

Tensor-Krylov Methods for Solving Systems of Nonlinear Equations: An Introduction and Comparison

Brett W. Bader

Department of Computer Science
University of Colorado at Boulder

2003 Workshop: Solution Methods for Large-Scale Nonlinear Problems

August 6, 2003

Acknowledgements



- Robert Schnabel (thesis advisor)

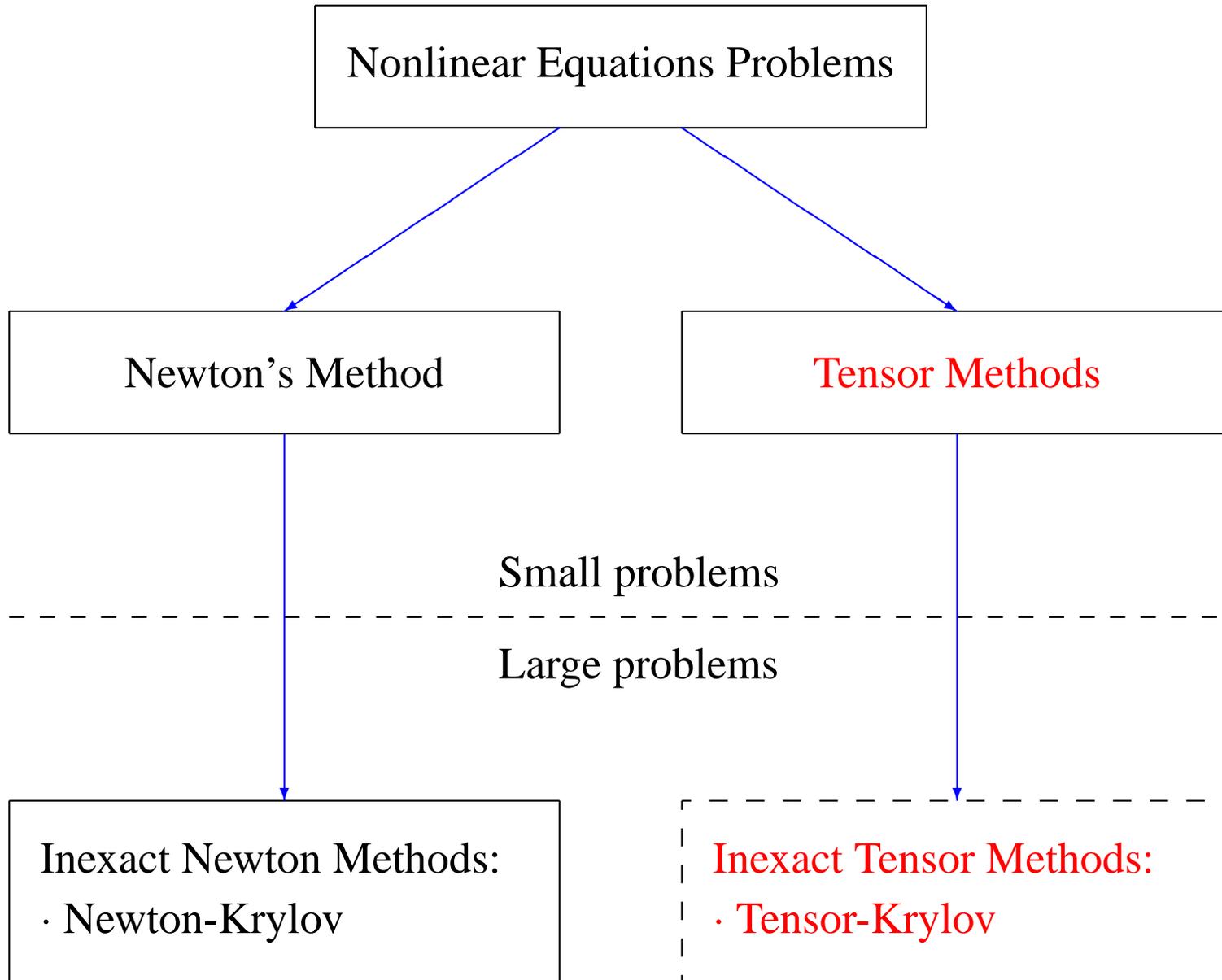


- Tamara Kolda (NOX)
- Roger Pawlowski (NOX, MPSalsa)
- John Shadid (MPSalsa)



- Joe Simonis (MPSalsa)

High-Level Outline



Nonlinear Equations Problem

Problem: Find x_* such that $F(x_*) = 0$, where $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$.

Notation:

x_k = current point

F_k = $F(x_k)$

J_k = $J(x_k) = F'(x_k)$

Areas of Research:

- Problems where $J(x_*)$ is ill-conditioned or singular
- Large-scale problems (e.g., PDE problems)
- Global strategies

Local Tensor Model

Taylor series expansion:

$$F(x_k + d) = F(x_k) + F'(x_k)d + \frac{1}{2}F''(x_k)dd + \mathcal{O}(d^3)$$

$\underbrace{\hspace{15em}}$
Second Order Approximation

Local model:

$$M_T(x_k + d) = F_k + J_k d + \frac{1}{2}T_k dd$$

$\underbrace{\hspace{4em}} \quad \underbrace{\hspace{4em}}$
Newton Tensor

Remarks:

1. $T_k \in \mathbb{R}^{n \times n \times n}$ supplies second-order information about $F(x)$ at x_k .
2. If using $T_k \equiv F''(x_k)$:
 - Would require $\frac{1}{2}n^3$ second partial derivatives
 - System of n quadratic equations in n unknowns

\Rightarrow **Not practical!**
3. More practical: $T_k = \sum_{i=1}^p u \otimes v \otimes w$

Practical Rank-one Tensor Method

(Schnabel and Frank, 1984)

Store a secant approximation to T_k only in the direction of the previous step.

$$\Rightarrow T_k = a \otimes s \otimes s$$

$$\begin{aligned} \text{where } s &= x_{k-1} - x_k \\ a &= \frac{2(F_{k-1} - F_k - J_k s)}{(s^T s)^2} \end{aligned}$$

$$M_T(x_k + d) = F_k + J_k d + \frac{1}{2} a (s^T d)^2$$

Remarks on direct method:

- 2 vectors of extra storage (s and a).
- Marginally more arithmetic than solving linear system (Newton's method).
- Reduces to 1 quadratic equation and an $(n - 1) \times (n - 1)$ linear system.

Tensor Method for Solving Nonlinear Equations

(Schnabel and Frank, 1984)

Now insert the local solver into a nonlinear solver framework...

Tensor Method:

Choose an initial x_0 ,

For $k = 0, 1, 2, \dots$, until termination, Do

1. Form local tensor model.
2. Find d that minimizes $\|M_T(x_k + d)\|_2$.
3. Update $x_{k+1} \leftarrow x_k + d$.
4. If x_{k+1} is not acceptable, then perform linesearch.

