

Consistency & Numerical Smoothing \Rightarrow Error Estimation

— *An Alternative of the Lax-Richtmyer Theorem*

Tong Sun

Department of Mathematics and Statistics
Bowling Green State University, Bowling Green, OH 43403
tsun@bgsu.edu

Abstract

We all know Lax-Richtmyer Theorem: Assuming consistency, convergence is equivalent to numerical stability. However, it is in practice very difficult to verify the numerical stability of a scheme while solving an evolution equation, especially if the equation is nonlinear and/or the scheme is complex. Consequently, a large gap exists between error analysis theory and numerical computation practice. We prove that one can use numerical smoothing to replace numerical stability in error estimation. As a property of the computed numerical solution, the numerical smoothing property can be monitored by a “smoothing indicator”. This approach has several advantages, including (1) it works for nonlinear equations and systems; (2) it works for any time stepping scheme, as long as the smoothing indicator remains bounded; (3) it allows error propagation to be analyzed with differential equations independently of numerical schemes, which makes long time error estimation possible. Hence, we can narrow the gap between theory and practice significantly. In this paper, we focus on the concept of numerical smoothing.

1 Introduction

In the past a few years, the author and his colleagues have published three papers [5,6,7] on long time error estimates of ODEs and parabolic PDEs, where the approach of numerical smoothing indicator was used. This paper is to summarize the approach in a fundamental error analysis theorem, namely, consistency & numerical smoothing \Rightarrow error estimation. This will be done in a general and intuitive way, in the next two sections.

In order to use the new theorem, in section 4, the concepts of numerical smoothing and smoothing indicator will be studied by using a linear model problem. For the first time, the boundedness of a smoothing indicator is proven for a model problem. Several remarks will be given to help readers understand the concepts from different aspects.

In section 5, an adaptive time stepping algorithm is presented based on the smoothing indicator. In the proof of the error estimate, one can see how the general theorem is applied in a concrete problem. The last section is devoted to discussing the advantages of the new approach and future research topics.

2 A brief review of numerical stability

Numerical stability is about how a numerical scheme propagates error. Figure 1 gives an explanation. $u(t)$ stands for a solution of an evolution equation. $u_N(t)$ stands for a numerical solution

obtained by using a numerical scheme. The error at t_{n+1} is split into local error and propagation error by $\check{u}_N(t_{n+1})$, where $\check{u}_N(t_{n+1})$ is assumed to be the non-computable numerical solution which would be obtained by using the same scheme, in the step $[t_n, t_{n+1}]$, with initial value $u(t_n)$.

$$u(t_{n+1}) - u_N(t_{n+1}) = u(t_{n+1}) - \check{u}_N(t_{n+1}) + \check{u}_N(t_{n+1}) - u_N(t_{n+1})$$

In most textbooks and papers, one can find that the local error is defined to be $u(t_{n+1}) - \check{u}_N(t_{n+1})$. A typical way of defining local error is to put a real solution in a scheme to find the residual, which is equivalent to looking at $u(t_{n+1}) - \check{u}_N(t_{n+1})$. One seems have to use $u(t_{n+1}) - \check{u}_N(t_{n+1})$ for local error in *a priori* error estimates, because the smoothness of $u(t)$ can be applied here.

As a consequence of the local error definition, one has to face the second part of the split error, namely, $\check{u}_N(t_{n+1}) - u_N(t_{n+1})$, which is obviously the error $u(t_n) - u_N(t_n)$ propagated by the scheme. Usually, one tries to prove that, in an appropriate norm,

$$\|\check{u}_N(t_{n+1}) - u_N(t_{n+1})\| \leq (1 + \beta\tau_n)\|u(t_n) - u_N(t_n)\|,$$

where $\tau_n = t_{n+1} - t_n$ is the stepsize. It would be ideal to have $\beta < 0$, with which one can prove uniform convergence in time. If $\beta = 0$, there will be a simple accumulation of local error into global error. Unfortunately, away from a linear model problem of an ODE or parabolic PDE, one often ends up getting a positive β . Both nonlinearity of the evolution equation and complexity of the numerical scheme will contribute to make β larger. In the global error estimate, one will see a factor $e^{\beta T}$, if the numerical solution is to be computed in $[t_0, t_0 + T]$.

