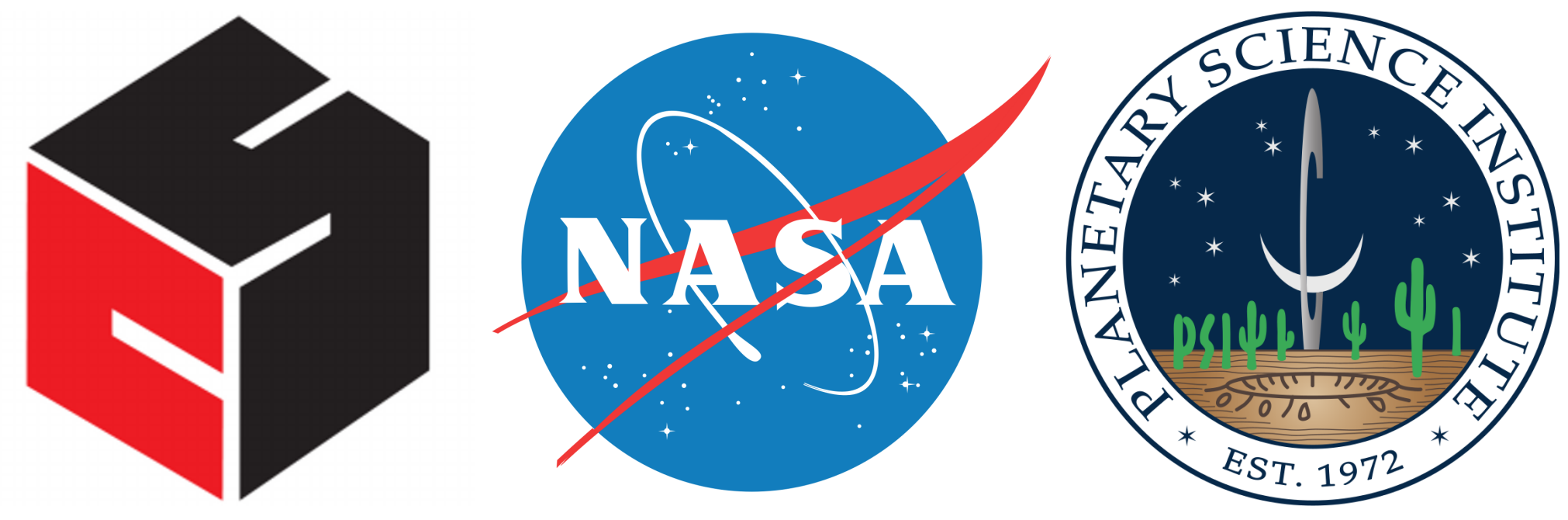




Illumination and Temperature on Rough Terrain: Fast Methods for Solving the Radiosity Equation

Samuel F. Potter (sfp@umiacs.umd.edu)^{†,§}, Norbert Schörghofer[§], and Erwan Mazarico[★]
[†]: University of Maryland Department of Computer Science, [§]: Planetary Science Institute, [★]: NASA Goddard Space Flight Center
Acknowledgement: this research was supported by the NASA Planetary Science Division Research Program

Thermal modeling on rough surfaces

- Airless bodies have strong horizontal temperature gradients due to shadows cast by rough topography
- Lunar cold traps exist because of terrain shadowing and are defined by surface temperature
- Thermal models must incorporate shadows, but also long- and short-wavelength radiation between surface elements [3]
- Element-to-element radiation dominates runtime**

Contribution

- We develop a fast algorithm for solving the equations governing scattering of long- and short-wavelength radiation
- Solve two discretized radiosity equations to compute temp.
- Offline, we precompute a compressed low-rank version of the discretized kernel matrix by compressing low-rank off-diagonal blocks using sparse SVDs
- Online, multiplication requires nearly $O(N)$ time**, where N is the number of triangular elements
- The matrix only depends on the geometry of the planet so can be used for simulations spanning a long time

Physical model [5]

- Energy balance on the surface:**

$$\epsilon\sigma T^4 = (1 - \rho)(E + Q) + \epsilon Q_{\text{IR}} \quad (1)$$

where:

ρ := albedo
 E := incoming solar radiation (insolation) [W m^{-2}]
 Q := reflected sunlight [W m^{-2}]
 Q_{IR} := thermal emission [W m^{-2}]
 T := temperature [K]
 ϵ := emissivity
 σ := Stefan-Boltzmann constant [$\text{W m}^{-2} \text{K}^{-4}$]

- Governing equations for scattering:**

$$Q(\vec{x}) = \frac{1}{\pi} \int_S \rho(\vec{y}) (E(\vec{y}) + Q(\vec{y})) F(\vec{x}, \vec{y}) dA(\vec{y}) \quad (2)$$

$$Q_{\text{IR}}(\vec{x}) = \frac{1}{\pi} \int_S (\epsilon\sigma T(\vec{y})^4 + (1 - \epsilon)Q_{\text{IR}}(\vec{y})) F(\vec{x}, \vec{y}) dA(\vec{y}) \quad (3)$$

where:

