Parallel implementation of slope stability computations: from MATLAB to HPC

Michal Béreš[1,2], Stanislav Sysala[1], Simona Bérešová[1,2], Tomáš Luber[1]

[1] *Institute of Geonics of the Czech Academy of Sciences*
[2] *Department of Applied Mathematics, VSB–Technical University of Ostrava*

PROBLEM SETTING AND INTRODUCTION

DAVIS-MODIFIED MC AND SSR

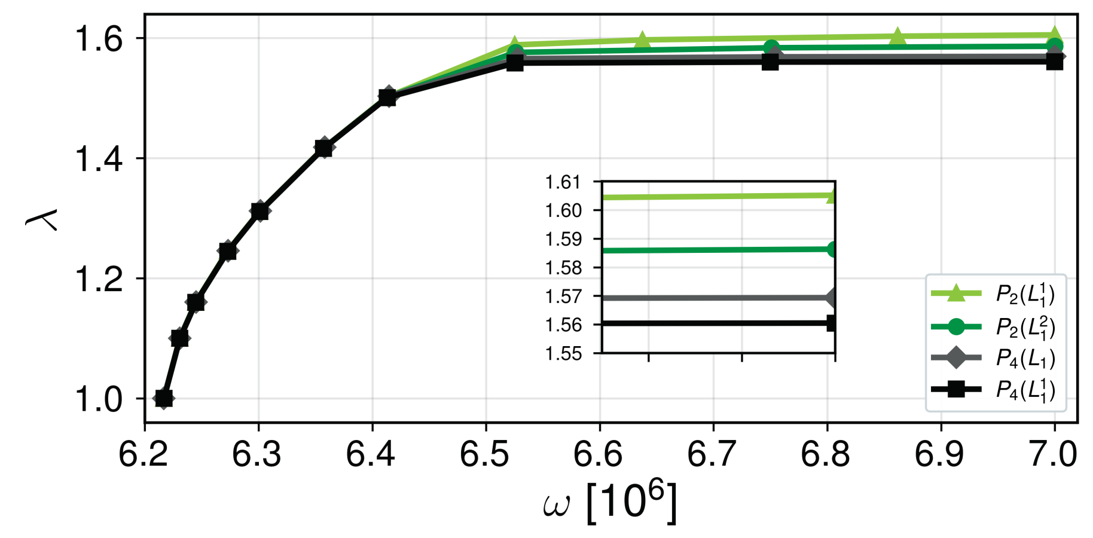
- Slope stability with Mohr–Coulomb plasticity.
- Davis–B SSR uses associated modified parameters [1].
- FoS is the limiting continuation value λ^*.

$$\tilde{c}_\lambda = \frac{c}{q(\lambda;\phi,\psi)}, \qquad \tan\tilde{\phi}_\lambda = \tan\tilde{\psi}_\lambda = \frac{\tan\phi}{q(\lambda;\phi,\psi)}$$

$$q = q_B = \lambda\frac{1-\sin\psi_\lambda\sin\phi_\lambda}{\cos\psi_\lambda\cos\phi_\lambda}$$

