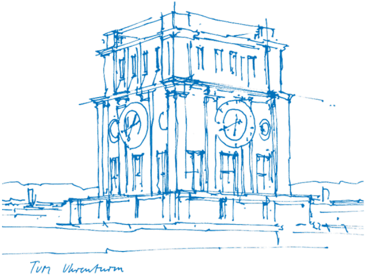
Avenues and Sightlines for High Performance Earthquake Simulation at Exascale

Oberseminar: Numerical methods in CSE

M. Bader (and many others!)
Technical University of Munich

Jan 8, 2026

This work has received funding from the European Union's Horizon 2020 research and innovation programme (grant agr. No 823844) and from the European High Performance Computing Joint Undertaking (grant agr. No 101093038).



Project Background

ChEESE/ChEESE-2P: → www.cheese2.eu

- CoE for exascale computing in the domain of Solid Earth
- **ChEESE**: 2018–2021, EU Horizon 2020
- **ChEESE-2P**: 2022–2026, EuroHPC JU

ScalaMIDA: (2024)

- Scalable Multi-Index Delayed Acceptance for non-linear earthquake modelling (with L. Seelinger and A.-A. Gabriel)
- funded as an **Inno4scale innovation study** (“Innovative Algorithms for Applications on European Exascale Supercomputers”; EuroHPC JU)

SeisSol: → www.seissol.org

- “big picture”: improve SeisSol for use within large workflows, esp. UQ
- focus for today: avenues for improving time to solution (on GPUs)
  → fused ensemble simulation, local time stepping

SeisSol – Tribute to Contributors

LMU Munich/UC San Diego – Geophysics:

- PI: **Alice-Agnes Gabriel**
- current: Fabian Kutschera, Jeremy Wong, Nico Schliwa, Thomas Ulrich
- former: Bo Li, Betsy Madden, Duo Li, Sara Aniko Wirp, Stephanie Wollherr, Taufiqurrahman, Zihua Niu

Technical University of Munich – HPC:

- current: **David Schneller, Vikas Kurapati**
- former: Ravil Dorozhinskii, Lukas Krenz, Sebastian Wolf; Carsten Uphoff; Alexander Breuer, Leonhard Rannabauer, Sebastian Rettenberger

Further/former members and contributors:

- Alexander Heinecke (Intel, TUM); Christian Pelties (LMU)
- Martin Käser (LMU), Michael Dumbser (Trento) and many more

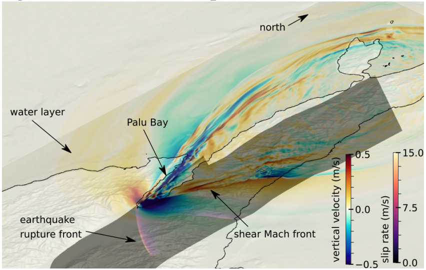
Part I

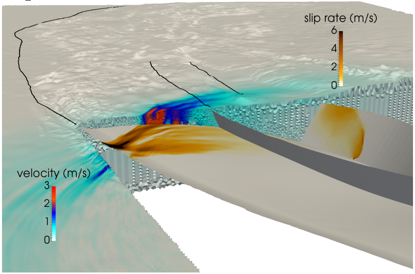
SeisSol – Extreme Scale Multi-Physics Earthquake Simulations

`http://www.seissol.org/` (all software open source)

Krenz et al. [3]: 3D acoustic-elastic coupling with gravity: the dynamics of the 2018 Palu, Sulawesi earthquake and tsunami.
Taufiqurrahman et al. [15]: Dynamics, interactions and delays of the 2019 Ridgecrest rupture sequence.
Uphoff et al. [6]: Extreme Scale Multi-Physics Simulations of the Tsunamigenic 2004 Sumatra Megathrust Earthquake.

Dynamic Rupture and Earthquake Simulation



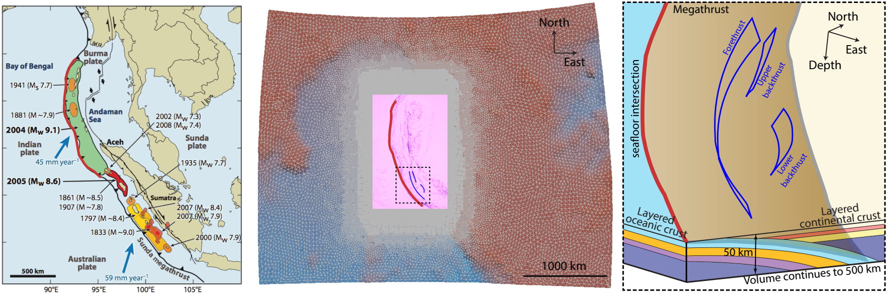


Dynamic rupture simulation with SeisSol: Sulawesi earthquake 2018 [3], Sumatra 2004 [6]

SeisSol – ADER-DG meets complex geometries: (www.seissol.org)

- adaptive tetrahedral meshes
  $\rightarrow$ complex geometries, heterogeneous media, multiphysics
- complicated fault systems with multiple branches
  $\rightarrow$ non-linear multiphysics dynamic rupture simulation
- ADER-DG: (adaptive) high-order discretisation in space and time
  $\rightarrow$ discontinuous Galerkin plus ADER (local) time stepping

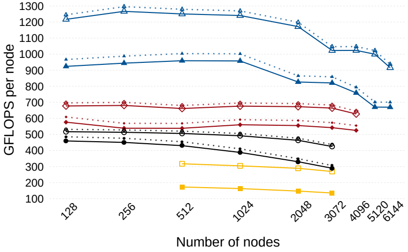
Earthquake-Tsunami Events – Sumatra 2004

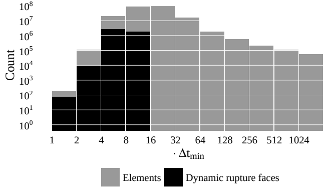


Domain, mesh and geometry of the Sumatra scenario (images from [6])

- multiscale: rupture extends of 1500 km, but happens on meter scale
- complex geometry: shallow angles in subduction zone; splay faults, topography, multiple material layers
- extremely long duration of earthquake: 500 s simulated time (over 3 Mio smallest time steps) → requires **adaptivity in space and time**

Sumatra Earthquake – HPC Challenges

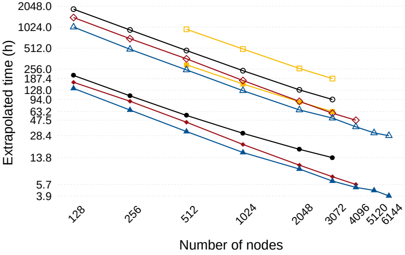


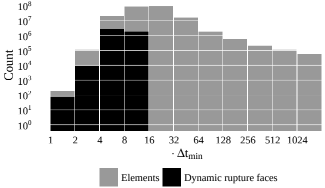


Sumatra: histogram of LTS clusters, extrapolated runtimes and parallel efficiency (results from [6])

- 221 million elements (111 billion degrees of freedom at order 6)
- 11 LTS clusters: “smallest” elements performed 3.3 Mio time steps
- dynamic rupture plus local time stepping with strong(!) scalability required
- CPU-centric: SuperMUC (‘M’) and Shaheen-II (‘S’) → Intel-“Haswell’; Cori (‘C’) → Intel-KNL

