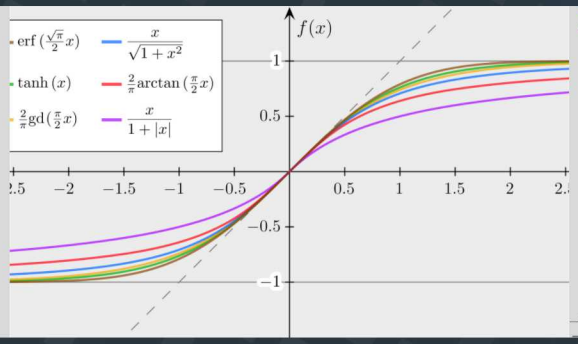
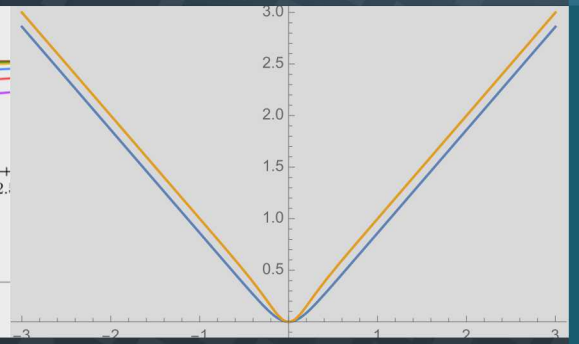
SAND2019-9340C

Exceptional service in the national interest

Sandia National Laboratories

$$\tau(U) = ch^2 |F'(U)| \frac{|U_x U_{xx}|}{U_x}$$





The Power and Secrets of the Sigmoid Function: A Numerical Swiss Army Knife

Bill Rider

Sandia National Laboratories, Albuquerque

wjrider@sandia.gov

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

The talk revolves around three major goals

The talk outline

- **Goal 1:** Adaptive methods are essential to modern computational methods (and models too) and involve switches in implementation most simply ”if then else” logical structure.
- **Goal 2:** For analysis reasons the switches used to define these switches should be implemented and described using functions. This has practical and theoretical utility
- **Goal 3:** The classical version of these functions are discontinuous, but this causes numerous problems. It is beneficial to remove the discontinuities and uses sigmoid functions to create smooth continuously differentiable versions of the switches.

Adaptive methods are central to modern numerics

Without adaptive methods our codes would have very little functionality.

- The upwind (characteristic) bias in discretization is essential to effective numerical methods.
- Overcoming Godunov's theorem created a broad class of adaptive numerical methods where the discretization adapts to the solution
  - –This is implemented in the form of limiters in the most basic form
  - –Stencil selection is another form
  - –Interface tracking is another less recognized form of adaptation
- Upwind methods and limiters (etc) are veritable cottage industries
- Most models used in codes are also adaptive based on physics.

Non-adaptive methods are limited in utility and usually quite fragile

The ideal implementation of the adaptive method

Methods all need to be put onto a computer for utility. Programmed logic is the standard means

- A simple code for an upwind method

$$\partial_t u + a\partial_x = 0$$

```
if (a > 0.0) then
    Δu = u_j − u_{j-1}
else
    Δu = u_{j+1} − u_j
```

One thing to note immediately is that the selection is asymmetric.

If-then-else constructions are essential for the simplest constructions of the adaptive methods

My first modification for this from decades ago

Methods all need to be put onto a computer for utility. Programmed logic is the standard means

- A modified upwind implementation

$$\partial_t u + a\partial_x = 0$$

if (a > smallvel) then

$$\Delta u = u_j - u_{j-1}$$

elseif (a < -smallvel) then

$$\Delta u = u_{j+1} - u_j$$

else

$$\Delta u = \tfrac{1}{2} (u_{j+1} - u_{j-1})$$

"Fixed" a symmetry issue in a multimaterial incompressible method.

If-then-else constructions are essential for the simplest constructions of the adaptive methods

How to replace the if-then-else with a function

We turn something that is programming into a concise mathematical function

- We can quite easily replace the previous implementation with a function

$$\Delta u = \tfrac{1}{2} (\text{Sign}(a) + 1) (u_j - u_{j-1}) + \tfrac{1}{2} (1 - \text{Sign}(a)) (u_{j+1} - u_j)$$

Depending how Sign(a) is implemented this might be symmetric, so its better.