- 3-step superlinear convergence on singular problems.
- Modest to significant improvement over Newton's method on nonsingular and singular problems. (Schnabel and Frank, 1984; Bouaricha, 1992)

Simple Example

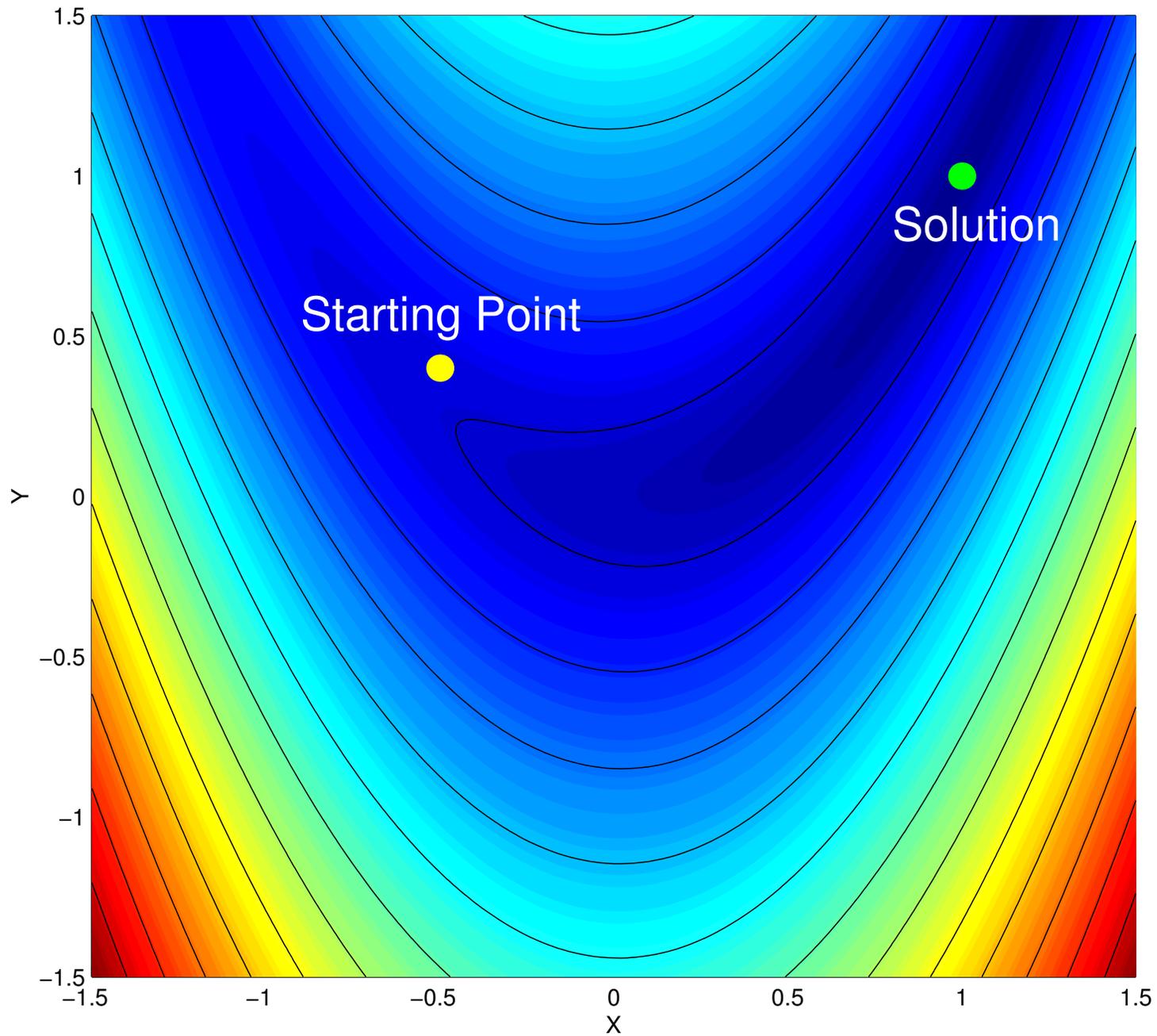
Modified Rosenbrock's function

$$F(x) = \begin{cases} f_1(x) & = 5(x_2 - x_1^2) \\ f_2(x) & = 1 - x_1 \end{cases}$$

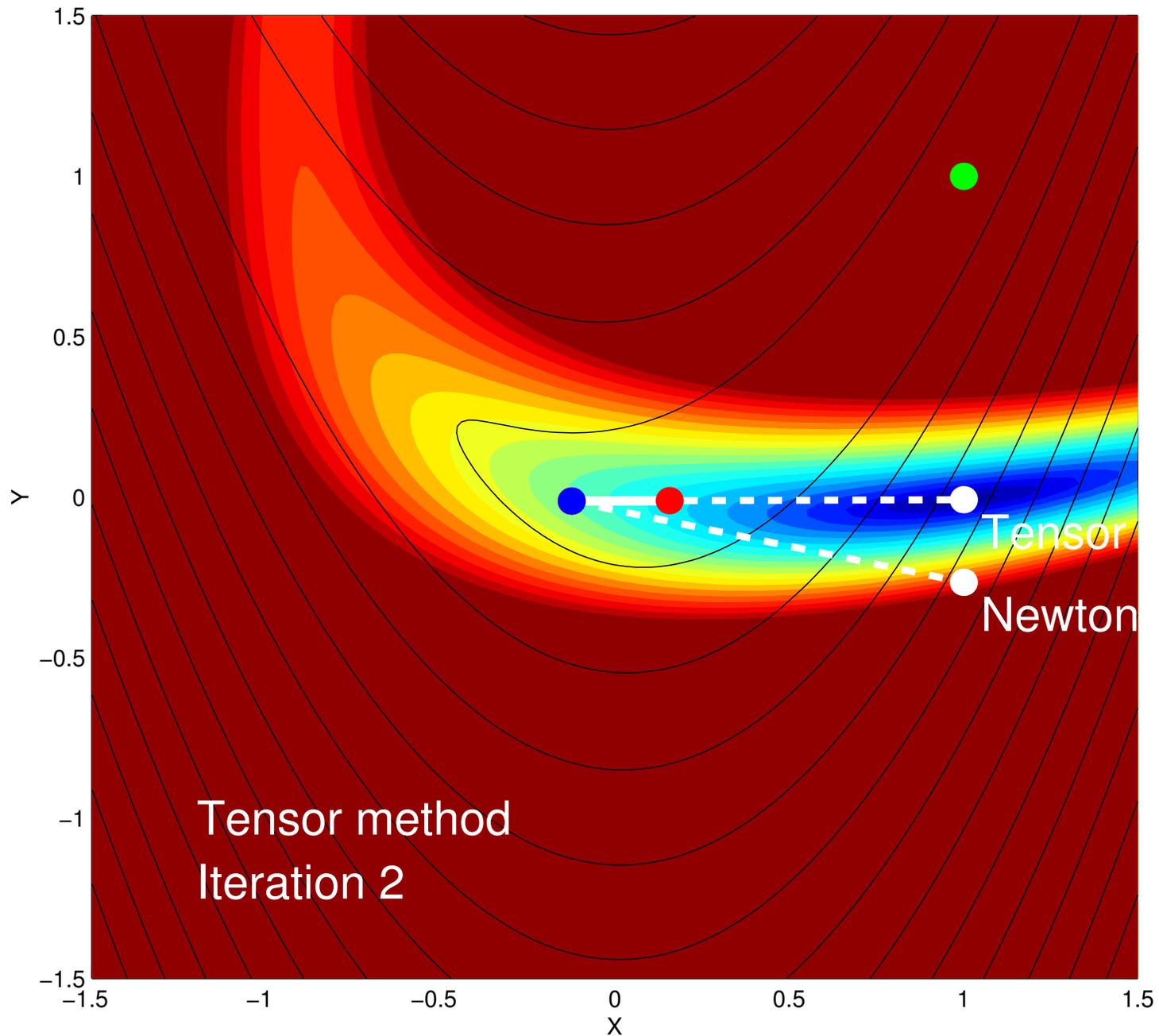
Plots will show:

- Contours of $\|F(x)\|$ and $\|M(x_k + d)\|$
- Full Newton and/or tensor step
- Accepted step

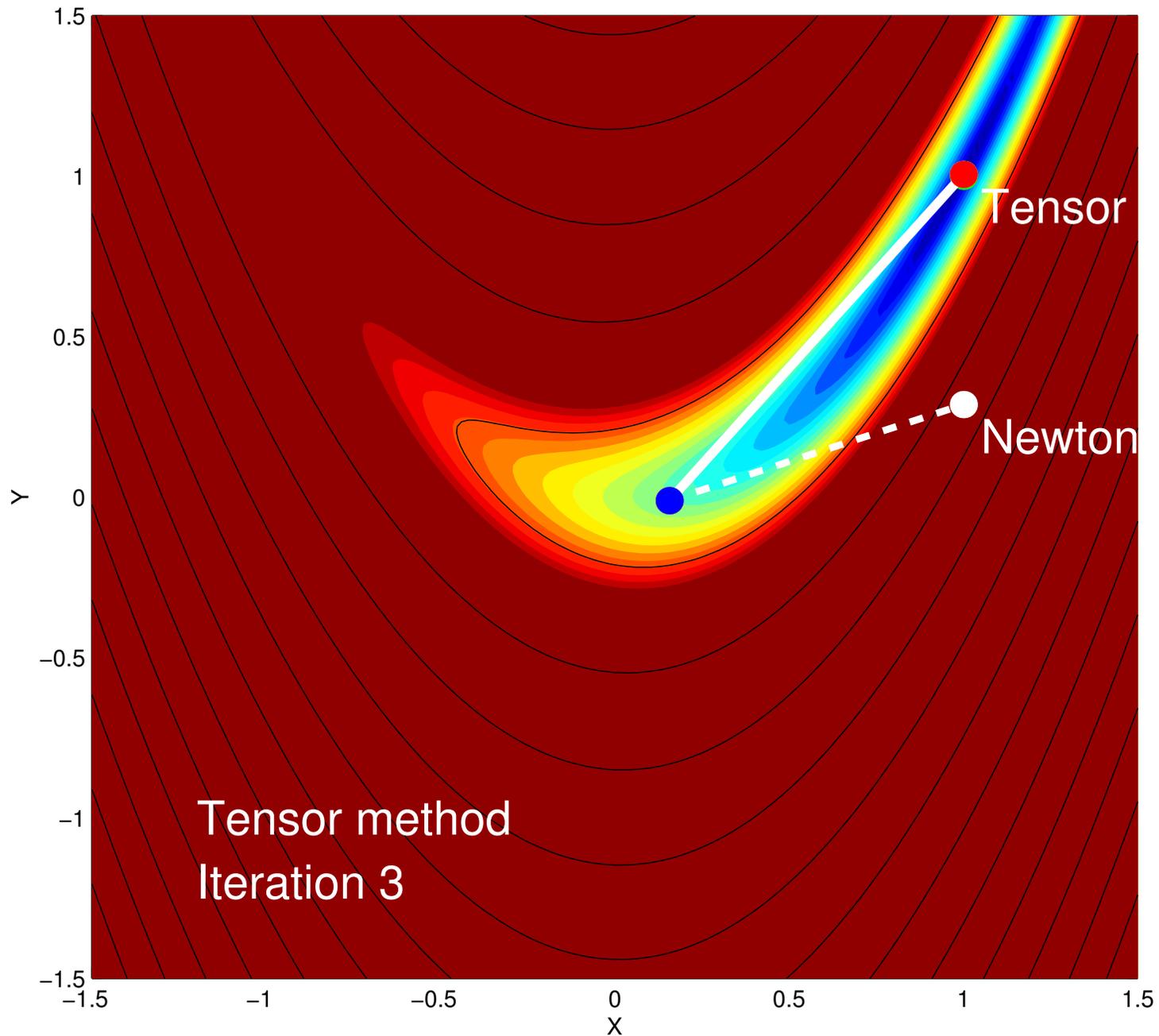
Simple Example



Simple Example



Simple Example



Research Objective

Large-scale tensor methods:

- Develop a Krylov-based iterative method that can solve the local tensor model in $\leq n$ steps.
- Expand to a fully featured “tensor-Krylov” method.
- Employ a “curvilinear linesearch” as the global strategy.
- Implement in production code and try solving Navier–Stokes problems.

Newton-Krylov Methods

(Brown and Saad, 1990; Brown and Hindmarsh, 1989; Chan and Jackson, 1986)

- Outer loop = Newton's method
- Inner loop = Linear Krylov subspace method
 - Residual $r_0 \equiv -F - Jd_0$
 - $\mathcal{K}_m(J, r_0) \equiv \text{span}\{r_0, Jr_0, J^2r_0, \dots, J^{m-1}r_0\}$
 - Find an approximate Newton step $d \in \mathcal{K}_m$ at each iterate.

Newton-GMRES:

Choose an initial x_0 ,

For $k = 0, 1, 2, \dots$, until termination, Do

1. Find step $d \in \mathcal{K}_m$ that minimizes $\|F_k + J_k d\|_2$.
2. Update $x_{k+1} \leftarrow x_k + d$.
3. If x_{k+1} is not acceptable, then perform linesearch.

\implies Now extend to a **tensor-Krylov** method.

The Tensor-Krylov Method

Tensor-Krylov Method:

Choose an initial x_0 ,

For $k = 0, 1, 2, \dots$, until termination, Do

1. Form local tensor model.
2. Find $d \in \mathcal{K}_m$ that minimizes $\|M_T(x_k + d)\|_2$.
3. Update $x_{k+1} \leftarrow x_k + d$.
4. If x_{k+1} is not acceptable, then perform **(curvilinear)** linesearch.

\Rightarrow Three methods for solving step 2...

\Rightarrow Introduce curvilinear linesearch...

Krylov-based Method for Solving Local Tensor Model

Find the step d that minimizes the tensor model

$$\min_{d \in \mathcal{K}_m} \left\| F_k + J_k d + \frac{1}{2} a (s^T d)^2 \right\|_2$$

where \mathcal{K}_m is a specially chosen Krylov subspace.

