Simulation of Nondifferentiable Models for Groundwater Flow and Transport

C. T. Kelley, K. R. Fowler, C. E. Kees

Department of Mathematics
Center for Research in Scientific Computation
North Carolina State University
Raleigh, North Carolina, USA
CMWR04, University of North Carolina
Chapel Hill, NC
June 14, 2004
Outline

- Nonsmooth models
  - Richards’ equation:
    - van Genuchten/Mualem formulae
  - Reactive transport
    - Freundlich isotherm
- What solvers must do
  - ODE/DAE formulations
  - Nonsmooth calculus and ADH
  - Temporal error estimation and control (time?)
- Conclusions
Collaborators

- ERDC: Stacy Howington, Charlie Berger, Jackie Hallberg
- NCSU: Jill Reese
- UNC: Casey Miller, Matthew Farthing, Joe Kanney
- Clemson: Lea Jenkins
- Mathworks: Mike Tocci
- Old Dominion: Glenn Williams
Richards’ Equation: pressure head form

\[ S_s S_a(\psi) \frac{\partial \psi}{\partial t} + \eta \frac{\partial S_a(\psi)}{\partial t} = \nabla \cdot [K(\psi)\nabla(z + \psi)] \]

- \(\psi\): pressure head
- \(S_s\): specific storage
- \(S_a(\psi)\): saturation
- \(\eta\): porosity
- \(K(\psi)\): hydraulic conductivity

C. T. Kelley – p.4
van Genuchten and Mualem formulae

\[
S_a(\psi) = \begin{cases} 
S_r + \frac{(1-S_r)}{[1+(\alpha|\psi|)^n]^m}, & \psi < 0 \\
1, & \psi \geq 0
\end{cases}
\]

\[
K(\psi) = \begin{cases} 
K_s \frac{[1-(\alpha|\psi|)^{n-1}[1+(\alpha|\psi|)^n]^{-m}]^2}{[1+(\alpha|\psi|)^n]^{m/2}}, & \psi < 0 \\
K_s, & \psi \geq 0
\end{cases}
\]

- \(S_r\): residual saturation
- \(\alpha\): coefficient for mean pore size
- \(K_s\): saturated hydraulic conductivity
- \(n\): measure of pore size uniformity; \(m = 1 - 1/n\)
Non-smooth nonlinearities

- $K$ is not Lipschitz continuous if $1 < n < 2$
- non-Lipschitz $K$ causes many problems
Non-smooth nonlinearities

- $K$ is not Lipschitz continuous if $1 < n < 2$
- non-Lipschitz $K$ causes many problems
  - Nonlinear solvers in implicit temporal integration fail
  - Bizarre nonphysical effects
  See Chris Kees’ poster
Non-smooth nonlinearities

- $K$ is not Lipschitz continuous if $1 < n < 2$
- non-Lipschitz $K$ causes many problems
  - Nonlinear solvers in implicit temporal integration fail
  - Bizarre nonphysical effects
    See Chris Kees’ poster
- Fix: interpolate (or fit data) with a spline
  - Speeds up the simulation
  - Makes the nonlinearity smooth or at least Lipschitz continuous
Non-smooth nonlinearities

- $K$ is not Lipschitz continuous if $1 < n < 2$
- non-Lipschitz $K$ causes many problems
  - Nonlinear solvers in implicit temporal integration fail
  - Bizarre nonphysical effects
    See Chris Kees’ poster
- Fix: interpolate (or fit data) with a spline
  - Speeds up the simulation
  - Makes the nonlinearity smooth or at least Lipschitz continuous
- ERDC ADH code uses PL splines
  Lipschitz continuous/not differentiable
Reactive Transport in Porous Media

Freundlich isotherm:

\[
\frac{C_s \eta}{\rho_b} = K_{\text{max}}(C, 0)^r
\]

Transport equation:

\[
(C + \frac{\rho_b}{\eta} K_{\text{max}}(C, 0)^r)_t + \nabla \cdot [Cv - D \nabla C] = 0
\]

---

- \(C_s\): equilibrium concentration in the solid phase
- \(r\): Freundlich exponent
- \(C\): Freundlich coefficient
- \(\rho_b\): bulk density of the solid phase
- \(\eta\): porosity
- \(v\): mean pore velocity
- \(D\): hydrodynamic dispersion tensor

