Nonlinear reaction-diffusion equations are an important and interesting class of problems. However, we almost never know the solution of a nonlinear reaction-diffusion model and consequently numerical computation is a principal tool for analysis. But the same features that make analysis difficult also make numerics difficult. To be on a firm scientific footing, a numerical analysis of a differential equation must be accompanied by a discussion of the error in the results.

In this paper, we are interested in obtaining accurate estimates of the error of a numerical solution, as opposed to inaccurate analytic upper bounds. In the first part of the paper, we describe a computational method for estimating error. The remainder of the paper is devoted to analyzing the method and applying it to some well-known models.

After spending a few years at Stanford, Mats now has a Postdoc with Claes Johnson at Chalmers. Roy is at the Center for Advanced Computing Research at Caltech.
To give a concrete focus, we start by presenting a numerical solution of a reaction-diffusion equation.

The bistable problem is a well-studied example of metastable behavior. The steady-state solutions 1 and -1 are stable and generic solutions tend to one of these two as time passes. But the evolution can take a long time because of competition between the two steady-states. Generic data forms a pattern of layers between the values of 1 and -1 and then the layers move more or less horizontally on an exponentially slow time scale (metastability) until they become close and then they merge on a rapid time scale.

The time scale for motion of layers is \( \exp(Cd/\varepsilon^{1/2}) \) where \( d \) is the distance between the closest two layers.
General Problem

\[ u = (u_i) \in \mathbb{R}^D \text{ solves} \]

\[
\begin{cases}
\dot{u}_i - \nabla \cdot (\varepsilon_i(u)\nabla u_i) = f_i(u), & \Omega \times (0, \infty) \\
u^p = 0, & \partial \Omega \times (0, \infty) \\
u(\cdot, 0) = u_0(\cdot), & \Omega
\end{cases}
\]

where

\[ \varepsilon_i(u) \geq \varepsilon > 0, \quad 1 \leq i \leq d \]
\[ \varepsilon_i(u) \equiv 0, \quad d < i \leq D \]

\(u^p\) is the parabolic part of \(u\)

\(u^o\) is the ordinary part of \(u\)

\(\varepsilon\) and \(f\) are smooth

\(\Omega\) is convex polygonal with boundary \(\partial \Omega\)

We study a potentially singular system of reaction-diffusion equations that combines parabolic problem with ordinary differential equations (that have no diffusion). Such systems are used to model many applications.

We assume the first \(d\) equations are parabolic, and there has to be at least one of these, and the remaining \(D-d\) are ordinary. There is no parabolic smoothing in the ordinary differential equations, so we have to be careful not to assume too much regularity on the solutions.

Note that we do NOT assume any kind of global Lipschitz condition on \(f\) or \(\varepsilon\). Such assumptions are not satisfied in practice.

By the way, there can be diffusion-dominated convection terms in the parabolic equations and we can handle other boundary conditions.
### Applications

- **bistable equation**
  - motion of domain walls in ferromagnetic materials

- **population models**
  - predator-prey, competition, symbiosis

- **Hodgkin-Huxley equations**
  - transmission across nerve axons

- **Fitz-Hugh-Nagumo equations**
  - simplified model for the Hodgkin-Huxley equations

- **superconductivity in liquids**

- **Field-Noyes equations**
  - Belousov-Zhabotinsky reaction in chemical kinetics

- **flame propagation**

- **morphogenesis**
  - models for the formation and motion of patterns

- **spread of rabies in foxes in Europe**

These are some of the many applications modeled by such problems. We compute examples of each of these in the paper.
Numerics are difficult

- Solutions typically exhibit multi-scaled behavior: metastability and rapid transients, pattern formation, internal and boundary layers, ...

- The nonlinear nature allows localized behavior in classes of solutions

- Usually both stability and regularity are important in determining the behavior of solutions

- Information about solutions is often needed over long time intervals
These results illustrate the importance of both stability and regularity in determining the behavior of solutions. They also illustrate the localized nature of regularity and stability properties in a nonlinear problem.

We plot solutions of the bistable problem. In both cases, we use data that is a small smooth perturbation of a steady-state solution. When we start near 1, which is stable, both solutions tend to 1 immediately. When we start near 0, which is unstable, the solutions form a pattern of layers. Note that the solutions form different patterns and the subsequent evolution of these two solutions is much different even though the initial data are very close.
What is the error of a numerical solution?