The absolute value function is closely related,

$$|x| = \sqrt{x^2} = x \, \text{sign}(x) = \frac{x}{\text{sign}(x)}$$

The functional description is much better for a variety of reasons

The ideal implementation of the adaptive method

Methods all need to be put onto a computer for utility. Programmed logic is the means

- A simple code for the minmod limiter

```
if (a*b > 0.0) then
    if (sign(a)*a < sign(b)*b) then
        Δu = a
    else
        Δu = b
else
    Δu = 0.0
```

If-then-else constructions are essential for the simplest constructions of the adaptive methods

The minmod can be functionally represented too

We have a few more options and some difficulties to overcome.

- This is the usual way to implement the minmod

  minmod(a,b) = sign(a) max(0.0, min(sign(a)*b,abs(a))

- The intrinsic sign(), abs(), min() or max() functions are not amenable to analysis.
- abs(a) = sqrt(a^2), max(a,b) = ½(a+b)+ ½ abs(a-b),
  min(a,b) = ½(a+b)- ½ abs(a-b), sign(a) = a/abs(a)
- minmod(a,b)= ½ (sign(a)+sign(b)) * (abs(a+b) – abs(a-b))

The simplest way to do this is not sufficient for analysis means

This functional relationship is extremely important

We can now see unambiguous connections between different methods

- Once we can write methods as functions we can perform analysis on them. For upwinding this has turned out to be very important.
- If we look at upwind as a flux instead of a difference, $f_{j+1/2} = (a\ u)_{j+1/2}$
- Here with a flux we transform the sign to an abs function and express the flux as $f_{j+1/2} = \frac{1}{2}(f_j + f_{j+1}) - \frac{1}{2}\ \mathrm{abs}(a_{j+1/2})(u_{j+1} - u_j)$
- Now the function starts to look far different, we have an average of the flux, minus a difference in the variable.
- Combining this with the flux from the opposite edge provides an expression that looks like a classical diffusion operator.

Having methods expressed at functions allows us to look at methods far more deeply

The importance of modified equation analysis

This is an essential result for the numerical solution of hyperbolic PDEs

- HHL '76 established positivity of coefficients and prelude to TVD, and entropy conditions via vanishing viscosity
- The concept of vanishing viscosity is essential

$$\partial_t u + \partial_x f(u) = \nu \partial_{xx} u \quad \nu \to 0^+$$

- The entropy condition was derived via the modified equation analysis, first order produces a viscous term as the leading truncation error (modified equation).
  - This mode of analysis largely disappeared with the focus moving to discrete conditions providing nonlinear stability

Harten, Amiram, James M. Hyman, Peter D. Lax, and Barbara Keyfitz. "On finite-difference approximations and entropy conditions for shocks." *Communications on pure and applied mathematics* 29, no. 3 (1976): 297-322.

Using functions has allowed methods to be analyzed and examined using mathematical analysis

Functional relationships used for limiters

We can write limiters without using if-then-else statements

- As I showed earlier we can similarly reformulate limiters to use expressions amenable to expanding differentially.
- This allows the analysis of the limiters in a form that unveils their impact on the approximations.
- I did this in service of understanding the effective subgrid model for ILES methods, the characteristic speed a is expressed as F'(U) and expressing the limiter as a modification of the subgrid "stress"

Grinstein, Fernando F., Len G. Margolin, and William J. Rider, eds. *Implicit large eddy simulation: computing turbulent fluid dynamics*. Cambridge university press, 2007.

The if-then-else is simpler in many ways, but cannot be analyzed using standard tools

High Resolution Methods and their Modified Equation the "limiters" in differential form

$$\tau(U) = ch^2 |F'(U)| \frac{|U_x U_{xx}|}{U_x}$$ minmod, or mineno

$$\tau(U) = ch^3 |F'(U)| \frac{|U_x U_{xxx}|}{U_x}$$ UNO

$$\tau(U) = ch^3 |F'(U)| \frac{(U_{xx})^2}{U_x}$$ WENO-3rd

$$\tau(U) = ch^5 |F'(U)| \frac{(U_{xxx})^2}{U_x}$$ WENO-5th

Problems with the functional form of methods

These also plague the if-then-else constructions

- We can so this exchange of if-then-else constructions quite broadly
- For all intents and purposes the functional representation is as good or better than the if-then-else logic, just less flexible.
- The functional representation may be less clear than the logic.
- While this is a real improvement in a number of respects a number of pathologies are present too.
- Many of these pathologies are associated with the sharp transitions and lack of smoothness at critical points in the function. These are likely to cause substantial problems in codes and calculations.