Sumatra Earthquake – HPC Challenges



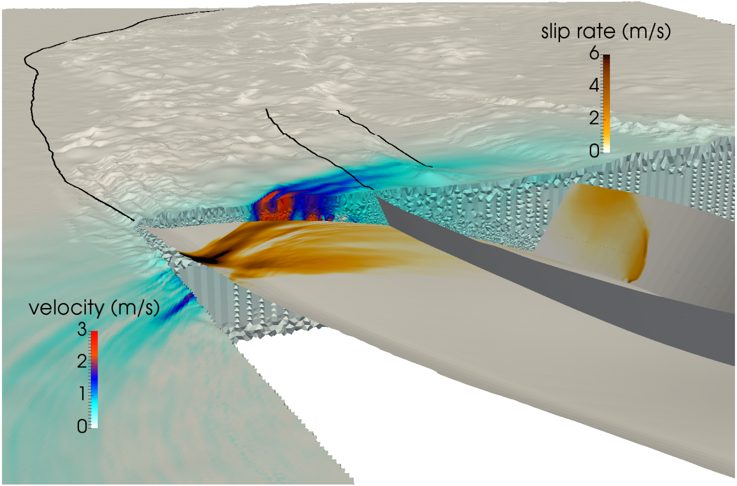


Sumatra: histogram of LTS clusters, extrapolated runtimes and parallel efficiency (results from [6])

- 221 million elements (111 billion degrees of freedom at order 6)
- 11 LTS clusters: “smallest” elements performed 3.3 Mio time steps
- dynamic rupture plus local time stepping with strong(!) scalability required
- CPU-centric: SuperMUC (‘M’) and Shaheen-II (‘S’) → Intel-“Haswell’; Cori (‘C’) → Intel-KNL

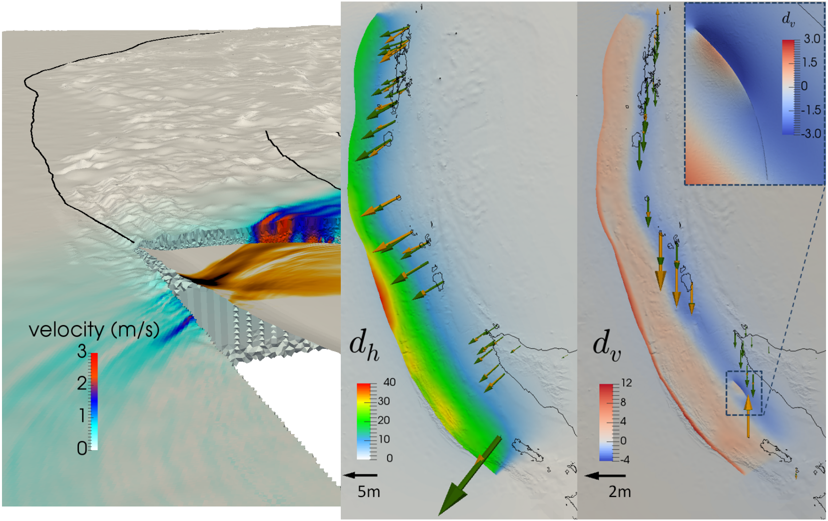
Sumatra 2004 – Results

Splay Fault Activation and Ocean Floor Displacements



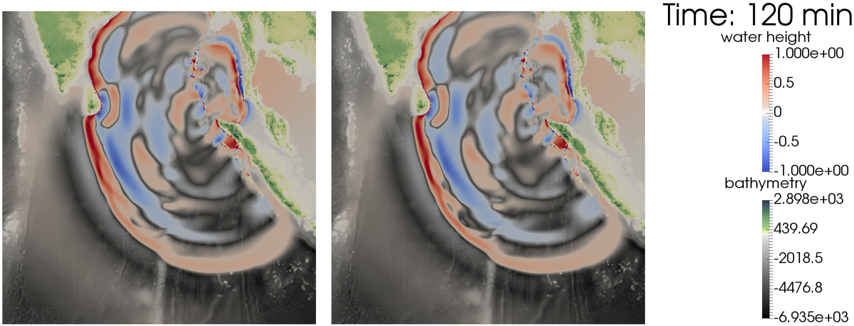
Sumatra 2004 – Results

Splay Fault Activation and Ocean Floor Displacements



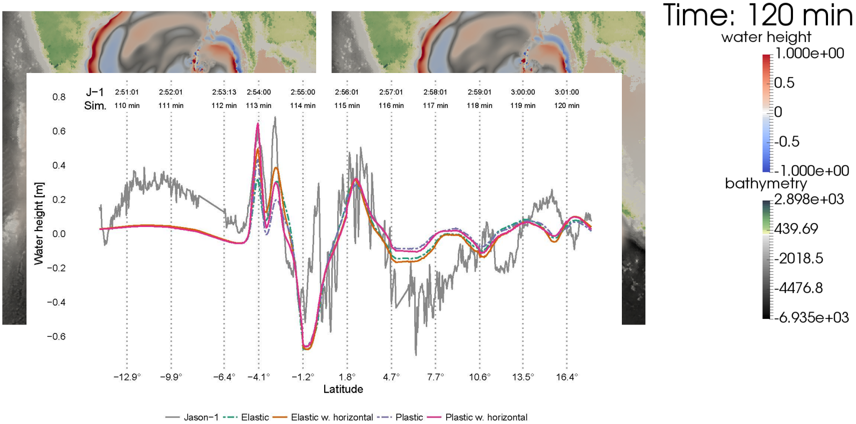
Sumatra 2004 – Impact of Earthquake Dynamics on Generated Tsunami

Purely elastic vs. off-fault plasticity (S. Wollherr [8], C. Uphoff):



Sumatra 2004 – Impact of Earthquake Dynamics on Generated Tsunami

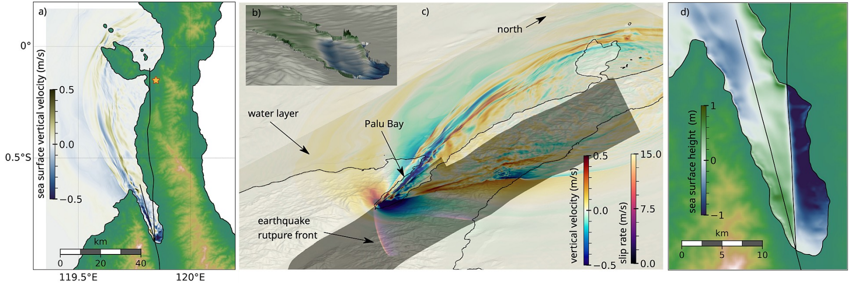
Purely elastic vs. off-fault plasticity (S. Wollherr [8], C. Uphoff):



(Comparison of sea surface height with altimeter data by Jason-1 satellite)

ChEESE Pilot: Earthquake–Tsunami Simulation

Fully coupled acoustic-elastic simulations of submarine earthquakes & tsunami generation

