Optimization and Application of the Fast and Efficient Approach to Electromagnetic Field Computations with Limited Computer Resources

Iurii B. Minin, Sergey A. Matveev

Fryazino branch of Kotelnikov IRE of RAS, Moscow State University

Agenda

- 1. Introduction & Motivation
- 2. Applications
- 3. Methods
- 4. Validation & Optimization
- 5. Numerical Experiments
- 6. Results
- 7. Conclusions
- 8. Appendix (Poster Summary)

Motivation & Problem Statement

- - Challenge: Solving Helmholtz equations in heterogeneous media
- - Application: Photonic and micro/nano device design
- - Limitation: High computational cost and memory requirements

Key Challenges

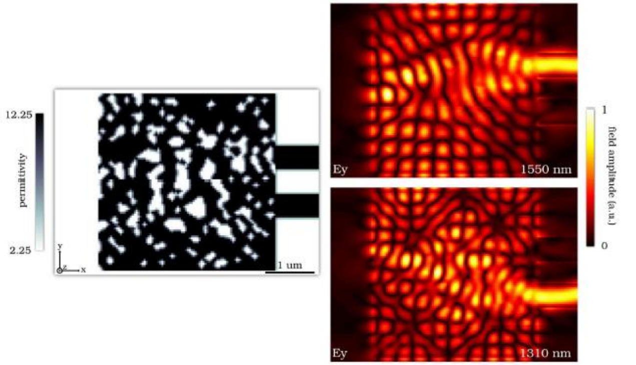
- - Standard solvers (Jacobi, Gauss-Seidel) converge too slowly
- - GMRES converges faster but consumes huge memory (Krylov subspace)
- - Large grids → memory explosion
- - Need efficient method for speed + memory balance

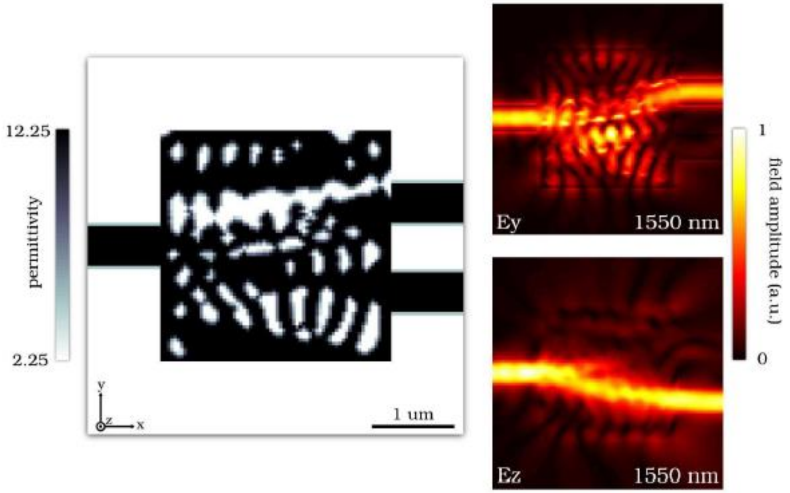
What's New?

- - Combination: GMRES(k) + FFT + Toeplitz + GPU
- - Achieved ~10× lower memory use with restarts
- - Up to 2× larger task deployment without MPI or new hardware
- - Validated with photonic device applications

Applications

- - TE/TM polarization splitt
- - Wavelength splitter
- - Bragg reflectors
- - Fiber Bragg gratings
- - General optimization of antennas and photonic structures





Research Objective

- - Develop fast, resource-efficient algorithm for 2D Helmholtz equation
- - Optimize trade-off: memory vs. computation speed
- - Handle large grids (up to 8192×8192)

Validation & Optimization Approaches

- - Validation with Green’s Function Integral Equation Method (GFIEM)
- - Bi-directional Evolutionary Structural Optimization (BESO)
- - Python-based ESO results (demonstrated effectiveness)
- - Enhances design domain optimization for photonic devices

