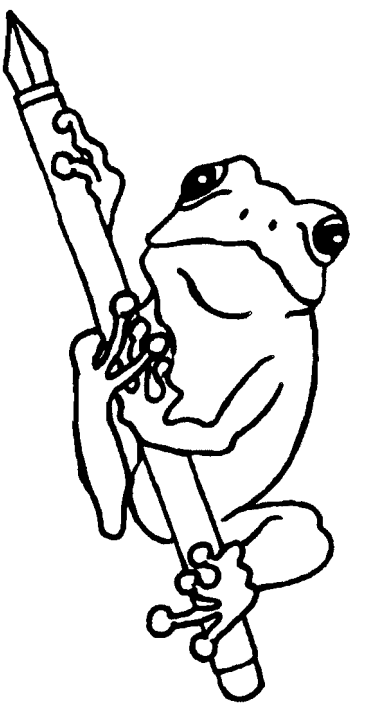
Stability and efficiency of waveform relaxation methods

EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computing Science

RANA 93–17
December 1993

Stability and Efficiency of Waveform Relaxation Methods

by
J.K.M. Jansen
R.M.M. Mattheij
M.T.M. Penders
W.H.A. Schilders



Reports on Applied and Numerical Analysis
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513
5600 MB Eindhoven
The Netherlands
ISSN: 0926-4507

Stability and Efficiency of Waveform Relaxation Methods

J.K.M. Jansen[1], R.M.M. Mattheij[1], M.T.M. Penders[1], W.H.A. Schilders[2]

[1] Department of Mathematics and Computing Science
Eindhoven University of Technology
[2] Applied Mathematics Group
Philips Research Laboratories

Abstract

We investigate the behaviour of Waveform Relaxation methods (WR) for some model problems. First it is shown how convergence (of the iteration) is related to stability of some one-step integration schemes. Then we investigate the computational complexity of a 1-D and 2-D heat equation when WR is used in combination with nested iteration and assess its efficiency, in particular compared to straightforward methods based on Gaussian elimination. Finally we present some results, showing the performance of WR.

November 24, 1993

1 Introduction

WR is a rather new iterative method for solving ordinary differential equations (ODEs). It was 1982 when WR was introduced by E. Lelarasmee in his Ph.D. dissertation on the time domain analysis of large scale nonlinear dynamical systems. In [2] the essential part of WR is explained to be the decomposition of large scale systems into smaller subsytems. The subsystems are solved separately on the whole time interval. The advantage of this approach is the possibility to use different discretizations and time stepping schemes for the various subsystems. The WR method was further investigated by White & Odeh in [7]. In this article the authors show that WR is a contraction mapping in a space with an exponentially scaled norm and converges uniformly with respect to all x in the spatial domain. In [4] Nevanlinna proves that WR even converges superlinearly on every finite interval [0,T]. He also notes that the iteration process may be ill-conditioned and correspondingly the actual algorithm numerically unstable. Furthermore he shows that it is quite well possible to predict from the computer data how the iteration has to stop.

An interesting feature of WR is the fact that it can be implemented on a parallel computer. Applications of this are extensively studied in [6]. An entirely different approach to WR can be found in [1]. Juang's analysis quantifies convergence properties of various iteration methods, and assesses the influence of equation ordering within the Gauss-Seidel procedure. Loosely speaking, the accuracy order of an approximation is one less than the number of matching terms in the Taylor expansions of the approximation and of the solution of the differential equation. Using the definition of accuracy, Juang shows in [1] the convergence of the WR method in an alternative way. Furthermore, it is investigated there how the ordering of the subsystems influences the order of accuracy.

Our aim is to look at WR used for solving PDEs. These PDEs may have dissipative or oscillatory behaviour. In §2 we introduce the WR method. Then, in §3, we analyse stability and convergence of WR applied to simple, but characteristic ODEs. In §4 we investigate efficiency of WR applied to the heat equation in 1-D and 2-D. Finally, in §5 we show some numerical examples sustaining the analysis.

2 WR Basics

To explain the basics of WR we consider the following autonomous ODE:

$$\begin{cases} \dot{x} = F(x) \\ x(0) = x_0 \end{cases}. \tag{1}$$

This ODE can be solved using an iterative method. To this end we reformulate (1) in such a way that the resulting equation induces a contraction mapping:

$$\dot{x} - G(x,y) = F(x) - G(x,y), \tag{2}$$

where $\mathbf{G}(\mathbf{x},\mathbf{y})$ is the *iteration function*. $\mathbf{G}(\mathbf{x},\mathbf{y})$ has to be chosen such, that $\mathbf{G}(\mathbf{z},\mathbf{z}) = \mathbf{F}(\mathbf{z})$. Denoting the k-th iterate by $\mathbf{x}^k$ this then leads to the iteration scheme:

$$\dot{\mathbf{x}}^{k+1} = \mathbf{G}(\mathbf{x}^{k+1}, \mathbf{x}^k). \tag{3}$$

In order to make this to practical procedure we now require G to satisfy the following, sometimes conflicting, conditions:

- the remaining ODE for x^{k+1} is easy to solve;
- the iteraton should converge rapidly.