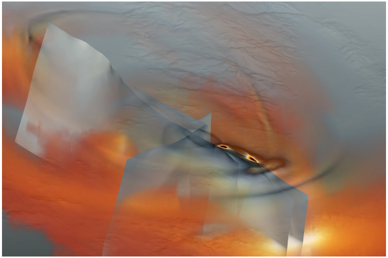


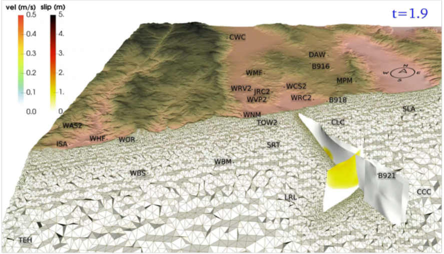
3D coupled elastic-acoustic simulation of the 2018 Sulawesi earthquake (Krenz et al. [3], SC21)

Example: 2018 Sulawesi Earthquake and local tsunami in Palu Bay

- project ASCETE: follow-up on linked modeling of earthquake (dynamic rupture) simulation and subsequent shallow-water simulation, e.g. [16]
- ChEESE: fully-coupled simulation of elastic (solid earth) and acoustic (ocean) layer → coverage of complex 3D bathymetry

Dynamic Rupture: 2019 Ridecrest Earthquakes





Dynamic rupture simulation of the 2019 Ridgecrest earthquakes [15] –
visualisation of seismic waves (left) and computational mesh (right)

Dynamics and interactions of the 2019 Ridgecrest rupture sequence [15]

- Mw 6.4 foreshock southwest of Searles Valley; subsequent Mw 7.1 mainshock in the Ridgecrest area
- rupture of a complex fault system with four major non-planar segments
- elasto-plastic materials using a Drucker-Prager yield criterion; rate-and-state friction law with strong velocity weakening

Part II

Code Generation in SeisSol: from Multiphysics via ADER-DG and Matrix Kernels to a Tensor DSL

Uphoff, Bader [5]: YATeTo – yet another tensor toolkit for discontinuous Galerkin methods and other applications. ACM TOMS 46(4), 2020.

Seismic Wave Propagation with SeisSol

Elastic Wave Equations: (velocity-stress formulation)

$$q_t + Aq_x + Bq_y + Cq_z = 0$$

$$\text{with} \quad q = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{13}, u, v, w)^T \in \mathbb{R}^9$$

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -\lambda - 2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\lambda & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\lambda & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu \\ -\rho^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda - 2\mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\rho^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0 & 0 \end{pmatrix}$$

- high order discontinuous Galerkin discretisation
- **ADER-DG**: high approximation order in space and time
- additional features: local time stepping, high accuracy of earthquake faulting (full frictional sliding)

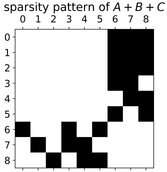
$\rightarrow$ Dumbser, Käser et al., e.g. [8]

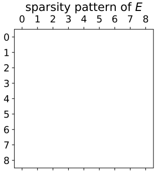
Seismic Wave Propagation with SeisSol

Elastic Wave Equations: (velocity-stress formulation)

$$q_t + Aq_x + Bq_y + Cq_z = 0$$

with $\quad q = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{13}, u, v, w)^T \in \mathbb{R}^9$





- high order discontinuous Galerkin discretisation
- **ADER-DG**: high approximation order in space and time
- additional features: local time stepping, high accuracy of earthquake faulting (full frictional sliding)

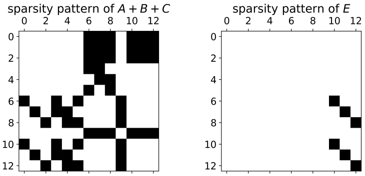
→ Dumbser, Käser et al., e.g. [8]

Seismic Wave Propagation with SeisSol

Poro-Elastic Wave Equations: (Wolf et al. [7])

$$q_t + Aq_x + Bq_y + Cq_z = Eq$$

with $q = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{13}, u, v, w, p, u_f, v_f, w_f)^T \in \mathbb{R}^{13}$



- high order discontinuous Galerkin discretisation
- **ADER-DG**: high approximation order in space and time
- additional features: local time stepping, high accuracy of earthquake faulting (full frictional sliding)

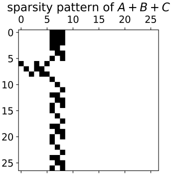
→ Dumbser, Käser et al., e.g. [8]

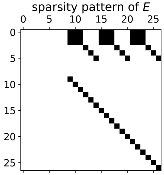
Seismic Wave Propagation with SeisSol

Visco-Elastic Wave Equations: (Uphoff et al. [4])

$$q_t + Aq_x + Bq_y + Cq_z = Eq$$

with $q = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{13}, u, v, w, \zeta_{11}^1, \ldots, \zeta_3^3)^T \in \mathbb{R}^{9+6L}$





- high order discontinuous Galerkin discretisation
- **ADER-DG**: high approximation order in space and time
- additional features: local time stepping, high accuracy of earthquake faulting (full frictional sliding)

→ Dumbser, Käser et al., e.g. [8]

Discontinous Galerkin Discretisation in SeisSol

Weak Form of the elastic wave equations:

$$\int_{T_k} q_t \phi_m \, d\vec{x} + \int_{T_k} (Aq_x + Bq_y + Cq_z)\phi_m \, d\vec{x} = 0$$

Apply chain rule and divergence theorem:

$$\int_{T_k} q_t \phi_m \, d\vec{x} = \int_{T_k} Aq(\phi_m)_x + Bq(\phi_m)_y + Cq(\phi_m)_z \, d\vec{x} - \int_{\partial T_k} F\phi_m \, d\vec{s}$$

Further choices:

- modal basis ϕ_m; ϕ_m orthogonal to obtain diagonal mass matrix
- hierachical (w.r.t polynomial degree) basis ϕ_m,
  leads to staircase pattern in stiffness matrices
- exact Riemann solver for linear flux F
- ADER time stepping ("Arbitrary high-order DERivatives")

SeisSol Numerics in a Nutshell – ADER-DG

Update scheme

$$Q_k^{n+1} = Q_k - \frac{|S_k|}{|J_k|} M^{-1} \left(\sum_{i=1}^{4} F^{-,i} I(t^n, t^{n+1}, Q_k^n) N_{k,i} A_k^+ N_{k,i}^{-1} + \sum_{i=1}^{4} F^{+,i,j,h} I(t^n, t^{n+1}, Q_{k(i)}^n) N_{k,i} A_{k(i)}^- N_{k,i}^{-1} \right)$$
$$+ M^{-1} K^{\xi} I(t^n, t^{n+1}, Q_k^n) A_k^*$$
$$+ M^{-1} K^{\eta} I(t^n, t^{n+1}, Q_k^n) B_k^*$$
$$+ M^{-1} K^{\zeta} I(t^n, t^{n+1}, Q_k^n) C_k^*$$

