The TAO Linearly-Constrained Augmented Lagrangian Method for PDE-Constrained Optimization

Todd Munson, Evan Gawlik, Jason Sarich, and Stefan Wild

Mathematics and Computer Science Division
Argonne National Laboratory

Tools for Structured Optimization Problems on High-Performance Architectures: Lagrangians and Complementarity

October 18, 2011

Outline

1. Motivation
2. Algorithm
3. Numerical Results

Electrical Resistivity

- Perform a set of experiments by
  - Driving direct current signal into the ground
  - Measuring the resulting potentials created
- Inverse problem to determine resistivity

$$
\begin{aligned}
\min_{u,v} \quad & \frac{1}{2}\sum_{i=1}^{N} \|Q_i u_i - d_i\|^2 + \frac{1}{2}\alpha \int_\Omega (v - \bar{v})^2 d\Omega \\
\text{subject to } & -\nabla \cdot (e^v \nabla u_i) = q_i \quad x \in \Omega \quad i = 1, \ldots, N \\
& \nabla u_i \cdot n = 0 \quad x \in \partial\Omega \quad i = 1, \ldots, N \\
& \int_\Omega u_i d\Omega = 0 \quad i = 1, \ldots, N
\end{aligned}
$$

  - Discretize the operators
  - Solve finite-dimensional approximation
  - Problems available in the Haber-Hanson test collection

Generic Formulation

- Solve the optimization problem

$$\begin{aligned} \min_{u,v} \quad & f(u,v) \\ \text{subject to } & g(u,v) = 0 \end{aligned}$$

  - u are state variables, v are design variables
  - λ are Lagrange multipliers on the constraint
  - Assume $\nabla_u g(u,v)$ is invertible
- Develop efficient methods with modest requirements
  - Evaluate objective and constraint residual
  - Evaluate gradient and Jacobian vector products
  - Linear solves with $\nabla_u g(u,v)$ and $\nabla_u g(u,v)^T$
    - Reuse iterative methods and preconditioners
    - Use a small number of linear solves per major iteration

Linearly-Constrained Augmented Lagrangian Method

1. Given u_k, v_k, and λ_k
2. Choose ρ_k and approximately solve

$$\begin{aligned} \min_{u,v} \quad & m(u,v) \equiv f(u,v) - g(u,v)^T \lambda_k + \tfrac{\rho_k}{2} \|g(u,v)\|^2 \\ \text{subject to } & A_k(u - u_k) + B_k(v - v_k) + g_k = 0 \end{aligned}$$

   - $g_k \equiv g(u_k, v_k)$
   - $A_k \equiv \nabla_u g(u_k, v_k)$ and $B_k \equiv \nabla_v g(u_k, v_k)$
3. Calculate λ_{k+1}

Newton Step Toward Feasibility

- Solve the linear system

$$A_k d_u = -g_k$$

- If no solution or $g_k^T A_k d_u \geq 0$ then enter feasibility restoration
- Choose ρ_k so that d_u is descent direction for merit function

$$\rho_k \geq \max\left\{\rho_{k-1}, \frac{d_u^T(A_k^T \lambda_k - \nabla_u f(u_k, v_k))}{g_k^T A_k d_u}\right\}$$

- Determine the step length

$$\alpha_k \in \arg\min_{\alpha \geq 0} m_k(u_k + \alpha d_u, v_k)$$

- Accept the step

$$\begin{aligned} u_k^0 &= u_k + \alpha_k d_u \\ v_k^0 &= v_k \end{aligned}$$

Modified Reduced-Space Step Toward Optimality

- Start with constrained optimization problem

$$\begin{aligned} \min_{u,v} \quad & m_k(u,v) \\ \text{subject to } & A_k(u-u_k) + B_k(v-v_k) + \alpha_k g_k = 0 \end{aligned}$$