For fast convergence it is necessary that G looks like F. One may e.g. take the so called *Newton WR* formula:

$$\mathbf{G}(\mathbf{x},\mathbf{y}) := \frac{\partial \mathbf{F}}{\partial \mathbf{x}} \cdot \mathbf{y}. \tag{4}$$

Other examples of choices for G, named after the equivalent iterative methods in matrix theory, are:

1. the *Jacobi WR* formula

$$\mathbf{G} := (g_1, \ldots, g_m)^T, \tag{5}$$

where

$$g_i(\mathbf{x}^{k+1}, \mathbf{x}^k) = f_i(x_1^k, \ldots, x_{i-1}^k, x_i^{k+1}, x_{i+1}^k, \ldots, x_m^k) \quad (i = 1, \ldots, m). \tag{6}$$

2. the *Gauss-Seidel WR* formula

$$\mathbf{G} := (g_1, \ldots, g_m)^T, \tag{7}$$

where

$$g_i(\mathbf{x}^{k+1}, \mathbf{x}^k) = f_i(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i^{k+1}, x_{i+1}^k, \ldots, x_m^k) \quad (i = 1, \ldots, m). \tag{8}$$

The WR method will be used with appropriate numerical integration schemes. An illustrative example of its use as a Picard-like iteration scheme based on analytical solutions can be found in [6], page 13:

Example 2.1
Consider the following ODE:

$$\dot{\mathbf{x}} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \mathbf{x}, \quad \mathbf{x}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{9}$$

with solution

$$\mathbf{x} = \begin{pmatrix} \sin(t) \\ \cos(t) \end{pmatrix}. \tag{10}$$

Applying Gauss-Seidel WR results in the following consecutive iterands:

- $\begin{cases} x_1^{(0)}(t) = 0 \\ x_2^{(0)}(t) = 1 \end{cases}$
- $\begin{cases} x_1^{(1)}(t) = t \\ x_2^{(1)}(t) = 1 - \frac{1}{2}t^2 \end{cases}$
- $\begin{cases} x_1^{(2)}(t) = t - \frac{1}{6}t^3 \\ x_2^{(2)}(t) = 1 - \frac{1}{2}t^2 + \frac{1}{24}t^4 \end{cases}$
- $\vdots$
- $\begin{cases} x_1^{(\nu)}(t) = \sum_{j=0}^{\nu-1} (-1)^j \frac{t^{2j+1}}{(2j+1)!} \\ x_2^{(\nu)}(t) = \sum_{j=0}^{\nu} (-1)^j \frac{t^{2j}}{(2j)!} \end{cases}$

One can easily see that the iteration picks up one additional term of the Taylor series every iteration sweep. In figure 1 we see this proces for the first terms of the Taylor series of $\sin(t)$. **Note**: It is interesting to mention that the numerical waveforms computed with Euler

Figure 1: First 8 terms of the Taylor series of $\sin(t)$

Forward are closely related to the analytical waveforms. The first two numerical waveforms of $\sin(t)$ are:

- $x^{(1)}(i \triangle t) = i \triangle t$
- $x^{(2)}(i \triangle t) = i \triangle t - (\frac{1}{6}i^3 - \frac{1}{2}i^2 + \frac{1}{3}i)(\triangle t)^3$

The analytical waveforms are respectively $x^{(1)}(t) = t$ and $x^{(2)}(t) = t - \frac{1}{6}t^3$.

It can be shown that this iteration converges to the solution:

$$x_1^{\infty}(t) = \sum_{j=0}^{\infty} (-1)^j \frac{t^{2j+1}}{(2j+1)!} = \sin(t), \tag{11}$$

and likewise for x_2. Now, it follows for the remainder:

$$S_\nu(t) := \left| \sum_{j=0}^{\nu-1} (-1)^j \frac{t^{2j+1}}{(2j+1)!} - \sin(t) \right| \leq \frac{t^{2\nu}}{(2\nu)!}, \qquad (t \geq 0). \tag{12}$$

From this example we can draw an important conclusion: the error is unbounded for larger values of t. This means that if we are computing approximations, too large values of t can

cause overflow. Moreover, computing approximations on an interval that is too large takes too much time. These considerations lead us to employ consecutive *windows*. A window is a time interval which should be chosen such that a relatively fast convergence is reached as compared to the whole interval on which we want to calculate the solution.

In order to get some insight into stability and convergence phenomena we shall consider the following homogeneous nth-order system of linear ordinary differential equations:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} \qquad \mathbf{x} = (x_1, \ldots, x_n)^T. \tag{13}$$

We shall assume that all eigenvalues of A have a negative real part. Applying Gauss-Seidel WR (according to (8)) results in:

$$\dot{x_1}^{k+1} = A_{11}x_1^{k+1} + A_{12}x_2^k + \ldots + A_{1n}x_n^k, \tag{14}$$

$$\vdots$$

$$\dot{x_n}^{k+1} = A_{n1}x_1^{k+1} + \ldots + A_{n,n-1}x_{n-1}^{k+1} + A_{nn}x_n^{k+1}, \tag{15}$$

where k is the iteration index.
Let

$$\mathbf{L} := \begin{pmatrix} A_{11} & 0 & \ldots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ A_{n1} & \ldots & \ldots & A_{nn} \end{pmatrix} \text{ and } \mathbf{B} := \begin{pmatrix} 0 & A_{12} & \ldots & A_{1n} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & A_{n-1,n} \\ 0 & \ldots & \ldots & 0 \end{pmatrix}, \tag{16}$$

$\mathbf{L}, \mathbf{B} \in \mathbb{R}^{n\times n}$, then we can write this system as:

$$\dot{\mathbf{x}}^{k+1} = \mathbf{L}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{x}^k. \tag{17}$$

We now like to know the solution at m specified points t_i, $1 \le i \le m$, say.
Let $\mathbf{x}_i := ((x_1)_i, \ldots, (x_n)_i)^T$ be the approximation of $\mathbf{x}(t_i)$ at t_i and define the global vector $\mathbf{y}^k$ by

$$\mathbf{y}^k := (\mathbf{x}_1^{k\,T} \ldots \mathbf{x}_m^{k\,T})^T. \tag{18}$$

We assume $\mathbf{x}_0^0$ to be known and hence we can take $\mathbf{x}_0^k = \mathbf{x}_0^0$ for $k > 0$.

