Waveform Relaxation for Second Order Differential Equation $y'' = f(x, y)$

Kazufumi Ozawa and Susumu Yamada[1]

Graduate School of Information Science,
Tohoku University, Kawauchi Sendai 980-8576, JAPAN

Abstract. Waveform relaxation (WR) methods for second order equations $y'' = f(t, y)$, $y(t_0) = y_0$, $y'(t_0) = y'_0$ are studied. For linear case, the method converges superlinearly for any splittings of the coefficient matrix. For nonlinear case, the method converges quadratically only for waveform Newton method. It is shown, however, that the method with approximate Jacobian matrix converges superlinearly. The accuracy, execution times and speedup ratios of the WR methods on a parallel computer are discussed.

1 Introduction

In this paper we propose a waveform relaxation (WR) method for solving $y'' = f(x, y)$ on a parallel computer. The basic idea of this method is to solve a sequence of differential equations, which converges to the exact solution, with a starting solution $y^{[0]}(x)$. In this iteration, we can solve each component(block) of the equation in parallel.

2 Linear Differential Equation

Consider first the linear equation of the form

$$y''(x) = Qy(x) + g(x), \quad y(x_0) = y_0, \quad y'(x_0) = y'_0, \quad y \in \mathbf{R}^m, \tag{1}$$

where we assume $-Q$ is a positive definite matrix. The Picard iteration for solving (1) is given by

$$y^{[\nu+1]''}(x) = Qy^{[\nu]}(x) + g(x), \quad y^{[\nu]}(x_0) = y_0, \; y^{[\nu]'}(x_0) = y'_0. \tag{2}$$

Here we consider the rate of convergence of (2) and propose an acceleration of the iteration.

The solutions of (1) and (2) are given by

$$y(x) = y_0 + (x - x_0)y'_0 + \int_{x_0}^{x} \int_{x_0}^{s} \{Qy(\tau) + g(\tau)\} \mathrm{d}\tau \mathrm{d}s, \tag{3}$$

$$y^{[\nu+1]}(x) = y_0 + (x - x_0)y'_0 + \int_{x_0}^{x} \int_{x_0}^{s} \{Qy^{[\nu]}(\tau) + g(\tau)\} \mathrm{d}\tau \mathrm{d}s, \tag{4}$$

respectively. By subtracting (3) from (4) we can obtain

$$\varepsilon^{[\nu+1]}(x) = \int_{x_0}^{x} \int_{x_0}^{s} Q\varepsilon^{[\nu]}(\tau)\mathrm{d}\tau\mathrm{d}s, \tag{5}$$

where $\varepsilon^{[\nu]}(x) = y^{[\nu]}(x) - y(x)$. Taking the norm of the left-hand side and assuming $||\varepsilon^{[0]}(x)|| \leq K(x - x_0)$, we have by induction

$$||\varepsilon^{[\nu]}(x)|| \leq K\frac{c_1^{\nu}(x - x_0)^{2\nu+1}}{(2\nu+1)!}, \tag{6}$$

where we set $c_1 = ||Q||$. The inequality shows that Picard iteration (2) converges on any finite interval $x \in [x_0, T]$ and the rate of convergence is superlinear. It is, however, expected that the method converges slowly if the system is stiff or the length of the interval is large.

Here we propose an acceleration of the Picard iteration using the splitting $Q = N - M$. The iterative method to be considered is

$$y^{[\nu+1]''}(x) + My^{[\nu+1]}(x) = Ny^{[\nu]}(x) + g(x), \tag{7}$$

where we assume that matrix M is symmetric and positive definite. The error of the iterate $y^{[\nu]}(x)$ satisfies the differential equation

$$\varepsilon^{[\nu+1]''}(x) + M\varepsilon^{[\nu+1]}(x) = N\varepsilon^{[\nu]}(x). \tag{8}$$

In order to obtain the explicit expression for $\varepsilon^{[\nu]}(x)$, here we define the square root and the trigonometric functions of matrices.

