Algorithm for Sparse Approximate Inverse Preconditioners Refinement in Conjugate Gradient Method

Labutin Ilya
Institute of Petroleum Geology and Geophysics SB RAS
Surodina Irina
Institute of Computational Mathematics and Mathematical Geophysics

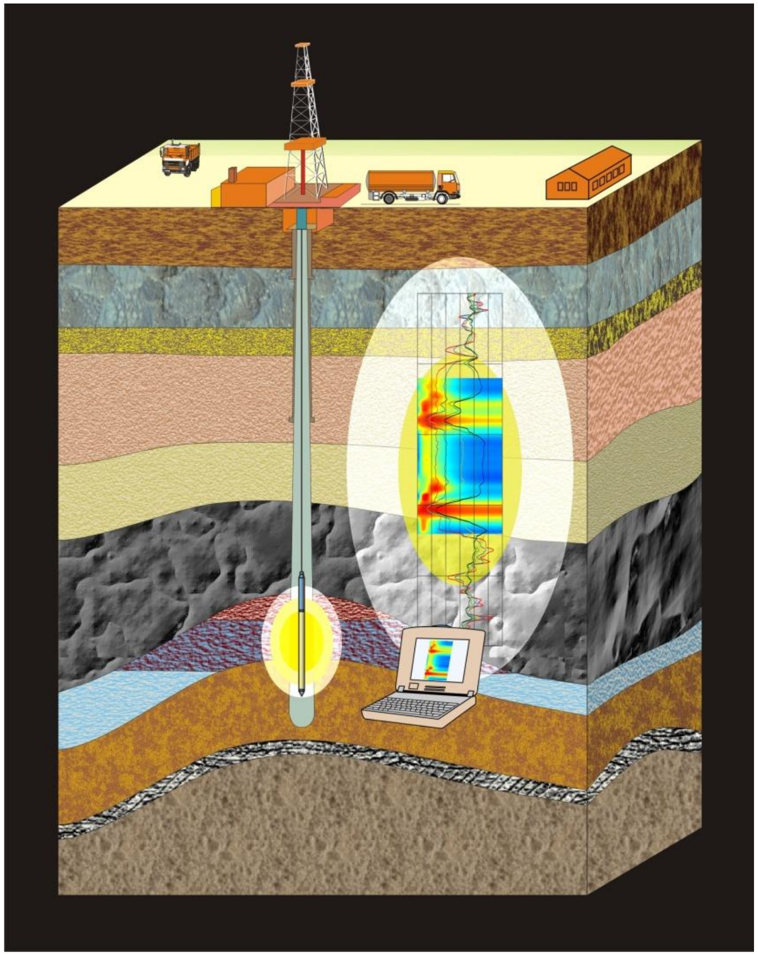
15'th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Verified Numerical Computations

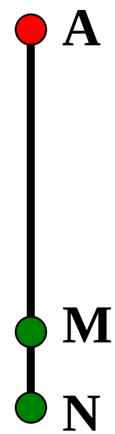
Novosibirsk, 2012

Agenda

- Motivation
- Parallel preconditioners
- Numerical experiments

Well resistivity logging





GZ1 – A0,4M0,1N
GZ2 – A1M0,1N
GZ3 – A2M0,5N
GZK – N0,5M2A
GZ4 – A4M0,5N

Potential field problem

$$\frac{1}{r}\frac{\partial}{\partial r}(\sigma \; r\,\frac{\partial U^{a}}{\partial r})+\frac{\partial}{\partial z}(\sigma \; \frac{\partial U^{a}}{\partial z})= \frac{1}{r}\frac{\partial}{\partial r}((\sigma_0-\sigma \;)\,r\,\frac{\partial U^{0}}{\partial r})+\frac{\partial}{\partial z}((\sigma_0-\sigma \;)\frac{\partial U^{0}}{\partial z})$$

