World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:7, No:11, 2013

The Performance of Alternating Top-Bottom Strategy for Successive Over Relaxation Scheme on Two Dimensional Boundary Value Problem

M. K. Hasan, Y. H. Ng, J. Sulaiman

***Abstract*—**This paper present the implementation of a new ordering strategy on Successive Overrelaxation scheme on two dimensional boundary value problems. The strategy involve two directions alternatingly; from top and bottom of the solution domain. The method shows to significantly reduce the iteration number to converge. Four numerical experiments were carried out to examine the performance of the new strategy.

***Keywords*—**Two dimensional boundary value problems, Successive Overrelaxation scheme, Alternating Top-Bottom strategy, fast convergence

I. Introduction

BOUNDARY value problem have been representing many real world problems such as electrostatics, wave travelling in guided conductors, torsion problem for beam, and heat phenomena. Research in boundary value problems was pioneered by Kneser in 1896 and followed by Mambriani in 1929 and then only by others; Hartman and Wintner in 1951, Pui-Kei 1963, Gross in 1963, Bebernes and Jackson in 1967, Seda in 1980, Srzednicki in 1987 and many others [1].

Successive Overrelaxation (SOR) is one of the important breakthroughs in numerical analysis. The method was proposed by Young [2] in his Ph. D. thesis at Harvard University. There are a lot of modification have been done to the SOR method over the few recent years. Othman in his Ph. D. thesis at Universiti Kebangsaan Malaysia for an instance, have implement the concept of half-sweep and quarter-sweep to the SOR [3]. The implementations succeed to increase the speed of computation of SOR scheme. Sulaiman et al. [4] re-construct the SOR scheme using triangular finite element and hybrid it with the Explicit Decouple Group (EDG) method and Red-Black ordering strategy.

Recently, Ng and Hasan [5] have investigate the implemention of half-sweep and quarter-sweep approach on SOR and its siblings; the Accelerative Over Relaxation (AOR) to solve two points boundary value problems.

In our previous paper, we have proposed a new ordering strategy. The strategy was called the Alternating Left Right (ALR) strategy and applied it on SOR method [6]. The ALR strategy successfully reduces the iteration number significantly. In this paper, we proposed new alternative ordering strategy; Alternating Top-Bottom (ATB) strategy for solving boundary value problems given in (1).

$$\nabla^2 U = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = f(x,y), \quad (x,y) \in \Omega. \tag{1}$$

with boundary condition given by

$$U(x,y) = g(x,y), \quad (x,y) \in \partial\Omega \tag{2}$$

The rest of this paper is organized as follows. The derivations of method are explained in Section II.

II. Re-Inventing Generalized Successive Overrelaxation Scheme

In this section, we discussed the development of generalized SOR formula. We discretize problem (1) with central difference approximation given in (3) and (4).

$$\frac{\partial^2 U}{\partial x^2} \approx \frac{U_{i-1,j} - 2U_{i,j} + U_{i+1,j}}{h^2}, \tag{3}$$

$$\frac{\partial^2 U}{\partial y^2} \approx \frac{U_{i,j-1} - 2U_{i,j} + U_{i,j+1}}{g^2}. \tag{4}$$

Replacing (3) and (4) in (1) gives

$$U_{i,j} = \frac{g^2 U_{i-1,j} + g^2 U_{i+1,j} + h^2 U_{i,j-1} + h^2 U_{i,j+1} - g^2 h^2 f_{i,j}}{2\left(h^2 + g^2\right)}$$

We introduced the relaxation parameter *w*, to the equation to link the current results *k+1* and previous results *k* to arrive at the generalized SOR formulation [5].

$$U_{i,j}^{k+1} = w\left(\frac{g^2 U_{i-1,j}^{k+1} + g^2 U_{i+1,j} + h^2 U_{i,j-1}^{k+1} + h^2 U_{i,j+1} - g^2 h^2 f_{i,j}}{2\left(h^2 + g^2\right)}\right) + (1-w)U_{i,j}^{k}. \tag{5}$$

M. K. Hasan is with the School of Information Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia (corresponding author; phone: 603-89216652; fax: 603-89256732; e-mail: khatim@ ftsm.ukm.my).

Y. H. Ng,is a master student with the School of Information Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia (e-mail: fido_ng87@ yahoo.com).

