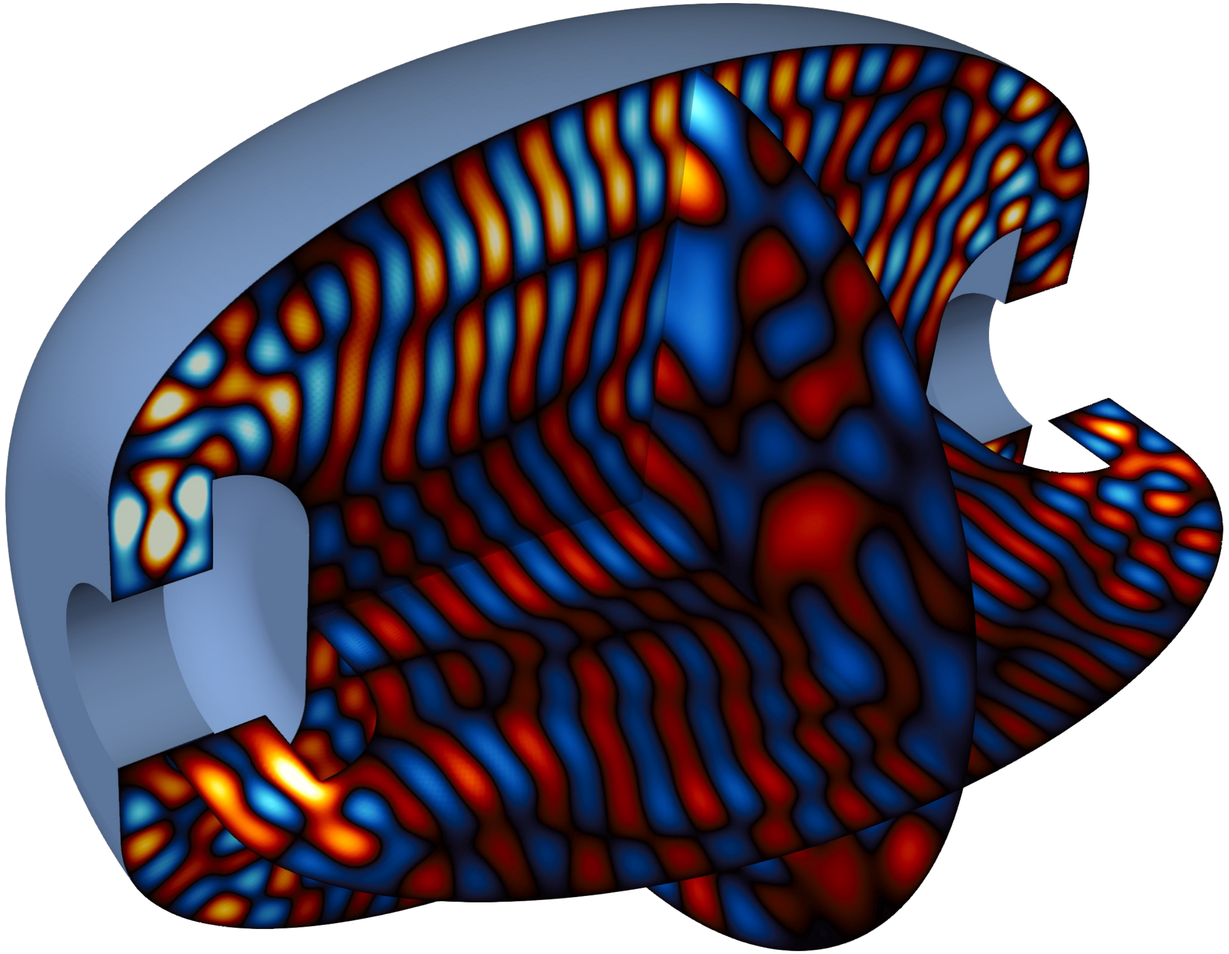
GPU Acceleration of IPDG in MFEM

Amit Rotem (Virginia Tech)
Joint work with:
Will Pazner (Portland State University), and
Daniel Appelö (Virginia Tech)