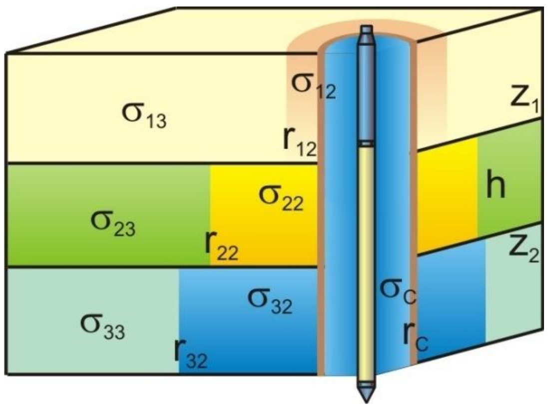
$$\hbar_i^{(r)}=\left(h_i^{(r)}+h_{i+1}^{(r)}\right)/2 \qquad h_i^{(r)}=r_i-r_{i-1} \qquad i=1,\ldots,N_r \qquad \left(V\right)_{\bar{r},ij}=\left(V_{ij}-V_{i-1,j}\right)/\,h_i^{(r)}$$

$$a_{ij}=\sigma(r_i-\frac{h_i^{(r)}}{2},z_j+\frac{h_j^{z}}{2}) \qquad b_{ij}=\sigma(r_i+\frac{h_i^{(r)}}{2},z_j-\frac{h_j^{(z)}}{2})$$

$$Ax=F$$

$$AV=-\frac{1}{r}\left(\bar{r}\,aV_{\bar{r}}\right)_{\hat{r}}-\left(bV_{\bar{z}}\right)_{\hat{z}}$$

$$F=\frac{1}{r}\left(\bar{r}\,(a-\sigma_0)U_{\bar{r}}^{0}\right)_{\hat{r}}+\left((b-\sigma_0)U_{\bar{z}}^{0}\right)_{\hat{z}}-\frac{(a-\sigma_0)}{r^2}U^{0}$$



r_{jl} - Radial bounds

z_j - Vertical bounds

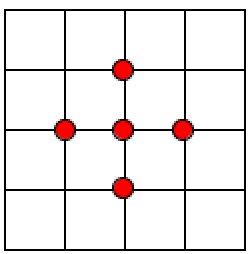
σ_{jl} - Conductivity

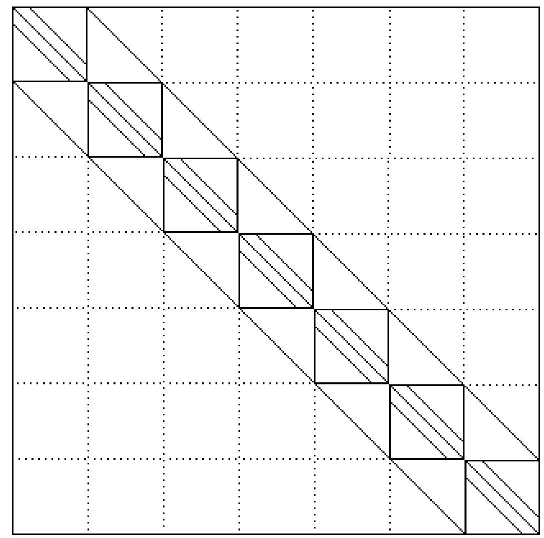
Дашевский Ю.А., Суродина И.В. Эпов М.И. Квази-трехмерное математическое моделирование диаграмм неосесимметричных зондов

Finite difference approximation

$$\frac{1}{r}\frac{\partial}{\partial r}\left(\sigma\ r\frac{\partial U^{a}}{\partial r}\right)+\frac{\partial}{\partial z}\left(\sigma\ \frac{\partial U^{a}}{\partial z}\right)=\frac{1}{r}\frac{\partial}{\partial r}\left((\sigma_{0}-\sigma\ )\, r\frac{\partial U^{0}}{\partial r}\right)+\frac{\partial}{\partial z}\left((\sigma_{0}-\sigma\ )\frac{\partial U^{0}}{\partial z}\right)$$

$$AV=-\frac{1}{r}\left(\bar{r}\,aV_{\bar{r}}\right)_{\hat{r}}-\left(bV_{\bar{z}}\right)_{\hat{z}}$$





$$k(A)=\lambda_{max}/\lambda_{min}$$

- Sparse
- Symmetric
- Positive definite
- Not strictly diagonally dominant matrix