C. T. Kelley – p.7
Nonsmoothness and a Fix

Nonlinearity is not Lipschitz continuous if \( 0 < r < 1 \).
Fix: Differential Algebraic Equation (DAE) formulation

Differential equation:

\[
m_t + \nabla \cdot [Cv - D \nabla C] = 0.
\]

Algebraic constraint:

\[
\left( \frac{\eta \max(m - C, 0)}{\rho_b K} \right)^{1/r} - C = 0.
\]

And now everything is differentiable,
Nonsmoothness and a Fix

Nonlinearity is not Lipschitz continuous if \(0 < r < 1\).
Fix: Differential Algebraic Equation (DAE) formulation
Differential equation:

\[
m_t + \nabla \cdot [Cv - D \nabla C] = 0.
\]

Algebraic constraint:

\[
\left( \frac{\eta \max(m - C, 0)}{\rho_b K} \right)^{1/r} - C = 0.
\]

And now everything is differentiable, but I’ve added an equation.
DAE and ODE Dynamics

\[ u_t = f(t, u), \quad u(0) = u_0 \quad \text{ODE} \]

\[ f(t, u, u_t) = 0, \quad u(0) = u_0, u'(0) = u'_0 \quad \text{DAE} \]

We have at most \text{Index-one} DAEs here. i.e. \text{Implicit Euler works}.
Initial data for \( u'(0) \) is the solver’s job.
What DAEs can do for you.

- Make temporal integration work better (Richards)
- Hide nonsmooth physics (Freundlich)
What DAEs can do for you.

- Make temporal integration work better (Richards)
- Hide nonsmooth physics (Freundlich)

It’s still your job to design good solvers

- regularity of the solution
- differentiability of the nonlinearity
- discretizations
- linear solvers and preconditioning
DAE formulation of Reactive Transport Equation

Two equations for $m$ and $C$

$$m_t = -\nabla \cdot [Cv - D\nabla C] = f(m, C) \quad \text{Differential Equation}$$

and

$$\left( \frac{\eta \max(m - C, 0)}{\rho_b K} \right)^{1/r} - C = g(m, C) = 0 \quad \text{Algebraic Constraint}$$

There’s no $C_t$ anywhere.
Discretize in space, and advance in time by solving

\[ m^{n+1} = m^n + h f(m^{n+1}, C^{n+1}), \]

\[ g(m^{n+1}, C^{n+1}) = 0. \]

So the equation is for \( u = (m, C)^T \).
Newton’s method

Solve

\[ F(u) = 0 \]

by

\[ u^+ = u^c + s, \quad F'(u_c)s = -F(u_c) \]
Newton’s method

Solve

\[ F(u) = 0 \]

by

\[ u^+ = u^c + s, \quad F'(u_c)s = -F(u_c) \]

Solve for \( F'(u_c)s = -F(u_c) \) for the step by

- Gaussian elimination (compute and factor matrix)
- Iterative method with computed (approximate) Jacobian
- Matrix free: iterative method, finite difference Jacobian-vector products
Newton’s method

Solve

\[ F(u) = 0 \]

by

\[ u^+ = u^c + s, \quad F'(u_c)s = -F(u_c) \]

Solve for \( F'(u_c)s = -F(u_c) \) for the step by

- Gaussian elimination (compute and factor matrix)
- Iterative method with computed (approximate) Jacobian
- Matrix free: iterative method, finite difference Jacobian-vector products

Everything works if \( F'(u^*) \) is nonsingular.
What do you feed the solver?

\[
F \begin{pmatrix} m \\ C \end{pmatrix} = \begin{pmatrix} m - m^n - hf(m, C) \\ g(m, C) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}
\]

Solve with Newton. Converged result is \((m^{n+1}, C^{n+1})^T\).
What do you feed the solver?

\[
F \left( \begin{array}{c} m \\ C \end{array} \right) = \left( \begin{array}{c} m - m^n - hf(m,C) \\ g(m,C) \end{array} \right) = \left( \begin{array}{c} 0 \\ 0 \end{array} \right)
\]

Solve with Newton. Converged result is \((m^{n+1}, C^{n+1})^T\).

For small \(h\),

\[
F' = \left( \begin{array}{cc} I - hf_m & -hf_C \\ g_m & g_C \end{array} \right)
\]

is nonsingular if \(g_C\) is nonsingular (aka index one).
Is RE a DAE?