S := surface of the planet or crater
 dA := surface area element [m^2]
 $F(\vec{x}, \vec{y}) := \frac{[\vec{n}(\vec{x}) \cdot (\vec{y} - \vec{x})]_+ [\vec{n}(\vec{y}) \cdot (\vec{x} - \vec{y})]_+}{\pi \|\vec{x} - \vec{y}\|^4} + V(\vec{x}, \vec{y})$
 $\vec{n}$:= surface normal on S
 $[x]_+ := \max(0, x)$ = positive part
 $V(\vec{x}, \vec{y}) := \begin{cases} 1 & \text{if } \vec{y} \text{ is visible from } \vec{x} \\ 0 & \text{otherwise} \end{cases}$

The radiosity method

- Equations (2) and (3) are radiosity integral equations:**

$$B(\vec{x}) = E(\vec{x}) + \rho(\vec{x}) \int_S G(\vec{x}, \vec{y}) B(\vec{y}) dA(\vec{y}) \quad (4)$$

where:

B := radiosity [W m^{-2}]
 E := self-emitted radiosity [W m^{-2}]
 ρ := albedo
 F := geometric kernel

- Midpoint collocation discretization of (4) gives the system:**

$$KB = (I - \rho F) B = E \quad (5)$$

$$F_{ij} = \frac{[\mathbf{n}_i^\top (\mathbf{p}_j - \mathbf{p}_i)]_+ [\mathbf{n}_j^\top (\mathbf{p}_i - \mathbf{p}_j)]_+}{\pi \|\mathbf{p}_i - \mathbf{p}_j\|^4} V_{ij} A_j \quad (6)$$

where F is the **view factor matrix** and:

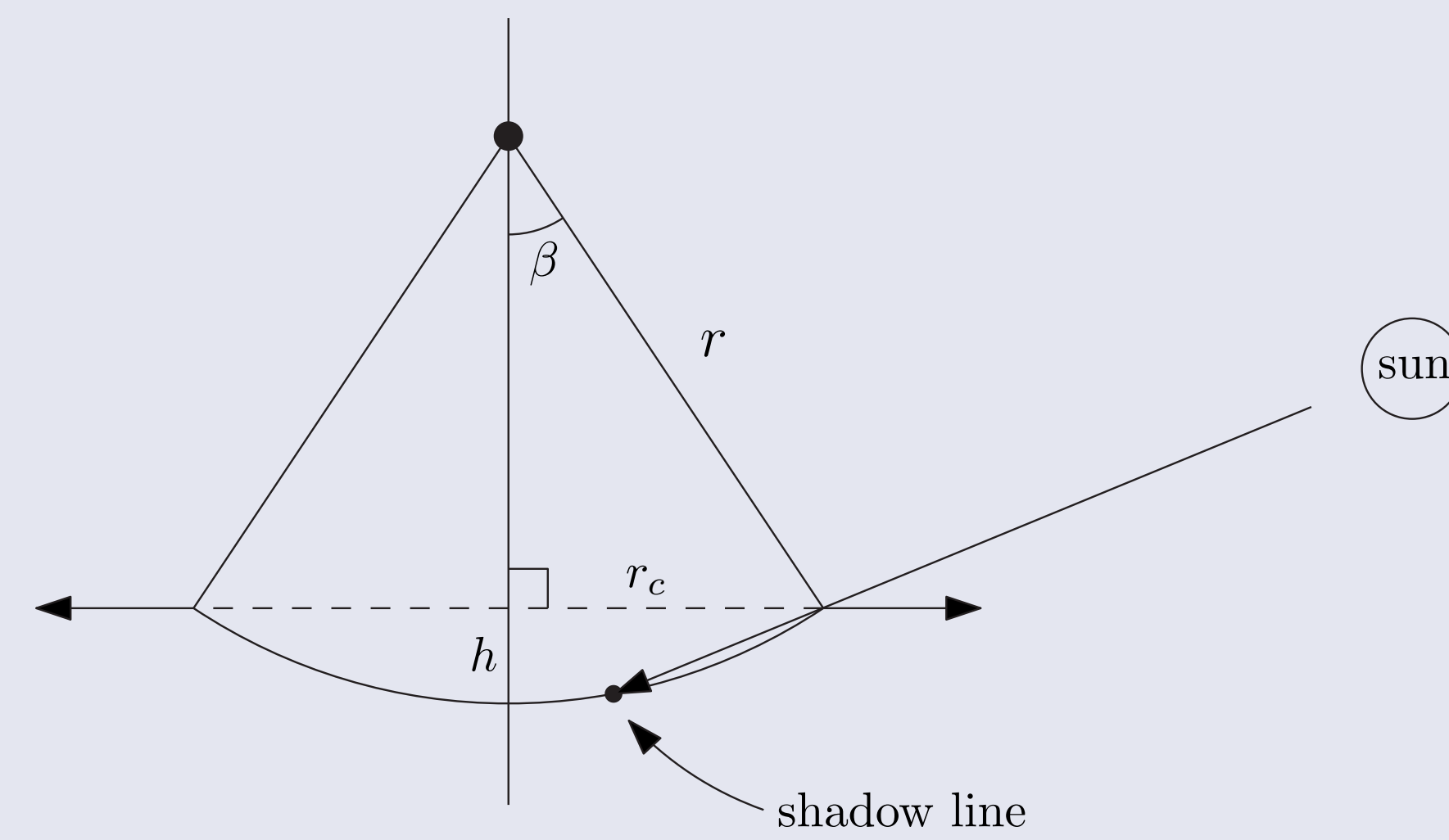
$\mathbf{p}_i$:= centroid of i^{th} triangle
 $\mathbf{n}_i$:= surface normal at $\mathbf{p}_i$
 A_i := area of i^{th} triangle
 V_{ij} := visibility between $\mathbf{p}_i$ and $\mathbf{p}_j$