Let λ_k $(k = 1, \ldots, m)$ be the eigenvalues of M, and P is a unitary matrix that diagonalizes M, i.e. $M = P\mathrm{diag}\,(\lambda_1, \ldots, \lambda_m)\, P^{-1}$, then using P the square root of M can be defined by

$$\sqrt{M} = P\mathrm{diag}\left(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_m}\right) P^{-1}.$$

For any square matrix H and scalar x, the trigonometric functions can be defined by

$$\cos(Hx) = \sum_{k=0}^{\infty} \frac{(-1)^k H^{2k}}{(2k)!} x^{2k}, \quad \sin(Hx) = \sum_{k=0}^{\infty} \frac{(-1)^k H^{2k+1}}{(2k+1)!} x^{2k+1}.$$

Using the functions defined above we have

$$\begin{aligned}
&\varepsilon^{[\nu+1]}(x) = \\
&(\sqrt{M})^{-1} \int_{x_0}^{x} \left(\cos(\sqrt{M}\tau)\sin(\sqrt{M}x) - \cos(\sqrt{M}x)\sin(\sqrt{M}\tau)\right) N\varepsilon^{[\nu]}(\tau)\mathrm{d}\tau \\
&\quad = (\sqrt{M})^{-1} \int_{x_0}^{x} \sin\left(\sqrt{M}(x - \tau)\right) N\varepsilon^{[\nu]}(\tau)\mathrm{d}\tau \\
&\quad = \int_{x_0}^{x} \int_{x_0}^{\tau} \cos\left(\sqrt{M}(x - \tau)\right) N\varepsilon^{[\nu]}(s)\mathrm{d}s\mathrm{d}\tau,
\end{aligned} \tag{9}$$

where the condition $\varepsilon^{[\nu+1]}(x)|_{x=x_0} = \frac{\mathrm{d}}{\mathrm{d}x}\varepsilon^{[\nu+1]}(x)|_{x=x_0} = 0$ is used. To bound $\varepsilon^{[\nu]}(x)$ if we use the Euclidean norm then we have

$$||\cos(\sqrt{M}x)|| \leq ||P|| \left|\left|\mathrm{diag}\left(\cos(\sqrt{\lambda_1}x), \ldots, \cos(\sqrt{\lambda_n}x)\right)\right|\right| ||P^{-1}|| \leq 1, \quad (10)$$

since $||P|| = ||P^{-1}|| = 1$. Assuming $||\varepsilon^{[0]}(x)|| \leq K(x - x_0)$ as before and letting $c_2 = ||N||$, we find by induction the inequality

$$||\varepsilon^{[\nu]}(x)|| \leq K\frac{{c_2}^{\nu}(x - x_0)^{2\nu+1}}{(2\nu+1)!}. \quad (11)$$

The result shows that although the rate of convergence of splitting method (7) remains superlinear, we can accelerate the convergence by making the value of $||N||$ small.

3 Nonlinear Differential Equation

Next we discuss the rate of convergence of the waveform relaxation for the second order nonlinear equation

$$y''(x) = f(x, y), \quad y(x_0) = y_0, \quad y'(x_0) = y_0', \quad y \in \mathbf{R}^m. \quad (12)$$

For the first order nonlinear equation $y' = f(x, y)$, the method called the waveform Newton method is proposed[5]. The method is given by

$$y^{[\nu+1]'}(x) - J_\nu y^{[\nu+1]}(x) = f(y^{[\nu]}(x)) - J_\nu y^{[\nu]}(x), \quad (13)$$

where J_ν is the Jacobian matrix of $f(y^{[\nu]})$. It is shown by Burrage[1] that the waveform Newton method converges quadratically on all finite intervals $x \in [x_0, T]$. In this section we first consider the convergence for the case that J_ν is replaced with an approximation in order to enhance the efficiency of the parallel computation.

Instead of (13) let us consider the iteration

$$y^{[\nu+1]'}(x) - \tilde{J}_\nu y^{[\nu+1]}(x) = f(y^{[\nu]}(x)) - \tilde{J}_\nu y^{[\nu]}(x), \quad (14)$$

where $\tilde{J}_\nu$ is an approximation to J_ν. Let $\varepsilon^{[\nu]} = y^{[\nu]} - y$ then we have

$$f(y^{[\nu]}) = f(y + \varepsilon^{[\nu]}) = f(y) + \frac{\partial f}{\partial y}\varepsilon^{[\nu]} + O(||\varepsilon^{[\nu]}||^2). \quad (15)$$