Classic a priori error bound

$$\|u(\cdot,t) - U(\cdot,t)\| \leq e^{Lt} \left( k^p + h^r \right) C(u)$$

- $U$ is the numerical solution
- $k \sim$ time step, $h \sim$ space step
- $C(u) \sim$ unknown derivatives of $u$
- $e^{Lt} \sim$ potential accumulation of error

$L$ is usually large

Accurate approximations of a metastable layer move on the same time scale as the true layer

Inaccurate computations move on the wrong time scales and can even become fixed

Of course, the importance of obtaining error estimates on numerical solutions is widely recognized and there is a vast literature containing error estimates for numerical methods. One might naturally wonder what there can be left to talk about.

This is the kind of error estimate that is the goal of most of the literature on numerical methods for differential equations. It is useless for actually estimating the error because it requires unknown information about the solution and typically it overestimates the accumulation of error very badly.

For the bistable example I showed previously, $L$ is around 3000. There is good evidence that the errors do not accumulate that rapidly in the bistable problem.

The problem is that this classic a priori analysis is very ambitious: it tries to estimate the error of ANY solution in a general class over ANY possible time in its evolution by using an analysis in which all perturbations accumulate. It has to cover the worse case from among a lot of choices.
Numerical solution of a linear system

\[ X \text{ is the computed solution of } Ax=b \]

error: \[ e = x - X \]

residual: \[ R = AX - b \]

dual problem: \[ A^T\phi = \psi, \quad ||\psi|| = 1 \]

variational analysis

\[ e \cdot \psi = e \cdot A^T\phi = Ae \cdot \phi = -R \cdot \phi \]

error estimate: \[ |e \cdot \psi| \leq ||\phi|| \cdot ||R|| \]

stability factor: \[ ||\phi|| \]

We have a much less ambitious goal: estimate the error of a particular numerical solution over a specific time range. As much as possible, we try to do this using information gleaned from the numerical solution itself after we have computed it. This is an a posteriori analysis.

It is simplest to describe the analysis in the context of solving algebraic systems of equations.

Note the analysis yields estimates on the projection of the error, not a norm. We have to investigate the meaning of this.
A nonlinear system of equations

\( X \) is the computed solution of \( f(x) = b \)

residual: \( R = f(X) - b \)

equation for \( R \): \( f(x) - f(X) = -R \)

Mean Value Theorem

\[
f(x) - f(X) = \int_0^1 f'(sx + (1-s)X)(x-X)ds
\]

linearized dual problem: \( A^T \phi = \psi, \| \psi \| = 1 \)

with

\[
A = \int_0^1 f''(sx + (1-s)X)ds
\]

Of course we must linearize and that involves the true solution. So the analysis cannot depend solely on the numerical solution, which raises issues discussed later.
Discretization

Time:

\[ t_{n-1}, t_n \quad k_n \]

\[ h_K = \text{longest side} \]

\[ |K| = \text{area} \]

Space:

triangulation \( T_n \)

Piecewise constant mesh functions

\[ k(x,t) = k_n \quad \text{on} \quad \Omega \times I_n \]

\[ h(x,t) = h_K \quad \text{on} \quad K \times I_n \]

On \( I_n \):

\[ U = \sum_{i=0}^{q} a_i(x) t^i, \quad a_i(x) \in V_n \]

\[ V_n = \{ \text{continuous piecewise linear functions on} \ T_n \} \]

The analysis, being variational, is easiest to describe in terms of space-time finite element discretizations.

We discuss several kinds of discretizations in the paper. Specifically, the so-called discontinuous Galerkin methods of order 1 and 3 (with superconvergence) and the continuous Galerkin method of order 2. I describe the cG(1) method because it is simplest.

The discretization can be described completely by the piecewise constant mesh functions.

