km²</th>
<th># Grid Cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>Île de France</td>
<td>12'012</td>
<td>1.2 \cdot 10^{10}</td>
</tr>
<tr>
<td>Germany</td>
<td>357'167</td>
<td>3.6 \cdot 10^{11}</td>
</tr>
<tr>
<td>France</td>
<td>632'834</td>
<td>6.3 \cdot 10^{11}</td>
</tr>
<tr>
<td>European Union</td>
<td>4’381’324</td>
<td>4.4 \cdot 10^{12}</td>
</tr>
</tbody>
</table>
Large scale simulation
Huge number of grid cells

Assumptions:

- horizontal element width 10 m
- 100 grid cells vertically

<table>
<thead>
<tr>
<th>Region</th>
<th>Area km²</th>
<th># Grid Cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>Île de France</td>
<td>12’012</td>
<td>1.2 · 10^{10}</td>
</tr>
<tr>
<td>Germany</td>
<td>357’167</td>
<td>3.6 · 10^{11}</td>
</tr>
<tr>
<td>France</td>
<td>632’834</td>
<td>6.3 · 10^{11}</td>
</tr>
<tr>
<td>European Union</td>
<td>4’381’324</td>
<td>4.4 · 10^{12}</td>
</tr>
</tbody>
</table>

We could simulate the area of the whole European Union with a resolution of 10 m with 4.4 · 10^{12} grid cells.
Large scale simulation
Massively Parallel Simulation of Flow and Transport with Supercomputers

Source: Forschungszentrum Jülich
How close can we get to $10^{12}$ grid cells today?
Richards’ equation

\[
\frac{\partial \theta}{\partial t} - \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \left( \frac{\partial p_l}{\partial x} - \rho_l g \cos(\gamma) \right) \right] + q = 0
\]

- Describes water transport away from full saturation
- Non-linear convection diffusion equation
Richards’ equation

\[
\begin{align*}
\frac{\partial \theta}{\partial t} - \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \left( \frac{\partial p_l}{\partial x} - \rho_l g \cos(\gamma) \right) \right] + q &= 0 \\
\frac{\partial \theta}{\partial p_l} \frac{\partial p_l}{\partial t} - K \frac{k_r(\theta)}{\mu_l} \frac{\partial^2 p_l}{\partial x^2} - \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \right] \cdot \left( \frac{\partial p_l}{\partial x} \right) + \rho_l g \cos(\gamma) \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \right] + q &= 0
\end{align*}
\]

- Describes water transport away from full saturation
- Non-linear convection diffusion equation
Richards’ equation

\[
\frac{\partial \theta}{\partial t} - \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \left( \frac{\partial p_l}{\partial x} - \rho_l g \cos(\gamma) \right) \right] + q = 0
\]

\[
\frac{\partial \theta}{\partial t} \frac{\partial p_l}{\partial t} - K \frac{k_r(\theta)}{\mu_l} \frac{\partial^2 p_l}{\partial x^2} - \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \right] \cdot \left( \frac{\partial p_l}{\partial x} \right) + \rho_l g \cos(\gamma) \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \right] + q = 0
\]

\[
\frac{\partial \theta}{\partial t} \frac{\partial p_l}{\partial t} - K \frac{k_r(\theta)}{\mu_l} \frac{\partial^2 p_l}{\partial x^2} - \left\{ \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \right] - \rho_l g \cos(\gamma) \frac{\partial}{\partial p_l} \left[ K \frac{k_r(\theta)}{\mu_l} \right] \right\} \frac{\partial p_l}{\partial x} + q = 0
\]

- Describes water transport away from full saturation
- Non-linear convection diffusion equation
- Degenerate parabolic equation
  \( \Rightarrow \) Can get elliptic, parabolic and (effectively) hyperbolic simultaneously at different locations (depending on locally varying coefficients)
Richards’ equation

\[
\frac{\partial \theta}{\partial t} - \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \left( \frac{\partial p_l}{\partial x} - \rho_l g \cos(\gamma) \right) \right] + q = 0
\]

\[
\frac{\partial \theta}{\partial p_l} \frac{\partial p_l}{\partial t} - K \frac{k_r(\theta)}{\mu_l} \frac{\partial^2 p_l}{\partial x^2} - \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \right] \cdot \left( \frac{\partial p_l}{\partial x} \right) + \rho_l g \cos(\gamma) \frac{\partial}{\partial x} \left[ K \frac{k_r(\theta)}{\mu_l} \right] + q = 0
\]

