Universität Hamburg

KlimaCampus

Klima-Exzellenz in Hamburg

Zentrum für Marine und Atmosphärische Wissenschaften

Max-Planck-Institut für Meteorologie

GKSS Forschungszentrum Geesthacht

Deutsches Klimarechenzentrum

Universität Hamburg

KlimaCampus

Efficient Adaptive Methods for Multi-Scale Geophysical Applications

16.04.2010

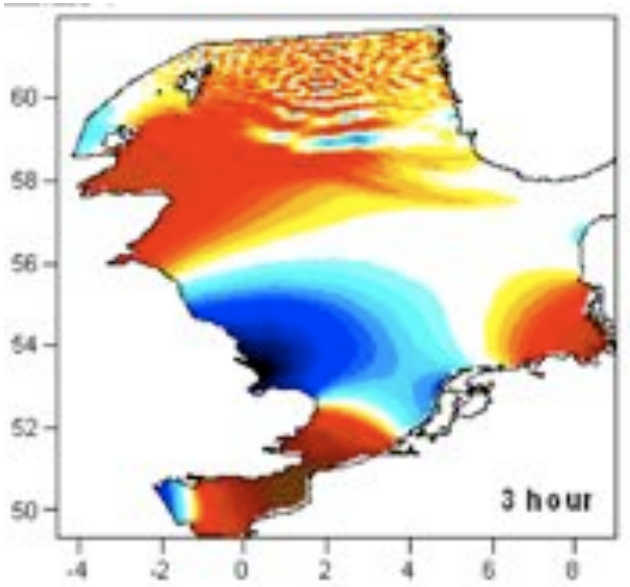
Jörn Behrens

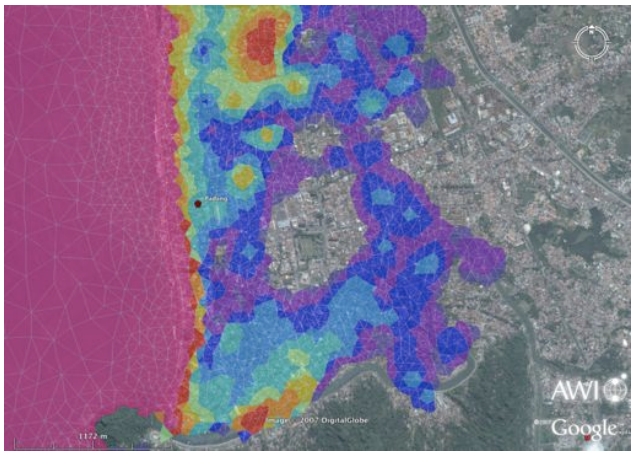
Universität Hamburg

Multiple scales – Tsunami

Spatial Scales:

- Tides:
  - Ocean-wide/ global range
  - O(10000) km
- Tsunami Wave Propagation:
  - Typical Wave Length
  - O(100) km
- Near-shore waves:
  - Shoaling effect
  - O(1-10) km
- Inundation (Parameterized):
  - No resolution of buildings, etc.
  - O(0.01-0.1) km

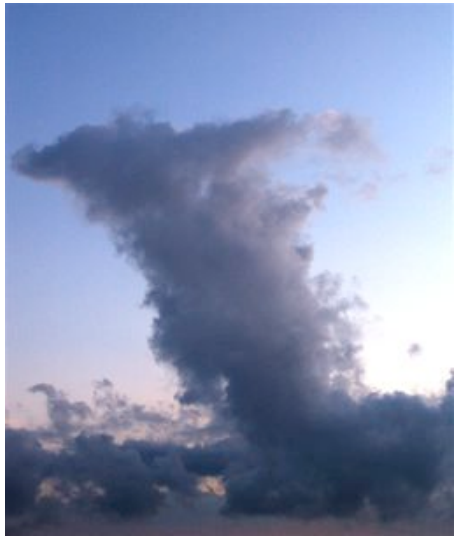


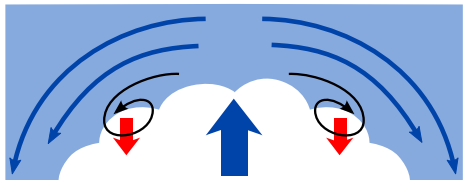


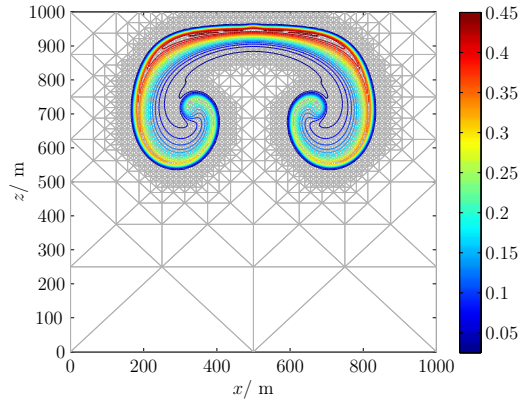
Multiple scales – Cloud Entrainment

Spatial Scales:

- Cloud cluster:
  - One single (tropical) cloud cluster
  - O(10) km^3
- Mixing processes at boundary:
  - Entrainment
  - O(1) m^3
- Microphysics:
  - Droplets, nucleation, etc.
  - O(10^{-3}–10^{-6}) m^3







Multiple scales – Computational consequences

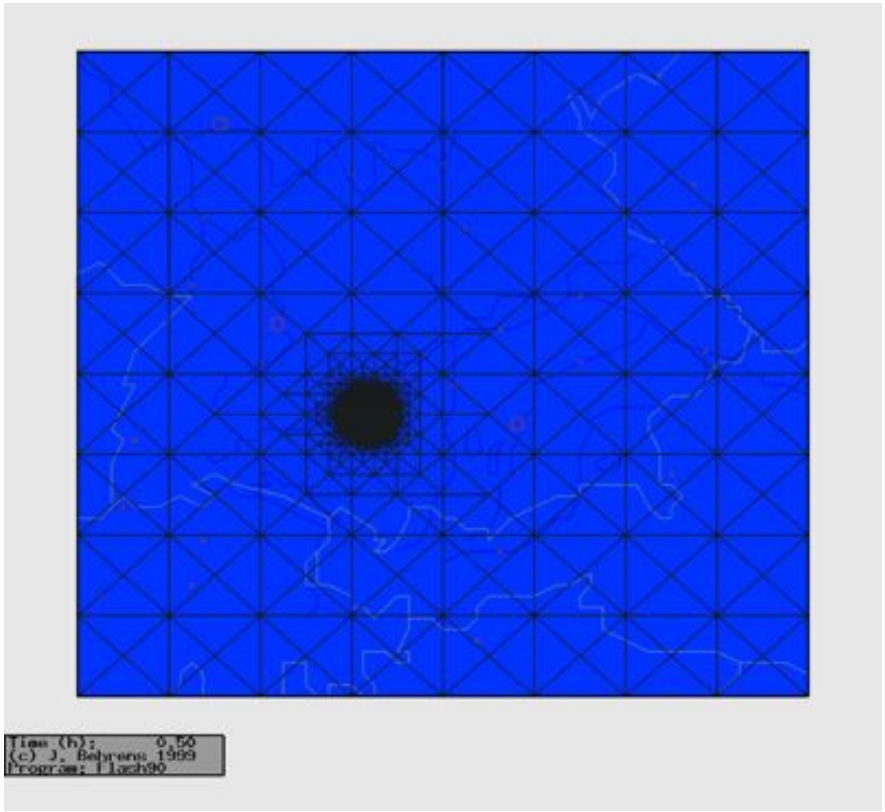
Computational cost uniform refinement

- Cloud cluster:
  - 10^4 grid points per dimension
  - $(10^4)^3 = 10^{12}$ grid points!
  - That is without microphysics!

- Tsunami:
  - 10^5 grid points per dimension
  - $(10^5)^2 = 10^{10}$ grid points
  - That does not resolve streets, etc.

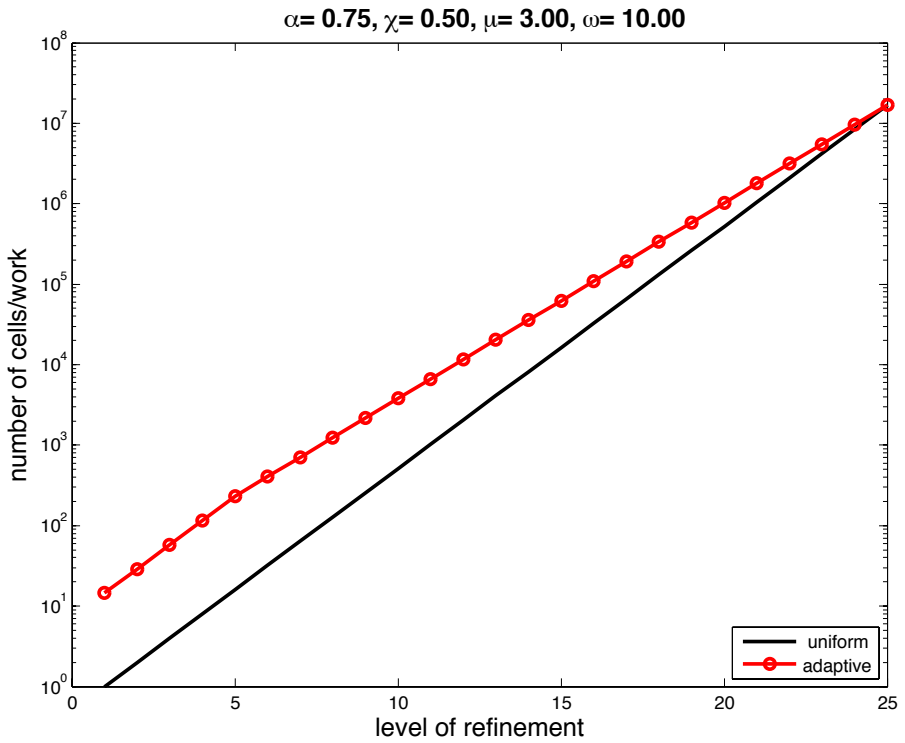
Adaptive Methods – Key Questions

- When to apply an adaptive method?
- What is needed for numerical efficiency?
- What is needed for computational efficiency?



Model for adaptive efficiency

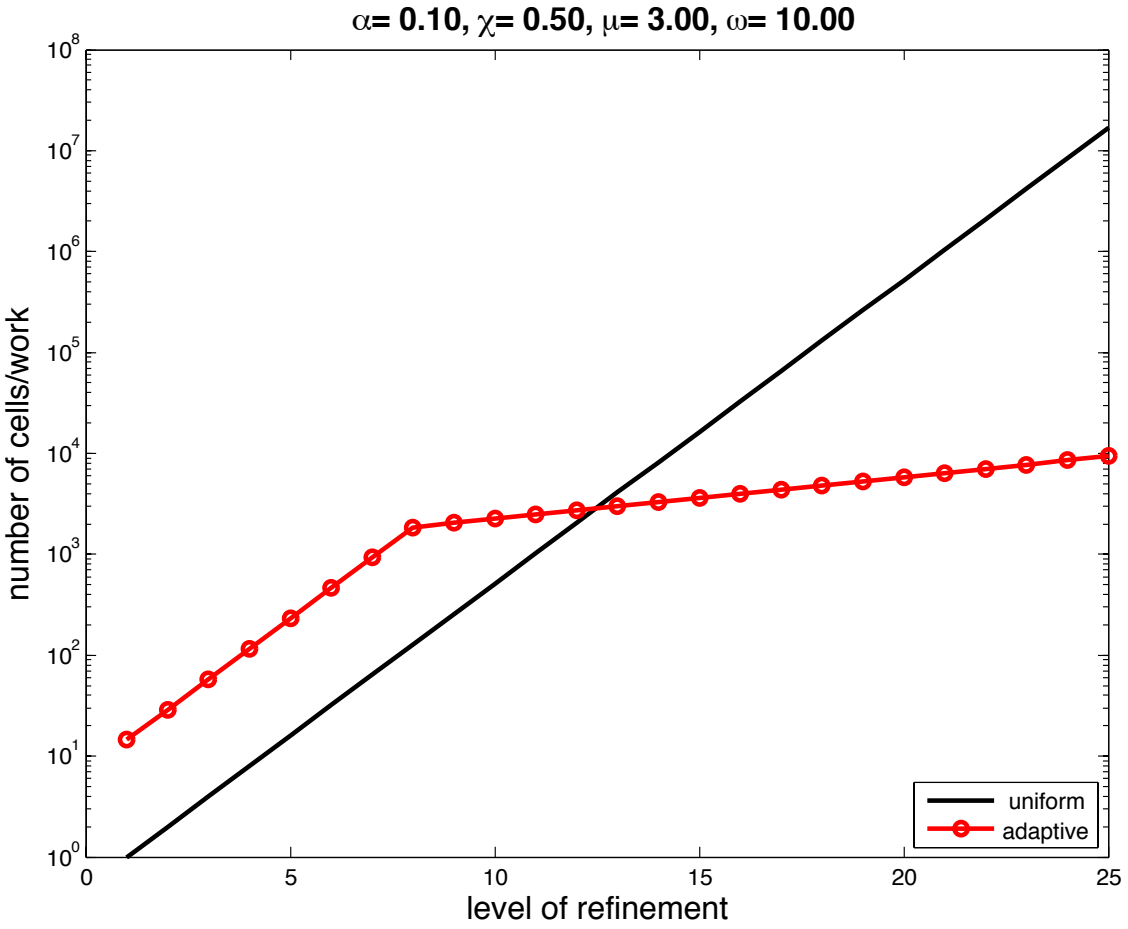
- Assumptions:
  - Tree refinement (binary tree here)
  - Overheads for
    - refinement criterion
    - Less efficient numerics
    - Mesh management
  - Refinement area as fraction
  - Computational work = no. of unknowns



- Example:
  - Area of refinement large (α=3/4)
  - Overheads large:
    - Refinement criterion (χ=1/2 of a computational step)
    - Numerics slow (ω= 10 times slower as uniform comp.)
    - Mesh management demanding (μ= 3 times a comp. step)

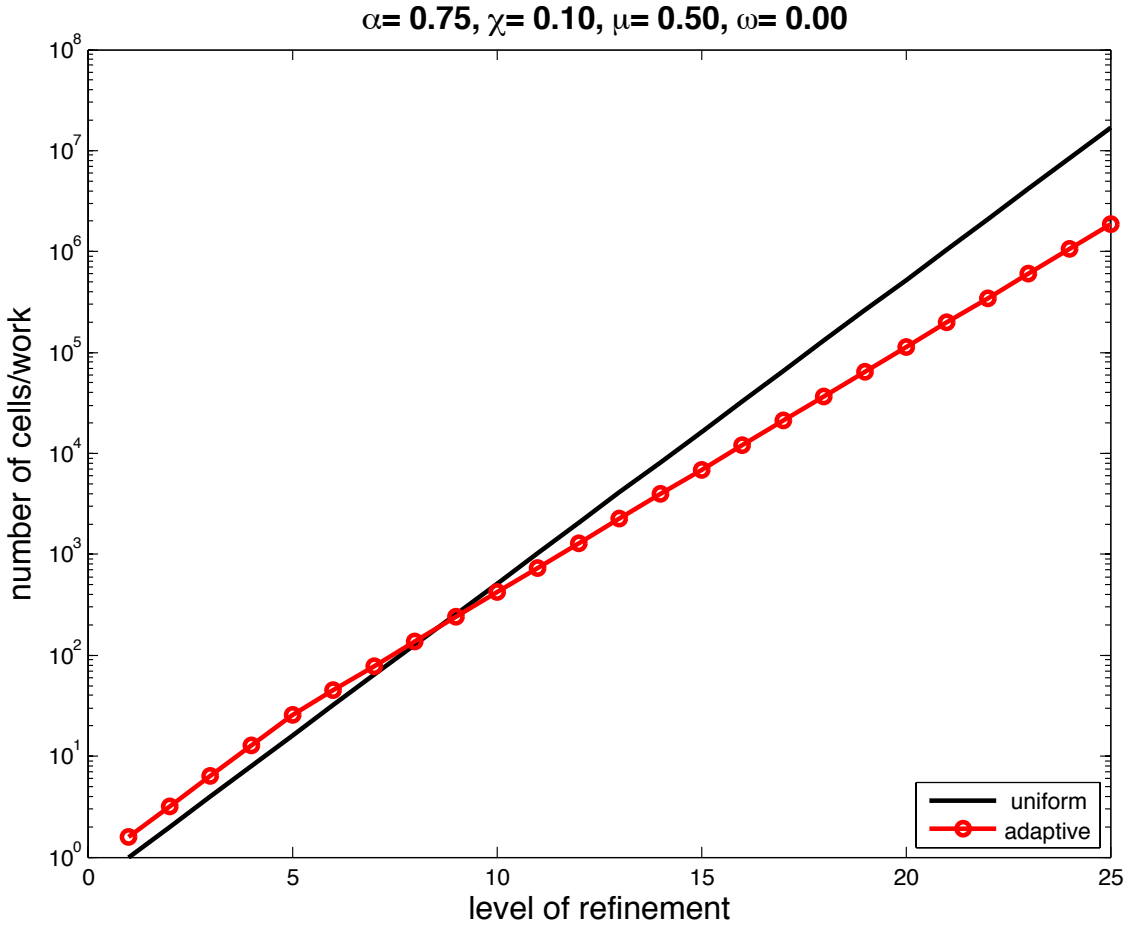
Model for adaptive efficiency – small adaptive area