- Solving the discrete radiosity system:**

- The system (5) can be solved in a small number of Neumann or Jacobi iterations (typically 2 to 5)
- Nearly perfectly conditioned since it's a discretized BIE
- Main challenge: fast multiplication by F**

The Ingersoll crater test problem [2]

- A crater formed from a spherical cap:**



- The steady state temperature for the Ingersoll crater has an exact solution for Lambertian reflectance inside the shadowed portion of the crater:**

$$T_{\text{Ingersoll}} = \frac{F_0 \sin(e_0)}{\sigma} \frac{1 - A}{1 - Af} \left(1 + \frac{A(1 - f)}{\epsilon} \right)^{1/4} \quad (7)$$

where σ , A , and ϵ are as before, and:

$f := S_c / (4\pi r^2)$
 S_c := crater surface area [m^2]
 F_0 := solar constant [W m^{-2}]
 e_0 := solar elevation relative to horizon

- We use this test problem to validate our numerical method**

Low rank compression of form factor matrix F

- Spatial partitioning:** use quadtree or octree to recursively partition triangular elements
- Low-rank interactions:** blocks of F that correspond to interactions between nonoverlapping cells in quadtree or octree are typically sparse with a dense low-rank subblock
- SVD compression:** compute SVD to find dense subblock and compress it within a given tolerance ϵ
- Best low-rank approximation by SVD:**

