Multilevel Probing Methods for Approximating Selected Entries of the Matrix Inverse

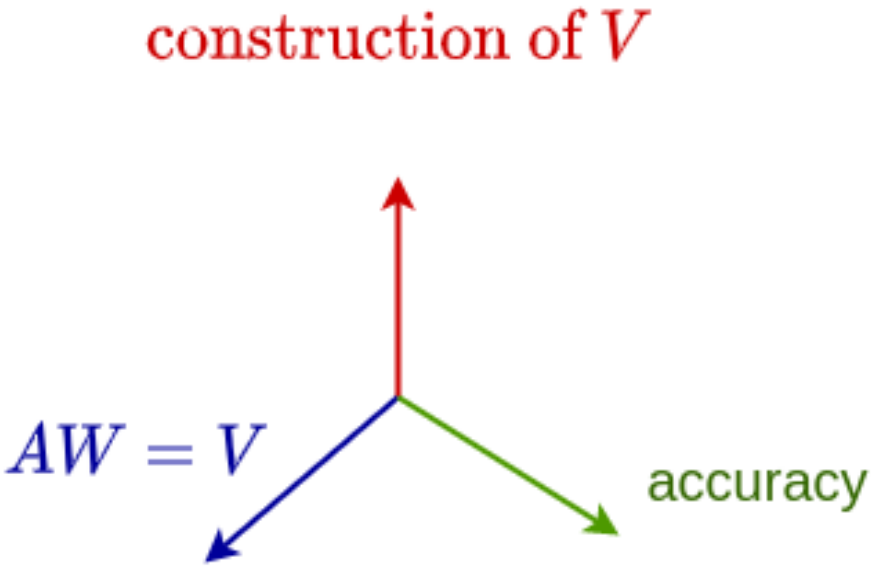
Vasileios Georgiou

Computer Engineering and Informatics Department
University of Patras

June 24, 2021

Goals

Approximate diag(A^{-1}) via MV products with orthogonal basis V



1. memory and #operations for constructing $V \downarrow$ and #RHS $\uparrow$
2. approximation of the diagonal blocks and off-diagonal entries
3. solvers for MRHS & sequences of linear systems

*Numerical library for large scale computation for manycore + GPUs

The Selected Inversion Problem

Queries (Q) on entries of A^{-1}

Exact Selection

Q_1: For $A \in \mathbb{C}^{n \times n}$ and subset Z, return $(A^{-1})_{ij}$ for all $(i,j) \in Z$

Approximate Selection: First define $T_Z : \mathbb{C}^{n \times n} \to \mathbb{C}^{n \times n}$

$$(T_Z(A))_{ij} = \begin{cases} (A)_{ij}, & \text{if } (i,j) \in Z \\ 0, & \text{otherwise} \end{cases}$$

Q_2: For $n \times n$ matrix A and the index subset Z, return $(B)_{ij}$ for all $(i,j) \in Z$ such that $\|B - T_Z(A^{-1})\| \leq \epsilon$

Special Case: $\text{diag}(A^{-1})$

Applications

- Quantum transport simulations



- Uncertainty quantification

$$\min \|Yz - b\|_2 \longrightarrow (C^{-1})_{ii} \text{ confidence on i-th sample}$$

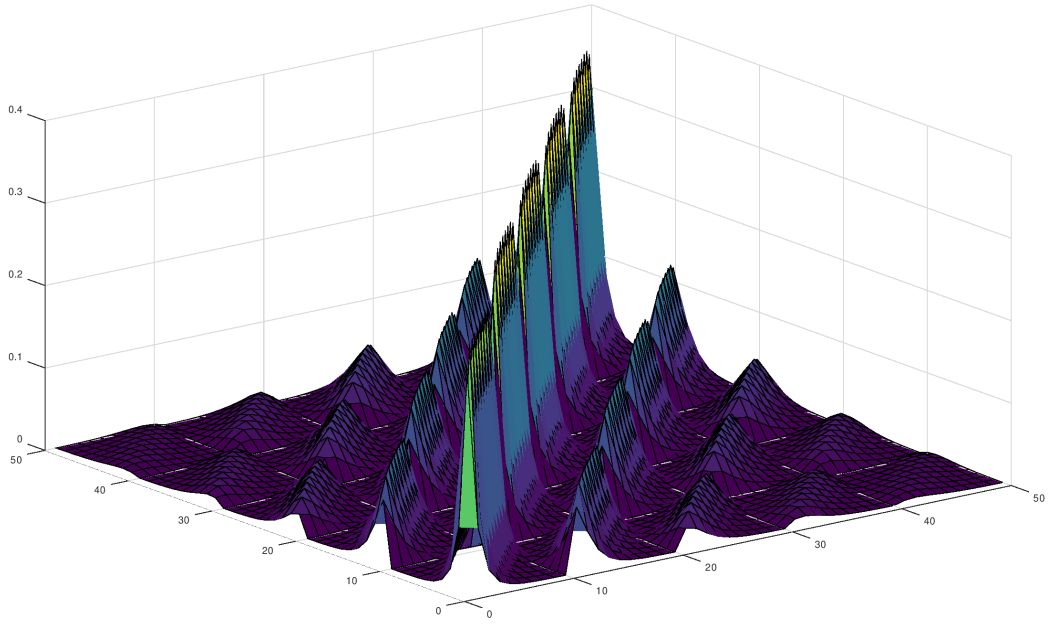
- Covariance estimation

$$f_\lambda(\Theta) = \underbrace{-\log(\det(\Theta)) + \mathrm{tr}(S\Theta)}_{\text{log-likelihood}} + \underbrace{\lambda|\Theta|_1}_{\text{regularizer}}$$

- Network analysis

$$f(A) = \alpha_0 I + \alpha_1 A + \alpha_2 A^2 + \ldots \alpha_{p-1} A^{p-1} \text{ where } A = I - \alpha G$$

Localization



"Sparsity does not take into account the size of the nonzeros, all being treated as equal" *M. Benzi*

$$\|A^{-1} - T_\delta(A^{-1})\| \leq \epsilon \text{ and } \mathrm{nnz}(T_\delta(A^{-1})) \leq N_\delta$$

T_δ: thresholding operator, N_δ: maximum number of nonzeros

Probing methods utilize significant structure $S(T_\delta(A^{-1}))$

Extracting diag(A^{-1}) with Probing Vectors

Probing method by [Tang and Saad, 2011] motivated by [Coleman and More, 1983] proposed method for diag(A^{-1})

$$S(T_\delta(A^{-1})) \approx (\wedge_{i=1}^{m} S(A))$$

probing vectors v_i tailored to extract diag(A^{-1}) generated by coloring graph with adj.matrix $\wedge_{i=1}^{m} S_A$

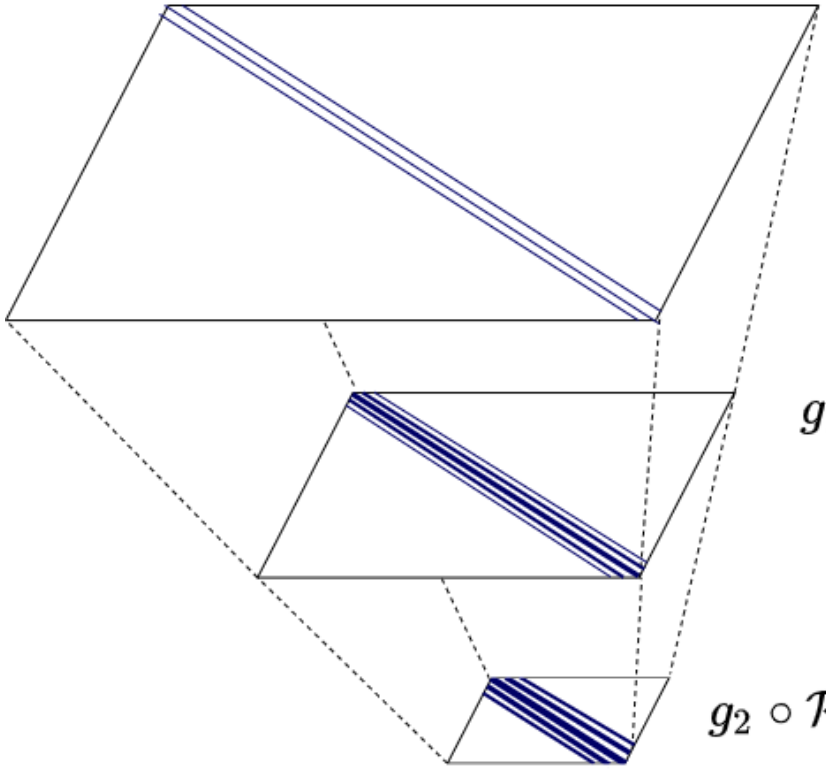
$$diag(A^{-1}) \approx \sum_{i=1}^{s} v_i \odot (A^{-1} v_i)$$

Error depends on entries not captured by $\wedge_{i=1}^{m} S(A)$

+ **Exact result for A^{-1} with the same sparsity pattern as A^k**
- **Requires computing and storing $\wedge_{i=1}^{m} S_A$, m can be large**

Related methods [Stathopoulos, Laeuchli, orginos 2013], [Laeuchli, Stathopoulos, Orginos 2020]

Multilevel Probing



$S(A)$

$S(A)$: nonzero patter of A
R_i : coarsening operator
g_i : expansion operator

$g_1 \circ \mathcal{R}_1(S(A))$

$g_2 \circ \mathcal{R}_2(g_1 \circ \mathcal{R}_1(S(A)))$

$$S(T_\delta(A^{-1})) \approx \underbrace{\wedge_{i=1}^{k} S(A)}_{\text{probing for } k=\xi_1\xi_2} \approx \overbrace{P}^{\text{prolong}} \circ \Bigg(\underbrace{\wedge_{i=1}^{\xi_2}(\mathcal{R}_2 \circ (\underbrace{\wedge_{i=1}^{\xi_1}(R_1 \circ S(A))}_{\text{level 1}}))}_{\text{level 2}} \Bigg)$$

General Algorithm

Algorithm 1 Multilevel Probing

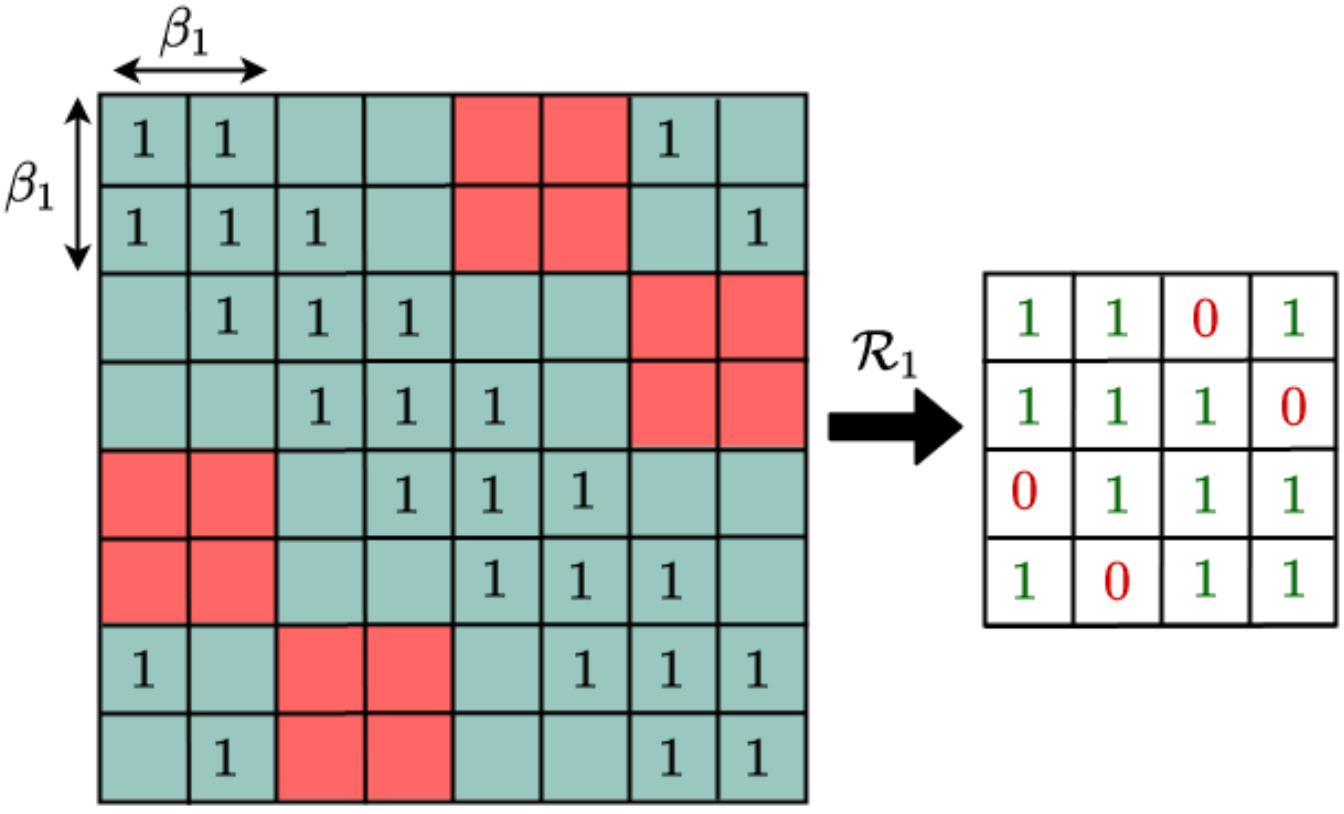
Input: $A \in \mathbb{C}^{n \times n}$, β, k, m

Output: d

1: Initialize $S_0 = S(A)$, $c = [\,]$
2: **for** $i = 1$ to p **do**
3: compute $\hat{S}_k \leftarrow R_i \circ S_{k-1}$ // coarsening
4: compute $S_k \leftarrow \wedge_{i=1}^{\xi_i} \hat{S}_k$ // expansion
5: $c \leftarrow \text{color}(S_p)$
6: generate right-hand sides V // vector-prolongation
7: solve $AW = V$
8: compute $d = \sum_{i=1}^{s\beta} W_i \odot V_i$

pros: Preserve $\text{nnz}(S_k) \leq \text{nnz}(S_0) = \text{nnz}(S(A))$ for all k
cons # RHS $\uparrow \Rightarrow$ amount of time spent on solving linear systems