Conjugate gradient method

$k = 0: Initialization:\ x_0, p_0 = r_0 = b - Ax_0$

$k \geq 0: While\, \frac{\|r_k\|}{\|r_0\|} > \varepsilon$

1. $q_k = Ap_k, \alpha_k = \frac{\|r_k\|^2}{{p_k}^T q_k}$
2. $x_{k+1} = x_k + \alpha_k p_k, r_{k+1} = r_k - \alpha_k q_k$
3. $\beta_k = \frac{\|r_{k+1}\|^2}{\|r_k\|^2}, p_{k+1} = r_{k+1} + \beta_k p_k$

- Scalar product
- Norm
- Vector updates
- Matrix vector product

Preconditioned conjugate gradient method

$M^{-1}Ax = M^{-1}b$

$A\,M^{-1}y{=}b,\ x{=}M^{-1}y$

$k = 0: Initialization:\ x_0, r_0 = b - Ax_0, Mz_0 = r_0, p_0 = z_0$

$k \geq 0: While\,\frac{\|r_k\|}{\|r_0\|} > \varepsilon$

1. $q_k = Ap_k, \alpha_k = \frac{{z_k}^T r_k}{{p_k}^T q_k}$
2. $x_{k+1} = x_k + \alpha_k p_k, r_{k+1} = r_k - \alpha_k q_k$
3. $Mz_{k+1} = r_{k+1}$
4. $\beta_k = \frac{{z_{k+1}}^T r_{k+1}}{{z_k}^T z_k}, p_{k+1} = r_{k+1} + \beta_k p_k$

- Need to solve additional linear system at each step
- Additional costs should not outweigh reduction of iterations
- Classical preconditioners (SOR, Incomplete Factorizations) are based on triangular decompositions – sequential task
- Simple preconditioners like Jacobi has limited impact on the efficiency
- Sparse Approximate Inverse – sequential setup phase
- Incomplete Poisson Preconditioner (***M.Ament, G.Knittel, A Parallel Preconditioned Conjugate Gradient Solver for the Poisson Problem on a Multi-GPU Platform***)

Schulz-Hotelling algorithm

$D_0 -$ initial approximation of A^{-1}

With $\|R_0\| \leq q \leq 1$, where $R_0 = I - AD_0$, we can build iterative process

$D_1 = D_0(I + R_0), R_1 = I - AD_1$

$D_2 = D_1(I + R_1), R_2 = I - AD_2$

……………………………………

……………………………………

$D_m = D_{m-1}(I + R_{m-1}), R_m = I - AD_m$

$$\|D_m - A^{-1}\| \leq \|D_0\| \frac{q^{2^m}}{1-q}$$

If $A = A^T$ and $D_0 = D_0{}^T$, then $D_m = D_m{}^T$

G. Schulz, Iterative Berechnung der reziproken Matrix, Z. Angew, Math. Mech. 13 (1933)
H. Hotelling, Analysis of a complex of statistical variables into principal components, J.Educ.Psych.,(1933)

Schulz-Hotelling algorithm

$$D_1 = D_0(I + R_0)$$

$$D_2 = D_0(I + R_0 + R_0^2 + R_0^3)$$

$$D_3 = D_0(I + R_0 + R_0^2 + R_0^4 + R_0^5 + R_0^6 + R_0^7)$$

$$D_m = D_0(I + R_0 + R_0^2 + \ldots + R_0^{2^m - 1})$$

Parallel preconditioners (Jacobi)

Initial inverse approximation:
$D_0 = diag\{{a_{11}}^{-1}, {a_{22}}^{-1}, \dots, {a_{nn}}^{-1}\}$