- Now:
  - Area of refinement small (α=1/10)
  - Rest unchanged



Model for adaptive efficiency – small adaptive area

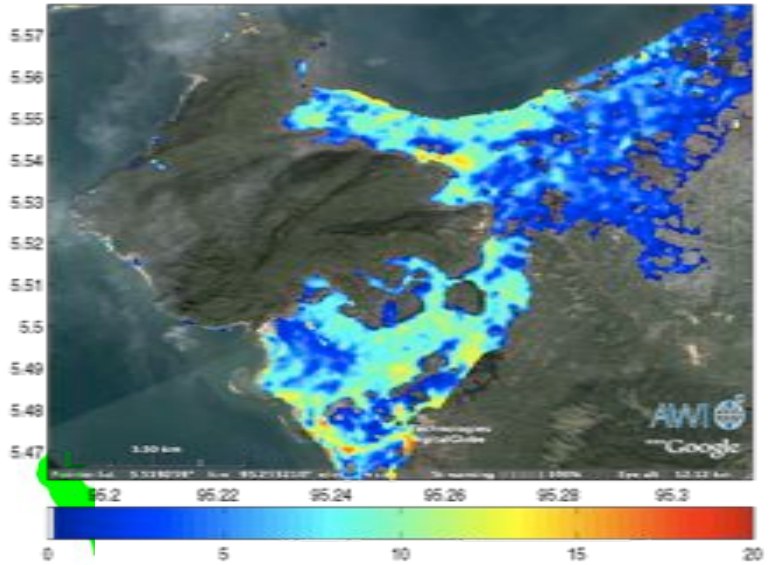
- Now:
  - Area of refinement large (α=3/4)
  - But efficient grid management and numerics!

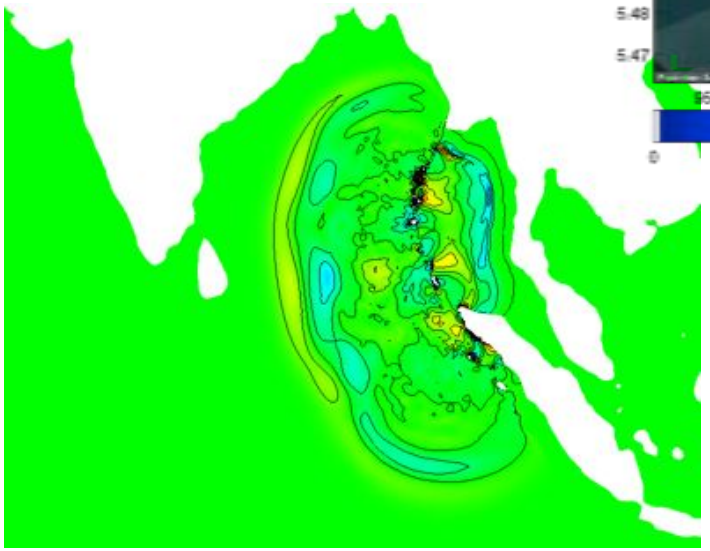


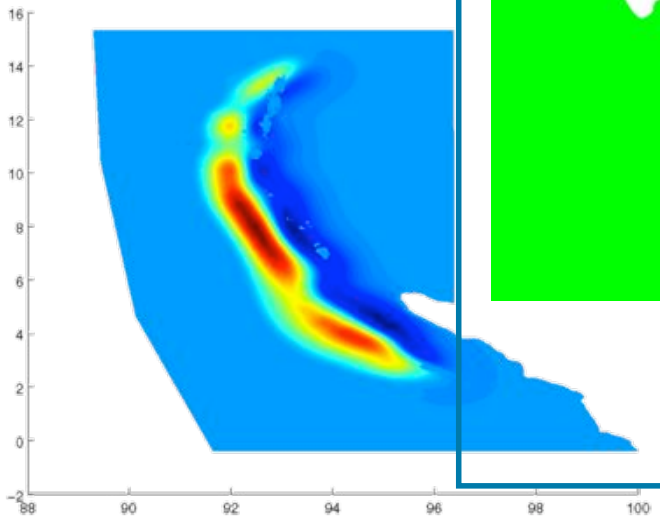
Simulation overview

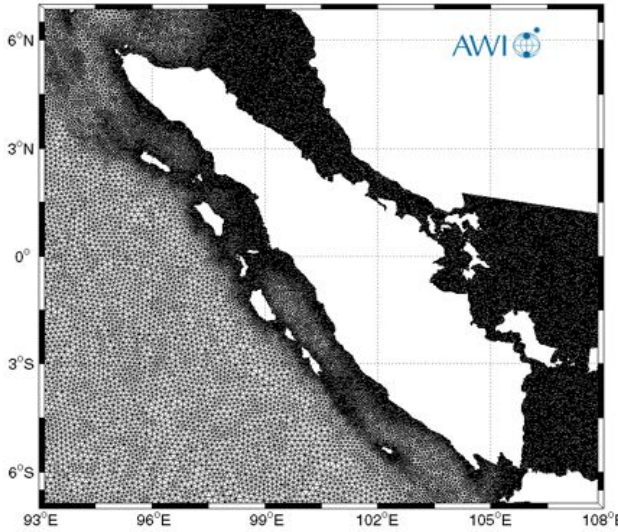
Simulation chain:

- Source
- Propagation
- Inundation









Refined mesh for seamless

Multi-scale representation

Universität Hamburg

KlimaCampus

Basic equations

Shallow Water Equations

Continuity equation (flux form)

$$\frac{\partial \zeta}{\partial t} + \nabla \cdot (\mathbf{v}(\zeta + H)) = 0$$

Momentum equation

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} + g\nabla\zeta + f \times \mathbf{v} + \frac{\mathrm{n}^2 \mathbf{v}|\mathbf{v}|}{\sqrt[3]{H}} - \nabla \cdot (K_h \nabla \mathbf{v}) = 0$$

Coriolis term

Bottom friction

Viscosity term

Boundary conditions

Radiation boundary condition (open/liquid boundary)

$$\mathbf{v} \cdot \mathbf{n} = \sqrt{\frac{g}{H + \zeta}}\,\zeta$$

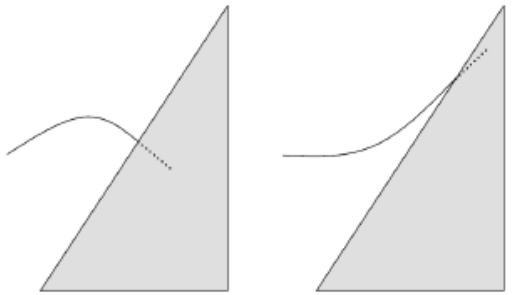
No-slip boundary condition (solid boundary)

$$\mathbf{v} \cdot \mathbf{n} = 0$$

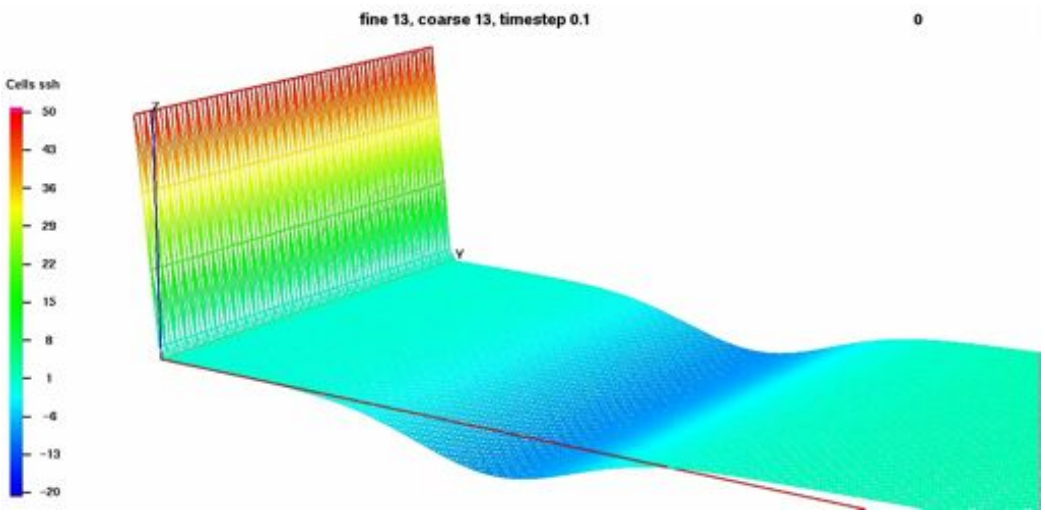
Note: inundation boundary conditions according to Lynett/Wu/Liu (2002)

