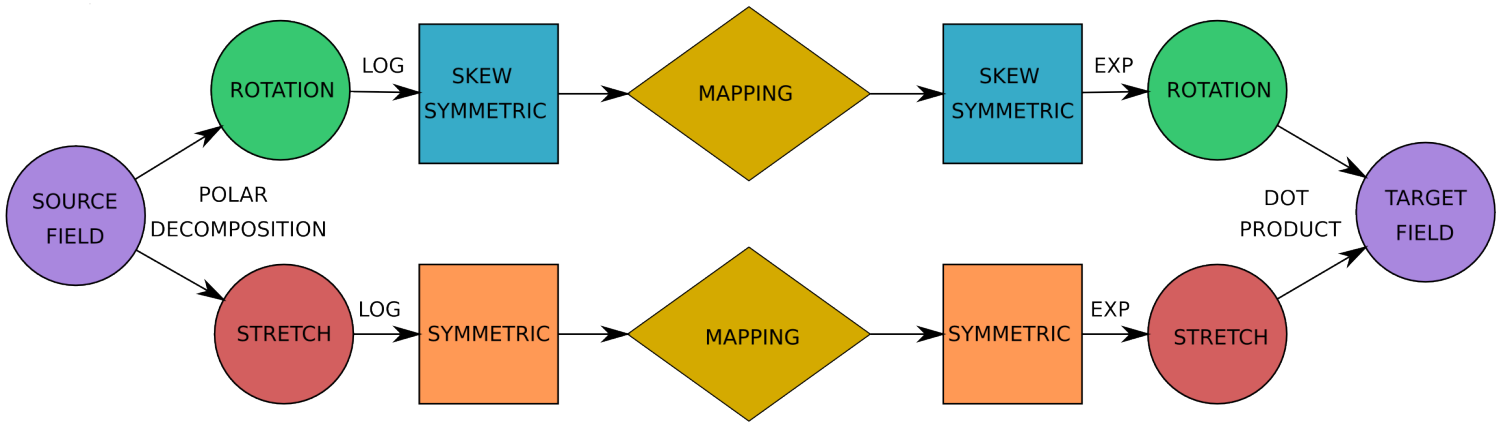
Exceptional service in the national

SAND2013-9451P

Sandia National Laboratories



Intrepid MiniTensor: A Small Vector and Tensor Library

Alejandro Mota[†], Jakob T. Ostien[†], WaiChing Sun[†], Qiushi Chen[‡]

[†] Mechanics of Materials, Sandia California; [‡] Glenn Department of Civil Engineering, Clemson University

U.S. DEPARTMENT OF ENERGY | NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Motivation

Sandia National Laboratories

- Implementation of complex constitutive models in Albany LCM (Laboratory for Computational Mechanics).
- Compact representation of expressions, ease of use.
- Expandable by leveraging the above.
- Optimization for small vectors and tensors.
- Accurate algorithms [Higham, 2008].
- Blitz++, TVMet, Eigen.

Features

- Vectors, second-order to fourth-order tensors.
- Static (for production) and dynamic (for research) storage.
- Storage replaceable with any linear access memory model.
- Basic manipulation, linear algebra, geometry and mechanics.
- Emphasis in accurate algorithms.
- Fully templated, plays well with Sacado.

Simple Case

Static Storage

```
#include "Intrepid_MiniTensor.h"
using namespace Intrepid;
using boost::tie;
...
Tensor<double, 3> const A(RANDOM_UNIFORM);
Tensor<double, 3> R, U;

tie(R, U) = polar_right(A);

Tensor<double, 3> const B = R * U;

double const error = norm(B - A) / norm(A);
...
```

Dynamic Storage

```
#include "Intrepid_MiniTensor.h"
using namespace Intrepid;
using boost::tie;
...
Tensor<double> const A(3, RANDOM_UNIFORM);
Tensor<double> R(3), U(3);

tie(R, U) = polar_right(A);

Tensor<double> const B = R * U;

double const error = norm(B - A) / norm(A);
...
```