$D_1 = D_0 + D_0(I - AD_0)$ – symmetric, 5-diagonal

$D_2 = D_1 + D_1(I - AD_1)$ – symmetric, 25-diagonal

$D_2 = 2D_1 + D_1AD_1$

$D_2 = D_1\left(I + {R_0}^2\right), \qquad R_0 = I - AD_0$

$I + {R_0}^2$ - 9-diagonal

$D_3 = D_2 + D_2(I - AD_2) = 2(2D_1 - D_1AD_1) - (2D_1 - D_1AD_1)A(2D_1 - D_1AD_1)$

$D_3 = 2D_1\left(I + {R_0}^2\right) - D_1\left(I + {R_0}^2\right)AD_1\left(I + {R_0}^2\right)$-symmetric, 113-diagonal

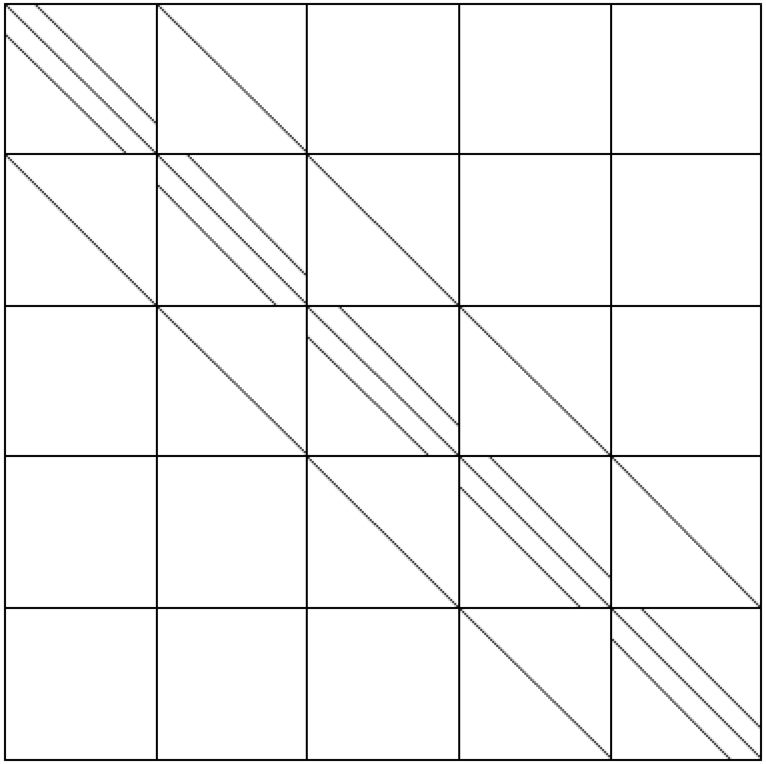
Parallel preconditioners (D_1)