Substituting (15) into (14) and ignoring the term $O(||\varepsilon^{[\nu]}||^2)$, we have

$$\varepsilon^{[\nu+1]'} = \tilde{J}_\nu \varepsilon^{[\nu+1]} + \left(J_\nu - \tilde{J}_\nu\right)\varepsilon^{[\nu]}, \quad (16)$$

and taking the inner product we have

$$\begin{aligned}\left\langle \frac{\mathrm{d}}{\mathrm{d}x}\varepsilon^{[\nu+1]}, \varepsilon^{[\nu+1]} \right\rangle &= \left\langle \tilde{J}_\nu \varepsilon^{[\nu+1]}, \varepsilon^{[\nu+1]} \right\rangle + \left\langle (J_\nu - \tilde{J}_\nu)\varepsilon^{[\nu]}, \varepsilon^{[\nu+1]} \right\rangle \\ &\leq \mu_1 ||\varepsilon^{[\nu+1]}||^2 + \mu_2 ||\varepsilon^{[\nu+1]}||\ ||\varepsilon^{[\nu]}||, \end{aligned} \quad (17)$$

where the norm $||\cdot||$ is defined by $||u||^2 :=< u, u >$, and μ_1 and μ_2 are the subordinate matrix norms of $\tilde{J}_\nu$ and $J_\nu - \tilde{J}_\nu$, respectively. The left-hand side of (17) can be written as

$$\left\langle \frac{\mathrm{d}}{\mathrm{d}x}\varepsilon^{[\nu+1]}, \varepsilon^{[\nu+1]} \right\rangle = \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}x}||\varepsilon^{[\nu+1]}||^2 = ||\varepsilon^{[\nu+1]}||\frac{\mathrm{d}}{\mathrm{d}x}||\varepsilon^{[\nu+1]}||,$$

so that, under the assumption that $||\varepsilon^{[\nu+1]}|| \neq 0$, we have

$$\frac{\mathrm{d}}{\mathrm{d}x}||\varepsilon^{[\nu+1]}|| \leq \mu_1||\varepsilon^{[\nu+1]}|| + \mu_2||\varepsilon^{[\nu]}||. \tag{18}$$

Assuming $||\varepsilon^{[0]}(x)|| \leq K(x - x_0)$ as before and using $||\varepsilon^{[0]}(x_0)|| = 0$, we have

$$\begin{aligned} ||\varepsilon^{[\nu+1]}(x)|| &\leq \mu_2 e^{\mu_1(x-x_0)} \int_{x_0}^{x} e^{-\mu_1(s-x_0)}||\varepsilon^{[\nu]}(s)|| \, \mathrm{d}s \\ &\leq \mu_2 e^{\mu_1(x-x_0)} \int_{x_0}^{x} ||\varepsilon^{[\nu]}(s)|| \, \mathrm{d}s, \end{aligned} \tag{19}$$

which leads to

$$||\varepsilon^{[\nu]}(x)|| \leq K\frac{(\mu_2 e^{\mu_1(x-x_0)})^\nu (x - x_0)^{\nu+1}}{(\nu+1)!}, \tag{20}$$

showing that the rate of convergence of iteration (14) is not quadratic but superlinear.

By the way, the second order nonlinear equation

$$y''(x) = f(x, y), \quad y(x_0) = y_0, \quad y'(x_0) = y_0', \quad y \in \mathbf{R}^m \tag{21}$$

can be rewritten as the first order system

$$z'(x) = (y'(x), f(x, y))^T \equiv F(z(x)), \tag{22}$$

where $z(x) = (y(x), y'(x))^T$. The waveform Newton method for (22) is given by

$$z^{[\nu+1]'}(x) - J_\nu z^{[\nu+1]}(x) = F(z^{[\nu]}(x)) - J_\nu z^{[\nu]}(x). \tag{23}$$

In this case the Jacobian matrix J_ν is given by

$$J_\nu = \frac{\partial F}{\partial z} = \begin{pmatrix} \frac{\partial y'}{\partial y} & \frac{\partial y'}{\partial y'} \\ \frac{\partial f}{\partial y} & \frac{\partial f}{\partial y'} \end{pmatrix} = \begin{pmatrix} 0 & I \\ \frac{\partial f}{\partial y} & 0 \end{pmatrix}, \tag{24}$$

where I is an identity matrix. This result shows that for second order equation (12), if we define the waveform Newton method analogously by

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}y^{[\nu+1]}(x) - \frac{\partial f}{\partial y}y^{[\nu+1]}(x) = f(y^{[\nu]}(x)) - \frac{\partial f}{\partial y}y^{[\nu]}(x), \tag{25}$$

then the method also converges quadratically. Moreover, from the same discussion as for the first order equation, we can conclude that if an approximate Jacobian $\tilde{J}_\nu$ is applied, then the rate of convergence of the iteration

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}y^{[\nu+1]}(x) - \tilde{J}_\nu y^{[\nu+1]}(x) = f(y^{[\nu]}(x)) - \tilde{J}_\nu y^{[\nu]}(x) \tag{26}$$

is superlinear.