More Complex Case

```
...
double const pi = acos(-1.0);

double phi = pi * random_uniform<double>();
double theta = 2.0 * pi * random_uniform<double>();

Vector<double, 3> const n(sin(phi) * sin(theta), cos(phi), sin(phi) * cos(theta));

phi = pi * random_uniform<double>();
theta = 2.0 * pi * random_uniform<double>();

Vector<double, 3> const m(sin(phi) * sin(theta), cos(phi), sin(phi) * cos(theta));

phi = 2.0 * pi * random_uniform<double>();
theta = 2.0 * pi * random_uniform<double>();

Vector<double, 3> const a = phi * m;
Vector<double, 3> const b = theta * m;

Tensor<double, 3> X = exp(skew(a));
Tensor<double, 3> Y = exp(skew(b));
Tensor<double, 3> D = exp(diag(Vector<double, 3>(RANDOM_NORMAL)));

Tensor<double, 3> A = X * D * transpose(Y);

Tensor<double, 3> U, S, V;

tie(U, S, V) = svd(A);

Tensor<double, 3> B = dot(U, dot_t(S, V));

double const error = norm(B - A) / norm(A);
...
```

Testing the Polar Decomposition

Sandia National Laboratories

$$J := \det \boldsymbol{F} = \lambda_1\lambda_2\lambda_3, \quad \theta := \log J, \qquad \gamma_i := \log \lambda_i, \quad \theta = \lambda_1 + \lambda_2 + \lambda_3 \tag{1}$$

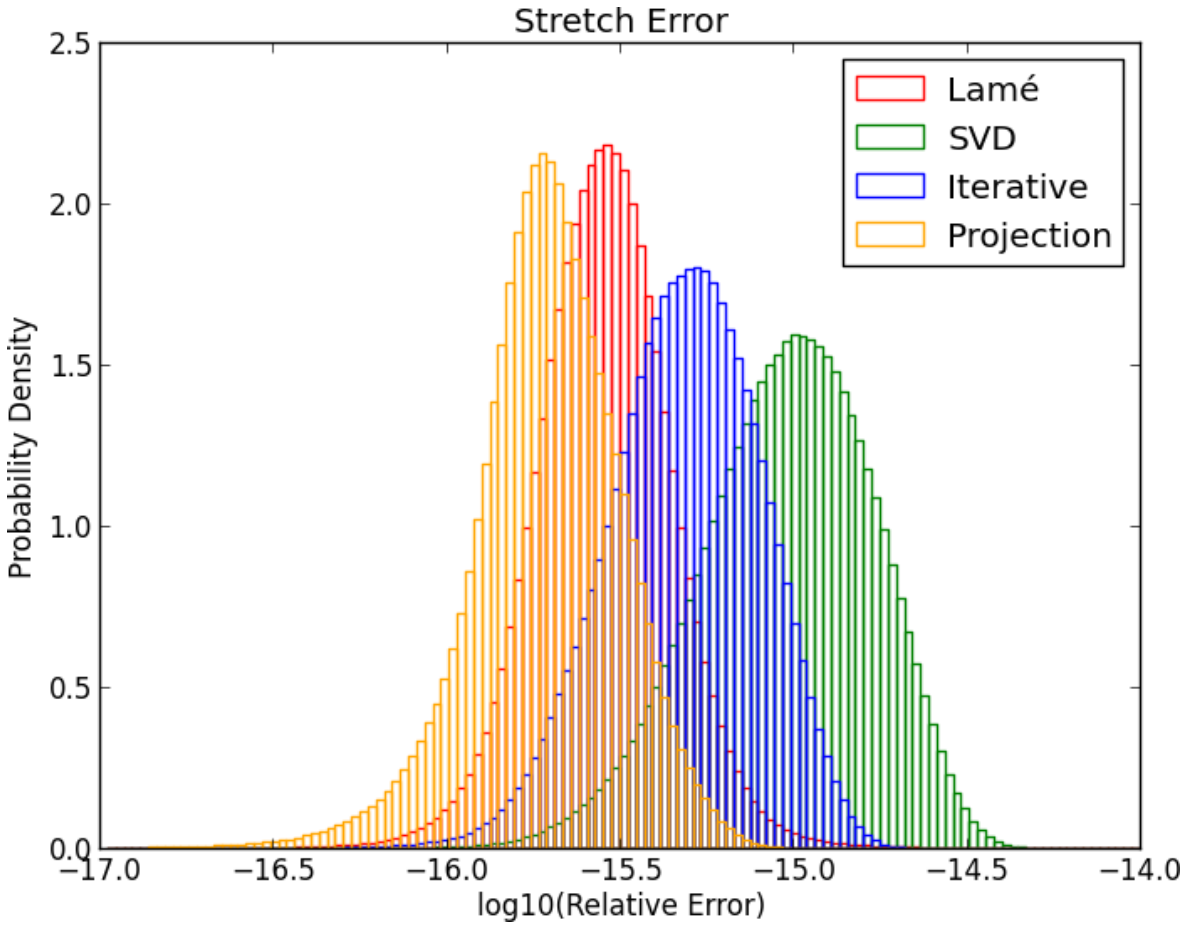
$$\gamma_i \sim N(\mu_\gamma, \sigma_\gamma) \quad \Rightarrow \quad \theta \sim N(3\mu_\gamma, \sqrt{3}\sigma_\gamma) \tag{2}$$