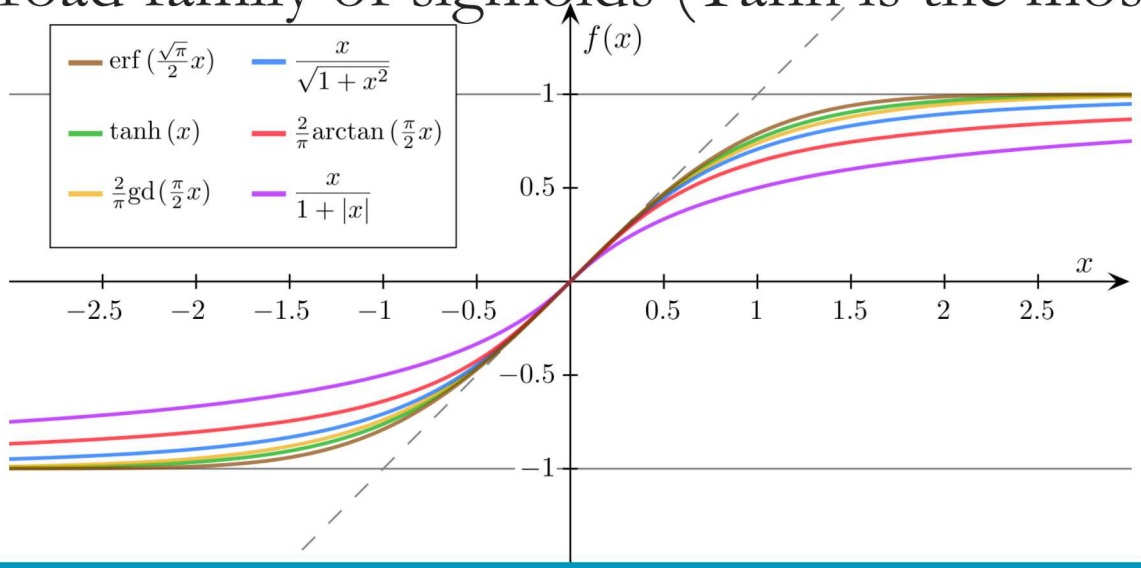
When we make large numerical changes based on infinitesimal changes we can expect problems

Fortunately there is a ready made solution to these issues used broadly in related fields.

The sigmoid function is used extensively in NN

There is a veritable library of sigmoid functions (see Wikipedia)

- One of the first things to recognize is the pervausive use of sigmoid functions in neural nets as activation functions.
- There is a broad family of sigmoids (Tanh is the most common)



We have a lot of options beyond the hyperbolic tangent

The sigmoids are highly controlled in smoothness

We have both significant control although the functions are highly nonlinear

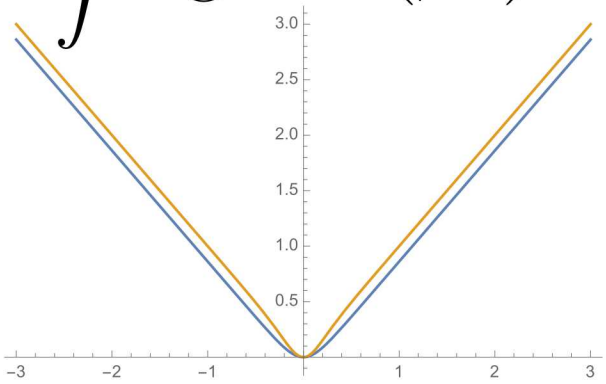
- We can control the steepness of the basis for any sigmoid through a single parameter, β
- This parameter can be chosen to provide the properties desired for the function. As $\beta >> 1$ (for most sigmoids) the function approaches a Heaviside function.
- At the origin (or shifted origin) the function is usually linear, but its wise to check the sigmoids asymptotics
- It is usual to analyze the method in these particular limits

This flexibility is one of the sigmoids greatest strengths and worst weaknesses too

How to compose the sigmoid into other functions

Having a smoothed sign function is not enough

- The sigmoid replaces the Sign function, i.e., $\tanh(\beta x) \approx \text{sign}(x)$
- Now we can replace the absolute value with $x \tanh(\beta x) \approx |x|$
- Another option is to integrate the sigmoid, $|x| \approx \int \text{sigmoid}(\beta x)\, dx$
  - For example $\int \tanh(\beta x)\, dx = \frac{1}{\beta} \cosh(\beta x)$



- Use a short hand, $\text{sign}_\beta(x)$ and $|x|_\beta$
- Other sigmoids can be used. For example different sigmoids may be more tractable analytically for differentiation, integration or analysis.