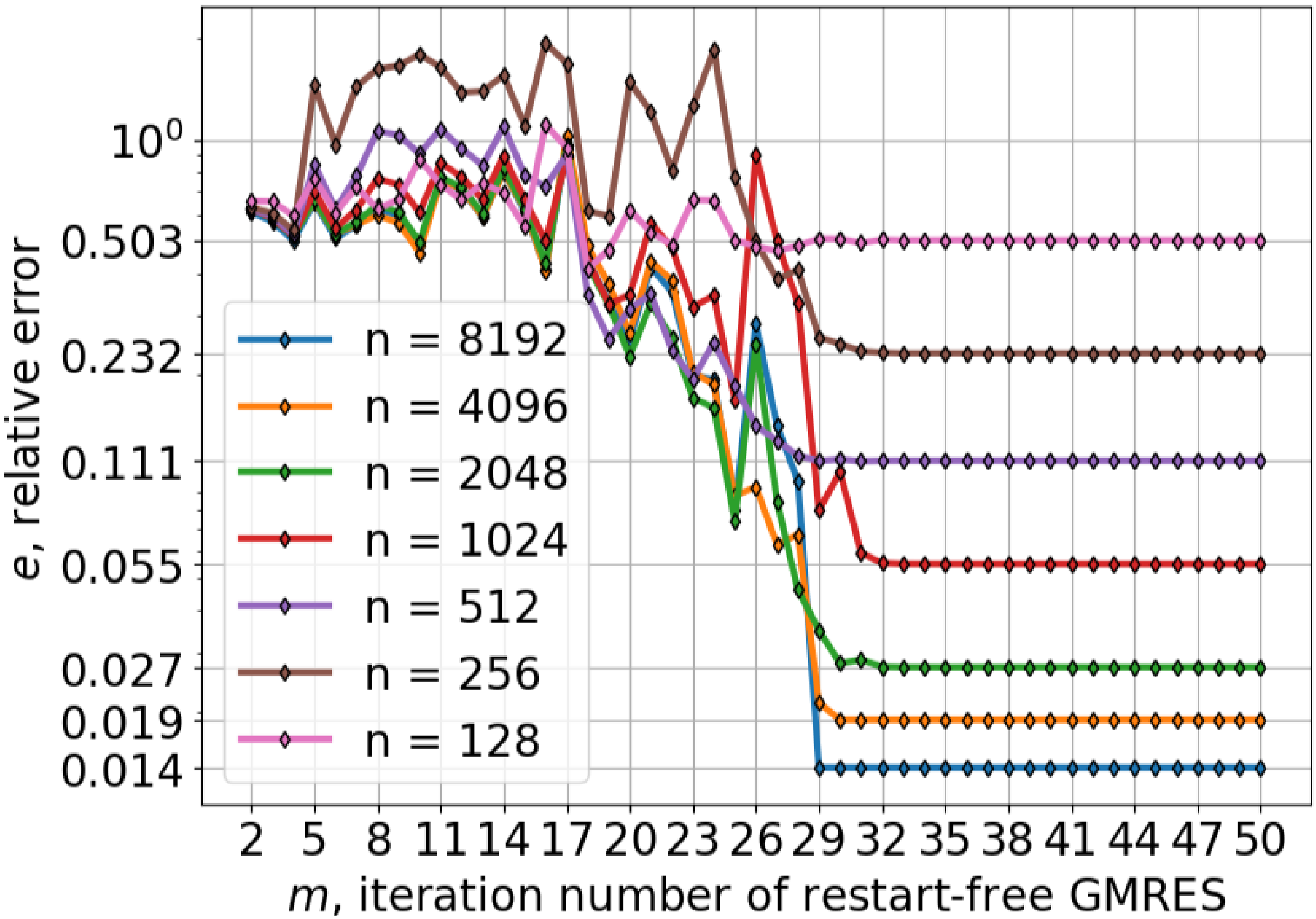
$$
\begin{cases}
\triangle u(r) + k_0^2\, u(r) = 0, \quad r \in \mathbb{R}^2 \backslash \Omega; \\
u(r) = 0, \qquad\qquad r \in \partial\Omega : \textit{Dirichlet boundary c.}; \\
\lim\limits_{|r| \to \infty} (\mathbf{i} k_0 (u - w)(r) - \partial_{|r|} (u - w)(r)) |r| = 0 \\
\qquad\qquad\qquad\qquad : \textit{Sommerfeld radiation c.};
\end{cases} \tag{5}
$$

where $\triangle$ is Laplace operator, r is space vector and k_0 is a wave number [46]. The Helmholtz equation solution can be represented for uniform grid of square two-dimensional space Ω through two-level Toeplitz matrix $H^{(2)} \in \mathbb{C}^{n^2 \times n^2}$ [46, 31, 38] in linear system matrix A:

$$
A\ x = I\ x - k_0^2\ (\epsilon - \tilde{\epsilon})\ H^{(2)}\ m * x, \tag{6}
$$

where $x \in \mathbb{C}^{n^2}$ is electric field distribution for uniform grid $\tilde{r} \in \mathbb{R}^{n \times n}$ of Ω, I is the identity matrix, operator "$*$" is Kronecker product and $m \in \{0;1\}^{n^2}$ is a binary mask of photonic component material distribution in the design domains. This is detailed in the previous work [38], where we utilized FFT for multiplication of Toeplitz matrix $H^{(2)}$ by vector. In addition, number of computations was reduced by using two-level Toeplitz matrix surrogate $G \in \mathbb{C}^{(2\ n-1) \times (2\ n-1)}$ [13, 38]. Number of Ω nodes is equal to n^2.

```
function matvec_T(T, x)
    T^e ← extend_T(T)
    x^e ← extend_vec(x)
    y^e ← IFFT (FFT (T^e) * FFT (x^e)): Kronecker product.
    return y^e
```


e, relative error
10^0
0.503
0.232
0.111
0.055
0.027
0.019
0.014
2 5 8 11 14 17 20 23 26 29 32 35 38 41 44 47 50
m, iteration number of restart-free GMRES
n = 8192
n = 4096
n = 2048
n = 1024
n = 512
n = 256
n = 128