The approximation is a piecewise space-time polynomial.
The cG(1) finite element method

\[ U \text{ solves the weak formulation} \]

\[ \int_{t_{n-1}}^{t_n} ( (\dot{U}_i, V_i) + (\varepsilon_i(U) \nabla U_i, \nabla V_i) ) \, dt \]

\[ = \int_{t_{n-1}}^{t_n} ( f_i(U), V_i ) \, dt, \quad V \in V_n \]

\((\cdot, \cdot)\) is the \(L_2\) inner product in space

and

\[ U_0 = P_0 u_0, \quad \lim_{t \downarrow t_{n-1}} U = P_n U_{n-1} \]

\[ U_n = U(t_n) = \lim_{t \uparrow t_n} U(t) \]

\[ P_n \text{ is the } L_2 \text{ projection on } V_n \]

A finite element solution satisfies the differential equation on average. To define the average, we use the weak form of the differential equation obtained by multiplying by a test function and then integrating over the space-time domain in question. We also integrate by parts in space to put half the derivatives on the test function and so avoid taking second derivatives of the finite element solution. The solution satisfies the weak form for all weights=test functions while the finite element solution satisfies it for test functions in a finite dimensional space.

The cG(1) method is continuous except across time nodes where the space mesh has changed. Then we project the approximant onto the new mesh.

With the right quadrature to evaluate the integrals in the weak form, this is equivalent to the implicit difference scheme using the five point stencil in space and the trapezoidal rule in time. Our analysis applies to difference schemes that can be obtained by using quadrature on one of the finite element methods and we discuss the technical details for this.
The residuals are computed by substituting the approximation into the weak form of the differential equation. It turns out we have to distinguish several sources of residuals because the different residuals contribute to the error in different amounts.

In side each triangle, where $U$ is smooth, we compute a space discretization residual by simply substituting $U$ into the differential equation. In addition, there is a residual that arises because $U$ is not smooth. In particular, the gradient of $U$ is discontinuous across triangle boundaries and the second residual is computed by averaging these jumps around the boundaries. In the case of constant diffusion, this ends up looking like a second order difference operator applied to $U$.

After discretizing in space, we get a set of ordinary differential equations in time which we then discretize with a finite element method (method of lines). We compute the residual of this approximation by substituting it into the ordinary differential equations. It looks almost like the space discretization residual except that it uses the discrete diffusion operator.
The space residuals become smaller after each pair of layers collapse, while the time residual is large at the initial transient and at the times when there is a collapse.

A cross section in space at time 88 shows that the residuals are generally large near the layers. The space residuals are actually large in the transition regions to the layers while the time residual is large in the layer itself.

The amount the residuals vary on a uniform mesh suggest that adaptive error control could be used effectively.
The linearized dual problem

For a pointwise estimate at $t_n$

$$
\begin{align*}
-\dot{\phi} - \nabla \cdot (\overline{e}_i \nabla \phi_i) + \sum_{j=1}^{D} \overline{\beta}_{ij} \cdot \nabla \phi_j &= \sum_{j=1}^{D} \overline{f}_{ij} \phi_j \\
\phi^p &= 0 \\
\phi(\cdot, t_n) &= \psi(\cdot)
\end{align*}
$$

where $t$ runs "backwards" and

$$
\begin{align*}
\overline{e}_i &= \int_0^1 e_i (us + U(1-s)) \, ds \\
\overline{\beta}_{ij} &= \int_0^1 \frac{\partial e_i}{\partial u_i} (us + U(1-s)) \nabla (u_j s + U_j(1-s)) \, ds \\
\overline{f}_{ij} &= \int_0^1 \frac{\partial f_j}{\partial u_i} (us + U(1-s)) \, ds
\end{align*}
$$

The dual equation is obtained by integrating the weak form of the differential equation by parts until all of the derivatives fall on the test function. It is a linear reaction-diffusion equation of the same type as the original problem. We linearize around an average of the true and approximate solutions.

The dual problem runs "backwards" in time but the time derivative is multiplied by a -1 to compensate.

Note that in the case when the diffusion coefficient depends on $U$, the dual problem has convection terms. If the diffusion varies a lot, then the dual problem can have strong convection, leading to boundary and internal layers.
The stability factors

Initial error

\[ S_0^0(0, t_n) = \| \phi(0) \| \]

Time discretization

\[ S_{t}^{\alpha}(0, t_n) = C_t^{\alpha} \int_0^{t_n} \| D_t^{\alpha} \phi \| \, dt \]

Space discretization

\[ S_{x}^{p}(0, t_n) = C_x^{p} \int_0^{t_n} \| D_x^{2} \phi^p \| \, dt \]

\[ S_{x}^{\alpha}(0, t_n) = \int_0^{t_n} \| \phi^o \| \, dt \]

\[ \| \| \] is the \( L_2 \) norm in space

\( C_t^{\alpha} \) and \( C_x^{\alpha} \) are interpolation constants

The stability factors are given by particular semi-norms on the dual solution.