J. Sulaiman is with the School of Science Technology, Universiti Malaysia Sabah, Locked Bag 2073, 88999 Kota Kinabalu, Sabah, Malaysia (e-mail: jumat@ums.edu.my).

Special thanks to Ministry of Education for FRGS grant to execute this research.

The SOR formulation used to approximate all node points in discretize solution domain given in Fig. 1.

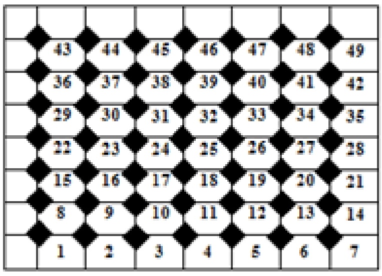


Fig. 1 Discretized solution domain using SOR scheme

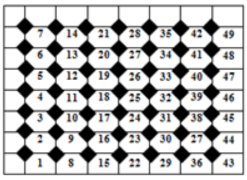
The algorithm for the SOR scheme follows Algorithm 1.

Algorithm 1. SOR algorithm

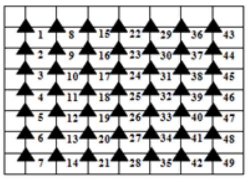
i. Define solution domain as Fig. 1.
ii. Initialize all parameters and matrices
iii. Start timing
iv. While not convergent, then,
   a. Calculate every ◆ using (5)
   b. Assign $U_{i,j}^{k} \leftarrow U_{i,j}^{k+1}$
   c. Update iteration
v. End timing
vi. Display output

III. The Alternating Top-Bottom Strategy

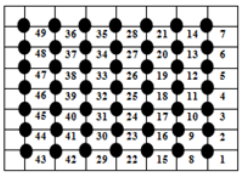
The Alternating Top-Bottom strategy proposed in this paper needs four set of solution domain as given in Fig. 2.



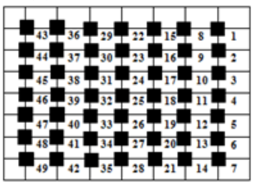
(a) solution domain 1



(b) solution domain 2



(c) solution domain 3



(d) solution domain 4

Fig. 2 Discretized solution domain using SOR scheme with ATB strategy

The direction of the solutions is given by the ordering in each solution domain. Nodes in solution domain 1 were solved first then nodes in solution domain 2 until nodes in solution domain 4 were solved.

Equation (5) can be used to solve nodes in solution domain 1 of Fig. 2 (a) but cannot be used to solve other solution domains. Modifications to (5) was made to enable nodes in other solution domains can be solved. To solve nodes in Fig. 2 (b), (5) was modified into (6).

$$U_{i,n-j}^{k+1} = w\left(\frac{g^2U_{i-1,n-j}^{k+1} + g^2U_{i+1,n-j} + h^2U_{i,n-j-1}^{k} + h^2U_{i,n-j+1}^{k+1} - g^2h^2 f_{i,n-j}}{2\left(h^2+g^2\right)}\right) + (1-w)U_{i,n-j}^{k}. \tag{6}$$

To solve nodes in Fig. 2 (c), (5) was modified into (7).

$$U_{n-i,j}^{k+1} = w\left(\frac{g^2U_{n-i-1,j}^{k} + g^2U_{n-i+1,j}^{k+1} + h^2U_{n-i,j-1}^{k+1} + h^2U_{n-i,j+1} - g^2h^2 f_{n-i,j}}{2\left(h^2+g^2\right)}\right) + (1-w)U_{n-i,j}^{k}. \tag{7}$$

To solve nodes in Fig. 2 (d), (5) was modified into (8).

$$U_{n-i,n-j}^{k+1} = w\left(\frac{g^2U_{n-i-1,n-j}^{k} + g^2U_{n-i+1,n-j}^{k+1} + h^2U_{n-i,n-j-1}^{k} + h^2U_{n-i,n-j+1}^{k+1} - g^2h^2 f_{n-i,n-j}}{2\left(h^2+g^2\right)}\right) + (1-w)U_{n-i,n-j}^{k}. \tag{8}$$