Discretize in space, and you have

\[ S_s S_a(\psi) \frac{\partial \psi}{\partial t} + \eta \frac{\partial S_a(\psi)}{\partial t} = N(\psi) \]

ODE solve: Use the chain rule and get

\[ \frac{\partial \psi}{\partial t} = \frac{N(\psi)}{S_s S_a(\psi) + \eta S'_a(\psi)}, \]

so implicit Euler is . . .
Implicit Euler for ODE formulation of RE

\[ \psi^{n+1} = \psi^n + h \frac{N(\psi^{n+1})}{S_s S_a(\psi^{n+1}) + \eta S'_a(\psi^{n+1})}, \]

performs poorly:
Implicit Euler for ODE formulation of RE

\[ \psi^{n+1} = \psi^n + h \frac{N(\psi^{n+1})}{S_s S_a(\psi^{n+1}) + \eta S'_a(\psi^{n+1})}, \]

performs poorly:

- small denominator,
Implicit Euler for ODE formulation of RE

\[ \psi^{n+1} = \psi^n + h \frac{N(\psi^{n+1})}{S_s S_a(\psi^{n+1}) + \eta S_a'(\psi^{n+1})}, \]

performs poorly:

- small denominator,
- small denominator is squared for the Jacobian,
Implicit Euler for ODE formulation of RE

\[ \psi^{n+1} = \psi^n + h \frac{N(\psi^{n+1})}{S_s S_a(\psi^{n+1}) + \eta S'_a(\psi^{n+1})}, \]

performs poorly:

- small denominator,
- small denominator is squared for the Jacobian,
- leading to many solver failures, which
Implicit Euler for ODE formulation of RE

\[ \psi^{n+1} = \psi^n + h \frac{N(\psi^{n+1})}{S_s S_a(\psi^{n+1}) + \eta S'_a(\psi^{n+1})}, \]

performs poorly:

- small denominator,
- \text{small denominator} is squared for the Jacobian,
- leading to many solver failures, which
- result in very small timesteps.
DAE formulation

\[ S_s S_a (\psi^{n+1}) (\psi^{n+1} - \psi^n) + \eta (S_a (\psi^{n+1}) - S_a (\psi^n)) = h N (\psi^{n+1}). \]

This is a lot better,

- larger time steps,
- happier nonlinear solver,
- error control easier to understand, and
- what most folks do.
ERDC ADH Code

• approximate VG-Mualem formulae with PL splines
• We explain the success of
  • finite difference approximation of Jacobians
  • Newton’s method for implicit time-stepping
  • first order error estimation and control
ADH temporal integration

Solve

\[ F(u) = S_s S_a(u)(u - \psi^n) + \eta (S_a(u) - S_a(\psi^n)) - hN(u) = 0, \]

with Newton’s method.
ADH temporal integration

Solve

\[ F(u) = S_s S_a(u)(u - \psi^n) + \eta (S_a(u) - S_a(\psi^n)) - hN(u) = 0, \]

with Newton’s method.
Approximate \( F'(u) \) by a finite difference Jacobian \( \partial_h F(u) \)

\[ u_+ = u_c - (\partial_h F(u_c))^{-1} F(u_c), \]

and you get good results. Why?
Nonsmooth Calculus

$F \in LIP$ implies $F$ differentiable a.e.
The generalized Jacobian (Clarke) at $u$ is

$$\partial F(u) = \text{co} \left\{ \lim_{u_j \to u; u_j \in D_F} F'(u_j) \right\}$$
Nonsmooth Calculus

\( F \in \text{LIP} \) implies \( F \) differentiable a.e.
The generalized Jacobian (Clarke) at \( u \) is

\[
\partial F(u) = \text{co} \left\{ \lim_{u_j \to u; u_j \in D_F} F'(u_j) \right\}
\]

You’d like to replace Newton’s method with

\[
u_{n+1} = u_n - V_n^{-1}F(u_n)
\]

where \( V_n \in \partial F(u_n) \).
Nonsmooth Calculus

\( F \in \text{LIP} \) implies \( F \) differentiable a.e.
The generalized Jacobian (Clarke) at \( u \) is

\[
\partial F(u) = \text{co} \left\{ \lim_{u_j \to u; u_j \in D_F} F'(u_j) \right\}
\]

You’d like to replace Newton’s method with

\[
u_{n+1} = u_n - V_n^{-1} F(u_n)\]

where \( V_n \in \partial F(u_n) \).
How do you compute \( V_n \)?
Can you use this stuff in the real world?
Piecewise smooth function: $\phi = \phi_l + \phi_r$

\[ \partial \phi(0) = [\phi'_l(0), \phi'_r(0)], \text{ a SET.} \]
Difference approximations

Scalar functions

\[ \partial_h \phi(u) = \frac{\phi(u + h) - \phi(u)}{h} \]

For Lipschitz functions:

\[ \partial_h \phi(u) \in \partial \phi(\bar{u}) + O(h) \]

where \( |u - \bar{u}| \leq h \).