As a convergence result, such an error estimate is alright. However, such an estimate is practically usable only for a very small T , a small fraction of the actually needed T . Even for some evolution equations, where the solution has some exponential contraction properties, or lives in some contractive objects, one usually cannot prove that the numerical solution converges uniformly to the “attractor”. The reason is that, in many cases, a nonlinear problem makes β positive. In addition, if a complex numerical scheme is used, it is extremely hard to reflect the contraction property of the original solution in the stability analysis. Moreover, any minor modification of a numerical scheme can force a numerical analyst to prove numerical stability of the new scheme from scratch.

Lax-Richtmyer Theorem states that, assuming a scheme is consistent with the equation, it is necessary and sufficient to have numerical stability, in order for the numerical solution to converge to the real solution. Because numerical stability is too difficult to prove, and it is necessary, many problems are solved without proper error estimates. This leaves a huge gap between error analysis theory and scientific computation practice.

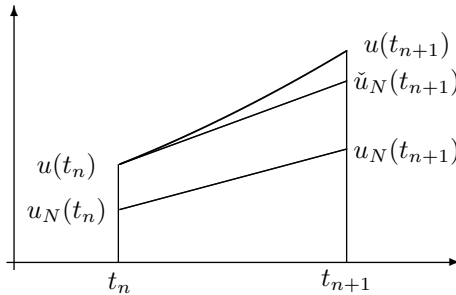


Figure 1. Numerical Stability

2

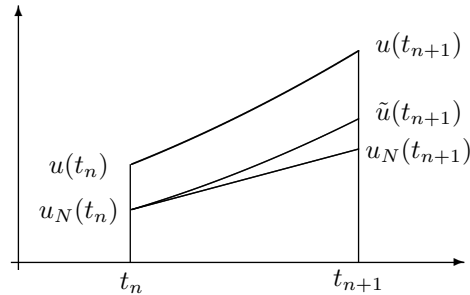


Figure 2. Numerical Smoothing

3 Consistency & Smoothing \Rightarrow Convergence

In Figure 2, the error is split in a different way. $u(t)$ and $u_N(t)$ are defined as before. For the entire time domain, $\tilde{u}(t)$ is defined piecewise. In each interval $(t_n, t_{n+1}]$, $\tilde{u}(t)$ is the solution of the evolution equation with initial value $u_N(t_n)$. Here the total error at t_{n+1} is split as

$$u(t_{n+1}) - u_N(t_{n+1}) = u(t_{n+1}) - \tilde{u}(t_{n+1}) + \tilde{u}(t_{n+1}) - u_N(t_{n+1}).$$

The first difference on the right hand side is the error $u(t_n) - u_N(t_n)$ propagated by the evolution equation. The second difference is the actual local error, which is the error of $u_N(t_{n+1})$ while approximating $\tilde{u}(t_{n+1})$.

At the first glance, one may find that the splitting in Figure 2 has two problems. Firstly, there is no position for numerical stability, since error propagation is given by the evolution equation. Secondly, the actual local error depends on the regularity of $\tilde{u}(t)$, which cannot be assumed, since $\tilde{u}(t)$ is not the original solution. These two problems may have prevented many people from further study of this error splitting.

In fact, any *a posteriori* local error estimates have to be based on actual local error. However, it has not been widely noticed that, if one has a reliable *a priori* or *a posteriori* actual local error estimate, one can use the evolution equation to estimate error propagation, there is no need to consider numerical stability, as shown in Figure 2.

This may seem unbelievable, but it is true. Numerical stability is necessary. However, proving numerical stability turns out to be not necessary. In fact, the assumption “if one has a reliable *a posteriori* actual local error estimate” is not a valid assumption, unless one can prove it. The missing piece of the puzzle is exactly **numerical smoothing**.