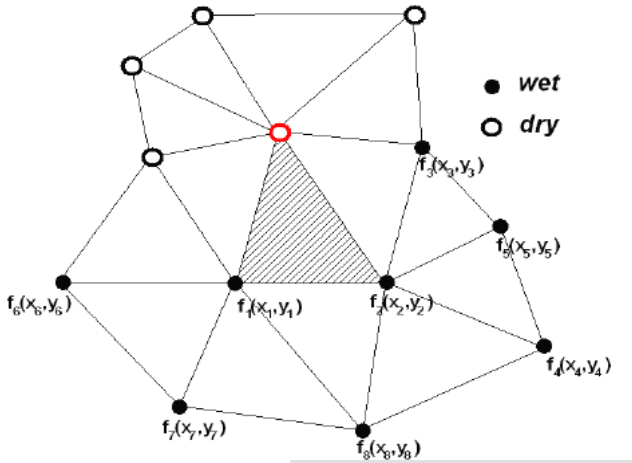
Inundation boundary conditions

Extrapolation of wave height



Extrapolation based on neighbors





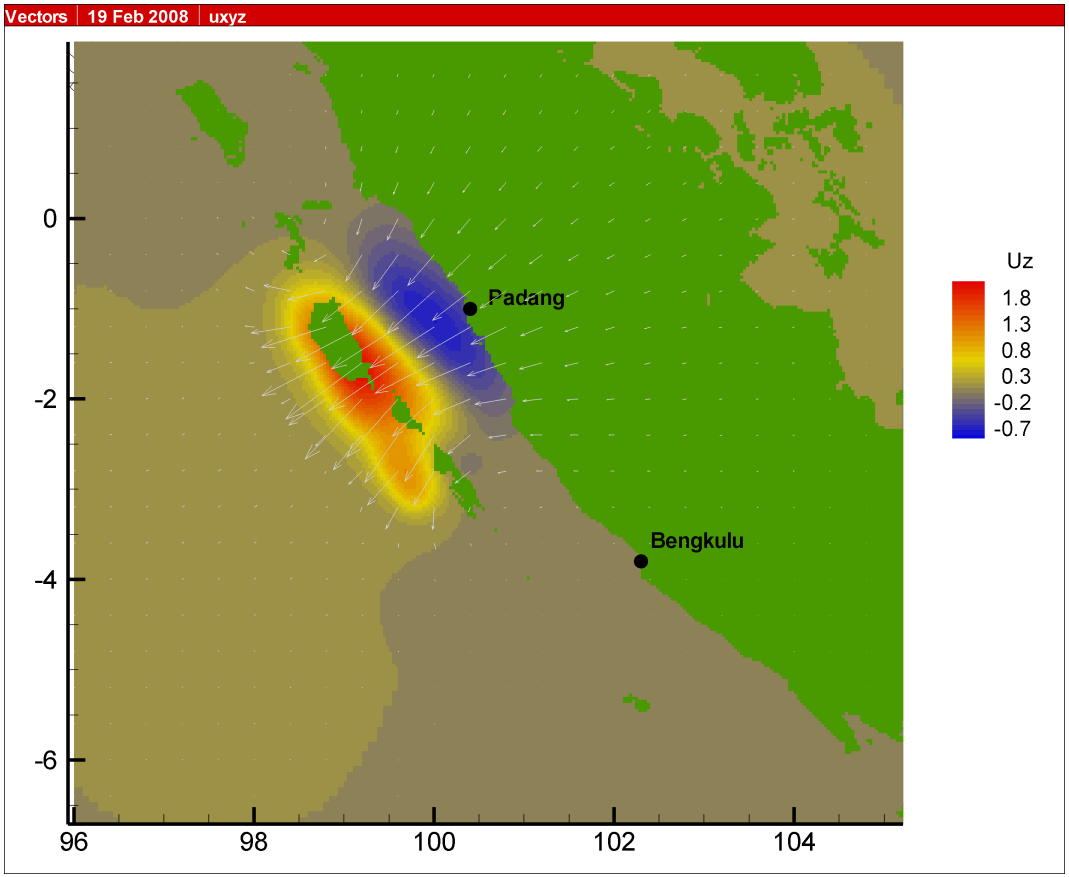
Lynnet, Wu, Liu, 2002

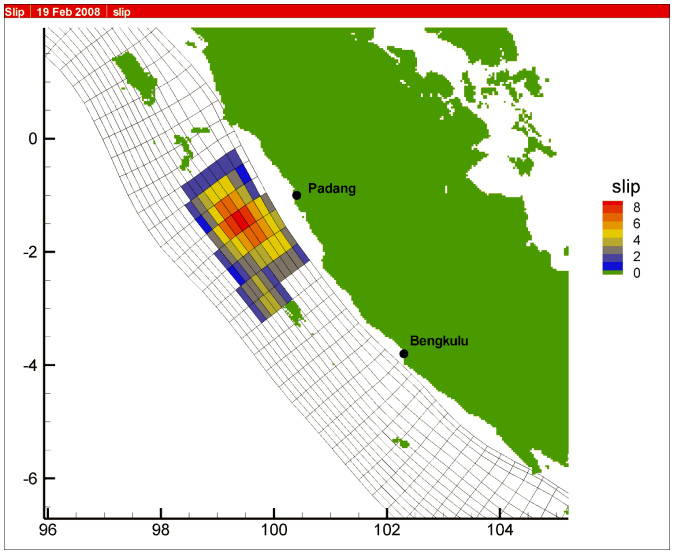
Universität Hamburg

KlimaCampus

Initial conditions

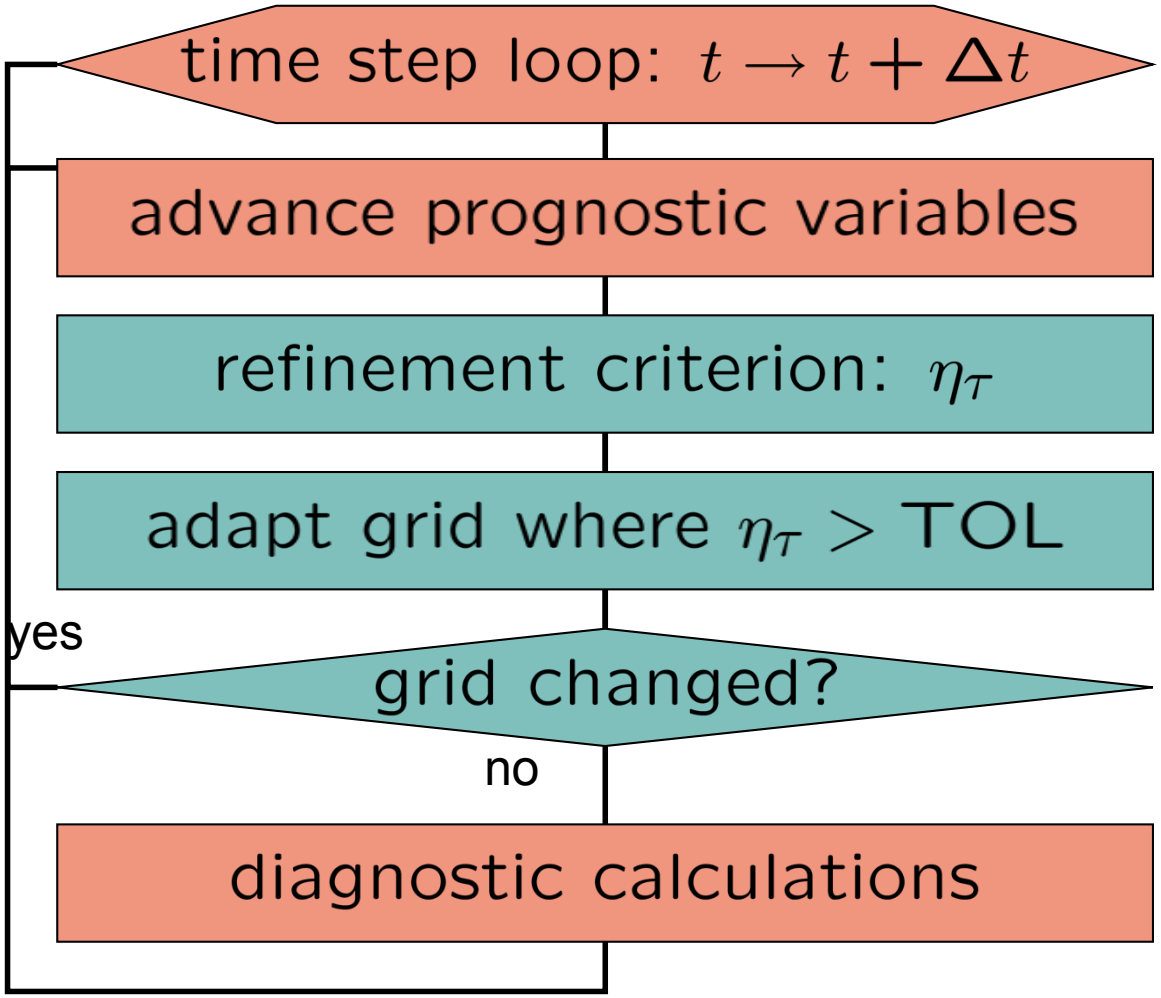
Complex slip distribution



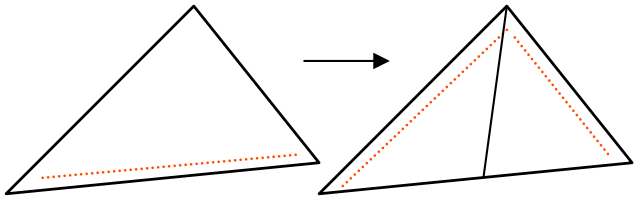


Babeyko (2007)

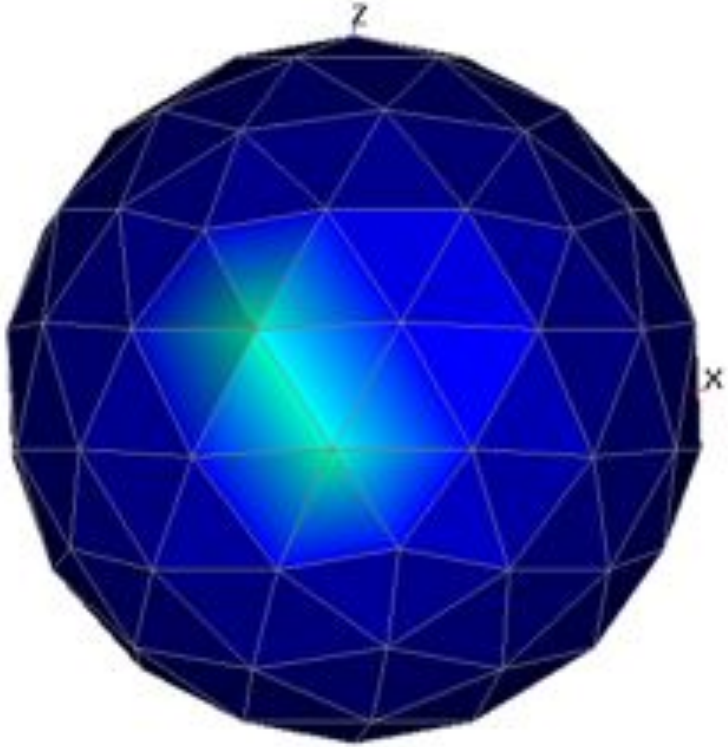
Adaptive algorithm



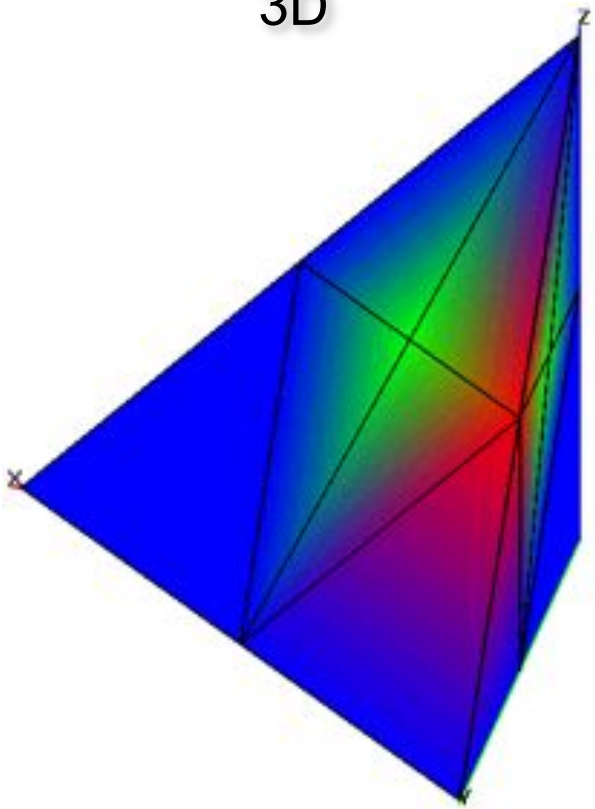
1. Efficient Mesh Refinement Strategy



2D



3D



Rivara (1984), Bänsch (1991), Grids created with amatos

2. Efficient Refinement Criterion

Gradient based criterion

$$\text{refine if} \quad \nabla h|_\tau > \theta_{\text{ref}} \max(\nabla h|_\tau)$$

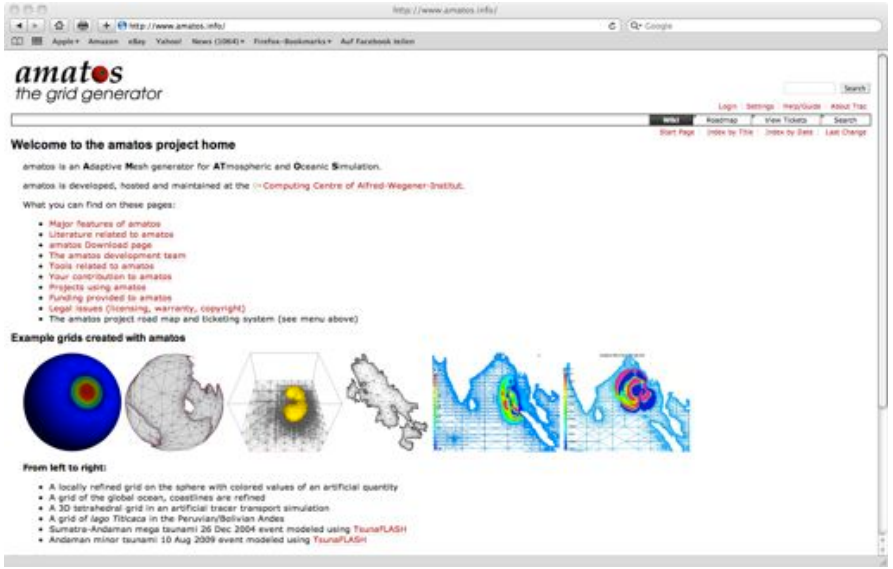
$$\theta_{\text{ref}} \text{ threshold value}$$

- ☑ Simple
- ☑ Easy to compute

- ☒ Not robust

3. Efficient Programming Framework

- Triangular grids with boundary
- Object oriented + gather/scatter
- Built in SFC ordering
- Simple programming interface
- Generic FEM/SEM support
- Coupling capability
- Parallel
- 2D plane and spherical geometries
- Documentation, open source
- http://www.amatos.info

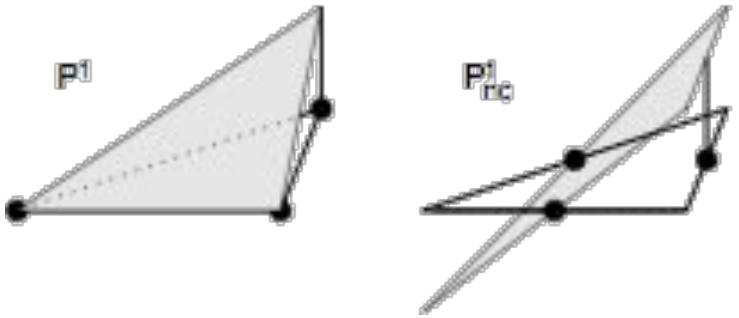


4. Efficient and robust (!) numerical scheme

P^1-P^1_{NC} finite element combination

Conforming linear for ζ and H:

$$\hat{h}: \quad \hat{h}_1 = 1 - x - y;\ \hat{h}_2 = x;\ \hat{h}_3 = y$$



Hanert et al. (2005)

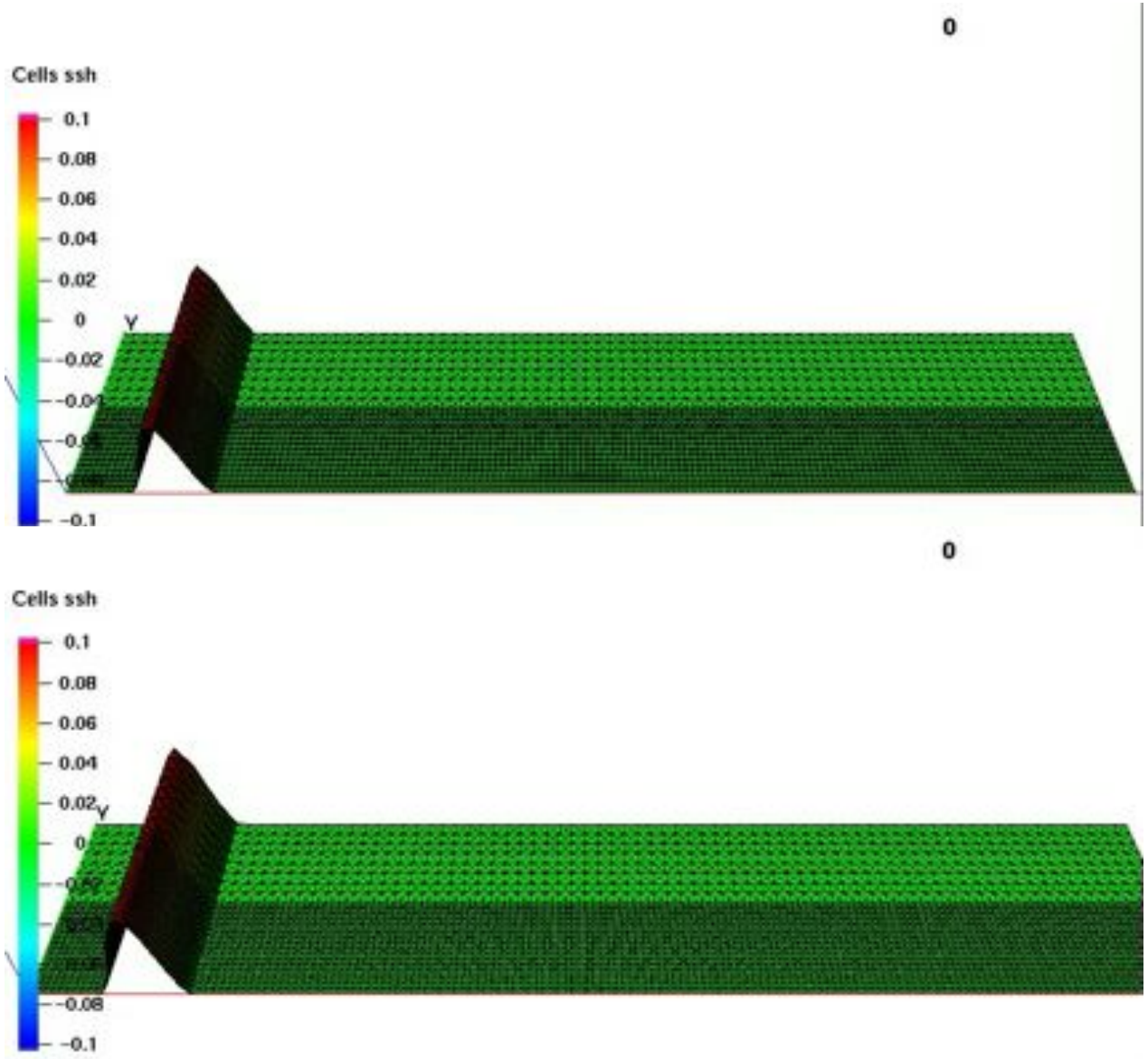
Non-Conforming linear for v

$$\hat{v}: \quad \hat{v}_1 = 1 - 2y;\ \hat{v}_2 = 2x + 2y - 1;\ \hat{v}_3 = 1 - 2x$$

Exact discrete mass conservation

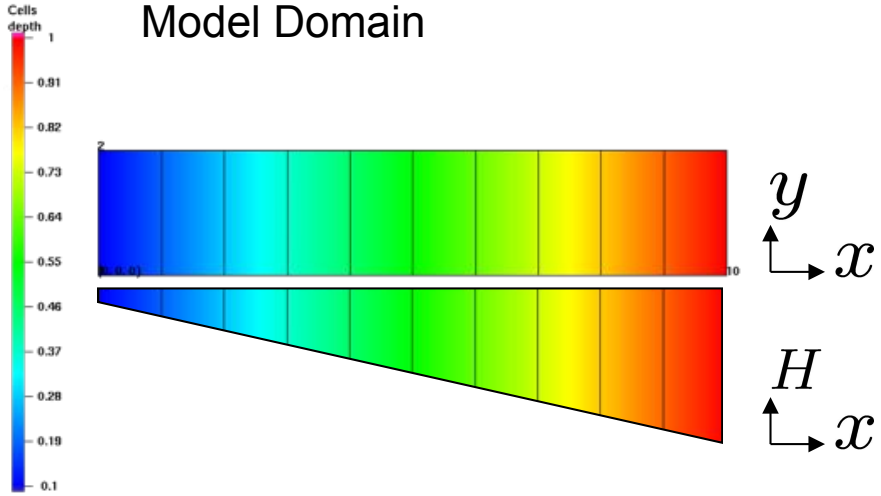
4. Efficient and robust (!) numerical scheme

