Nonlinear Equation Solvers and Circuit Simulations

Challenges and Opportunities

Tamara Kolda and Roger Pawlowski
Sandia National Labs
`{tgkolda,rppawlo}@sandia.gov`

Objectives

- Nonlinear equations in circuit simulation
- Methods for solving nonlinear equations
- The NOX software package
- Preliminary numerical results Xyce/NOX
- Conclusions

Additional Thanks

- NOX Team and Users
  - Russ Hooper
- Xyce Development Team
  - Eric Keiter
  - Scott Hutchinson
  - David Day
  - Tom Russo
  - Rob Hoekstra

Circuit Simulation

- Netlist
  - Resistors, Capacitors, Inductors, Diodes, etc.
- Create equations via Kirchoff's Laws
  - KVL - Sum of voltage drops around each closed loop is zero
  - KCL - Zero net current entering each node
- System of Differentiable Algebraic Equations (DAEs)

DAEs

- DC Operating Point - Simulating DC response in an analog circuit (often used to get consistent initial conditions for the transient analysis)
  - Nonlinear equations are hard to solve.
- Transient Analysis - Simulating the response of a circuit over time
  - Nonlinear equations are easy to solve (relatively speaking)

Nonlinear Equations

- Nonlinear system of equations:

$$F(x) = \begin{bmatrix} F_1(x) \\ \vdots \\ F_n(x) \end{bmatrix} = 0$$

- Jacobian:

$$J(x)_{ij} = \frac{\partial F_i}{\partial x_j}(x) \in \mathbb{R}^{n \times n}$$

Newton's Method

- Linear Model at Iteration k: $M_k(s) \approx F(x_k + s)$

$$M_k(s) = F(x_k) + J(x_k)s$$

- Newton Equation:

$$J(x_k)s_k = -F(x_k)$$

- Iteration:

$$x_{k+1} = x_k - \underbrace{J(x_k)^{-1}F(x_k)}_{s_k}$$

Pros and Cons of Newton's Method

+ Locally Q-quadratically convergent
  (if the Jacobian is nonsingular at solution)
+ Solves linear problems in one iteration
- Requires Jacobian calculation at each iteration
- Requires solution of Newton equation at each iteration, and we get into trouble if the Jacobian is singular or ill-conditioned.
- Not globally convergent

[Dennis & Schnabel 1983]

Solving the Newton Equation

$$J\,s = -F$$

- Sparse Direct Method
- Iterative Method (aka inexact Newton)
  - Solve to fixed tolerance
  - Forcing - Vary the convergence tolerance on the linear solve depending on the predictive power of the linear model *[Eisenstat and Walker, SISC, 1996]*

Pros and cons for either approach.

Globalization of Newton's Method

- Convert to minimization problem:

$$\min f(x) = \frac{1}{2}\|F(x)\|_2^2$$

- Quadratic Model at Iteration k:

$$m_k(s) = \underbrace{\frac{1}{2}\|F(x_k)\|_2^2}_{f(x_k)} + \underbrace{F(x_k)^T J(x_k)}_{\nabla f(x_k)^T} s + s^T \underbrace{J(x_k)^T J(x_k)}_{B_k} s$$

- Globalization options: line search, trust region, continuation

Line Search Methods

- aka Damped Newton
- Want to insure $f(x_k)$ strictly decreasing
- Iteration:

$$x_{k+1} = x_k - \underbrace{\lambda_k}_{\text{Step Length}} \underbrace{J(x_k)^{-1}F(x_k)}_{\text{Newton Step}}$$

- Two method we use are:
  - Backtracking (for max-norm decrease)
  - Moré-Thuente line search

Trust Region

- Define a region in which we "trust" the model:

$$\min m_k(s) \quad \text{subject to} \quad \|s\|_2 \leq \delta_k$$

- Iteration:

$$x_{k+1} = x_k + \underbrace{s_k}_{\text{Trust Region Step}}$$

- To approximately solve the subproblem:
    - Dogleg method

NOX - Nonlinear Equation Solver

- Object-oriented software package in C++
- Development began in earnest last October
- Already being integrated with a variety of codes for different applications
  - Applications – Solid Mechanics, Electrical Circuits, Radidation Transport, Reacting Flows
  - Sandia Codes – Xyce, SIERRA (Adagio, Premo, Goma/Aria), ALEGRA (Nevada), MPSalsa, FEAP, Tahoe*

