A Posteriori Error Analysis,
Correction and Grid Adaptation

Mike Giles
Oxford University Computing Laboratory

September/October 2001
Outline

Engineering objectives:
- 'best' accuracy for a given CPU cost
- useful error bounds for key outputs

Next 5 lectures:
- present research at Oxford University towards above goals
- Mike Giles
  - error analysis/correction, FV methods
- Endre Süli
  - error bounds/adaptation, FE methods

Lecture 1: A posteriori error analysis
Outline

Lecture 1:
- key ideas, errors and bounds
- linear algebra theory

Lecture 2:
- linear p.d.e.’s
- inhomogeneous b.c.’s
- defect correction

Lecture 3:
- nonlinear p.d.e.’s
- grid adaptation
- future challenges

Lecture 1: A posteriori error analysis
Objectives

1) improve accuracy for given CPU cost:
   - reduce errors through grid adaptation
   - correct errors, to leading order

2) bound remaining error:
   - guaranteed or not?
   - tight or not?
Different kinds of error

1) solution error \(|u - u_h|\)

- \(u\) = analytic solution
- \(u_h\) = numerical solution

Question: what norm to use?

Answer: whatever makes analysis feasible!

A bit flippant – but contains the key truth that this error is often unrelated to engineers’ concerns.
Different kinds of error

2) functional errors – errors in derived output quantities of interest to engineers

In CFD, these are usually integral quantities:
- lift or drag
- integrated heat flux or mass flux
- average temperature
- total production of NOx

In structural analysis, often point quantities:
- maximum stress
- maximum temperature

Lecture 1: A posteriori error analysis
Different kinds of error bound

1) *a priori* bound: there are positive constants $c, p$ such that

$$\text{Error} < c h^p$$

where $h$ is the representative grid spacing.

Analysis of the truncation error gives $p$, but quantifying $c$ is tough since it depends on the (unknown) solution $u$. 

Lecture 1: *A posteriori* error analysis
Different kinds of error bound

2) Guaranteed \textit{a posteriori} bound: there is a computable function $e(u_h)$ such that

$$\text{Error} < e(u_h)$$

$e(u_h)$ measures the extent to which $u_h$ fails to satisfy the original p.d.e. or its b.c.'s.

The ratio $e(u_h)/\text{Error}$ defines the efficiency (or tightness) of the bound:

- 1 (perfect)
- 2-10 (useful)
- >100 (useless?)

Lecture 1: \textit{A posteriori} error analysis
Different kinds of error bound

3) Asymptotic \textit{a posteriori} bound: there is an unknown $h_0$ and a computable function $e(u_h)$ such that

$$\text{Error} < e(u_h) \quad \text{for all } h < h_0$$

Example: if

$$\text{Error} = 1.36 \, h^2 + 0.77 \, h^4,$$

then $1.37 \, h^2$ is an asymptotic bound which is valid for $0.77 \, h^4 < 0.01 \, h^2$. 

Lecture 1: \textit{A posteriori} error analysis
Different kinds of error bound

Some comments:

1) *A priori* bounds are useful for the order of accuracy, but are usually not at all tight.

2) Guaranteed *a posteriori* bounds are the ideal, but can be hard to obtain, especially for nonlinear problems.

3) Asymptotic *a posteriori* bounds are more achievable, but given choice between greater order of accuracy or an asymptotic bound, I would always choose the improved accuracy.
Different kinds of error bound

Why? Suppose $u_h^{(1)}$ has error $c_1 h^2$

$u_h^{(2)}$ has error $c_2 h^4$

then $1.1 \left| u_h^{(1)} - u_h^{(2)} \right|$ provides a near-perfect

asymptotic error bound for $u_h^{(1)}$ but I’d rather just use $u_h^{(2)}$!

However, for certain applications
(e.g. safety critical cases) I’d be happy to sacrifice
accuracy for guaranteed error bounds.
Our aims

So, our objective is to use *a posteriori* error analysis to

- adapt grid to reduce errors
  (Süli & Houston)
- correct leading order error
  (Giles & Pierce)

and then try to bound the remaining error.
### Our aims

<table>
<thead>
<tr>
<th>Giles &amp; Pierce</th>
<th>Süli &amp; Houston</th>
</tr>
</thead>
<tbody>
<tr>
<td>FV/FE/FD</td>
<td>finite element</td>
</tr>
<tr>
<td>adaptation</td>
<td>adaptation</td>
</tr>
<tr>
<td>elliptic p.d.e.</td>
<td>elliptic p.d.e.</td>
</tr>
<tr>
<td>Euler equations</td>
<td>convection/diffusion incompressible N-S</td>
</tr>
</tbody>
</table>

Lecture 1: *A posteriori* error analysis
Adjoint in linear algebra

The analysis of functional errors naturally introduces the use of adjoint equations.

This is a key concept, fundamental to many lectures this week, so we begin with linear algebra.

Suppose we want to evaluate the scalar product

\[ g^T u \]

where \( u \) is the solution of

\[ Au = f \]
An equivalent *dual* treatment is to evaluate the product

\[ v^T f \]

where the \( v \) is the solution of the *adjoint* (or *dual*) equation

\[ A^T v = g \]

The equivalence comes from

\[ v^T f = v^T A u = g^T u \]
So, to obtain a linear output functional from a linear system of equations, we can *either* solve the original (primal) equations *or* solve the adjoint (dual) equations.