$r_{i^* j^*}$
ω
$E(r_{ij})=e^{-ikr_{ij}}$

Methods (Part I)

- - Green's Function Integral Equation Method (GFIEM)
- - Toeplitz matrices for efficient matrix-vector operations
- - FFT acceleration: $O(n \log n)$ vs $O(n^2)$

💡 **GMRES(k) Advantage: Restarts limit Krylov subspace size, reducing memory usage.**

Methods (Part II) – GMRES

- - GMRES solver constructs a Krylov subspace
- - Memory consumption grows with Krylov vectors
- - GMRES(k) with restarts limits subspace size
- - Restart strategy balances memory vs iterations
- 💡 Enables very large grids without MPI or cluster expansion

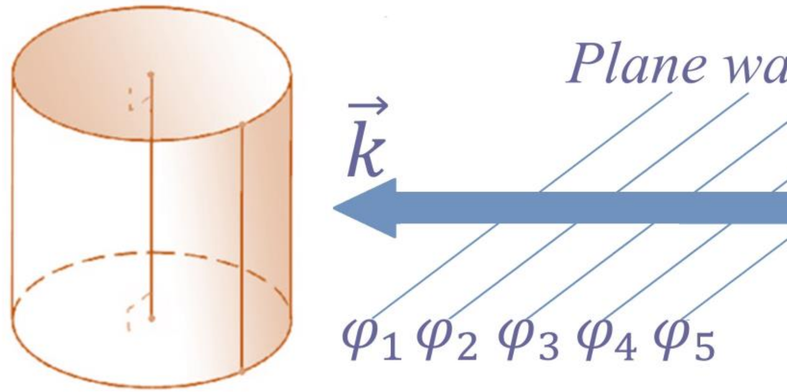
Numerical Experiments

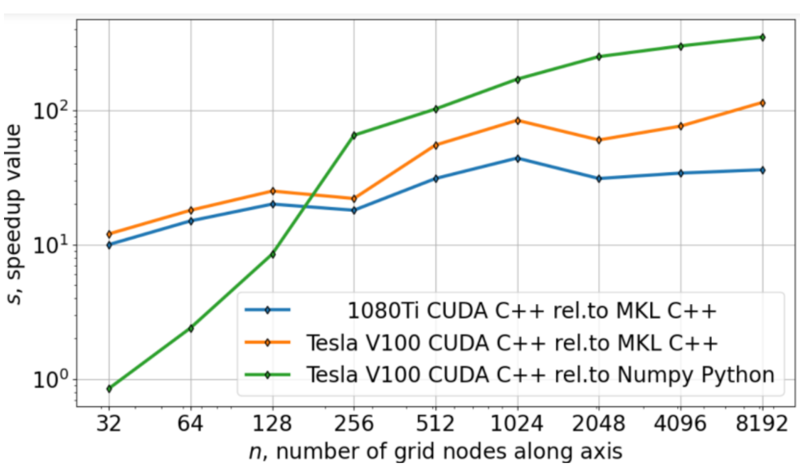
- \- Problem: Gustav Mie scattering on a cylinder
- \- Grid sizes: 128×128 to 8192×8192
- \- Accuracy target: residual ≤ 0.01
- \- Platforms: CUDA GPU vs. CPU (MKL, Python)