VIRGINIA TECH

Interior Penalty DG (IPDG)

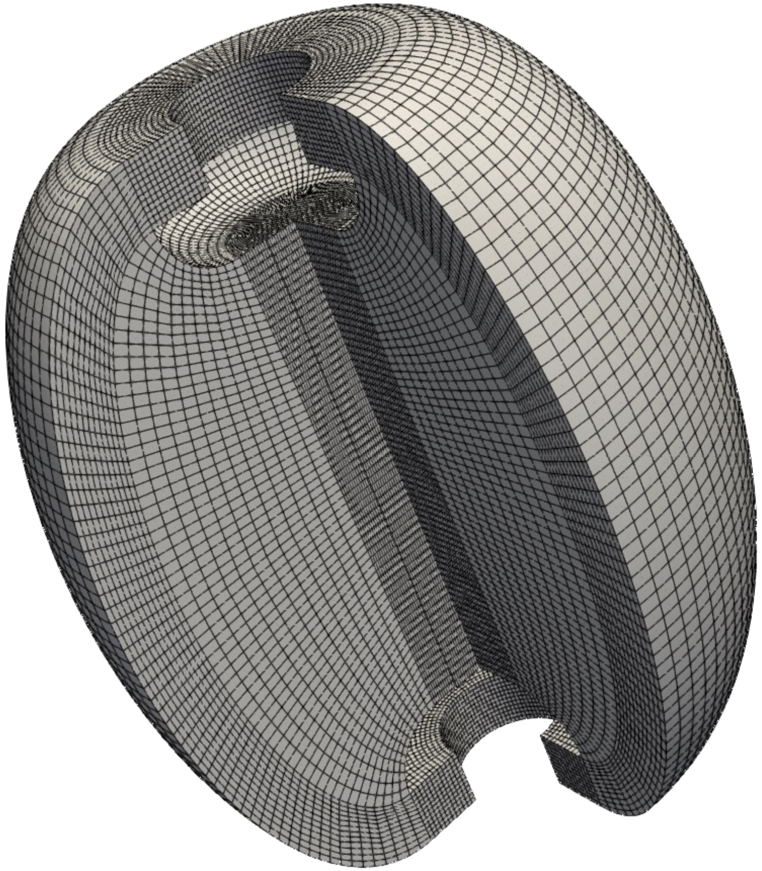
$$B_h(u,v) \coloneqq \sum_{k=1}^{K} (c^2 \nabla u, \nabla v)_{I_k} - \langle \{\mathrm{c}^2 \partial_{\mathbf{n}} u\}, [\![v]\!] \rangle_{\partial I_k}$$
$$+\sigma \langle [\![u]\!], \{c^2 \partial_{\mathbf{n}} v\} \rangle_{\partial I_k} + \kappa \langle \{h^{-1} c^2\} [\![u]\!], [\![v]\!] \rangle_{\partial I_k}.$$

- $\sigma = -1,\ \kappa > \kappa_0 > 0$ symmetric interior penalty method (SIPDG).
- GPU implementation of the volume term $(c^2 \nabla u, \nabla v)$ already exists in MFEM. Interior penalty terms $\langle \cdot, \cdot \rangle$ are new!