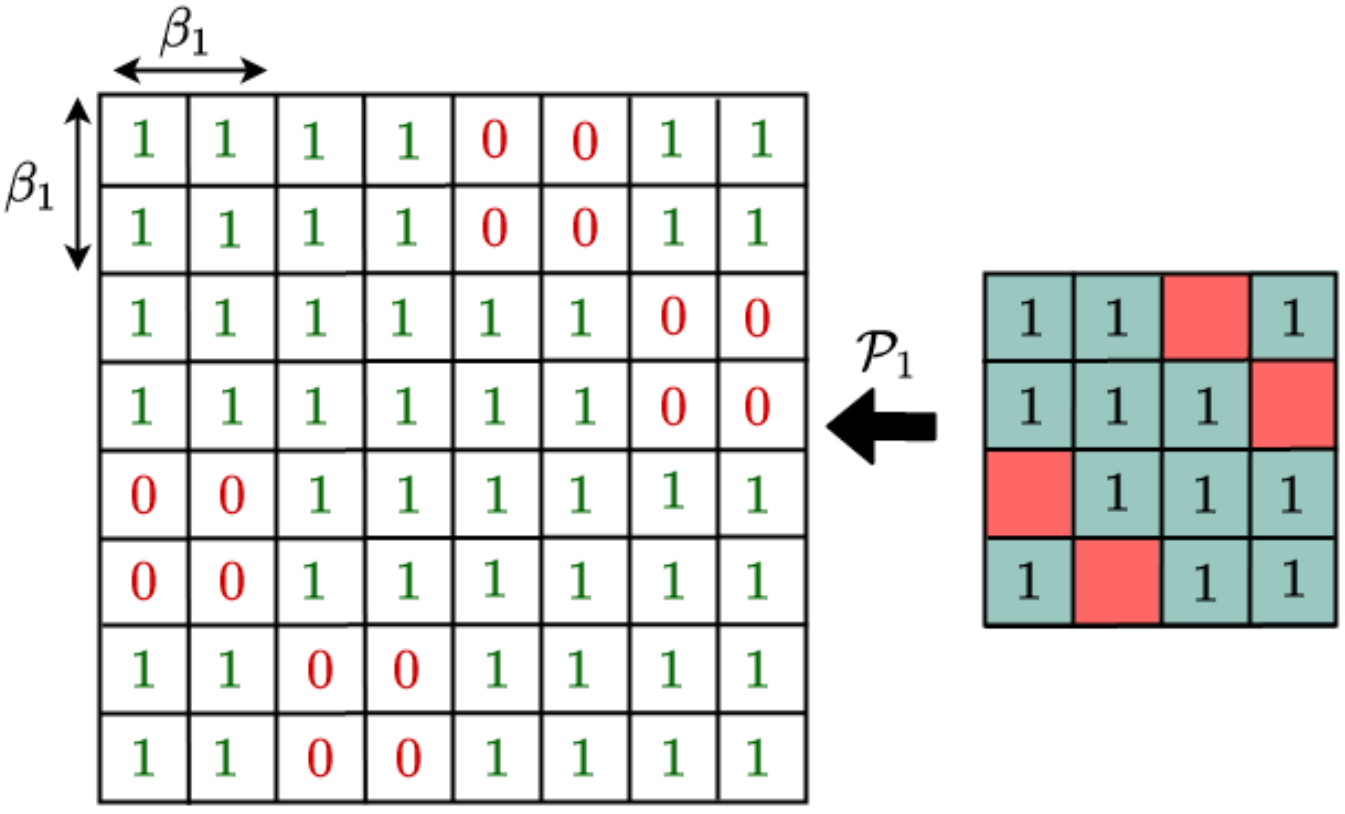
Coarsening Operator - Blocking



coarsening operator $\mathcal{R} : \{0,1\}^{n \times n} \to \{0,1\}^{n/\beta_1 \times n/\beta_1}$

$$(R(A))_{ij} = \begin{cases} 1 \text{ if } \; A_{IJ} \neq 0_{\beta \times \beta} \\ 0 \text{ else} \end{cases}$$

Prolongation Operator - Blocking



prolongation operator $\mathcal{P} : \{0,1\}^{n/\beta_1 \times n/\beta_1} \to \{0,1\}^{n \times n}$

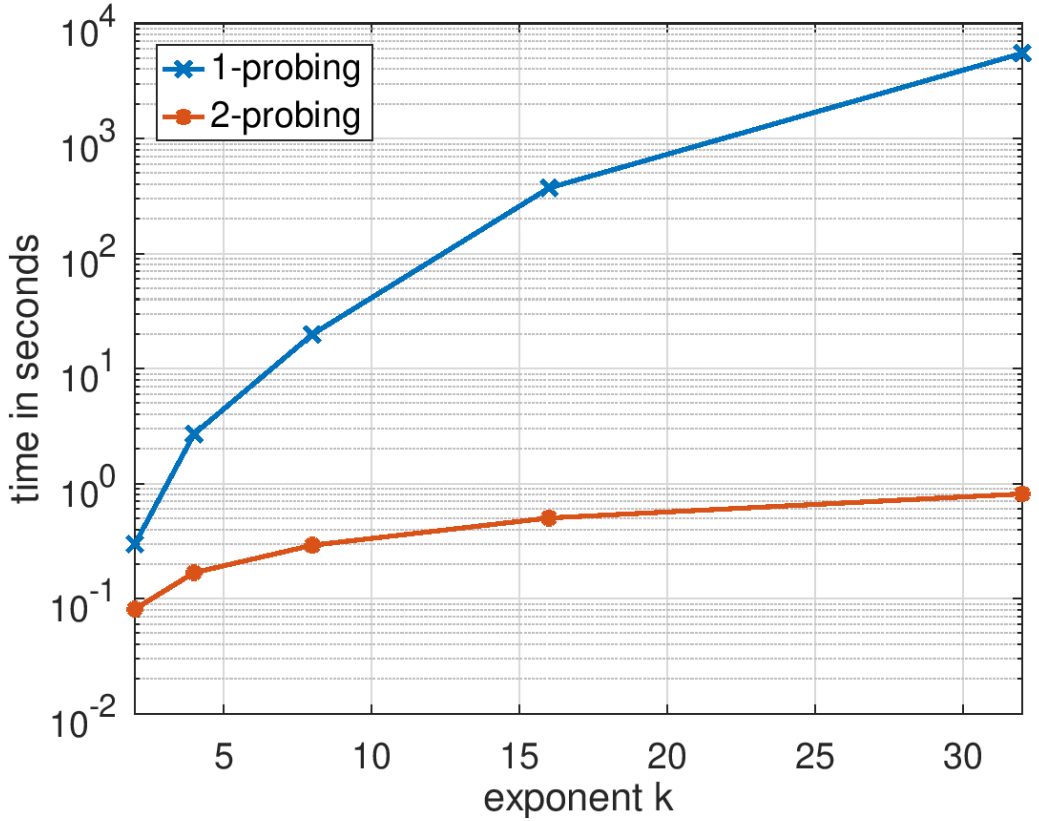
$$\mathcal{P}(A) = A \otimes ee^\top, \; (P(A))_{ij} = \begin{cases} 1 \\ 0 \end{cases}$$

Block-probing vs Probing - probing runtimes

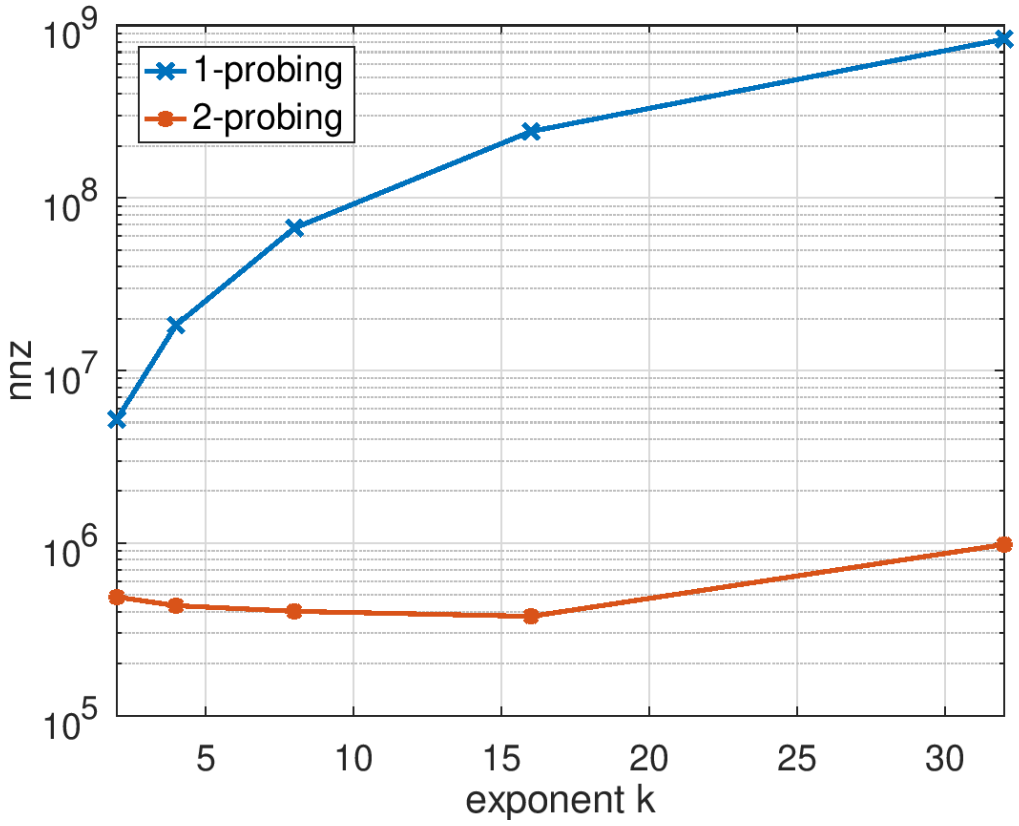
65356 $\times$ 65356 covariance matrix generated from

$$(A)_{ij} = \begin{cases} \left(1 - \frac{\text{dist(i, j)}}{\alpha}\right)^{\beta}, & \text{if dist}(i,j) \leq \alpha \\ 0, & \text{otherwise} \end{cases}$$

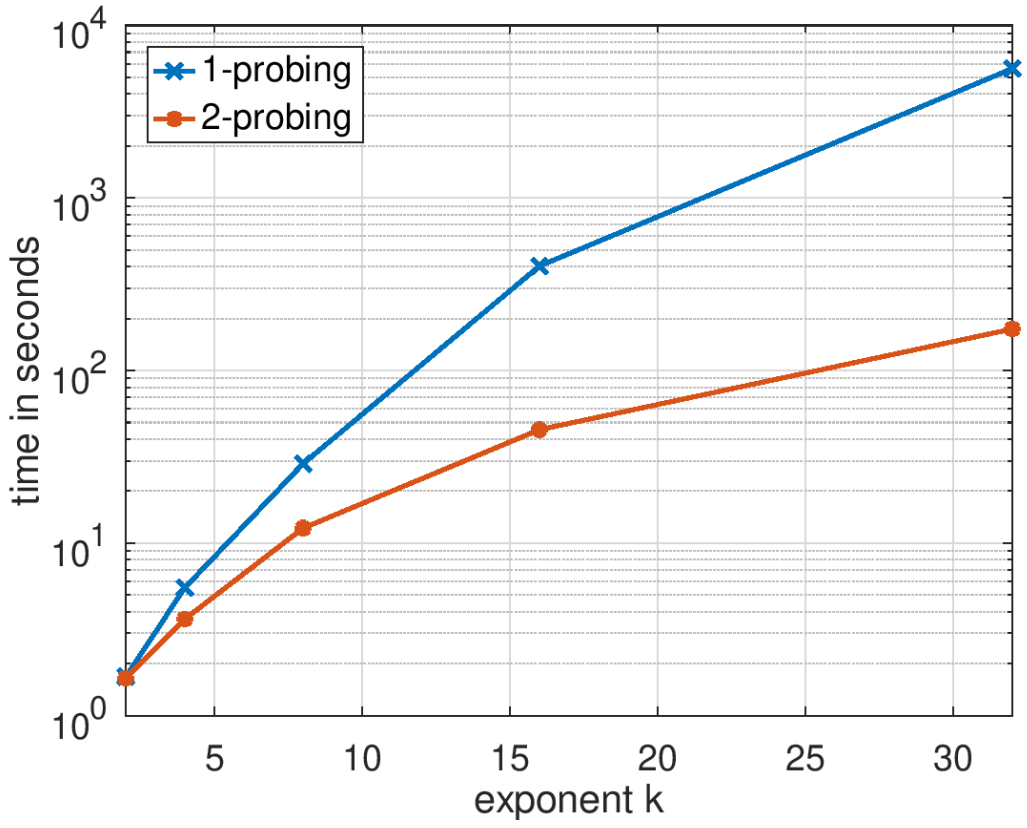
for $\alpha = 3$, $\beta = 5$. Use PCG with Jacobi preconditioning & 25 iterations



Block-probing vs Probing - number of nonzeros



Block-probing vs Probing - total runtimes



Matrices with localized inverse usually expected to be well conditioned. Several results suggest that decay rate $\sim \kappa(A)$

Error

Let vectors $d, \hat{d}, d^*$ be

- $d \leftarrow \text{diag}(A^{-1}) = (A^{-1} \odot I)e$: exact values
- $\hat{d} \leftarrow (A^{-1}V)e$, as computed by numerical solver
- $\tilde{d} \leftarrow (A^{-1}V)e$, for exact solution of $AW = V$

Relative error $\mathcal{E}$ bounded as follows:

$$\mathcal{E} = \left\| (\hat{d} - d) \oslash d \right\|_\infty \leq \underbrace{\left\| (\hat{d} - \tilde{d}) \oslash d \right\|_\infty}_{\mathcal{E}_{\text{solve}}} + \underbrace{\left\| (\tilde{d} - d) \oslash d \right\|_\infty}_{\mathcal{E}_{\text{probe}}}$$

Necessary to bound both terms and consider tradeoffs

$$\mathcal{E}_{\text{probe}} = \left\| [(A^{-1} \odot VV^\top)e] \oslash [(A^{-1} \odot I)e] - e \right\|_\infty$$

Block-probing vs Probing - Error

For symmetric $n \times n$ matrix A and hierarchy $H(\beta, k, p) = \{S^{(i)}_{\beta,k}\}$ generated by the block coarsening

$$\mathcal{E}_{\text{probe}}(S^{(p)}_{k,\beta}) \leq \mathcal{E}_{\text{probe}}(\wedge_{i=1}^{k^p} S(A))$$