There is a stability factor corresponding to each source of residual: initial error, time discretization, space discretization. If we use quadrature or allow errors in the coefficients of the differential equation, there are stability factors associated to those as well. It is important to keep track of these different factors because the various residuals contribute to the error at different rates.
Stability factors for the heat equation

The stability factors for the heat equation behave as we might expect. The heat equation damps out perturbations as time passes, so the stability factor corresponding to error in the initial data tends to zero. The stability factors for discretization tend to a constant near 1: after some time, the only error found at any step is the error made on that step. There is no accumulation from past errors.
The a posteriori error estimate

Theorem  For the cG(1) method,
\[
\left| (e(t_n), \psi) \right| \leq S_x^p(0, t_n) \left( \| h^2 R_x^p(U) \| + \| h^2 R_x^p(U) \| \right)
+ S_x^o(0, t_n) \| (I - P) R_x^o(U) \|
+ S_t^1(0, t_n) \| k R_t(U) \|
+ S_0(0, t_n) \| e(0) \|
\]

The norms are \( L_2 \) in space and maximum in time

Computing the estimate

- Compute \( U \)
- Solve the dual problem numerically
- Compute residuals and stability factors
- Compute the estimate

The estimate is made up of a linear combination of residuals scaled by the appropriate stability factors.

To stress the a posteriori nature of this estimate, we show how it is used.
Questions

- Are the residuals and stability factors defined?
- Is it worth computing the a posteriori error estimate?
- Can the quantities in the estimate be computed?
- Does the a posteriori estimate imply the method converges?

The analysis I described so far is the first part of the paper. This is really the easy part, the hard part is understanding what the estimate means and how to use it.

We can summarize the issues as four questions. The bulk of the paper is devoted to answering these as best we can, sometimes with analysis and sometimes with computational evidence.

We don’t answer all of these completely and there are some interesting questions for the future.
We present a new way to analyze a posteriori error estimates.

To analyze the residuals on each interval, we compare $U$ to a “local” solution $u$ that satisfies the differential equation on the interval in question beginning with data obtained by smoothing the last value of $U$. It is necessary to smooth $U$ to avoid technical issues arising from non-smooth data that destroy convergence. We can smooth $U$ two ways: by evolving the heat equation for a short time starting with $U$ or by solving a Poisson problem using the discrete diffusion operator applied to $U$ as the right-hand side.
The chaotic Lorenz problem

We show two solutions of the Lorenz problem that start at the same point but which are computed with different accuracies. The solution on the right becomes 100% inaccurate at around t=18. Yet we can see that the residual is not large there. On the other hand, the stability factor suddenly increases in value around this time. The numerical solution becomes inaccurate because it becomes very sensitive to perturbations in a particular region of phase space and not because the differential equation is difficult to solve. The errors that have been accumulating before this time are magnified rapidly in this region.

Any kind of “local” error control will fail to see the change in stability and the subsequent increase in error.
We mark the region of phase space associated with large changes in the stability factors. Notice the two solutions that are very close but then become very far apart in a short time.
In the bistable problem, we see there is a slow exponential growth of error during the metastable periods, but there is a rapid decay of error during the transients when layers collapse. This allows long time computations to be performed accurately.
The bistable problem in two dimensions

In two dimensions, the time scale for evolution in the bistable problem is much different. Now the evolution is governed by “motion by mean curvature” meaning that the normal velocity of a transition layer is proportional to the sum of the principle curvatures of the layer. This time scale is only polynomial, curvature/epsilon, as opposed to the exponential time scale in metastable solutions of the one dimensional bistable problem.

We choose data and epsilon so the time scale of the solution is similar to the example in one dimension displayed before. Plotting the stability factors, we see that the two dimensional version is simply not as sensitive to errors. This is understandable, small perturbations smooth very rapidly because of the effects of curvature.
The stability factors are a property of the solution and are more or less independent of discretization with sufficient accuracy. What can be adjusted is the size of the residuals that multiply the stability factors. So indirectly the stability factors give the time scale over which accurate solutions can be computed, i.e. as long as we can keep the residuals sufficiently small.
Two important issues

- How do we choose the data for the dual problem?