Explicit RK nodal DG method (See Giraldo)



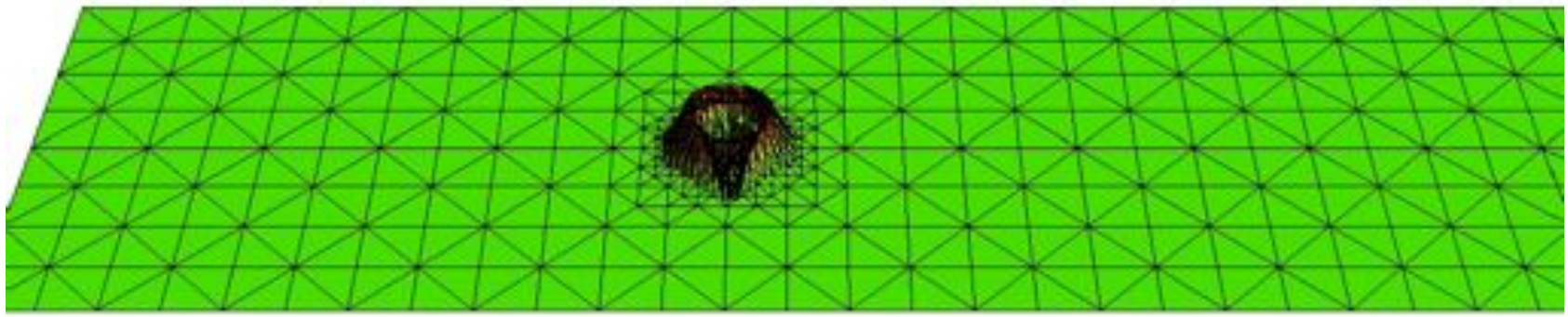
Adaptive mesh refinement – simple example

Model Domain

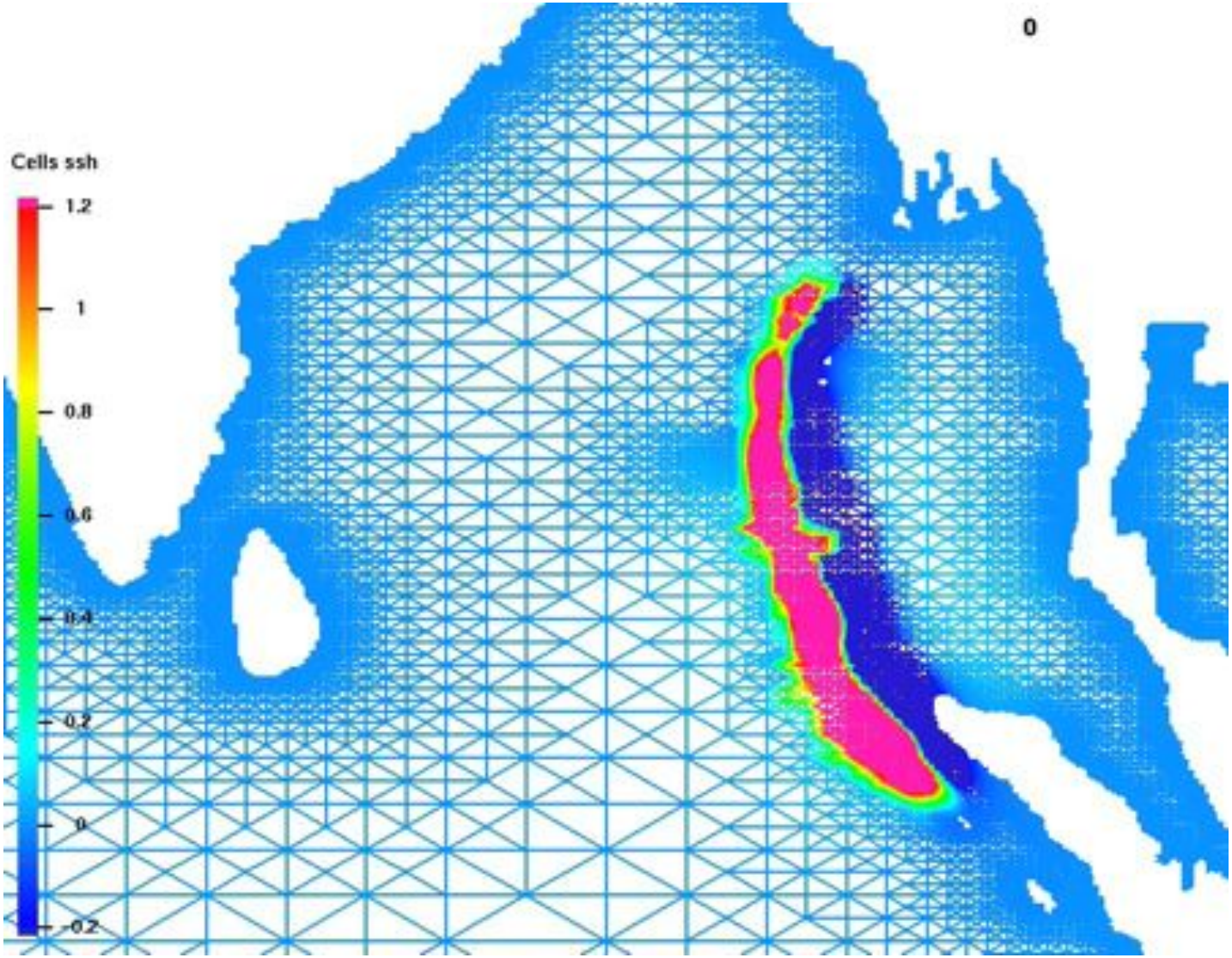


Initial values

$(3, 1)$



Andaman-Sumatra Rupture



J.B., W. S. Pranowo

Error estimation II (averaging)

Instead of

$$\text{refine if} \quad \nabla h|_\tau > \theta_{\text{ref}} \max(\nabla h|_\tau)$$

θ_{ref} threshold value

Use

$$\eta = \|h|_\tau - \bar{h}\|, \quad \text{with } \bar{h} = \frac{1}{A} \sum_{\text{neighbors t}} h|_t,$$

$$\text{refine if} \quad \eta > \theta_{\text{ref}} \bar{\eta}, \quad \text{with } \bar{\eta} \text{ mean error}$$

θ_{ref} threshold value

- ☑ Simple
- ☑ Easy to compute
- ☑ Robust

Error estimation III

Instead of

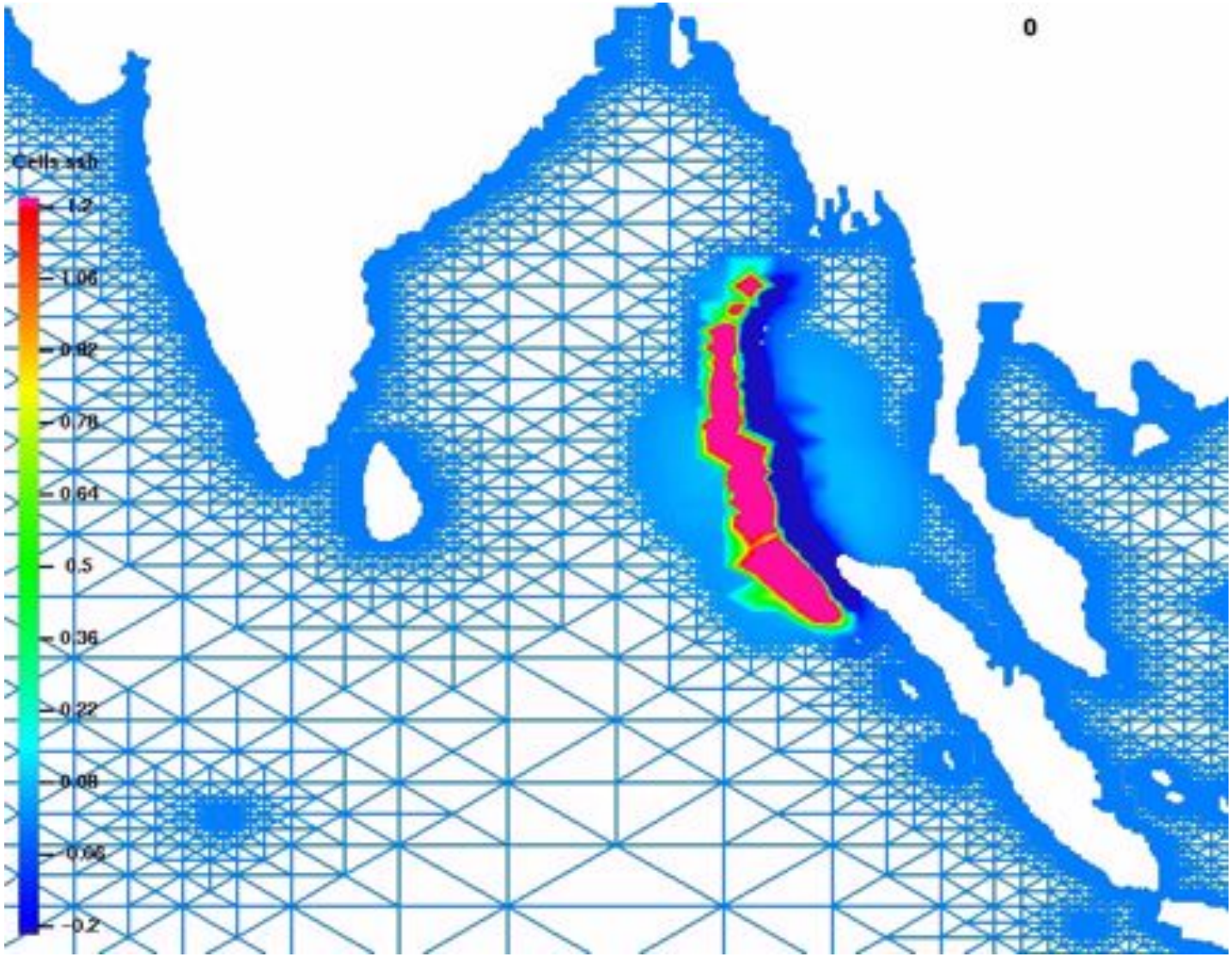
$$\text{refine if} \quad \nabla h|_\tau > \theta_{\text{ref}} \max(\nabla h|_\tau)$$
$$\theta_{\text{ref}} \text{ threshold value}$$

Use

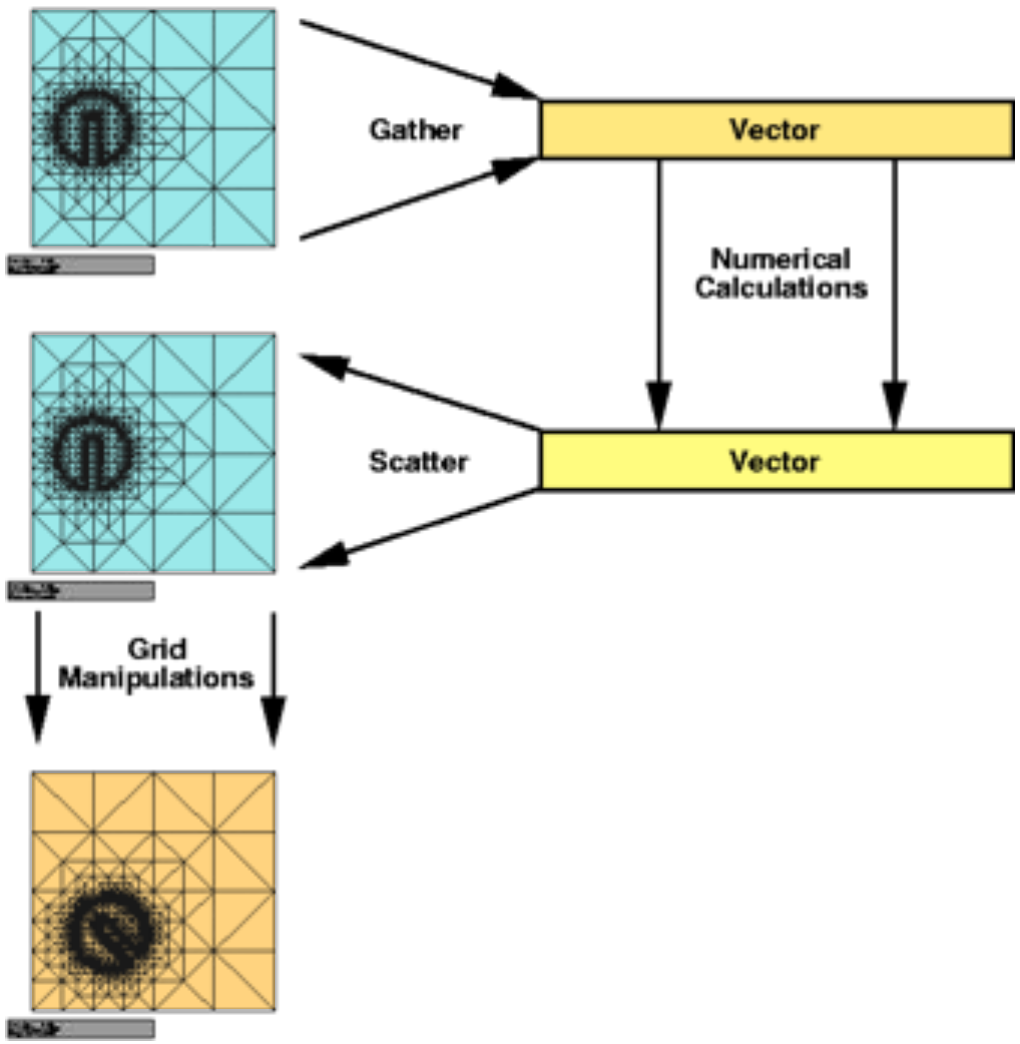
$$\text{refine if} \quad h|_\tau > \theta_{\text{ref}} \max(h|_\tau)$$
$$\theta_{\text{ref}} \text{ threshold value}$$

- ☑ Even simpler
- ☑ Even easier to compute
- ☑ Robust for this application!