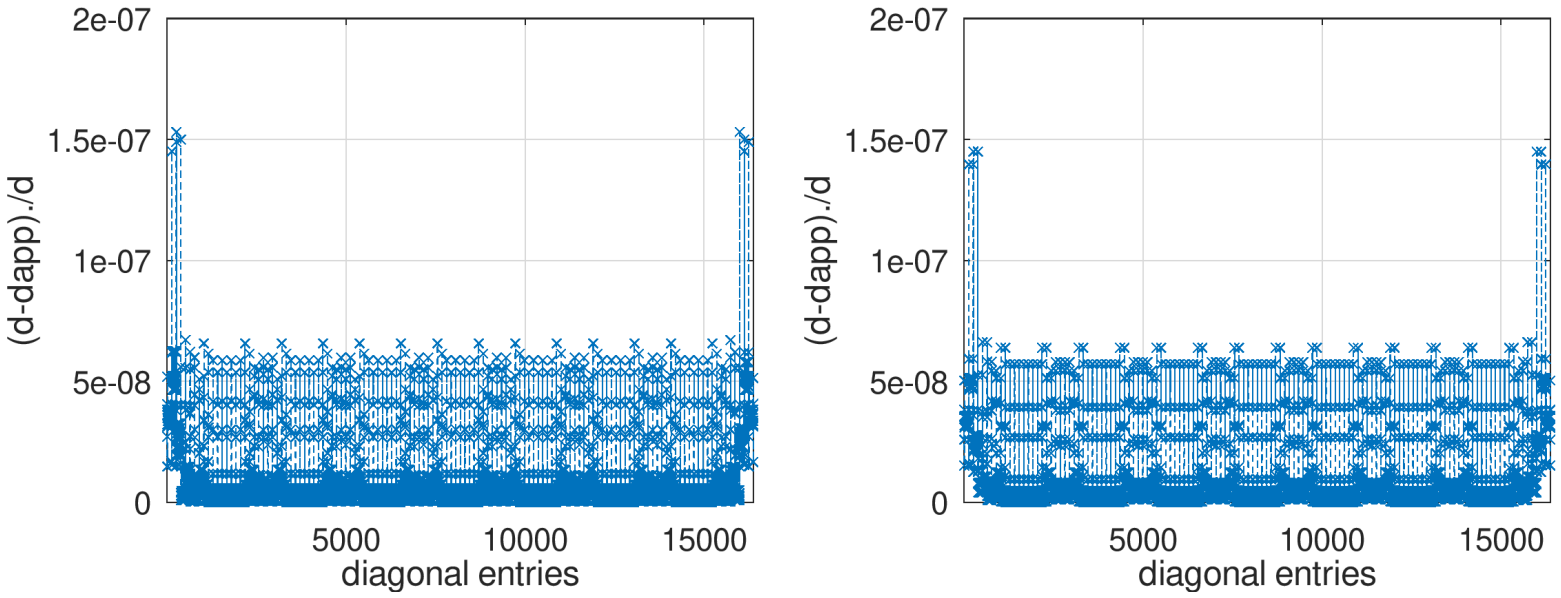


Figure: On the left: (1, 3)-probing. On the right (2, 3)-probing

Theorem (Benzi and Razouk)

Let $A \in \mathbb{C}^{n \times n}$ a diagonalizable matrix and $f(A)$ exhibits decay away from an nonzero pattern described by

$$|(f(A))_{ij}| \leq \kappa(X) C \lambda^{dist(i,j)} \text{ where}$$

$0 \leq \lambda < 1$, $\kappa(X)$ *the condition number of the diagonalizer matrix X and C a constant.*

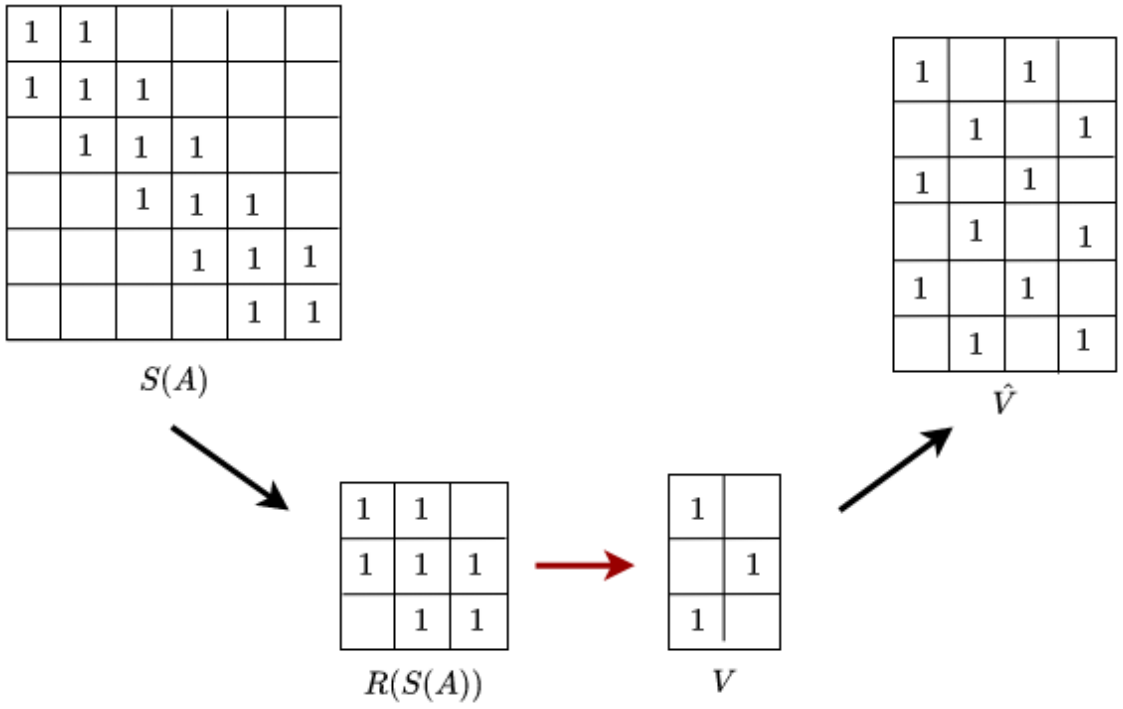
If the number of z-distant neighbors of every node of G_A is given by $N_z = \alpha z$ for constant α and all $z = 1, \ldots, \text{diam}(G_A)$, and probing based on $\wedge_{i=1}^{p} S(A)$, then

$$\mathcal{E}_{\text{probe}} \leq \kappa(X) C \frac{(1+p)\lambda^{p+1} - p\lambda^{p}}{|f_{\text{min}}|(1-\lambda)^2} \text{ where}$$

f_{min}, C and λ follow from the theorem by [Benzi and Razouk].

Approximating the diagonal blocks of A^{-1}

For A with sparsity structure $S(A)$



The diagonal blocks can be extracted exatly from the equation

$$d_\beta(A) = \sum_i (A^{-1} V_i) \bullet V_i \text{ where } \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \bullet \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} = \begin{bmatrix} X_1 Y_1 \\ X_2 Y_2 \end{bmatrix}$$

Approximating off-diagonal entries (I)

Proposition (Coleman & More, 1983)

All nonzeros A are contained in columns of $W = AV$. Vectors $V_{:,i}$ formed by coloring graph with adjacency matrix $\wedge_{i=1}^{2} S(A)$

Since $\wedge_{i=1}^{k} S(A) \approx S(T_\delta(f(A)))$ and

$$\wedge_{i=1}^{2}(\wedge_{j=1}^{k} S(A)) \odot \left[P_{\beta^{k+1}} \circ \overbrace{(\underbrace{\wedge_{i=1}^{2}(R_\beta}_{\text{extra step}} \circ \underbrace{S_{\beta,k}^{p}}_{p \text{ level}}))}^{G} \right] = \wedge_{i=1}^{2}(\wedge_{j=1}^{k} S(A))$$

$$V \leftarrow \text{color}(G)$$

Filtering Problem:

1) Extracting nonzero $(P_{\beta^{k+1}} \circ S_{\beta,k}^{p+1})_{ij}$ can require a lot of memory
2) Slow decay can lead to noise terms being evaluated

Solution: fill entries of $A^{-1}V$ based on $\wedge_{i=1}^{\xi} S(A)$ where $\xi \ll p$