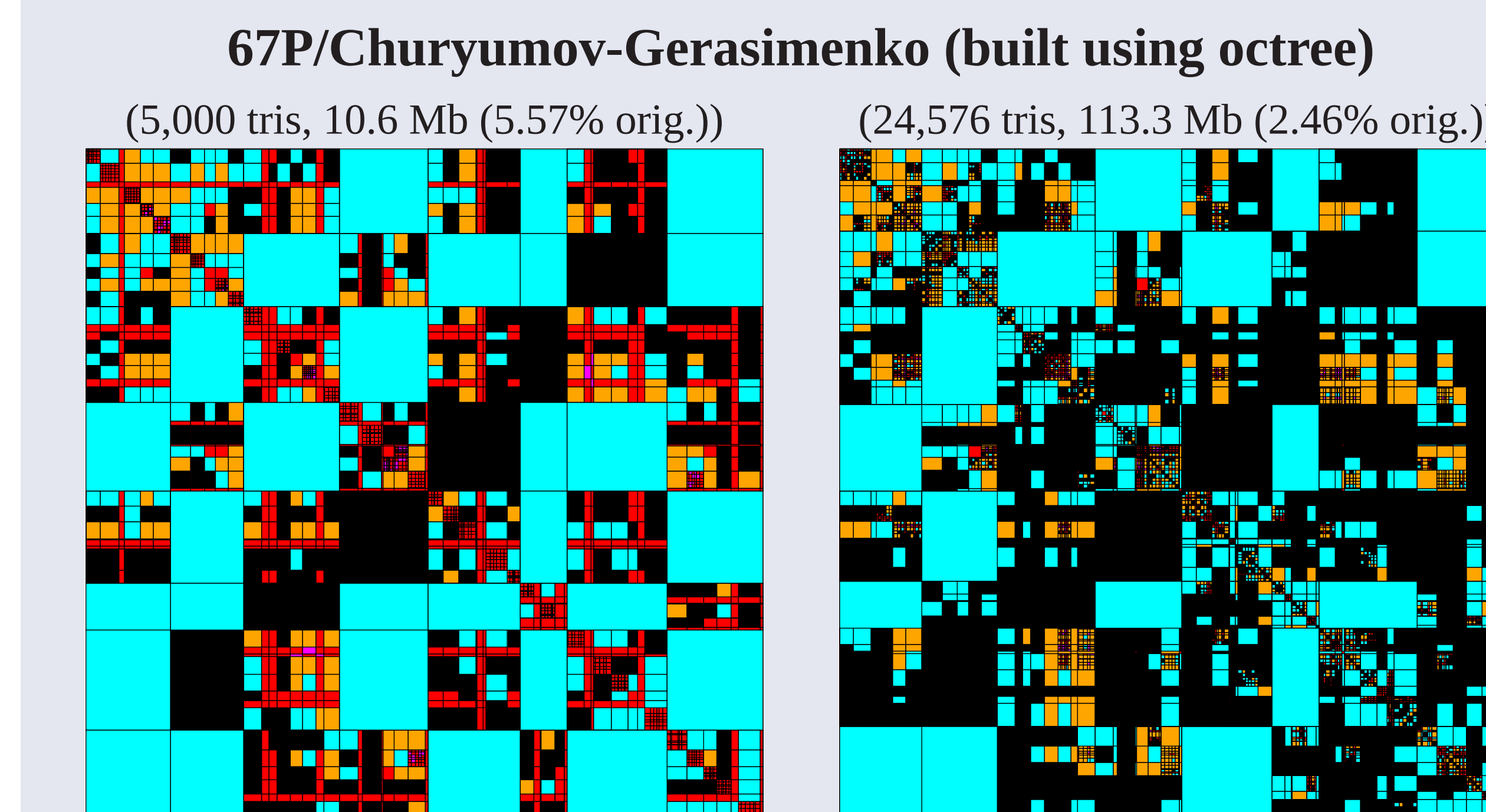
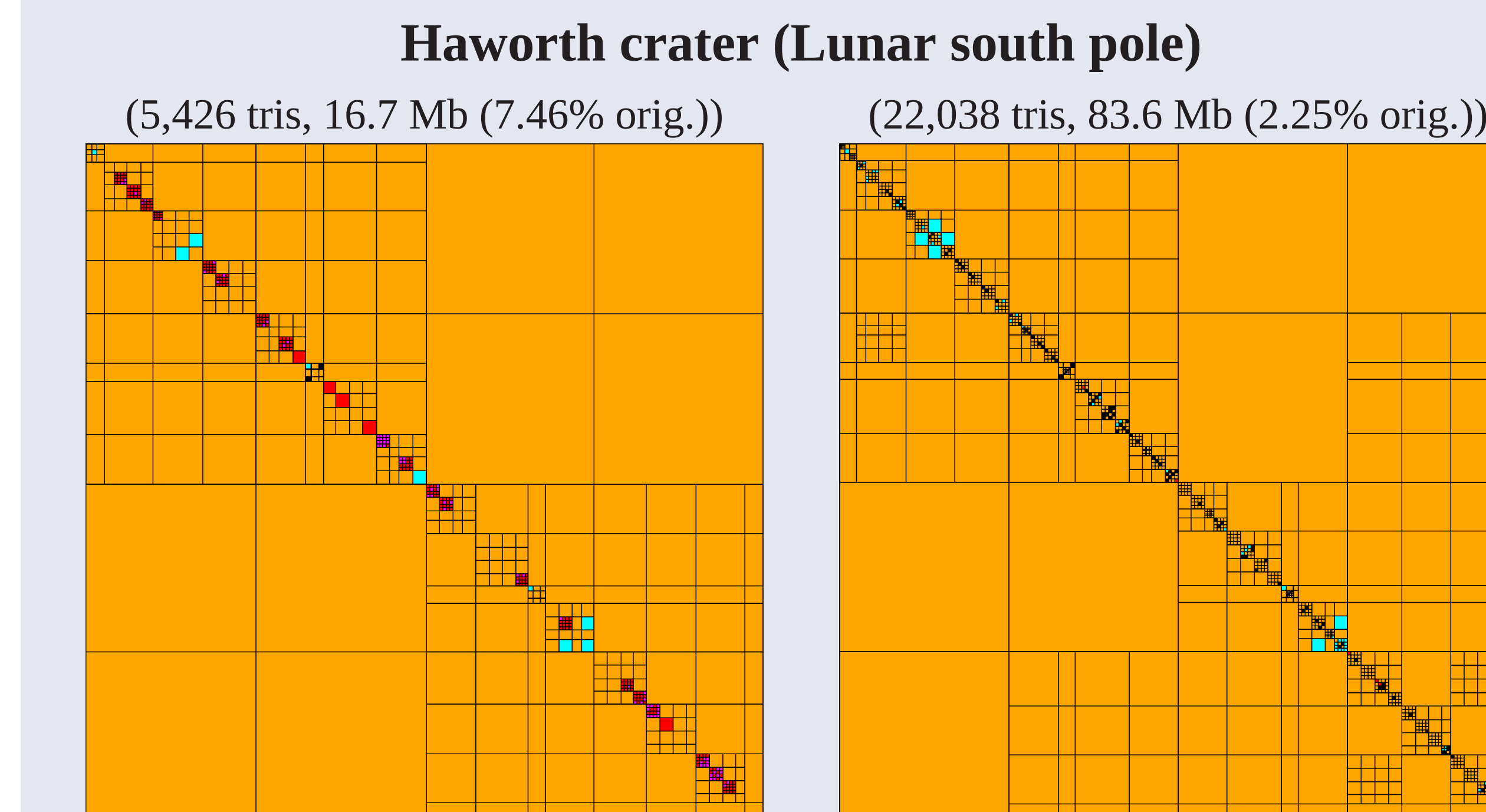
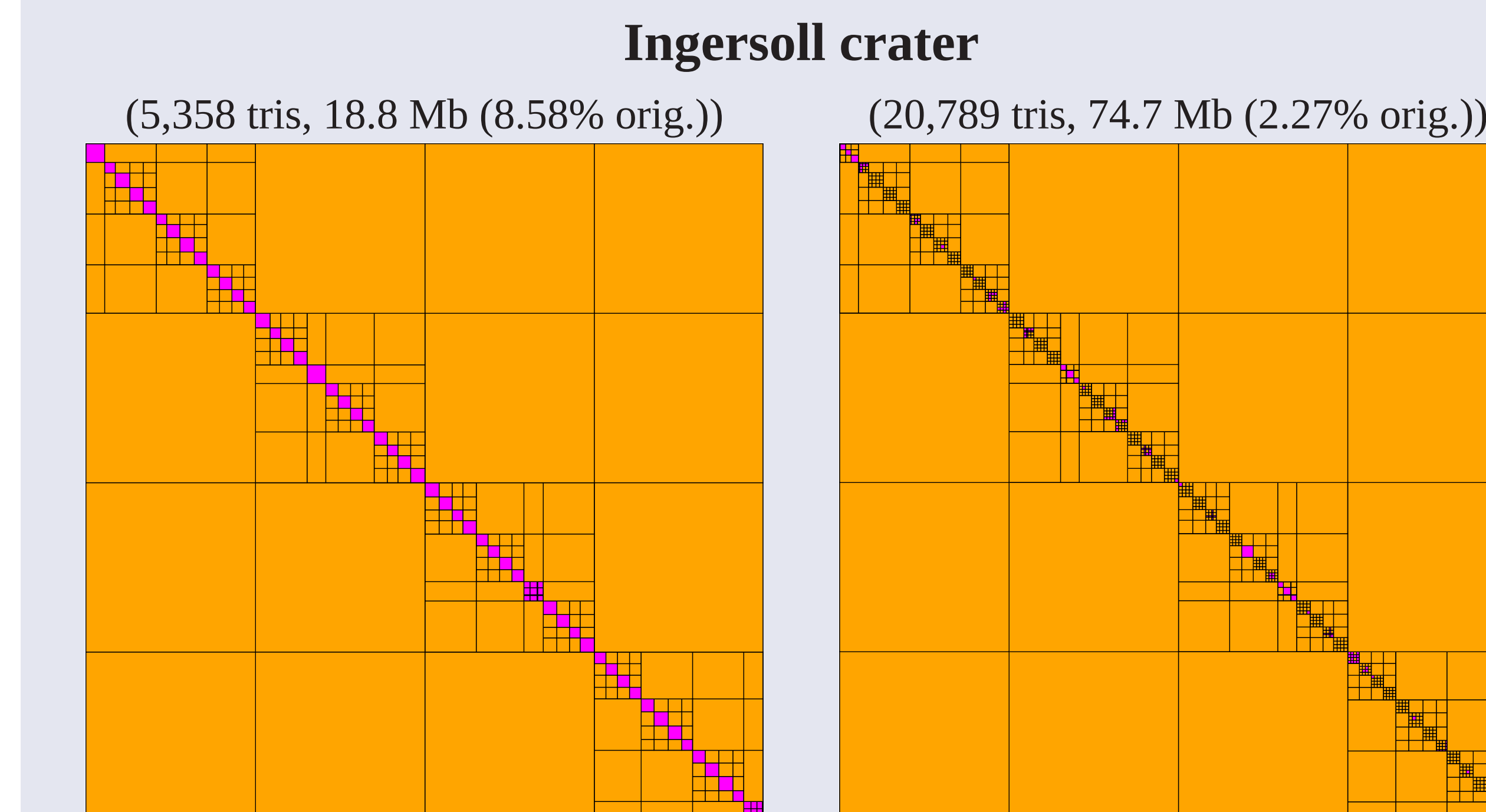
$$\min_{\text{rank}(\mathbf{F}) \leq k} \|\mathbf{F} - \mathbf{F}_k\|_2 = \sigma_{k+1} \quad (8)$$