Let's follow Figure 2 to see how it works. The following plain arguments sketch the proof of a new theorem. $u(t_{n+1}) - \tilde{u}(t_{n+1})$ depends on the error propagation behavior of the evolution equation. For the actual local error $\tilde{u}(t_{n+1}) - u_N(t_{n+1})$ to be of order τ^q of the timestep size τ , it is sufficient if $\tilde{u}(t)$ has bounded q -th order time derivative in $[t_n, t_{n+1}]$. Consequently, one just needs to establish some kind of smoothness of $u_N(t_n)$, which is the initial value $\tilde{u}(t)$ in $(t_n, t_{n+1}]$. **To this end, it is necessary and sufficient that the scheme used to obtain $u_N(t_n)$ has certain smoothing effect.** Summarizing the arguments, one gets

Theorem. Consistency & numerical smoothing \Rightarrow convergence.

The error splitting in Figure 2 is less popularly used compared to that in Figure 1, which is in almost every textbook. Nevertheless, it was well known. For example, in [9, p.160], both splitting were given, then the error analysis in Theorem 3.4 was proven by using the error splitting in Figure 2 here. However, the assumptions on the actual local error and the boundedness of the partial derivative of the right hand side of the differential equations do not apply to stiff ODE systems or parabolic PDEs. The missing piece of the puzzle, numerical smoothing, was not identified in [9], or other publications.

In the next two sections, the reader will see how to establish the numerical smoothing property of a scheme and use it for error estimation. For more details, see [5,6,7].

4 Smoothing indicator

One question may arise at this point: Is it easier or harder to study numerical smoothing compared to numerical stability? The answer to the question is based on the simple fact that numerical smoothing is a property of the **computed** numerical solution. It is hard to compute an indicator for the numerical stability property of a scheme, because the auxiliary numerical solution $\check{u}_N(t)$ in Figure 1 is not computable. However, it is easy to compute an indicator on the numerical smoothing property of a computed numerical solution. In a linear model problem, one can also prove the boundedness of a smoothing indicator. The following model problem is employed to illustrate the procedure.

Let Ω be an L-shaped domain. Consider the linear equation

$$u_t = \Delta u + f \quad (1)$$

with both boundary condition and initial condition written as $u = 0$. For simplicity, let $f \in L^2(\Omega)$ be constant in time. There is no essential difficulty to extend the results to a time-dependent f . It is well known that, at the steady state, u is usually not even in $H^2(\Omega)$.

Let $S_h \subset H_0^1(\Omega)$ be a conforming finite element space. Let Δ_h be the discrete Laplacian operator and P_h be the L^2 projection to S_h . Then the semi-discrete solution $u_h \in S_h$ is the solution of the ODE

$$u_{h,t} = \Delta_h u_h + P_h f.$$

Let $v_h = u_{h,t} \in S_h$, $w_h = v_{h,t} \in S_h$, and $z_h = w_{h,t} \in S_h$, they satisfy

$$v_{h,t} = \Delta_h v_h$$

with initial condition $v_h(t_0) = u_{h,t}(t_0) = \Delta_h u_h(t_0) + P_h f = P_h f$, because $u_h(t_0) = 0$,

$$w_{h,t} = \Delta_h w_h$$

with initial condition $w_h(t_0) = v_{h,t}(t_0) = \Delta_h v_h(t_0) = \Delta_h P_h f$, and

$$z_{h,t} = \Delta_h z_h$$

with initial condition $z_h(t_0) = w_{h,t}(t_0) = \Delta_h w_h(t_0) = \Delta_h \Delta_h P_h f$, respectively. Obviously, since $f \in L^2(\Omega)$, both w_h and z_h have a transient process due to initial singularities in the domain Ω and near the boundary $\partial\Omega$, then they become bounded as t increases. In this sense, the semi-discrete scheme is a smoothing scheme. The smoothing property of the original solution is well known.