We shall consider discretizations of Euler Forward and Euler Backward and analyse the stability and convergence of the resulting schemes. Applying Euler Forward to (17) results in:

$$\mathbf{x}_{i+1}^{k+1} = \mathbf{x}_i^{k+1} + \triangle t\mathbf{L}\mathbf{x}_i^{k+1} + \triangle t\mathbf{B}\mathbf{x}_i^k, \qquad 1 \le i \le m-1. \tag{19}$$

In terms of the global vector, (19) results in:

$$\mathcal{A}\mathbf{y}^{k+1} = \mathcal{B}\mathbf{y}^k + \mathbf{c}, \tag{20}$$

where $\mathbf{y}^0$ is known (see above),

$$\mathcal{A} := \begin{pmatrix} \mathbf{I}_n & \mathbf{O}_n & \ldots & \ldots & \mathbf{O}_n \\ -(\mathbf{I}_n + \triangle t\mathbf{L}) & \ddots & \ddots & & \vdots \\ \mathbf{O}_n & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \mathbf{O}_n \\ \mathbf{O}_n & \ldots & \ldots & -(\mathbf{I}_n + \triangle t\mathbf{L}) & \mathbf{I}_n \end{pmatrix}, \tag{21}$$

$$\mathcal{B} := \begin{pmatrix} \mathbf{O}_n & \mathbf{O}_n & \dots & \dots & \mathbf{O}_n \\ \triangle t\mathbf{B} & \mathbf{O}_n & \ddots & & \vdots \\ \mathbf{O}_n & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \mathbf{O}_n \\ \mathbf{O}_n & \dots & \dots & \triangle t\mathbf{B} & \mathbf{O}_n \end{pmatrix} \text{ and } \mathbf{c} := \begin{pmatrix} (\mathbf{I}_n + \triangle t\mathbf{A})\mathbf{x}_0^0 \\ \mathbf{0}_n \\ \vdots \\ \vdots \\ \mathbf{0}_n \end{pmatrix}, \tag{22}$$

$\mathcal{A}, \mathcal{B} \in \mathbb{R}^{nm \times nm}$ and $\mathbf{c} \in \mathbb{R}^{nm}$. $\mathbf{O}_n$ is the zero-matrix and $\mathbf{I}_n$ the identity in $\mathbb{R}^{nm \times nm}$. $\mathbf{0}_n$ is the zero vector in $\mathbb{R}^n$.
Simple calculation reveals that:

$$\mathcal{A}^{-1} = \begin{pmatrix} \mathbf{I}_n & \mathbf{O}_n & \dots & \dots & \mathbf{O}_n \\ \mathbf{I}_n + \triangle t\mathbf{L} & \mathbf{I}_n & \ddots & & \vdots \\ \vdots & \mathbf{I}_n + \triangle t\mathbf{L} & \ddots & \ddots & \vdots \\ (\mathbf{I}_n + \triangle t\mathbf{L})^{m-2} & \vdots & \ddots & \ddots & \mathbf{O}_n \\ (\mathbf{I}_n + \triangle t\mathbf{L})^{m-1} & (\mathbf{I}_n + \triangle t\mathbf{L})^{m-2} & \dots & \mathbf{I}_n + \triangle t\mathbf{L} & \mathbf{I}_n \end{pmatrix}. \tag{23}$$

From this it can be seen that

$$\mathcal{A}^{-1}\mathcal{B} = \begin{pmatrix} \mathbf{O}_n & \dots & \dots & \dots & \mathbf{O}_n \\ \triangle t\mathbf{B} & \ddots & & & \vdots \\ (\mathbf{I}_n + \triangle t\mathbf{L})(\triangle t\mathbf{B}) & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ (\mathbf{I}_n + \triangle t\mathbf{L})^{m-2}(\triangle t\mathbf{B}) & \dots & (\mathbf{I}_n + \triangle t\mathbf{L})(\triangle t\mathbf{B}) & \triangle t\mathbf{B} & \mathbf{O}_n \end{pmatrix}. \tag{24}$$

Apparently $\mathcal{A}^{-1}\mathcal{B}$ is lower triangular with only zeros on the diagonal. Hence all eigenvalues of $\mathcal{A}^{-1}\mathcal{B}$ are zero, i.e. $\rho(\mathcal{A}^{-1}\mathcal{B}) = max\{\,|\lambda| \mid \lambda \in \sigma(\mathcal{A}^{-1}\mathcal{B})\,\} = 0$. So convergence of the WR iteration is always guaranteed when Euler Forward is applied, i.e. there is *no* convergence restriction. Note, however, that convergence may be slow if $\mathcal{A}^{-1}\mathcal{B}$ is very skew. In order to assess the stability of this discretization we may simply consider the stability of the limiting case, i.e. $k \to \infty$. We then see that (19) results in

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \triangle t\mathbf{L}\mathbf{x}_i + \triangle t\mathbf{B}\mathbf{x}_i \tag{25}$$

$$\Leftrightarrow \mathbf{x}_{i+1} = (\mathbf{I} + \triangle t\mathbf{A})\mathbf{x}_i. \tag{26}$$

Hence, requiring stability is equivalent to requiring

$$\rho(\mathbf{I} + \triangle t\mathbf{A}) \leq 1. \tag{27}$$

This coincides with the 'normal' Euler Forward stability condition.