$$\mu_J = \exp\left(\mu_\theta + \frac{\sigma_\theta^2}{2}\right) = \exp\left(3\mu_\gamma + \frac{3\sigma_\gamma^2}{2}\right) \tag{3}$$

$$\begin{aligned} \sigma_J^2 &= \left[\exp(\sigma_\theta^2) - 1\right] \exp(2\mu_\theta + \sigma_\theta^2) \\ &= \left[\exp(3\sigma_\gamma^2) - 1\right] \exp(6\mu_\gamma + 3\sigma_\gamma^2) \end{aligned} \tag{4}$$

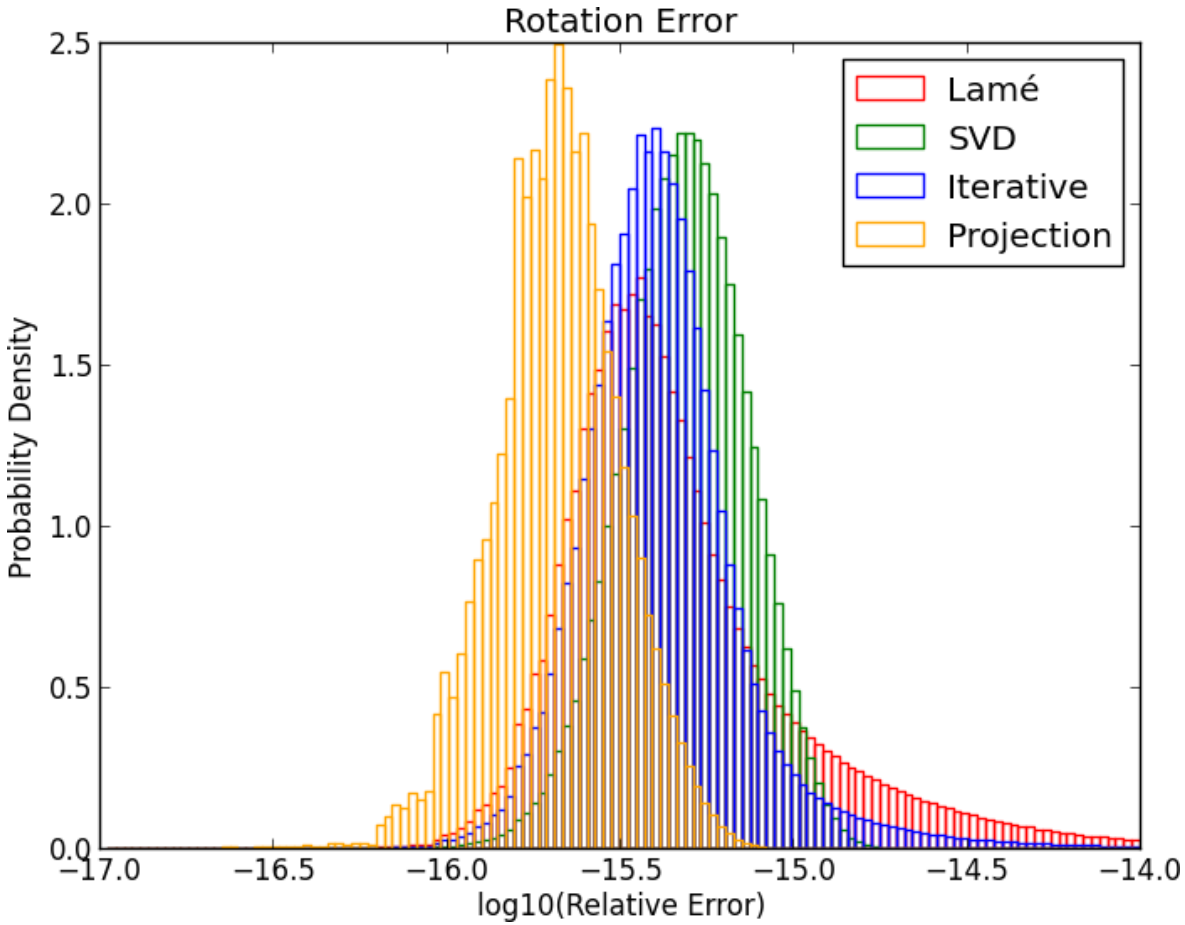
$$\mu_J \leftarrow 1, \quad \sigma_J \leftarrow 2, \quad N \leftarrow 10^7$$