- What is the effect of linearizing around the approximation instead of the average of the true and approximate solutions?

There are two main difficulties to be addressed.
Choosing data for the dual problem

We estimate the size of a projection

\[ \left| (e(t_n), \psi) \right| \]

not a norm.

- Choose \( \psi \) to estimate the error in a specific functional of the solution

- Special choices of data, for example \( \psi = e_n / \| e_n \| \), yield an estimate on \( \| e_n \| \)

We have to consider the fact that we estimate a projection and not a norm carefully. A projection can certainly be zero even if the error is non zero!

Actually, this usually a feature rather than a problem. Very often in practice, it is not the solution of the differential equation that is interesting, but some quantity computed from the solution. For example, lift and drag on the surface of an object in a fluids model. If we can choose the data for the dual problem to give this quantity, then the dual problem indicates the sensitivity of that quantity to error. This can lead to an enormous gain in efficiency as compared to computing to keep some general norm of the error under control.

Special, but impractical, choices of data yield estimates on the norm of the error. We analyze the possibility of choosing this data using probability.
Estimating the error in a functional

\[
\begin{cases}
-\nabla \cdot (a(x, y) \nabla u) = 1 & [0,1] \times [0,4] \\
u = 0 & \text{boundary}
\end{cases}
\]

\[
a(x, y) = \frac{1}{2} \left( 1 + \tanh(7(y - .8) + 25|x - .5|) \right)
\]

The problem is nearly singular at (.5,0)
The stability of the solution varies greatly

To show the possible gain that comes from estimating the error in a specific functional of the solution, we consider the problem of computing the solution of a nearly singular elliptic problem. We compute the solution in a long thin domain while the diffusion becomes nearly zero at one end. The corresponding solution has a large “spike” near the singular point.
Briefly, to do adaptive error control we use a different form of the estimate in which we keep the local stability factors together with the residuals on each element. The goal of the adaptive meshing is to equidistribute the local contributions over the mesh under the constraint of keeping the total estimate below a tolerance. This makes the mesh optimal in some sense. Our approach is different than standard approaches because an element will be refined for two reasons: residual and stability.
Measuring the error in two ways

Data for the dual problem:

We measure the error in two ways corresponding to two choices of initial data for the dual problem. We first measure the average error by choosing the data to be one everywhere. The second choice is an approximate delta function that gives the error at the far end (.5,3.5).
Measuring the error in two ways

Meshes after seven refinements:

We started both computations with 4x16 meshes and a tolerance of $10^{-4}$. After seven refinements, we end up with these two meshes. The average error has resulted in 5760 elements giving an accuracy of .015 while the computation for the error at the point (.5,3.5) has used 210 elements to get an error of .00006. The point error computation requires orders of magnitude fewer elements to reach the same accuracy as an computation for the average error. The corresponding solutions are plotted and we can see how inaccurately the solution has been computed while still achieving the accuracy in the desired point value.
A probability result

**Theorem**  By doing more work, we can improve the accuracy of the a posteriori error estimate and the probability that the error is bounded by the computed estimate.

Given $n$ random unit vectors $\{\psi_i\}$, we compute the probability that the maximum of the projections of the error on the $\{\psi_i\}$ is larger than a fixed fraction of the norm of the error.

The probability decreases geometrically with dimension but we use the fact the error is in $H^{2-\varepsilon}$ to apply the result to reaction-diffusion equations.

Probability is a convenient way to talk about reliability. This results says that increasing work yields more accuracy and more reliability.

To apply the result to reaction-diffusion equations, we represent the error in terms of a Fourier series and use the decay of the coefficients to truncate the series at a finite point with an error that can be estimated.
We show the stability factors corresponding to three different choices of initial data for the Lorenz and bistable problems.
Altering the linearization

The important issue is the degree of nonlinearity of the problem in the sense of whether or not nearby trajectories have similar stability properties.

Linearization is always an issue when analyzing numerical solutions of nonlinear differential equations. The issue here is whether the error estimate can be “fooled”, i.e. can the estimate remain small when the error is large?