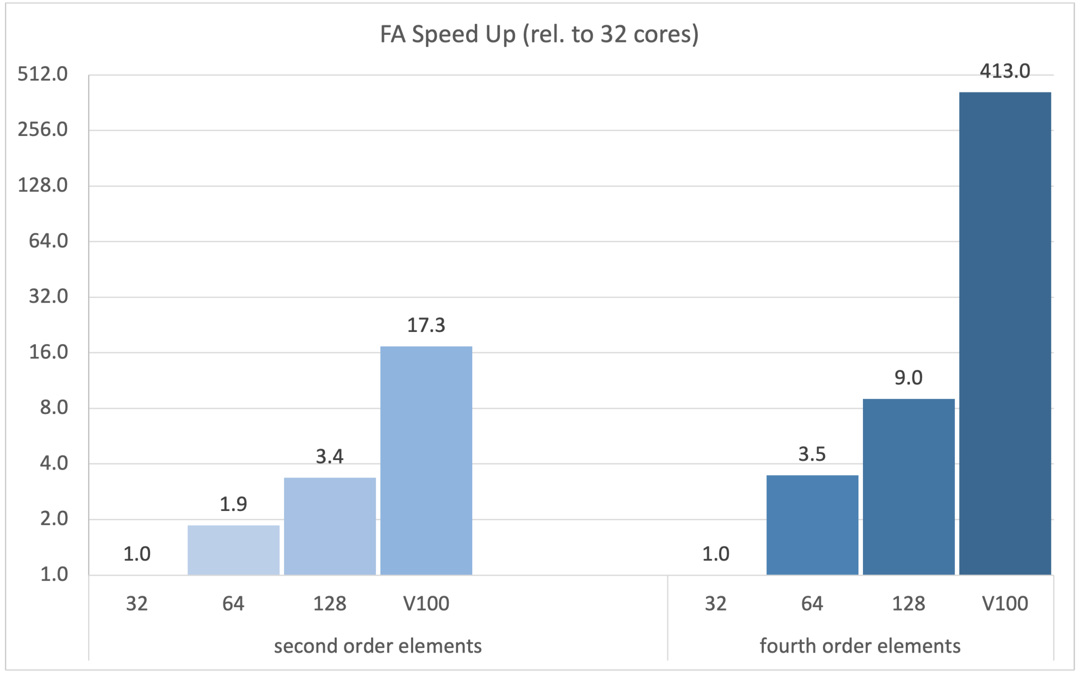
Goal: accelerate on GPU using sum factorization for tensor product elements.

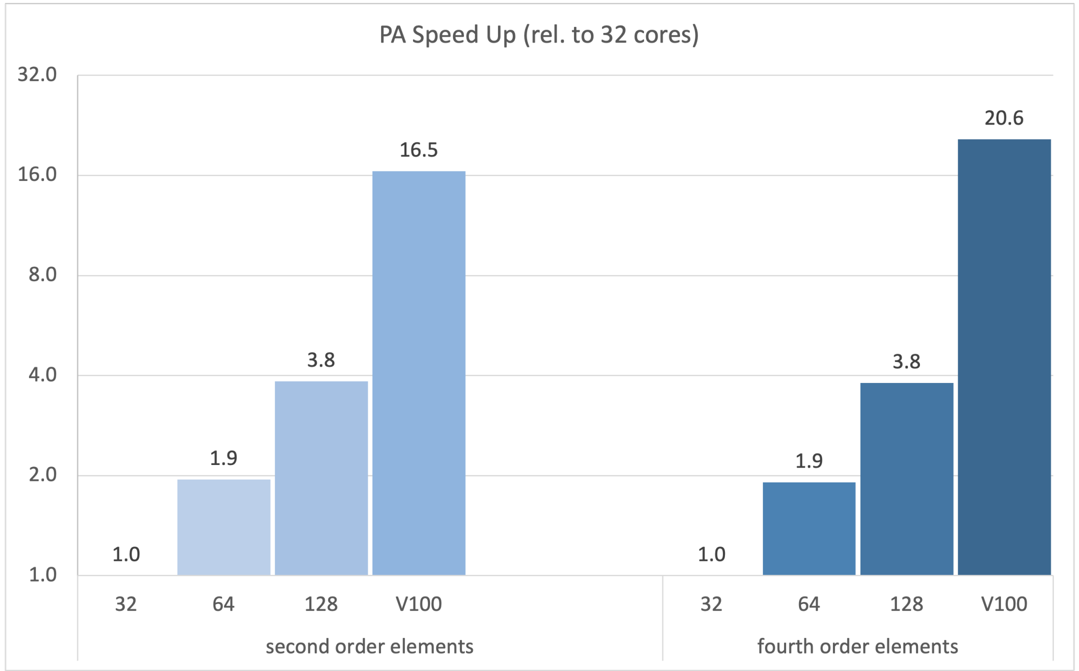
Benchmark Example:

- average wall time of $\boldsymbol{A}u$.
- 3D mesh with ~300k elements.
- Second order $(P = 2)$ and fourth order $(P = 4)$.
- Full Assembly vs. Partial Assembly.
- CPU: Intel Xeon Platinum 8260
- GPU: Nvidia V100

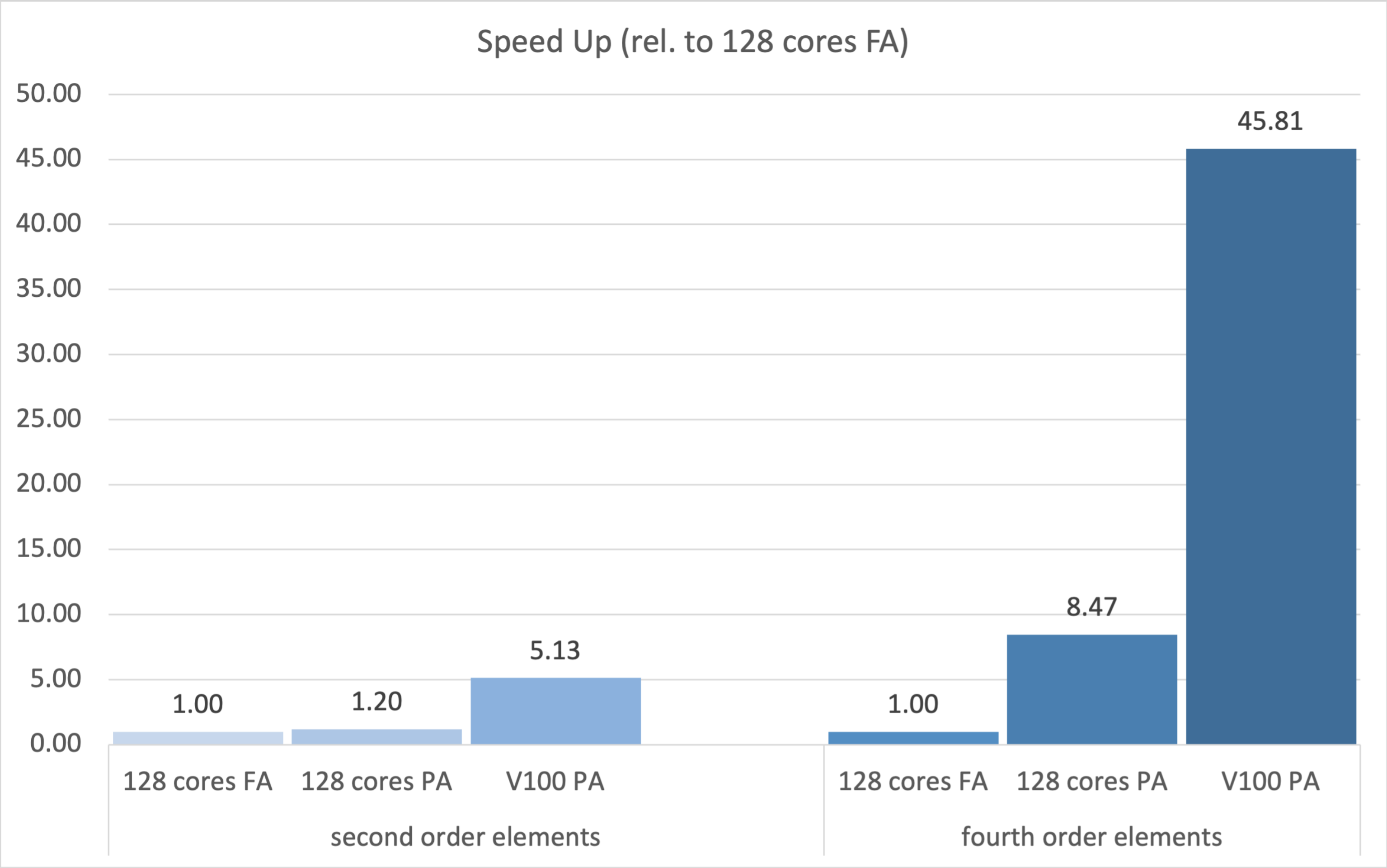


Speed-Up


FA Speed Up (rel. to 32 cores)
512.0
256.0
128.0
64.0
32.0
16.0
8.0
4.0
2.0
1.0
1.0
1.9
3.4
17.3
1.0
3.5
9.0
413.0
32
64
128
V100
32
64
128
V100
second order elements
fourth order elements