Polar Decomposition: Stretch



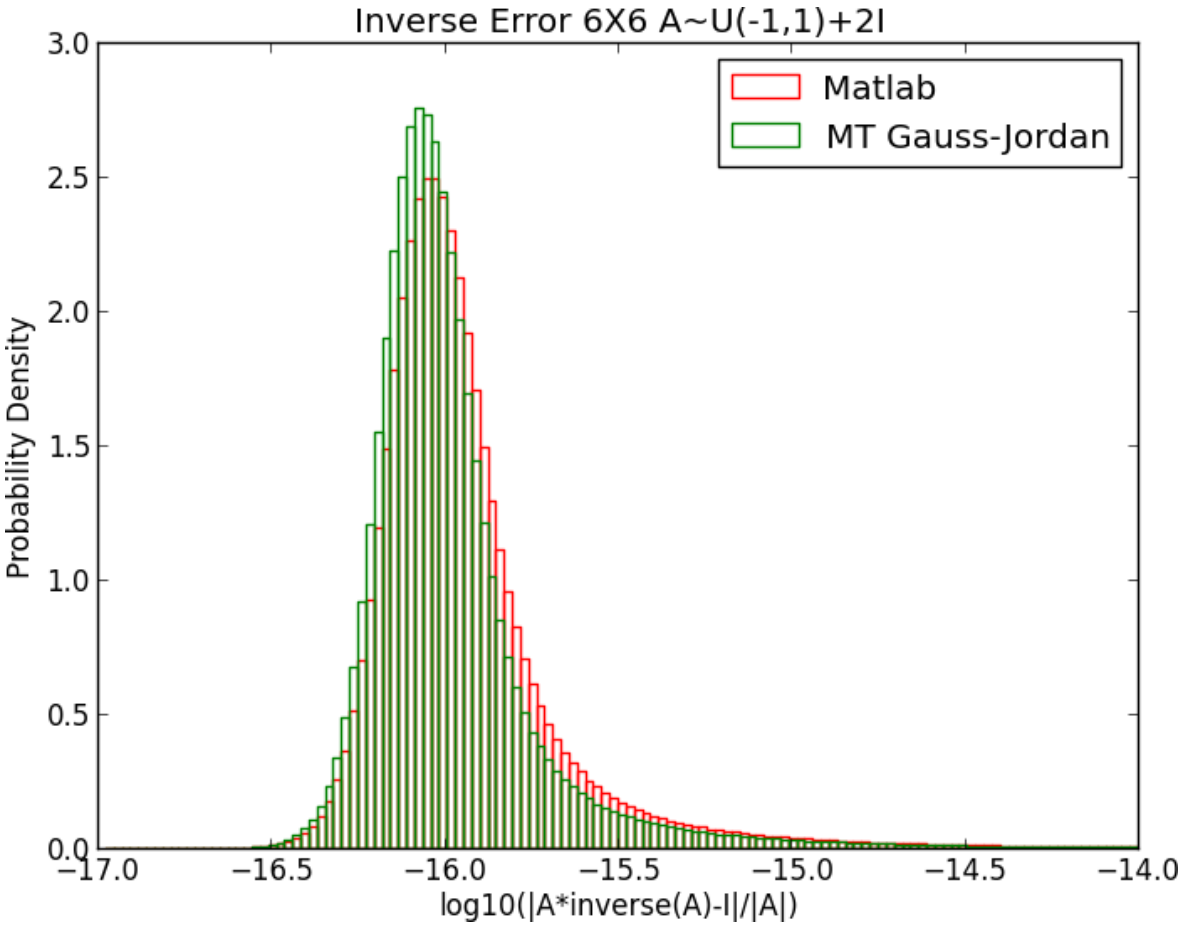
	μ	σ	t (s)
Lamé	-15.55	0.20	8.17
SVD	-15.02	0.25	23.43
Iterative	-15.32	0.22	11.49
Projection	-15.71	0.21	10.13