- Form reduced-space problem

$$\min_{v} \; m_k(u_k - A_k^{-1}(B_k(v-v_k) + \alpha_k g_k), v)$$

- Perform a change of variables and simplify

$$\min_{d_v} \; m_k(u_k^0 - A_k^{-1} B_k d_v, v_k^0 + d_v)$$

- Apply a limited-memory quasi-Newton method

Limited-Memory Quasi-Newton Method

1. Solve the quadratic approximation

$$\min_{d_v} \frac{1}{2} d_v \tilde{H}_{k,i} d_v + \tilde{g}_{k,i}^T d_v$$

   - $\tilde{g}_{k,i}$ is the reduced gradient

$$\tilde{g}_{k,i} = \nabla_v m_k(u_k^i, v_k^i) - B_k^T A_k^{-T} \nabla_u m_k(u_k^i, v_k^i)$$

   - $\tilde{H}_{k,i}$ is positive-definite reduced Hessian approximation
   - Based on properties of $\tilde{H}_{k,i}$ we can easily compute

$$d_v = -\tilde{H}_{k,i}^{-1} \tilde{g}_{k,i}$$

2. Perform line search to determine step length β_i
3. Update variables and Hessian approximation and repeat

Full-Space Line Search

- Avoid computing reduced-gradient in line search
- Recover full-space direction

$$d_u = -A_k^{-1} B_k d_v$$

- Linear constraint satisfied along full-space direction
- Calculate the step length

$$\min_{\beta \geq 0} m_k(u_k^i + \beta d_u, v_k^i + \beta d_v)$$

- Update

$$\begin{aligned} u_k^{i+1} &= u_k^i + \beta_k d_u \\ v_k^{i+1} &= v_k^i + \beta_k d_v \end{aligned}$$

Algorithm Overview

1. Determine u_k^0 and v_k^0 (one forward solve)
2. For $i = 1, \ldots$
   1. Compute reduced gradient (one adjoint solve)
   $$\tilde{g}_{k,i} = \nabla_v m_k(u_k^i, v_k^i) - B_k^T A_k^{-T} \nabla_u m_k(u_k^i, v_k^i)$$
   2. Update Hessian approximation using BFGS formula
   3. Calculate the direction
   $$d_v = -\tilde{H}_{k,i}^{-1} \tilde{g}_{k,i}$$
   4. Recover full space direction (one forward solve)
   $$d_u = -A_k^{-1} B_k d_v$$
   5. Find the step length β_k with a line search
   6. Update $u_k^{i+1} = u_k^i + \beta_k d_u$ and $v_k^{i+1} = v_k^i + \beta_k d_v$
3. Accept the new iteration
   - Compute reduced gradient at final point (one adjoint solve)
   - Update Hessian approximation
   - Set $\lambda_{k+1} = A_k^{-T} \nabla_u m_k(u_k^i, v_k^i)$

Computational Cost

- One forward solve per major iteration for Newton direction
- One adjoint solve per minor iteration for reduced gradient
- One forward solve per minor iteration for full step
- One adjoint solve per major iteration for Lagrange multipliers