Interfacing NOX

- Abstract interface to vector representation
  - update (daxpy), norm, dot, abs, reciprocal, scale, clone, copy, length
  - currently supports Epetra/Aztec, PETSc
- Abstract interface to problem/linear solver
  - compute and access functionality for solution, residual, Jacobian matrix and Newton direction; applyJacobian; clone; copy
  - easy interfaces, as for Xyce and other Sandia codes

NOX Methods

- Methods
  - Line search methods
    - Direction - Newton, Cauchy, LM-Broyden [cf. Kelley, SIAM, 1995], Tensor [Schnabel and Frank, SINUM, '84]
    - Line Search - None, Backtracking, Moré-Thuente, Polynomial
  - Trust-region methods
    - Double Dogleg
  - Nonlinear CG
  - Continuation (via LOCA)

More on NOX

- Easy to switch between methods
- Many different stopping criteria including the ability to add custom stopping conditions
  - For Xyce, we have a custom convergence test using a weighted norm based on the *previous* time step (in transient mode)
- Robustness??
  - For both line search and trust region methods, we use a “recovery step” in the Newton-like direction whenever the subproblem fails.

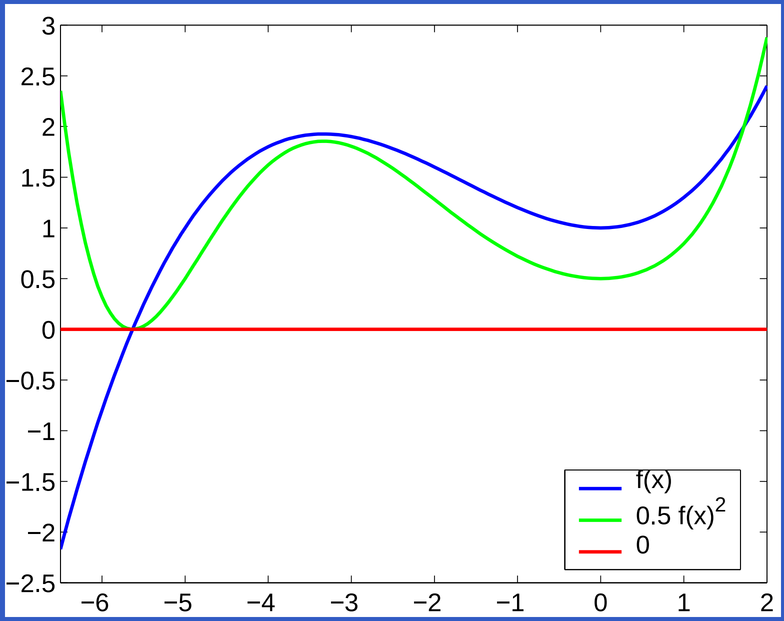
Preliminary Numerical Results

Comparison of Number of Newton Solves

Circuit	Newton			Line Search			Trust Region		
	Iter	For	Lim	Iter	For	Lim	Iter	For	Lim
latch	F	F	11	31	22	F	20	20	14
comparator	43	26	21	155	183	24	54	29	25
rca	F	F	7	15	13	7	24	20	7
schmitecl	F	F	31	8	10	33	12	12	F
nand_bjt	F	F	12	13	13	F	8	17	F
vref	F	F	16	30	18	18	F	F	23
diode	F	F	6	4	12	6	9	10	6
npn1	F	F	6	10	9	6	9	13	6
pnp1	F	F	7	10	14	7	14	20	7

Difficulties

- Even 1-D problems are hard
- Globalization
  - Global vs. local “minimums”
- Ill-conditioning in Jacobian
- Sensitivity to solver parameters



We have a lot more to do. . .

- Improving existing methods
- Parametric studies on solver options
- Adding Tensor and LM-Broyden's methods as well as Continuation
- More sophisticated approaches to voltage limiting

Resources

- Xyce (Circuit Simulator)
  - Scott Hutchinson, sahutch@sandia.gov
- NOX (Nonlinear Solver)
  - Tamara Kolda, tgkolda@sandia.gov
  - Roger Pawlowski, rppawlo@sandia.gov
- Epetra (Linear Algebra)
  - Mike Heroux, maherou@sandia.gov
- Trilinos (includes Epetra, NOX, etc)
  - Mike Heroux, maherou@sandia.gov