Approximating off-diagonal entries (III) - Reconstruction

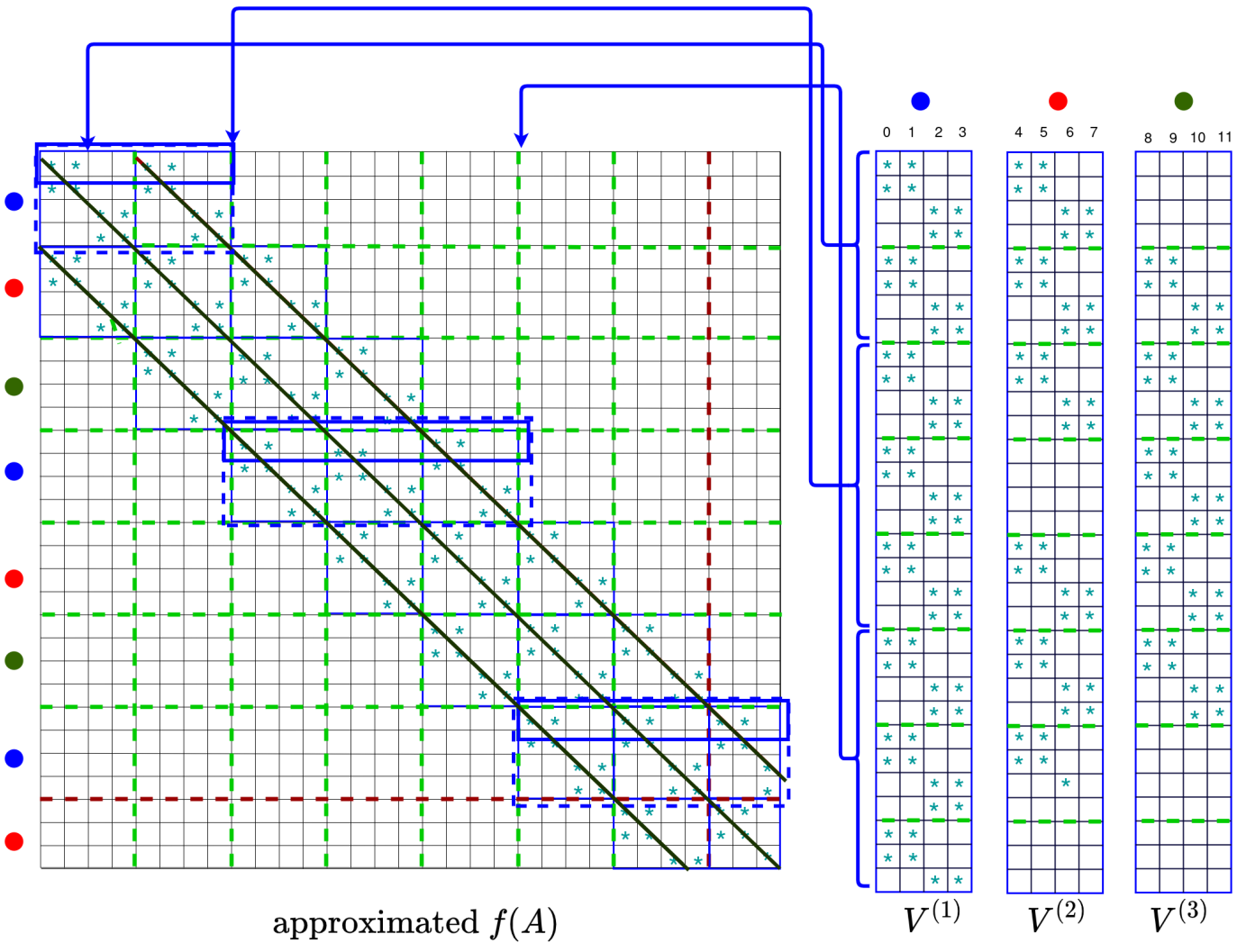


Figure: reconstructing entries of approximate inverse

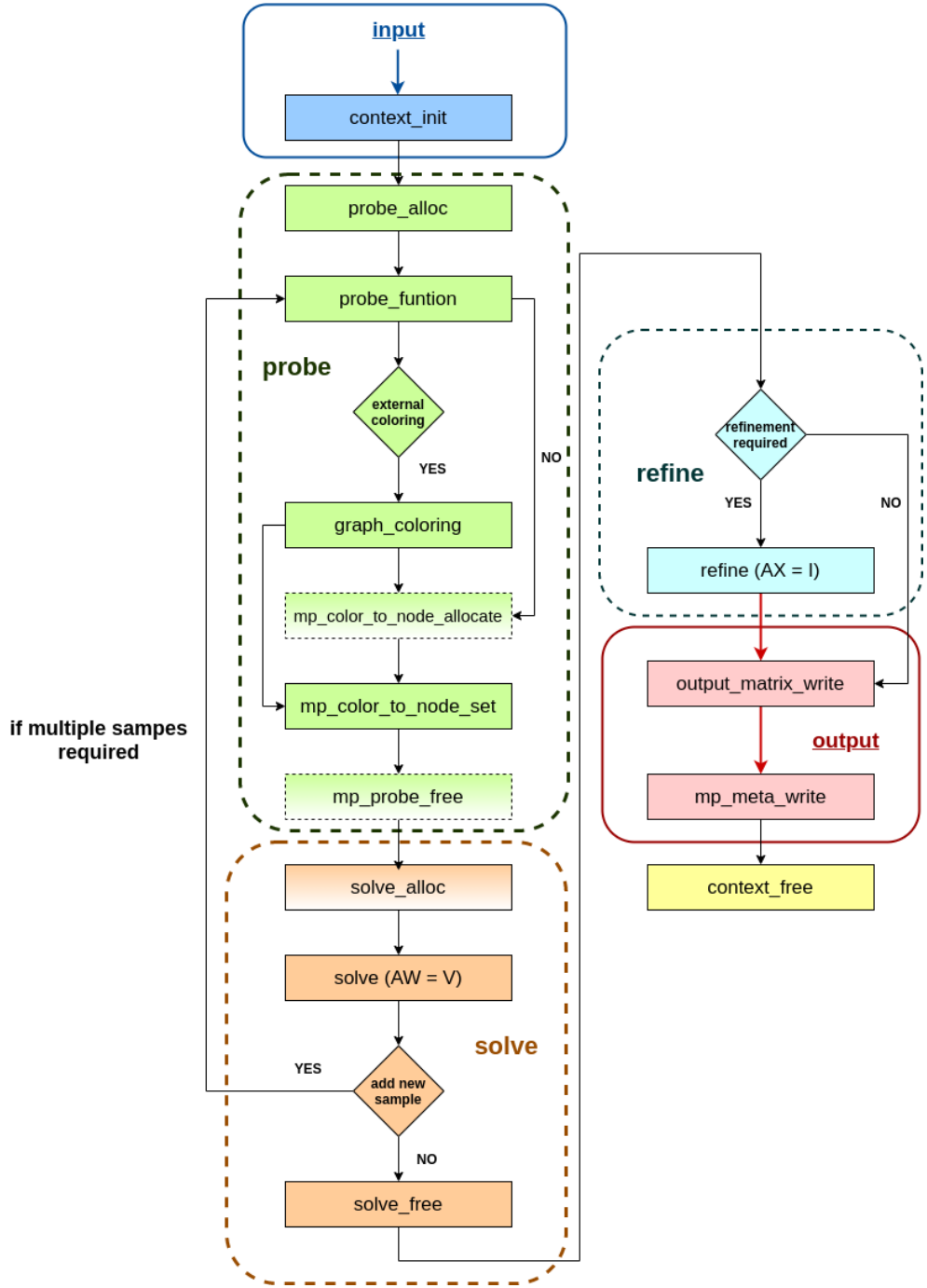
Software & numerical experiments

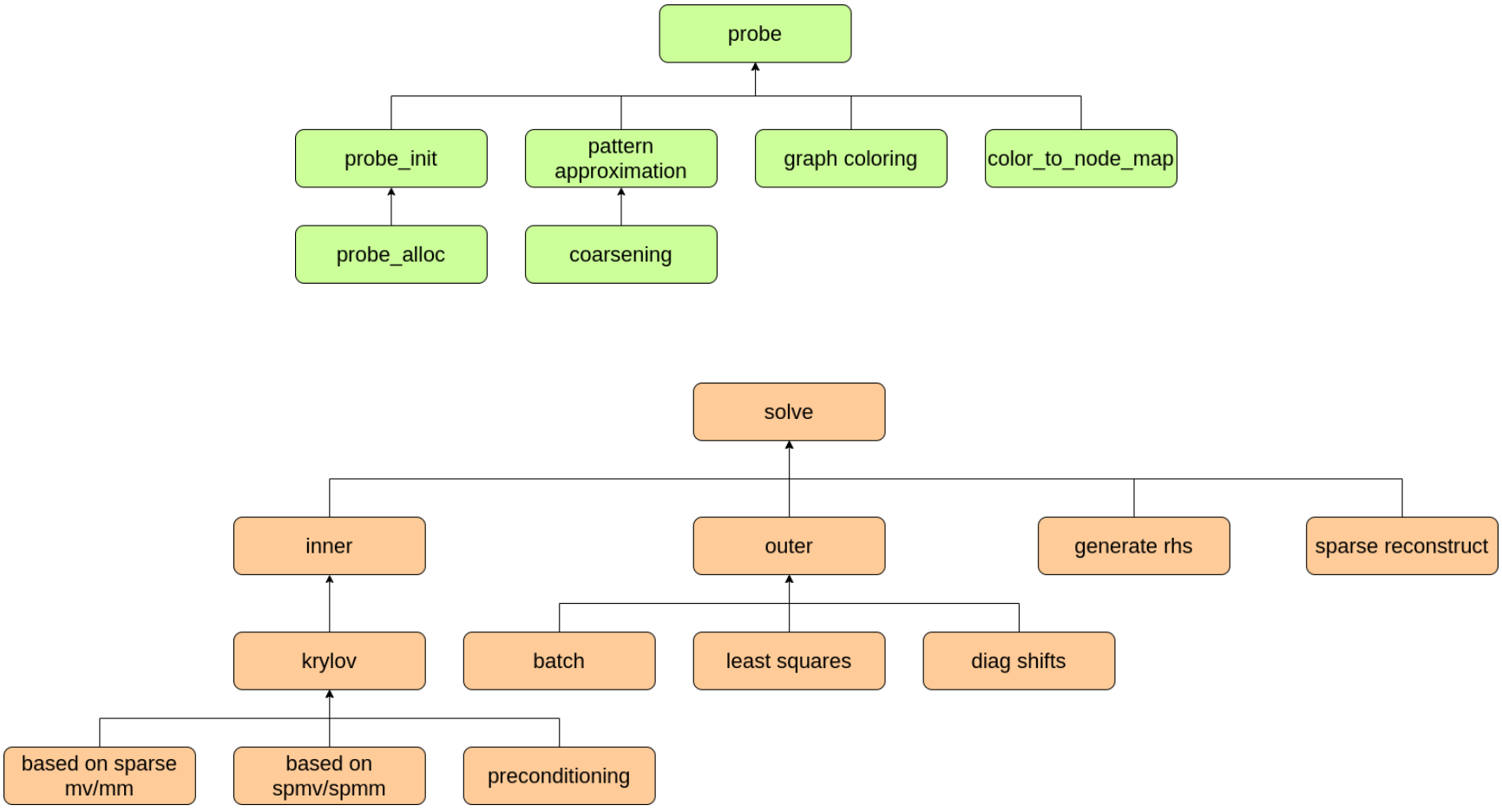
approximate inverse structure + solving system with MRHS

IMPLEMENTATIONS ON HIGHLY PARALLEL HETEROGENEOUS ARCITECTURES

- Written in C, uses MKL & CUDA libraries
- Block probing, Pattern sampling, low-memory variants
- SRHS, block, global variants of CG, QMR, GMRES for symmetric, complex-symmetric, hermitian matrices
- Approximate inverse preconditioning
- Least squares fitting methods

Numerical experiments (in this talk): Emphasis on demonstrating advantages of the block probing method. Not focused on solvers & scalability.


input
context_init
probe_alloc
probe_funtion
probe
external coloring
YES
NO
graph_coloring
mp_color_to_node_allocate
mp_color_to_node_set
mp_probe_free
if multiple sampes required
solve_alloc
solve (AW = V)
solve
YES
add new sample
NO
solve_free
refinement required
refine
YES
NO
refine (AX = I)
output_matrix_write
output
mp_meta_write
context_free

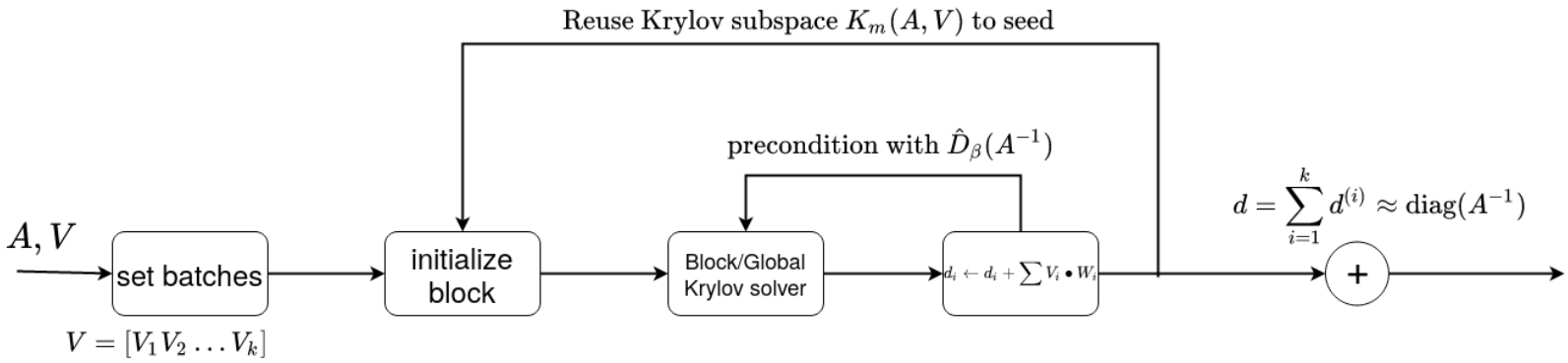

probe
probe_init
pattern approximation
graph coloring
color_to_node_map
probe_alloc
coarsening
solve
inner
outer
generate rhs
sparse reconstruct
krylov
batch
least squares
diag shifts
based on sparse mv/mm
based on spmv/spmm
preconditioning

algorithm 1: approximate sparse inverse M

1. c $\leftarrow$ probing(A, β, k, p)
2. M $\leftarrow$ solver_outer(A, c)
3. if required refine(A, M)

solver: 2-level organization [outer, inner]

- outer: generate batch $V^{(i)}$, reconstructs M, update vectors for deflation
- inner: solves $AW^{(i)} = V^{(i)}$

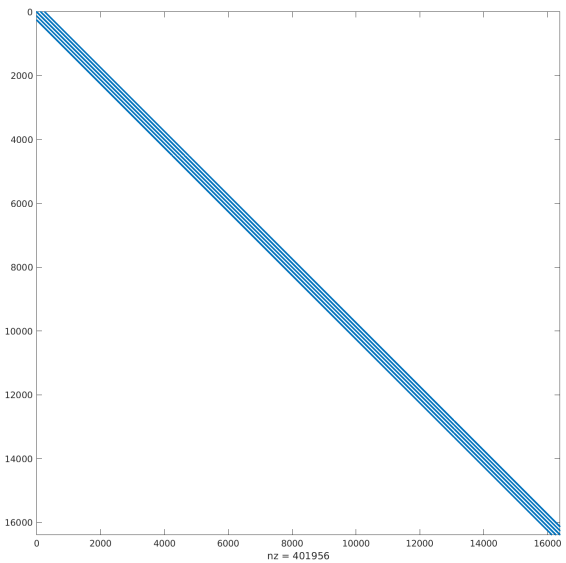


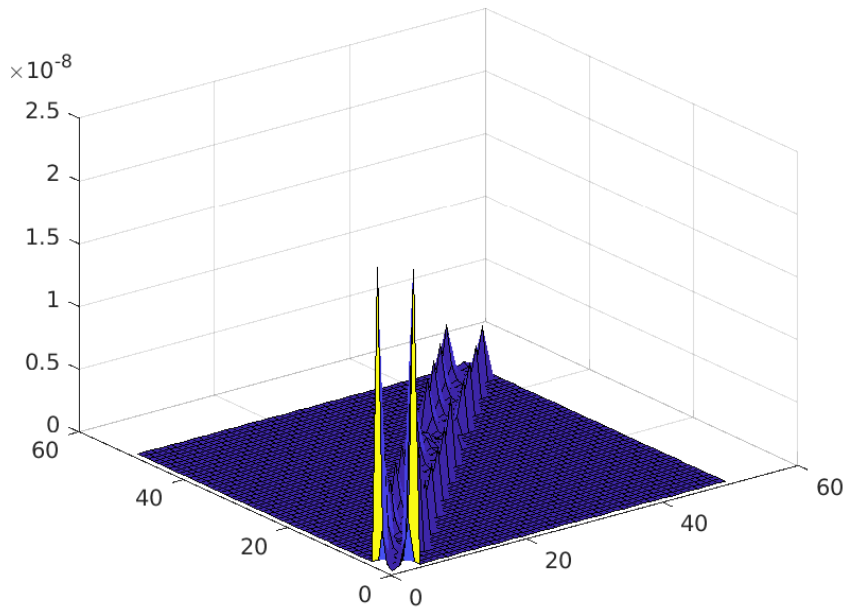
Experiments with covariance matrices

Dataset 1: Uncertainty quantification, on 2D grid

$$(A)_{ij} = \begin{cases} \left(1 - \frac{\text{dist(i, j)}}{\alpha}\right)^{\beta}, & \text{if dist}(i,j) \leq \alpha \\ 0, & \text{otherwise} \end{cases}$$

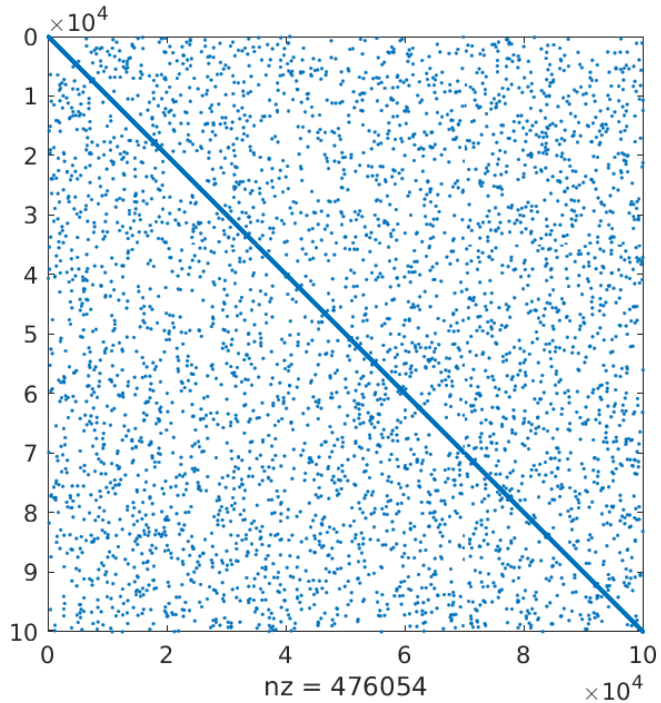
Use 35 iterations of PCG (max) with jacobi preconditioning, (2, 3)-probing, $\mathcal{E} = 3.7 \times 10^{-7}$, computation of 4×4 blocks





Covariance matrix from gaussian random fields

50000×50000 sampled covariance matrix generated from 125 sampled vectors. (Many thanks to Aryan Eftekhari and Olaf Schenk for sharing the code that generates the matrices).



n	s	nrhs	blk	nlevels	E
5e4	125	992	4	2	4.79e-05
5e4	125	2124	6	2	3.48e-05

Experiments with standard laplacian operators

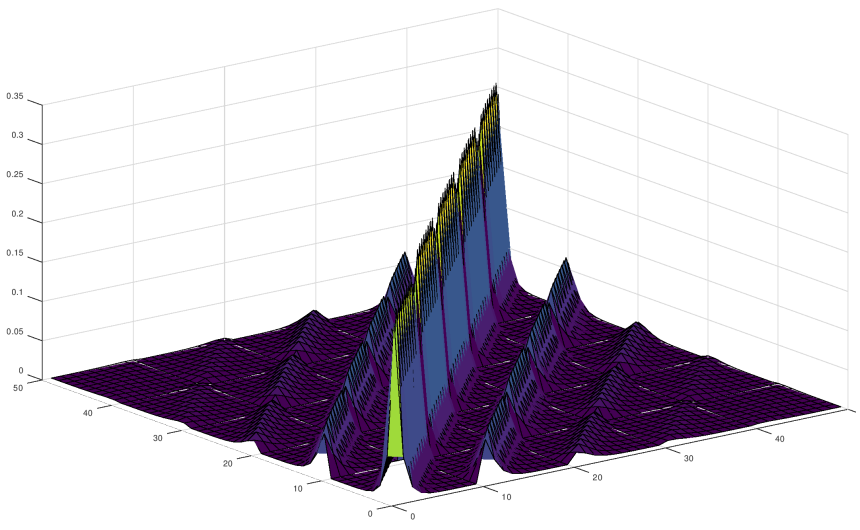
Dataset 2: Laplacian matrices with diagonal shifts:

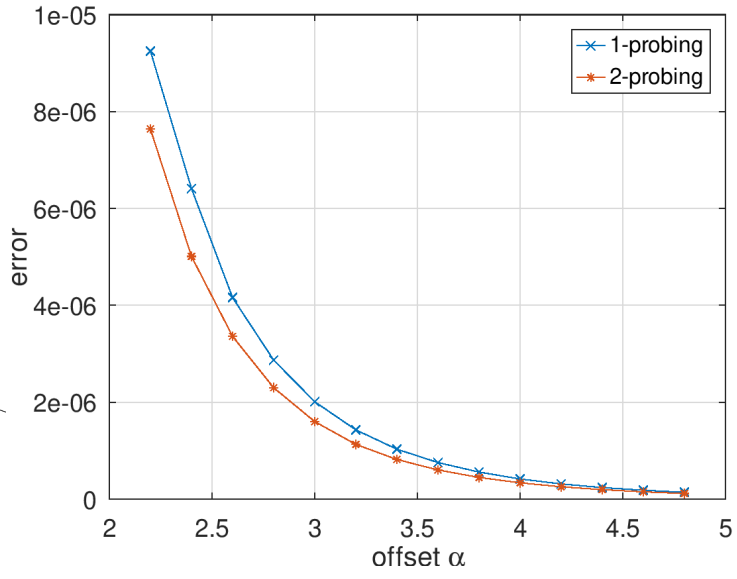
$$A = D - H + \alpha I$$

$H \in \{0,1\}^{n \times n}$: depends on the domain (2D, 3D)

D: diagonal matrix, depends on boundary conditions

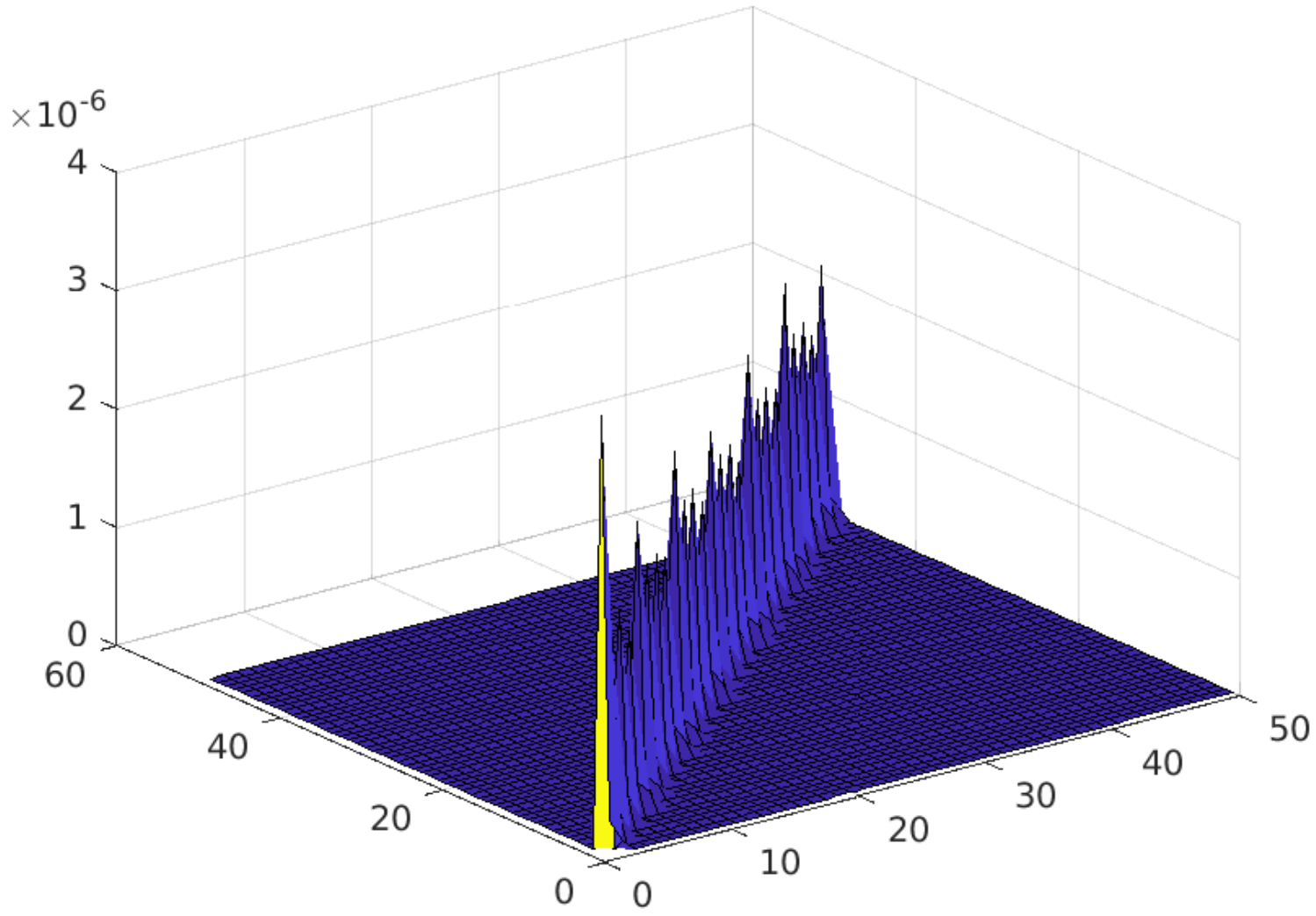
α: shift that controls diagonal dominance





Numerical Experiments for the diagonal blocks

(2, 2)-block probing, 30 iterations of PCG, jacobi preconditioning
$\alpha = 3.2, \mathcal{E} = 8.2 \times 10^{-6}$



Numerical Experiments for off-diagonal entries

(2, 3)-block probing, 30 iterations of PCG, jacobi preconditioning
$\alpha = 3.6$, mask: $S(A)$, $\mathcal{E} = 3.8 \times 10^{-3}$

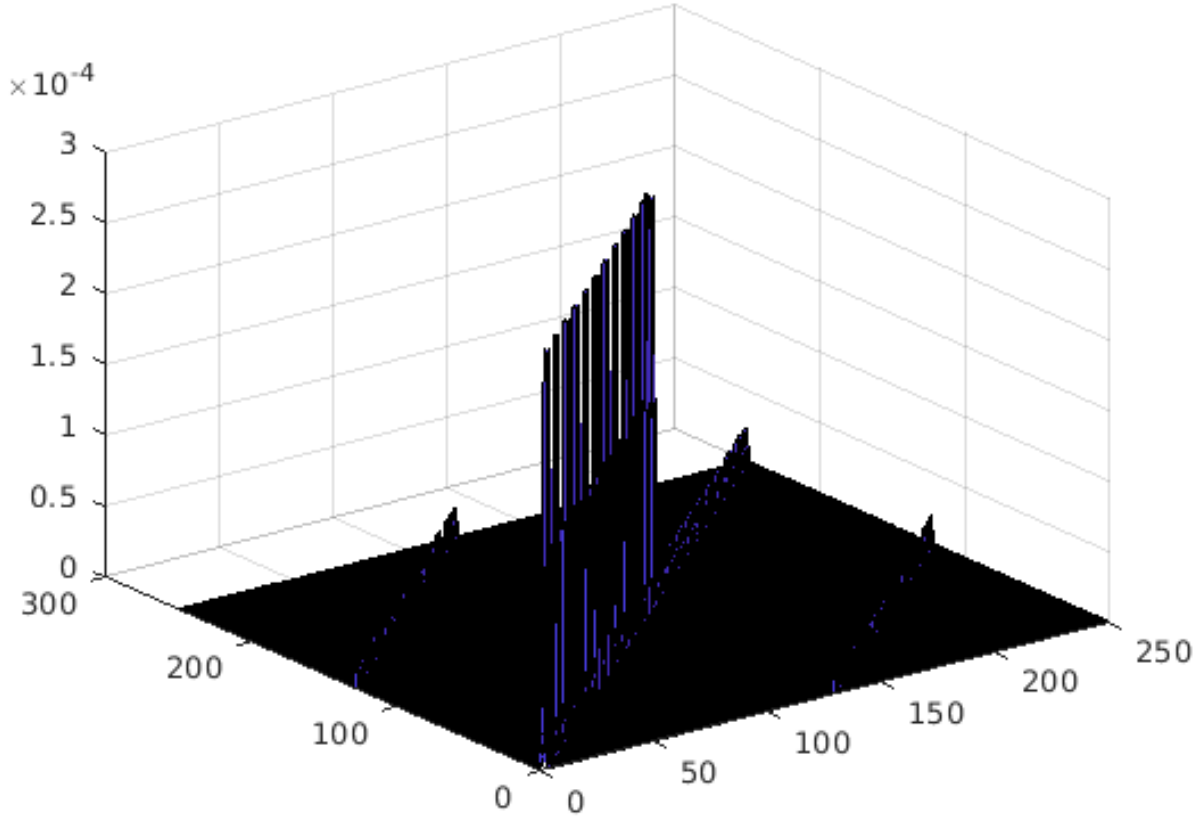


Figure: Entrywise relative error, of entries on $S(A)$ (top 250×250)

(2, 2)-block probing, 30 iterations of PCG, jacobi preconditioning
$A = H + D +$ 6*diag(rand(n, 1)), mask: $S(A)$ $\mathcal{E} = 1.6 \times 10^{-6}$

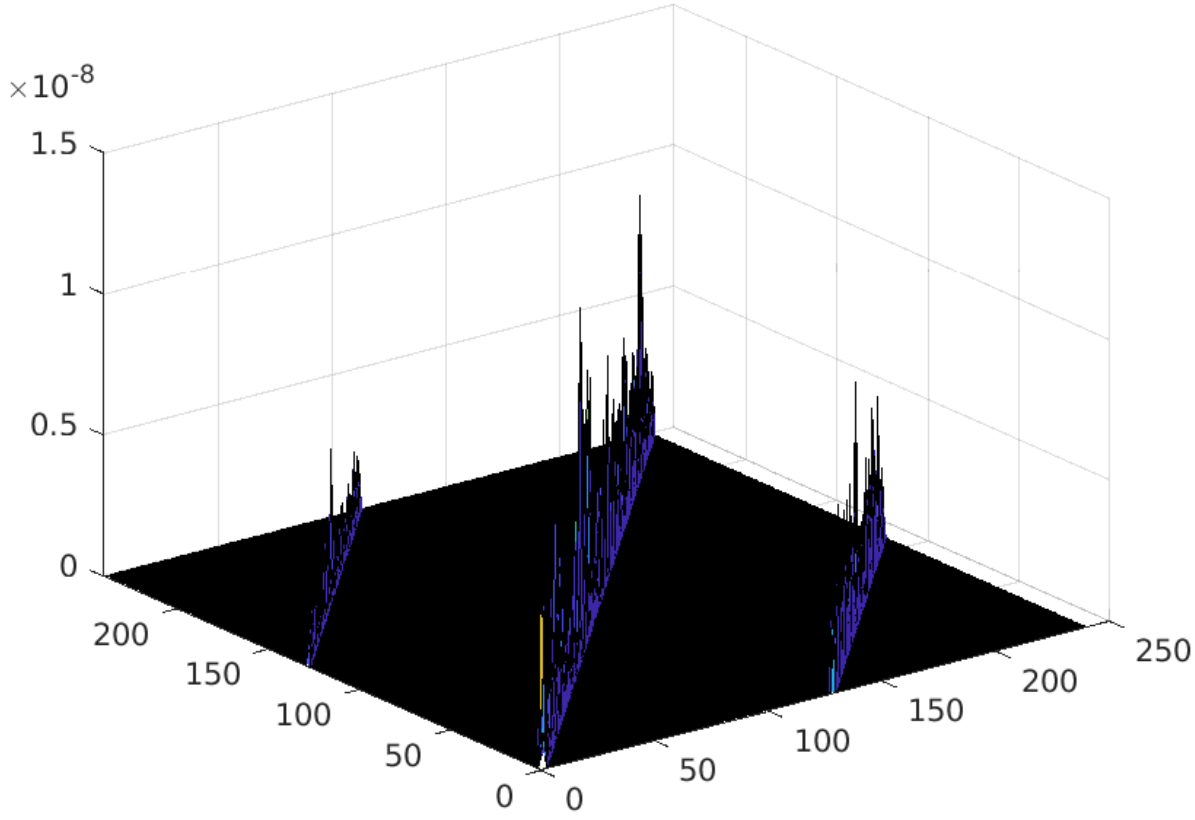
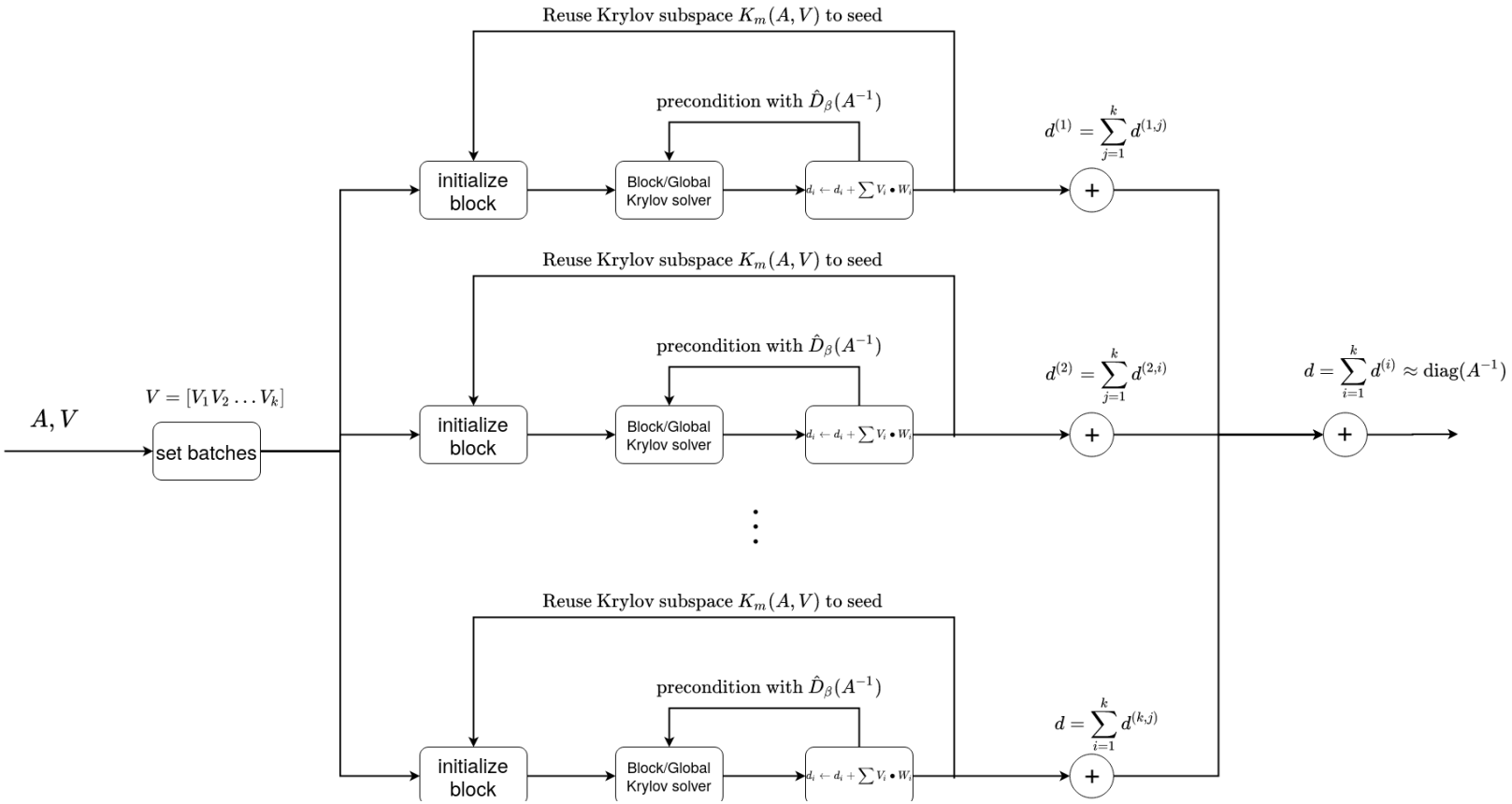