where k is a fixed rank, $\mathbf{F}_k = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^\top$ is computed from the rank k truncated SVD of F , and σ_i is the i^{th} singular value of F

- This approach is similar to the $\mathcal{H}$ -matrix format [1]

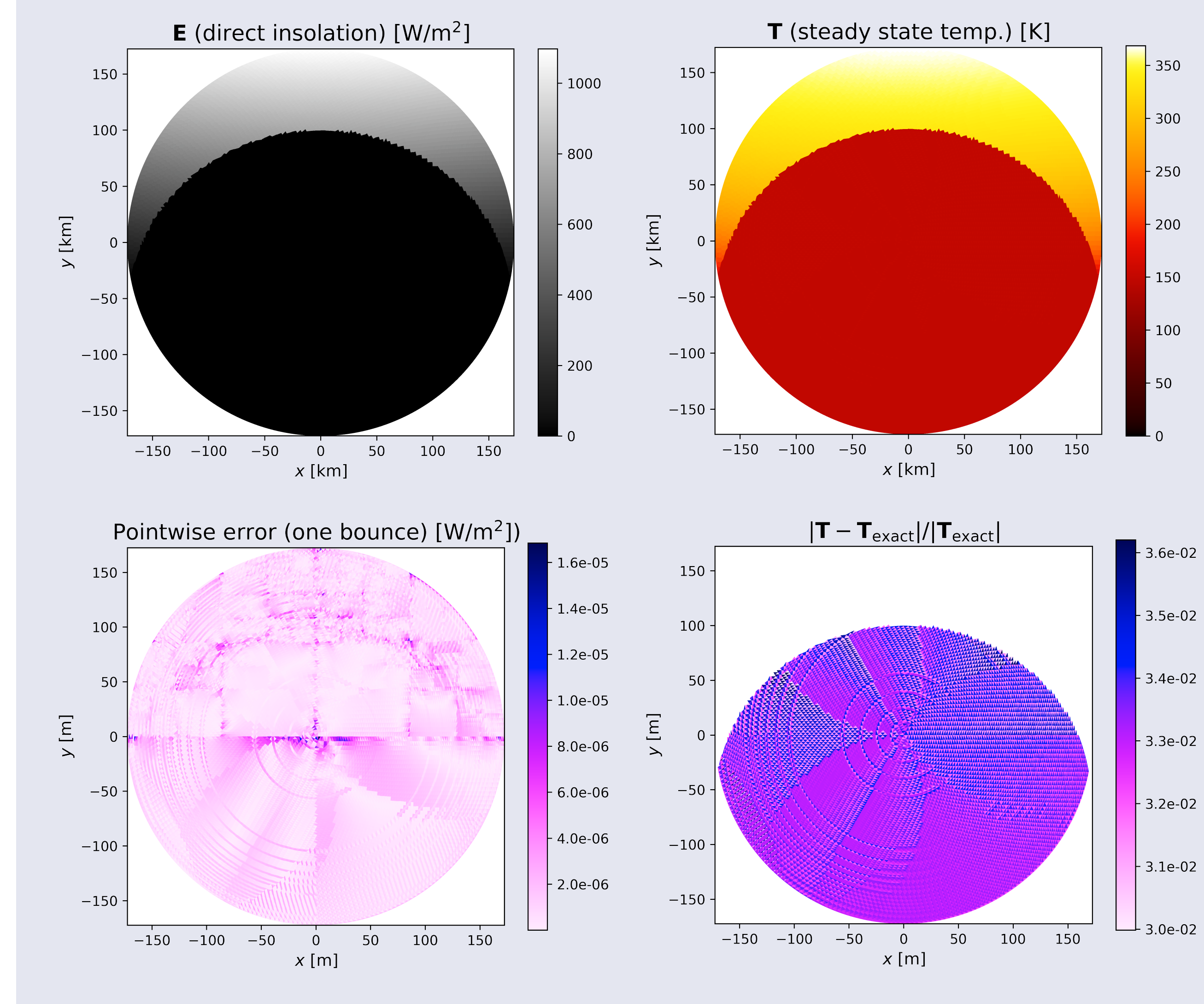
Examples of compressed F matrices

■, ■*: SVD block, ■: sparse block, ■: dense block, ■: zero block
 (*: the cyan SVD blocks can be thought of as “especially sparse” SVD blocks)