As before, let $u_N(t)$ stand for a fully discrete numerical solution. It is reasonable to desire that the numerical solution share the smoothing property of the continuous and the semi-discrete solutions. Therefore, a smoothing indicator should be defined as follows:

Definition. At a time node t_n , the smoothing indicator is defined as either

$$S_n^{(2)} = (\bar{u}_n, \bar{v}_n, \bar{w}_n)$$

or

$$S_n^{(3)} = (\bar{u}_n, \bar{v}_n, \bar{w}_n, \bar{z}_n)$$

where $\bar{u}_n = u_N(t_n)$, $\bar{v}_n = \Delta_h \bar{u}_n + P_h f$, $\bar{w}_n = \Delta_h \bar{v}_n$, and $\bar{z}_n = \Delta_h \bar{z}_n$. Moreover,

(a) if there is a constant M , such that $\|S_n^{(q)}\| \leq M(1 + \|S_0^{(q)}\|)$ for all n , then the scheme is called a smoothness keeping scheme;

(b) if there is a constant M , such that, for any given initial value, there is a constant s , such that $\|S_n^{(q)}\| \leq M$ for all $t_n > s$, then the scheme is called a smoothing scheme. \square

The next lemma confirms that the implicit Euler scheme is a smoothing scheme.

Lemma 1. If the semi-discrete problem is discretized by the implicit Euler scheme with any variable timestep sizes, the fully implicit scheme is smoothing.

Proof: Since the implicit Euler scheme is locally of order 2, it is appropriate to consider $S_n^{(2)} = (\bar{u}_n, \bar{v}_n, \bar{w}_n)$. Let the implicit Euler scheme be written as

$$\frac{u_N(t_{n+1}) - u_N(t_n)}{\tau_n} = \Delta_h u_N(t_{n+1}) + P_h f.$$

According to the definition of \bar{v}_n and \bar{w}_n ,

$$\bar{v}_n = \Delta_h u_N(t_n) + P_h f, \quad \bar{w}_n = \Delta_h \bar{v}_n,$$

hence

$$\begin{aligned} \frac{\bar{v}_{n+1} - \bar{v}_n}{\tau_n} &= \Delta_h \frac{u_N(t_{n+1}) - u_N(t_n)}{\tau_n} = \Delta_h (\Delta_h u_N(t_{n+1}) + P_h f) = \Delta_h \bar{v}_{n+1}, \\ \frac{\bar{w}_{n+1} - \bar{w}_n}{\tau_n} &= \Delta_h \frac{\bar{v}_{n+1} - \bar{v}_n}{\tau_n} = \Delta_h \Delta_h \bar{v}_{n+1} = \Delta_h \bar{w}_{n+1}. \end{aligned}$$

Repeatedly using these formulas, one can get

$$\begin{aligned} \bar{v}_{n+1} &= \prod_{k=0}^n (I - \tau_k \Delta_h)^{-1} \bar{v}_0 = \prod_{k=0}^n (I - \tau_k \Delta_h)^{-1} P_h f, \\ \bar{w}_{n+1} &= \prod_{k=0}^n (I - \tau_k \Delta_h)^{-1} \bar{w}_0 = \prod_{k=0}^n (I - \tau_k \Delta_h)^{-1} \Delta_h P_h f. \end{aligned}$$

Since Δ_h is negative definite, $\|(I - \tau_k \Delta_h)^{-1}\| < 1$. Consequently $\|\bar{v}_{n+1}\| < \|P_h f\|$ remains bounded.

As for \bar{w}_{n+1} , first notice the following two inequalities:

$$\|(I - \beta \Delta_h)^{-1} \Delta_h\| \leq 1/\beta,$$

$$\|(I - \alpha \Delta_h)^{-1} (I - \beta \Delta_h)^{-1} \Delta_h\| \leq \|[I - (\alpha + \beta) \Delta_h]^{-1} \Delta_h\|$$

for any α and β . Now,

$$\|\bar{w}_{n+1}\| = \left\| \prod_{k=0}^n (I - \tau_k \Delta_h)^{-1} \Delta_h \right\| \|P_h f\| \leq \|(I - (t_{n+1} - t_0) \Delta_h)^{-1} \Delta_h\| \|P_h f\| \leq \frac{\|P_h f\|}{t_{n+1} - t_0}.$$

Thus it is proven that the scheme is smoothing. \square

Lemma 2. If the semi-discrete problem is discretized by the Crank-Nicolson scheme with any variable timestep sizes, the fully implicit scheme is a smoothness keeping scheme.