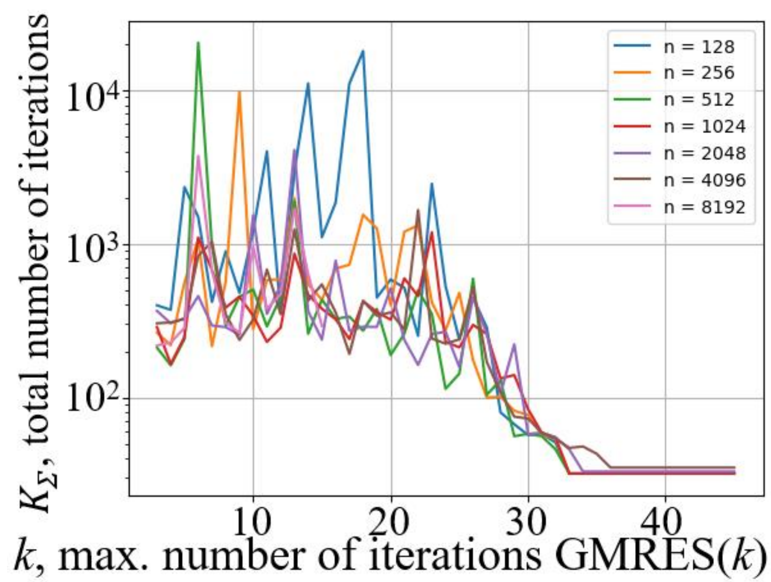
Results (Part I) – Computation Time

- - GPU acceleration yields large speedup:
-   • Up to ×114 vs MKL C++
-   • Up to ×350 vs NumPy Python
- - Efficiency increases with grid size
- - Restarted GMRES reduces memory ~10×
- Toeplitz matrix instead of dense matrix