We have some latitude and options to produce the library of functions we need.

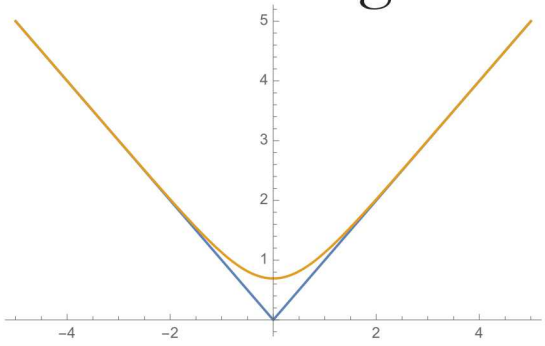
Here is an example of a particularly useful function

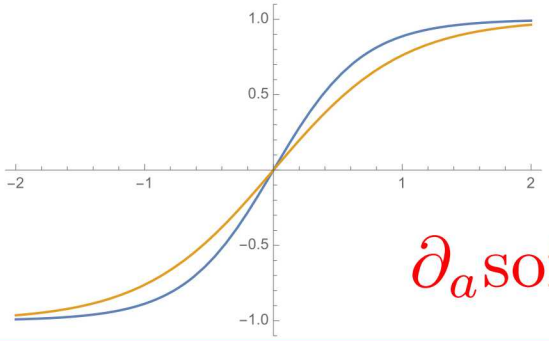
With a little work one can derive a family of functions

- Starting with the need to smooth optimization out the Smooth max (and min) was derived

$$\text{softmax}(a,b) = \log\left[\frac{1}{\beta}\left(\exp(\beta a) + \exp(\beta b)\right)\right]$$

- We can easily derive an absolute value from this $|a| \approx \text{softmax}(-a, a)$
- Then a derived sigmoid follows, $\text{softsign}(a) = \dfrac{a}{\text{softmax}(-a, a)}$





$$\partial_a \text{softabs}(a) = tanh(a)$$

This function is structurally different from other sigmoids and potentially important

Using the sigmoids for "upwinding"

Using either the raw "Sign" function or the derived "Abs" function can smooth out upwinding

- Once we have our sigmoids in hand we can use them for constructing "classical" upwind formulas,
- For example we have the flux formula

$$F_{j+1/2} = \frac{1}{2}\left(F_j + F_{j+1}\right) - \frac{1}{2}\left|\lambda\right|_\beta \left(U_{j+1} - U_j\right)$$

- We can choose the smoothing to have a similar effect to the Harten-Hyman entropy fix if we choose our functions wisely.
- Here we want to use the "softabs" to avoid the numerical viscosity going to zero identically.

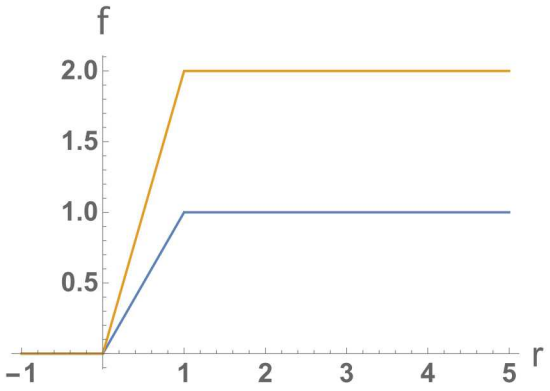
This is a very clever way to remove the "special" behavior at zero upwinding velocity

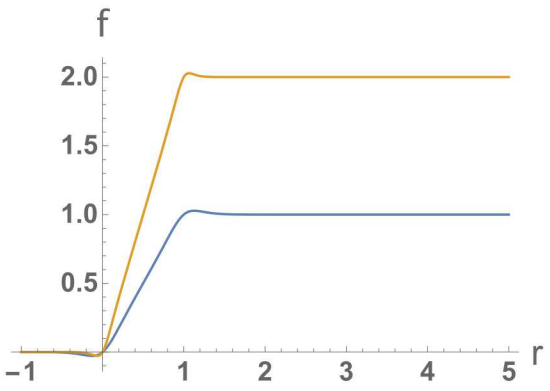
Using the sigmoids to implement limiters

These functions can be used to remove all sharp transitions without harming function