- Describes water transport away from full saturation
- Non-linear convection diffusion equation
- Degenerate parabolic equation
  ⇒ Can get elliptic, parabolic and (effectively) hyperbolic simultaneously at different locations (depending on locally varying coefficients)
Numerical approach of $\mu \varphi$ (muPhi)

Central aim: maximal stability and robustness, local mass conservation

- Method of lines approach:
  - Cell-centred finite volume scheme for spatial discretisation with upwinding of capillary pressure and harmonic averaging to calculate conductivities
  - Tensor-product grid
  - Implicit Euler scheme for time discretisation

- Newton’s method for linearisation of the non-linear equations with
  - Line search
  - Inexact solution of the linear equation system

- BiCGStab and parallel algebraic multi-grid preconditioner from Dune-ISTL for linear equation system

- Parallelisation approach
  - Domain decomposition
  - One element overlap in every direction
    $\Rightarrow$ local assembly of Jacobian possible
  - MPI-based communication with maximal $2d$ neighbours
Weak scalability test for $\mu \varphi$

- $h = 0.01 \text{ m} \Rightarrow$ horizontal domain size 0.64 m to 347.52 m
- Miller similarity scaling: $K \propto s^2$, $\theta \propto 1/s$
- $s$ taken from auto-correlated random field, correlation length 1 m horizontally and 0.2 m vertically
- Reference van Genuchten Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$\theta_s$</th>
<th>$\theta_r$</th>
<th>$K_s$</th>
<th>$n$</th>
<th>$\alpha$</th>
<th>$\tau$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>0.34</td>
<td>0.0</td>
<td>40.0 cm/h</td>
<td>2.0</td>
<td>5.0</td>
<td>0.5</td>
</tr>
</tbody>
</table>

- Constant flux of 2 mm/h at top, $p_c = 0$ at bottom, no-flux at sides
- One time step of one hour, $10^{-5}$ reduction in non-linear defect required

O. Ippisch: Contributions to the large-scale Simulation of Flow and Transport in Porous Media, Habilitationsschrift, Universität Heidelberg
Weak scalability test for $\mu \varphi$

- BlueGene/P JUGENE (4 cores per node) and BlueGene/Q JUQUEEN (16 cores per node, 2 hyper-threads per core) at Jülich Supercomputing Center
  - $P \cdot 64 \times 64 \times 128$ degrees of freedom
  - Bluegene/P: up to $34752 \times 34752 \times 128$ on 294,849 cores
    $\Rightarrow$ 154 billion unknowns
  - Bluegene/Q: up to $57344 \times 65536 \times 128$ on 917,504 threads
    $\Rightarrow$ 481 billion unknowns
Weak scalability test for $\mu \psi$

Results

**Block Scenario**
- 64 cores
- $512 \times 512 \times 128$ grid cells

**Large Scenario**
- 1024 cores
- $2048 \times 2048 \times 128$ grid cells
Weak scalability test for $\mu\varphi$

Bluegene/P: Scalability and Parallel Efficiency

![Graph showing Total Computation Time and Parallel Efficiency](image)

- **Total Computation Time**
  - Runtime (w.o. I/O)
  - Unsaturated Testcase

- **Parallel Efficiency**
  - Efficiency Unsaturated Testcase
  - defect homogeneous
  - defect block
  - defect large
  - matrix homogeneous
  - matrix block
  - matrix large
  - coarsening homogeneous
  - coarsening block
  - coarsening large
  - solver homogeneous
  - solver block
  - solver large

O. Ippisch (IfM, TU Clausthal)
Weak scalability test for $\mu \varphi$

Bluegene/P: Scalability and Parallel Efficiency

Total Computation Time

Runtime (w.o. I/O)

Unsaturated Testcase

Parallel Efficiency

Efficiency Unsaturated Testcase

⇒ Scalable solution up to 294’849 cores and $1.5 \cdot 10^{11}$ unknowns, similar results for the elliptic case
Weak scalability test for $\mu \varphi$ (only Block scenario)