Proof: Since the Crank-Nicolson scheme is locally of order 3, it is natural to consider $S_n^{(3)} = (\bar{u}_n, \bar{v}_n, \bar{w}_n, \bar{z}_n)$. The proof is similar to that of Lemma 1. \square

Remark 1. In general, it is only desired to keep the smoothing indicator bounded. For the purpose of error control in L^2 norm on $u_h - u_N$, one does not need to prove that \bar{v}_n , \bar{w}_n and \bar{z}_n approximate $u_{h,t}$, $u_{h,tt}$ and $u_{h,ttt}$, respectively.

Remark 2. For the Crank-Nicolson scheme, one can show that, if $\tau = \gamma h$ where γ is a positive constant and h is the meshsize of a quasi-uniform mesh, the Crank-Nicolson scheme does not have sufficient smoothing to smooth out an initial singularity showing in $\bar{w}_0 = \Delta_h P_h f$ and $\bar{z}_0 = \Delta_h \Delta_h P_h f$. For smoothing, γ has to depend on h . However, Lemma 2 confirms that it is alright to use the Crank-Nicolson scheme with constant γ when a transient process has finished.

Remark 3. It is easy to verify that the explicit Euler scheme is not a smoothness keeping scheme, unless the familiar stability condition $\tau < Ch^2$ is satisfied, for a proper constant C . Furthermore, it takes $\tau < Ch^2/2$ to make the explicit Euler scheme having proper smoothing.

Remark 4. For more complex equations and schemes, it is not always possible to prove if the scheme is smoothing or not. However, a smoothing indicator is always defined as a function of the computed numerical solution, so it is always computable.

Remark 5. A smoothing indicator has two functions. First, it can be used to monitor if a scheme is anti-smoothing or lack of smoothing. In addition, one can use the smoothing indicator to adaptively determine the next timestep size. An interesting fact is that, by adaptively determining timestep sizes toward error control, one can safely use a conditionally smoothing or smoothness-keeping scheme.

Remark 6. The smoothing effect only applies to the time derivatives of the solutions. For example, $u_{h,tt} = v_{h,t} = w_h = \Delta_h(\Delta_h u_h + P_h f)$ becomes bounded when t is away from t_0 . Meanwhile, $\Delta_h \Delta_h u_h \rightarrow -\Delta_h P_h f$ as $t \rightarrow \infty$, it converges to a non-integrable distribution when $h \rightarrow 0$.

5 Adaptive time stepping

Theorem. Assume that

(a) $u_N(t)$ is the numerical solution of equation (1), computed with the finite element method and **any** time stepping scheme.

(b) Let $\tilde{u}_h(t)$ be defined piecewise. In each interval $(t_k, t_{k+1}]$, $\tilde{u}_h(t)$ is the semi-discrete solution with initial value $u_N(t_k)$. For $q = 2$ or 3 , there is a constant C_q , such that the actual local error between $\tilde{u}_h(t_{k+1})$ and $u_N(t_{k+1})$ satisfies,

$$\|\tilde{u}_h(t_{k+1}) - u_N(t_{k+1})\| \leq C_q \tau_k^q \left\| \frac{\partial^q}{\partial t^q} \tilde{u}_h(\xi_k) \right\|,$$

for a $\xi_k \in [t_k, t_{k+1}]$.

(c) $s > 0$ is a constant, independent of the mesh and timestep sizes, $\theta_s = e^{ms} < 1$, where $m < 0$ is the first eigenvalue of the Laplace operator Δ in $H_0^1(\Omega)$.

(d) The time step sizes τ_k are chosen so that every point of the form $t_0 + ns$ is a node.

(e) The smoothing indicator $S_n^{(q)}$ remains bounded during the computation.

- (f) For a given $\epsilon > 0$, in each step $[t_k, t_{k+1}]$, $\tau_k = t_{k+1} - t_k$ is chosen so that $\tau_k^{q-1} \|S_k^{(q)}\| \leq \epsilon$.
(g) The mesh is locally refined so that the semi-discrete solution is estimated

$$\|u(t) - u_h(t)\| \leq C_s h^2 \|f\|,$$

where h is the coarsest mesh size.