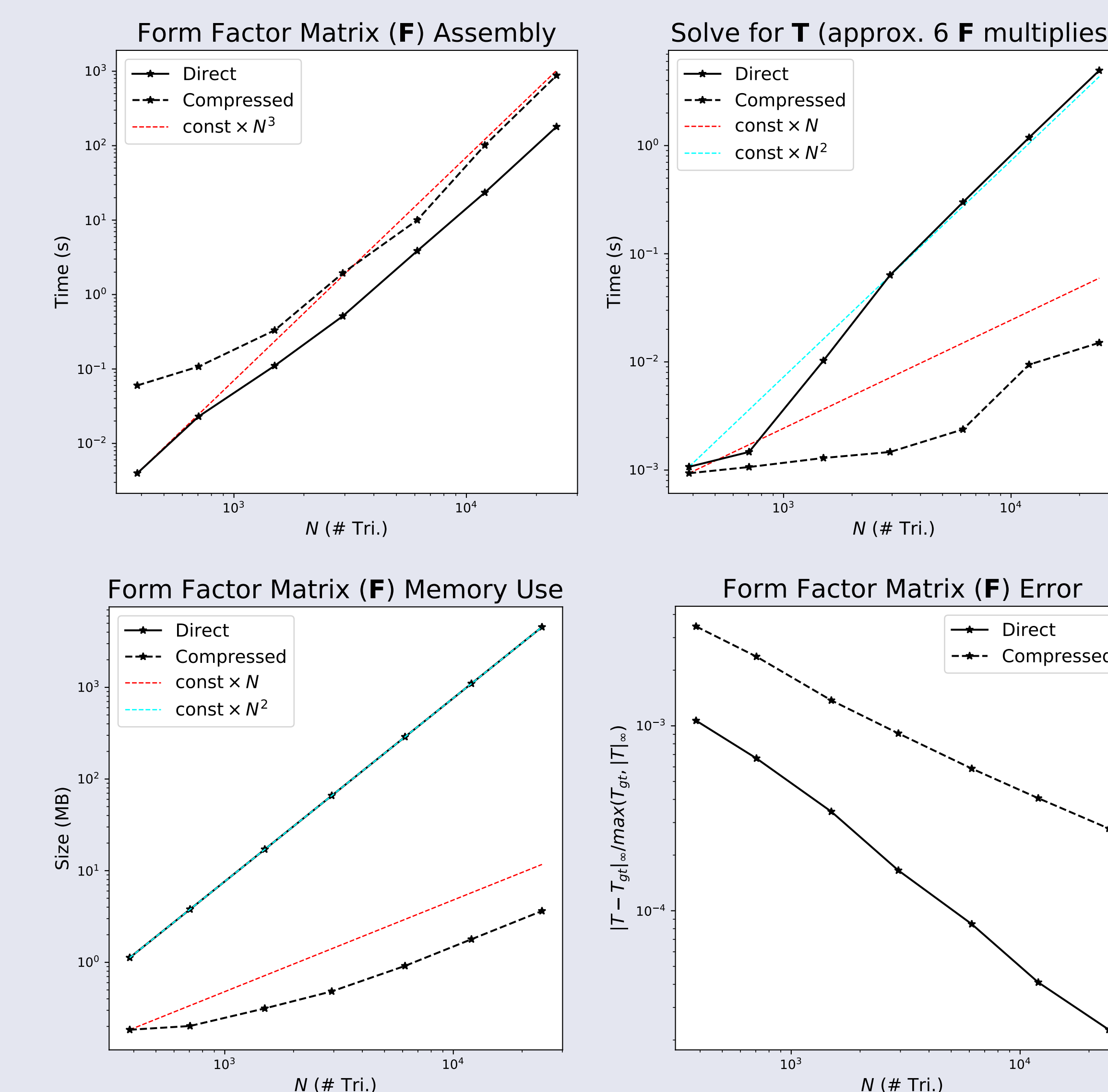
Ingersoll test problem: numerical results

- Only part of the crater is illuminated (*top-left*)
- The steady state temperature in the shadow is constant (*top-right*)
- Pointwise error after one bounce is below the $\epsilon = 10^{-4}$ tolerance (*bottom-left*)
- Error between T and exact Ingersoll temperature is uniform (*bottom-right*)
- Errors are sensitive to the triangulation (*bottom-left/right*)