Bluegene/Q: Scalability and Parallel Efficiency

**Total Computation Time**

Runtime (without I/O)

non-linear Parabolic Test Case

**Parallel Efficiency**

Efficiency non-linear Parabolic Test Case
Weak scalability test for $\mu \varphi$ (only Block scenario )

Bluegene/Q: Scalability and Parallel Efficiency

⇒ Scalable solution up to 917'504 threads (458'752 cores) and $4.8 \cdot 10^{11}$ unknowns
Solute transport

\[ \frac{\partial (\theta c_s)}{\partial t} - \nabla \cdot [D(\theta) \nabla c_s - c_s j_l] + q_s = 0 \]

- Second-order Godunov scheme with minmod slope-limiter for convective terms
- Second-order cell-centred Finite Volume scheme in space and Explicit Euler scheme for time discretisation
Solute transport

\[
\frac{\partial (\theta c_s)}{\partial t} - \nabla \cdot [D(\theta) \nabla c_s - c_s j] + q_s = 0
\]

- Second-order Godunov scheme with minmod slope-limiter for convective terms
- Second-order cell-centred Finite Volume scheme in space and Explicit Euler scheme for time discretisation
File I/O

Setup

- Massive I/O demand:
  with 294’849 cores 618 GB input, 7.4 TB output

- JUQUEEN storage cluster
  - IBM General Parallel File System (GPFS), GPFS block size: 4 MB
  - Maximum usable band width measured (Wolfgang Frings, JSC):
    120 GB/sec (output) 65 GB/sec (input)

- Input with either
  - parallel HDF5: processes read data from 3D structure file
  - SIONlib (1 file sector block per core): data already pre-partitioned in chunks for each process

- Output with SIONlib
  - 1 file per I/O node
  - each process writes its data in continuous block
  - sequential post-processing on high-memory servers
File I/O
Results

⇒ Up to 47 GB/s on input and 81 GB/s on output.
File I/O

Results

⇒ Up to 47 GB/s on input and 81 GB/s on output.
260 s for output of solution with 21 TB on the largest grid.
Brute force solution of the problem is possible, but . . .

- With lower-order methods like Finite-Volumes, we can only exploit a fraction of the computation power due to memory bandwidth restrictions. These will get worse with next generation supercomputers.
Next Generation Supercomputers

There’s no chance that the iPhone is going to get any significant market share.

*Steve Ballmer, USA Today, April 30, 2007*
Exascale computers are expected to have

- More cores per CPU ($\geq 1000$)
- Wider SIMD units ($\geq 512$ bit)
- Less memory per core ($\leq 100$ MB/core)
- Reduced memory bandwidth per core
- Some accelerator architecture (GPGPU, as Intel Phi will be discontinued)
This development can already be seen today in the world’s currently fastest supercomputers

- Sunway TaihuLight (93.0 PFlop/s, 40’960 × 260 core CPUs, 126 MB/core)
- Tianhe-2 (33.9 PFlop/s, 16’000 × 2 × 12 core CPUs, 2.7 GB/core, Intel Phi)
- Piz Daint, Swiss National Supercomputing Centre (19.6 PFlop/s, 5’272 × 12 core CPUs, 2.7 GB/core, NVIDIA GPGPU)
Brute force solution of the problem is possible, but . . .

- With lower-order methods like Finite-Volumes, we can only exploit a fraction of the computation power due to memory bandwidth restrictions. These will get worse with next generation supercomputers.

- The kinematic wave approximation, the most common model for surface run-off, produces non-physical results.
How to model surface runoff?

1. Shallow-water equations

\[ \frac{1}{g} \frac{\partial v}{\partial t} + \frac{v}{g} \frac{\partial v}{\partial x} + \frac{\partial h}{\partial x} + (S_f - S_o) = 0 \]
How to model surface runoff?

1. Shallow-water equations

\[ \frac{1}{g} \frac{\partial v}{\partial t} + \frac{v}{g} \frac{\partial v}{\partial x} + \frac{\partial h}{\partial x} + (S_f - S_o) = 0 \]

2. Diffusive wave approximation (III + IV)

3. Kinematic wave approximation (IV)

comparative cost

I  II  III  IV
How to model surface runoff?

