SOLUTION OF SINGULAR PERTURBATION PROBLEMS VIA THE DOMAIN DECOMPOSITION METHOD ON SERIAL AND PARALLEL COMPUTERS

Paul A. Farrell*
Department of Mathematics & Computer Science
Kent State University
Kent, OH 44242, U.S.A.

Igor P. Boglaev Vadim V. Sirotkin
Institute of Microelectronics Technology
Russian Academy of Sciences, Moscow District, 142432
Chernogolovka, Russia.

Abstract:- Iterative algorithms for domain decomposition suitable for the efficient solution of singular perturbation problems are considered. Convergence results are established. Numerical examples are presented to demonstrate the effectiveness of the iterative algorithms on parallel computers.

1. Introduction

We are interested in iterative algorithms for domain decomposition, which reduce the given problem to sequences of boundary value problems on appropriate subdomains. An iterative algorithm highly suitable for parallel computing has been constructed in [1]. Here, we analyze and illustrate this algorithm for the solution of singularly perturbed two-point boundary value problems.

Consider the following two semi-linear singular perturbation problems. The first one is the selfadjoint problem

$$L_\epsilon u(x) \equiv \epsilon u'' = f(x,u), \qquad x \in \Omega, \qquad \Omega = (0,1), \tag{1a}$$

$$u(0) = u_0, \qquad u(1) = u_1, \tag{1b}$$

$$f_u \geq \beta_0^2,\ \beta_0 = \text{const} > 0, \qquad (f_u = \partial f/\partial u), \tag{1c}$$

where $\epsilon \equiv \mu^2$, μ is a small positive parameter. The solution of (1a)-(1c) has boundary layers at $x = 0, 1$. The width of these boundary layers is of the order of $h_\epsilon = \mu \mid \ln(\mu) \mid /\beta_0$. For simplicity, we assume here that the solution $u(x)$ exhibits a boundary layer only at $x = 0$ (that is that the reduced solution satisfies the boundary condition (1b)).

The second problem is the non-selfadjoint problem

$$L_\epsilon u(x) \equiv \epsilon u'' + \alpha(x)u' = f(x,u), \qquad x \in \Omega, \qquad \Omega = (0,1), \tag{2a}$$

$$u(0) = u_0, \qquad u(1) = u_1, \tag{2b}$$

$$\alpha(x) \geq \alpha_0 = \text{const} > 0, \qquad f_u \geq 0, \tag{2c}$$

where ϵ is a small positive parameter. The solution of (2a)-(2c) has a boundary layer at $x = 0$ of width $h_\epsilon = \epsilon \mid \ln(\epsilon) \mid /\alpha_0$.

In this paper we consider two iterative algorithms based on domain decomposition for the solution of the semi-linear singularly perturbed problems (1) and (2). Domain decompositions and iterative algorithms for these problems are introduced in section 2. Section 3 gives convergence results for these algorithms. In section 4 we present numerical examples and compare the performance of these iterative algorithms on serial and parallel computers.

2. Iterative Algorithms

We introduce the decomposition of the domain $\Omega = (0,1)$ into two overlapping subdomains Ω_1 and Ω_2:

$$\Omega_1 = (0, x_r), \qquad \Omega_2 = (x_l, 1), \qquad 0 < x_l < x_r < 1.$$

Now consider two sequences of functions $\{v^n(x)\}$, $\{w^n(x)\}$, $n \geq 1$, satisfying the equations:

$$L_\epsilon v^n(x) = f(x,v^n), \qquad x \in \Omega_1, \quad v^n(0) = u_0, \quad v^n(x_r) = v_r^n, \tag{3a}$$

$$L_\epsilon w^n(x) = f(x,w^n), \qquad x \in \Omega_2, \quad w^n(x_l) = w_l^n, \quad w^n(1) = u_1. \tag{3b}$$

Here L_ϵ is defined by (1a) or (2a) and u_0, u_1 by (1b) or (2b), respectively.

We now construct two iterative algorithms. The first one, A1, is the normal Schwarz alternating procedure. The boundary conditions v_r^n, w_l^n from (3a), (3b) are defined by

$$v_r^{n+1} = w^n(x_r), \qquad w_l^n = v^n(x_l), \tag{4}$$

where an initial guess v_r^1 must be prescribed.