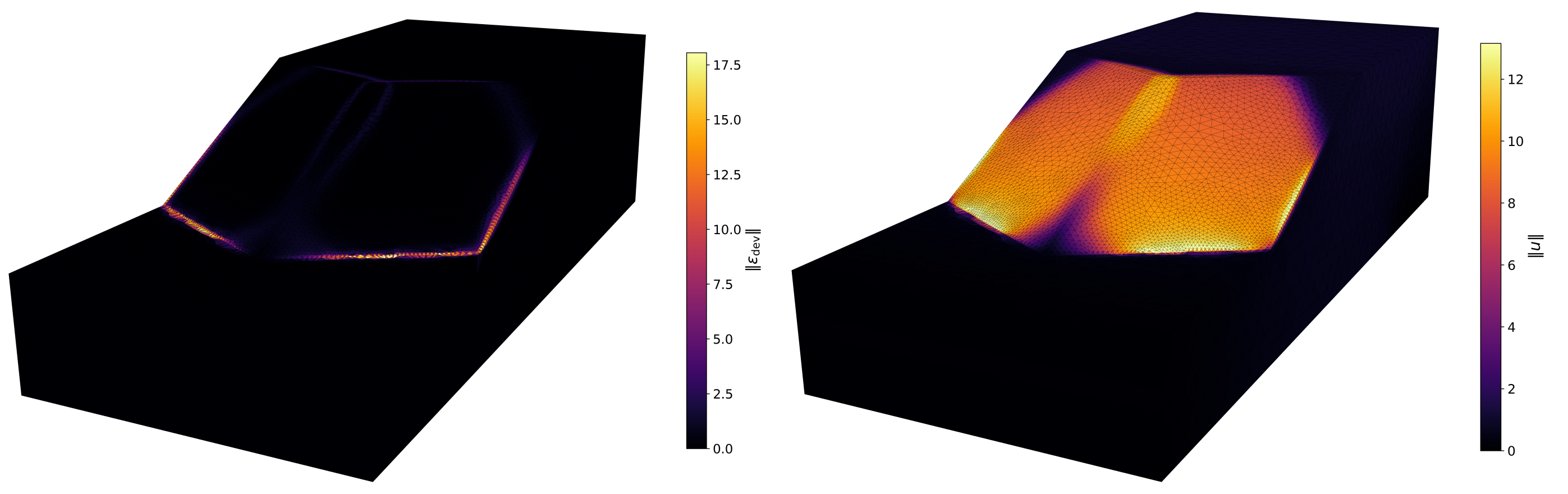
$$\phi_\lambda = \arctan\frac{\tan\phi}{\lambda}, \qquad \psi_\lambda = \arctan\frac{\tan\psi}{\lambda}$$

CONTINUATION CURVES



Computed pairs (ω_k, λ_k) approach λ^*; P2/P4 curves check solver reproducibility [1,4].

FINAL 3D FIELDS



Deviatoric strain marks the shear band; displacement magnitude shows the sliding mass.

NUMERICAL SOLUTION

INDIRECT CONTINUATION PROBLEM

- Control: work $\omega = b^\top u_\omega$.
- $F_\lambda(u) = \nabla\mathcal{Q}_\lambda(u)$: internal-force map.
- Unknowns: $(u_\omega, \lambda_\omega)$.
- Larger ω: $\lambda_\omega \to \lambda^*$ [2].

find $(u_\omega, \lambda_\omega):\ F_{\lambda_\omega}(u_\omega) = b; \quad b^\top u_\omega = \omega$

SEMISMOOTH NEWTON STEP

- Damped semismooth Newton solves the constrained SSR system.
- $K^{\varrho}_{\lambda_j,j}$ is the regularized tangent matrix.
- $G^{o}_{\lambda}(u)$: generalized derivative of $F_\lambda(u)$ with respect to λ.

$$K^{\varrho}_{\lambda_j,j}s^j + G^{o}_{\lambda_j}(u^j)\,\delta\lambda_j = b - F_{\lambda_j}(u^j); \quad b^\top s^j = 0$$

Two RHS share one tangent matrix; superposition recovers $\delta\lambda_j, s^j$.

LINEAR SOLVERS AND DEFLATION

- Near λ^*, tangent systems are ill-conditioned.
- FGMRES is preferred to CG for better numerical stability.
- Deflation basis: continuation states and recent Newton solves.
- It targets near-kernel modes from plastic-flow states [1].

FROM MATLAB TO PETSC

COMPARISON

Matched continuation/Newton iterations and physical results on benchmark cases.