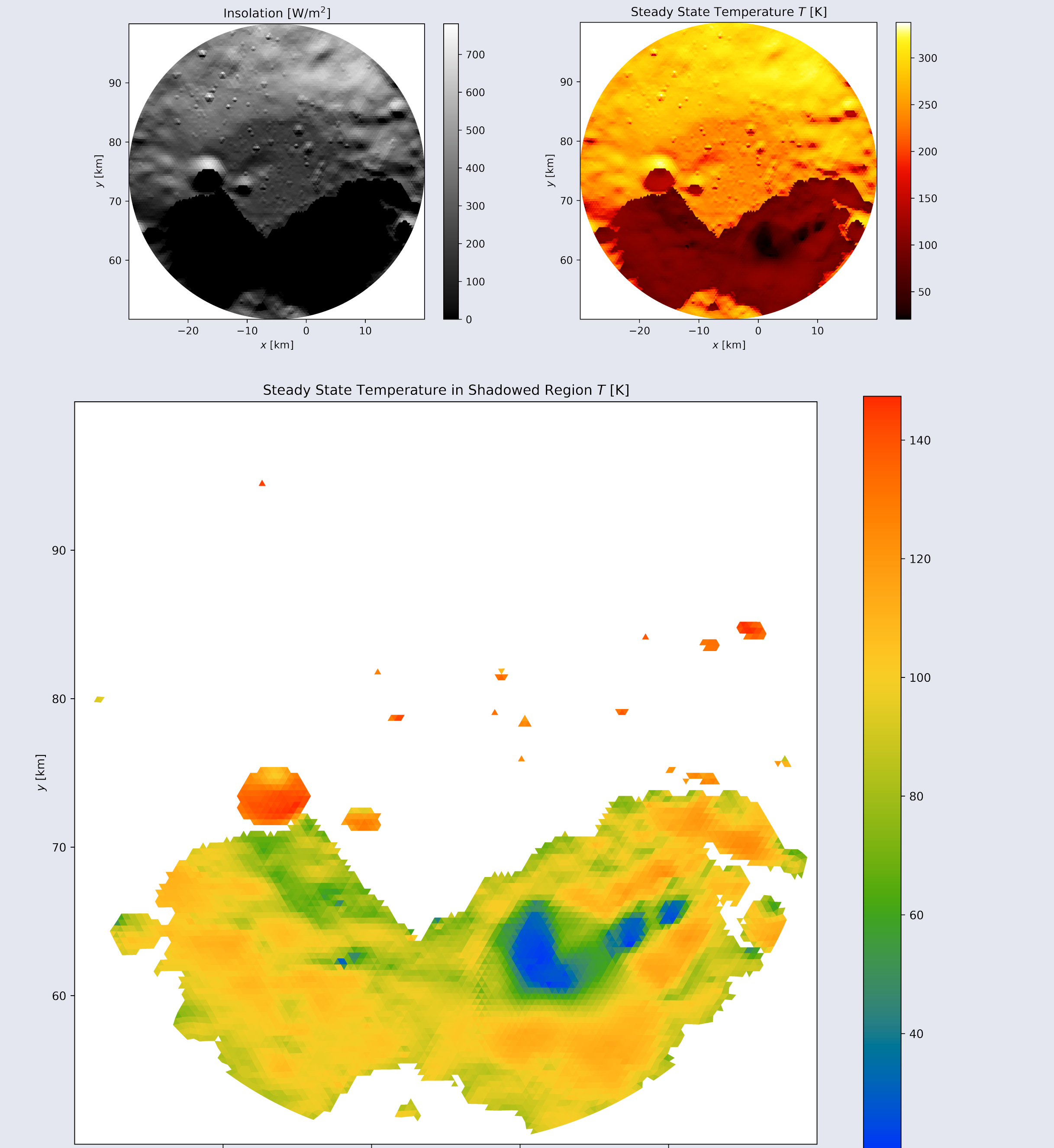
Ingersoll test problem: performance results

- We build the compressed F matrix with a tolerance of $\epsilon = 10^{-4}$ and a maximum SVD rank of $k = 60$
- We compare the exact steady state temperature in the shadow region given by (7) with the numerical steady temperature vs. problem size (*bottom-right*)



Lunar south pole: Haworth crater test problem

- Lunar south pole terrain from DEM obtained by LOLA [6]
- High-quality triangle mesh constructed using Python version of distmesh [4]
- Example here is physically unrealistic because of steady state assumption, but solving the steady state system is the necessary step for solving the time-dependent problem efficiently**
- The temperature is computed using the compressed form factor matrix
- The temperature plotted below approximates the temperature using the exact, dense form factor matrix *pointwise*, **validating the fast method**



References

- Wolfgang Hackbusch. *Hierarchical matrices: algorithms and analysis*, volume 49. Springer, 2015.
- Andrew P Ingersoll, Tomas Svitek, and Bruce C Murray. Stability of polar frosts in spherical bowl-shaped craters on the Moon, Mercury, and Mars. *Icarus*, 100(1):40–47, 1992.
- David A Paige, Matthew A Siegler, Jo Ann Zhang, Paul O Hayne, Emily J Foote, Kristen A Bennett, Ashwin R Vasavada, Benjamin T Greenhagen, John T Schofield, Daniel J McCleese, et al. Diviner lunar radiometer observations of cold traps in the moon’s south polar region. *science*, 330(6003):479–482, 2010.
- Per-Olof Persson and Gilbert Strang. A simple mesh generator in MATLAB. *SIAM review*, 46(2):329–345, 2004.
- Norbert Schörghofer. Planetary Code Collection. <https://github.com/nschorgh/Planetary-Code-Collection>, 2018.
- David E Smith, Maria T Zuber, Gregory A Neumann, Erwan Mazarico, Frank G Lemoine, James W Head III, Paul G Lucey, Oded Aharonson, Mark S Robinson, Xiaoli Sun, et al. Summary of the results from the lunar orbiter laser altimeter after seven years in lunar orbit. *Icarus*, 283:70–91, 2017.