The second algorithm, A2, is constructed using the interfacial problem

$$L_\epsilon z^n(x) = f(x,z^n), \qquad x \in \Omega_{\text{inf}} = (X_l, X_r), \tag{5a}$$

$$z^n(X_l) = v^n(X_l), \qquad z^n(X_r) = w^n(X_r),$$

where $X_l < x_l < x_r < X_r$. Here the boundary conditions from $(3a), (3b)$ are determined by the following condition rather than by (4),

$$v_r^{n+1} = z^n(x_r), \qquad w_l^{n+1} = z^n(x_l), \tag{5b}$$

where the initial guesses v_l^1 and w_r^1 are given.

Algorithm A1 is a serial procedure, since the solution v^n of (3a) must be obtained in order to determine the boundary condition $w_l^n = v^n(x_l)$ used in (3b). Thus (3a) and (3b) are executed in lockstep fashion. Algorithm A2 can however be carried out by parallel processing, since on each iteration step problems (3a) and (3b) can be solved concurrently to give both v^n and w^n.

3. Convergence of the Iterative Algorithms

We now formulate convergence results for algorithms A1 and A2.

Theorem 1. *If $x_l < x_r$, then the iterative algorithm (3), (4) (that is the Schwarz alternating procedure) converges to the solutions of problems (1) and (2) with linear rates $0 < q_\mu^{A1} < 1$ and $0 < q_\epsilon^{A1} < 1$ respectively, where*

$$q_\mu^{A1} = \exp[-2\beta_0(x_r - x_l)/\mu],$$
$$q_\epsilon^{A1} = \exp[-\alpha_0(x_r - x_l)/\epsilon].$$

Theorem 2. *If $X_l < x_l < x_r < X_r$, then the iterative algorithm (3), (5) converges to the solutions of problems (1) and (2) with linear rates $0 < q_\mu^{A2} < 1$ and $0 < q_\epsilon^{A2} < 1$ respectively, where*

$$q_\mu^{A2} = \max\{\exp[-\beta_0(X_r - x_l)/\mu], \exp[-\beta_0(x_r - X_l)/\mu]\},$$
$$q_\epsilon^{A2} = \max\{\exp[-\alpha_0(X_r - x_l)/2\epsilon], \exp[-\alpha_0(x_r - X_l)/2\epsilon]\}.$$

Remark. Theorems 1 and 2 can be generalized straightforwardly to multi-domain decomposition.

4. Numerical Results

We emphasize here, as is clear from Theorem 2, that the convergence results for algorithm A2 are independent of the singularly perturbed character of problems (1) and (2). To construct effective numerical methods for algorithm A2, it is necessary to take into account the fact that the solution of problems (1) and (2) have a boundary layer of size h_ϵ at $x = 0$.

We introduce the "natural" decomposition of the original domain Ω in which the boundary layer is localized in subdomain Ω_1 and the region where the solution is smooth is included in Ω_2, that is we require:

$$x_l \geq h_\epsilon. \tag{6}$$

Effective numerical methods for singular perturbation problems, such as those based on special nonuniform grids (*cf.* [2],[3]), exhibit the property of uniform convergence with respect to the small parameter. These special grids are constructed in such a way that the number of grid points inside the boundary layers is approximately equal to the number of grid points outside the layers. Thus, if (6) holds and, on subdomains Ω_1, Ω_2 special nonuniform grids are used, the computational cost of the numerical method for problem (3a) on Ω_1 is approximately equal to that for (3b) on Ω_2. This property is very important for implementation of algorithm A2 on parallel computers,

*Supported by The Research Council of Kent State University

since it avoids loss of efficiency due to one processor being idle. The size of the overlap domain $[x_l,\ x_r]$ and hence of the interfacial region $[X_l,\ X_r]$ also affects the cost of an A2-iteration, since the solution of the problem on the interfacial region represents a sequential part of the A2 algorithm. It also worth mentioning, that condition (6) decreases the number of grid points needed for the interfacial problem (5) thus minimizing the time for its solution.