: $\mathcal{O}(n \log n)$ instead of $\mathcal{O}(n^2)$

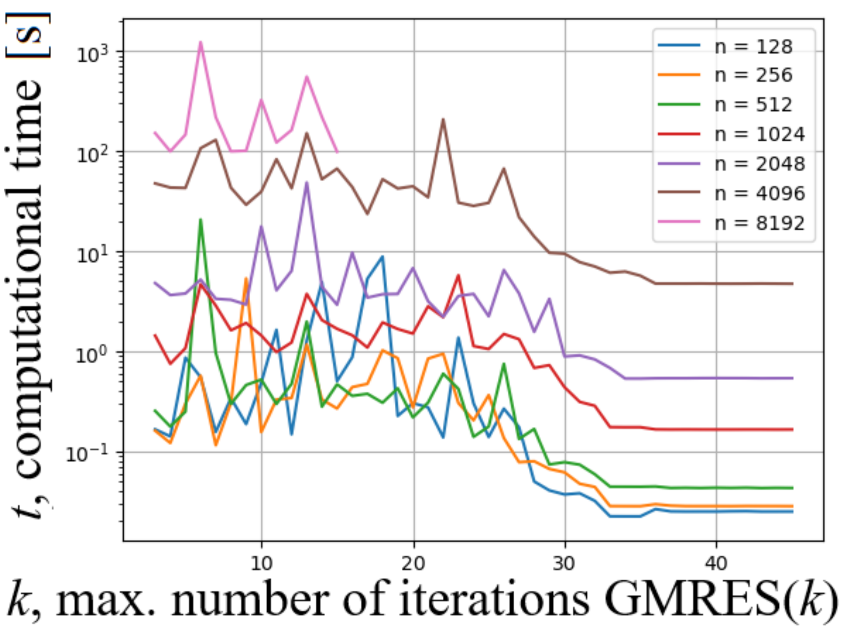


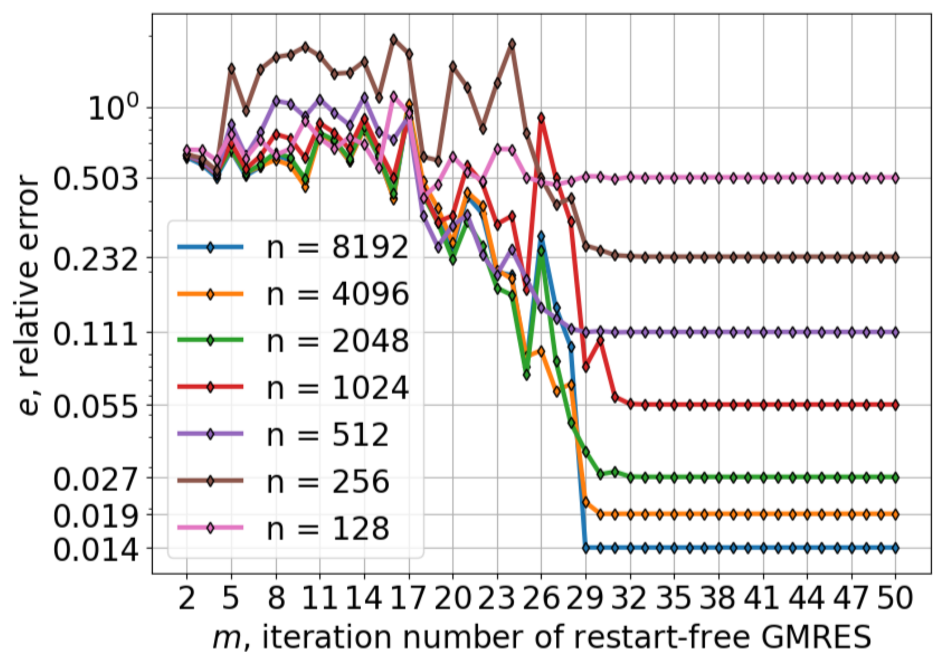




Results (Part II) – Iterations & Errors

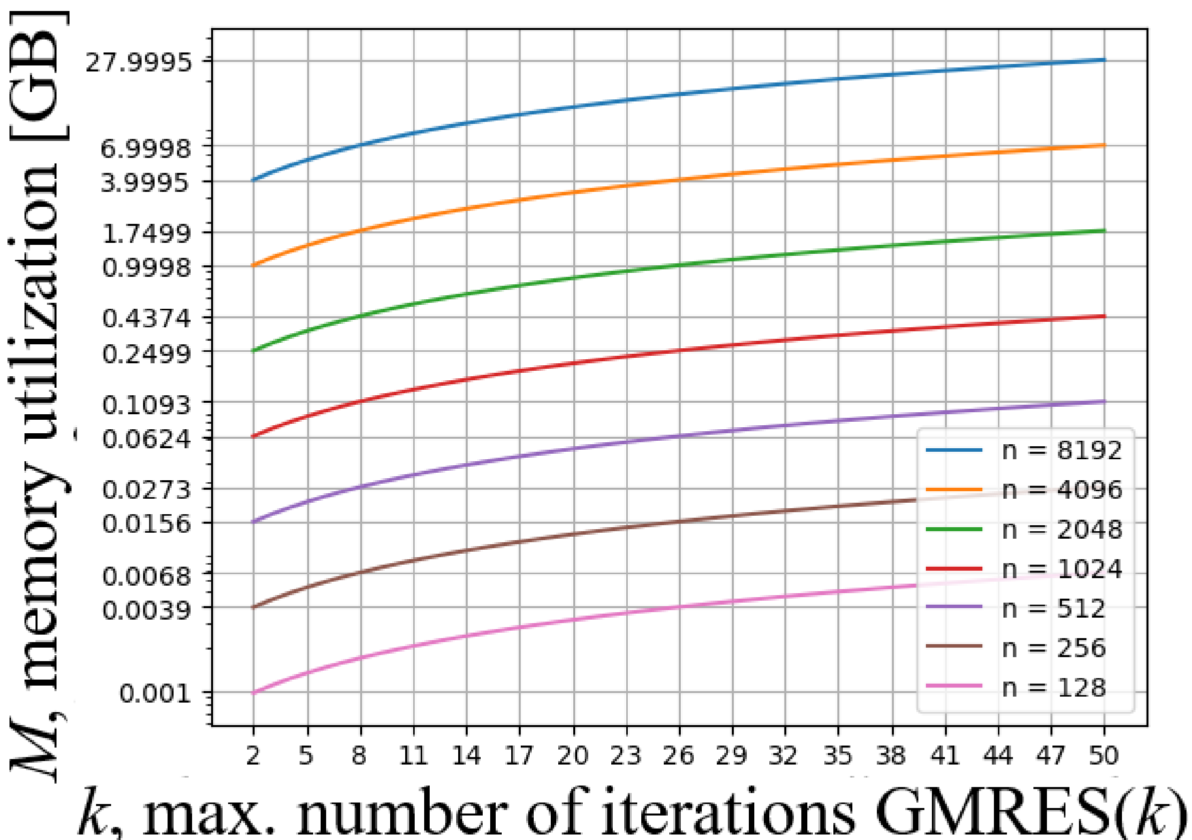
- - For k ≥ 35, convergence without restarts (small grids)
- - Larger grids require restarts (memory overflow)
- - Relative error stabilizes after ~23 iterations





Results (Part III) – Memory Usage

- - Memory grows quadratically with grid size
- - GPU overflow at 8192×8192 without restarts
- - Restart strategy makes large grids feasible



Applications Spotlight

- - TE/TM polarization splitter
- - Wavelength splitters
- - Bragg reflectors
- - Fiber Bragg gratings
- - Photonic IC optimization
- 👉 Shows direct impact of efficient solver on device design

Conclusions in brief

- - Fast: GPU acceleration + FFT + Toeplitz
- - Memory-Efficient: GMRES(k) with restarts
- - Scalable: handled grids up to 8192×8192
- - Practical: applied to photonic devices and IC design

Conclusions

- - Efficient FFT-accelerated GMRES solver developed
- - Trade-off: memory savings ↔ more iterations
- - GPU parallelization crucial (best on NVIDIA V100)
- - Future: non-uniform grids (higher accuracy, but may break Toeplitz optimization)
- - GMRES(k) restarts enabled larger problem sizes
- without expanding hardware or using MPI
- - Measurements showed up to ~2× task deployment increase
- - GMRES with restarts may diverge when using incremental Krylov bases
- - Smaller surrogate matrices maintain computational accuracy
- - Non-uniform grids could enhance accuracy but complicate Toeplitz system setup

