A new efficiently and massively parallel algorithm for the solution of elliptic systems of partial differential equations

I. Cordero-Carrión

in collaboration with

J.E. Adsuara, M.A. Aloy and P. Cerdá-Durán

Departamento de Matemáticas, University of Valencia (Spain)

CoCoNuT Meeting, Garching, 2017

Outline

- The Scheduled Relaxation Jacobi (SRJ) method
- Optimal case and analytical solution: the Chebyshev-Jacobi method (CJM)
- Numerical results
- Parallel implementation
- Conclusions

The Scheduled Relaxation Jacobi (SRJ) method

The SRJ method:

- Generalization of the classical Jacobi method to solve systems of linear equations, in particular derived from elliptic partial differential equations (ePDEs).
- Simplicity and robustness like the original Jacobi method (preconditioner).
- Consists of executing a series of weighted Jacobi steps to get an acceleration in the convergence of the method.
- Set of P different ω_i relaxation factors ($\omega_i > \omega_{i+1}$) applied q_i times each factor (Yang and Mittal, JCP 2014):

$$\overbrace{\underbrace{\omega_1 J \ldots \omega_1 J}_{q_1} \underbrace{\omega_2 J \ldots \omega_2 J}_{q_2} \ldots \underbrace{\omega_P J \ldots \omega_P J}_{q_P}}^{M} \underbrace{\omega_1 J \ldots \omega_1 J}_{q_1} \ldots \quad (1)$$

The Scheduled Relaxation Jacobi (SRJ) method

- Original work:
  - Schemes from $P = 2$ to $P = 5$.
  - Higher P not possible: numerical resolution of a non linear system of equations of $O(P^2)$.
  - Higher P seems to imply a faster convergence.
- Proceedings CEDYA 2015 (Adsuara et al.): Technical modifications in the numerical resolution of the system of equations make possible to reach $P = 10$. Confirmation of a faster convergence.
- Adsuara et al., JCP 2015: Analytical resolution of part of the equations. We reached $P = 15$. High values of P develop numerical noise.
- Amplification factor $G(\kappa)$: growth of error from one iteration to the next one ($|G| < 1$ to guarantee convergence).
  - Weighted Jacobi: $G(\kappa) = 1 - \omega\kappa$.
  - SRJ: $G_M(\kappa) = \prod_{n=1}^{P}(1 - \omega_n\kappa)^{q_n}$.

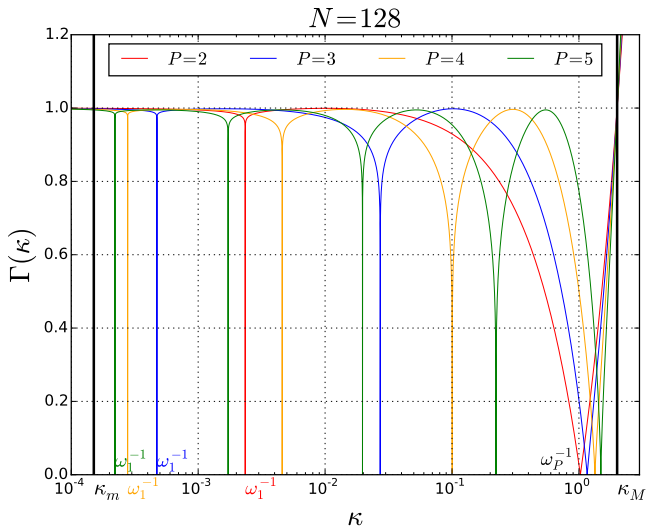
  $\kappa(k_i)$ is a function of the wave-numbers obtained from a von Neumann analysis of the system of linear equations resulting from the discretization of ePDEs by finite differences.

The Scheduled Relaxation Jacobi (SRJ) method

In the original work was argued that the optimal weights and repetition numbers minimize the maximum per-iteration amplification factor:

$$\Gamma_M(\kappa) = \sqrt[M]{|G_M(\kappa)|} = \prod_{n=1}^{P} |1 - \omega_n \kappa|^{\frac{q_n}{M}} \tag{2}$$

$\kappa \in [\kappa_{\min}, \kappa_{\max}]$ correspond to the minimum and maximum weight numbers allowed by the discretization mesh and boundary conditions.



$\Gamma(0) = 1$. For solving the min-max problem, it is imposed that $\Gamma(\kappa_i) = \Gamma(\kappa_{i+1}),\ i = 0, \ldots, P-1$, ($\kappa_i$ are the local extrema).