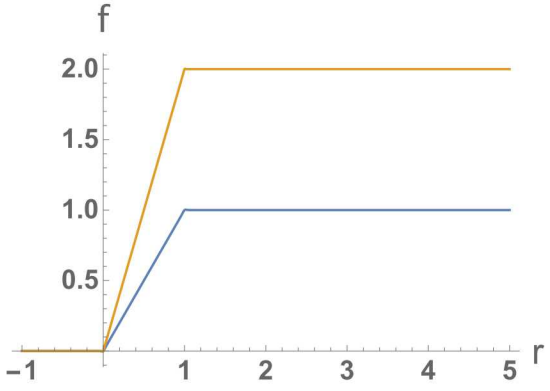
- Using the previous definitions for max, min and minmod using only the sign function and absolute value function we can implement smoothed limiters.

$\text{sign}(r) = \tanh(5r)$





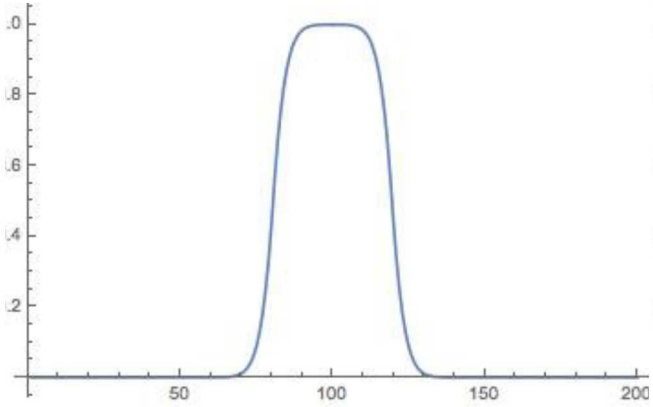
$\text{sign}(r) = \tanh(50r)$



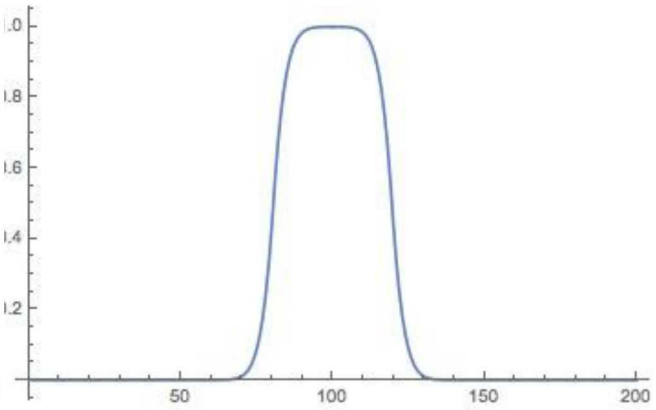
Sigmoids can do this in many many ways

Results using the smoothed intrinsic limiters

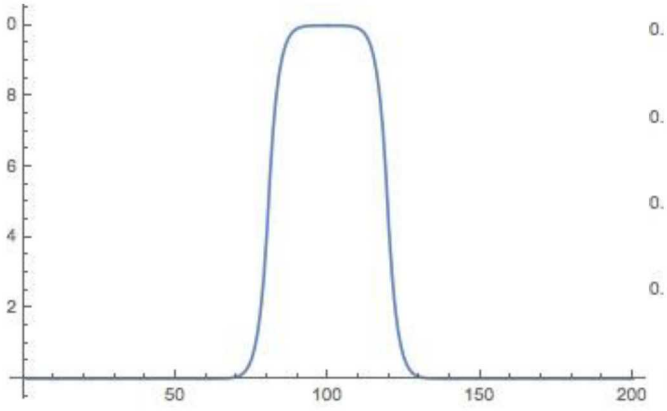
The results are really good actually!



Error= 0.0314099 0.00649721 0.438233



Error= 0.0313124 0.00648103 0.4384



Error= 0.0257311 0.00578931 0.420996

Normal functions $\quad \operatorname{sign}(r) = \tanh(10r)$ $\quad \operatorname{sign}(r) = \dfrac{x}{|x| + 0.1}$

$$\operatorname{abs}(r) = |x| + 0.1$$

Using the sigmoids to discretize

Originally used as interface tracking with the THINC scheme

- A sigmoid is a monotone basis by construction

$$u_j(x) = \frac{1}{2}\left(u_{j-1} + u_{j+1}\right) + \frac{1}{2}\left(u_{j+1} - u_{j-1}\right)\tanh\left[\beta\left(x - x_c\right)\right]$$

- We can use other sigmoids in place of the tanh used in THINC

$$u_j(x) = \frac{1}{2}\left(u_{j-1} + u_{j+1}\right) + \frac{1}{2}\left(u_{j+1} - u_{j-1}\right)\text{sigmoid}\left[\beta\left(x - x_c\right)\right]$$

- The key is to solve for the centering point, x_c such that the basis is mean-preserving.

$$u_j = \int_{x_{j-1/2}}^{x_{j+1/2}} u_j(x)dx$$

Other sigmoids can be used and potentially profitably. These methods may be extremely important.