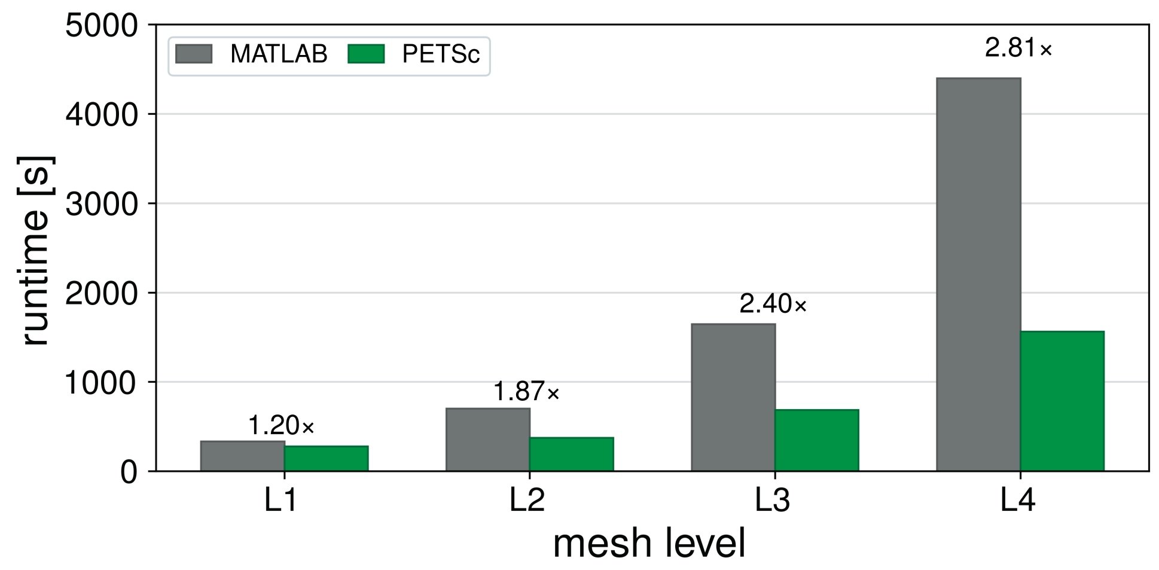
	MATLAB	PETSc
Constitutive relationship	vectorized	distributed
Assembly	sparse matmult	distributed
Preconditioner	HYPRE/AGMG	PETSc MG/HYPRE
Elements	P1/P2	P1/P2/P4
Visualisation	MATLAB	PyVista/ParaView

TANGENT ASSEMBLY AND PRECONDITIONING

- MATLAB bottlenecks: single-node/socket runs, global sparse assembly, and limited preconditioner choices.
- Tangent assembly builds owned rows on overlapped subdomains, limiting communication.
- Higher-order elements give P1/P2/P4 MG levels; coarse solves use HYPRE/GAMG or LU/MUMPS by size.