- Start with the block Krylov subspace $\mathcal{K}_0 = \text{span}\{s, a, F_k\}$.
- Use Arnoldi process to build an m dimensional orthogonal subspace and banded Hessenberg.
- Perform series of **plane rotations** to reduce Hessenberg system to:
 - Triangular system of $m - 1$ linear equations in m unknowns.
 - Four quadratic equations in 1 unknown.
- Solve for single unknown and then solve resultant linear system.

Call this method:

TK3

Two More Krylov-based Methods

$$\text{Local model: } M_T(x_k + d) = F_k + J_k d + \frac{1}{2} a (s^T d)^2$$

- Looks like a linear system involving a linear combination of 2 right-hand sides:

$$J_k d = -F_k - \frac{1}{2} a \beta^2 \quad \text{where } \beta \equiv s^T d$$

- Vector s not in right-hand side so $\text{span}\{s, Js, J^2s, \dots\}$ does little to help convergence.
- Start with the block Krylov subspace $\mathcal{K}_0 = \text{span}\{a, F_k\}$.
- Build $\mathcal{K}_m = \text{span}\{a, F_k, Ja, JF_k, J^2a, J^2F_k, \dots\}$.
- Solve the tensor model

$$\min_{d \in \mathcal{K}_m} \left\| F_k + Jd + \frac{1}{2} a (s^T d)^2 \right\|_2$$

TK2

$$\min_{d \in \{s\} \cup \mathcal{K}_m} \left\| F_k + Jd + \frac{1}{2} a (s^T d)^2 \right\|_2$$

TK2+

Similarities Among the Methods

- May control quality of inexact tensor step to within a specified tolerance, in contrast to other large-scale tensor methods.
- Formulations can handle:
 - Restarting
 - Left/right preconditioning
 - Most technology developed for GMRES

Differences Among the Methods

TK3

- Simpler to program
- Larger block for better memory performance?

TK2 & TK2+

- Typically fewer Jacobian-vector products needed to converge

- Each may be formulated as a block or scalar method.

JV versus *Jv*

- Block implementation more efficient in terms of memory performance.
(Fewer accesses to Jacobian)
- Scalar implementation more straightforward.

Tensor-GMRES Method of Feng and Pulliam

(Feng and Pulliam, 1997)

1. Find Newton-GMRES step
2. Save \mathcal{K}_m^N (i.e., V_m and \bar{H}_m)
3. Solve the **projected** tensor model

$$\min_{d \in \{d_0\} \cup \mathcal{K}_m^N} \left\| F_k + Jd + \frac{1}{2} P a (s^T d)^2 \right\|_2$$

where P is the projection matrix

$$P \equiv Y(Y^T Y)^{-1} Y^T, \quad Y \equiv J_k[V_m, d_0]$$

Notes:

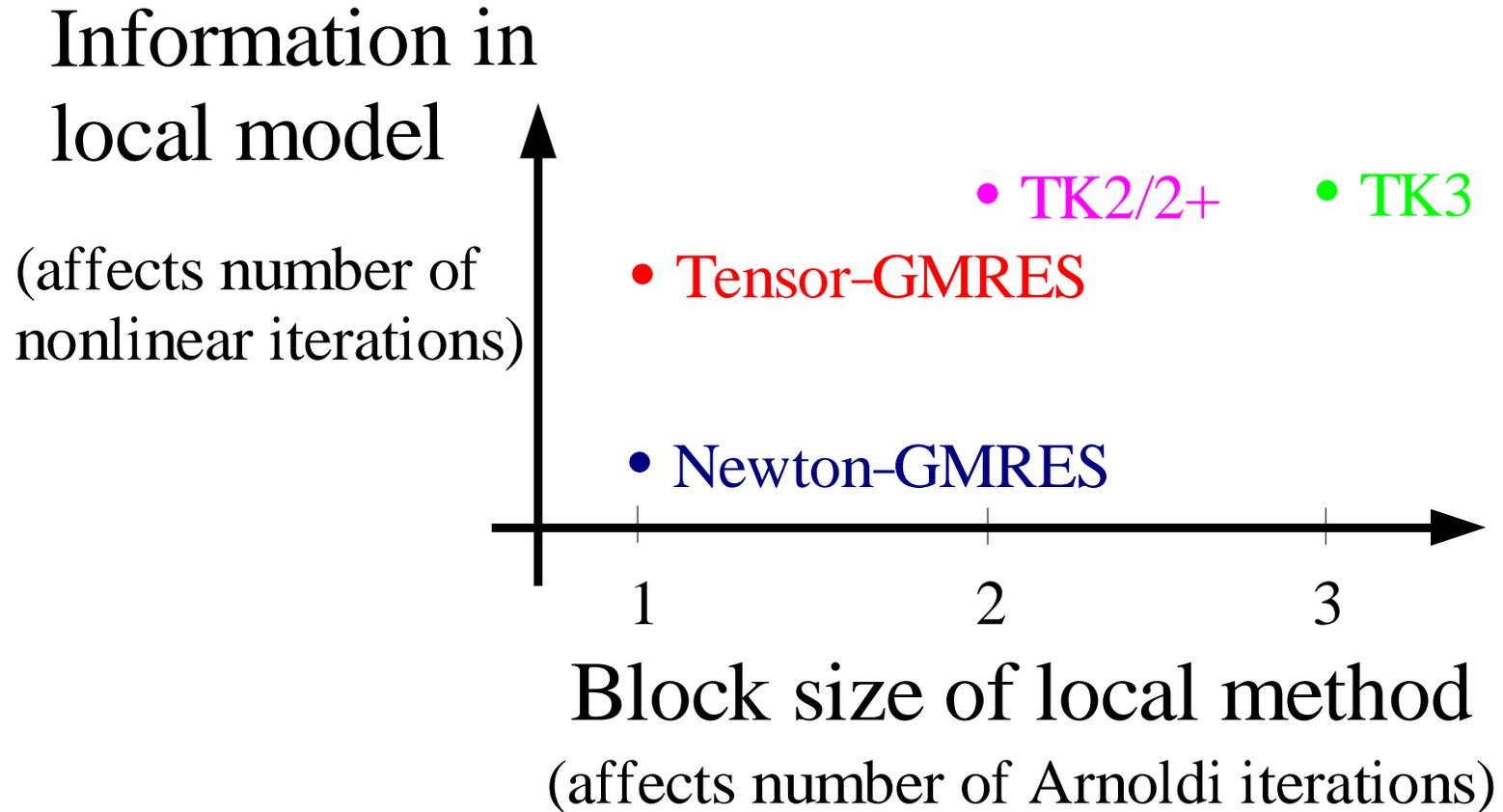
- Retains 3-step superlinear convergence properties (for ideal tensor method)
- How much will P and \mathcal{K}_m^N affect the quality of the step?
(e.g., preconditioning and restarting)

Method Comparison

	Cost per Local Solve	
Newton-GMRES	$\mathcal{O}(nm^2)$	
Tensor-Krylov (TK3)	GMRES + $10n + \underline{4nm} + 6m^2$	
Tensor-Krylov (TK2)	GMRES + $4n + \underline{3nm} + 6m^2 + 1(\mathbf{Jv})$	
Tensor-Krylov (TK2+)	GMRES + $7n + \underline{4nm} + \frac{17}{2}m^2 + 1(\mathbf{Jv})$	
Tensor-GMRES (Feng-Pulliam)	GMRES + $5n + \underline{4nm} + 2m^2 + 1(\mathbf{Jv})$	

	Strength	Weakness
Newton-GMRES	General use	Singular/ill-cond. problems
Tensor-Krylov (all)	Solves $M_T(x_k + d)$	Block-Krylov style
Tensor-GMRES (Feng-Pulliam)	GMRES front end	Projected $M_T(x_k + d)$ on \mathcal{K}_m^N

Graphical Comparison of Methods



Tensor-Krylov methods: Trade more inner iterations for fewer outer iterations.