We now present the results of some numerical experiments using iterative algorithms A1 and A2. In the case of algorithm A2 we shall consider the implementation on a shared memory multiprocessor, the Sequent Balance B21000, in the Department of Mathematics and Computer Science at Kent State University. This has 26 processors, each with a 32-bit National Semiconductor NS32032 capable of 0.75 MIPS, and 32 MB of shared memory. The Balance operating system, DYNIX, provides the ability to bind processes to processors, using the processor affinity facility, and also a utility `team`, which modifies system parameters to permit more accurate timings. The latter gives the highest priority to the program and disables swapping, page fault frequency adjustments and process aging. These privileges allow a user program to execute with a minimum of system overhead to distort benchmark times. The coding used the Sequent parallel directives to parallelize do loops in the Fortran code, library calls to the microsecond clock for the timings, and to the parallel processing library to set and manage the number of processors and to synchronize after the solution of the interfacial problem.

Example 1. We consider problem (1), where $f(x,u) = 1 - e^{-u}$, $u_0 = 1$, $u_1 = 0$. Introduce a non-equidistant grid $\omega_x = \{x_i,\ 0 \le i \le N_x\}$. The subdomains Ω_1, Ω_2 and $\Omega_{\inf}$ from (3), (5a) are chosen in the forms: $x_l = h_\epsilon = x_j$, $x_r = x_k$, $0 < j < k < N_x$, $k - j \ge 1$, $X_l = x_{j-1}$, $X_r = x_{k+1}$.

In the boundary layer $[0, h_\epsilon]$, the mesh generating function is a logarithmic type function similar to that given in [2]. We approximate the differential equation (1a) by a simple variable-mesh difference formula. The nonlinear algebraic systems (after discretization of (3) and (5)) are solved by a one-step Newton method. In Table 1 and Table 3, we give the number of iterations, to achieve an error of 10^{-5}, for the direct (undecomposed) method from [2], K_s, and for iterative algorithms A1 and A2, K_{A1} and K_{A2} respectively, for various μ and overlapping interval sizes $h = x_r - x_l$. In Table 1 the number of mesh points $N_x = 101$, $j = 51$ and $k \ge 52$ and in Table 3 $N_x = 501$, $j = 251$ and $k \ge 252$. It should be noted that these experiments indicate that the number of iterations is bounded independent of ϵ and is approximately constant for sufficiently small ϵ. Table 2 and Table 4 give

	K_s	K_{A1}					K_{A2}				
μ \ h		.1	.05	.01	.005	.001	.1	.05	.01	.005	.001
0.1	4	6	10	33	58	202	8	11	26	40	121
0.01	4	4	4	4	4	4	4	4	4	5	10
0.001	4	4	4	4	4	4	4	4	4	4	4
0.0001	4	4	4	4	4	4	4	4	4	4	4

Table 1: Number of iterations for problem (1) for $N_x = 101$

the speedups $S_d = t_d/t_{A2}$, and $S_{A1} = t_{A1}/t_{A2}$, with respect to the direct method and with respect to algorithm A1. Here t_d is the execution time for the direct method and t_{A1} for algorithm A1 on one processor, and t_{A2} for algorithm A2 on two processors of the Sequent Balance. It should be remarked that in all cases A2 is faster than A1. Note that the dominant effect here is the number of iterations required. In general, one does not expect a two-fold speedup (that is $S = 2$) for A2 over either the direct method or A1 since, due to the interfacial problem, A2 is not perfectly parallelizable. To make an approximate theoretical estimate of the speedup expected, recall that all the problems involved are solutions of tri-diagonal linear systems. Hence the cost is proportional to the number of grid points. On this basis *for a single iteration* $S \le 100/(50 + 2ninf)$, where *ninf* is the number of points in the interfacial region. This speedup is achieved for the overall time only if $K_{A2} \le K_s$, and this requires that there be sufficient points in the interfacial region. Thus an optimum strategy is to choose *ninf* as small as possible subject to this requirement. In the case of problem (1) here this would give an optimum $S = 1.81$. The remaining degradation seen in the tables can be attributed to the overhead for parallel directives and bus contention.