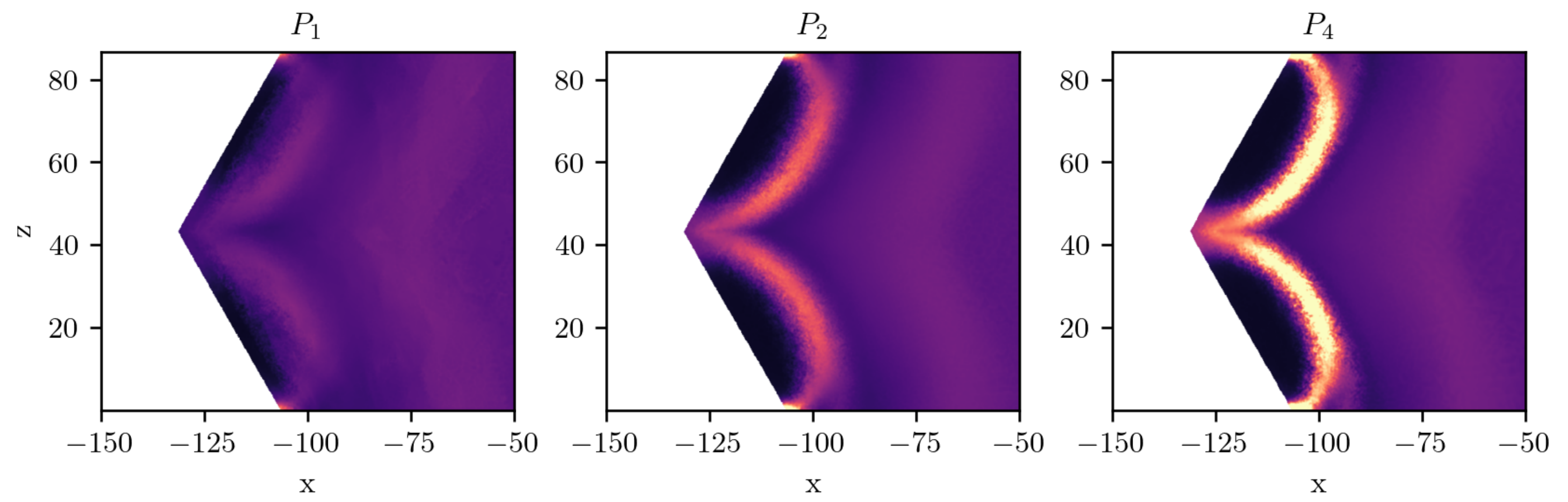
INCREASED PERFORMANCE ON SAME HARDWARE

MATLAB threads
vs
PETSc MPI



THE PROBLEM FAVOURS HIGHER ORDER DISCRETISATIONS

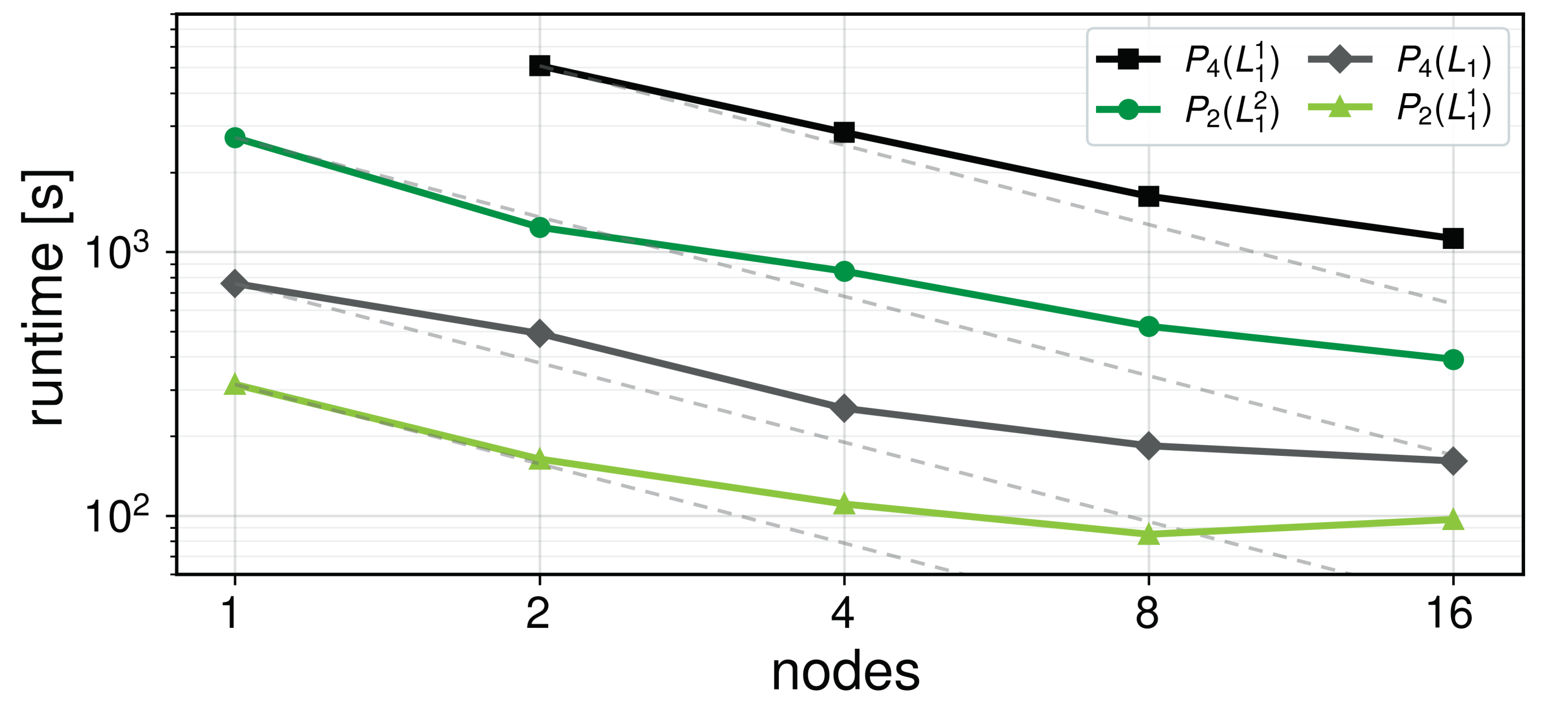
FAILURE-ZONE RESOLUTION: P1 / P2 / P4



Comparison of discretisations with the same number of DOFs: failure-zone localisation quality.