4 Numerical Experiments

4.1 Linear-Equations

In order to examine the efficiency of the waveform relaxations, we solve first the large linear system of second order differential equations. Consider the wave equation

$$\begin{cases} \dfrac{\partial^2 u}{\partial t^2} = \dfrac{\partial^2 u}{\partial x^2}, \quad 0 \le x \le 1, \\ u(x,0) = \sin(\pi x), \quad u(0,t) = u(1,t) = 0, \end{cases} \tag{27}$$

where the exact solution is $u(x,t) = \cos(\pi t)\sin(\pi x)$. The semi-discretization by 3-point spatial difference yields the system of linear equations

$$y''(t) = Qy(t), \quad y(t) = (u_1, \ldots, u_m)^T, \tag{28}$$

where

$$Q = \frac{1}{(\Delta x)^2} \begin{pmatrix} -2 & 1 & & & 0 \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ 0 & & & 1 & -2 \end{pmatrix} \in \mathbf{R}^{m \times m}, \quad \Delta x = \frac{1}{m+1}.$$

In the splitting method given here we take M as the block diagonal matrix given by $M = \mathrm{diag}(M_1, M_2, \ldots, M_\mu)$, and each block M_l is the tridiagonal matrix given by

$$M_l = \frac{1}{(\Delta x)^2} \begin{pmatrix} 2 & -1 & & & 0 \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ 0 & & & -1 & 2 \end{pmatrix} \in \mathbf{R}^{d \times d}, \; l = 1, 2, \ldots, \mu, \; \mu = m/d,$$

where we have assumed that m is divisible by d and, as a result, $\mu = m/d$ is an integer. In the implementation, the system is decoupled into μ subsystems and each of the subsystems is integrated concurrently on different processors, if the number of processors available are greater than that of the number of the subsystems. If this is not the case each processor integrates several subsystems sequentially. In any way our code is designed so as to have a uniform work load across the processors. The basic method to integrate the system is the 2-stage Runge-Kutta-Nyström method called the indirect collocation, which has an excellent stability property[3]. In our experiment we set $m = 256$ and integrate the system from $x = 0$ to 1 with the stepsize $h = 0.1$ under the stopping criteria $||y^{[\nu]} - y^{[\nu-1]}|| \le 10^{-7}$. The result on the parallel computer KSR1, which is a parallel computer with shared address space and distributed physical memory(see e.g. [6]), is shown in Table 1.

Table 1. Result for the linear problem on KSR1

d	iterations	CPU time(sec) serial	CPU time(sec) parallel	S_p	S'_p	E	E'	P	$\mu = m/d$
1	11579	566.02	61.92	9.14	2.58	0.57	0.16	16	256
2	6012	586.73	53.79	10.91	2.97	0.68	0.19	16	128
4	3052	362.94	31.44	11.54	5.08	0.72	0.32	16	64
8	1462	240.81	19.56	12.31	8.17	0.76	0.51	16	32
16	688	181.74	13.94	13.04	11.5	0.81	0.72	16	16
32	427	201.56	27.11	7.43	5.89	0.93	0.73	8	8
64	403	369.86	97.27	3.80	1.64	0.95	0.41	4	4
128	403	791.10	407.41	1.94	0.39	0.97	0.20	2	2
256	1	159.76	–	–	1.00	–	–	1	1

$$S_p = \frac{\text{CPU time(serial)}}{\text{CPU time(parallel)}}, \quad S'_p = \frac{\text{CPU time(serial, } d = 256)}{\text{CPU time(parallel)}},$$

$$E = \frac{S_p}{P}, \quad E' = \frac{S'_p}{P}, \quad P = \text{number of processors}$$

4.2 Nonlinear Equations

Next we consider the nonlinear wave equation given by

$$v_i'' = \exp(v_{i-1}(x)) - 2\exp(v_i(x)) + \exp(v_{i+1}(x)), \quad i = 0, \pm 1, \pm 2, \ldots, . \tag{29}$$