Cauchy Kovalewski

$$I(t^n, t^{n+1}, Q_k^n) = \sum_{j=0}^{J} \frac{(t^{n+1} - t^n)^{j+1}}{(j+1)!} \frac{\partial^j}{\partial t^j} Q_k(t^n)$$

$$(Q_k)_t = -M^{-1} \left((K^{\xi})^T Q_k A_k^* + (K^{\eta})^T Q_k B_k^* + (K^{\zeta})^T Q_k C_k^* \right)$$

YATeTo: “Yet Another Tensor Toolkit”

Code generation for small matrix and tensor chain products [5]:

1. Write down scheme in Einstein notation (with **“ensemble index” s**)

$$Q_{\mathbf{s}kp}^{\mu(\tau+1)} := Q_{\mathbf{s}kp}^{\mu\tau} + \hat{R}_{km}^{\sigma} f_{mn}^{h(\sigma,\mu)} R_{ln}^{j(\sigma,\mu)} \mathcal{I}_{\mathbf{s}lq}^{m(\sigma,\mu)} (A^{-})_{pq}^{\sigma}.$$

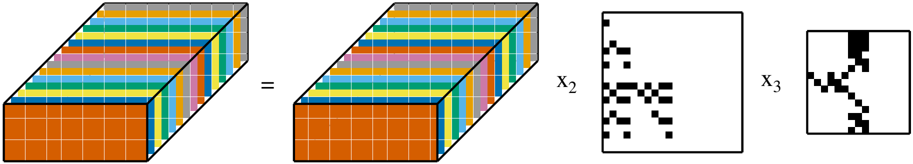
2. Transcription to YATeTo DSL:

```
Q = Tensor('Q', (nsims, nbf, nquant))
...
neighbourFlux = lambda h,j,i: Q['skp'] <= Q['skp'] +\
  Rhat[i]['km'] * f[h]['mn'] * R[j]['ln'] * I['slq'] * Am['pq']
g.addFamily('neighboringFlux', simpleParameterSpace(3,4,4),\
  neighbourFlux)
```

3. Instantiate generated kernel object, set pointers to data, call
`krnl.execute(h, j, i);`

Sparse vs. Dense (Small) Matrix Operations

Determine equivalent sparsity patterns: (Uphoff, [4,5])

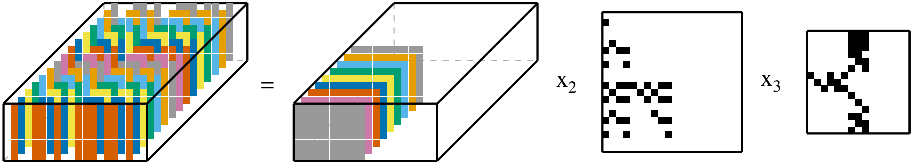


Code generation for chains of matrix/tensor multiplications: (Uphoff, [5])

- optimisation of chain products: find best-possible execution order (strength reduction), memory layout, index permutations, etc.
- pass binary operations to hardware-optimized libraries, e.g. libxsmm [9] for small BLAS; **PSpaMM** (part of SeisSol) for sparse matrix/tensor
- auto-tuning in SeisSol to chose dense/sparse/blocked-sparse matrices

Sparse vs. Dense (Small) Matrix Operations

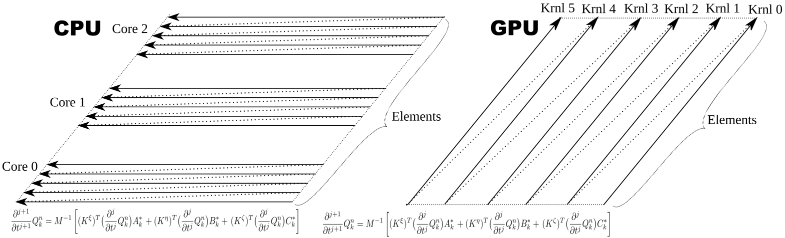
Determine equivalent sparsity patterns: (Uphoff, [4,5])



Code generation for chains of matrix/tensor multiplications: (Uphoff, [5])

- optimisation of chain products: find best-possible execution order (strength reduction), memory layout, index permutations, etc.
- pass binary operations to hardware-optimized libraries, e.g. libxsmm [9] for small BLAS; **PSpaMM** (part of SeisSol) for sparse matrix/tensor
- auto-tuning in SeisSol to chose dense/sparse/blocked-sparse matrices

Parallelism on CPUs vs. GPUs



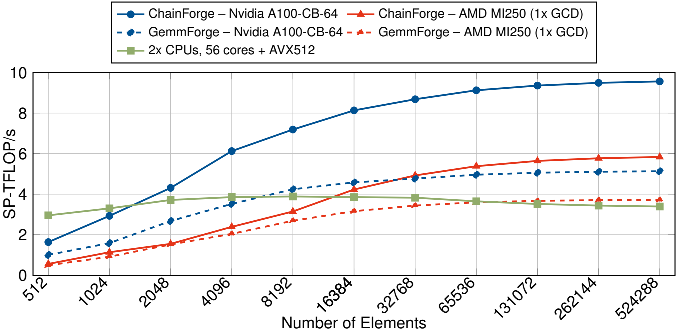
CPU

- loop over all elements
  $\rightarrow$ OpenMP-type parallel for
- sequence of small-BLAS operations on element-local data
- high arithmetic intensity and cache-friendly

GPU (Dorozhinskii et al. [2])

- sequence of kernel calls: “parallel for” on all elements of the cluster
- thus: batched smallGEMMs or chains of small GEMMs
- fuse generated batched kernels to better exploit shared memory

SeisSol Performance on GPUs



Influence of the size of GTS/LTS clusters on CPU and GPU performance.

- strong influence of arithmetic intensity on GPUs: code generator **ChainForge** fuses chains of batched BLAS kernels [2]
- kernels stay memory-bound on GPUs
- on GPUs: massive loss of performance for less than 16k elements

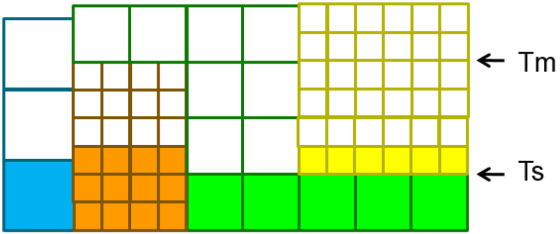
Part III

Adaptivity in Space and Time: Local Time Stepping

Breuer et al. [1]: Petascale local time stepping for the ADER-DG Finite Element method.

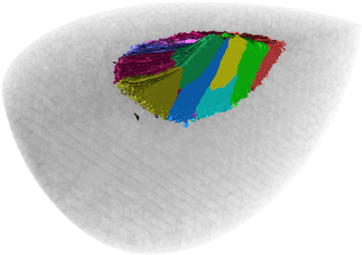
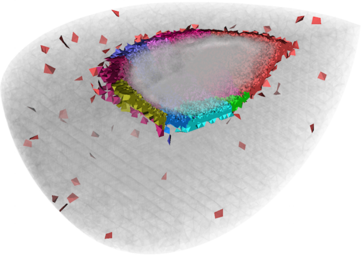
Uphoff et al. [6]: Extreme Scale Multi-Physics Simulations of the Tsunamigenic 2004 Sumatra Megathrust Earthquake.