Algorithm 2. SOR-ATB algorithm

i. Define solution domain as Fig. 2.
ii. Initialize all parameters and matrices
iii. Start timing
iv. While not convergent GLOBALLY, then,
   a. While not converge for ◆
      i. Calculate using (5)
      ii. Assign $U_{i,j}^{k} \leftarrow U_{i,j}^{k+1}$
   b. While not converge for ▲
      i. Calculate using (6)
      ii. Assign $U_{i,n-j}^{k} \leftarrow U_{i,n-j}^{k+1}$
   c. While not converge for ●
      i. Calculate using (7)
      ii. Assign $U_{n-i,j}^{k} \leftarrow U_{n-i,j}^{k+1}$
   d. While not converge for ■
      i. Calculate using (8)
      ii. Assign $U_{n-i,n-j}^{k} \leftarrow U_{n-i,n-j}^{k+1}$
   e. Update iteration
      i. End timing
      ii. Display output

IV. Performance Analysis

To analyze the performance of ATB strategy on SOR scheme, we conducted four experiments as listed below.

1. $f(x,y) = -2, \Re, (x,y) \in (0,1) \times (0,1),$
   $u(x,0) = u(x,1) = x(1-x), u(0,y) = 0,$
   $u(1,y) = \sinh(\pi)\sin(\pi(y)).$
2. $f(x,y) = xe^y, \Re, (x,y) \in (0,2) \times (0,1),$
   $u(x,0) = x, u(x,1) = xe, u(0,y) = 0, u(2,y)2e^y.$
3. $f(x,y) = (x^2 + y^2)e^{xy}, \Re, (x,y) \in (0,2) \times (0,1),$
   $u(x,0) = u(0,y) = 1, u(x,1) = e^x, u(2,y) = e^{2y}.$
4. $f(x,y) = (x^2 + y^2)e^{xy},$
   $\Re, (x,y) \in (0,\pi) \times (0, \frac{\pi}{2}),$
   $u(x,0) = \cos(x), u(x, \frac{\pi}{2}) = 0, u(0,y) = \cos(y),$
   $u(\pi,y) = -\cos(y).$

Throughout the experiments, we utilized w between 1.135 and 1.257. We use $\max\left\{\left|u_{i,j}^{k+1} - u_{i,j}^{k}\right|\right\} \leq 10^{-10}$ as the stopping criteria for all experiment.

TABLE I
COMPARISON BETWEEN SOR AND SOR WITH ATB FOR EXPERIMENT 1

Method	$n \times n$	*k*	*t* (sec.)
SOR	3136	3515	239.06
	6724	7236	1051.53
	11664	12174	3018.51
	17956	18286	7043.44
	25600	25536	13121.32
	34596	33898	24478.06
	44944	43347	37715.62
SOR-ATB	3136	1050	209.92
	6724	1924	792.06
	11664	2957	2159.50
	17956	4112	4722.21
	25600	5359	8503.59
	34596	6675	14287.86
	44944	8038	22153.93

From results display in Table I, we analyze the improvement in iteration and in computing time. Additionally, we also analyze computing time per iteration for both classical SOR and SOR-ATB. By applying ATB ordering strategy, the convergences achieved faster by 70.12% to 81.46% in iteration. However the complexity of SOR-ATB is higher than SOR since in SOR-ATB four equations have to be evaluated in an iteration compared to SOR's. This scenario increases the computing time in iteration as displayed in Table V. The effect of higher complexity per iteration appears on the computing time for SOR-ATB which was also faster by 12.19% to 41.63% but not as much gain as in iteration number.

Results displayed in Table II show that for experiment 2, by applying ATB ordering strategy, the convergences achieved faster by 78.05% to 86.62% in iteration. The computing time for SOR-ATB was also faster by 30.33% to 54.00%. While the computing time per iteration displayed in Table V.

TABLE II
COMPARISON BETWEEN SOR AND SOR WITH ATB FOR EXPERIMENT 2

Method	$n \times n$	*k*	*t* (sec.)
SOR	3136	3444	282.52
	6724	7084	1205.98
	11664	11912	3509.93
	17956	17883	8166.33
	25600	24963	16079.30
	34596	33124	28178.45
	44944	42343	45679.57
SOR-ATB	3136	753	196.81
	6724	1358	736.22
	11664	2076	1958.54
	17956	2883	4655.10
	25600	3761	8010.41
	34596	4693	12960.70
	44944	5666	21202.02