The Chebyshev-Jacobi method (CJM)

- More analytical simplifications allow us to reach $P = 28$... but resulting in $M = 1800$ for a reasonable resolution $(N = 256)$. Following the procedure of the original paper, increasing P implies smaller values of $\Gamma(\kappa)$ but a total number of iterations M per cycle significantly larger than P.
- The total number of iterations can be chosen to be equal to M without loss of generality. We should find the optimal scheme for fixed values of M minimizing $|G_M(\kappa)|$, and then choose M such that this maximum value is similar to the residual needed to solve a particular problem. The obtained results lead us to conjecture that the optimal SRJ scheme is the one with all weights strictly different, $P = M$.
- There is a unique polynomial satisfying the conditions $G_M(0) = 1$ and $G_M(\kappa_i) = -G_M(\kappa_{i+1}), i = 0, \ldots, M-1$, with $\kappa_0 = \kappa_{\min}$ and $\kappa_M = \kappa_{\max}$, proportional to the Chebyshev polynomial of first kind of degree M, $T_M(\kappa)$:

$$\tilde{G}_M(\tilde{\kappa}) = \frac{T_M(\tilde{\kappa})}{T_M(\tilde{\kappa}(0))}; \qquad \tilde{\kappa} = \frac{2(\kappa - \kappa_0)}{(\kappa_M - \kappa_0)} - 1. \tag{3}$$

The Chebyshev-Jacobi method (CJM)

- The set of weights can be derived from the roots of $T_M(\tilde{\kappa})$:

$$\omega_i = 2\left[\kappa_M + \kappa_0 - (\kappa_M - \kappa_0)\cos\left(\frac{\pi(2i-1)}{2M}\right)\right]^{-1}. \qquad (4)$$

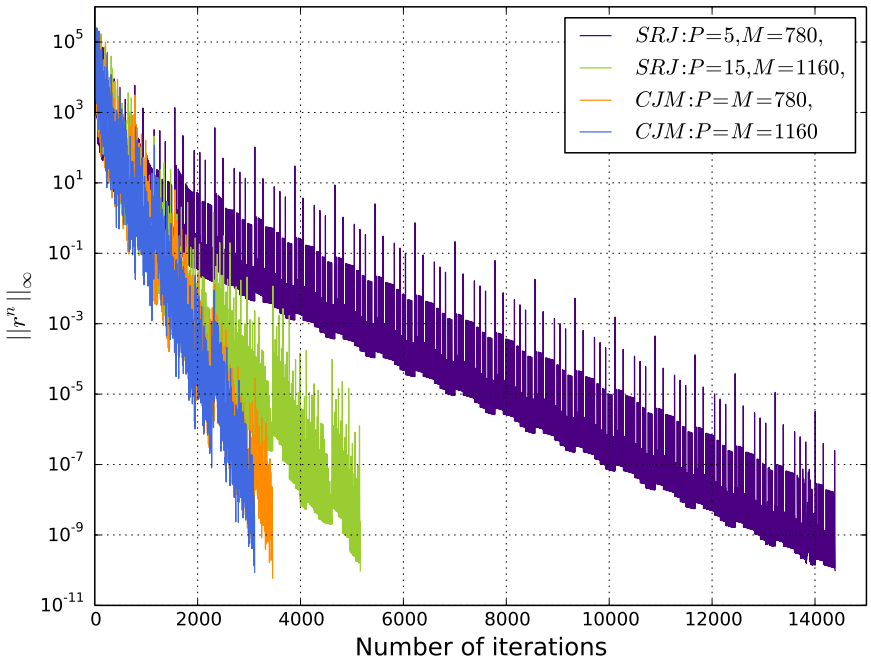
  The resulting scheme (Chebyshev-Jacobi method (CJM), hereafter) is closely related to a Chebyshev iteration or semi-iteration; these methods appeared in the literature as special implementations of the non-stationary or semi-iterative Richardson's method. Two important differences:
  - We do not need to compute the maximum and minimum eigevalues of the corresponding matrix. We use instead a straightforward von Neumann analysis, which can be applied to matrices that are not necessarily consistently ordered (NCO) (for exemple, high-order discretizations).
  - Young tried this procedure but turned out to be unstable, consequence of a bad ordering of the weights. We follow the one from the SRJ method.
- The average rate of convergence of the CJM in a cycle of M iterations is $R_M = \frac{1}{M}\log|T_M(\tilde{\kappa}(0))|$, leading to $|G_M(\kappa)| < 1$.
- Markoff's theorem can be used to demostrate that the CJM is actually the optimal SRJ scheme and that $\max|G_M(\kappa)|$ reduces when increasing M.