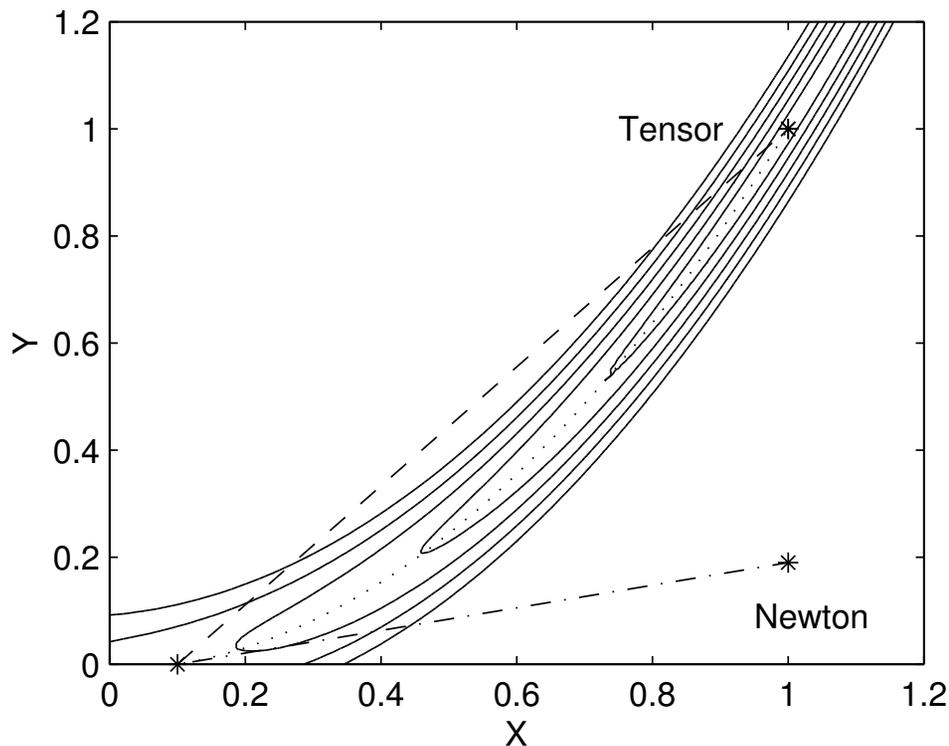
New Idea: Curvilinear Linesearch as Global Strategy

$$M_{\lambda T}(x_k + d) = \lambda F_k + J_k d + \frac{1}{2} a (s^T d)^2$$



Curvilinear step $d_T(\lambda)$

Contours of $\|M_T(x_k + d)\|_2$



General properties:

- $\|M_T(x_k + d_T(\lambda))\|_2$ increases monotonically from $\lambda = 1 \rightarrow 0$
- Asymptotically approaches d_N
- Resembles trust region method

Large-scale Problems

- Need a large-scale implementation of Tensor-Krylov → **NOX**
- Need several benchmark PDE problems for testing → **MPSalsa**
- Test 3 fluid flow problems

NOX: A C++ Objected-Oriented Nonlinear Equation Solver Package



Solver layer

.....

Abstract layer

.....

Linear algebra

.....

Application
interface

- Object-oriented C++ code using abstract and concrete classes for the construction and solution of nonlinear problems.
- Abstraction isolates the solver layer from...
 - Vector and matrix representation
 - Linear solver and/or preconditioners
 - Application interface ($F(x)$, $J(x)$)
- Nonlinear solvers and global strategies are written in a modular fashion to accommodate the user's linear solver package and parallel configuration.
- Includes several state-of-the-art solvers and is easily extensible for new solvers.

MPSalsa: 2D and 3D Parallel Reacting Flow Code



- Galerkin/Least-Squares Finite Element formulation on unstructured grids
- Creates fully coupled system of equations (fluid flow, heat transfer, and multi-component mass transfer with finite rate chemical reactions)
- Solves laminar and turbulent, low Mach number, reacting flows
- Slate of robust algorithms (not used in this comparison)
- Used to create test problems:
 - Backward-facing step
 - Lid driven cavity
 - Thermal convection
- **Reference for test cases:** J. N. Shadid, R. S. Tuminaro, H. F. Walker, “An Inexact Newton Method for Fully Coupled Solution of the Navier-Stokes Equations with Heat and Mass Transport,” *J. Computational Physics*, **137**, 155–185 (1997).

Numerical Tests

- Stopping conditions:

- Function reduction: $\|F(x_k)\| \leq 10^{-2} \|F(x_0)\|$

- Weighted step length: $\frac{1}{\sqrt{n}} \|W d_k\| < 1$

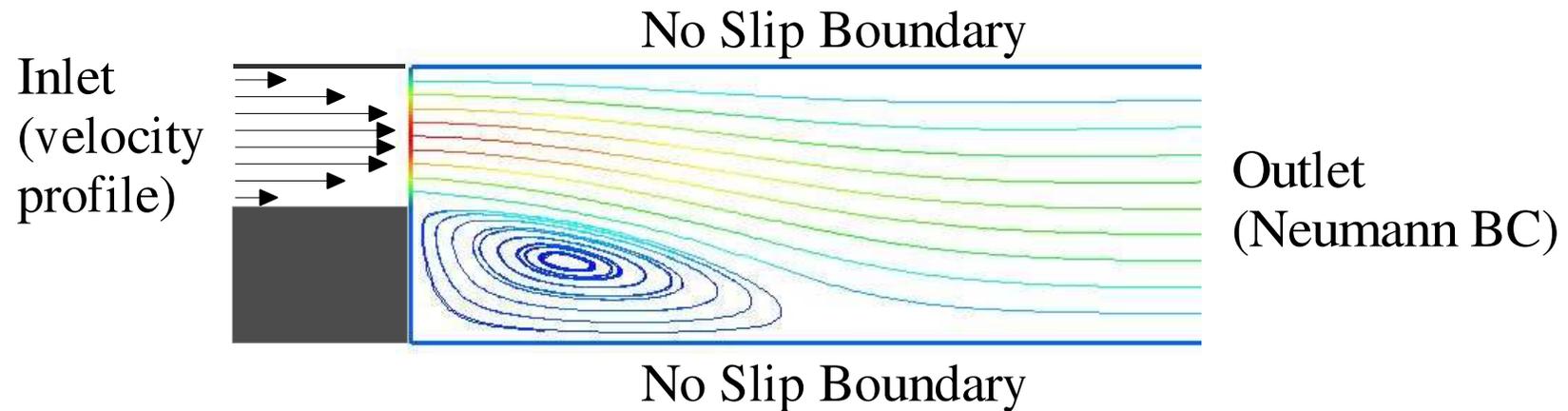
where: d_k = full Newton or tensor step

W = diagonal scaling matrix with entries

$$W_{ii} = \frac{1}{10^{-3} |x_{k_i}| + 10^{-8}}$$

- Initial starting vector $x_0 = 0$
- Constant forcing term $\eta_k = 10^{-4}$ in local solves
- Right Preconditioning using an ILUT preconditioner *
- Max 250 Arnoldi iterations and no restarts *
- Enabled maximum accuracy in Jacobian *
- No function or variable scaling *
- * Note: Conditions different from (Shadid et al., 1997) so results here will differ.