Next we turn to Euler Backward. Applying this discretization to (17) results in

$$\mathbf{x}_{i+1}^{k+1} = \mathbf{x}_i^{k+1} + \triangle t\mathbf{L}\mathbf{x}_{i+1}^{k+1} + \triangle t\mathbf{B}\mathbf{x}_{i+1}^{k}, \qquad (1 \leq i \leq m-1). \tag{28}$$

Again we see that if the iteration converges the solution is the same as obtained using Euler Backward in combination with the MOL. Rewriting (28) for global vectors gives the following iteration:

$$\mathcal{A}\mathbf{y}^{k+1} = \mathcal{B}\mathbf{y}^k + \mathbf{c}, \tag{29}$$

where, again, $\mathbf{y}^0$ is known and now

$$\mathcal{A} := \begin{pmatrix} \mathbf{I}_n - \triangle t\mathbf{L} & \mathbf{O}_n & \dots & \dots & \mathbf{O}_n \\ -\mathbf{I}_n & \mathbf{I}_n - \triangle t\mathbf{L} & \ddots & & \vdots \\ \mathbf{O}_n & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \mathbf{O}_n \\ \mathbf{O}_n & \dots & \mathbf{O}_n & -\mathbf{I}_n & \mathbf{I}_n - \triangle t\mathbf{L} \end{pmatrix}, \tag{30}$$

$$\mathcal{B} := \begin{pmatrix} \triangle t\mathbf{B} & \mathbf{O}_n & \dots & \dots & \mathbf{O}_n \\ \mathbf{O}_n & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \mathbf{O}_n \\ \mathbf{O}_n & \dots & \dots & \mathbf{O}_n & \triangle t\mathbf{B} \end{pmatrix} \text{ and } \mathbf{c} := \begin{pmatrix} \mathbf{x}_0^0 \\ \mathbf{0}_n \\ \vdots \\ \vdots \\ \mathbf{0}_n \end{pmatrix}, \tag{31}$$

where $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{nm \times nm}$ and $c \in \mathbb{R}^{nm}$.
From this it follows that:

$$\mathcal{A}^{-1} = \begin{pmatrix} (\mathbf{I}_n - \triangle t\mathbf{L})^{-1} & \mathbf{O}_n & \dots & \dots & \mathbf{O}_n \\ (\mathbf{I}_n - \triangle t\mathbf{L})^{-2} & (\mathbf{I}_n - \triangle t\mathbf{L})^{-1} & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbf{O}_n \\ (\mathbf{I}_n - \triangle t\mathbf{L})^{-m} & (\mathbf{I}_n - \triangle t\mathbf{L})^{-m+1} & \dots & \dots & (\mathbf{I}_n - \triangle t\mathbf{L})^{-1} \end{pmatrix}. \tag{32}$$

So we find that $\mathcal{A}^{-1}\mathcal{B}$ equals:

$$\begin{pmatrix} (\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B}) & \mathbf{O}_n & \dots & \dots & \mathbf{O}_n \\ (\mathbf{I}_n - \triangle t\mathbf{L})^{-2}(\triangle t\mathbf{B}) & (\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B}) & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbf{O}_n \\ (\mathbf{I}_n - \triangle t\mathbf{L})^{-m}(\triangle t\mathbf{B}) & (\mathbf{I}_n - \triangle t\mathbf{L})^{-m+1}(\triangle t\mathbf{B}) & \dots & \dots & (\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B}) \end{pmatrix}. \tag{33}$$

Thus convergence is easy to assess. We only have to require that the moduli of the eigenvalues of $(\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B})$ be less than one. Because of the block structure of $\mathcal{A}^{-1}\mathcal{B}$ we immediately find:

$$\rho((\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B})) \leq 1. \tag{34}$$

as necessary and sufficient to let (28) converge.

Similar to the Euler Forward discretization we can find out about stability by considering the case $k \to \infty$. From (28) we thus deduce:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \triangle t\mathbf{L}\mathbf{x}_{i+1} + \triangle t\mathbf{B}\mathbf{x}_{i+1} \tag{35}$$

$$\Leftrightarrow \mathbf{x}_{i+1} = (\mathbf{I} - \triangle t\mathbf{A})^{-1}\mathbf{x}_i. \tag{36}$$

Stability is apparently equivalent to requiring $\rho((\mathbf{I} - \triangle t\mathbf{A})^{-1}) \leq 1$. Because all eigenvalues of A have a negative real part, this condition is always satisfied and hence we have no stability restriction.

In [6] an analoguous algebraic proof is given for general multistep methods. An analytic proof is given by Miekkala and Nevanlinna in [3] using the Picard-Lindelöf iteration.