Andaman-Sumatra Rupture

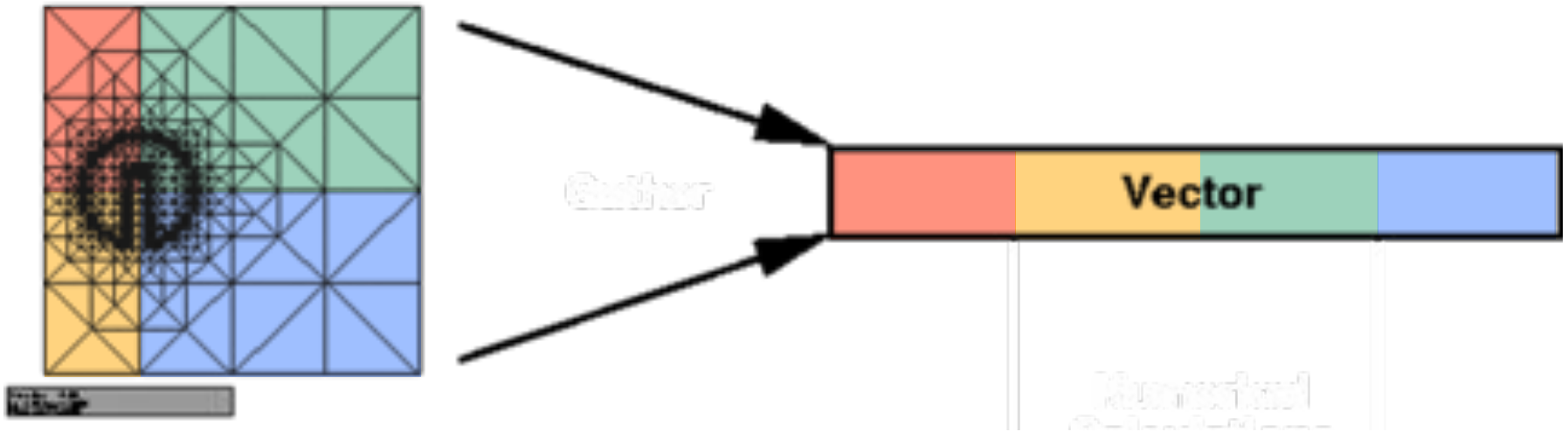


5. Efficient Data Management



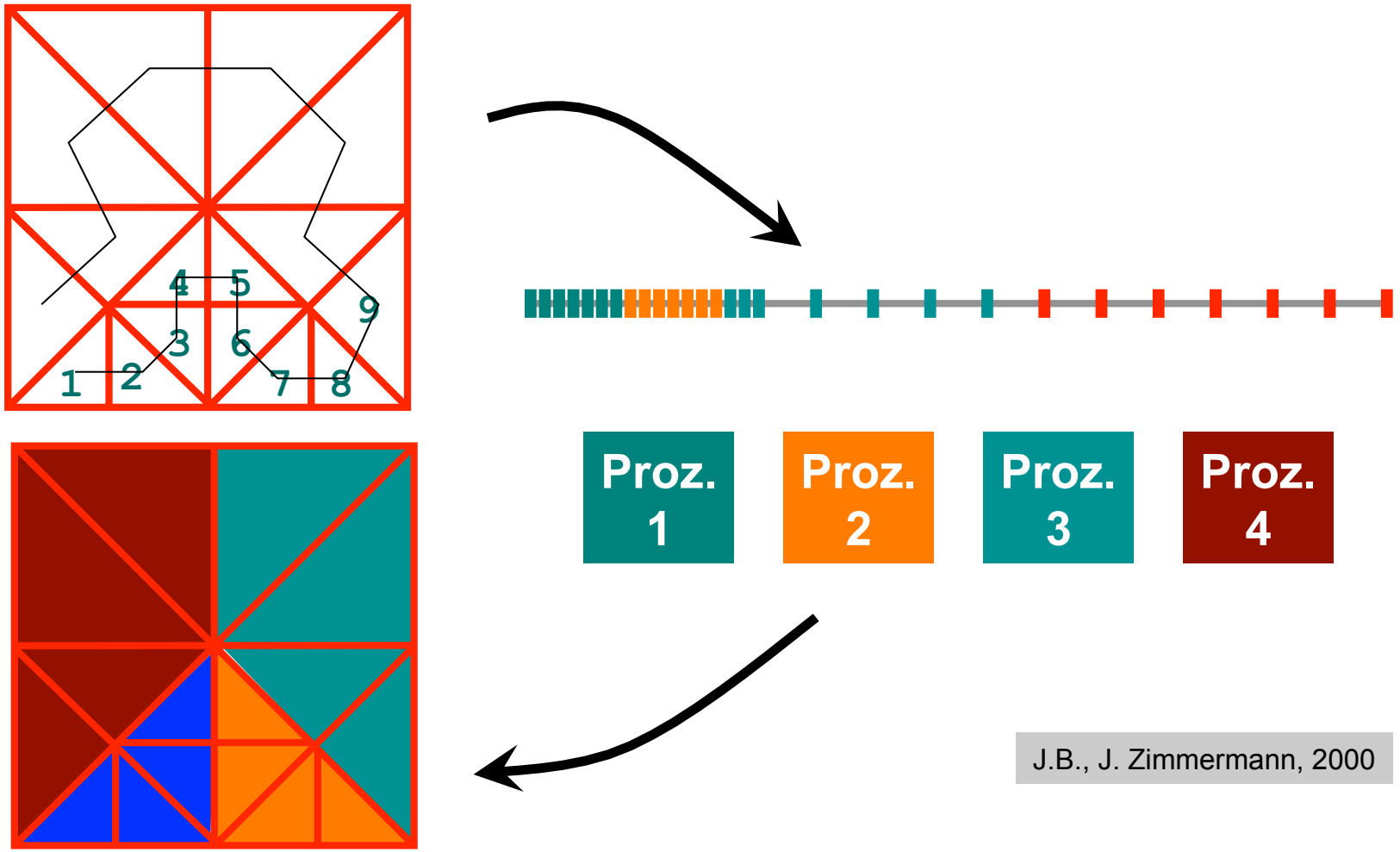
Data Locality

We need data to stay local!



Optimization Hierarchy Level 1: Parallel Partitioning

Space-filling curve (Sierpinski)

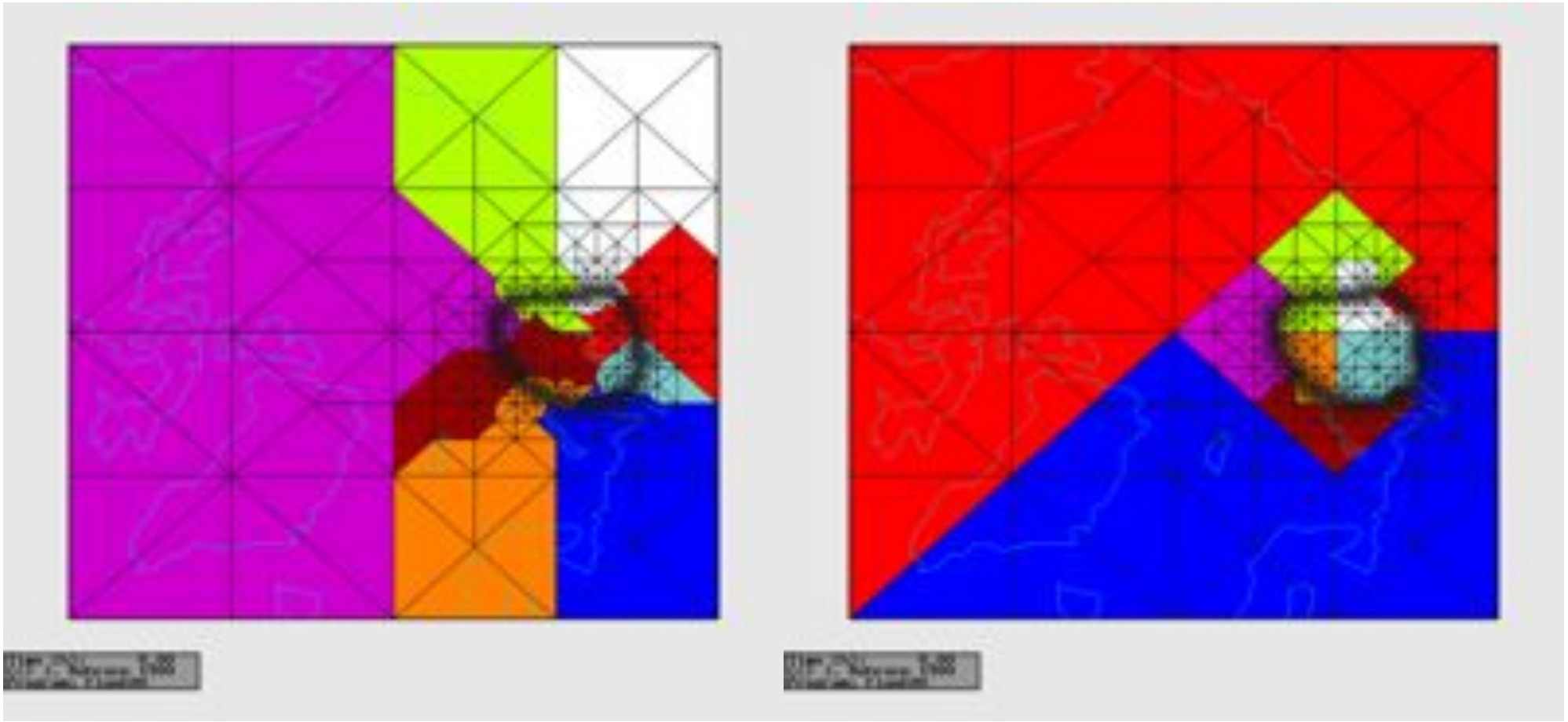


J.B., J. Zimmermann, 2000

Optimization Hierarchy Level 1: Parallel Partitioning

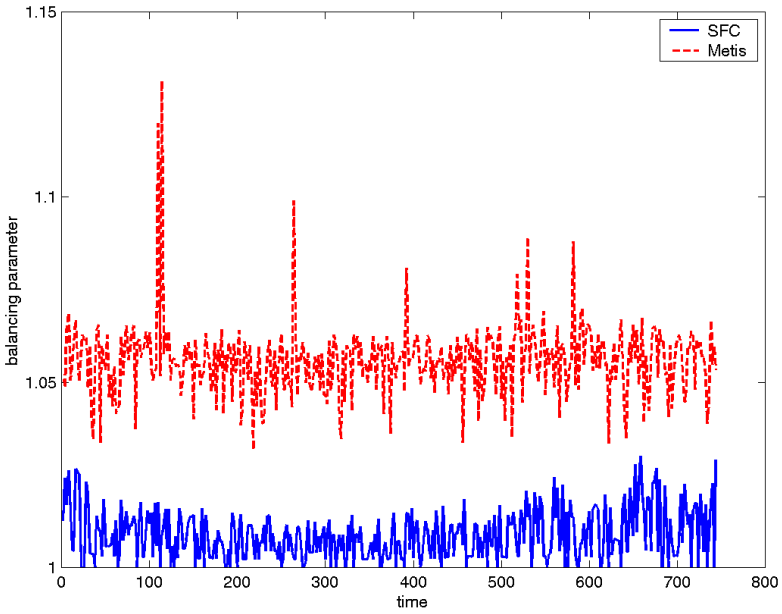
METIS

SFC

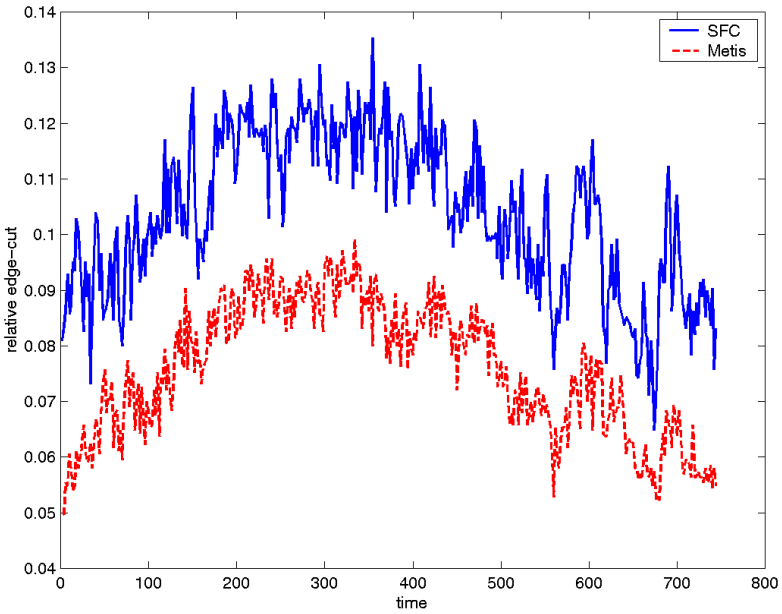


Load Balancing and Edge Cut for Tracer Experiment

Load balancing



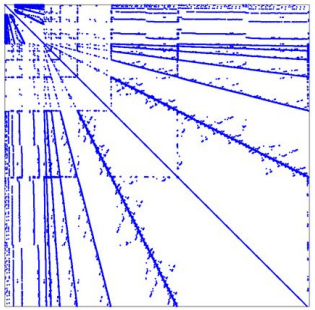
Edge-cut



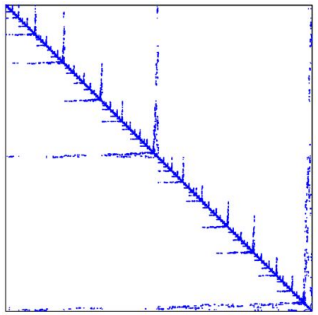
SFC: J. B., J. Zimmermann (2000), Metis: G. Karypis, V. Kumar (1998)

Optimization Hierarchy Level 2: Matrix Optimization

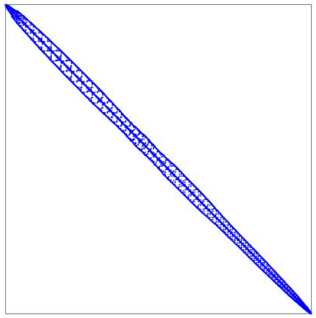
Structure of matrix



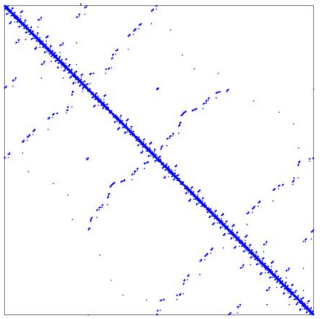
tree-sorted



quotient
minimum degree

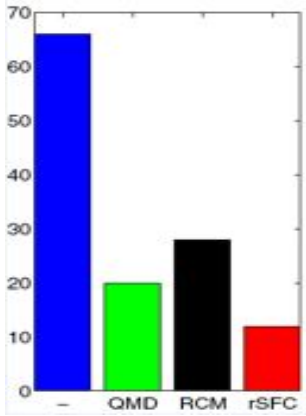


reverse
Cuthill-McKee

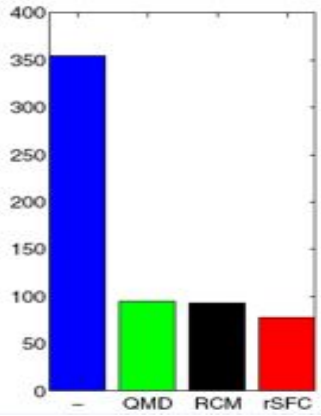


reverse SFC

- System with ~200.000 unknowns
- Utilizes preconditioned BiCGStab
- ILU pre-conditioning



Iterations

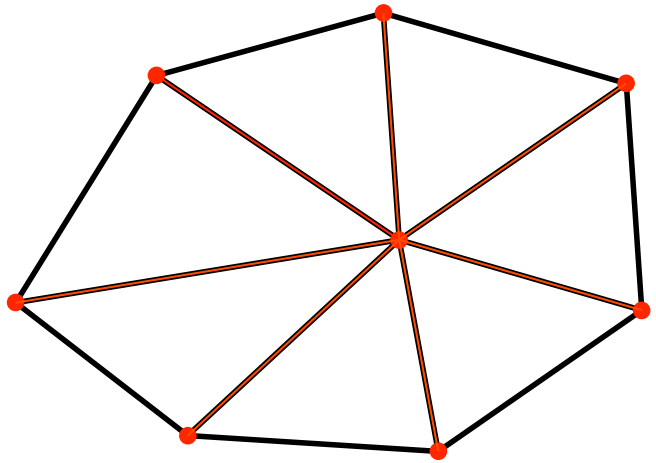


Time

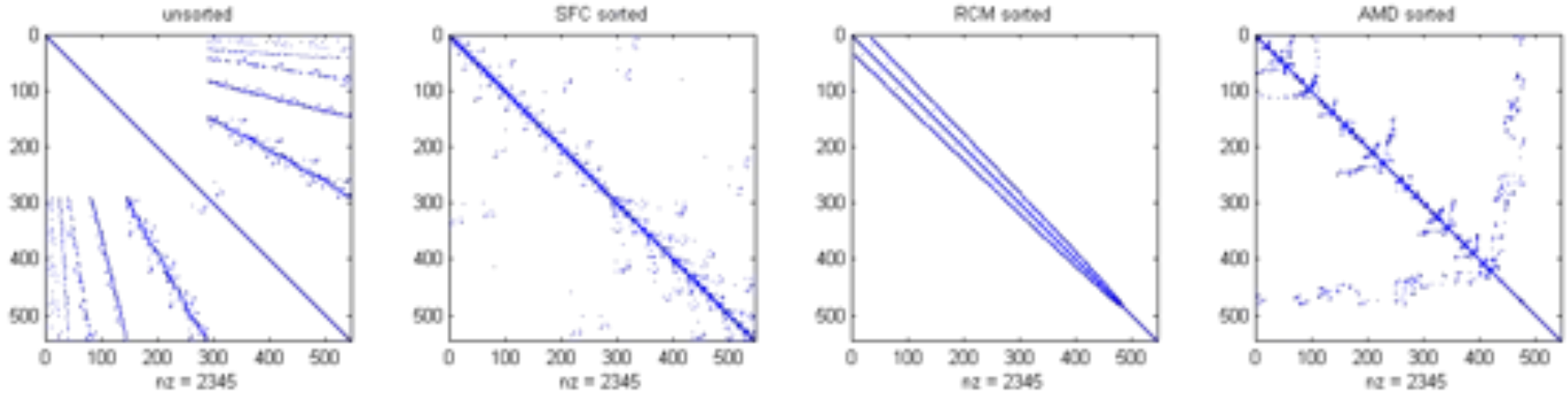
J. B., N. Rakowski, S. Frickenhaus, et al. (2003)