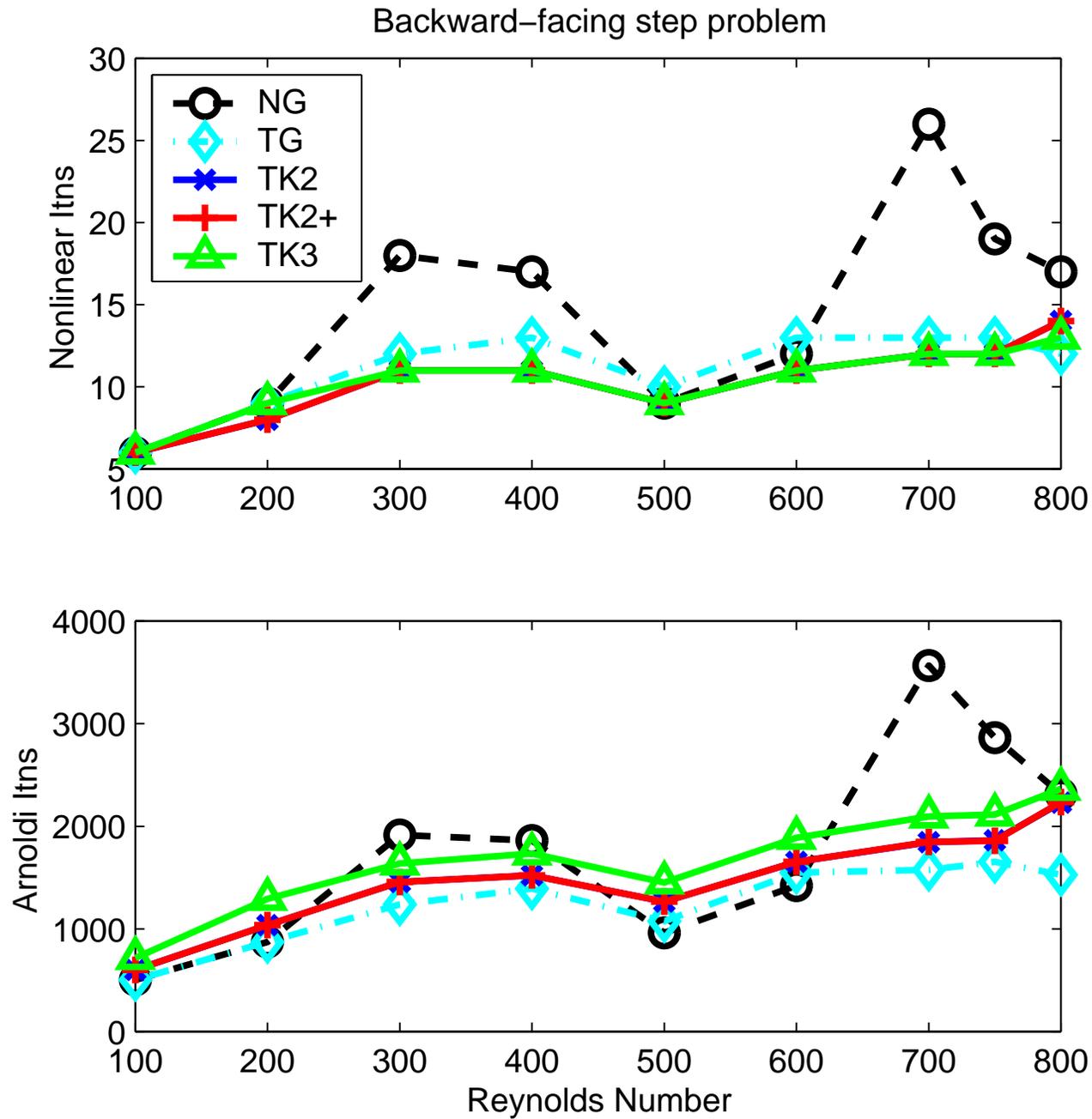
Backward-facing Step Problem



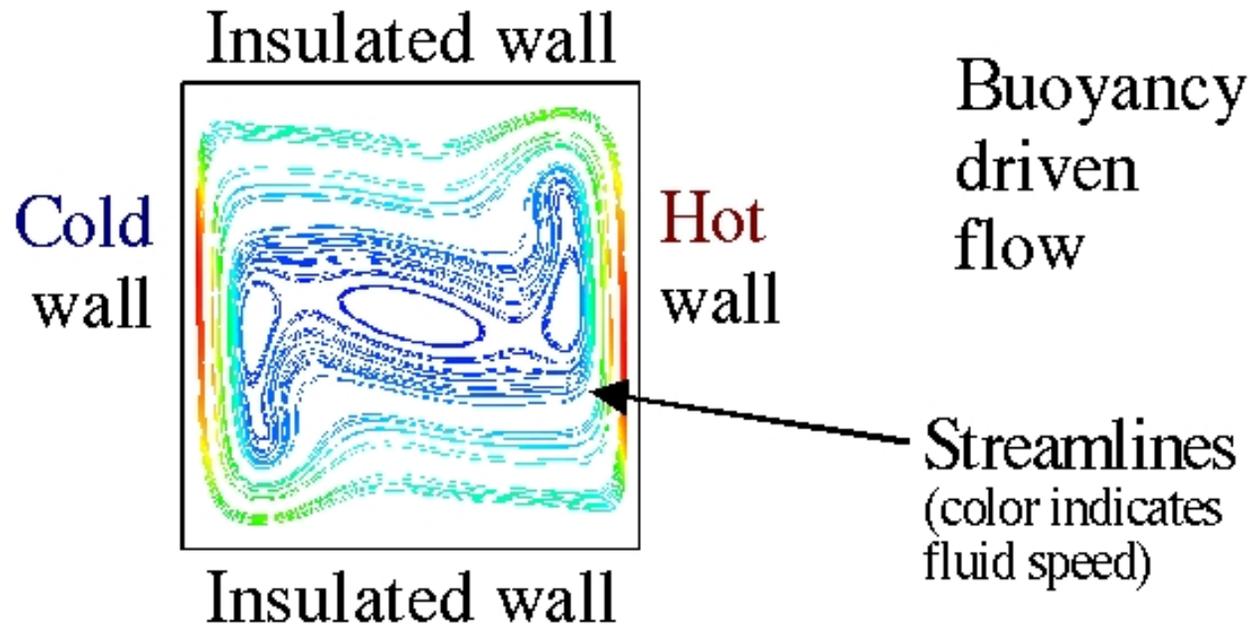
$$\begin{aligned}\nabla \cdot u &= 0 \\ \text{Re } u \cdot \nabla u + \nabla P - \nabla^2 u &= 0\end{aligned}$$