3 WR applied to second order ODE

As remarked before we like to get insight in stability and even more in convergence behaviour of WR for simple situations. In particular we like to include some oscillatory cases and hence need ODEs of order $n = 2$ at least. In this section we shall mainly survey the results for the general second order case (still with two eigenvalues with negative real part). The tedious calculations involved in this analysis are given in [5].
Consider the matrix:

$$\mathbf{A} := \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \tag{37}$$

Assume A has eigenvalues λ and μ, both with negative real part. Our aim is to derive conditions for convergence and stability of the WR iteration in terms of the time step $\triangle t$. One can show that for Euler Forward we have the following property:

Property 3.1
To guarantee a stable WR iteration for Euler Forward, we have:

$$\triangle t < \frac{2}{\max(|\lambda|, |\mu|)}, \qquad \text{if } \lambda, \mu \text{ are real,} \tag{38}$$

and:

$$\triangle t < \frac{-(\lambda + \mu)}{\lambda\mu}, \qquad \text{if } \lambda, \mu \text{ are complex.} \tag{39}$$

■

Note that we have no convergence condition because $\rho(\mathcal{A}^{-1}\mathcal{B}) = 0$.

Similarly one can show the following property for Euler Backward:

Property 3.2
Euler Backward is convergent in the following cases. If $\beta\gamma \geq 0$ then there is no condition. If $\beta\gamma < 0$, the condition depends on the sign of $\alpha\delta + \beta\gamma$, summarized in table 1. In this table we use $(\triangle t)_2 = \frac{(\alpha+\delta)-\sqrt{(\alpha-\delta)^2-4\beta\gamma}}{2(\alpha\delta+\beta\gamma)}$. ■

Although not treated here, it is obvious that the trapezoidal rule and indeed all θ-methods in between will exhibit similar convergence properties as Euler Backward. Since we shall

exclusively deal with dissipative problems we may trust to have convergence for those methods.

4 WR applied to the heat equation in 1-D and 2-D

From a complexity point of view there is hardly anything to be gained when applying WR to an ODE with full matrices. Actually, WR may be even slower if no particular solution properties are present. This situation may be different for sparse systems, in particular those arising from discretized partial differential equations (PDEs).

We shall investigate the situation for 1-D and 2-D heat equations where we apply Euler Backward (and in computational examples the trapezoidal as well which has discretisation errors that allow the time step to be of the order of the spatial step).

Consider the following 1-D linear initial-boundary value problem:

$$\begin{cases} \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \pi^2 sin(\pi x) & 0 < x < 1, t > 0, \\ u(x,0) = g(x) & 0 < x < 1, \\ u(0,t) = u(1,t) = 0 & t > 0. \end{cases} \tag{40}$$

The limiting stationary solution of this PDE is given by:

$$u(x,t) = \sin(\pi x). \tag{41}$$

In order to solve this PDE numerically we discretize by method of lines (MOL), i.e. first the spatial derivatives in the PDE are replaced by finite differences. We shall use a central difference scheme on an equispaced grid. This leads to the following ODE for the time-variable t (if x is *not* a boundary point):

$$\frac{du}{dt}(x,t) \doteq \frac{1}{h^2}(u(x-h,t) - 2u(x,t) + u(x+h,t)) + \pi^2 \sin(\pi x). \tag{42}$$

We discretize the time variable using Euler Backward. This results in:

$$u_r^{s+1} = u_r^s + \frac{\triangle t}{h^2}\left[u_{r-1}^{s+1} - 2u_r^{s+1} + u_{r+1}^{s+1}\right] + \pi^2 \triangle t \sin(\pi x), \tag{43}$$

where u_r^s is an approximation of $u(rh, s\triangle t)$. We then formulate a system of equations at every point:

$$\begin{pmatrix} 1+2\frac{\triangle t}{h^2} & -\frac{\triangle t}{h^2} & & \\ -\frac{\triangle t}{h^2} & \ddots & \ddots & \\ & \ddots & \ddots & -\frac{\triangle t}{h^2} \\ & & -\frac{\triangle t}{h^2} & 1+2\frac{\triangle t}{h^2} \end{pmatrix} \mathbf{u}^{s+1} = \mathbf{u}^s + \mathbf{c}, \tag{44}$$

which can be solved using Gaussian elimination. The computational complexity W_{GE} of this method follows from:

$$W_{GS} = C_{GE} \times T, \tag{45}$$

where C_{GE} is the computational complexity of Gaussian elimination:

$$C_{GE} = \mathcal{O}(n), \tag{46}$$

and T is the number of time points. We find:

$$W_{GE} = \mathcal{O}(n \times T). \tag{47}$$

We can now formulate the following:

Theorem 1 *There is no convergence condition for WR Gauss-Seidel when applied to (42).*

Proof
Let $\sigma := \frac{\Delta t}{\Delta x^2}$. The theory in §2 is immediately applicable, so the condition for convergence is given by (34):

$$\rho((\mathbf{I}_n - \Delta t\mathbf{L})^{-1}(\Delta t\mathbf{B})) \leq 1. \tag{48}$$

The matrix $\mathbf{I}_n - \Delta t\mathbf{L}$ can be written as:

$$\begin{pmatrix} 1+2\sigma & & & \\ -\sigma & \ddots & & \\ & \ddots & \ddots & \\ & & -\sigma & 1+2\sigma \end{pmatrix} \tag{49}$$

or

$$\mathbf{I}_n - \Delta t\mathbf{L} = (1+2\sigma)\begin{pmatrix} 1 & & & \\ -\frac{\sigma}{1+2\sigma} & \ddots & & \\ & \ddots & \ddots & \\ & & -\frac{\sigma}{1+2\sigma} & 1 \end{pmatrix} =: (1+2\sigma)(\mathbf{I}_n - \mathbf{N}). \tag{50}$$