💡 **GMRES(k) Advantage: Enabled ~2× larger problem sizes without extra hardware or MPI.**

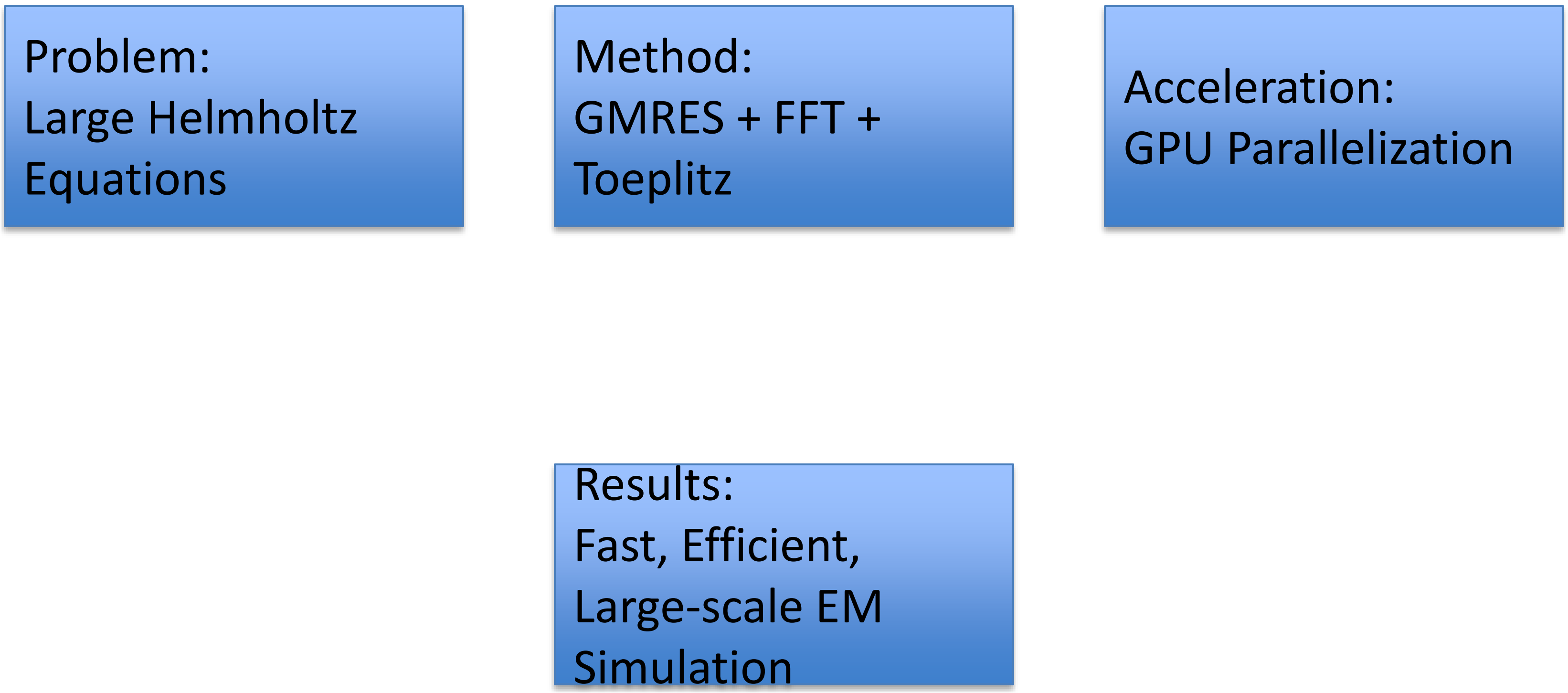
Acknowledgments

- - Funded by Russian Science Foundation project 25-11-00392
- - Support from state task of Fryazino branch of Kotelnikov IRE of RAS

Executive Summary

- • Problem: Solving large-scale Helmholtz equations for EM fields is computationally expensive
- • Approach: FFT-accelerated GMRES solver with Toeplitz matrices + GPU parallelization
- • Challenge: Memory usage grows with Krylov subspace size
- • Solution: GMRES(k) restarts → ~10× lower memory, enabling larger grids
- • Results: Up to 114× faster (vs CPU), 350× faster (vs Python); feasible on grids up to 8192×8192
- • Key Impact: ~2× larger task deployment without extra hardware or MPI
- • Applications: Design of photonic components, IC optimization

Methodology Flow Diagram



Summary

- - Developed an FFT-accelerated GMRES solver for 2D Helmholtz problems
- - GMRES requires Krylov subspace; restarts (GMRES(k)) reduce memory ~10×
- - Restart strategy enabled grids up to 8192×8192 on GPUs
- - Trade-off: lower memory vs. more iterations
- - GPU acceleration (V100) provided up to 114× faster vs CPU, 350× vs Python
- - Applications: TE/TM splitters, wavelength splitters, Bragg reflectors, IC optimization
- - Future work: non-uniform grids (higher accuracy, but Toeplitz limitations)