Then we have the following global error estimate: For any node of the form $t_0 + ns$ from t_0 to ∞ ,

$$\|u(t_0 + ns) - u_N(t_0 + ns)\| \leq \frac{C_q s \epsilon}{1 - \theta_s} + C_s h^2 \|f\|.$$

Proof: For this linear problem, one can find in [8] a proof of assumption (g), which implies

$$\|u(t_0 + ns) - u_h(t_0 + ns)\| \leq C_s h^2 \|f\|.$$

It leaves to prove that

$$\|u_h(t_0 + ns) - u_N(t_0 + ns)\| \leq \frac{C_q s \epsilon}{1 - \theta_s}.$$

For each interval $[t_k, t_{k+1}]$, from the equations about w_h and z_h , it is obvious that

$$\left\| \frac{\partial^q}{\partial t^q} \tilde{u}_h(\xi_k) \right\| \leq \|S_k^{(q)}\|.$$

Hence, by (b) and (f),

$$\|\tilde{u}_h(t_{k+1}) - u_N(t_{k+1})\| \leq C_q \tau_k^q \|S_k^{(q)}\| \leq C_q \tau_k \epsilon.$$

Let $t_0 + ns = t_j$, $t_0 + (n+1)s = t_{j+p}$. Let \hat{u}_h be defined piecewise. In each interval $(t_0 + ns, t_0 + (n+1)s]$, \hat{u}_h is the semi-discrete solution with initial value $u_N(t_0 + ns)$. For the linear semi-discrete equation, it is easy to verify that

$$\|u_h(t_0 + (n+1)s) - \hat{u}_h(t_0 + (n+1)s)\| \leq e^{m_s} \|u_h(t_0 + ns) - u_N(t_0 + ns)\| = \theta_s \|u_h(t_0 + ns) - u_N(t_0 + ns)\|$$

and

$$\|\hat{u}_h(t_{k+1}) - \tilde{u}_h(t_{k+1})\| \leq e^{m\tau_k} \|\hat{u}_h(t_k) - u_N(t_k)\| \leq \|\hat{u}_h(t_k) - u_N(t_k)\|.$$

Within an interval $[t_0 + ns, t_0 + (n+1)s]$,

$$\begin{aligned} & \|\hat{u}_h(t_0 + (n+1)s) - u_N(t_0 + (n+1)s)\| \\ &= \|\hat{u}_h(t_{j+p}) - u_N(t_{j+p})\| \\ &\leq \|\hat{u}_h(t_{j+p}) - \tilde{u}_h(t_{j+p})\| + \|\tilde{u}_h(t_{j+p}) - u_N(t_{j+p})\| \\ &\leq \|\hat{u}_h(t_{j+p-1}) - u_N(t_{j+p-1})\| + C_q \epsilon \tau_{j+p-1} \\ &\leq \|\hat{u}_h(t_j) - u_N(t_j)\| + \sum_{k=j}^{j+p-1} C_q \tau_k \epsilon \\ &= C_q s \epsilon. \end{aligned}$$

Finally, for any n ,

$$\begin{aligned}
& \|u_h(t_0 + ns) - u_N(t_0 + ns)\| \\
& \leq \|u_h(t_0 + ns) - \hat{u}_h(t_0 + ns)\| + \|\hat{u}_h(t_0 + ns) - u_N(t_0 + ns)\| \\
& \leq \theta_s \|u_h(t_0 + (n-1)s) - u_N(t_0 + (n-1)s)\| + C_q s \epsilon \\
& \leq \sum_{k=0}^{n-1} \theta_s^k C_q s \epsilon \\
& \leq \frac{C_q s \epsilon}{1 - \theta_s}.
\end{aligned}$$

This completes the proof. \square

Remark 1. For the linear problem here, the exponential contraction property of the semi-discrete solutions is easy to verify. So it is used in the proof. When the contraction property of a semi-discrete solution is hard to verify, there is an error splitting technique in [5] and [7], with which one can prove the same error estimate by only using the contraction property of the original solution of the evolution equation.