- Hessian approximation carried over between major iterations

TAO Implementation

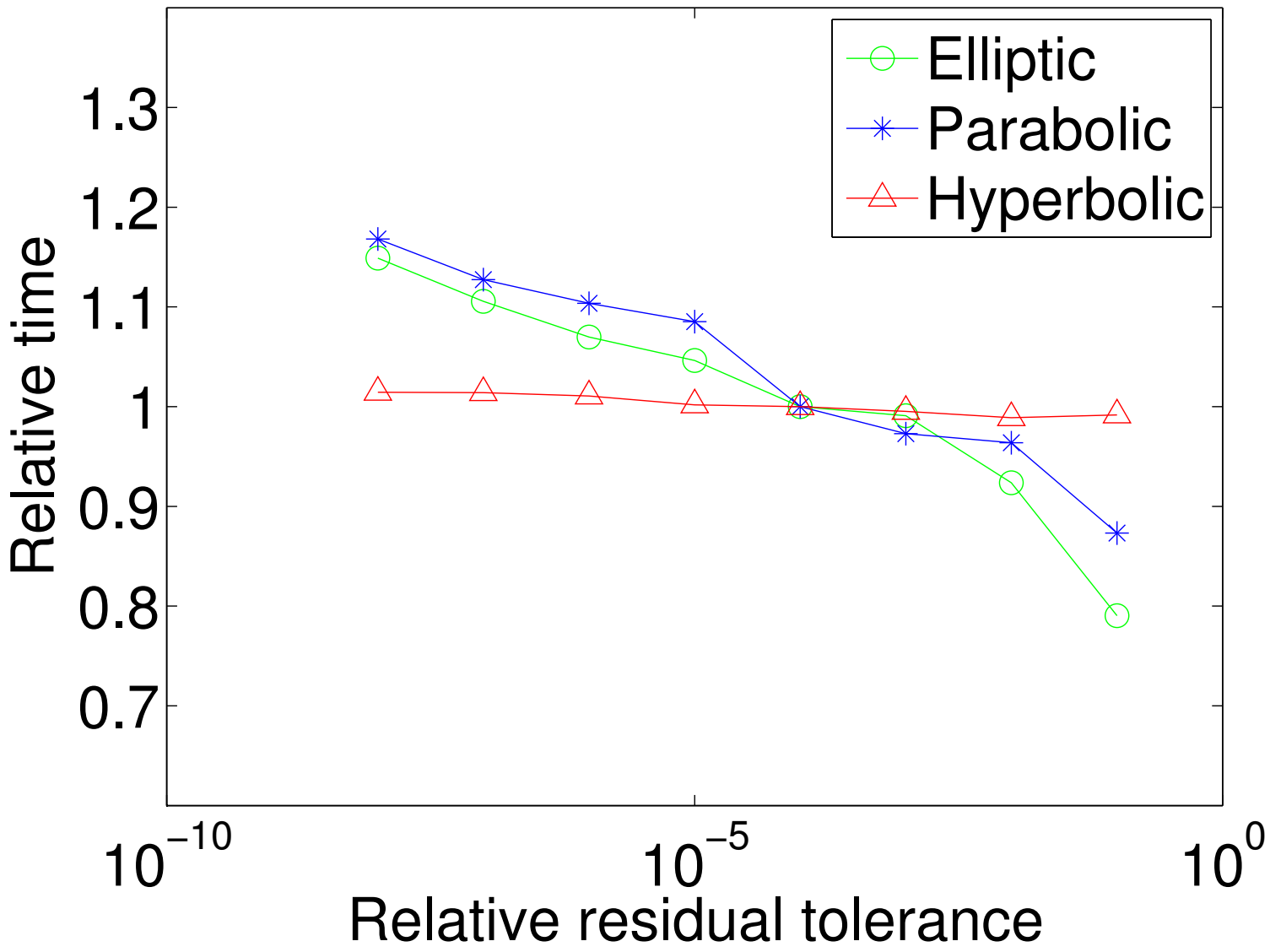
- Method implemented in Toolkit for Advanced Optimization
  - Employs PETSc for linear solvers and preconditioners
  - User provides code to evaluate needed functions
  - Available in upcoming TAO 2.0 release
- Some computational details
  - Overall convergence tolerance of 10^{-4}
  - Linear solver performed to relative tolerance of 10^{-4}
  - Limited-memory quasi-Newton approximation
    - BFGS approximation with a rank-5 matrix
    - Scaling matrix updated using Broyden updates
  - Moré-Thuente line search
    - Based on cubic interpolation
    - Guarantee that iterates satisfy Wolfe conditions