$$D_1 = D_0 + D_0(I - AD_0)$$

$$D_1 3(i) = 1/A3(i)$$

$$D_1 4(i) = -A4(i)/(A3(i+1) * A3(i))$$

$$D_1 5(i) = -A5(i)/(A3(i+1) * A3(i))$$

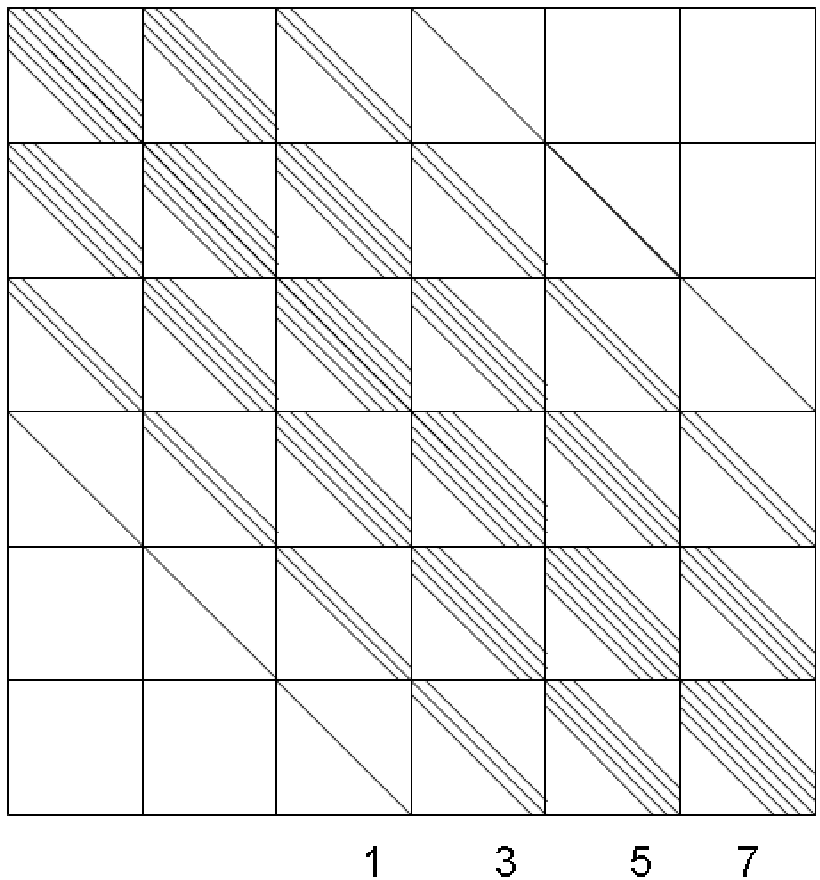


Parallel preconditioners (D_2)

$D_2 = D_1 + D_1(I - AD_1)$ – symmetric, 25-diagonal

$\boldsymbol{D_2 = 2D_1 + D_1 A D_1}$

$D_2 = D_1\left(I + {R_0}^2\right), \quad R_0 = I - AD_0, I + {R_0}^2$ - 9-diagonal



Parallel preconditioners (D_3)

$D_3 = D_2 + D_2(I - AD_2) = 2(2D_1 - D_1AD_1) - (2D_1 - D_1AD_1)A(2D_1 - D_1AD_1)$

$D_3 = 2D_1(I + {R_0}^2) - D_1(I + {R_0}^2)AD_1(I + {R_0}^2)$-symmetric, 113-diagonal

15	13	11	9	7	5	3	1		
13	15	13	11	9	7	5	3	1	
11	13	15	13	11	9	7	5	3	1
9	11	13	15	13	11	9	7	5	3
7	9	11	13	15	13	11	9	7	5
5	7	9	11	13	15	13	11	9	7
3	5	7	9	11	13	15	13	11	9
1	3	5	7	9	11	13	15	13	11
	1	3	5	7	9	11	13	15	13
		1	3	5	7	9	11	13	15

Parallel preconditioners (SSOR)

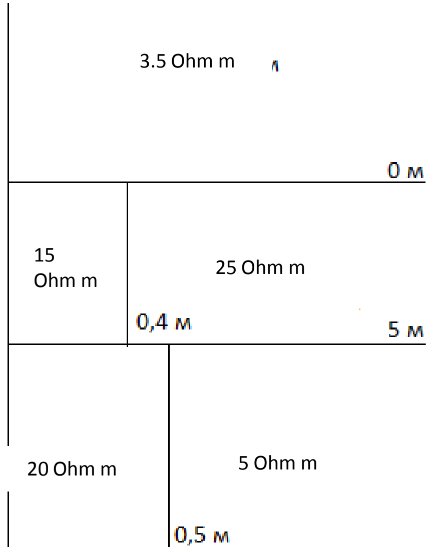
$A = L + D + L^T, D -$ diagonal, $L -$ lower triangular
$M = KK^t$, where $K = \frac{(\overline{D}+L)\overline{D}^{-1/2}}{\sqrt{2-\omega}}, 0 < \omega < 2, \overline{D} = \left(\frac{1}{\omega}\right) D$
$K^{-1} = \sqrt{2 - \omega\overline{D}}(I + \overline{D}^{-1}L)^{-1}\overline{D}^{-1}$
$K^{-1} \approx \sqrt{2 - \omega}\overline{D}^{\frac{1}{2}}[I - \overline{D}^{-1}L + (\overline{D}^{-1}L)^2 - (\overline{D}^{-1}L)^3 + \cdots]\overline{D}^{-1}$
$\overline{K} = \sqrt{2 - \omega}\overline{D}^{\frac{1}{2}}(I - \overline{D}^{-1}L)\overline{D}^{-1} = \sqrt{2 - \omega}\overline{D}^{-\frac{1}{2}}(I - L\overline{D}^{-1})$
SSOR-AI preconditioner is defined as $\overline{M} = \overline{K}^T\overline{K}$