PA Speed Up (rel. to 32 cores)
32.0
16.0
8.0
4.0
2.0
1.0
1.0
1.9
3.8
16.5
1.0
1.9
3.8
20.6
32
64
128
V100
32
64
128
V100
second order elements
fourth order elements

Partial Assembly vs. Full Assembly

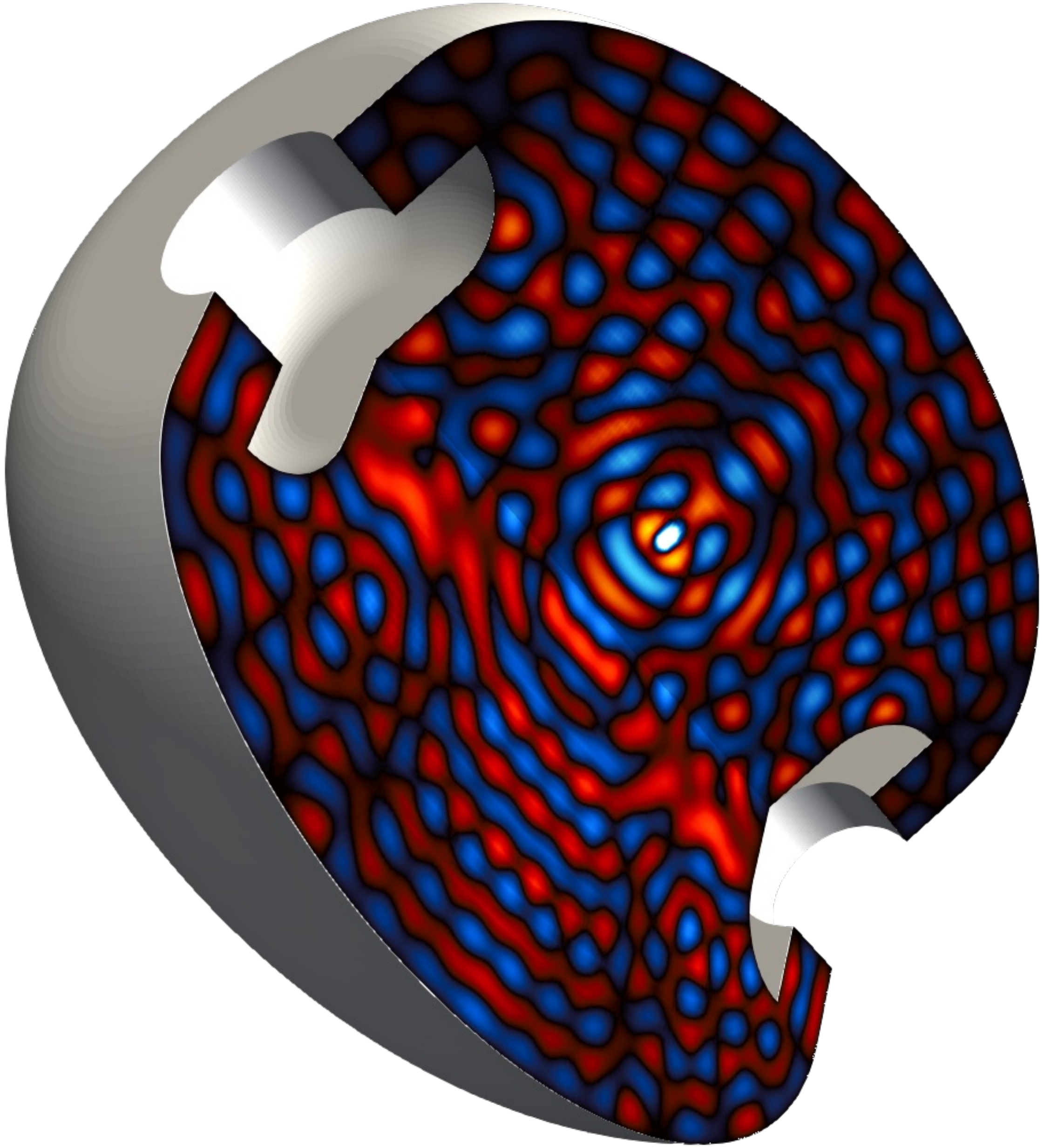


Thank you! Questions?