Remark 2. If the boundedness of the smoothing indicator is proven, one can use the theorem as a uniform convergence or *a priori* error estimate result. If it is too hard to prove the boundedness, or too hard to prove the boundedness sharply, or too hard to obtain a sharp bound of the smoothing indicator while it varies in time, one can use the theorem as an efficient *a posteriori* error estimate result.

Remark 3. The word “**any**” in condition (a) indicates the powerfulness of the numerical smoothing analysis compared to numerical stability. A numerical analyst does not need to focus on the technical details of proving stability of a complex scheme any more. All needed are just proving local consistency and verifying boundedness of the smoothing indicator.

6 Conclusion remarks

There are several advantages of using numerical smoothing over numerical stability in error analysis.

1. Since only the computed numerical solution is involved, loss of additivity in a nonlinear problem will not stop the smoothing indicator computation. Hence numerical smoothing analysis works for nonlinear problems.

2. No matter what numerical schemes are used for time-discretization, the smoothing indicator at each t_n is computed in the same way. Hence an error estimate can be computed regardless of scheme complications.

3. Error propagation is estimated by using evolution equations instead of schemes. Hence one can apply any contraction property of the solutions and other objects in the solution spaces. This is actually what makes long time error estimation possible [6,7].

4. Adaptive time-stepping is natural, since the regularity of the solutions are revealed by the smoothing indicator [6].

5. The additional cost on computing the smoothing indicator is almost negligible. As shown in the formulas, computing the smoothing indicator involves the action of Δ_h and a mass matrix

inversion. The stiffness matrices corresponding to Δ_h are computed anyway, while the mass matrix inversion can be avoided by mass lumping.

It is worthwhile to observe what happens when a scheme is unstable, for example, the explicit Euler scheme. In most cases, the scheme behaves badly on fast decaying eigen-components of the original solution, making the scheme “unstable”. Meanwhile, the slowly decaying eigen-components are usually treated properly by the scheme. This is the case in both stiff ODEs and parabolic PDEs. Perhaps many of the unstable schemes should have been called “anti-smoothing schemes” in the first place.

Many open problems remain to be studied in this new topic, such as,

- local space/time adaptive computation,
- smoothing indicator for finite difference, finite volume and other finite element methods,
- study of contraction properties of PDE solutions,
- improving time-step level error propagation,
- stronger and weaker norm estimates,
- problems with small parameters,
- quasi-linear parabolic equations,
- multi-physics systems, and
- application to hyperbolic problems, etc..

References

- [1] K. Eriksson and C. Johnson, *Adaptive finite element methods for parabolic problems. V. Long-time integration*, SIAM J. Numer. Anal., 32 (1995), pp.1750-1763.
- [2] S. Larsson, *Numerical analysis of semilinear parabolic problems*, published in “The Graduate Student’s Guide to Numerical Analysis ’98”, edited by M. Ainsworth, J. Levesley and M.Marletta, SSCM Vol. 26, Springer-Verlag, 1999, pp.83-117.
- [3] P.D.Lax and R.D.Richtmyer, *Survey of the stability of linear finite difference equations*, Comm. Pure Appl. Math. 9 (1956), pp.267–293.
- [4] A. Stuart and A. Humphries, *Dynamical Systems and Numerical Analysis*, Cambridge University Press, 1996.
- [5] T. Sun, *Stability and error analysis on partially implicit schemes*, Numerical Methods for Partial Differential Equations, 21 (2005), pp.843-858.
- [6] T. Sun and R. Ewing, *Long-time error estimation and a stability indicator*, Dynamics of Continuous, Discrete and Impulsive Systems Series B: Applications & Algorithms, 9 (2002), pp.115-129.
- [7] T. Sun and D. Fillipova, *Long-time error estimation on semi-linear parabolic equations*, Journal of Computational & Applied Mathematics, 185 (2006), pp.1-18.
- [8] V. Thomee, *Galerkin Finite Element Methods for Parabolic Problems*, Springer, New York, 1997.
- [9] E. Hairer, S. Nørsett and G. Wannar, *Solving Ordinary Differential Equations I: Nonstiff Problems*, Springer-Verlag, New York, 1987.