Same story for scalar constitutive laws in PDEs.

If you differentiate in coordinate directions!
Difference approximation accuracy

\[ \phi'_l(0) + O(h) \leq \partial_h \phi(u) \leq \phi'_r(0) + O(h), \text{ so } \partial_h \phi(u) \in \partial \phi(0) + O(h) \]
A Lipschitz function $F$ is semismooth (Mifflin, Pang, Qi) if

$$\lim_{w \to 0, V \in \partial F(u+w)} \frac{\|F(u+w) - F(u) - Vw\|}{\|w\|} = 0.$$ 

and semismooth of order 1 at $u$ if

$$F(u + w) - F(u) - Vw = O(\|w\|^2)$$

for all $w \in \mathbb{R}^N$ and $V \in \partial F(u+w)$ as $w \to 0$.

What you need for local convergence of Newton’s method. Piecewise smooth functions are semismooth of order 1.
Why semismoothness?

If

- $F$ semismooth of order $1$,
- $F(u^*) = 0$, and
- everything in $\partial F(u^*)$ uniformly nonsingular,
- $u_c$ near $u^*$,

then if

$$u_+ = u_c - V^{-1}F(u_c),$$
where $V \in \partial F(u_c)$,
Why semismoothness?

If

- $F$ semismooth of order 1,
- $F(u^*) = 0$, and
- everything in $\partial F(u^*)$ uniformly nonsingular,
- $u_c$ near $u^*$,

then if

$$u_+ = u_c - V^{-1}F(u_c), \text{ where } V \in \partial F(u_c),$$

you get fast local convergence

$$\|u_+ - u^*\| = O(\|u_c - u^*\|^2).$$
Convergence Proof, \( e = u - u^* \)

Semismoothness \((u \leftarrow u^*, w \leftarrow e_c, u + w \leftarrow u_c)\) implies

\[
F(u_c) - Ve_c = O(\|e_c\|^2)
\]
Convergence Proof, $e = u - u^*$

Semismoothness ($u \leftarrow u^*, w \leftarrow e_c, u + w \leftarrow u_c$) implies

$$F(u_c) - V e_c = O(\|e_c\|^2)$$

Subtract $u^*$ from both sides of

$$u_+ = u_c - V^{-1}F(u_c),$$
Convergence Proof, $e = u - u^*$

Semismoothness $(u \leftarrow u^*, w \leftarrow e_c, u + w \leftarrow u_c)$ implies

$$F(u_c) - Ve_c = O(\|e_c\|^2)$$

Subtract $u^*$ from both sides of

$$u_+ = u_c - V^{-1}F(u_c),$$

to get

$$e_+ = e_c - V^{-1}F(u_c) = e_c - e_c + O(\|e_c\|^2) = O(\|e_c\|^2).$$
So what’s up with ADH?

\[ u_+ = u_c - (\partial_h F(u_c))^{-1} F(u_c) \]

and

\[ \partial_h F(u_c) \in \partial F(\tilde{u}) + O(h) \]
So what’s up with ADH?

\[ u_+ = u_c - ( \partial_h F(u_c) )^{-1} F(u_c) \]

and

\[ \partial_h F(u_c) \in \partial F(\tilde{u}) + O(h) \]

which implies

\[ e_+ = O( \| e_c \|^2 + \| e_c \| h + h ). \]

Looks just like Newton if \( \| e_c \| \gg \sqrt{h} \).
Iterative Linear Solvers

ADH uses preconditioned Krylov linear solvers. Termination on small relative linear residual,

$$\| F(u_c) + \partial_h F(u_c)s \| \leq \eta_c \| F(u_c) \| .$$

Convergence,

$$e_+ = O(\| e_c \|^2 + \| e_c \| (\eta_c + h) + h).$$
Iterative Linear Solvers

ADH uses preconditioned Krylov linear solvers. Termination on small relative linear residual,

$$\| F(u_c) + \partial_h F(u_c)s \| \leq \eta_c \| F(u_c) \|.$$  

Convergence,

$$e_+ = O(\| e_c \|^2 + \| e_c \| (\eta_c + h) + h).$$

Tradeoffs:

- Keep $\eta$ small (accurate Newton step), for nonlinear performance,
- but not too small, to minimize linear solver cost.
Optimal difference increment

$\varepsilon_F$: error in evaluation (eg floating point roundoff)
Include this in $V$ to get

$$V(u) \in \partial F(\bar{u}) + O(h + \varepsilon_F/h)$$
Optimal difference increment

\( \epsilon_F \): error in evaluation (eg floating point roundoff)
Include this in \( V \) to get

\[
V(u) \in \partial F(\bar{u}) + O(h + \epsilon_F/h)
\]

So, if \( \|e_n\| = \sqrt{h} \), then

\[
e_{n+1} = O\left((h + \epsilon_F/h)\|e_n\| + \|e_n\|^2 + h\right) = O\left(\frac{\epsilon_F}{h^{1/2}} + h\right)
\]
Optimal difference increment

\( \varepsilon_F \): error in evaluation (eg floating point roundoff)
Include this in \( V \) to get

\[
V(u) \in \partial F(\bar{u}) + O(h + \varepsilon_F/h)
\]

So, if \( \|e_n\| = \sqrt{h} \), then

\[
e_{n+1} = O((h + \varepsilon_F/h)\|e_n\| + \|e_n\|^2 + h)
\]

\[
= O\left(\frac{\varepsilon_F}{h^{1/2}} + h\right)
\]

which is minimized if \( h = O(\varepsilon_F^{2/3}) \approx 10^{-10} \) in IEEE.
Temporal Error Estimation and Control

Process: for $u' = F(u)$, $F$ Lipschitz continuous
Goal: local truncation error $< \tau$.

- Begin with $u^n$ and $u^{n-1}$,
Temporal Error Estimation and Control

Process: for $u' = F(u)$, $F$ Lipschitz continuous
Goal: local truncation error $< \tau$.

- Begin with $u^n$ and $u^{n-1}$,
- Let $u^p$ be a linear predictor to $u^{n+1}$,

$$u^p = u^n + h_n \frac{u^n - u^{n-1}}{h_{n-1}}$$
Temporal Error Estimation and Control

Process: for $u' = F(u)$, $F$ Lipschitz continuous
Goal: local truncation error $< \tau$.

• Begin with $u^n$ and $u^{n-1}$,
• let $u^p$ be a linear predictor to $u^{n+1}$,

$$u^p = u^n + h_n \frac{u^n - u^{n-1}}{h_{n-1}}$$

• Compute implicit Euler step, $u^{n+1}$ by solving

$$u^{n+1} = u^n + h_n F(u_{n+1})$$
Temporal Error Estimation and Control

Process: for $u' = F(u)$, $F$ Lipschitz continuous
Goal: local truncation error $< \tau$.

- Begin with $u^n$ and $u^{n-1}$,
- let $u^p$ be a linear predictor to $u^{n+1}$,

$$u^p = u^n + h_n \frac{u^n - u^{n-1}}{h_{n-1}}$$

- Compute implicit Euler step, $u^{n+1}$ by solving

$$u^{n+1} = u^n + h_n F(u_{n+1})$$

- Compare $u^{n+1}$ and $u^p$ to estimate error and change step size.
Details, details, details

- Estimate Lipschitz constant of $u'$ by

$$L = 2\|u^{n+1} - u^p\|/|2h_n^2 - h_nh_{n-1}|$$
Details, details, details

• Estimate Lipschitz constant of $u'$ by

$$L = 2\|u^{n+1} - u^p\|/|2h_n^2 - h_nh_{n-1}|$$

• Estimated local truncation error is $Lh_n^2/2$. 
Details, details, details

- Estimate Lipschitz constant of $u'$ by

$$L = 2\|u^{n+1} - u^p\|/|2h_n^2 - h_nh_{n-1}|$$

- Estimated local truncation error is $Lh_n^2/2$.
  - $> \tau$? Enforce $Lh_n^2/2 < .9\tau$, try again.
Details, details, details

- Estimate Lipschitz constant of $u'$ by

$$L = 2 \| u^{n+1} - u^p \| / |2h_n^2 - h_nh_{n-1}|$$

- Estimated local truncation error is $Lh_n^2/2$.
  - $> \tau$? Enforce $Lh_n^2/2 < .9\tau$, try again.
  - Too many nonlinear iterations, reduce $h_n$, try again.
- This was the problem in ODE form of RE!
Details, details, details