Polar Decomposition: Rotation



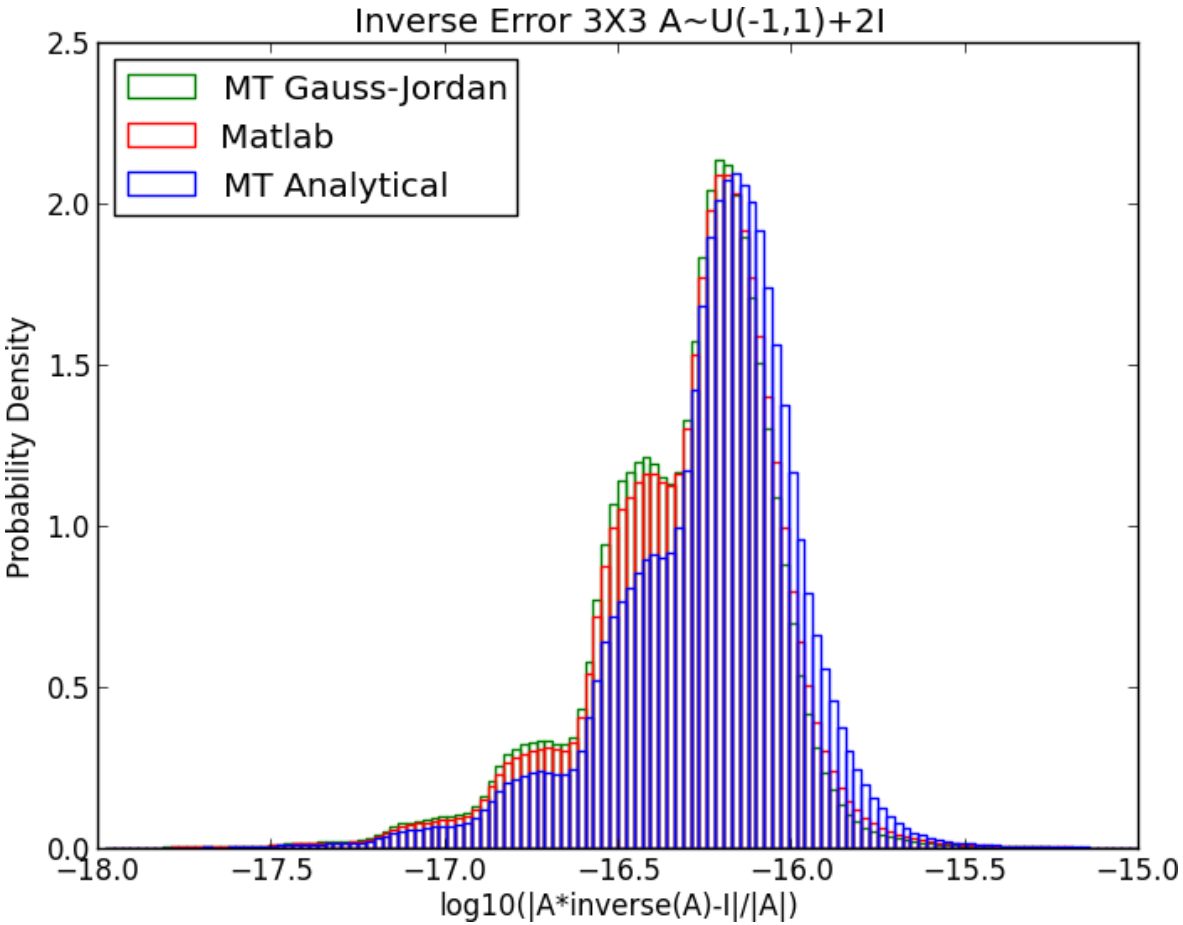
	μ	σ	t (s)
Lamé	-15.33	0.35	8.17
SVD	-15.31	0.18	23.43
Iterative	-15.38	0.24	11.49
Projection	-15.68	0.18	10.13

Inverse: Numerical



	μ	σ
Matlab	-15.94	0.28
MT Gauss-Jordan	-15.98	0.27

Inverse: Analytical



	μ	σ
MT Analytical	-16.23	0.27
Matlab	-16.30	0.27
MT Gauss-Jordan	-16.32	0.27

Work in Progress

Sandia National Laboratories

- Expression templates may not offer much performance gain.
- Support for complex numbers and corresponding algorithms.
- Better and faster algorithms for some current functionality.
- Extend functionality.

References I

N.J. Higham. *Functions of Matrices: Theory and Computation*. SIAM, Philadelphia, 2008.