$(\mathbf{I}_n - \Delta t\mathbf{L})^{-1}$ can now easily be calculated:

$$(\mathbf{I}_n - \Delta t\mathbf{L})^{-1} = \frac{1}{1+2\sigma}(\mathbf{I}_n - \mathbf{N})^{-1} = \frac{1}{1+2\sigma}\sum_{i=0}^{n-1}\mathbf{N}^i, \tag{51}$$

Hence:

$$\| (\mathbf{I}_n - \Delta t\mathbf{L})^{-1}\|_\infty \leq \frac{1}{1+2\sigma}\sum_{i=0}^{\infty}\left(\frac{\sigma}{1+2\sigma}\right)^i = \frac{1}{1+\sigma} \tag{52}$$

So:

$$\rho((\mathbf{I}_n - \Delta t\mathbf{L})^{-1}(\Delta t\mathbf{B})) \leq \| (\mathbf{I}_n - \Delta t\mathbf{L})^{-1}(\Delta t\mathbf{B})\|_\infty \leq \frac{\sigma}{1+\sigma} < 1. \tag{53}$$

For larger values of σ (in particular when $\triangle t = \mathcal{O}(h)$):

$$\rho((\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B})) \doteq 1 - \frac{1}{1+\sigma} \tag{54}$$

■

From (54) we see that for small values of σ, ρ differs from 1 significantly. This can be used to speed up the iterative proces. Let a grid M_l be defined by:

$$M_l := \{(x_i, t_j) | 0 \leq i \leq n \wedge 0 \leq j \leq T \wedge x_{i+1} - x_i = h_l \wedge t_{j+1} - t_j = \triangle t_l\}. \tag{55}$$

We introduce *nesting of equations* according to the following algorithm:

```
Nested iteration

1.i=0
2.repeat
     3.create initial approximation on grid M_i
     4.repeat
          5.solve the PDE on grid M_i using WR Gauss-Seidel
       until [convergence]
     6.project solution on grid M_{i+1}
     7.i=i+1
  until [desired grid is reached]
```

New grids can be chosen according to ones own needs. The convergence of the iterative process can be deduced as follows. Assume we have an initial guess at grid M_{l-1}. The maximum accuracy we can reach at this grid is $\mathcal{O}(h_{l-1}^2)$. If we have a solution with error $\mathcal{O}(h_{l-1}^2)$, the projection on grid M_l is used as the initial guess on M_l, which consequently contains an error $\mathcal{O}(h_{l-1}^2)$. Note that linear interpolation to prolongate M_{l-1} on M_l induces errors $\mathcal{O}(h_l^2)$; so the initial error on grid M_l is $\mathcal{O}(h_{l-1}^2)$. The error in the i-th iterand on grid M_l obeys:

$$\| \mathbf{e}_{l_i} \|_\infty \leq \rho^i \| \mathbf{e}_{l_0} \|_\infty. \tag{56}$$

On grid M_l we have $\| \mathbf{e}_{l_0} \|_\infty = \mathcal{O}(h_{l-1}^2)$. The iteration can stop if the maximum accuracy is obtained:

$$\| \mathbf{e}_{l_i} \|_\infty \leq c h_l^2 \Rightarrow \rho^{l_i} \leq \left(\frac{h_l}{h_{l-1}}\right)^2. \tag{57}$$

From this it follows that we need at most I_l iterations, where:

$$I_l := \frac{2\log\left(\frac{h_l}{h_{l-1}}\right)}{\log(\rho)} \tag{58}$$

iterations to reach maximum accuracy on grid M_l.

In the following we assume+ geometrically nested grids, i.e.

$$x_{i+\frac{1}{2}} = \frac{1}{2}(xi + x_{i+1}), t_{i+\frac{1}{2}} = \frac{1}{2}(t_i + t_{i+1}). \tag{59}$$

Because of this geometric nesting, the computational complexity W_{WR} at a certain window is determined by the complexity at the finest grid on that window, as the sum of an increasing geometrically series is approximately equal to the maximum term. Therefore we have:

$$W_{WR} = \mathcal{O}(I_{WR} \times n \times T). \tag{60}$$

The number of iterations I_{WR} follows from (58):

$$I_{WR} = \frac{-\log(4)}{\log(\frac{\sigma}{1+\sigma})} \tag{61}$$

$$\doteq (1+\sigma)\log(4). \tag{62}$$

We find:

$$W_{WR} = \mathcal{O}(\frac{n^2}{T} \times n \times T) = \mathcal{O}(n^3) \tag{63}$$

If we compare (47) and (63), we can only conclude that WR performs worse than Gaussian elimination in this 1-D case.