- Estimate Lipschitz constant of $u'$ by

$$L = 2\|u^{n+1} - u^p\|/|2h^2_n - h_nh_{n-1}|$$

- Estimated local truncation error is $Lh^{2n}/2$.
  - $> \tau$? Enforce $Lh^{2n}/2 < .9\tau$, try again.
  - Too many nonlinear iterations, reduce $h_n$, try again. This was the problem in ODE form of RE!
  - $h_n$ ok? Enforce $Lh^{2n+1}/2 < .9\tau$
Details, details, details

- Estimate Lipschitz constant of $u'$ by
  $$L = 2||u^{n+1} - u^p|| / |2h_n^2 - h_n h_{n-1}|$$

- Estimated local truncation error is $Lh_n^2/2$.
  - $> \tau$? Enforce $Lh_n^2/2 < .9\tau$, try again.
  - Too many nonlinear iterations, reduce $h_n$, try again. This was the problem in ODE form of RE!
  - $h_n$ ok? Enforce $Lh_{n+1}^2/2 < .9\tau$

- Completely rigorous if we’re getting the Lipschitz constant right.
Numerical Experiments

Compare

\[ L_{n+1} = 2\|u_{n+1} - u^p\|/|2h_n^2 - h_nh_{n-1}| \]

with

\[ L(u_{n+1}) \frac{\|F(u_{n+1}) - F(u^n)\|}{t_{n+1} - t_n} \]

You want to see

\[ r_n = \frac{L_n}{L(u^n)} \geq 1. \]

Study, RE for two media with \( 1 < n < 2. \)
## Media Properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>clay</th>
<th>silt</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>1.09</td>
<td>1.37</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>0.244</td>
<td>0.478</td>
</tr>
<tr>
<td>( S_r )</td>
<td>0.179</td>
<td>0.074</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.33</td>
<td>0.40</td>
</tr>
<tr>
<td>( K_s )</td>
<td>1.10808e-5</td>
<td>1.1801e-03</td>
</tr>
<tr>
<td>( T_{final} )</td>
<td>600 days</td>
<td>150 days</td>
</tr>
<tr>
<td>( maxh )</td>
<td>10 days</td>
<td>5 days</td>
</tr>
</tbody>
</table>

\( \tau = 10^{-2}, h_0 = 10^{-9} \)
Clay

\[ r = \frac{L_n}{L(u_n)} \]

\[ r = 1 \]
Silt

\[ r = \frac{L}{L(u_n)} \]

\[ r = 1 \]
Conclusions

- Nonsmooth nonlinearities are not a disaster
Conclusions

- Nonsmooth nonlinearities are not a disaster
  - You can finesse them (Reactive Transport).
Conclusions

• Nonsmooth nonlinearities are not a disaster
  • You can finesse them (Reactive Transport).
  • You can confront them directly (RE).
Conclusions

- Nonsmooth nonlinearities are not a disaster
  - You can finesse them (Reactive Transport).
  - You can confront them directly (RE).
- Semismoothness is all you need for Newton’s method.
Conclusions

- Nonsmooth nonlinearities are not a disaster
  - You can finesse them (Reactive Transport).
  - You can confront them directly (RE).
- Semismoothness is all you need for Newton’s method.
  - Exotic math; software needn’t know about it.
Conclusions

- Nonsmooth nonlinearities are not a disaster
  - You can finesse them (Reactive Transport).
  - You can confront them directly (RE).
- Semismoothness is all you need for Newton’s method.
  - Exotic math; software needn’t know about it.
  - Solvers work, so error control also works.
Conclusions

- Nonsmooth nonlinearities are not a disaster
  - You can finesse them (Reactive Transport).
  - You can confront them directly (RE).
- Semismoothness is all you need for Newton’s method.
  - Exotic math; software needn’t know about it.
  - Solvers work, so error control also works.
  - Implemented and working in ADH.
Conclusions

- Nonsmooth nonlinearities are not a disaster
  - You can finesse them (Reactive Transport).
  - You can confront them directly (RE).
- Semismoothness is all you need for Newton’s method.
  - Exotic math; software needn’t know about it.
  - Solvers work, so error control also works.
  - Implemented and working in ADH.
- High-order methods in time seem to work. Why?