- Incompressible, steady-state flow: continuity and momentum (2D) equations
- Difficulty/nonlinearity controlled via the Reynolds number (Re)
- 20×400 mesh \rightarrow 24,000 unknowns

Backward-facing Step Results



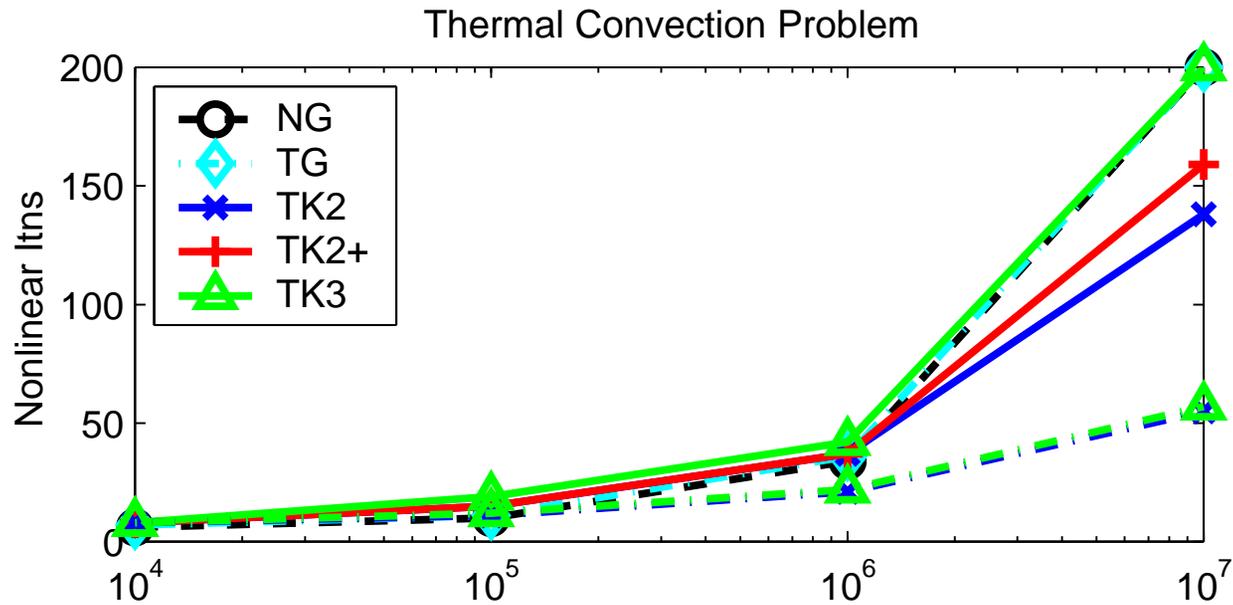
Thermal Convection Problem



$$\begin{aligned}\nabla \cdot u &= 0 \\ \text{Re } u \cdot \nabla u + \nabla P - \nabla^2 u + \text{Ra } T &= 0 \\ \text{Re } u \cdot \nabla u + \nabla T + \frac{1}{\text{Pr}} \nabla^2 T &= 0\end{aligned}$$

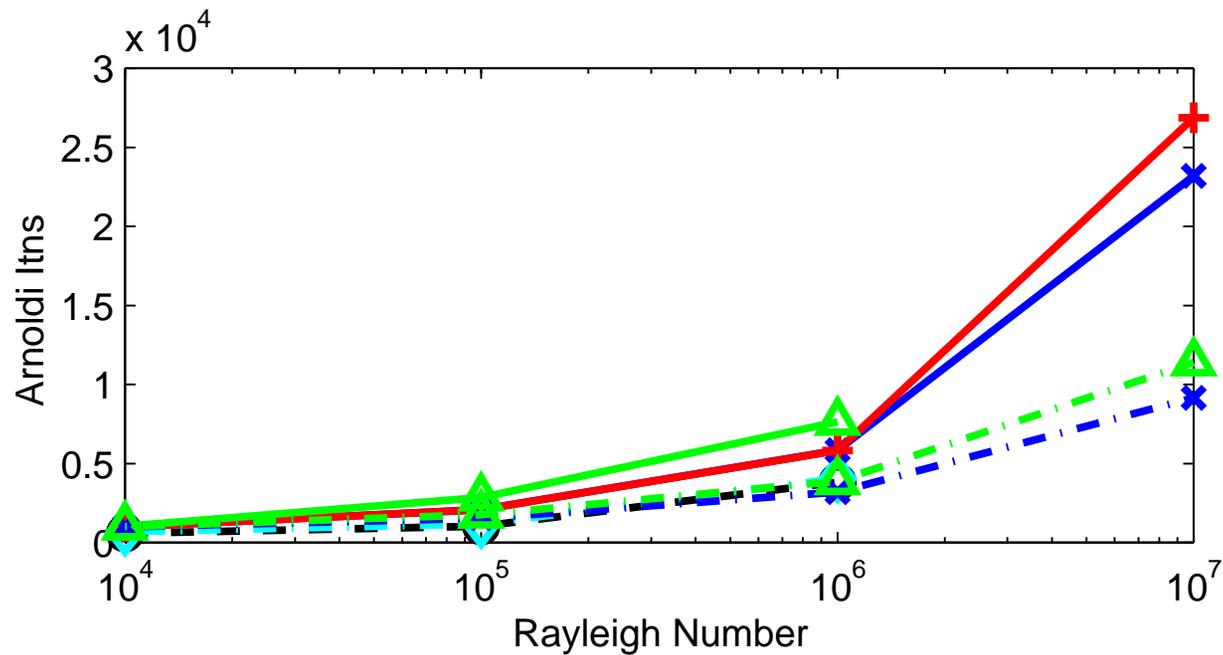
- Incompressible, steady-state flow: continuity, momentum (2D), and energy equations
- Difficulty/nonlinearity controlled via the Rayleigh number (Ra)
- 100×100 mesh \rightarrow 40,000 unknowns