This equation describes the behaviour of the well-known Toda lattice and has the soliton solution given by

$$v_i(x) = \log(1 + \sinh^2 \tau \, \operatorname{sech}^2(i\tau - w(x+q))), \qquad i = 0, \pm 1, \pm 2, \ldots,$$

$$w = \sinh \tau,$$

where q is an arbitrary constant. The equation to be solved numerically is not (29) but its m-dimensional approximation given by

$$\begin{cases} y_i'' = \exp(y_{i-1}) - 2\exp(y_i) + \exp(y_{i+1}), & i = 2, \ldots, m-1 \\ y_1'' = 1 - 2\exp(y_1) + \exp(y_2), \\ y_m'' = \exp(y_{m-1}) - 2\exp(y_m) + 1 \end{cases} \tag{30}$$

$$\text{initial conditions}: \quad y_i'(0) = v_i'(0), \; y_i(0) = v_i(0),$$

where we have to choose a sufficiently large m to lessen the effect due to the finiteness of the boundary. As the parameters we take $q = 40$, $\tau = 0.5$ and $w = \sinh 0.5$. The approximate Jacobian $\tilde{J}_\nu$ used here is the block diagonal matrix given by

$$\tilde{J}_\nu(i,j) = \begin{cases} J_\nu(i,j), & d(l-1)+1 \le i,j \le dl, \; l = 1, \ldots, m/d \\ 0, & \text{otherwise}, \end{cases}$$

where we have assumed m is divisible by d as before.

In our numerical experiment we use the KSR1 parallel computer, and integrate the 1024-dimensional system from $x = 0$ to 5 with stepsize $h = 0.05$ under the stopping criteria $||y^{[\nu]} - y^{[\nu-1]}|| \leq 10^{-10}$. The result is shown in Table 2.

In our algorithms, if the value of d becomes small then the degree of paral-

Table 2. Results for the nonlinear problem on KSR1

d	iterations	CPU time(sec)		S_p	S'_p	E	E'	P	$\mu = m/d$
		serial	parallel						
1	17	53.98	6.10	8.85	3.49	0.55	0.22	16	1024
2	15	61.44	6.43	9.55	3.31	0.60	0.21	16	512
4	14	58.14	6.24	9.32	3.41	0.58	0.21	16	256
8	14	60.19	6.37	9.46	3.34	0.59	0.21	16	128
16	13	54.33	5.87	9.25	3.63	0.58	0.23	16	64
32	12	50.81	5.71	8.91	3.73	0.56	0.23	16	32
64	6	25.74	3.05	8.43	6.98	0.53	0.44	16	16
128	6	26.79	4.78	5.60	4.46	0.70	0.56	8	8
256	4	20.33	6.43	3.16	3.31	0.79	0.83	4	4
512	4	21.03	11.84	1.78	1.71	0.89	0.86	2	2
1024	4	21.30	—	—	1.00	–	–	1	1

$$S_p = \frac{\text{CPU time(serial)}}{\text{CPU time(parallel)}}, \quad S'_p = \frac{\text{CPU time(serial, } d = 1024)}{\text{CPU time(parallel)}}$$

P =number of processors

$$E = \frac{S_p}{P}, \quad E' = \frac{S'_p}{P}$$

lelism becomes large, since the number of the subsystems $\mu(= m/d)$ becomes large and, moreover, the cost of the LU decomposition that takes place in the inner iteration in each of the Runge-Kutta-Nyström schemes, which is proportional to d^3, becomes small. However, as is shown in the tables, it is in general true that the smaller the value of d, the larger the number of the WR iterations, which means the increase of the overheads such as synchronisation and communications. Therefore, small d implementation is not necessary advantageous. In our experiments we can achieve the best speedups when the number of subsystems is equal to that of processors.

References

1. Burrage, K., Parallel and Sequential Methods for Ordinary Differential Equations, Oxford University Press, Oxford, 1995.
2. Hairer, E., Nørsett S. P., and Wanner, G., Solving Ordinary Differential Equations I (Nonstiff Problems, 2nd Ed.), Springer-Verlag, Berlin, 1993.

3. van der Houwen, P. J., Sommeijer, B.P. Nguyen Huu Cong, Stability of collocation-based Runge-Kutta-Nyström methods, BIT 31(1991), 469-481.
4. Miekkala, U. and Nevanlinna, O., Convergence of dynamic iteration methods for initial value problems, SIAM J. Sci. Comput. 8(1987), pp.459-482.
5. White, J. and Sangiovanni-vincentelli, A.L., Relaxation Techniques for the Simulation of VLSI Circuits, Kluwer Academic Publishers, Boston, 1987.
6. Papadimitriou P., The KSR1 — A Numerical analyst's perspective, Numerical Analysis Report No.242, University of Mancheter/UMIST.