- Slope stability depends crucially on precise failure-zone estimation.
- Refined grids are usually needed; adaptivity [1] helps, but here element degree changes localisation more than grid density alone.
- Coarser high-order grids give sharper localisation and steadier FoS trends.
- Trade-off: extreme memory growth; full P4 solver at 4.8M DOFs does not fit on one node.

SCALING ON IT4I KAROLINA SUPERCOMPUTER



case	free DOFs
P4(L_1)	616,322
P4(L_1^1)	4,823,254
P2(L_1^1)	616,322
P2(L_1^2)	4,823,254

- Scaling on Karolina nodes.
- Same free DOFs; different P2/P4 sparsity and memory.
- Coarser P4: sharper localisation, more conservative FoS trend, faster than finer P2.
- Scaling is not optimal for high node counts.

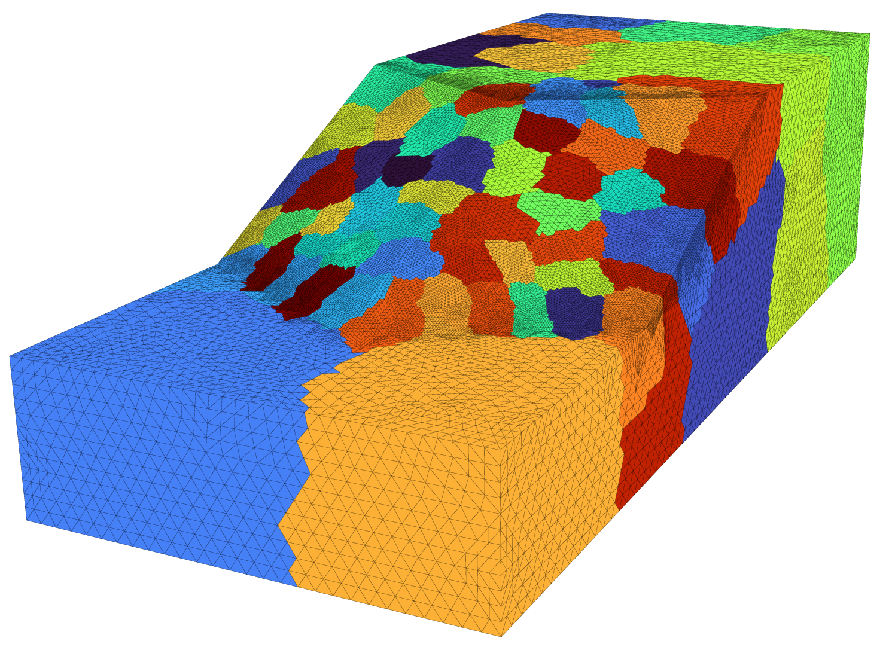
PETSC/PETSC4PY IMPLEMENTATION

PETSC/PETSC4PY

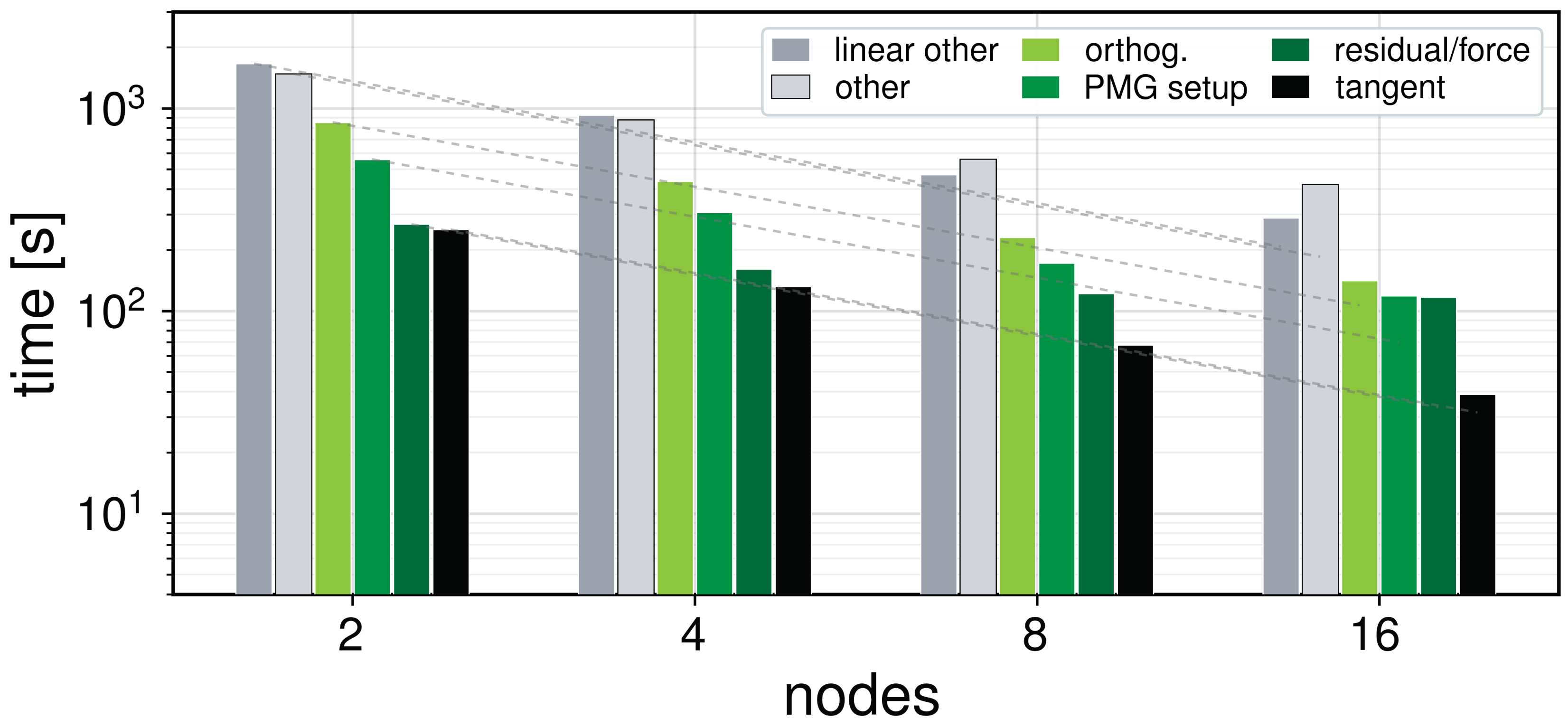
- Python/petsc4py controls continuation, Newton, deflation and solver orchestration.
- PETSc provides distributed matrices/vectors, KSP/PC setup and MPI execution.
- C/Cython handles tangent/force and constitutive loops; Python stays adjustable [4].

MULTIGRID PRECONDITIONER

- PMG: $P_1 \to P_2 \to P_4$; mixed coarse meshes when needed.
- Transfers: FE interpolation; transpose restrictions.
- Smoothers: Richardson/SOR or Chebyshev/Jacobi.
- Coloured subdomains show MPI ownership; overlap supports owned-row assembly.
- Coarse: LU/redundant LU, HYPRE or GAMG by size.



SCALING BREAKDOWN INTO PARTS



P4(L_1^1) timing per solution part.

REFERENCES

[1] S. Sysala et al., *Advanced continuation and iterative methods for slope stability analysis in 3D*, Comput. Struct., 2025.
[2] S. Sysala et al., *Convex optimization problems inspired by geotechnical stability analysis*, SIAM J. Optim., 2025.
[3] github.com/sysala/slope_stability
[4] github.com/Beremi/slope_stability_petsc4py

michal.beres@ugn.cas.cz | stanislav.sysala@ugn.cas.cz | simona.beresova@ugn.cas.cz | tomas.luber@ugn.cas.cz