Acknowledgement: Supported in part by NSF 2208164 & 2210286 & STINT initiation grant.

References:

- MFEM: mfem.org
- Appelö, D., Garcia, F., & Runborg, O. (2020). WaveHoltz: Iterative Solution of the Helmholtz Equation via the Wave Equation. *SIAM Journal on Scientific Computing*, *42*(4), A1950–A1983. https://doi.org/10.1137/19M1299062



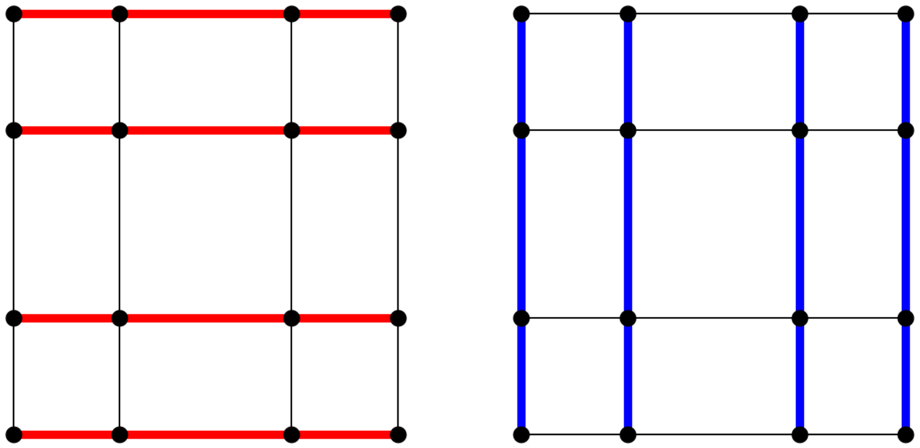
Sum Factorization

- Sum factorization:

$$B(u,v) \leftrightarrow \boldsymbol{A}u = (A^x \otimes I \otimes I + I \otimes A^y \otimes I + I \otimes I \otimes A^z)u.$$

$$(\boldsymbol{A}u)_{ijk} = \sum_{m=1}^{P} A^x_{im} u_{mkj} + A^y_{jm} u_{imk} + A^z_{km} u_{ijm}\,.$$

- $O\left(P^{2d}\right) \rightarrow O\left(P^{d+1}\right)$ operations.
  - Same complexity as finite differences!
- $O\left(P^{2d}\right) \rightarrow O(P^d)$ memory.
- Faster setup.



WaveHoltz

$$\hat{u}^{n+1} = \Pi \hat{u}^n.$$

Where

$$\Pi \hat{u} = \frac{2}{T} \int_0^T \left(\cos(\omega t) - \frac{1}{4} \right) u(x,t) dt, \qquad T = \frac{2\pi}{\omega}.$$

Here,

$$u_{tt} = \nabla \cdot (c^2 \nabla u) - f(x) \cos(\omega t)\,.$$
$$u(x,0) = \hat{u}(x), \qquad u_t(x,0) = 0.$$

- Fixed point $\hat{u} = \Pi \hat{u}$ iff $\nabla \cdot (c^2 \nabla \hat{u}) + \omega^2 \hat{u} = f$.
- Equivalent linear system:

$$(I - S)\hat{u} = \pi_0, \qquad S\,\hat{u} = \Pi \hat{u} - \pi_0, \qquad \pi_0 = \Pi 0.$$

- After discretization $I - S$ is symmetric positive definite → Solve with conjugate gradient.