1. Shallow-water equations

2. Diffusive wave approximation (III + IV)

3. Kinematic wave approximation (IV)

\[
\frac{1}{g} \frac{\partial v}{\partial t} + \frac{v}{g} \frac{\partial v}{\partial x} + \frac{\partial h}{\partial x} + (S_f - S_o) = 0
\]
How to model surface runoff?

1. Shallow-water equations
2. Diffusive wave approximation (III + IV)
3. Kinematic wave approximation (IV)

\[
\frac{1}{g} \frac{\partial v}{\partial t} + \frac{v}{g} \frac{\partial v}{\partial x} + \frac{\partial h}{\partial x} + (S_f - S_o) = 0
\]

I II III IV
How to model surface runoff?

1. Shallow-water equations
2. Diffusive wave approximation (III + IV)
3. Kinematic wave approximation (IV)

\[
\frac{1}{g} \frac{\partial v}{\partial t} + \frac{v}{g} \frac{\partial v}{\partial x} + \frac{\partial h}{\partial x} + (S_f - S_o) = 0
\]
How to model surface runoff?

1. Shallow-water equations

$$\frac{1}{g} \frac{\partial v}{\partial t} + \frac{v}{g} \frac{\partial v}{\partial x} + \frac{\partial h}{\partial x} + (S_f - S_o) = 0$$

2. Diffusive wave approximation (III + IV)

3. Kinematic wave approximation (IV)
Brute force solution of the problem is possible, but . . .

- With lower-order methods like Finite-Volumes, we can only exploit a fraction of the computation power due to memory bandwidth restrictions. These will get worse with next generation supercomputers.

- The kinematic wave approximation, the most common model for surface run-off produces non-physical results.

- Just to resolve structural soil units, structured grids need much more degrees of freedom than necessary for accuracy.

- Steep infiltration and evaporation fronts can occur in porous media as well as horizontal flow inside soils, but usually only for short times at certain positions.
Brute force solution of the problem is possible, but . . .

- With lower-order methods like Finite-Volumes, we can only exploit a fraction of the computation power due to memory bandwidth restrictions. These will get worse with next generation supercomputers.

- The kinematic wave approximation, the most common model for surface run-off produces non-physical results.

- Just to resolve structural soil units, structured grids need much more degrees of freedom than necessary for accuracy.

- Steep infiltration and evaporation fronts can occur in porous media as well as horizontal flow inside soils, but usually only for short times at certain positions.

We need a next generation model
Next Generation Land Surface Model: General Approach

- Subsurface flow: Richards’ equation
- Surface runoff: Diffusive wave approximation
- 2.5D unstructured grid
- Adaptive grid refinement
  ⇒ Higher-order Discontinuous Galerkin scheme
Next Generation Land Surface Model: General Approach

- Subsurface flow: Richards’ equation
- Surface runoff: Diffusive wave approximation
- 2.5D unstructured grid
- Adaptive grid refinement
- \( \Rightarrow \) Higher-order Discontinuous Galerkin scheme
Subsurface Flow: Richards’ Equation
Numerical Approach

\[ \partial_t[\theta_w(\psi_m)] - \nabla \cdot (K(\theta_w(\psi_w))(\nabla \psi_w - g e_z)) = 0 \]

**Discretization**

- SIPG\(^2\) DG-scheme,
- Weights depending on saturation
- Conductivity calculation with upwind value of potential
- Diagonally Implicit Runge-Kutta (DIRK) method in time

---

1 by Ole Klein, Interdisciplinary Center for Scientific Computing (IWR), Heidelberg University
Surface Runoff: Diffusive Wave Approximation

Numerical Approach

\[ \partial_t h - \nabla \cdot \left[ \frac{(h - h_0)^\alpha}{C \cdot \| \nabla (h + z) \|^{1-\gamma}} \cdot \nabla (h + z) \right] = f \]

Surface Runoff: Diffusive Wave Approximation

Numerical Approach

\[ \partial_t h - \nabla \cdot \left[ \frac{(h - h_0)^\alpha}{C} \cdot \frac{\nabla(h + z)}{\|\nabla(h + z)\|^{1-\gamma}} \right] = f \]