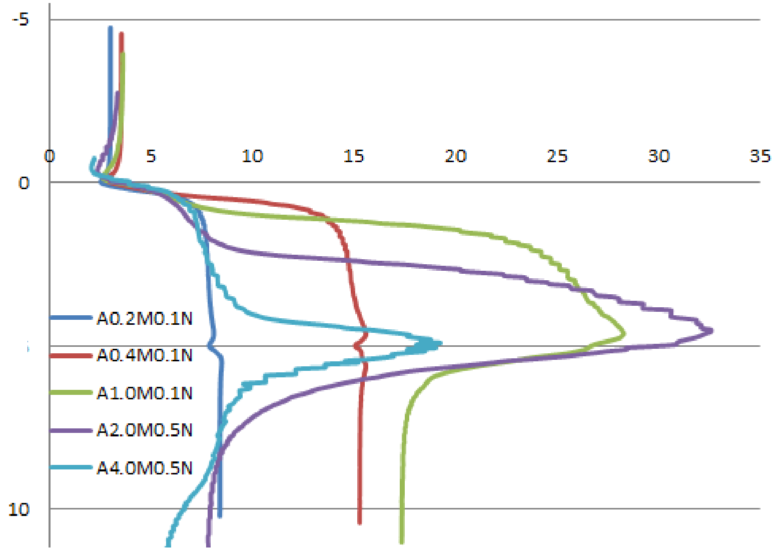
Let's take as D_0 in Schulz-Hotelling series

*-***Rudi Helfenstein, Jonas Koko, Parallel preconditioned conjugate gradient algorithm on GPU***