Test Problems

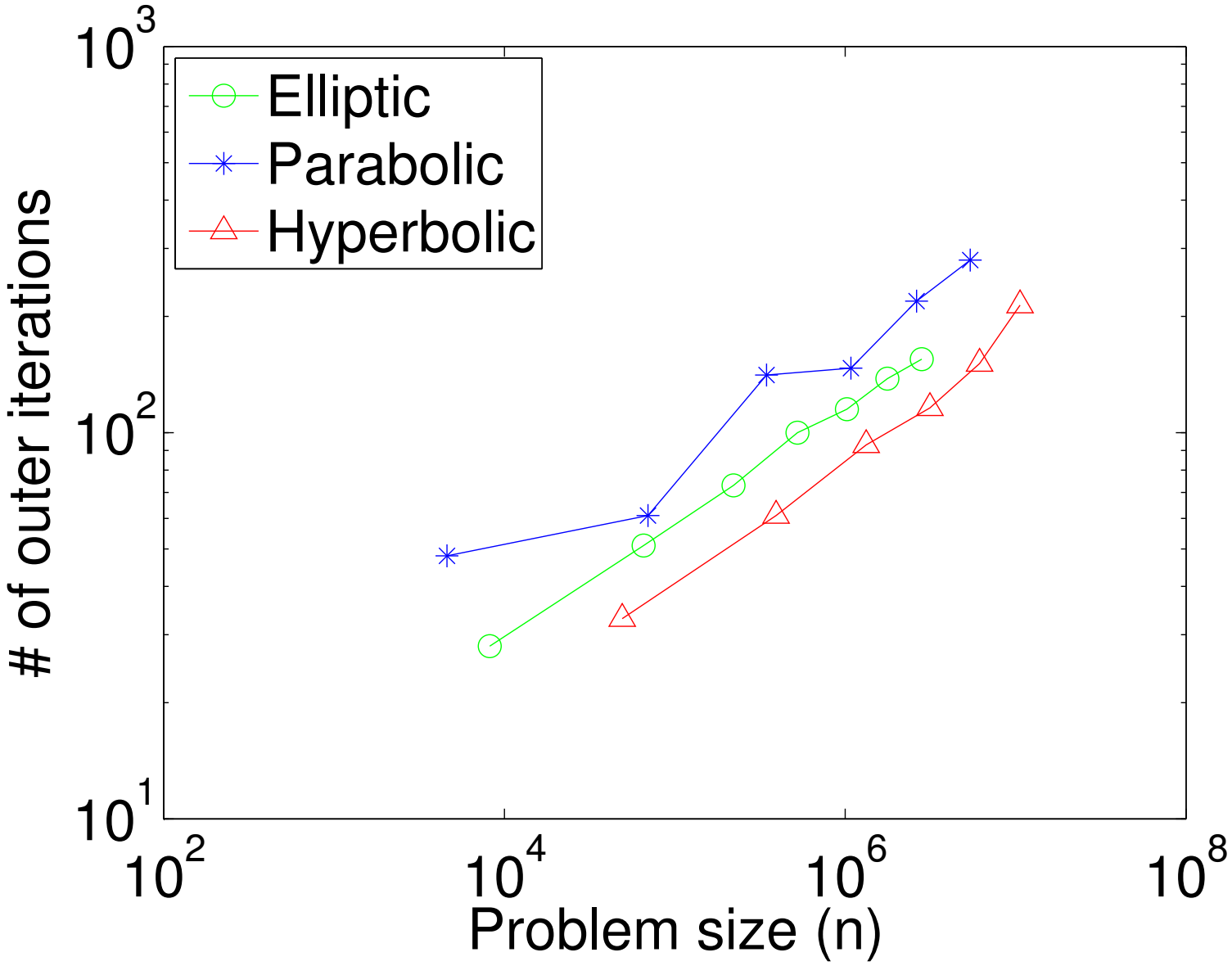
- Haber-Hanson test problems
  - Elliptic – electrical resistivity
  - Hyperbolic – optical tomography
  - Parabolic – mass transfer
- Discretizations and iterative methods given
  - Conjugate gradient method for symmetric matrices
  - Generalized minimum residual for non-symmetric matrices
  - Jacobi and successive over relaxation preconditioners

Problem	Dimension	Time-dependent	# State	# Design	Total (n)
Elliptic	3	No	$m_x^3 m_e$	m_x^3	$m_x^3(m_e+1)$
Parabolic	3	Yes	$m_x^3 m_t$	m_x^3	$m_x^3(m_t+1)$
Hyperbolic	2	Yes	$m_x^2 m_t$	$2m_x^2 m_t$	$3m_x^2 m_t$