Figure: Entrywise relative error, of entries on $S(A)$ (top 250×250)

Shared memory parallelism

1st level: parallelism in blas and sparse blas (probing + solver)
2nd level: 1 thread per batch $V^{(i)}$ (solver only)



65536×65536 covariance matrix, 15 CG iterations
block probing: $\beta = 2$, $p = 4$

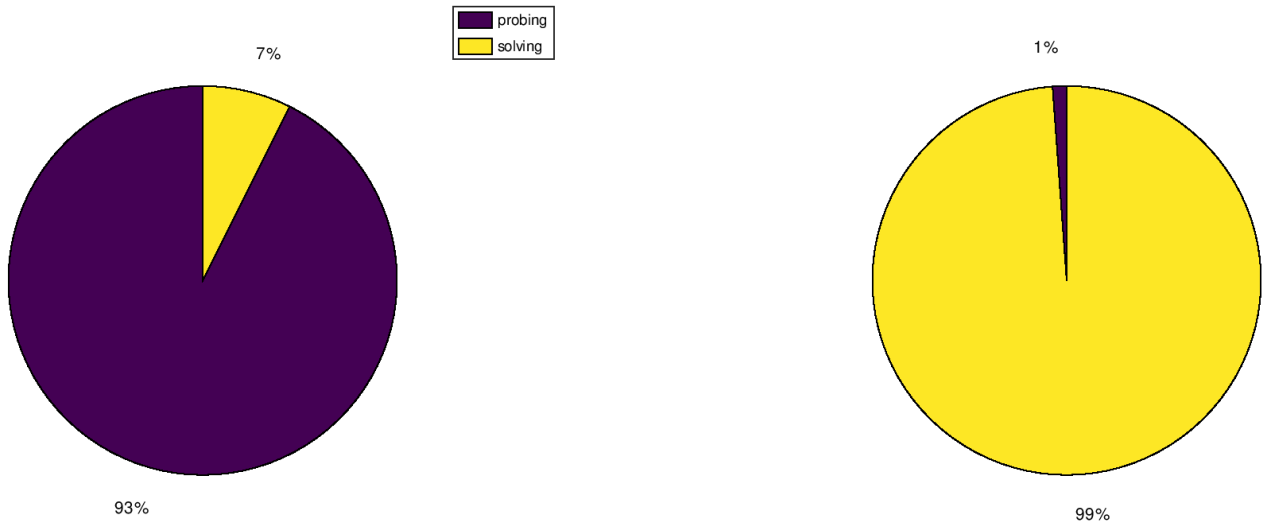
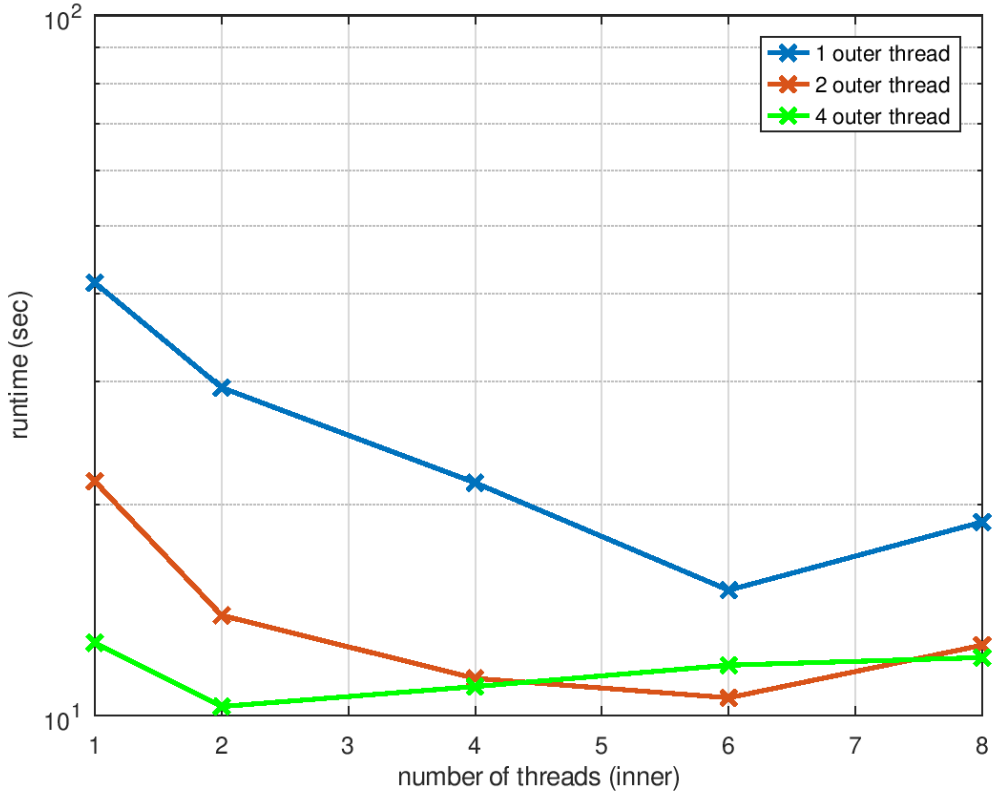


Figure: On the left: $(1, 16)$-probing. On the right: $(2, 4)$-probing

Total runtime 406 $\rightarrow$ 44.1 sec, dominated by solving $AW^{(i)} = V^{(i)}$

inner-threading: BLAS parallelism,
outer-threading: in outer_solver only



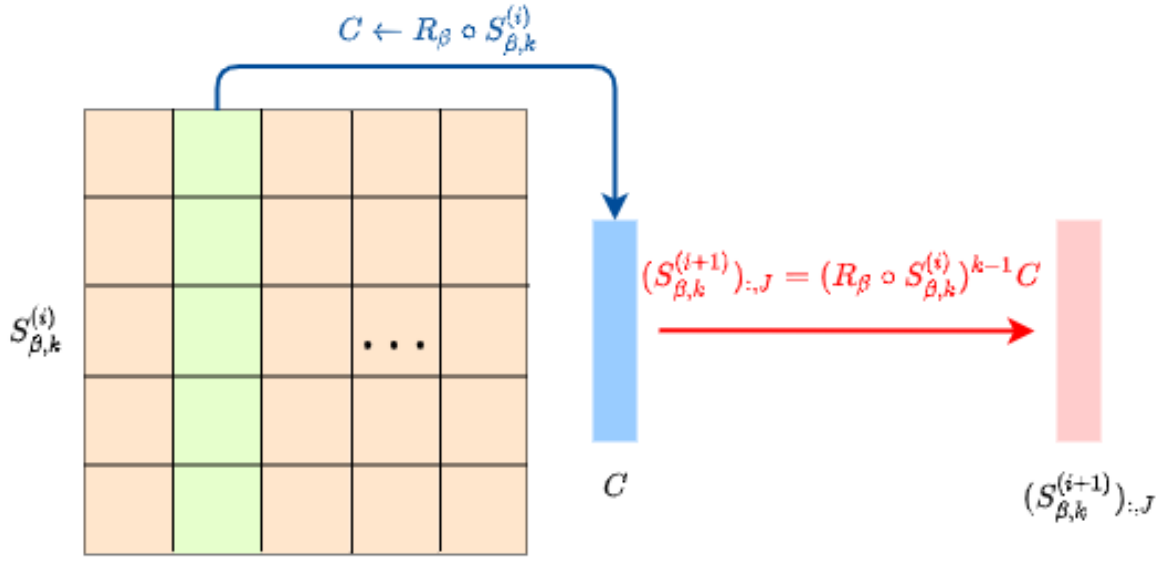
#inner-threads $\times$ #outer-threads active at outer solver

Batch block probing

Practical problems:

1. For very "dense" matrices $S^{(p)}_{\beta,k}$ becomes dense
2. For very sparse matrices do not scale

Solution: Update columns of $S^{(i)}_{\beta,k}$ in batches for $i = 1 \ldots, p$



$S^{i+1}_{\beta,k\ :,J} \leftarrow \left[\wedge_{i=1}^{k-1}(\underline{R_\beta \circ S^{(i)}_{\beta,k}})\right] \wedge C$ requires knowledge of $S^{(i)}_{\beta,k}$

$\Rightarrow$ difficult to compute + parallelize (dependencies on non-active blocks of columns)

Solution 1: Use only 2 levels since $S^{(0)}_{\beta,k} = R_\beta \circ S(A)$ with "large" enough β (assuming A can be loaded in memory).

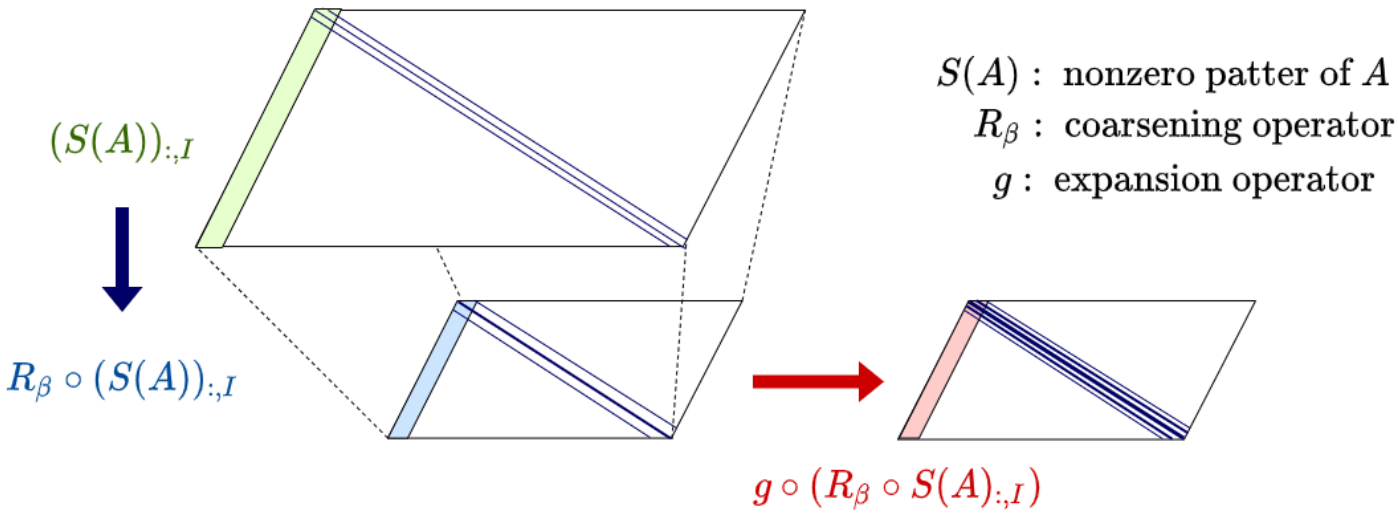


Figure: Block probing in batches

Pros; simple implementation Cons: single level $\Rightarrow \beta \uparrow \Rightarrow RHS \uparrow$ (compared to non-batch version)

Solution 2: Construct the coarsened hierarchy $\{A^{(i)}\}_9$ of A, and $A^{(i)}$ for the expansion operations.

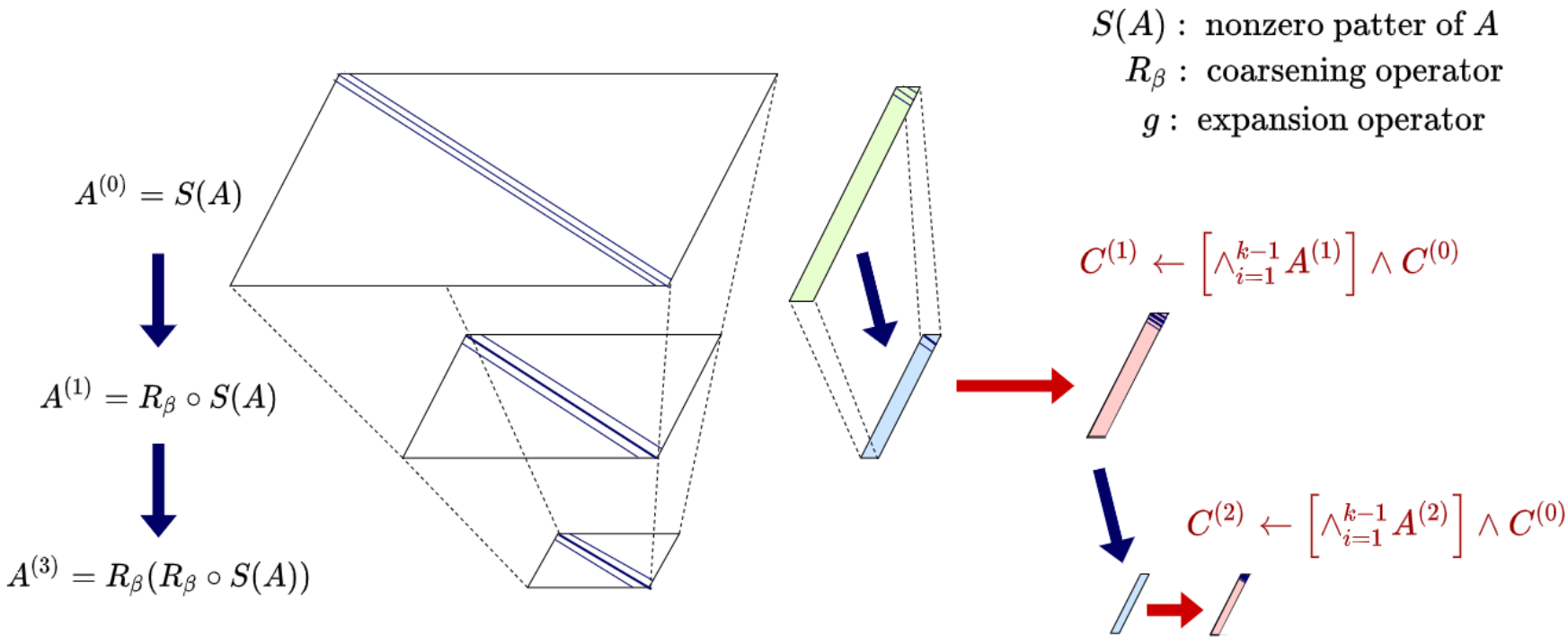


Figure: Block probing in batches & using coarsened patterns of A

Pros; can capture $S(T_\delta(A^{-1}))$ on a finer scale.
Cons: requires careful selection of k to avoid the same problems as "solution 1". Currently working on automated selection of k.