Numerical examples: First numerical test

Laplace equation in 2D in Cartesian coordinates with Neumann boundary conditions:

$$\begin{cases} \partial_{xx}u(x,y) + \partial_{yy}u(x,y) = 0, & (x,y) \in (0,1)\times(0,1) \\ \partial_x u(x,y)|_{x=0} = \partial_x u(x,y)|_{x=1} = 0, y \in (0,1) \\ \partial_y u(x,y)|_{y=0} = \partial_y u(x,y)|_{y=1} = 0, x \in (0,1). \end{cases} \tag{5}$$

Second-order 5-point formula for the spatial discretization of the Laplacian operator. 256×256 uniform zones.
$||r^n||_\infty = \text{máx}_{ij}\,|u_{ij}^n - u_{ij}^{n-1}|$.

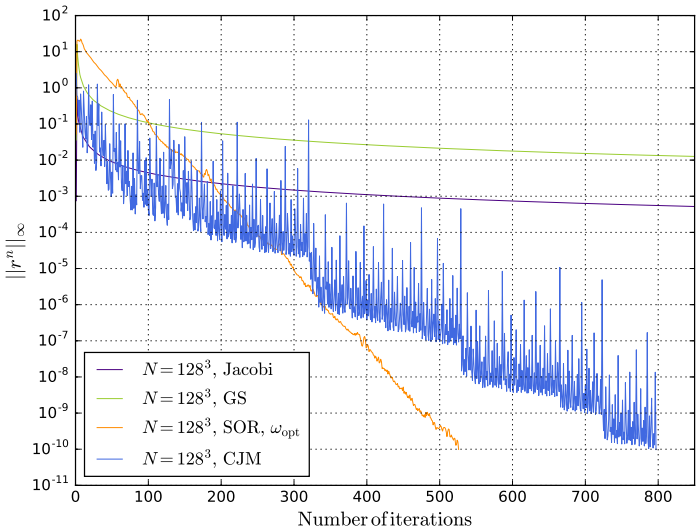


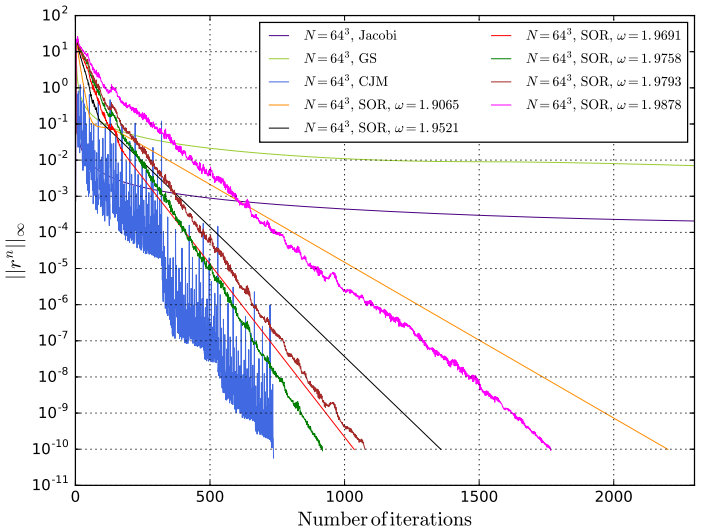
Numerical examples: Second numerical test

Poisson equation with a source term in 3D Cartesian coordinates with Dirichlet boundary conditions:

$$\nabla^2\phi(x,y,z) = -4\pi\rho, \qquad \rho = \frac{Q}{4\pi R^3}. \tag{6}$$

Charge Q. Second-order 7-point stencil. Uniformly grid $N = 128$.
SOR and CJM similar behaviour. SOR less iterations but: (i) optimal SOR weight cannot be computed when dealing with NCO matrices; (ii) CJM is trivially parallelized while SOR requires multicolor schemes.