	$S_d = t_d/t_{A2}$					$S_{A1} = t_{A1}/t_{A2}$				
μ \ h	.1	.05	.01	.005	.001	.1	.05	.01	.005	.001
.1	0.71	0.55	0.22	0.15	0.05	1.23	1.59	2.04	3.33	2.94
.01	1.37	1.49	1.59	1.27	0.62	1.61	1.72	1.79	2.17	2.44
.001	1.11	1.23	1.49	1.54	1.56	1.41	1.52	1.72	1.75	1.56
.0001	0.87	0.98	1.28	1.37	1.49	1.20	1.32	1.54	1.61	1.72

Table 2: Speedups S_d and S_{A1} for problem (1) with $N_x = 101$

	K_s	K_{A1}					K_{A2}				
μ \ h		.1	.05	.01	.005	.001	.1	.05	.01	.005	.001
0.1	5	6	10	33	58	202	10	15	33	56	145
0.01	4	4	4	4	6	14	4	4	5	6	11
0.001	4	4	4	4	4	4	4	4	4	4	4
0.0001	4	4	4	4	4	4	4	4	4	4	4

Table 3: Number of iterations for problem (1) for $N_x = 501$

	$S_d = t_d/t_{A2}$					$S_{A1} = t_{A1}/t_{A2}$				
μ \ h	.1	.05	.01	.005	.001	.1	.05	.01	.005	.001
.1	0.77	0.54	0.20	0.15	0.06	1.06	1.23	1.49	1.96	2.63
.01	1.54	1.64	1.37	1.14	0.61	1.75	1.85	1.52	1.92	2.44
.001	1.35	1.49	1.67	1.67	1.67	1.61	1.72	1.85	1.85	1.85
.0001	1.06	1.23	1.54	1.59	1.67	1.39	1.52	1.75	1.82	1.85

Table 4: Speedups S_d and S_{A1} for problem (1) with $N_x = 501$

Example 2. We consider problem (2), where $\alpha = 1 + x$, $f(x,u) = 1 - e^{-u}$, $u_0 = 1$, $u_1 = 0$. We approximate problem (2) by a difference scheme on the special nonuniform grid from [3]. The subdomains Ω_1, Ω_2 and $\Omega_{\inf}$ are chosen in the same form as in Example 1. The numerical results are presented in Table 5 and 6. We should remark that the anomalous

	K_s	K_{A1}					K_{A2}				
ϵ \ h		.1	.05	.01	.005	.001	.1	.05	.01	.005	.001
0.1	3	7	12	46	81	295	11	17	36	43	51
0.01	3	3	3	7	12	43	4	4	6	7	8
0.001	3	3	3	3	3	6	3	3	3	3	4
0.0001	2	2	2	2	2	2	3	3	3	3	3
0.00001	2	2	2	2	2	2	2	2	2	2	2

Table 5: Number of iterations for problem (2) for $N_x = 101$

	$S_d = t_d/t_{A2}$					$S_{A1} = t_{A1}/t_{A2}$				
ϵ \ h	.1	.05	.01	.005	.001	.1	.05	.01	.005	.001
.1	0.43	0.31	0.16	0.13	0.11	1.06	1.23	2.08	3.33	10.00
.01	1.11	1.22	0.87	0.76	0.67	1.27	1.35	2.04	3.03	10.00
.001	1.39	1.52	1.56	1.56	1.15	1.61	1.72	1.72	1.72	2.50
.0001	0.97	1.05	1.08	1.08	1.08	1.11	1.16	1.19	1.19	1.19
.00001	1.35	1.52	1.54	1.54	1.54	1.64	1.61	1.64	1.75	1.75

Table 6: Speedups S_d and S_{A1} for problem (2) with $N_x = 101$

result in Table 6 for $\epsilon = .0001$ is due to the fact that $K_{A2} = 3$, whereas $K_s = K_{A1} = 2$. This occurred since, after 2 iterations, A2 had only reduced the error to $9.9e-5$. After 3 iterations it was $6.6e-9$. If we had chosen an error of $5.0e-6$ rather than $1.0e-5$ this anomaly would not have appeared.

References

[1] Boglaev,I.P., Sirotkin,V.V.: Domain decomposition technique for singularly perturbed problems and its parallel implementation. In: Proc. 13th IMACS World Congress on Computation and Applied Mathematics, Dublin, (J.J.H. Miller, R. Vichnevetsky, eds.), pp.522-523, 1991.

[2] Boglaev,I.P.: A numerical method for a quasilinear singular perturbation problem of elliptic type. USSR Comput. Maths. Math. Phys., 28, pp.492-502 (1988).

[3] Boglaev,I.P.: Numerical method for quasilinear parabolic equation with boundary layer, USSR Comput. Maths. Math. Phys., 30, pp.716-726, (1990).