Optimization Hierarchy Level 3: Cache Optimization

Nearest neighbor communication (vertex-wise)

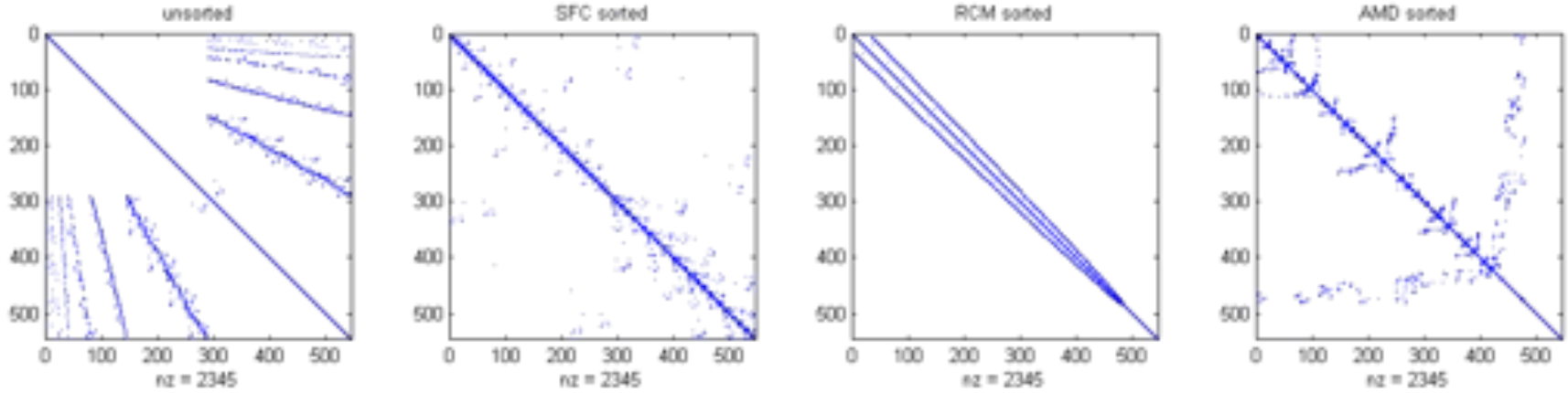


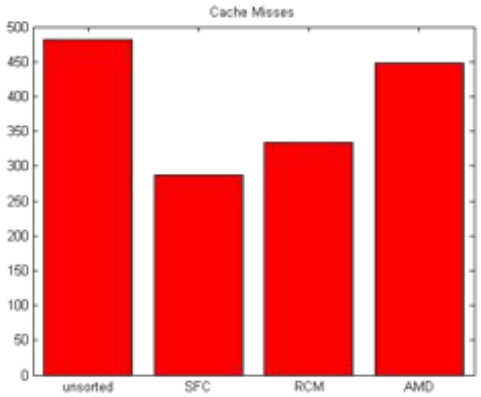
Connectivity matrix with different orderings



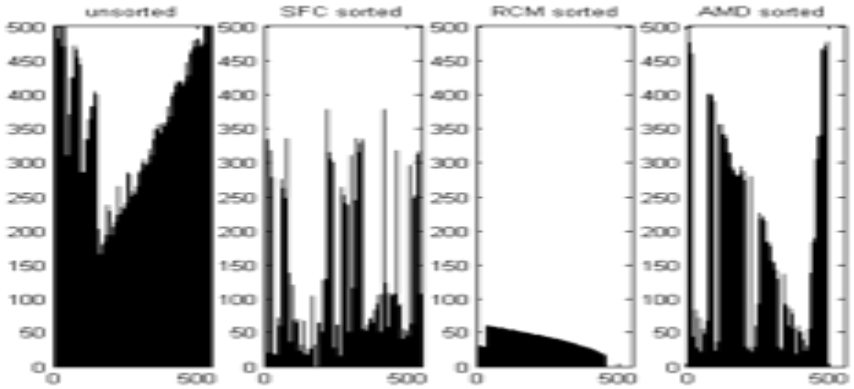
Optimization Hierarchy Level 3: Cache Optimization

Connectivity matrix with different orderings





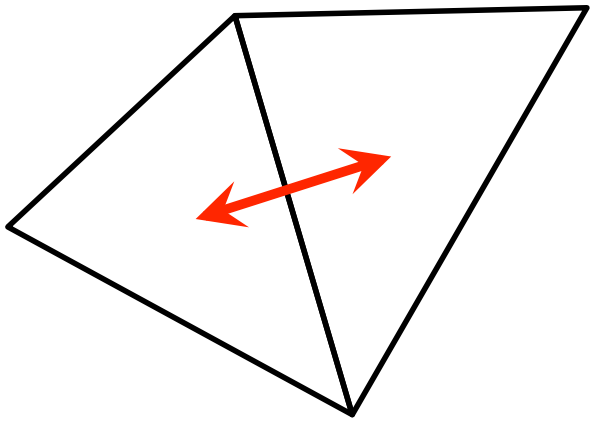
Cache misses



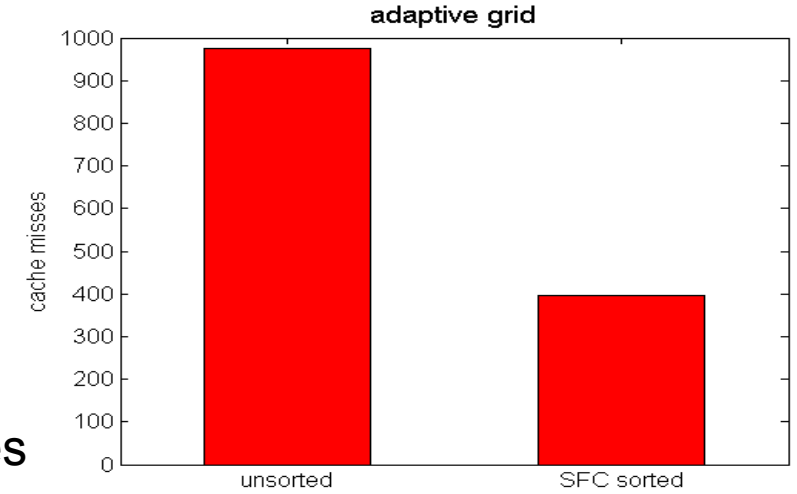
Distance structure

Optimization Hierarchy Level 3: Cache Optimization

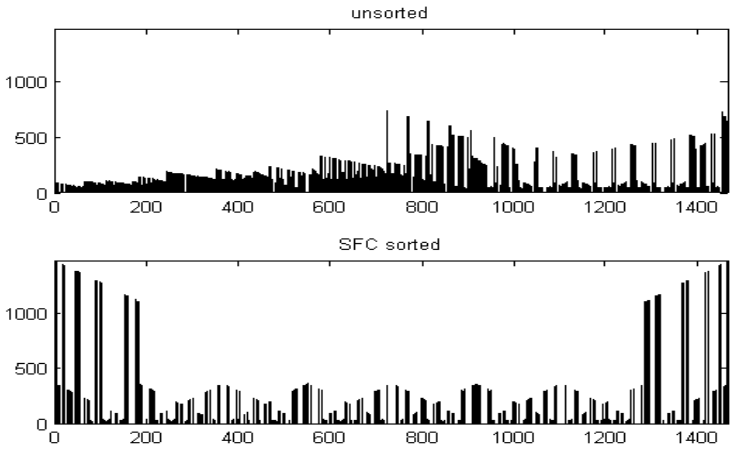
Nearest neighbor communication (element-wise)



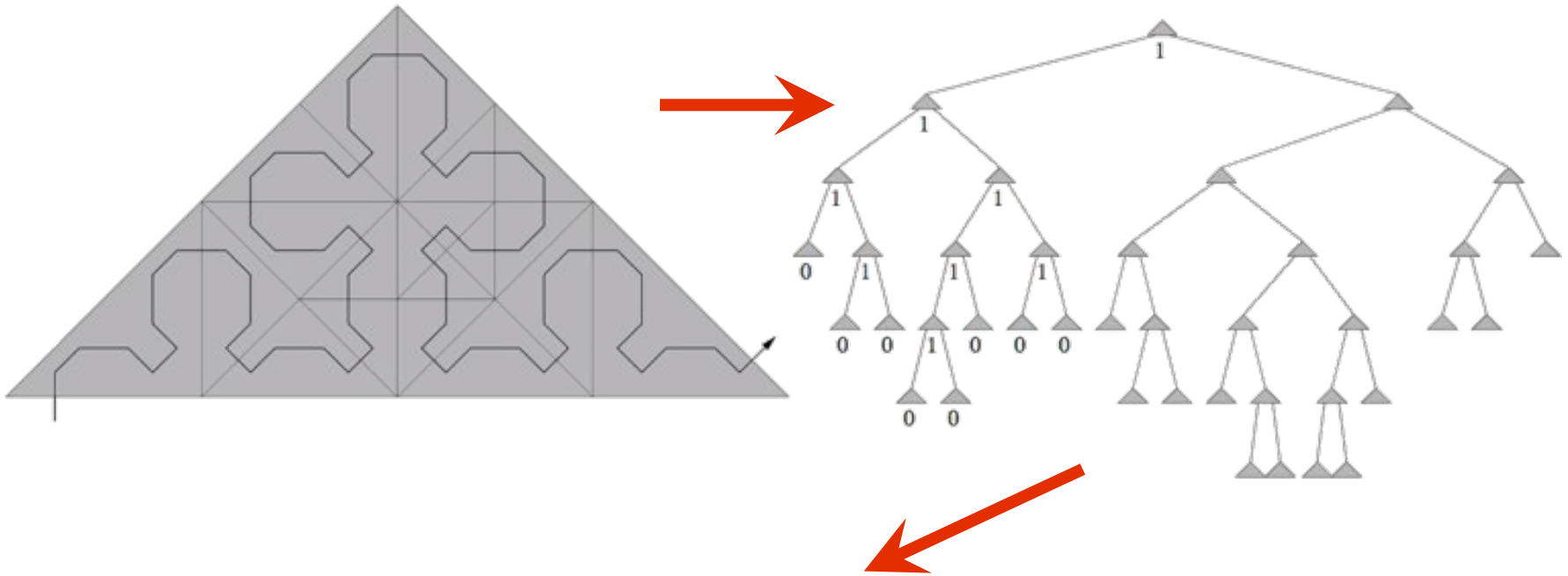
Cache misses



Distance structure



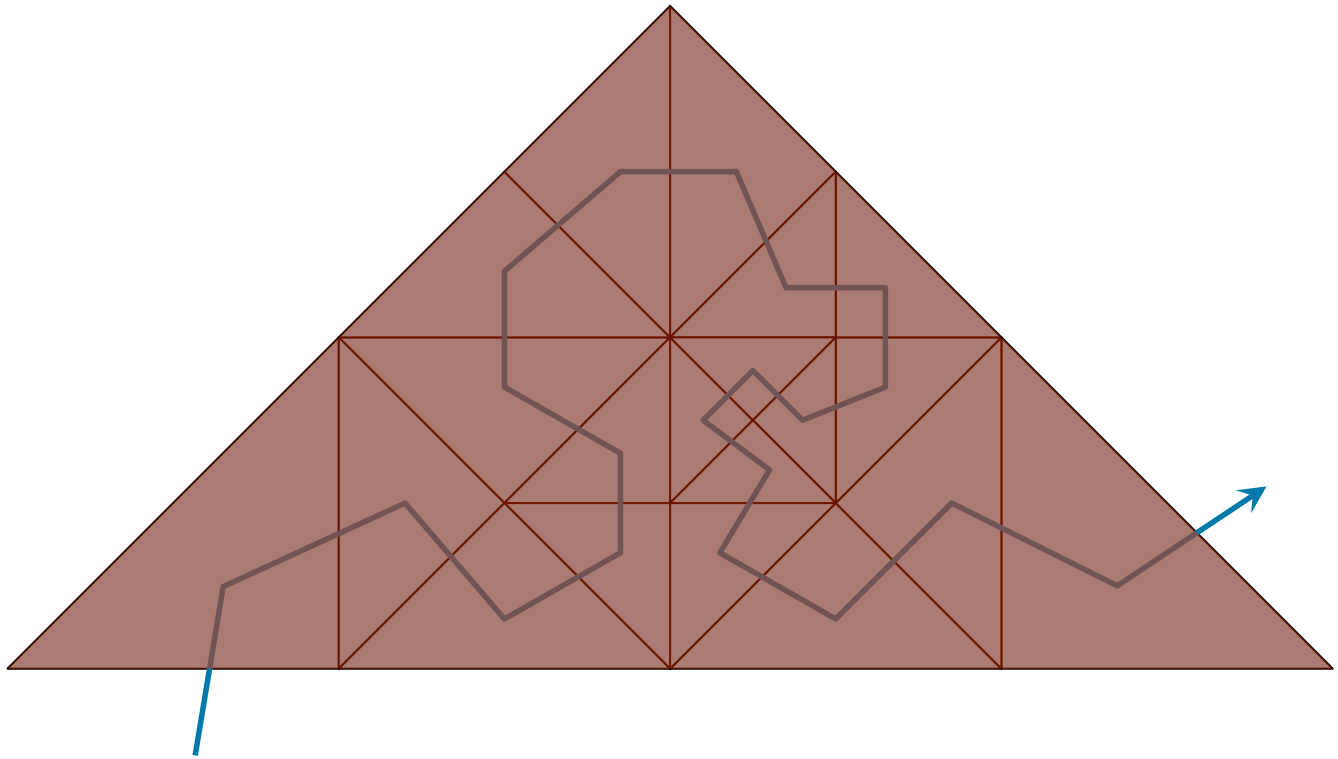
Observation 1: Refinement tree



1110100111000100...