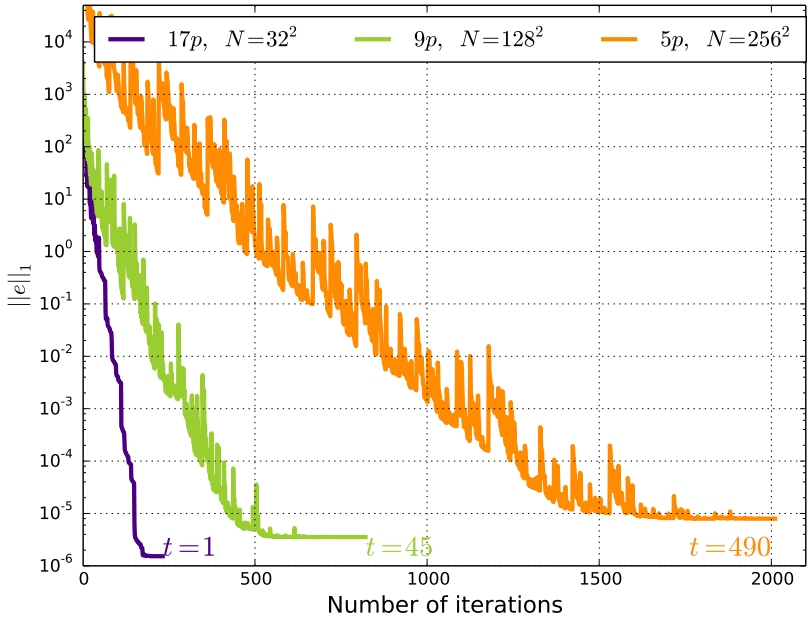




Same equation subject to relection symmetry (homogeneous Neumann boundary conditions). $N = 64$. For CJM, same weights as previous plot.
No analytical expression for optimal SOR weight (very sensitive).

Numerical examples: Third numerical test

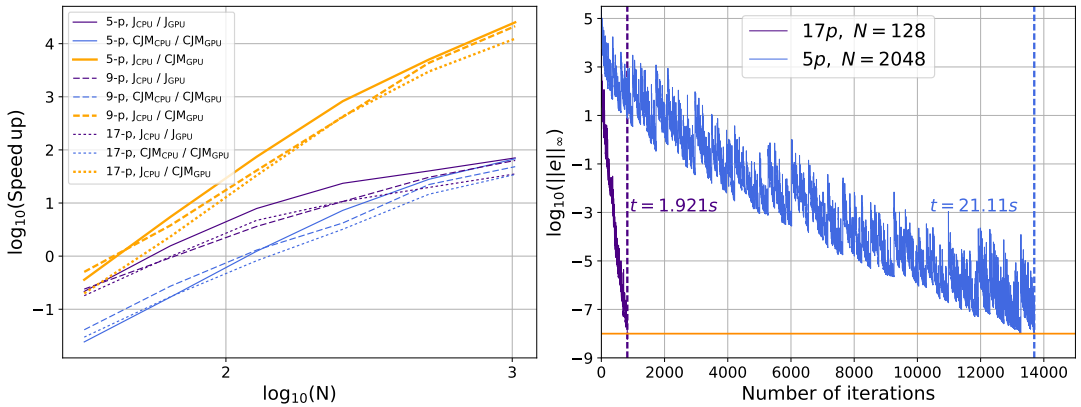
9-point and 17-point discretization of the Laplacian in 2D: $\Delta u = -(x^2 + y^2)\, e^{xy}$, in the unit square with Dirichlet boundary conditions (analytic solution $u(x, y) = -e^{xy}$).
Similar behaviour between CJM and SOR (9-point known, 17-point computed numerically for several values). 9-point needs a four colors parallel implementation. 17-point has a non-unique strategy with different convergence rates. CJM trivially parallelizable.



Not only the number of iterations increases when employing low order discretizations of the Laplacian, but also that the computational time needed to achieve a prescribed norm-1 error is substantially larger.

CJM and GPUs

- CJM perfectly suited for GPUs. CUDA technology of NVIDIA: Tesla K40c (Kp) and a GeForce GTX Titan X (Mx). Mx more recent. Double precision arithmetics works better on Kp. Memory large enough to transfer whole data in one transfer between CPU and GPU device. CUDA Occupancy Calculator.
- Laplacian 2D from last numerical test. $N = 1024$.



- Low resolutions: most of the time consumed by the data transfer. Above a certain turnover, the slopes stabilize in a very similar way.
- Larger improvement for the Jacobi method than for the CJM, but this larger relative speed up reduces with increasing resolution or considering higher-order discretizations. Better speed up factor for CJM extrapolating results (not yet explored) to even higher resolution.
- In addition to the smaller resolution needed for the higher-order discretizations, we have a reduction of one order of magnitude both in the number of iterations and in the computational time.

Conclusions

- Optimal coefficients for the SRJ method resulting in the so-called CJM.
- Easy to implement, robust and analytical calculation of weights a priori (roots of the corresponding Chebyshev polynomials). No need of computation of eigenvalues.
- 2D and 3D ePDEs, high-order discretizations (NCO matrices). Similar performance of CJM and SOR, but no need of multi-coloring parallelization strategy (non-unique).
- Trivial parallelization of CJM. Tested with CUDA in two different GPU arquitectures: speed up by several orders of magnitude in combination with high-order discretizations.