TABLE III
COMPARISON BETWEEN SOR AND SOR WITH ATB FOR EXPERIMENT 3

Method	$n \times n$	*k*	*t* (sec.)
SOR	3136	3448	383.70
	6724	7095	1625.00
	11664	11931	4729.20
	17956	17914	11037.96
	25600	25008	21735.59
	34596	33185	38430.47
	44944	42423	61863.18
SOR-ATB	3136	755	283.90
	6724	1361	1057.13
	11664	2082	2881.87
	17956	2894	6180.34
	25600	3776	11329.61
	34596	4714	19489.29
	44944	5695	30070.54

Table III shows that in experiment 3, the implementation of ATB ordering strategy to SOR method improves the convergence by 78.10% to 86.58% in iteration. The computing time for SOR-ATB was also faster by 26.01% to 51.39%. While the computing time per iteration increase as displayed in Table V.

Experiment 4 results displayed in Table IV. The results justify that implementing the ATB ordering strategy, the convergences achieved faster by 83.65% to 87.69% in iteration. The computing time for SOR-ATB was also faster by 44.20% to 55.06%. While the computing time per iteration displayed in Table V.

TABLE IV
COMPARISON BETWEEN SOR AND SOR WITH ATB FOR EXPERIMENT 4

Method	$n \times n$	*k*	*t* (sec.)
SOR	3136	2642	326.19
	6724	5226	1327.25
	11664	8512	3760.83
	17956	12438	8399.54
	25600	16960	16343.66
	34596	22054	28509.24
	44944	27708	44138.96
SOR-ATB	3136	432	182.01
	6724	796	705.39
	11664	1231	1891.08
	17956	1720	4147.70
	25600	2253	7781.82
	34596	2819	12813.06
	44944	3410	20115.82

TABLE V
COMPARISON OF COMPUTING TIME PER ITERATION FOR SOR AND SOR-ATB

Method	$n \times n$	Exp 1	Exp 2	Exp 3	Exp 4
SOR	3136	0.06801	0.08203	0.11128	0.12346
	6724	0.14531	0.17024	0.22903	0.25397
	11664	0.24794	0.29465	0.39637	0.44182
	17956	0.38518	0.45665	0.61616	0.67531
	25600	0.51383	0.64412	0.86914	0.96365
	34596	0.72210	0.85069	1.15806	1.29270
	44944	0.87008	1.07879	1.45824	1.59300
SOR-ATB	3136	0.19992	0.26137	0.37602	0.42130
	6724	0.41167	0.54213	0.77672	0.88616
	11664	0.73029	0.94341	1.38418	1.53621
	17956	1.14839	1.61467	2.13557	2.41145
	25600	1.58678	2.12986	3.00042	3.45398
	34596	2.14050	2.76170	4.13434	4.54525
	44944	2.75615	3.74197	5.28016	5.89906

V. CONCLUSION

In this paper, we have implement SOR method with ATB strategy. The method has improves the iteration number and computing time needed to converge. However the strategy increase number of function evaluation per iteration.

ACKNOWLEDGMENT

Authors thanks to various reviewers' comments to improve the paper.

REFERENCES

[1] J. Andres, G. Gabor, and L. Gorniewicz, "Boundary Value Problems on Infinite Intervals," Transactions of The American Mathematical Society, vol. 351, no. 12, pp. 4861-4903, August 1999.
[2] D. M. Young, Iterative Methods for Solving Partial Difference Equations of Elliptic Type, Ph. D. Thesis, Harvard University Cambridge. Massachusets (unpublished).
[3] M. Othman, Implementation of Parallel and Sequential Algorithms for Solving Linear System Iteratively on Shared Memory Multiprocessors Computer System, Ph. D. Thesis, Universiti Kebangsaan Malaysia, Bangi (unpublished).
[4] J. Sulaiman, M. Othman, and M. K. Hasan, "Red-Black EDG-SOR Iterative Method using Triangle Finite Element Approximation for 2D Poisson Equations," in 2007 Computational Science and Its Applications, LNCS 4707, Part III, pp. 298-308.
[5] Y. H. Ng, and M. K. Hasan, "Investigation of Steady State Problems via Quarter Sweep Schemes," Sains Malaysiana, vol. 42, no. 6, pp. 837-844, June 2013.
[6] Y. H. Ng, and M. K. Hasan, "Steady State Simulator using Alternate Left Right Approach," in 2013 AIP Conf. Proceedings of 20th National Symposium on Mathematical Sciences, AIP Conf. Proc. 1522, pp. 757-761.