Refinement tree as bitstream

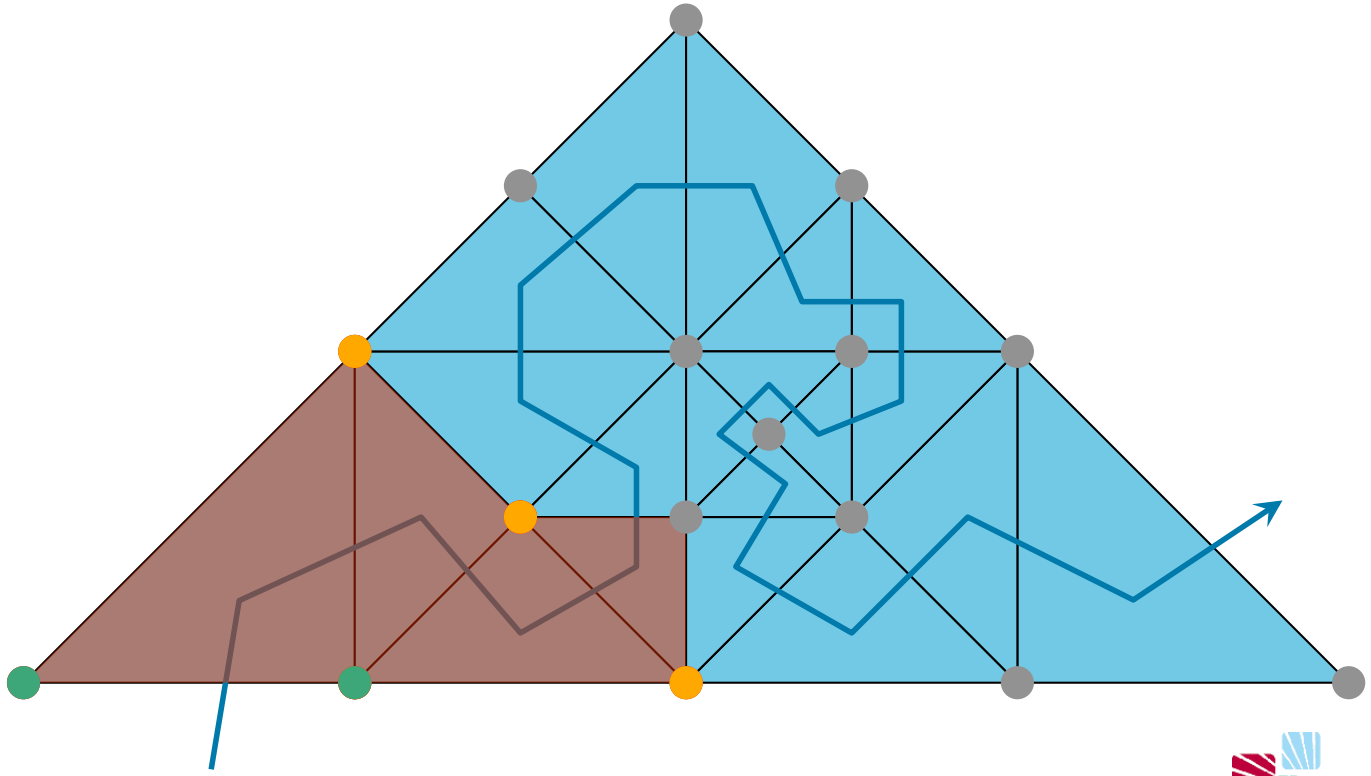
Observation 2: Element-wise computation



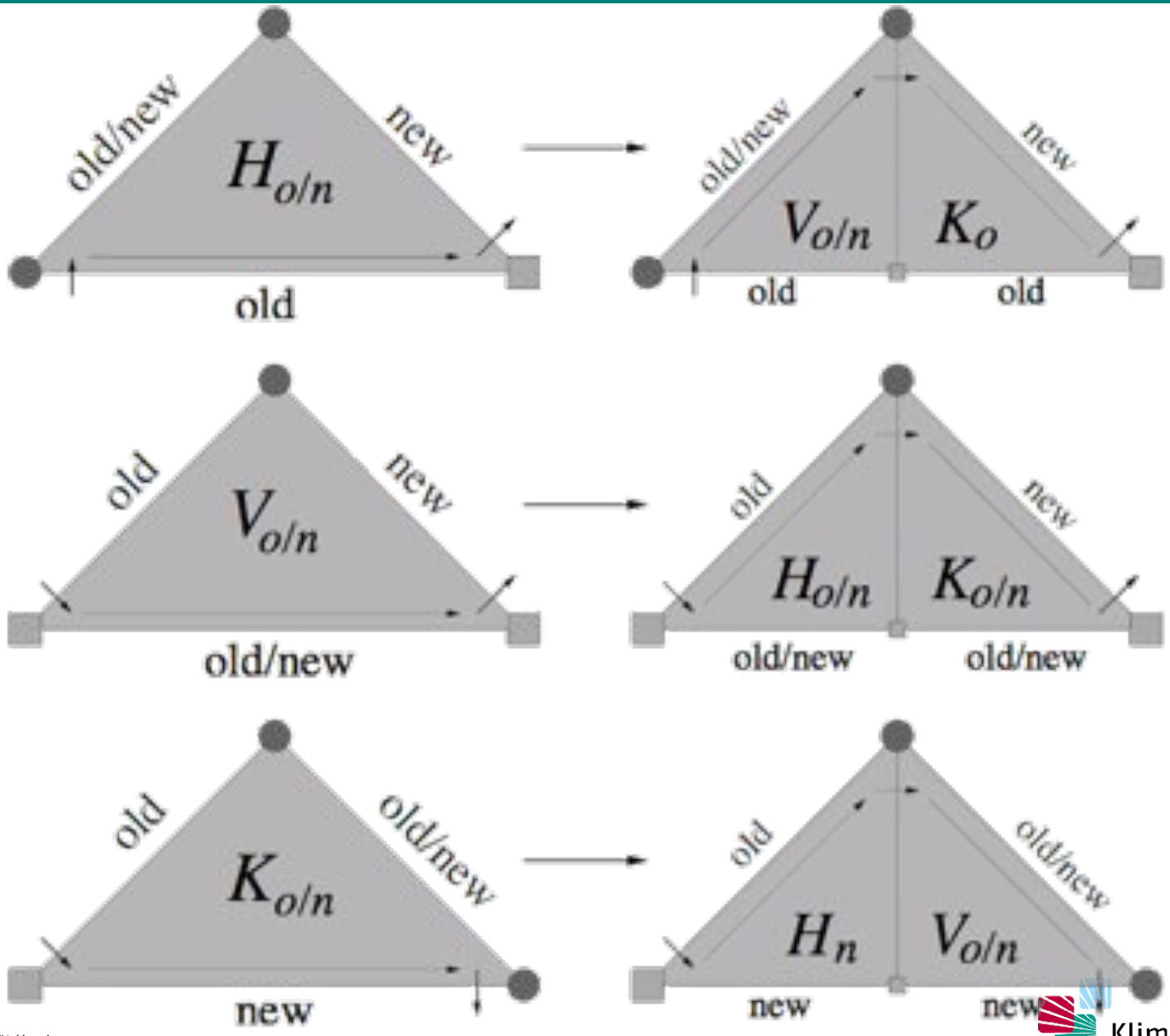
- Depth first traversal induces Sierpinski curve
- Element by element computation

Observation 3: Nodal computations

- not yet accessed
- just being computed
- waiting for further computation
- computations finished

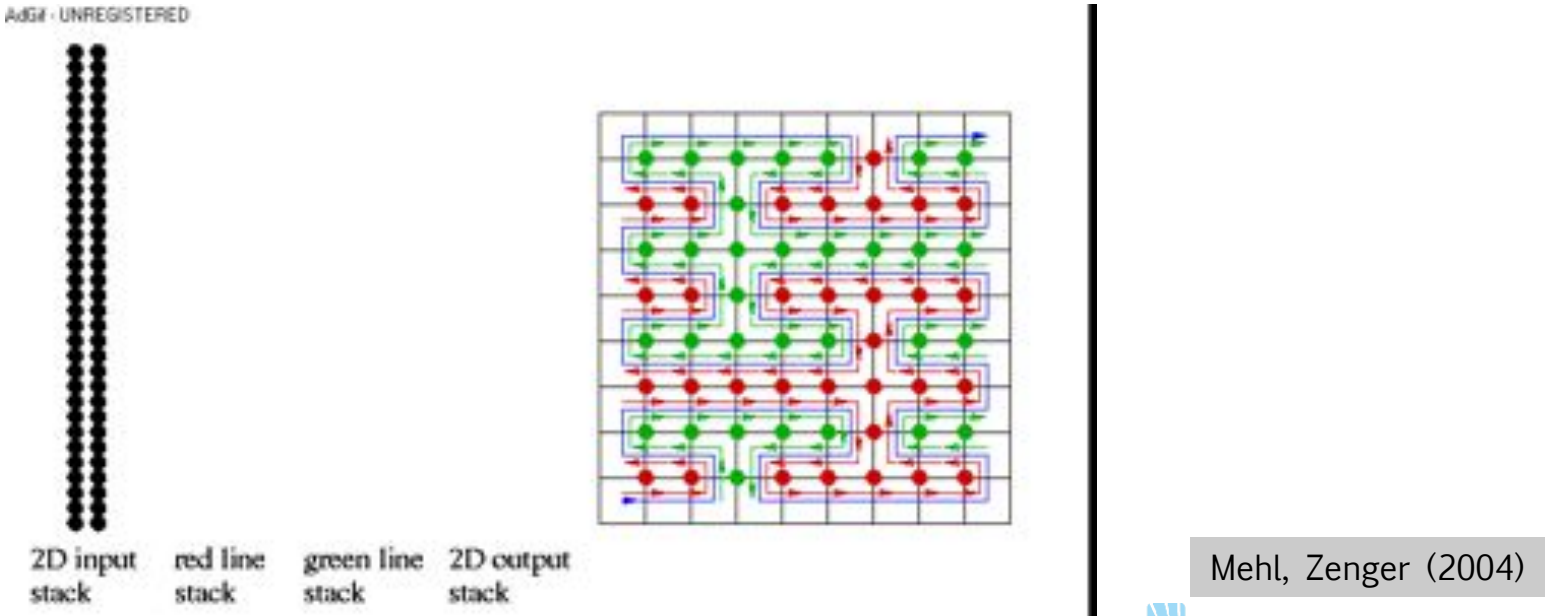


Observation 4: Tabulated element types



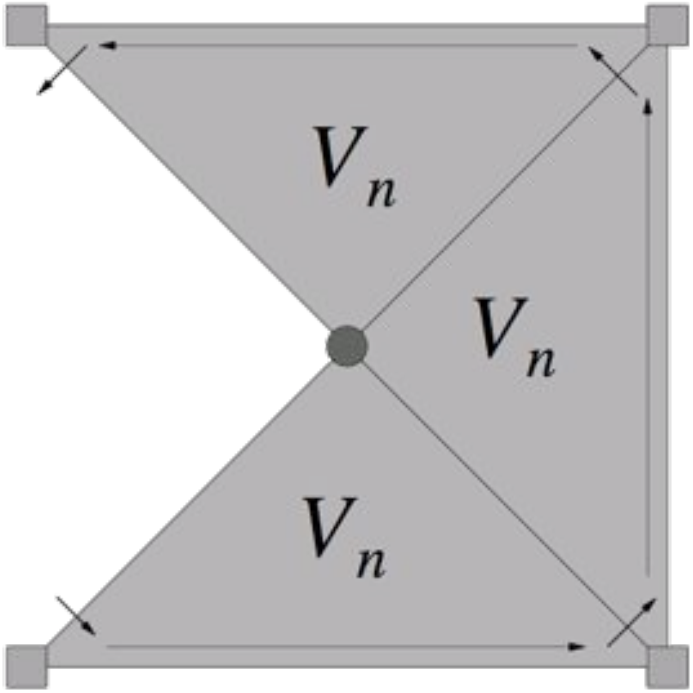
Idea

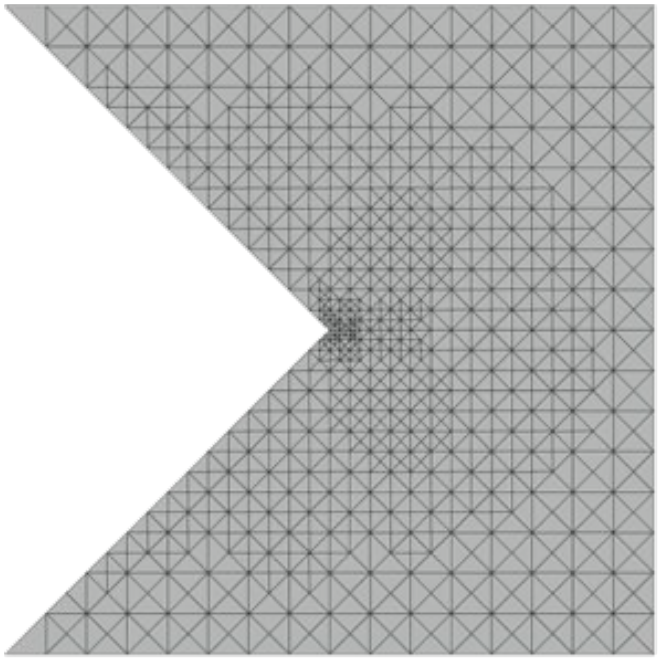
- Linearize grid structure by depth-first-traversal/Sierpinski curve
- Take element types for computing order
- Use heap data structures for storing intermediate values
- Use input/output stream



Mehl, Zenger (2004)

