On Preconditioning Nonsymmetric Matrix Iterations

O. Heinreichsberger, S. Selberherr, M. Stiftinger

Technical University Vienna
Institute for Microelectronics
Vienna (Austria)

K.P. Traar

SIEMENS AG Austria
Vienna (Austria)

Fast iterative solution methods for large, sparse, but not necessarily symmetric positive definite linear equations $Ax = b$ usually use the Chebyshev or the conjugate gradient method, or, more generally, a residual minimization method to accelerate the convergence of a basic stationary iterative scheme. Such schemes are obtained by a regular [6] splitting of the coefficent matrix A

$$A = Q - (Q - A) \tag{1.1}$$

and setting

$$Qx_{n+1} = (Q - A)\, x_n + b \tag{1.2}$$

or equivalently

$$x_{n+1} = \left(I - Q^{-1}A\right) x_n + Q^{-1}b \tag{1.3}$$

The convergence of the stationary matrix iterative method (1.3) starting with some initial approximation x_0 will converge the faster, the better the product $Q^{-1}A$ approximates the unity matrix.
The splitting matrix Q, which can be viewed as an easily invertible approximate to A, is also termed the *left preconditioning* matrix. Conversely, setting $y_n = Qx_n$, one obtains the right preconditioned iterative scheme based on the same splitting

$$y_{n+1} = \left(I - AQ^{-1}\right) y_n + b \tag{1.4}$$

The product AQ^{-1} is the *right preconditioning* matrix. If Q can be factored e.g. $Q = Q_1 Q_2$ then a *split preconditioned* iteration may be constructed with $y_n = Q_2 x_n$

$$y_{n+1} = \left(I - Q_1^{-1} A Q_2^{-1}\right) y_n + Q_1^{-1} b \tag{1.5}$$

Since these preconditioning variants base on the same matrix splitting, the iteration matrix G has the same spectral radius in all three cases

$$\rho(G) = \rho\left(I - Q^{-1}A\right) = \rho\left(I - AQ^{-1}\right) = \rho\left(I - Q_1^{-1} A Q_2^{-1}\right) \tag{1.6}$$

and hence the convergence rate is the same in exact arithmetic. For certain matrix structures the split preconditioning (1.5) has algorithmic advantages. In this work we present a comparison of preconditioning matrices Q applied to the solution of drift–diffusion type equations arising in semiconductor device simulators.
The selection of a specific Q is guided by the following imperatives:

1. Q should approximate the spectrum of A.
2. Q should easily be computable.
3. Computing e.g. $q = Q^{-1}Ap$,$q = AQ^{-1}p$ or $q = Q_1^{-1} A Q_2^{-1} p$ should be computationally feasible. Since the basic linear operations ('BLAS') in iterative solvers such as matrix-vector multiplies, vector updates and dotproducts can be executed very fast on vector computers, also preconditioning backsolves should be vectorizable, a requirement that is often not easy to meet.

The following preconditioners will be discussed [2][3]:

1. $Q = I$ the null preconditioner. In this case (1.3) is the Richardson method.
2. $Q = D$ the (block) Jacobi preconditioner. D is the (block) tridiagonal part of A.
3. $$Q = \frac{1}{2-\omega}\left(\frac{1}{\omega}D - L\right)\left(\frac{1}{\omega}D\right)^{-1}\left(\frac{1}{\omega}D - U\right) \tag{1.7}$$
   the SSOR preconditioner. L,U are the stricly upper and lower triangular parts of A, and ω is the overrelaxation parameter in the interval $[1, 2]$.

4. $Q = \left(\tilde{D} + \tilde{L}\right) \tilde{D}^{-1} \left(\tilde{D} + \tilde{U}\right)$ the incomplete LU factorization (ILU) preconditioner. $\tilde{L}$ and $\tilde{U}$ are strictly lower and upper triangular matrices and $\tilde{D}$ is (block) diagonal. A prescribed sparsity pattern for the triangular factors allow an approximation of A of different levels. The diagonal matrix $\tilde{D}$ is constructed such that certain algebraic properties of the linear operator are preserved, most important of which is the requirement [5]:

$$diag\,(Q) = diag\,(A) \tag{1.8}$$

or alternatively Gustafsson's modification [4]:

$$columnsum\,(Q) = columnsum\,(A) \tag{1.9}$$

or a parametrized combination of both.
It will be shown on representative test examples taken from the three-dimensional device simulator MINIMOS, that the incomplete factorization preconditioners of level 1 exhibit a computational optimum, concerning storage requirements and arithmetic work for the incomplete factorization at the beginning of the iteration and the particular backsolves at every iteration.
We present implementations on vector supercomputers [1] such as the CRAY-2, the FUJITSU VP200, vector-concurrent mini-supercomputers such as the ALLIANT FX40, and a 6-processor parallel DIGITAL VAX 6260 computer.

References

[1] Ashcraft, C.C, Grimes, R.G., "*On Vectorizing Incomplete Factorization and SSOR Preconditioners*" SIAM Journal of Scientific and Statistical Computing, Vol. 9, No. 1, January 1988, pp. 122-151.

[2] Concus, P., Golub, G.H., Meurant, G., "*Block Preconditioning for the Conjugate Gradient Method*" SIAM Journal of Scientific and Statistical Computing, Vol. 6, No. 1, January 1985, pp. 220-252.

[3] Doi, S., Harada, N., "*Tridiagonal Factorization Algorithm: A Preconditioner for Nonsymmetric System Solving on Vectorcomputers*" Journal of Information Processing, Vol. 11, No. 1, 1987.

[4] Gustafsson, I., "*A Class of First Order Factorization Methods*" BIT, 18 (1978), pp. 142-156.

[5] Meijerink, H., Vorst, H., "*An Iterative Solution Method for Linear Systems of which the Coefficient Matrix is a Symmetric M-Matrix*", Math. Comp., 31 (1977), pp. 148-162

[6] Varga, R., "*Matrix Iterative Analysis*", Prentice Hall, 1962