The THINC method is extremely similar to ACM

Steepeners and artificial compression can be used to track interfaces sharply

- In classical limiter-based methods, THINC is most analogous to anti-diffusive limiters (ultra- and superbee), the artificial compression method or contract steepeners in PPM
- THINC (or sigmoid-based methods) have the advantage of a monotone basis by construction, but a lack of means-preservation without solving a nonlinear equation to enforce it.
- We know that anti-diffusive methods recover first-order convergence at material interfaces, and we can assume that sigmoid based methods will do the same.

We have a new and entirely different approach to getter sharp, but differentiable representations

We can look at a series of special cases for analysis

We can understand the limits of sigmoids through these cases

- If we examine the sigmoid function for a couple of simple cases the analysis is straightforward
- A bounded linear function is monotone, and the Heaviside function is basically identical to using the SLIC version of a VOF method

$$u_j = \frac{1}{\Delta x} \int_{x_{j-1/2}}^{x_{j+1/2}} \frac{1}{2}\left(u_{j-1} + u_{j+1}\right) + \frac{1}{2}\left(u_{j+1} - u_{j-1}\right)\left(x - x_c\right) dx$$

$$x_c = \frac{u_j - \frac{1}{2}\left(u_{j-1} + u_{j+1}\right)}{\frac{1}{2}\left(u_{j+1} - u_{j-1}\right)} \approx \frac{1}{2}\frac{\partial_{xx} u}{\partial_x u}$$

We have rendered these nonlinear functions, linear in these limiting cases

The Heaviside function integration is similar

These two cases give us some bounding information and connections

- The Heaviside function is the sign function or a sigmoid with a very large value for β

$$u_j(x) = \frac{1}{2}\left(u_{j-1} + u_{j+1}\right) + \frac{1}{2}\left(u_{j+1} + u_{j-1}\right)\mathrm{H}(x - x_c)$$

- Solve for the integral $$x_c = \frac{u_j - \frac{1}{2}\left(u_{j-1} + u_{j+1}\right)}{\frac{1}{2}\left(u_{j+1} - u_{j-1}\right)} \approx \frac{1}{2}\frac{\partial_{xx} u}{\partial_x u}$$

- It is exactly the same. So we see a basic pattern. This result looks almost identical for the modified equation for the limited downwind scheme, which is a close to the SLIC method (and a simple version)

We see a gratifying connection between sigmoid discretization and anti-dissipative methods

Models and submodels have switches too

Constitutive models, multimaterial closure, turbulence modeling

- A more general issue is the floating point switches used in modeling
- Examples are multimaterial closures, turbulence models and strength modeling (weather and climate modeling is replete with examples)
- Naïve implementations would use copious if-then-else constructs introducing discontinuous behavior into the codes
- This behavior would wreck havoc on nonlinear solvers, symmetry preservation and regression testing.

These switches usually are not known so well as to justify the sharpness of changes

Other benefits of sigmoids are regression testing

This can be very time consuming if small variations change results substantially

- If sharp transitions in a code's behavior are removed then small order "epsilon" differences in floating points can be more easily tolerated.
- We demonstrated this on the multimaterial closure in Alegra where we replaced if-then-else constructs with sigmoidal switches (Tanh based)
- This principle can be applied far more generally where floating point logic is used.
  - –Limiters and upwinding are the two most obvious examples

Get rid of if-then-else or sharp functions for better stability of regression tests

Summary and conclusions

Sigmoids are used pervasively, but probably not enough

- Adaptive methods are essential – upwinding, limiters, models all change based on local conditions
- The standard implementation is "if-then-else" but functional representation is useful for analysis and understanding
- These implementations are sharp and allow small variations to produce large changes. This is potentially extremely problematic
- Sigmoids are a class of smooth functions that can replace these sharp functions with a highly controlled transition width.
- Sigmoids can be used to smooth adaptive methods, produce new ones

Hopefully this can spur more of you to examine this interesting implementation idea