- Calculation of scaled gradient crucial

\[ ^3 A. \text{Ern}, \ A. \ F. \ Stephansen \text{ and P. Zunino. } "\text{A discontinuous Galerkin method with weighted averages for advection–diffusion equations with locally small and anisotropic diffusivity.}\" \text{IMA Journal of Numerical Analysis 29.2 (2008): 235-256.} \]
Surface Runoff: Diffusive Wave Approximation

Numerical Approach

\[ \partial_t h - \nabla \cdot \left[ \frac{(h - h_0)^\alpha}{C} \cdot \frac{\nabla(h + z)}{||\nabla(h + z)||^{1-\gamma}} \right] = f \]

- Calculation of scaled gradient crucial
- SIPG\(^3\) DG-scheme in space
- Weighted average of gradients
- Upwind value of water height for calculation of conductivities
- Semi-implicit approach with DIRK scheme in time
- Automatic time step adaptation

In each time step

1. set $h^{t+1,0} = h^t$
2. calculate one time step of subsurface flow with step size $\tau$ and coupling through boundary condition
In each time step

1. set \( h_{t+1,0} = h^t \)
2. calculate one time step of subsurface flow with step size \( \tau \) and coupling through boundary condition

Signorini-Type Boundary Condition in step 2:

- From water height, evaporation and precipitation calculate maximal flux over soil boundary
- If flux is negative
  - calculate Dirichlet flux in subsurface with pressure head \( h_{t+1,k} \)
  - Use smaller one (into) soil
- If flux is positive
  - Calculate Dirichlet flux with given minimal potential \( \psi_{\text{lim}} \)
  - Use smaller one (out of) soil
- Store boundary fluxes \( j_{t+1,k}^{\text{Richards}} \)
Surface-Subsurface Coupling

- In each time step
  1. Set \( h^{t+1,0} = h^{t} \)
  2. Calculate one time step of subsurface flow with step size \( \tau \) and coupling through boundary condition
  3. Calculate \( n \) steps of surface flow with time step \( \tau/n \) and source term

\[
f^{t+1,k} = f_{\text{Richards}}^{t+1,k} - (f_{\text{evap}}^{t+1,k} + f_{\text{prec}}^{t+1,k})
\]

Signorini-Type Boundary Condition in step 2:

- From water height, evaporation and precipitation calculate maximal flux over soil boundary
- If flux is negative
  - Calculate Dirichlet flux in subsurface with pressure head \( h^{t+1,k} \)
  - Use smaller one (into) soil
- If flux is positive
  - Calculate Dirichlet flux with given minimal potential \( \psi_{\text{lim}} \)
  - Use smaller one (out of) soil
- Store boundary fluxes \( j^{t+1,k}_{\text{Richards}} \)
Surface-Subsurface Coupling

In each time step

1. set \( h^{t+1,0} = h^t \)
2. calculate one time step of subsurface flow with step size \( \tau \) and coupling through boundary condition
3. calculate \( n \) steps of surface flow with time step \( \tau/n \) and source term

\[
f_{t+1,k}^{t+1,k} = j_{\text{Richards}}^{t+1,k} - (j_{\text{evap}}^{t+1} + j_{\text{prec}}^{t+1})
\]

4. if necessary set \( k = k + 1 \), go to 2.

Signorini-Type Boundary Condition in step 2:

- From water height, evaporation and precipitation calculate maximal flux over soil boundary
- If flux is negative
  - calculate Dirichlet flux in subsurface with pressure head \( h^{t+1,k} \)
  - \( \Rightarrow \) Use smaller one (into) soil
- If flux is positive
  - Calculate Dirichlet flux with given minimal potential \( \psi_{\text{lim}} \)
  - \( \Rightarrow \) Use smaller one (out of) soil
- Store boundary fluxes \( j_{\text{Richards}}^{t+1,k} \)
Properties of Coupling Approach

- No tracking of wetting-front boundary necessary
- No obviously non-physical solutions
- Mass conservative solution
- May introduce non-differentiability
**Implementation**

- Implemented using **DUNE-PDElab**
- Software framework for the easy and efficient solution of PDEs
- High flexibility: variety of grids, parallel computations, adaptivity
- Nice code exploiting features of C++14

![Dune](dune-project.org)
Test Case: Surface Runoff and Infiltration into Coarse Sand