This can happen when the numerical solution is more stable than the true solution. In this bistable problem, if we compute with a very coarse mesh, the numerical approximation of a metastable layer actually becomes stable and fixed. If we use 21 nodes in the previous bistable example, then the left-hand well collapses at the wrong time and the right-hand well becomes stable. We can see this behavior reflected in the stability factors, unfortunately. Luckily the error estimate for this solution is over 200, so we are not likely to trust the results.
Accuracy experiment

\[ \bar{u} \] is computed using 513 elements and time steps smaller than .00004

\[ U \] is computed using 129 elements and time steps greater than .004

To test the accuracy of the estimates, we compare the estimate for an inaccurate numerical solution to the “error” obtained by subtracting the inaccurate solution from a very accurate solution. The nearly constant ratio of the error to bound shows that the accumulation of errors is predicted very well.

The decrease in the ratio after the first transient is due to the form of the error estimate and the fact we are not using adaptive error control. Using adaptive error control to smooth out the residuals reduces this effect.

By the way, we think that tests on accuracy of error estimates should be performed on inaccurate computations, not very accurate computations.
The a posteriori bound does not imply convergence because the stability factors can grow when the discretization is refined.

Finite time blowup

\[ \dot{u} = u^2 \Rightarrow u = \frac{1}{1-t} \]

The a posteriori estimate does not imply convergence, as an example of blowup demonstrates.
Parabolic stability comes to the rescue

Ways to improve stability

- Assume the problem is strongly dissipative and $f$ is globally Lipschitz continuous

  This implies essentially linear behavior

- Assume conditions that guarantee the existence of an invariant region inside of which any solution remains for all time

  This allows a wide variety of behavior inside the region while avoiding problems due to the possibility of blowup and loss of regularity

Classic analysis assumes a strong dissipative nature which means there is little or very slow accumulation of errors. Practical problems rarely satisfy such conditions.
**Invariant rectangles**

A generalized rectangle in solution space with sides parallel to the coordinate axes.

**Invariant rectangle condition**

\[
\nu_R(u) \cdot f(u, \cdot, \cdot) \leq 0 \quad u \in \partial R
\]

\(\nu_R(u)\) is the outward unit normal to \(\partial R\).

---

All but one of the models listed earlier in the talk admit invariant rectangles.
Consequences of an invariant rectangle

If invariant rectangles exist for both $U$ and $u$:

- The residuals and stability factors can be bounded uniformly with respect to data in the rectangle.
- The a posteriori error estimate implies convergence.
- The stability factors can be approximated uniformly well in some sense.

This means that the classic a priori error analysis can be avoided entirely!
An approximate invariant rectangle

Using the a posteriori error estimate to control the error, we can try to keep $U$ inside an approximation of the invariant rectangle for $u$

But rigorous a priori bounds on the stability factors grow rapidly as time passes, leading to a severe time-dependent constraint on the size of the residuals, unless we are clever

To gain these benefits, we need invariant rectangles for both the true and numerical solutions.

The idea is to use the a posteriori estimate to control the error of $U$. If $U$ is close to the true solution and the true solution remains inside a rectangle, then $U$ cannot be far away. The problem is that the error estimate grows with time, so that $U$ may “escape” after all.
Preservation of a “fuzzy” rectangle

We assume there are concentric rectangles \( R_i \subset R_o \) such that \( f \) satisfies a minimum angle condition in the region between \( \partial R_i \) and \( \partial R_o \)

We prove that \( U \) remains in \( R_o \) for all time provided the initial data is in \( R_i \) and the residuals are kept smaller than a fixed tolerance independent of time

These assumptions introduce a fundamental time scale into the problem. This is the time it takes for any solution of the differential equation that is in the outer rectangle to enter the inner rectangle. We can estimate this.

When the numerical solution is outside the inner rectangle but in the outer rectangle, we compare it to a nearby true solution which drags the approximation along when it moves back inside. Thus, we only have to compare the numerical solution to true solutions over the fundamental time, which is relatively short.
Exact preservation of an invariant rectangle

This is related to whether the finite element method satisfies a maximum principle when applied to the heat equation

We modify the methods by using the lumped mass quadrature to evaluate the integrals in space

Under what amounts to a severe C-F-L condition on the time steps, any invariant rectangle for the true solution is also invariant for the modified Galerkin methods

This analysis is related to the results of David Hoff in his beautiful paper on preservation of invariant rectangles under discretization by finite difference methods.