Cluster-Oriented Local Time Stepping



- “time steps” to update grid cells ⇝ time step size depends on size of grid cell and speed of seismic waves in the cell
- “global time stepping”: advance at smallest allowed time step τ_{min}
- “local time stepping”: update cells with adaptive time step sizes
- ADER-DG: any cell can advance at its individual (allowed) time step, **but** needs to obey to an “evolve condition”

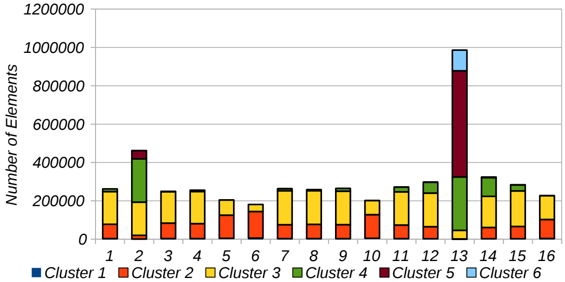
Cluster-Oriented Local Time Stepping

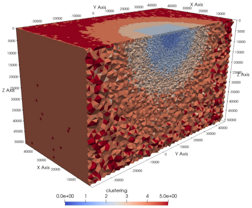



"Cluster-oriented multi-rate" local time stepping: (Breuer et al. [1]):

- combine grid cells with similar time step size into **clusters**
- enforce integer ratio (power of 2 or 3) between time step sizes (**multi-rate**), i.e. $\Delta t \in [\tau_{\text{min}}, 2\tau_{\text{min}}), [2\tau_{\text{min}}, 4\tau_{\text{min}}), \ldots$
- advance clusters with uniform time step size
  $\leadsto$ challenging parallelisation and parallel load balancing

Scalability on GPUs – Partitioning for LTS

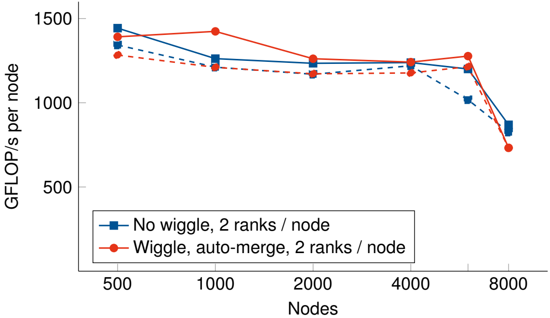


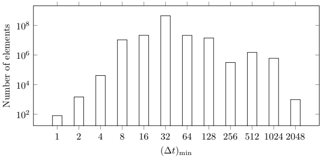


Histogram of elements per cluster for each MPI process (16 MPI ranks)

- local time stepping leads to different update rate for each element
- graph partitioning (ParMetis) to balance computational load per partition
  ⇝ may lead to imbalance in number of elements per node
- on GPU: partition no longer fits onto GPU memory
  ⇝ forces us to use smaller partitions than desired w.r.t. performance

LTS: Strong Scaling and Performance



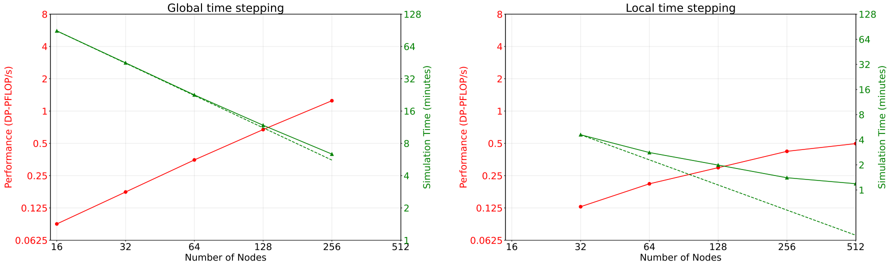


Strong scaling on Frontera for the Palu scenario (mesh with 518 million elements)

Scalability of Palu-Sulawesi Senario on up to 8,000 nodes of Frontera

- nearly 1.5 TFLOP/s per node ($\sim$40% of peak, using dense GEMMs $\rightsquigarrow$ 20% peak in terms of “productive” flop/s)
- substantial impact of local-time-stepping algorithm: “wiggle factor” $\rightsquigarrow$ start clusters from $\omega \cdot \Delta t_{\mathrm{min}}$ instead of Δt_{min}

LTS: Scaling and Performance on Frontier (GPU)

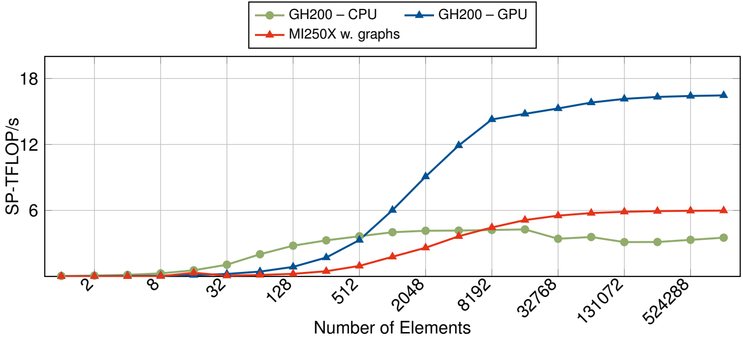


Strong scaling and performance on Frontier for the Kahramanmaras scenario (mesh with 175 million elements) – comparing global (left) and local(right) time stepping.

Performance vs. time to solution of LTS on Frontier

- on 16–512 nodes (4 AMD Instinct MI250X GPUs per node)
- $> 2\times$ performance advantage of GTS
- $\sim 5\times$ improvement (on 32 nodes) in solution time due to LTS

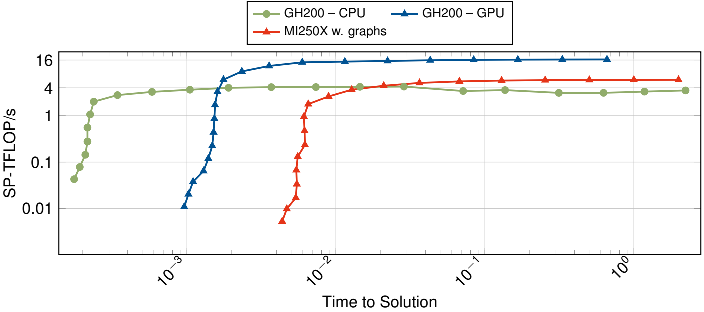
Revisited: SeisSol Performance on GPUs



Influence of the size of GTS/LTS clusters on performance: CPU vs. GPU of Grace Hopper superchip; AMD MI 250X

- strong influence of arithmetic intensity on GPUs:
  massive loss of performance for less than 8–32k elements

Revisited: SeisSol Performance on GPUs



Influence of the size of GTS/LTS clusters on performance: CPU vs. GPU of Grace Hopper superchip; AMD MI 250X

- strong influence of arithmetic intensity on GPUs:
  massive loss of performance for less than 8–32k elements
- reasons: launch overhead of GPU kernels, lack of parallelism

Summary: LTS Challenges

Partitioning

- relies on graph partitioning (ParMetis) for weighted graphs
- node weights: computational effort per element depends on mesh, size of clusters and even on size of generated partitions
- edge weights to reflect communication between clusters
- multi-constraint partitioning: need to fit partitions into (esp. GPU) memory

LTS Algorithm and Implementation

- for small clusters: lack of parallelism is a fundamental limitation
  $\rightsquigarrow$ use "part of a GPU" (or CPU instead of GPU)?
- impact on clustering: "wiggle effect", can/should small clusters be merged?
- are clusters really such a good idea for GPUs?