Pattern Sampling

For block probing $(P_{\beta^k} \circ S^p_{\beta,k}) \odot S(A^{kp}) = S(A^{kp}) \Rightarrow \mathcal{E}_\beta \leq \mathcal{E}_1$ for $\beta > 1$.

What if $(P_{\beta^k} \circ S^p_{\beta,k}) \odot S(A^{kp}) = S(A^{kp})$ holds only on average?

$\Rightarrow$ Use sampling as the coarsening operator.

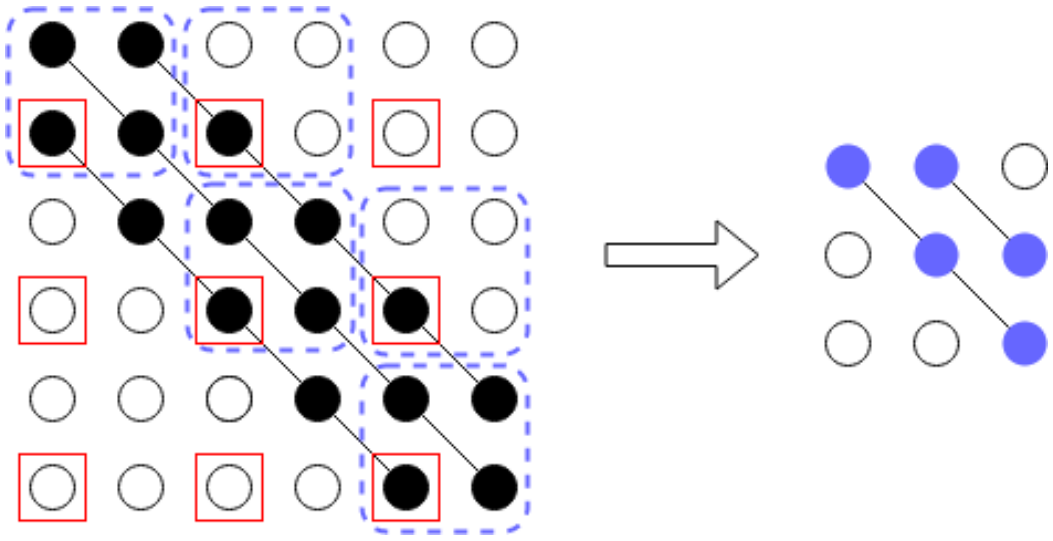


Figure: Red colored squares: sampled entries at even rows and odd columns. Blue colored doted squares: blocks captured by blocking.

Various ways to sample patterns; eg. edge, vertex, pattern sampling methods.

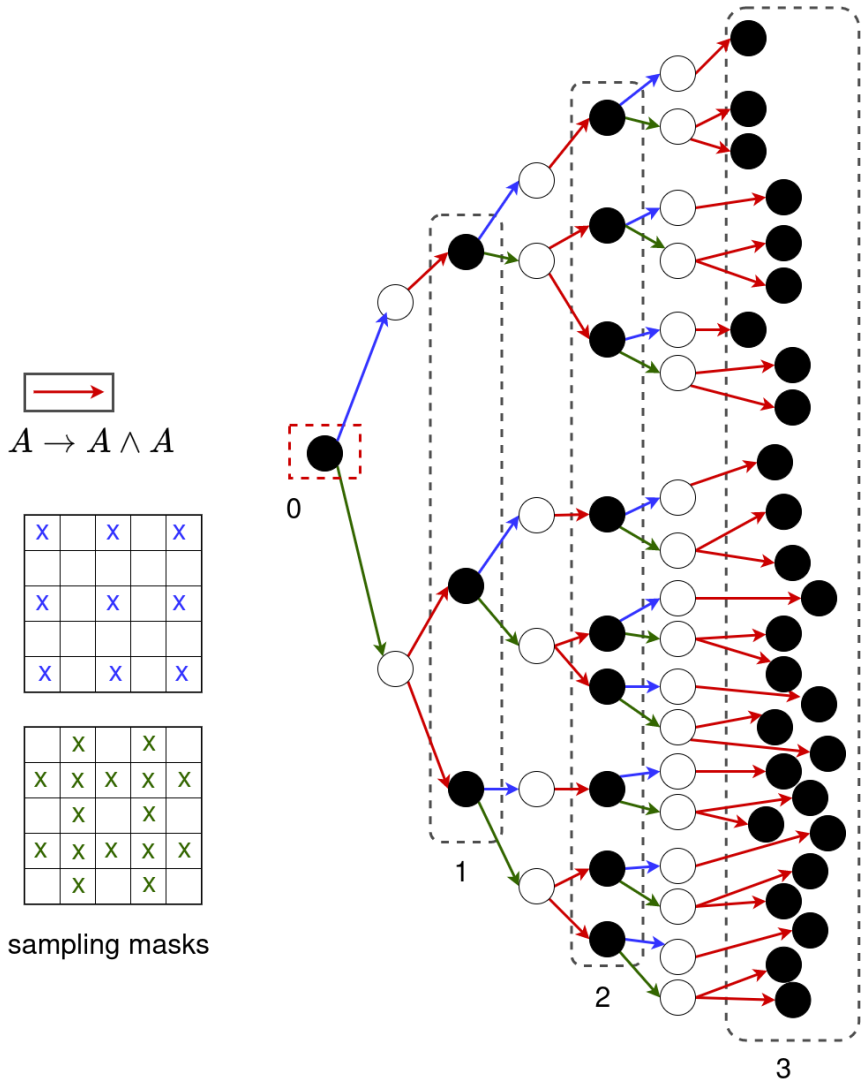


Figure: Tree representations of coarsening and expansion operations for constructing approximate patterns.

Comparing with blocking and "classical" probing

Dataset 1: Uncertainty quantification, on 2D grid

$$(A)_{ij} = \begin{cases} \left(1 - \frac{\text{dist(i, j)}}{\alpha}\right)^{\beta}, & \text{if } \text{dist}(i,j) \leq \alpha \\ 0, & \text{otherwise} \end{cases}$$

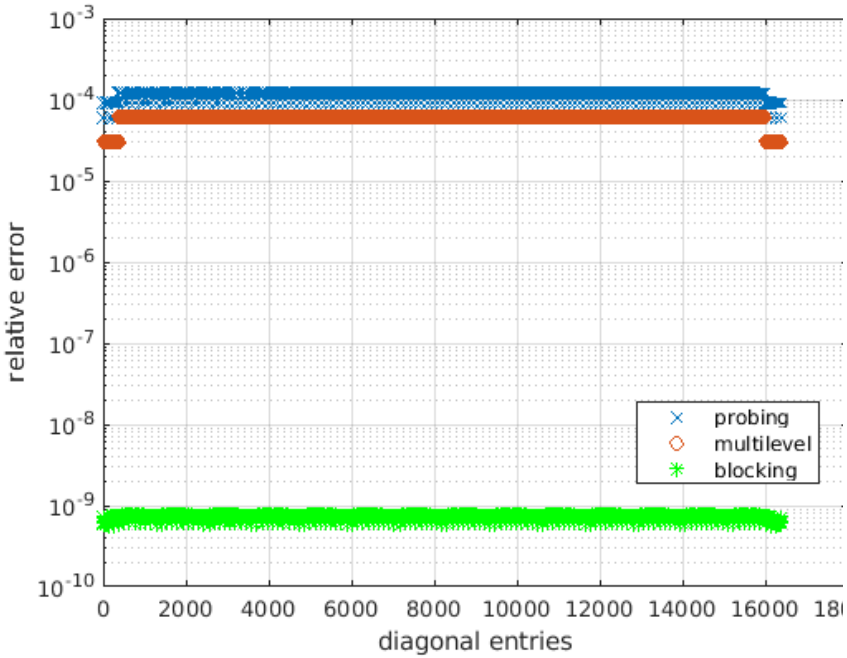


Figure: $\alpha = 2$, $\beta = 5$

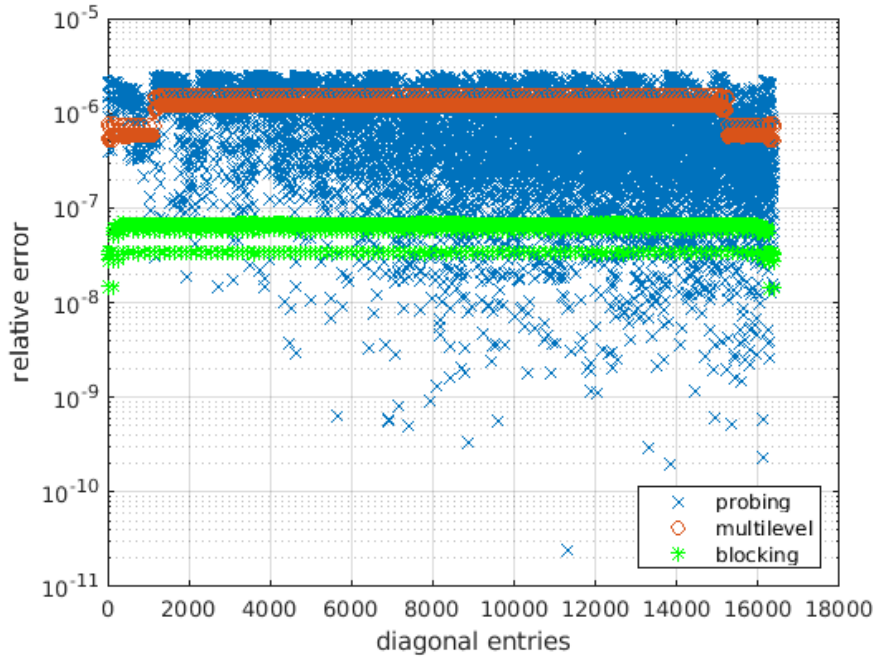


Figure: $\alpha = 5$, $\beta = 5$

Least squares solver

Use multilevel probing as initializer to low rank or quadrature methods

$$A^{-1} = I + \alpha_0 A + \alpha_1 A^2 + \cdots + \alpha_{k-1} A^k + \cdots + \alpha_{n-2} A^{n-1}$$

approximate by k-first terms

$$A^{-1} \approx I + \alpha_0 A + \alpha_1 A^2 + \cdots + \alpha_{k-1} A^k$$
$$d(A^{-1}) \approx \mathbf{e} + \alpha_0 d(A) + \alpha_1 d(A^2) + \cdots + \alpha_{k-1} d(A^{k-1})$$

rewrite as a rectangular system of equations $Ca = z$ where:

$C = [\mathbf{e}, d(A), \ldots, d(A^{k-1})]$, $a = [\alpha_0, \alpha_1, \ldots, \alpha_{k-1}]^\top$, $z = d(A^{-1})$

both a and z are unkowns $\rightarrow$ cannot solve system $\Rightarrow$ replace with

$$\mathbf{e} \approx d(A) + \alpha_0 d(A^2) + \cdots + \alpha_{k-2} d(A^{k-1}) + \alpha_{k-1} d(A^k)$$

New system of equations $\hat{C}a = \mathbf{e}$ for
$\hat{C} = [d(A), d(A^2), \ldots, d(A^k)]$, $a = [\alpha_0, \alpha_1, \ldots, \alpha_{k-1}]^\top$

1) Solve $\hat{C}a = \mathbf{e}$
2) reconstruct approximation $d(A^{-1}) \approx Ca$

Algorithm 2 batch_ls_solve: approximates $\mathrm{d}_\zeta(A^{-1})$

Input: $A \in \mathbb{C}^{n \times n}$, β, k, p, color_array, $\zeta \leq \beta$
Output: M (approximate sparse inverse)

for $i = 1$ to s **do**
  $V^{(i)} \leftarrow$ generate_rhs(ζ, β, k, color_array)
  $W^{(i)} \leftarrow AV^{(i)}$,
  $\hat{C}(:, I+1) \leftarrow$ reconstruct($W^{(i)}, \zeta$)
  swap($V^{(i)}$, $W^{(i)}$)
  for $j = 2$ to p **do** // applies Horner's method
    $W^{(i)} \leftarrow (I + A)V^{(i)}$
    $\hat{C}(:, J+1) \leftarrow$ reconstruct($W^{(i)}, \zeta$)
    swap($V^{(i)}$, $W^{(i)}$)
  find y that minimizes $\|\hat{C}a - e\|_2$, $d_\zeta \leftarrow$ reconstruct(C, a)

Sparsified Krylov

By applying the lanczos procedure on $AV = W$

$$AU = UH$$

So for $j = 0, 1, \ldots$

$$Au_j = \beta_{j+1} u_{j+1} + \alpha_j u_j + \beta u_{j-1}$$
$$\beta_{j+1} u_{j+1} = Au_j - \alpha_j u_j - \beta u_{j-1}$$

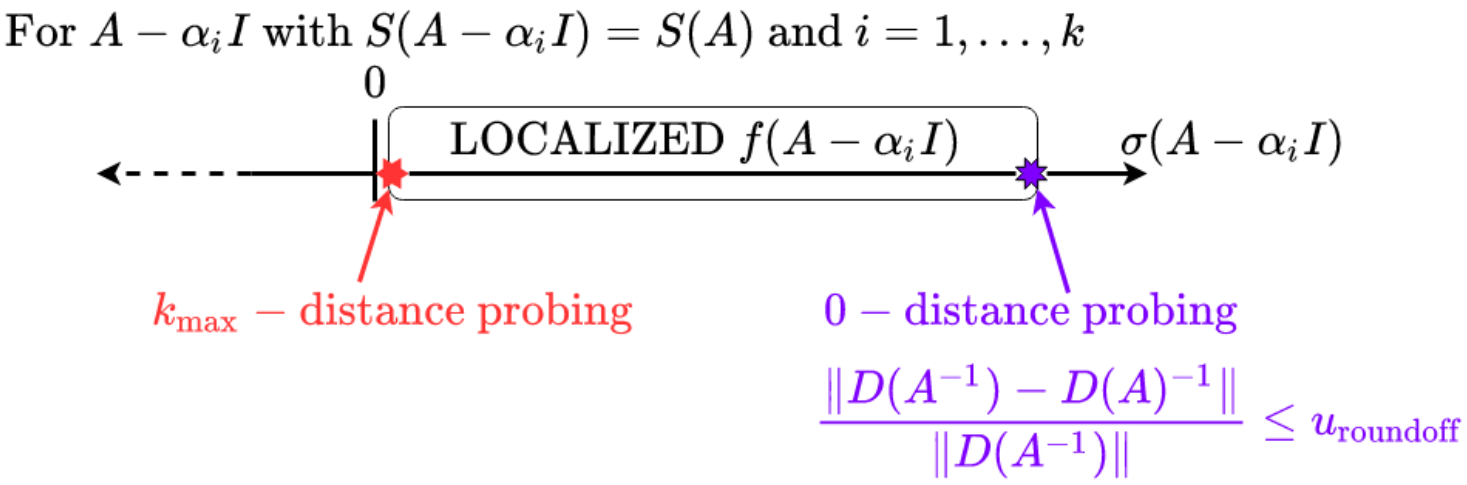
After computing u_{j+1} the nonzeros of $u_{j-1} \odot b^{(i)}$ can be stored.