**Initial Conditions:**
- Constant water table heights at surface \( \Gamma_t \)
- Static hydraulic equilibrium on \( \Omega \)

**Boundary Conditions:**
- Static hydraulic equilibrium on \( \Gamma_b \)
- Zero flux on \( \Gamma_r \) and \( \Gamma_l \) (including surface)
Infiltration

![Infiltration Diagram](image)

- Water content
- Matric potential

Time: 0 sec
Infiltration

Time = 0 sec

Time = 1 sec

Time = 5 sec

Time = 19 sec

Time = 174 sec

Time = 2000 sec
Subsurface Inflow from Downslope

Time = 1 sec

Time = 2 sec

Time = 40 sec

Time = 109 sec
Subsurface Inflow from Upslope

Time = 1 sec

Time = 2 sec

Time = 40 sec

Time = 109 sec
3-dimensional Infiltration in Parallel
Efficient Implementation of the approach with EXA-DUNE

- Joint project (2012-2019) in the SPPEXA Special Priority Program of DFG:
  - TU Clausthal (O. Ippisch)
  - TU Dortmund (S. Turek),
  - Heidelberg University (P. Bastian)
  - TU Kaiserslautern (O. Iliev)
  - University of Münster (C. Engwer, M. Ohlberger)
  - University of Stuttgart (D. Göddeke)

- Open-source reusable exa-scale PDE solver components
- Exploit modern hardware without sacrificing generality
- Multiscale methods and uncertainty quantification
- Increased resiliency, parallel I/O
- Application domain: Flow in porous media

http://dune-project.org/
EXA-DUNE: Main Goals

1. Increased Node-Level Performance
   - Main idea: FLOPS are nearly for free, memory access is expensive (in time and energy)
   - SIMD and accelerator support for linear algebra
   - Hybrid parallelism for linear algebra and matrix assembly
   - Increased arithmetic intensity by higher-order schemes and sum-factorisation

2. Additional Levels of parallelism
   - Multi-level Finite-Elements schemes
   - Multi-level Monte-Carlo schemes

3. Generic and efficient Parallel File I/O

4. Fault Tolerance/Resiliency

5. Next-generation land-surface model as prototypical application
High-performance implementation using code-generation approach

Facilitates use of sum-factorisation for higher-order schemes
High-performance implementation using code-generation approach

Facilitates use of sum-factorisation for higher-order schemes

Code-generator developed and implemented by Dominic Kempf in group of Peter Bastian at IWR, Heidelberg

Based on UFL language from FEniCS project

Dominic Kempf
In any FEM scheme we need to evaluate base functions at all quadrature points:

\[ \hat{u}(\xi_i) = \sum_{j \in J} x_j \hat{\phi}_j(\xi_i) \]

⇒ Exploit tensor product structure of the basis and the quadrature rule!

\[ \hat{u}(\xi_{i1} \ldots i_d) = \sum_{j_d \in J^{(d)}} \cdots \sum_{j_1 \in J^{(1)}} A^{(d)}_{i_d,j_d} \cdots A^{(1)}_{i_1,j_1} x_{j_1} \ldots j_d \]

\[ = \sum_{j_d \in J^{(d)}} A^{(d)}_{i_d,j_d} \cdots \sum_{j_2 \in J^{(2)}} A^{(2)}_{i_2,j_2} \sum_{j_1 \in J^{(1)}} A^{(1)}_{i_1,j_1} x_{j_1} \ldots j_d \]

\[ = y_{j_2} \ldots j_d y_{i_1} \ldots i_d \]

\[ A^{(k)}_{i,j} = \hat{\theta}_i(\xi_j^{(k)}) \]: Evaluations of the 1D Basis at the 1D quadrature points.
Generalizations:

- For $\partial_k u(\xi_{j_1...j_d})$, set $A_{i_{k,j_k}}^{(k)} := \theta'_{j_k}(\xi_{i_k})$

- Test function application works the same:
  - Input: $f^\phi(\xi_{j_1...j_d})$ at all quadrature points
  - Output: Residual contribution $f^\phi \hat{\phi}_{i_1...i_d}$
  - Matrices $\theta$ are transposed

- Evaluations on facets can be further optimized by reordering directions

- Geometric quantities can be expressed as FE functions

Advantages:

- Reduction of algorithmic complexity

- Tensor Operations allow lots of fused multiply adds (FMA)
The whole assembly process is split as follows:

- Calculate input quantities *at all quadrature points*:
  - FE functions
  - geometric quantities

- Quadrature Loop
  - Calculate contributions $f_\hat{\phi}$, $f_{\partial_k \hat{\phi}}$

- Apply test function to $f_\hat{\phi}$, $f_{\partial_k \hat{\phi}}$
User input

UFL Input File
Weak Forms and Function Spaces

Driver
Simulation Workflow

Form Compiler

LocalOperator
Integration kernels

ParameterClass
Parameter Functions

Simulation executable

CMake

optionally
Form compiler approach

Mathematical problem
PDE problem in residual formulation

UFL
Python formulation in domain specific language

preprocessed UFL
Apply preprocessing from
UFL with custom extensions

loo.py: intermediate representation
of the loop nest for the PDE kernel

transformations

C++ PDELab code
LocalOperator, driver and parameter class

Automated analysis and profiling
of compiled source

Dune High-level Backend
to grid, geometry and basis

Numerics
Basis structure
Polynomial Degree
Grid
...

Hardware
Multithreading
Memory bandwidth
SIMD units
...

Vectorization Backend
VC, VCL: C++ intrinsics wrapper

O. Ippisch (IfM, TU Clausthal)
Large Scale Simulation
Maison de la Simulation
Benchmark setup

Problem:
- Diffusion reaction problem with full permeability tensor
- SIPG DG discretization on a structured, axiparallel grid
- MPI-parallelism to saturate machine
- 100MB of DOFs per core

Measuring GFlops/s:
- Counting operations: Instrumented C++ floating point type
- Separate executables for time and FLOP measurements

Compiler: gcc 6.3
Hardware in use

- Intel Haswell
  - Intel Xeon Processor E5-2698 v3
  - 16 cores
  - 2.3 GHz, AVX-heavy: 1.9 GHz
  - Peak performance: 30.4 GFlops/s per core \(\Rightarrow\) 486.4 GFlops/s

- Intel Xeon Phi (Knights Landing)
  - Intel Xeon Phi(TM) CPU 7250
  - 68 cores
  - 1.4 GHz, AVX-heavy: 1.2 GHz
  - Peak performance: 38.4 GFlops/s per core \(\Rightarrow\) 2610 GFlops/s
  - Clustering modes off
  - MCDRAM in use as NUMA node

- **68c KNL@215W**
- **16c Haswell@135W**

![Graph showing performance comparison between 68c KNL@215W and 16c Haswell@135W across different polynomial degrees.](image-url)
Conclusions

- Simulation of large-scale problems with a resolution sufficiently high to give parameters a meaning, is coming close.
- Our approach for large-scale simulation of water and solute transport is highly scalable in $P$ (up to 917'504 threads) and $N$ (up to $4.8 \cdot 10^{11}$ unknowns)
- Efficient parallel I/O of the huge data sets possible
- Exascale computers will have a much more demanding internal structure (more SIMD, more cores, less memory, accelerator cards)
- Started development of Next Generation Land-Surface Model
- New numerical approach for coupled solution of Diffusive Wave approximation and Richards’ equation works
- Higher-order DG-scheme with sum-factorisation achieves extreme efficiency even with automatic code generation
Outlook

- Finish high-performance implementation
- Development of special unstructured parallel 2.5D grid
- Large scale simulations of real world problems
- Add one-dimensional elements for rivers and rivulets
- Include Plant model
Thank you for your attention.

Acknowledgements:

- from IfM, TU Clausthal:
  - Dmitry Mazilkin
- From IWR, Heidelberg University:
  - Peter Bastian
  - Markus Blatt
  - Jö Fahlke
  - Felix Heimann
  - Dominic Kempf
  - Ole Klein
  - Steffen Müthing
- German Research Foundation (DFG)
- Jülich Supercomputing Center
Virtual soil systems: solute transport

$1024 \times 1024 \times 810 \approx 850 \cdot 10^6$ unknowns, 30252 time steps simulated in 18.7 hours on 128 nodes (4096 processes) of JUQUEEN (Bluegene/Q)