Part IV

Fused Simulations and UQ Workflows

Kruse, Niu, Wolf, Lykkegaard, Bader, **Gabriel**, **Seelinger** [10]:
Scalable Bayesian Inference of Large Simulations via Asynchronous Prefetching Multilevel Delayed Acceptance, PASC25

Fused Ensemble Simulations in SeisSol

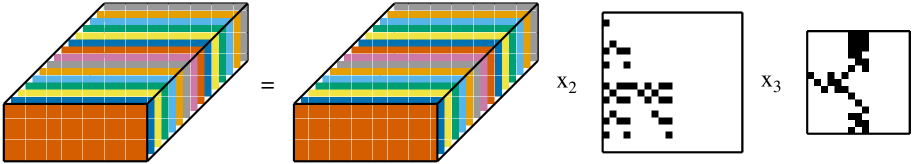
"Fused Ensemble Simulations":

- combine a set of simulations into one fused simulation run
- run "in parallel": typically requires identical mesh, physics, numerical scheme, etc.
- but: different source terms, parameters, etc.
- targeting workflows in uncertainty quantification, hazard assessment, inverse problems, etc.

Avenues (in SeisSol!) for Speedup with Fused Simulations:

- code generation for element-local sparse/dense tensor/matrix operations
  $\rightarrow$ reduce overhead in sparse operations
- local time stepping
  $\rightarrow$ better amortize overhead and performance loss for small clusters
- possibly further savings in I/O, pre-/postprocessing, …?

Code Generation for Chains of Tensor Operations

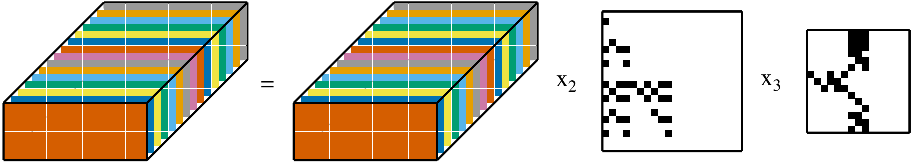


YATeTo generates sequences of binary GEMM/tensor operations:

(Uphoff, [5])

- optimisation of chain products: find best-possible execution order (strength reduction), memory layout, index permutations, etc.
- pass binary operations to hardware-optimized libraries, e.g. libxsmm [9] for small BLAS; **PSpaMM** (part of SeisSol) for sparse matrix/tensor

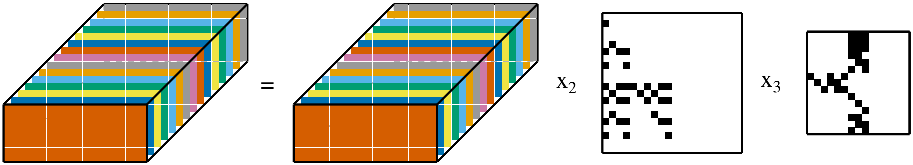
Code Generation for Chains of Tensor Operations



Single simulations: sequence of binary sparse GEMMs

- based on full unrolling of element operations → omit zero operations (and elements)
- dense-sparse: padding of dense matrix to SIMD lengths (column-major)
- sparse-dense: “block sparse” → enforce SIMD operations with block padding and masking

Code Generation for Chains of Tensor Operations



Fused simulations: sequence of binary dense-sparse operations

- tensor of quantities → fused simulations as fastest-running index
- dense-sparse: no dense padding, **if ...**
- sparse-dense: no SIMD padding/“block sparsity”, **if ...**
- **IF: number of fused simulations determined by SIMD length**

Example: Parallelized Multi-level MCMC

MLMCMC – Multilevel Markov Chain Monte Carlo:

- solve inverse UQ problems: compute likelihood that parameters explain given measurements
- Markov Chain Monte Carlo (MCMC) type methods: only require forward model, but very high computational cost (“samples” = model evaluations)
- Multilevel MCMC: exploit models of different resolution
  $\rightarrow$ run most model evaluations on “cheap” models
- **parallelisation of MLMCMC**: multiple chains can be run in parallel, multilevel telescoping sum evaluated in parallel [12, e.g.]

MLDA – Multilevel Delayed Acceptance: (Lykkegaard et al. [11]

- final sample of a coarser-level subchain used as a proposal for next finer level
- **Kruse et al. [10]**: parallel efficiency of many chains diminishes (due to burn-in, mixing); thus: add within-chain parallelization through prefetching

Parameter Inversion – Plastic Deformation

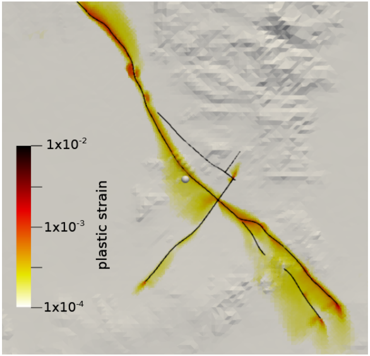
Dynamic rupture earthquake simulation:

- 2019 Mw 6.4 Searles Valley foreshock and Mw 7.1 Ridgecrest mainshock [15]
- With off-fault plasticity in SeisSol (Wollherr et al. [8]):
  determine friction angle and plastic cohesion parameter (Drucker-Prager yield criterion)

3-Level MLDA with prefetching:

- surrogate model plus two SeisSol levels: discretisation order 3 and 4
- 440 accepted models in MLDA chains, 1112 order-3 and 525 order-4 SeisSol simulations

Off-fault non-linear damage:



Parameter Inversion – Plastic Deformation

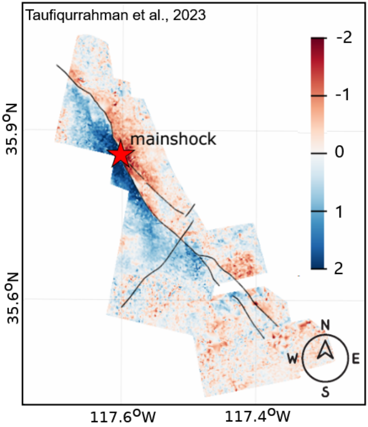
Dynamic rupture earthquake simulation:

- 2019 Mw 6.4 Searles Valley foreshock and Mw 7.1 Ridgecrest mainshock [15]
- With off-fault plasticity in SeisSol (Wollherr et al. [8]): determine friction angle and plastic cohesion parameter (Drucker-Prager yield criterion)

3-Level MLDA with prefetching:

- surrogate model plus two SeisSol levels: discretisation order 3 and 4
- 440 accepted models in MLDA chains, 1112 order-3 and 525 order-4 SeisSol simulations

Surface displacement data:



UM-Bridge: UQ and Modeling Bridge

L. Seelinger, V. Cheng-Seelinger, A. Davis, M. Parno, A. Reinarz [13]

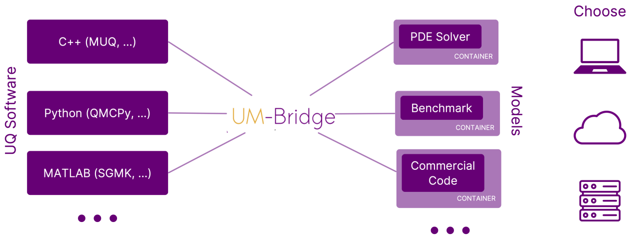


Image from: `https://um-bridge-benchmarks.readthedocs.io/`

“Democratizing Uncertainty Quantification:” [14]

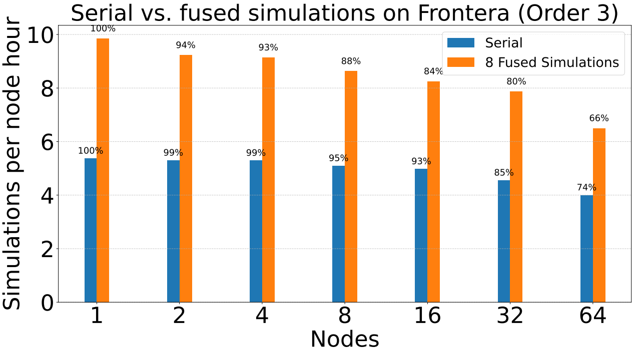
- high-level abstraction and software protocol
- connects versatile UQ algorithms to numerical models – from prototype to HPC scale
- released on `https://github.com/um-bridge`

Fused Simulations in UM-Bridge Workflows

Realising fused ensemble simulations:

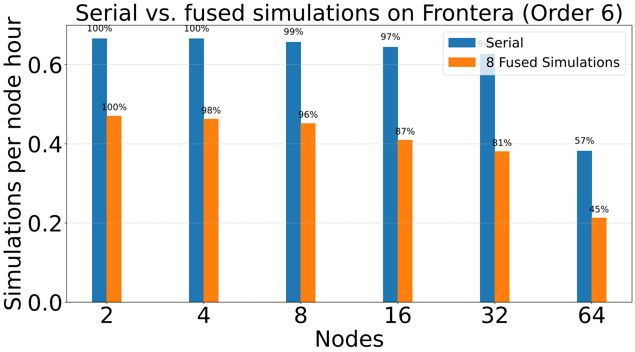
1. UM-Bridge client sends simulation parameters to a **batcher**
2. batcher collects the simulation parameters, batches them as per the configuration and sends the vector to the UM-Brige server
3. UM-Bridge HPC loadbalancer receives request and spans SLURM jobs
4. **UM-Bridge wrapper** for SeisSol processes the request:
   produce input files, run SeisSol, evaluate misfits for fused simulations
5. resulting misfit values are sent back to the batcher
6. batcher disaggregates the results and returns them to UM-Bridge client

Fused Ensemble Simulations – TPV13 Benchmark



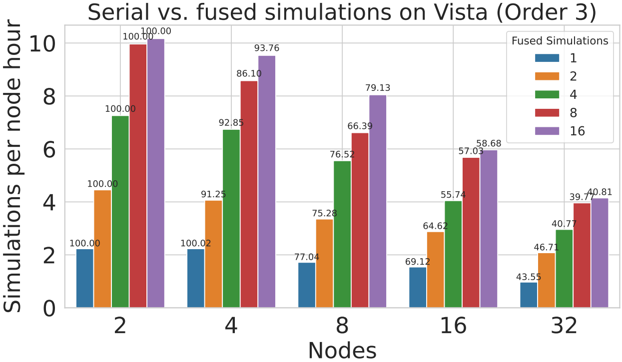
- TPV-13 benchmark with ≈1.4 million elements (small scenario)
- **Frontera** supercomputer (TACC): Intel Xeon Platinum 8280 (Cascade Lake, 28 cores, 2.7 GHz, 32 KB L1)
- discretisation order 3 vs. 6

Fused Ensemble Simulations – TPV13 Benchmark



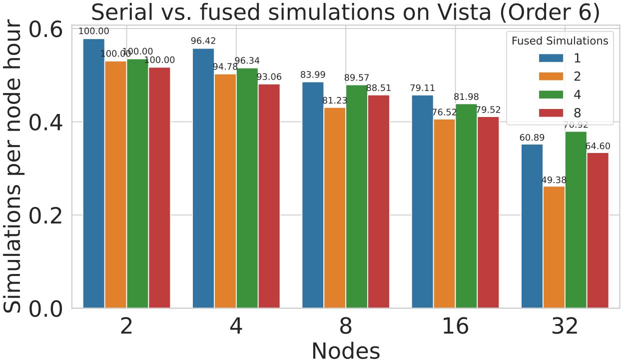
- TPV-13 benchmark with ≈1.4 million elements (small scenario)
- **Frontera** supercomputer (TACC): Intel Xeon Platinum 8280 (Cascade Lake, 28 cores, 2.7 GHz, 32 KB L1)
- discretisation order 3 vs. 6

Fused Ensemble Simulations – TPV13 Benchmark



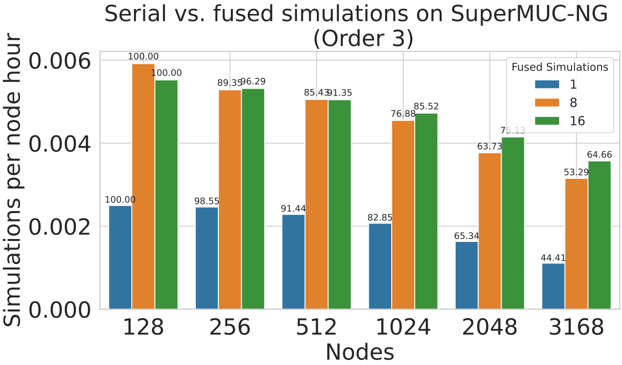
- TPV-13 benchmark with ≈1.4 million elements (small scenario)
- **Vista** supercomputer (TACC): NVIDIA Grace-Grace Superchip (2×72 cores, 3.4 GHz, 64 KB L1)
- discretisation order 3 vs. 6

Fused Ensemble Simulations – TPV13 Benchmark



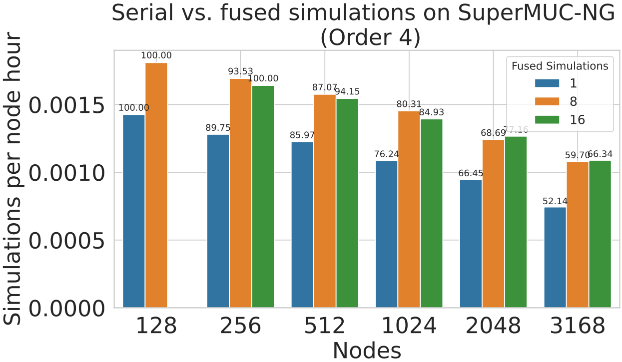
- TPV-13 benchmark with ≈1.4 million elements (small scenario)
- **Vista** supercomputer (TACC): NVIDIA Grace-Grace Superchip (2×72 cores, 3.4 GHz, 64 KB L1)
- discretisation order 3 vs. 6