Now consider the 2-D analogue of (40):

$$\begin{cases} \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + 2\pi^2 sin(\pi x) sin(\pi y) & 0 < x < 1, t > 0, \\ u(x,y,0) = g(x,y) & 0 < x, y < 1, \\ u(0,y,t) = u(1,y,t) = 0 & 0 < y < 1, t > 0, \\ u(x,0,t) = u(x,1,t) = 0 & 0 < x < 1, t > 0. \end{cases} \tag{64}$$

The stationary solution of this PDE is given by:

$$u(x,y,t) = \sin(\pi x)\sin(\pi y). \tag{65}$$

As in the 1-D case we find:

$$\mathbf{I}_n - \triangle t\mathbf{L} = \begin{pmatrix} 1+4\sigma & & & & & \\ -\sigma & \ddots & & & & \\ \vdots & \ddots & \ddots & & & \\ -\sigma & & \ddots & \ddots & & \\ & \ddots & & \ddots & \ddots & \\ & & -\sigma & \dots & -\sigma & 1+4\sigma \end{pmatrix}, \tag{66}$$

or:

$$\rho((\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B})) \leq \| (\mathbf{I}_n - \triangle t\mathbf{L})^{-1}(\triangle t\mathbf{B})\|_\infty \leq \frac{2\sigma}{1+2\sigma} \tag{67}$$

It can be seen that the computational complexity for Gaussian eliminiation is much larger than in the 1-D case. For the 2-D case we have:

$$W_{GS} = C_{GS} \times T \tag{68}$$

where C_{GE} is the computational complexity for Gaussian elimination at a certain time point. In the 2-D case the matrix involved has also a banded structure, but now with bandwith n, where n is the number of points in one spatial direction. We therefore find:

$$C_{GE} = \mathcal{O}(n^2 \times n \times n) = \mathcal{O}(n^4). \tag{69}$$

So,

$$W_{GE} = \mathcal{O}(n^4 \times T). \tag{70}$$

The computational complexity for WR in the 2-D case can be derived like in the 1-D case. For the number of iterations needed to reach convergence on the finest grid, we find:

$$\begin{aligned} I_l &= \frac{-\log(4)}{\log(\rho)} && (71)\\ &= \frac{-\log(4)}{\log\left(\frac{2\sigma}{1+2\sigma}\right)} && (72)\\ &\doteq (1+2\sigma)\log(4) && (73) \end{aligned}$$

The computational complexity, identical to (60), equals:

$$\begin{aligned} W_{WR} &= \mathcal{O}(I_l \times n^2 \times T) && (74)\\ &= \mathcal{O}(\frac{n^2}{T} \times n^2 \times T) && (75)\\ &= \mathcal{O}(n^4) && (76) \end{aligned}$$

According to (68) and (76) we conclude that WR is more efficient than direct time stepping with Gaussian elimination in the 2-D case. Roughly speaking it is a factor n faster.

5 Numerical results

We have tested WR (Gauss-Seidel and Euler Backward), combined with nested iteration wherever necessary, for several problems. Our first example is a problem with a solution that is periodic in time.

Example 5.1
Consider the problem

$$\begin{cases} \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \sin(\pi x)\sin(\pi y)\cdot & \\ \qquad (2\pi^2\cos(2\pi\omega t) - 2\pi\omega\sin(2\pi\omega t)) & 0 < x < 1, t > 0,\\ u(x,y,0) = 0 & 0 < x, y < 1,\\ u(0,y,t) = u(1,y,t) = 0 & 0 < y < 1, t > 0,\\ u(x,0,t) = u(x,1,t) = 0 & 0 < x < 1, t > 0. \end{cases} \tag{77}$$

We start by taking $\omega = \omega(t) = 1$. One can see that for $t \gg 1$, the solution is given by

$$u(x,y,t) = \sin(\pi x)\sin(\pi y)\cos(2\pi t). \tag{78}$$

If we use (as an obvious choice) the time-windows $[0,1]$, $[1,2]$ and $[2,3]$ and set $h = \Delta t$, we obtain the following table (where $n = \frac{1}{h}$). Here the waveform on the first window is simply taken equal to 0 and on subsequent windows equal to the solution at the previous window. The column conv_{test} is the actually estimated contraction factor (through taking norms of iterates), whereas $\text{conv}_{theor.}$ denotes the theoretical contraction factor (cf. (67)). Note the slight overhead on the first window.

Example 5.2
Table (2) shows that the actual waveform is not exploited by this method. Apparently the interpolation on a coarse mesh, needed to start on the next window, causes the method to start almost from scratch. Hence we show in table (3), (4) and (5) the results on window 2 and 3 when started with the waveform of these windows for various finer grids. This then shows the great success of WR for periodic problems; Note however, that solutions need to be in a fairly periodic steady state form to have full advantage; see e.g. second and third window in (5).

Example 5.3
Our final example contains a problem which is nonperiodic (in time), but which nevertheless exhibits a sort of waveform (at least is periodic asymptotically). Using $\omega = \omega(t) = \frac{t}{1+t}$, its solution reads:

$$u(x,y,t) = \sin(\pi x)\sin(\pi y)\cos(2\pi\omega t), \tag{79}$$

where

$$\omega = \omega(t) = \frac{t}{1+t}. \tag{80}$$

Using the experience of the previous example we now employ nested iteration on the first window ($[0,1]$) and a fairly fine mesh on subsequent windows ($[1,2]$ and $[2,3]$). Again the approximation on $[0,1]$ was taken to be equal to zero, but equal to the values on the previous window for subsequent intervals, see table (6) and (7).

We conclude that nested iteration is an excellent idea for efficiently computing solutions on a window. Once a waveform is found we can proceed with the solution on the finer grids as an initial value for subsequent windows.