Fast & Efficient Electromagnetic Field Computations with GMRES(k) and GPU Acceleration

Problem
- Large Helmholtz equations for EM fields
- High computational & memory costs

Method
- GMRES solver with Krylov subspace
- GMRES(k) restarts to limit memory
- FFT + Toeplitz matrices
- GPU parallelization

Results
- 10× lower memory with restarts
- Up to 114× faster (vs CPU)
- Up to 350× faster (vs Python)
- Grids up to 8192×8192 feasible

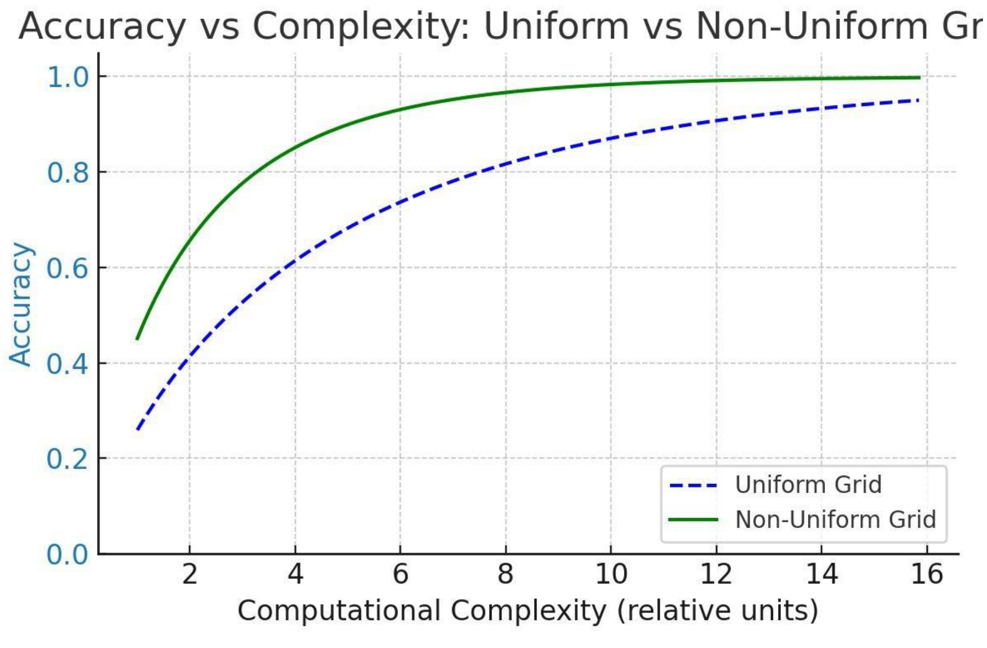
Impact
- ~2× larger task deployment without MPI or new hardware
- Applications: Photonic design, IC optimization

Performance Comparison

- CPU (MKL C++): baseline
- GPU (CUDA V100): up to 114× faster
- Python (NumPy): 350× slower than GPU
- GMRES(k) with restarts: ~10× lower memory, 2× larger tasks

Future Work

- - Non-uniform grids may improve accuracy
- - But break Toeplitz structure (no FFT acceleration)
- - Requires new approaches for efficiency