Thermal Convection Problem Results

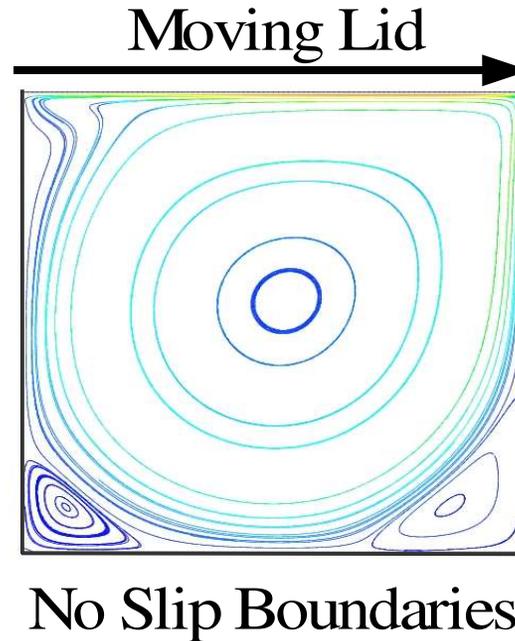


← Failures: NG, TG, TK3

← Curvilinear linesearch:
TK2, TK3



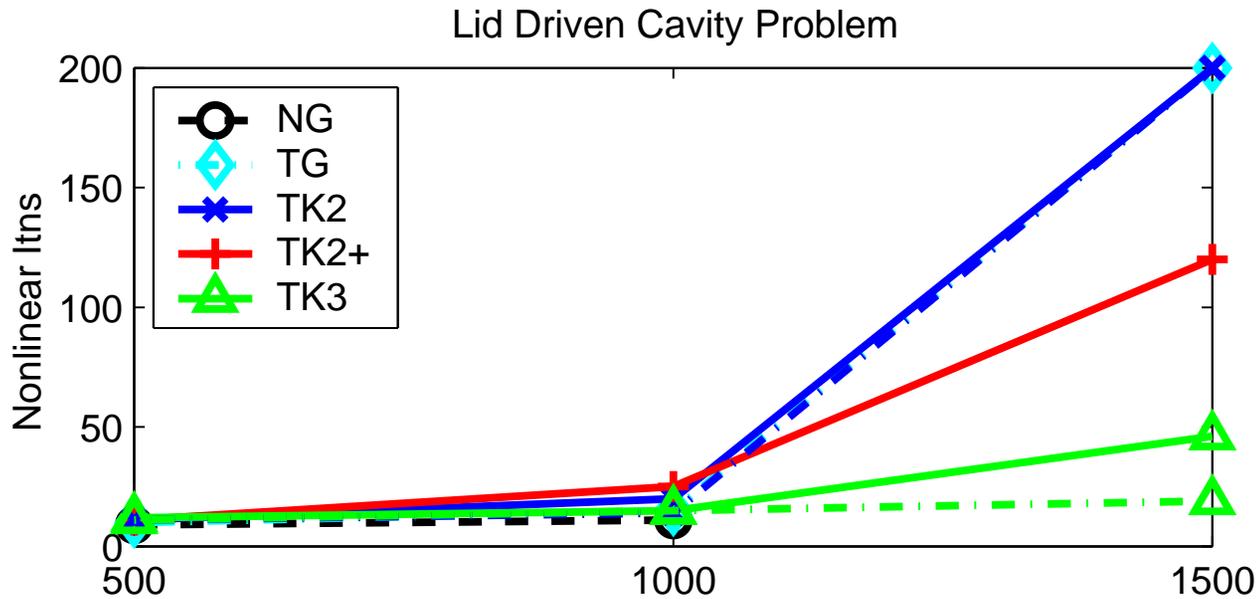
Lid Driven Cavity Problem



$$\begin{aligned}\nabla \cdot u &= 0 \\ \text{Re } u \cdot \nabla u + \nabla P - \nabla^2 u &= 0\end{aligned}$$

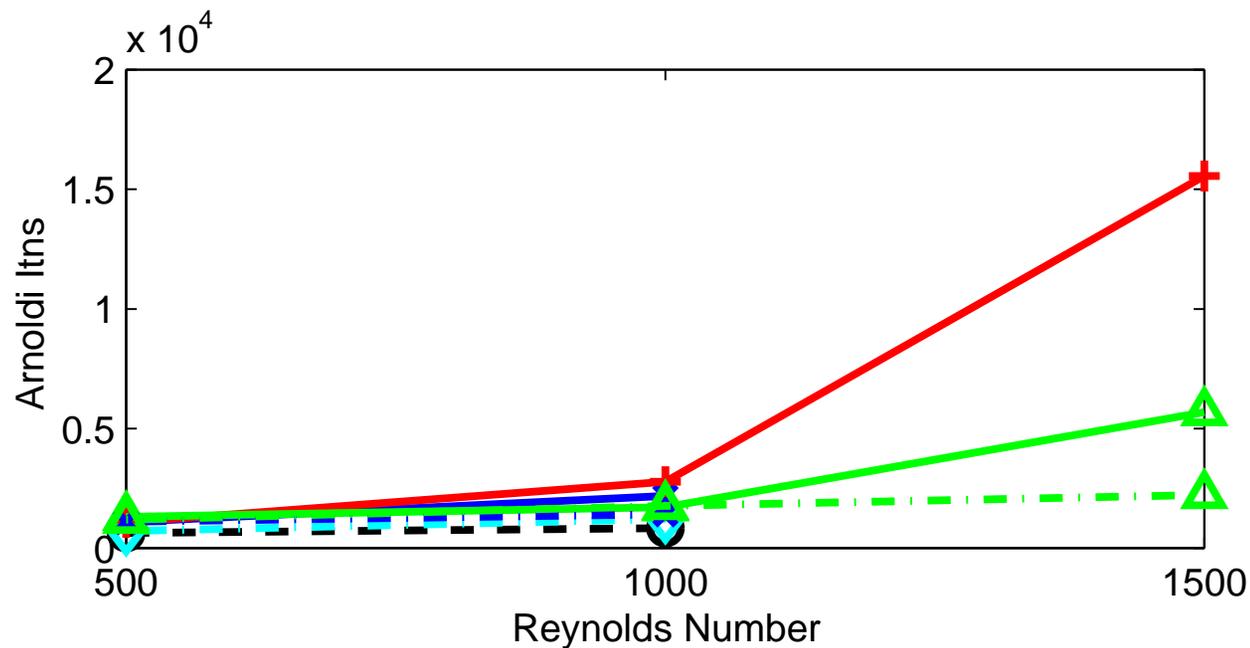
- Incompressible, steady-state flow: continuity and momentum (2D) equations
- Difficulty/nonlinearity controlled via the Reynolds number (Re)
- 100×100 mesh \rightarrow 30,000 unknowns

Lid Driven Cavity Results



← Failures: TG, TK2

NG: linesearch failure
at Re=1500 but could
solve Re=2000



Summary

- Derived three Krylov-based methods for iteratively solving $M_T(x_k + d)$.
- Developed a “curvilinear linesearch,” which resembles a trust region method and adds greater flexibility in search direction.
- Implemented the tensor-Krylov methods in NOX for solving large-scale problems.
- Results so far have shown the following:
 - Tensor methods are beneficial on ill-conditioned and singular problems.
 - Tensor-Krylov methods tend to be **more robust** than Newton-GMRES (when using constant forcing term).
 - Tensor-Krylov methods can be **more efficient** than Newton-GMRES, particularly on harder problems.
 - Tensor-GMRES is a competitive algorithm.

Thus, tensor-Krylov methods are useful on difficult problems that Newton-GMRES might have trouble with.

Future Research

Several unique directions to pursue:

- Use block implementation in tensor-Krylov methods for better memory efficiency (i.e., fewer accesses to Jacobian).
- Adapt method so it can use stand-alone linear algebra packages.
- Add an adjustable forcing term for greater robustness.

Contact Information

- Questions?
- Comments?
- More info?

Brett.Bader@Colorado.edu

<http://www.cs.colorado.edu/users/bader>