References

[1] Gear, C.W. & Juang, F.
The speed of waveform methods for ODEs
In *Applied and Industrial Mathematics*, pp. 37-48
Kluwer. Boston, 1991

[2] Lelarasmee, E., Ruehli, A.E. & Sangiovanni-Vincentelli, A.L.
The Waveform Relaxation Method for Time-Domain Analysis of Large Scale IC
In *IEEE Transactions on computer aided design of integrated circuits and systems*, vol. CAD-1, no. 3, pp. 131-145
IEEE, 1982

[3] Miekkala, U. & Nevanlinna, O.
Convergence of dynamic iteration methods for initial value problems
In *SIAM J. Sci. Stat. Comput.*, vol. 8, no. 4, pp. 459-482
Society for Industrial and Applied Mathematics, 1987

[4] Nevanlinna, O.
Remarks on Picard-Lindelöf iteration
BIT 29, pp. 328-346. 1989

[5] Penders, M.T.M
Waveform Relaxation
Master's thesis, Eindhoven University of Technology
Eindhoven, 1994 (to appear)

[6] Vandewalle, S.
The parallel solution of parabolic partial differential equations by multigrid waveform relaxation methods
PhD, Katholieke Universiteit Leuven
Leuven, 1992

[7] White, J., Vincentelli, A.S., Odeh, F. & Ruehli, A.
Waveform Relaxation: Theory and Practice
In *Transactions of the society for computer simulations*, vol. 2, no. 1, pp. 95-133
The Society of Computer Simulation, 1985.

	$\alpha\delta + \beta\gamma \geq 0$	$\alpha\delta + \beta\gamma < 0$
$\alpha < 0,\ \delta < 0$	no condition	$\Delta t < (\Delta t)_2$
$\alpha < 0,\ \delta > 0$	not relevant	$\Delta t < (\Delta t)_2$
$\alpha > 0,\ \delta < 0$	not relevant	$\Delta t < (\Delta t)_2$

Table 1: Convergence conditions Euler Backward

window	n	conv_{test}	$\text{conv}_{theor.}$	iters_{test}	$\text{iters}_{theor.}$
1	2	1.0000	0.8000	2	3
	4	0.8637	0.8889	2	6
	8	0.9493	0.9412	11	11
	16	0.9785	0.9697	32	23
	32	0.9898	0.9846	80	45
2	2	1.0000	0.8000	2	3
	4	0.8458	0.8889	2	6
	8	0.8802	0.9412	3	11
	16	0.9665	0.9697	21	23
	32	0.9897	0.9846	76	45
3	2	1.0000	0.8000	2	3
	4	0.8525	0.8889	2	6
	8	0.8808	0.9412	3	11
	16	0.9669	0.9697	21	23
	32	0.9895	0.9846	76	45

Table 2: $\omega(t) = 1$

window	n	conv_{test}	$\text{conv}_{theor.}$	iters_{test}	$\text{iters}_{theor.}$
1	2	1.0000	0.8000	2	3
	4	0.8637	0.8889	2	6
	8	0.9493	0.9412	11	11
	16	0.9785	0.9697	32	23
	32	0.9898	0.9846	80	45
2	8	0.9144	0.9412	2	11
	16	0.9665	0.9697	8	23
	32	0.9898	0.9846	74	45
3	8	1.0446	0.9412	2	11
	16	0.9648	0.9697	2	23
	32	0.9898	0.9846	58	45

Table 3: $\omega(t) = 1$

window	n	conv_{test}	$\text{conv}_{theor.}$	iters_{test}	$\text{iters}_{theor.}$
1	2	1.0000	0.8000	2	3
	4	0.8637	0.8889	2	6
	8	0.9493	0.9412	11	11
	16	0.9785	0.9697	32	23
	32	0.9898	0.9846	80	45
2	16	0.9667	0.9697	21	23
	32	0.9901	0.9846	72	45
3	16	0.9782	0.9697	2	23
	32	0.9892	0.9846	8	45

Table 4: $\omega(t) = 1$

window	n	conv_{test}	$\text{conv}_{theor.}$	iters_{test}	$\text{iters}_{theor.}$
1	2	1.0000	0.8000	2	3
	4	0.8637	0.8889	2	6
	8	0.9493	0.9412	11	11
	16	0.9785	0.9697	32	23
	32	0.9898	0.9846	80	45
2	32	0.9902	0.9846	169	45
3	32	0.9898	0.9846	2	45

Table 5: $\omega(t) = 1$

window	n	conv_{test}	$\text{conv}_{theor.}$	iters_{test}	$\text{iters}_{theor.}$
1	2	1.0000	0.8000	2	3
	4	0.6490	0.8889	2	6
	8	0.9024	0.9412	7	11
	16	0.9708	0.9697	24	23
	32	0.9900	0.9846	78	45
2	8	0.8653	0.9412	12	11
	16	0.9635	0.9697	16	23
	32	0.9901	0.9846	70	45
3	8	0.8699	0.9412	8	11
	16	0.9642	0.9697	18	23
	32	0.9900	0.9846	74	45

Table 6: $\omega(t) = \frac{t}{1+t}$

window	n	conv_{test}	$\text{conv}_{theor.}$	iters_{test}	$\text{iters}_{theor.}$
1	2	1.0000	0.8000	2	3
	4	0.6490	0.8889	2	6
	8	0.9024	0.9412	7	11
	16	0.9708	0.9697	24	23
	32	0.9900	0.9846	78	45
2	16	0.9632	0.9697	64	23
	32	0.9900	0.9846	69	45
3	16	0.9625	0.9697	47	23
	32	0.9900	0.9846	70	45

Table 7: $\omega(t) = \frac{t}{1+t}$