Results – Extended

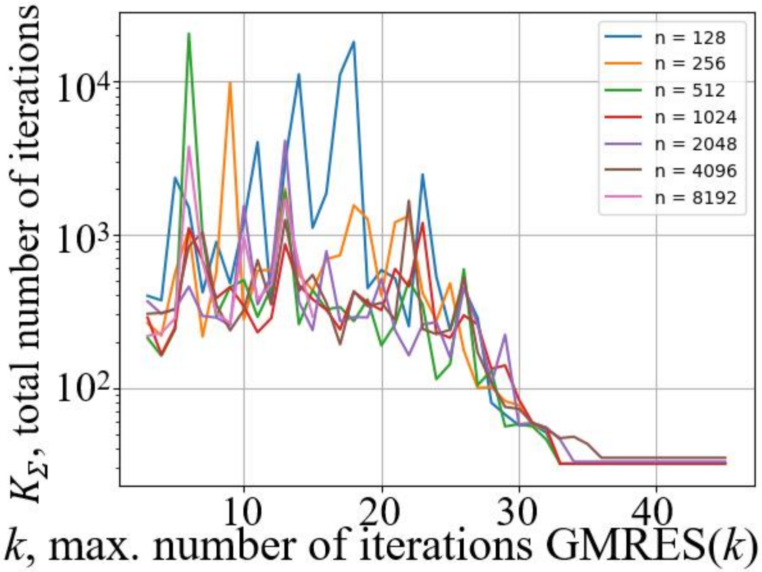


Fig. 7 – Computation time vs grid size

Results – Extended

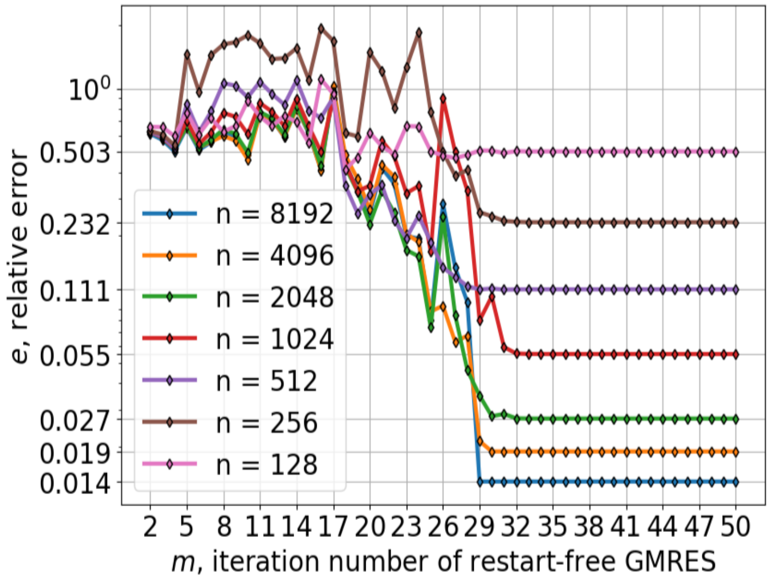


Fig. 6 – Iterations vs grid size

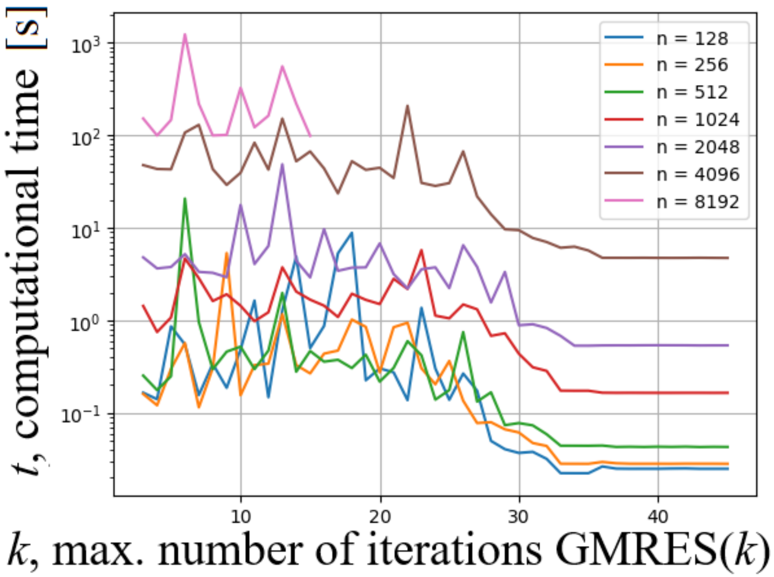


Fig. 5 – Error behavior with iterations

Results – Extended

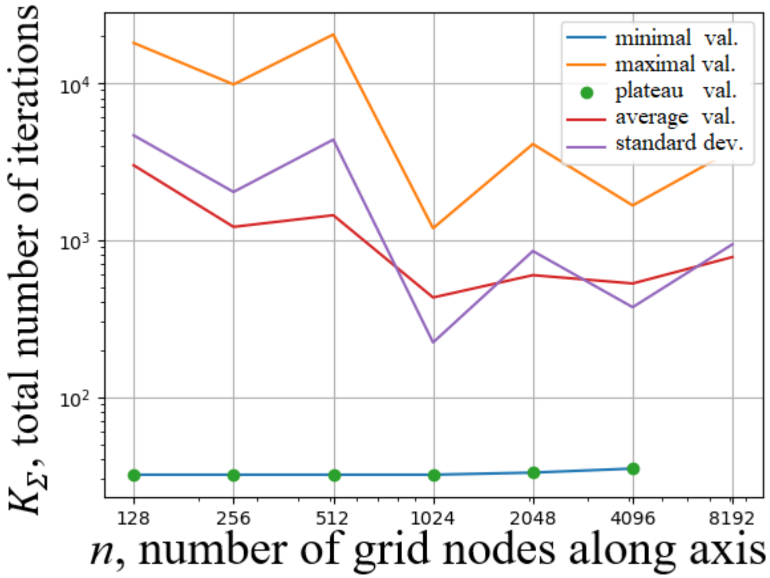


Fig. 8 – Memory usage vs grid size

Appendix – Additional Figures