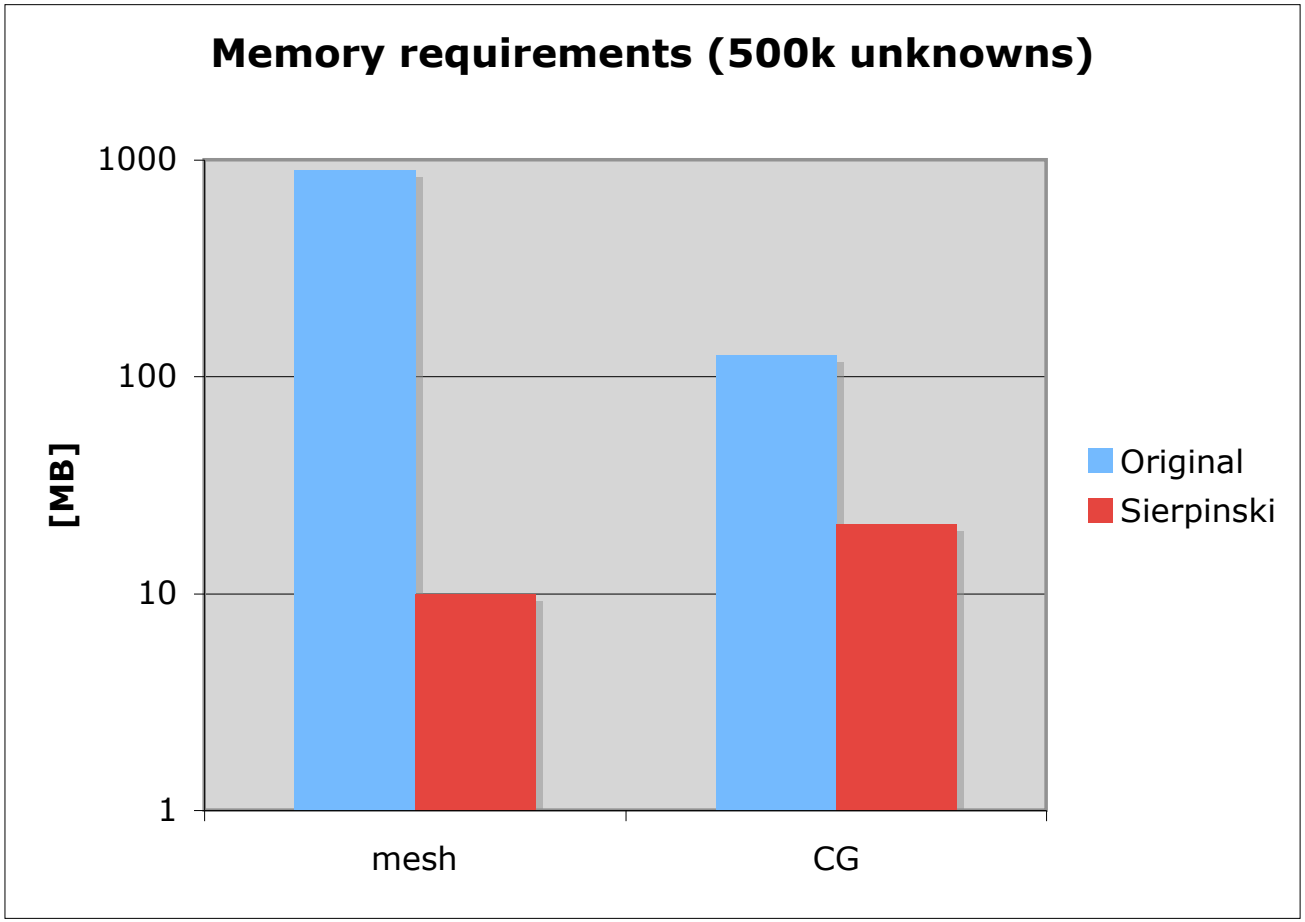
Preliminary result Reentrant corner





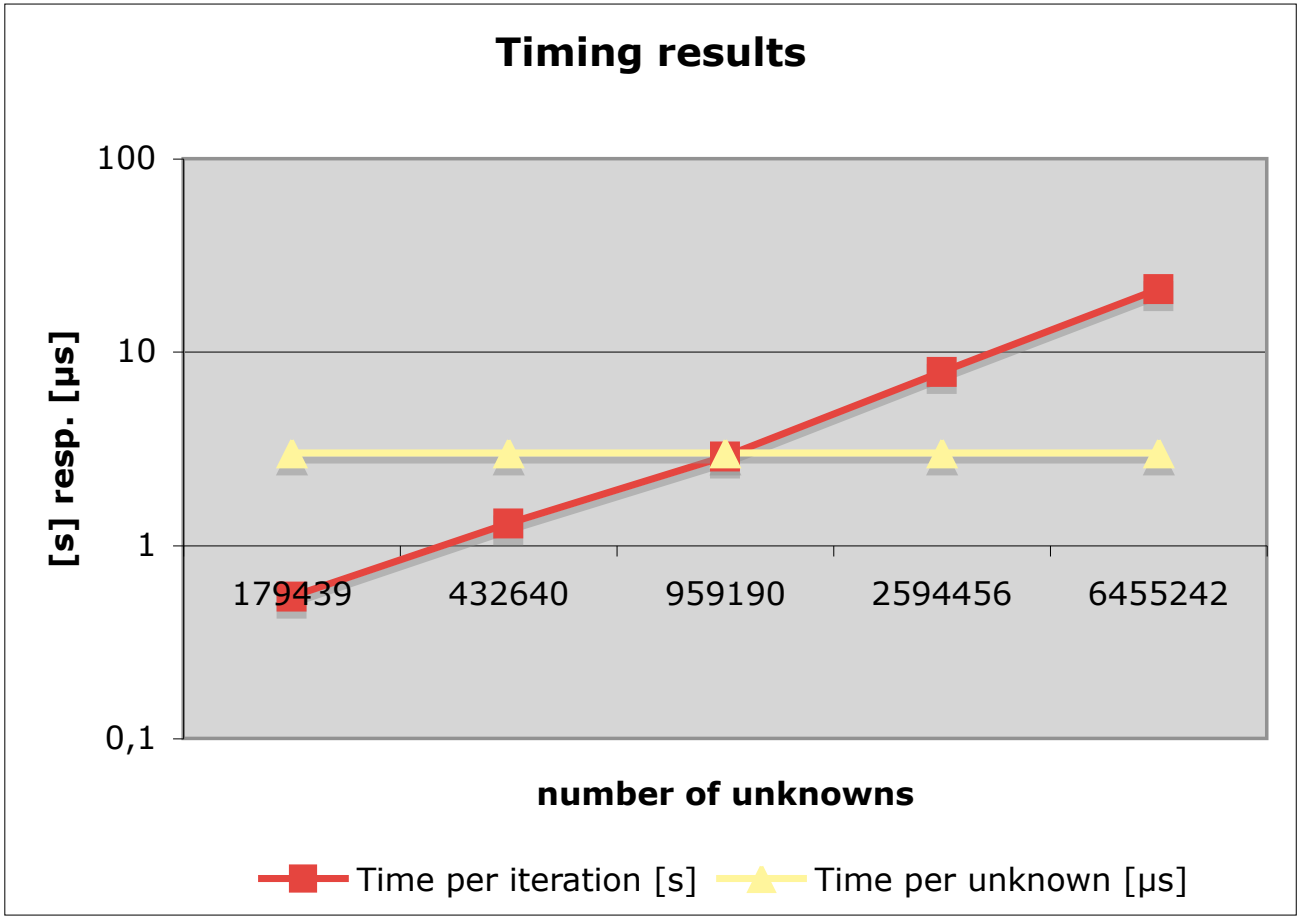
- Poisson‘s eq. with local refinement
- Solver: MG preconditioned CG
- Refinement criterion: hierarchical basis based (Deuflhard)

Preliminary result Reentrant corner



J.B., M. Bader (2009)

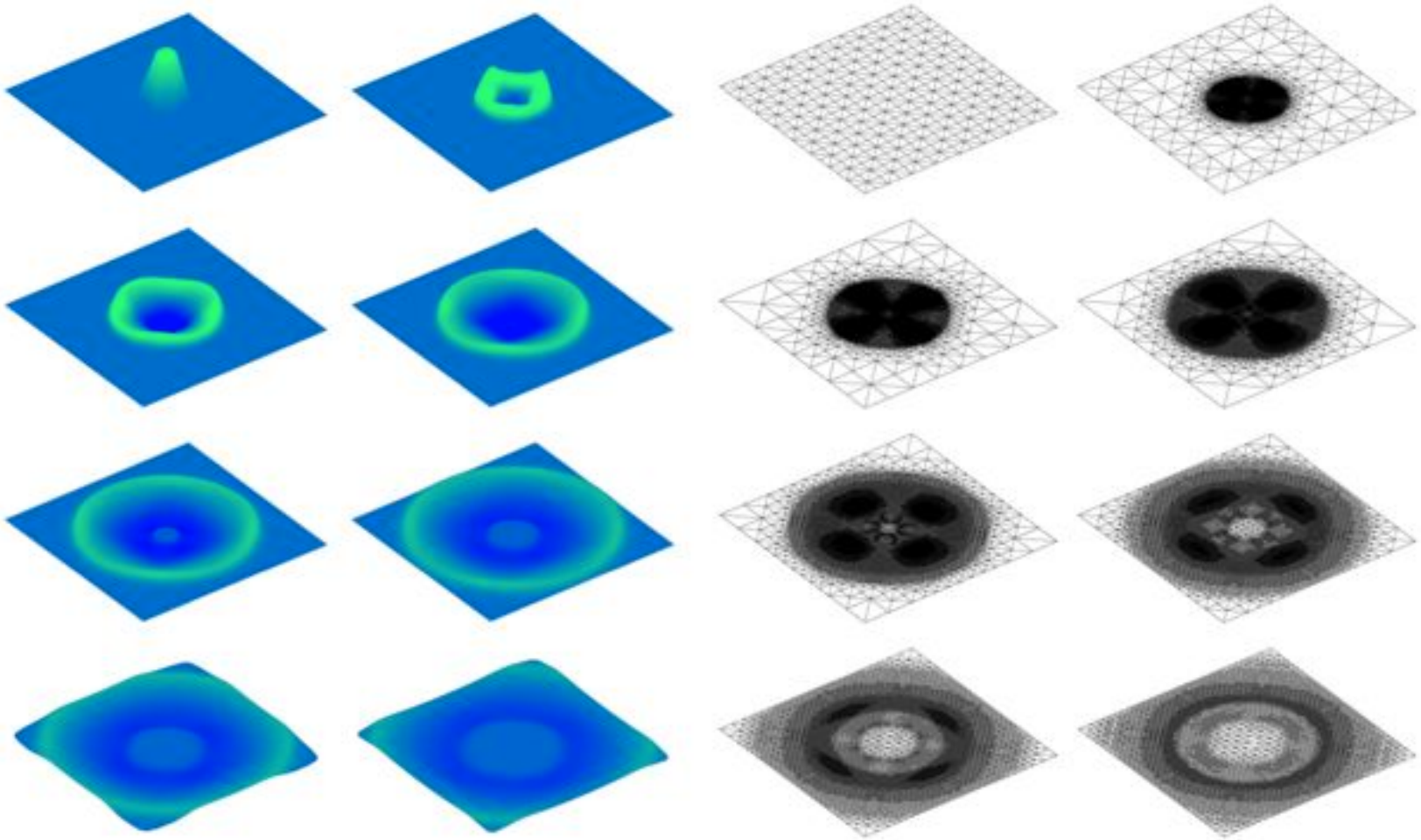
Preliminary result Reentrant corner



- 99% cache hits

J.B., M. Bader (2009)

Results for SWE



Bader et al. (2010)

Results for SWE

Memory requirements

level	number of elements	number of unknowns	memory [in MB] float	memory [in MB] double	bytes per element float	bytes per element double
16	131,072	393,216	7.7	14	60	112
18	524,288	1,572,864	27	53	54	106
20	2,097,152	6,291,456	106	210	53	105
22	8,388,608	25,165,824	422	838	53	105
16	131,072	1,179,648	11	22	88	176
18	524,288	4,718,592	43	85	86	170
20	2,097,152	18,874,368	170	338	85	169
22	8,388,608	75,497,472	678	1355	85	169

TABLE 4.1

Memory requirement for solving the shallow equations on uniformly refined grids. The data in the upper half was obtained when using constant ansatz functions for ξ, ξ_u, and ξ_v (3 unknowns per element); the results in the lower half are for linear ansatz functions (3×3 unknowns per element).

Bader et al. (2010)

Results for SWE

Runtime

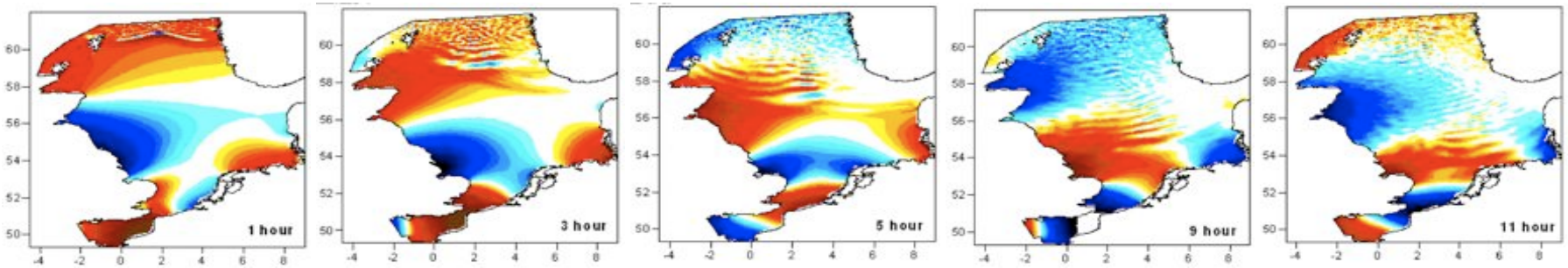
level	number of elements	number of unknowns	runtime [in s] total	runtime [in s] per element	throughput [in MB/s]
16	131,072	393,216	0.072	$5.49 \cdot 10^{-7}$	390
18	524,288	1,572,864	0.282	$5.38 \cdot 10^{-7}$	376
20	2,097,152	6,291,456	1.217	$5.80 \cdot 10^{-7}$	345
22	8,388,608	25,165,824	5.068	$6.04 \cdot 10^{-7}$	331
16	131,072	1,179,648	0.147	$1.12 \cdot 10^{-6}$	299
18	524,288	4,718,592	0.579	$1.10 \cdot 10^{-6}$	294
20	2,097,152	18,874,368	2.308	$1.10 \cdot 10^{-6}$	293
22	8,388,608	75,497,472	9.376	$1.12 \cdot 10^{-6}$	289

TABLE 4.2

Execution times to compute the shallow equations on uniformly refined grids using constant (upper half) and linear basis (lower half) functions in double precision. Given is the average runtime for one time step, the respective runtime per grid element, and the resulting memory bandwidth used.

Bader et al. (2010)

Future application – wave interaction and tsunamis



A. Androsov et al. (2010)

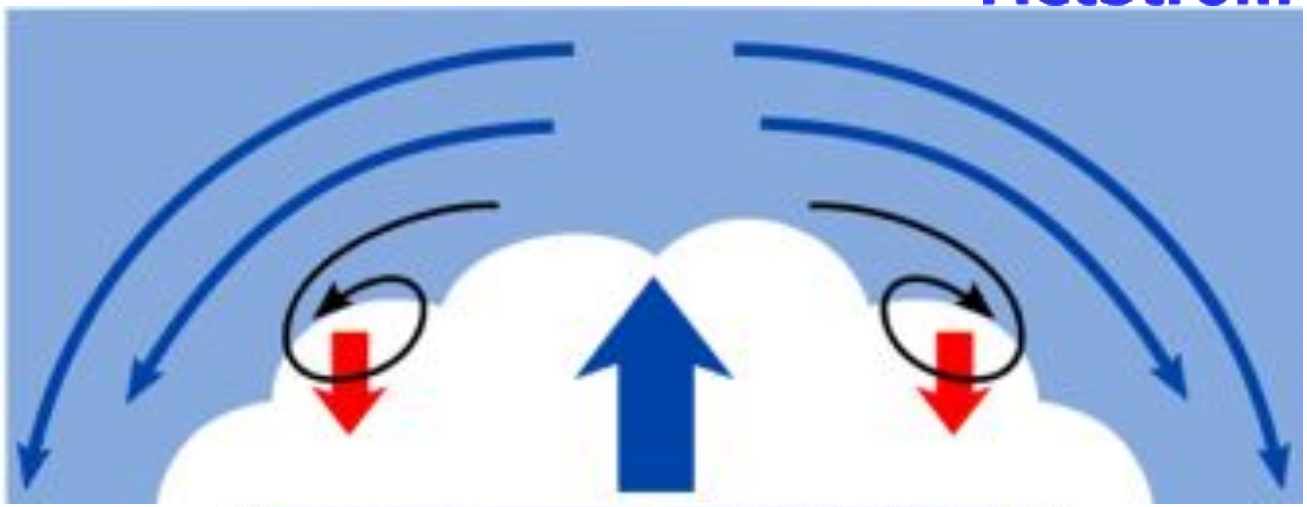
F. Fehsenfeld
M. Ehrhard
F. X. Giraldo

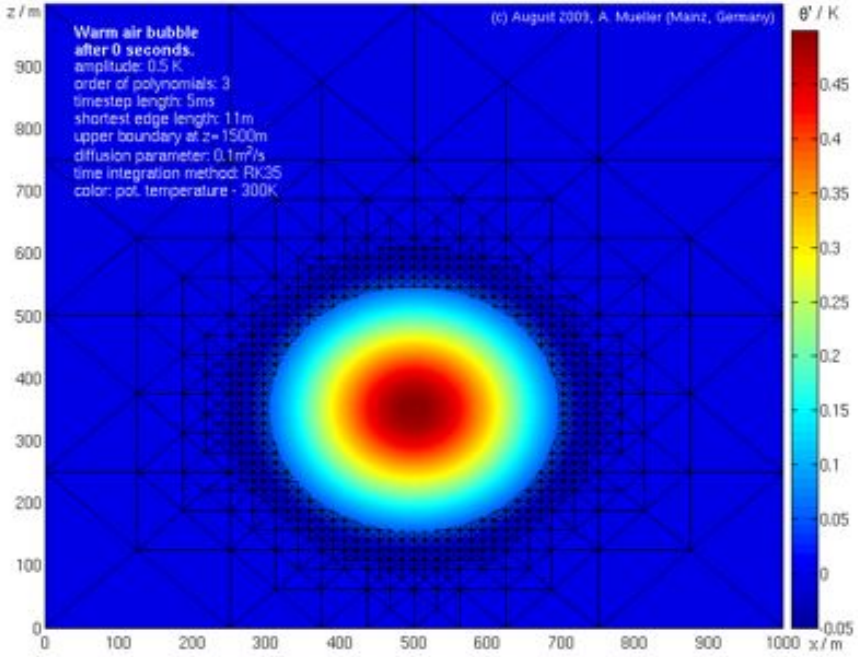
Universität Hamburg

KlimaCampus

Future application – cloud simulation

MetStröm





Müller (pers. comm. 2010)

Universität Hamburg

KlimaCampus

Conclusions

- Multiple scales need numerical representation
- Model for adaptive computation gain
- 5 Efficiency ingredients (meshing, refinement, programming, numerics, data management)
- 3 Hierarchies of optimization
- Radical approach with SFC
- Future applications