Scaling Study: 2023 Kahramanmaras Earthquake



- 2023 Kahramanmaras earthquake doublet, with ≈175 million elements
- **SuperMUC-NG** supercomputer (LRZ): Intel Xeon Platinum 8147 (Skylake, 24 cores, 2.7 GHz, 32 KB L1)
- discretisation order 3 vs. 4 – higher HW-FLOP/s for serial runs at order 4

Scaling Study: 2023 Kahramanmaras Earthquake



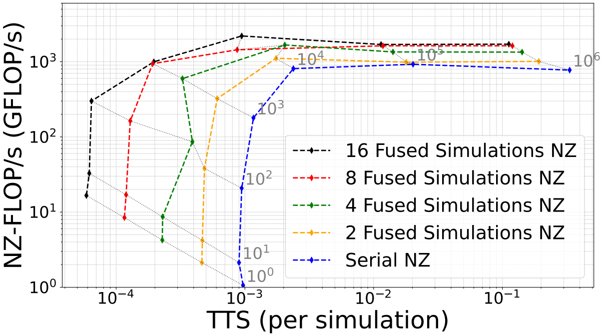
- 2023 Kahramanmaras earthquake doublet, with ≈175 million elements
- **SuperMUC-NG** supercomputer (LRZ): Intel Xeon Platinum 8147 (Skylake, 24 cores, 2.7 GHz, 32 KB L1)
- discretisation order 3 vs. 4 – higher HW-FLOP/s for serial runs at order 4

Parameter Inversion:

Metric	Fused	Serial
Simulation jobs for order 3	48	342
Actual parameter runs for order 3	384 (48 × 8)	342
Simulation jobs for order 4	72	72
Node hours for order 3	645	863
Node hours for order 4	415	437
Node hours in total	**1060**	**1300**

- MLDA workflow for 2019 Searles Valley earthquake (Ridgecrest), 4 million elements
- **SuperMUC-NG** supercomputer (LRZ): Intel Xeon Platinum 8147 (Skylake, 24 cores, 2.7 GHz, 32 KB L1)
- Parallelisation over parallel chains of MLDA

Compare: Fused Ensembles on Vista



NZ-FLOP/s for increasing cluster size on Vista

- almost equally strong “ramp up” of performance (due to many-core parallelism)
- kernel launch overhead amortised over fused simulations
- 14× speedup for 16 fused simulation on cluster of 100 elements!

Fused Simulations for Large Workflows

Summary and Next Steps:

- ensemble simulations, partially supported by **code generators**
- more “productive” FLOP/s (sparsity, padding)
- support for tensor instructions work-in-progress for GPUs
- common challenge for high order: cache, shared memory, etc.
- strong performance impact of **local time stepping**
- in general: particularly attractive for low-order simulations
  ($\rightarrow$ bulk of simulations in multi-level UQ workflows)

Acknowledgements: special thanks go to ...

- the entire SeisSol team and all former contributors
- for supercomputing support and CPU time: TACC (Frontera, Vista), LRZ (SuperMUC-NG), SCEC and DOE-INCITE (Frontier), CINECA (Leonardo), CSC (LUMI); EuroHPC JU

SeisSol Publications

[1] A. Breuer, A. Heinecke, M. Bader: Petascale local time stepping for the ADER-DG Finite Element method. Proc. IPDPS16.

[2] R. Dorozhinskii, G. Brito Gadeschi, M. Bader: *Fused GEMMs towards an efficient GPU implementation of the ADER-DG method in SeisSol.* Concurr. & Comput.: Pract. & Exp., 2024.

[3] L. Krenz, C. Uphoff, T. Ulrich, A.-A. Gabriel, L.S. Abrahams, E.M. Dunham, M. Bader: *3D acoustic-elastic coupling with gravity: the dynamics of the 2018 Palu, Sulawesi earthquake and tsunami.* SC21.

[4] C. Uphoff, M. Bader: *Generating high performance matrix kernels for earthquake simulations with viscoelastic attenuation*. 2016 Int. Conf. HPC & Simulation (HPCS 2016).

[5] C. Uphoff, M. Bader: *Yet Another Tensor Toolbox for discontinuous Galerkin methods and other applications*. ACM TOMS 46(4). `https://doi.org/10.1145/3406835`.

[6] C. Uphoff, S. Rettenberger, M. Bader, E. H. Madden, T. Ulrich, S. Wollherr and A.-A. Gabriel: *Extreme Scale Multi-Physics Simulations of the Tsunamigenic 2004 Sumatra Megathrust Earthquake.* SC17 Best Paper Award.

[7] S. Wolf, M. Galis, C. Uphoff, A.A. Gabriel, P. Moczo, D. Gregor, M. Bader: *An efficient ADER-DG local time stepping scheme for 3D HPC simulation of seismic waves in poroelastic media.* J. Comput. Phys. 455, 2022.

[8] S. Wollherr, A.-A. Gabriel, P.M. Mai: *Landers 1992 “reloaded”: an integrative dynamic earthquake rupture model.* JGR Solid Earth 124(7), pp. 6666–6702.

References

[8] M. Dumbser, M. Käser: *An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes – II. The 3d isotropic case.* Geophys. J. Int. 167(1), 2006.

[9] A. Heinecke, G. Henry, M. Hutchinson, H. Pabst: *LIBXSMM: Accelerating Small Matrix Multiplications by Runtime Code Generation*, SC16.

[10] M. Kruse, Z. Niu, S. Wolf, M. Lykkegaard, M. Bader, A.-A. Gabriel, L. Seelinger: *Scalable Bayesian inference of large simulations via asynchronous prefetching Multilevel Delayed Acceptance.* PASC '25: https://doi.org/10.1145/3732775.3733581

[11] M.B. Lykkegaard, T.J. Dodwell, C. Fox, G. Mingas, R. Scheichl: *Multilevel Delayed Acceptance MCMC.* SIAM/ASA J. Uncertainty Quantification 11 (1), 2023.

[12] L. Seelinger, A. Reinarz, L. Rannabauer, M. Bader, P. Bastian, R. Scheichl: *High performance uncertainty quantification with parallelized Multilevel Markov Chain Monte Carlo.* SC21.

[13] L. Seelinger, V. Cheng-Seelinger, A. Davis, M. Parno, A. Reinarz: *UM-Bridge: Uncertainty quantification and modeling bridge.* JOSS 8(83), 4748, 2023.

[14] L. Seelinger et al.: *Democratizing Uncertainty Quantification.* J. Comput. Phys. 521, 2025.

[15] T. Taufiqurrahman, A.-A. Gabriel, Duo Li, T. Ulrich, Bo Li, S. Carena, A. Verdecchia, F. Gallovič: *Dynamics, interactions and delays of the 2019 Ridgecrest rupture sequence.* Nature 618, 2023.

[16] T. Ulrich et al.: *Coupled, physics-based modeling reveals earthquake displacements are critical to the 2018 Palu, Sulawesi Tsunami.* Pure Appl. Geophys. 176, 4069–4109 (2019).