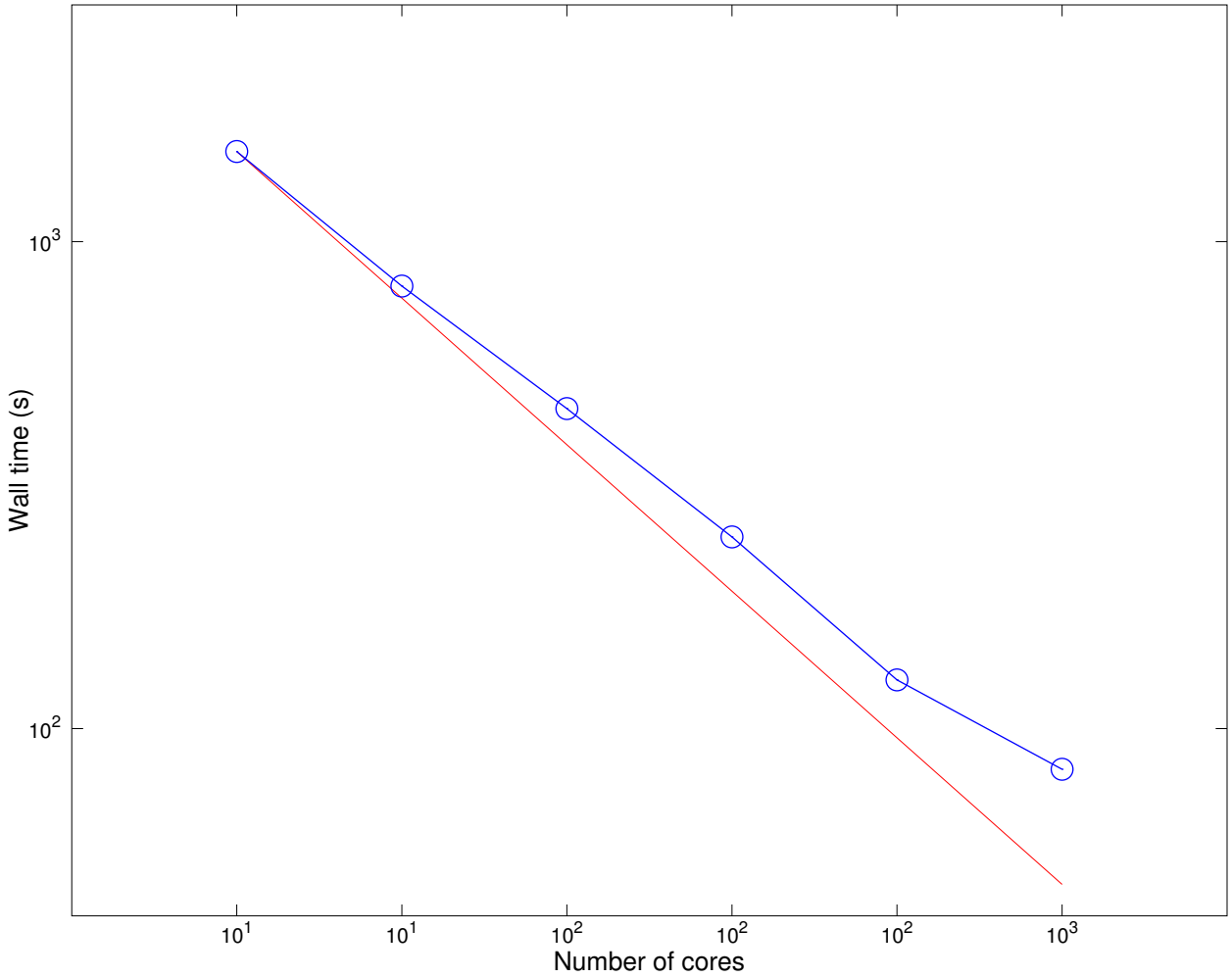
Sensitivity to Linear Solver Accuracy



Algorithm Scaling



Strong Scaling of Elliptic Problem ($m_x = 96$)



Conclusion

- Method is effective on the test problems
- Requires minimal information from the user
  - No need for the Hessian of Lagrangian
  - Use same iterative method for forward and adjoint
  - Preconditioner swaps sides in adjoint solves
- Cost dominated by (parallel) forward and adjoint solves
- Preliminary results indicate good strong scaling
- Available in upcoming TAO 2.0 release

`http://www.mcs.anl.gov/tao`