We are now going to look at how this result can be used in two different ways:

- design optimisation
- error analysis and correction

(Why discuss design optimisation? – because this is why most people write adjoint codes!)
Design optimisation

Want to minimise a nonlinear objective function $J(U)$, where $U$ is the solution of the nonlinear equations $N(U, \alpha) = 0$.

For aerodynamic design, may have
- $\alpha$ – geometric design variables
- $J(U)$ – drag
- $N(U, \alpha)$ – discrete flow equations
Design optimisation

For a single $\alpha$, can linearise about a base solution $U_0$ to get:

$$\frac{dJ}{d\alpha} = g^T u, \quad Au = f$$

where

$$u \equiv \frac{dU}{d\alpha}, \quad g^T = \frac{\partial J}{\partial U}, \quad A = \frac{\partial N}{\partial U}, \quad f = -\frac{\partial N}{\partial \alpha}.$$ 

For multiple $\alpha$ each has a different $f$, but the same $g$, so the adjoint approach is much cheaper!
Linear error analysis

Back to the original problem: evaluate $g^T u$
subject to

\[ Au = f, \]

and the dual problem: evaluate $v^T f$
subject to

\[ A^T v = g \]

Suppose we have approximate solutions $\tilde{u}, \tilde{v}$. 

Lecture 1: *A posteriori* error analysis
Linear error analysis

Then, we have

\[ g^T u = g^T \tilde{u} + g^T (u - \tilde{u}) \]
\[ = g^T \tilde{u} + v^T A (u - \tilde{u}) \]
\[ = g^T \tilde{u} + \tilde{v}^T A (u - \tilde{u}) + (v - \tilde{v})^T A (u - \tilde{u}) \]
\[ = \underbrace{g^T \tilde{u} + \tilde{v}^T (f - A\tilde{u})}_{\text{computable}} + \underbrace{(v - \tilde{v})^T A (u - \tilde{u})}_{\text{very small}} \]

Hence, the computable quantity

\[ g^T \tilde{u} + \tilde{v}^T (f - A\tilde{u}) \]

is much more accurate than \( g^T \tilde{u} \) or \( \tilde{v}^T f \).
Linear error analysis

Note the form of the error correction term

$$\tilde{v}^T (f - A\tilde{u}).$$

- $f - A\tilde{u}$ is the residual error in solving the equations $Au = f$
- $\tilde{v}$ is an error weighting, showing the effect of the residual error on the output quantity $g^T u$
Nonlinear error analysis

Suppose we want to evaluate $J(U)$, where $U$ is the solution of the nonlinear equations

$$N(U) = 0.$$ 

Given an approximate solution $\tilde{U}$, we define

$$u = \tilde{U} - U,$$

and then linearisation gives

$$J(U) \approx J(\tilde{U}) + g^T u, \quad Au \approx f,$$

where

$$g^T = \frac{\partial J}{\partial U}, \quad A = \frac{\partial N}{\partial U}, \quad f = -N(\tilde{U}).$$
Nonlinear error analysis

If $v$ is defined to satisfy the adjoint equation

$$A^T v = g,$$

then we obtain

$$J(U) \approx J(\tilde{U}) + v^T f \approx J(\tilde{U}) - v^T N(\tilde{U}).$$

Hence, the quantity

$$J(\tilde{U}) - v^T N(\tilde{U})$$

is a more accurate estimate for $J(U)$ than $J(\tilde{U})$ alone.

Lecture 1: *A posteriori* error analysis
Again note the form of the error correction term

\[ v^T N(\tilde{U}) \]

- \( N(\tilde{U}) \) is the residual error in solving the nonlinear equations
- \( v \) is the error weighting, showing the effect of the residual error on the output quantity \( J(U) \)
Nonlinear error analysis

This algebraic approach to error correction is a zero-dimensional version of the differential theory of Giles & Pierce (1998).

It has been used by Darmofal & Venditti (1999) for error correction with the quasi-1D Euler equations, obtaining \( \tilde{U} \) and an approximate adjoint solution \( \tilde{\nu} \) on a fine grid by interpolation of solutions on a much coarser grid.
Nonlinear error analysis

Darmofal & Venditti also used the analysis for grid adaptation.

Splitting the error correction into a summation over all of the grid nodes,

\[ \tilde{v}^T N(\tilde{U}) \equiv \sum_i \tilde{v}^T_i N_i(\tilde{U}) \]

they refined the grid around nodes for which \( \tilde{v}^T_i N_i(\tilde{U}) \) was large. This was particularly effective at producing grid refinement near the boundaries where a lower-order discretisation had been used.
Nonlinear error analysis

The technique has also been derived independently by Langlois (1998) for use in correcting the effects of machine roundoff errors.

In this application, each equation in $N(U) = 0$ corresponds to one machine instruction, relating one output to one or more inputs, and $J(U)$ is the value of the output from the final instruction.
Nonlinear error analysis

$\tilde{U}$ is the solution computed with finite precision arithmetic, so $N(\tilde{U})$ corresponds to the roundoff error in each operation.

The (approximate) adjoint solution is obtained by reverse mode Automatic Differentiation (a technique which holds great promise for the rapid development of adjoint codes).

The error correction then gives a value for the final output with approximately twice as many correct digits.
Key Points

- What do you want from a calculation?
- What errors do you care about?
- Are error bounds guaranteed?
- Are error bounds useful?
- Is adaptation strategy efficient?
- Adjoint equations arise naturally when concerned about particular outputs
- Theory is (relatively) simple for algebraic equations

Lecture 1: *A posteriori* error analysis