Numeric experiments

Probes: 5
Tool positions: 100





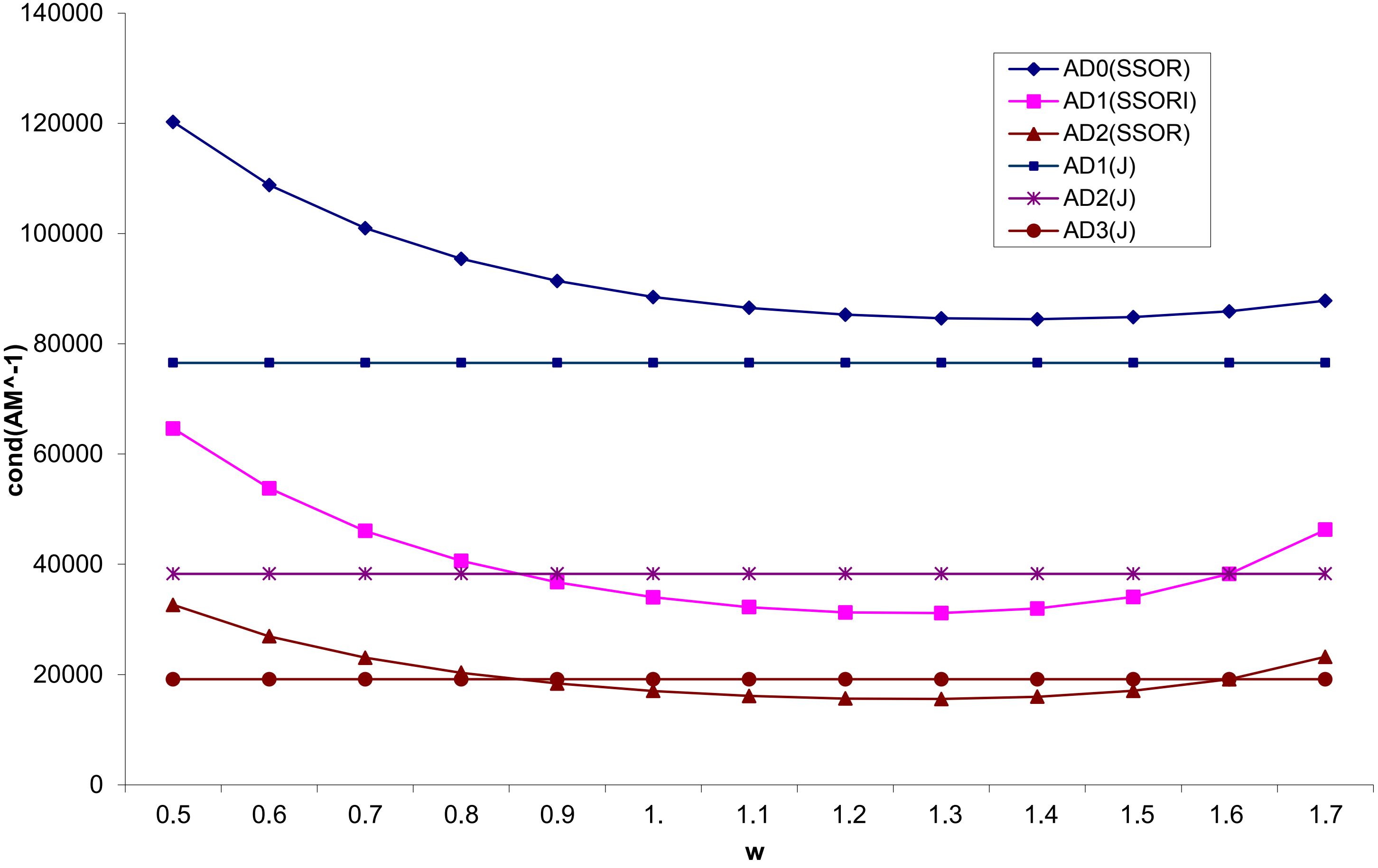
Grid: 88x200, matrix 17600x17600
GPU: NVIDIA® GeForce® GTX 480

Condition number of AM^{-1}

Matrix	Condition number
A	$4.5377 * 10^7$
AJ	$3.0542 * 10^5$
$AD_1(J)$	$7.6542 * 10^4$
$AD_2(J)$	$3.8271 * 10^4$
$AD_3(J)$	$1.9135 * 10^4$
$AD_0(SSOR)$	$8.4605 * 10^4$
$AD_1(SSOR)$	$3.1132 * 10^4$
$AD_2(SSOR)$	$1.5556 * 10^4$

Grid: 88x200, matrix 17600x17600
GPU: NVIDIA® GeForce® GTX 480

Condition number of AM^{-1}



Grid: 88x200, matrix 17600x17600

GPU: NVIDIA® GeForce® GTX 480

Performance

Method	Iterations	Time, sec	Method	Iterations	Time, sec
CG	219322	62,7			
$D_0 = J$	2428	0.477			
$D_1(J)$	1209	0.188	$D_0 = SSOR$	1309	0.228
$D_2(D_1)$	859	0.178	$D_1(SSOR)$	784	0.177
$D_2(R^2)$	859	0.174			
$D_3(D_1)$	608	0.172	$D_2(SSOR)$	554	0.179

	$D_3(D_1)$	$D_1(SSOR)$	$D_2(SSOR)$
Time, sec	14	17	15

Grid: 88x200, matrix 17600x17600

GPU: NVIDIA® GeForce® GTX 480

Conclusion

- A parallel preconditioner is presented
- A sparse approximate inverse is computed explicitly
- Computation of the preconditioner is inherently parallel (well suitable for GPU)

Thank you