$\longrightarrow$ No relevant information is lost.

For system of equations $AX = B$ with s rhs $\tilde{U} = \hat{U} \odot [e^\top \otimes B]$ has $O(nm)$ nonzero entries instead of $O(nms)$.

- basis U for krylov subspace $K_{m-1}(A, V)$ of size $n \times ms$
- the sparsified basis has exaclty $O(nm)$ nonzeros

Sequence of Linear Systems with αI Perturbations



As long as perturbations $\alpha_i I$ do not change the order of $|\lambda_j|$

$$U\Lambda(A - \alpha_i I)U^\top = U(\Lambda(A) - \alpha_i I)U^\top$$
$$\text{and } Uf(\Lambda(A - \alpha_i I))U^\top = Uf(\Lambda(A) - \alpha_i I)U^\top$$

$\Rightarrow$

Krylov subspace $K_m(A, V) = \{V, AV, \ldots, A^{m-1}V\} = \text{span}(U_m)$ can be reused

Implementation in the [outer, inner] solver framework

Outer: batch_seq_solve

Inner: global_lanczos_sparsified

Algorithm 3 batch_seq_solve: Computes $((A - \alpha_z I)^{-1})_{ij}$ for $\{\alpha_z\}_z$

Input: $A \in \mathbb{C}^{n \times n}$, β, k, m, color_array

Output: M (approximate sparse inverse)

1: **for** $i = 1$ to s **do**

2: $\quad V^{(i)} \leftarrow$ generate_rhs(β, k, color_array)

3: $\quad [U_m, H_m] =$ global_lanczos_sparsified(A, $V^{(i)}$, m)

4: $\quad$ solve $H_m y = \|U_m\|_F e_1$

5: $\quad M_z \leftarrow$ reconstruct(U_m, y_m, color_array, β, k)

6: $\quad$ **for** $z = 2$ to $|\{a_z\}|$ **do**

7: $\quad\quad H_m \leftarrow H_m + \alpha_z I$

8: $\quad\quad$ solve $H_m y = \|U_m\|_F e_1$

9: $\quad\quad M_z \leftarrow$ reconstruct(U_m, y_m, color_array, β, k)

Local Krylov Iterations

Observation: Probing as a preprocessor reduces

$$AX = I_n \longrightarrow AX = V, \;\; V \in \{0,1\}^{n \times s}$$

Improvements when $v \in \hat{K}_p(A, b) = \{v_0, v_1, \ldots, v_{p-1}\}$ stores only nonzeros.

$$\text{nnz}(v_k) \leq \deg_{\max}^{k-1}(A) \cdot \text{nnz}(v_0)$$

For solving $AX = I_n$ this becomes

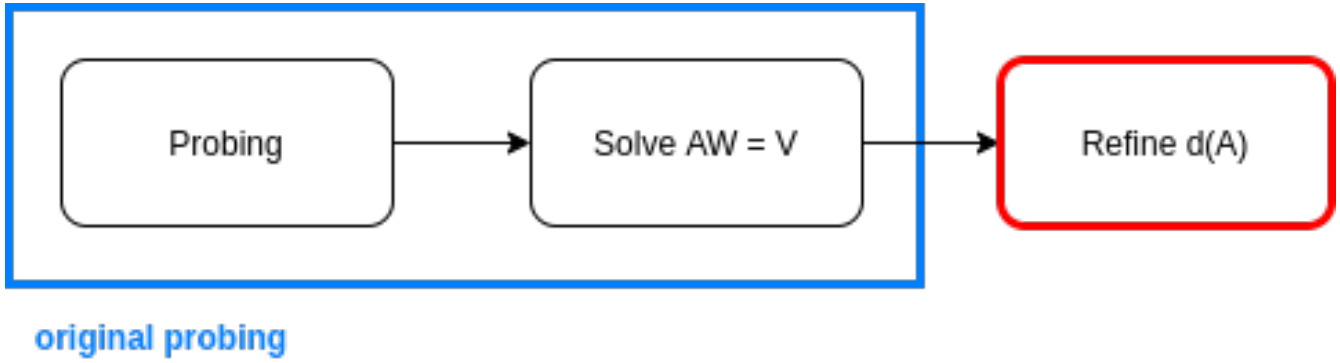
$$\text{nnz}(v_k) \leq \deg_{\max}^{k-1}(A)$$

So $AX = I_n$ is solved in $O(n \cdot \deg_{\max}^{iterations})$ flops

Improvement: can utilize incomplete products $\rightarrow$ localized methods
Difficult to analyze

Iterative Refinement

Augment original scheme



We would like for the refinement step to not be bounded by $\mathcal{E}_{\text{probe}}$

How? $\Rightarrow$ solve $AX = I$ (probing tries to avoid this...)

Use local krylov iterations, to solve $AX = I$ using as $X_0 = M$ as the approximation from known values, also M forms a preconditioner

$$MAX = M \text{ with } X_0 = M$$

Observations:

1. Not exactly refinement in the sense of solving linear systems.

$$\textbf{Problem 1: } W = \text{argmin}\|AW - V\|$$
$$\textbf{Problem 2: } X = \text{argmin}\|AX - I\|$$

   Problem 1 is not equivalent to Problem 2

2. Due to observation 1, Newton method should be more suitable. We use Krylov solvers due to availability.

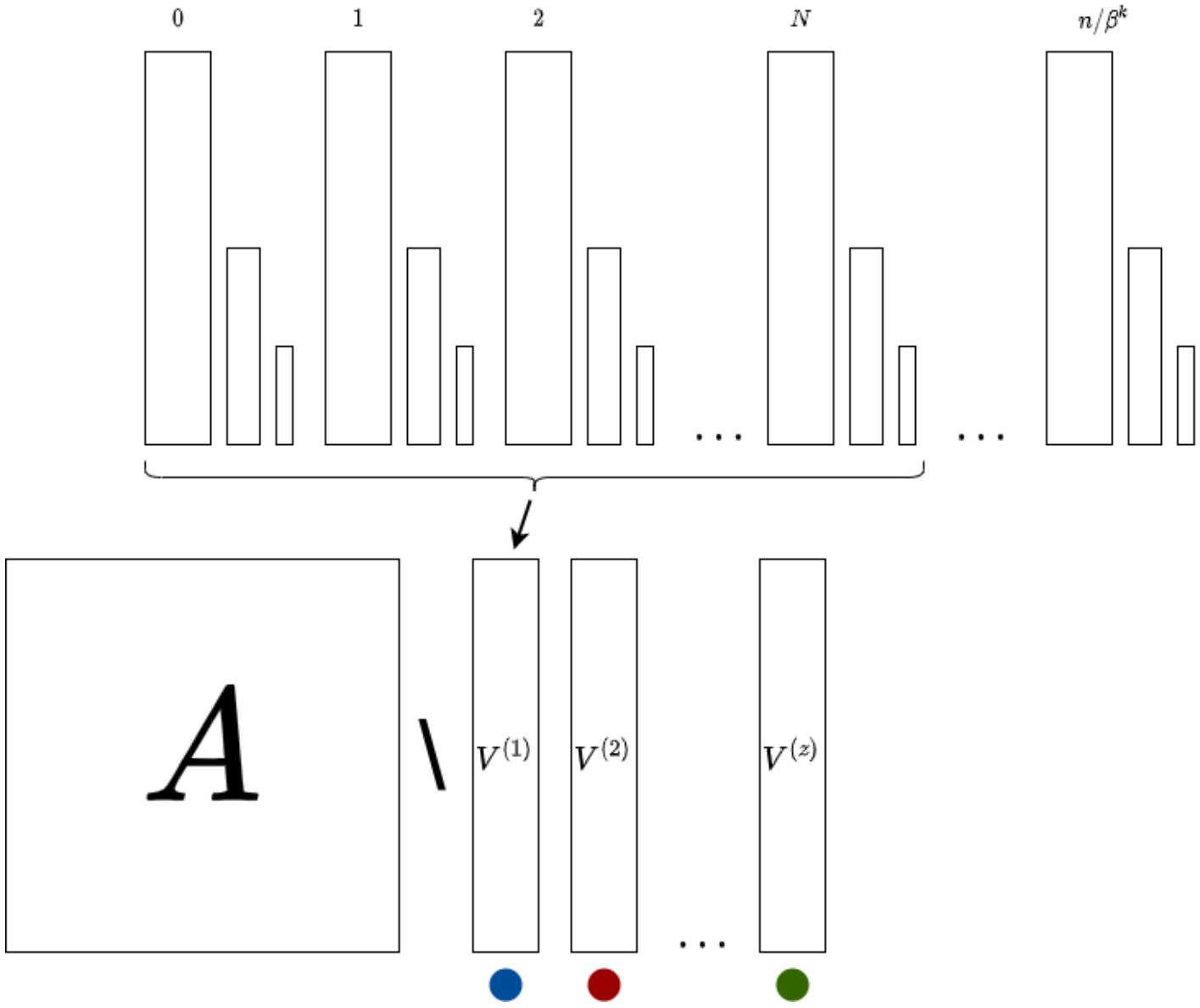
3. Preconditioning with M does not reduce significantly the condition number **but** running lanczos decomposition on $M^{1/2}AM^{1/2}$ generates localized vectors $\{u_i\}_i$ (they can become sparse by thresholding)

4. Easy to parallelize, evaluate runtime, compare residual with probing and choose whether to use for all RHS of I

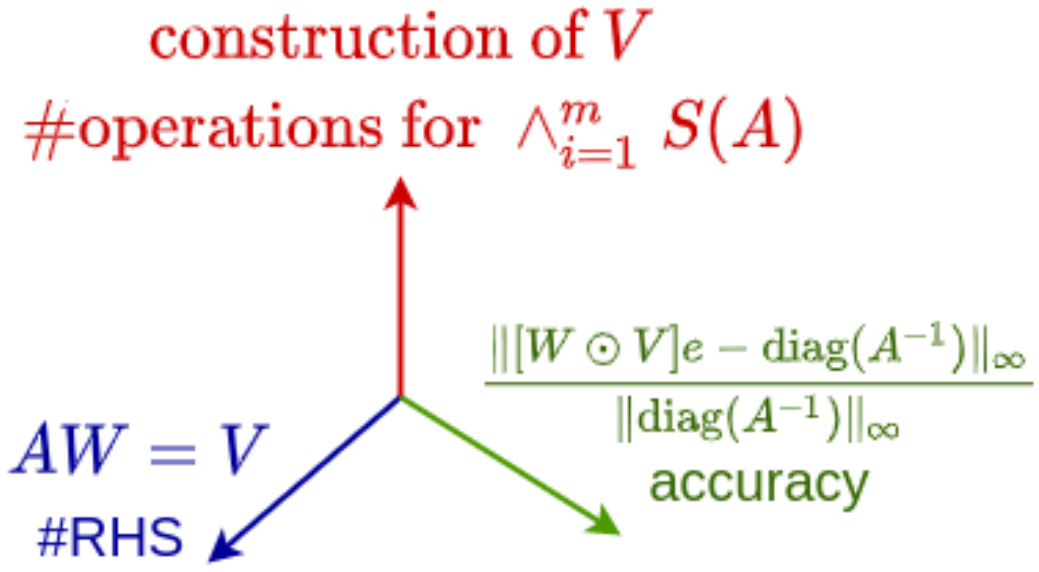
For matrix $A + \alpha I$, $\alpha = 6$.

method	grid size	s	blk	nlevels	error
ls_horner	128×128	248	2	3	5.4e-5
global_ls_horner	128×128	248	2	3	4.3e-5
global_lanczos_spars	128×128	248	2	3	1.3e-6
global_lanczos_ref_20	128×128	248	2	3	8.4e-8

In development: Combining batch probing and local(/ized) iterations $\longrightarrow$ asynchronous method for GPUs.



Conclusions: Focus on utilizing possible tradeoffs of probing



Did not cover:

1. Hybrid probing - uses exclusive and inclusive prolongations to control #RHS
2. GPU implementations
3. Detailed comparison of solvers
4. Use of preconditioning & deflation
5. Eigenvalue, condition number estimation (similar to "traditional" probing methods)

Future work

- Mixed precision support (+half precision)
- Combination with SQUIC method for covariance estimation
- Combination with multigrid solvers (?)
- Exploit approximate low-rank of the off-diagonal blocks
- Use to fill-in blocks of approximate inverse preconditioners

Related work:

1. (Tang 2011) $\rightarrow$ PDES, covariance estimation
2. (Andreas Stathopoulos, 2012-2021) $\rightarrow$ trace estimation, LQCD applications
3. (Eftekhari 2021) $\rightarrow$ SQUIC, covariance estimation
4. (Andreas Frommer 2021) $\rightarrow$ theoretical analysis of probing

Thank you

Questions?