

Hyperelasticity 101

What is a Hyperelastic Material?

- Hyperelastic materials can experience large strains (up to 500%) and most of it if not all is recoverable.
- For example, rubber is a hyperelastic material and its behavior is reminiscent of a viscous fluid during its processing to shape.
- The vulcanization and/or curing of rubber type materials causes their polymer chains to crosslink which allows the material to fully recover from elastic deformations.
- Load-Extension behavior is nonlinear.
- Nearly incompressible – one exception is some rubber foam materials where large volume changes can be achieved.

Why use Hyperelastic Materials?

- Low Cost
- Flexible
- Resilient – can work in many environments (moisture, pressure, heat)

Where are Hyperelastic Materials Used?

- Automotive (tires, belts, hoses, mounts)
- Aerospace (remember the failed O-ring on the space shuttle?)
- Biomedical/Dental Industries (artificial organs, wheelchairs, implantable surgical devices)
- Packaging (Styrofoam)
- Sports (equipment safety, shoes, helmets)

Commonly used Variables for Hyperelastic Material Models

Because hyperelastic materials can undergo large deformations while remaining elastic, their material models depend on variables that account for large deformations. Some variables to define are:

- Deformation Gradient: the fundamental measure of deformation in continuum mechanics. It is the second order tensor which maps line elements in the reference configuration into line elements (consisting of the same material particles) in the current configuration. The deformation gradient includes both deformation and rigid body rotation.

$$F_{ij} = \delta_{ij} + \frac{\partial u_i}{\partial X_j}$$

- Right Cauchy-Green Deformation Tensor: physically, gives the square of local change in a line element due to deformation. Invariants of the right Cauchy Green deformation tensor are often used in the expressions for strain energy density functions.

$$C_{ij} = F_{ki}F_{kj}$$

- Green-Lagrange Strain Tensor (sometimes referred to simply as Green Strain Tensor): represents large strains. It contains derivatives of the displacements with respect to the original

configuration. The values therefore represent strains in material directions, similar to the behavior of the Second Piola-Kirchhoff stress.

$$E_{ij} = \frac{1}{2}(C_{ij} - \delta_{ij})$$

- Strain energy density function: a scalar valued function that relates the strain energy density of a material to the deformation gradient through the right Cauchy Green deformation tensor. Note that the strain energy density function can also be written as a function of the Left Cauchy Green Deformation tensor or the principal stretches, but the Right Cauchy Green Deformation tensor is used in most of the equations given in this paper. A strain energy density function is used to define a hyperelastic material by postulating that the stress in the material can be obtained by taking the derivative of W with respect to the strain.

$$W = \hat{W}(C)$$

- Second Piola-Kirchhoff stress: measure of stress in the reference configuration. Independent of rigid body motion and of the choice of coordinate system.

$$S_{ij} = 2 \frac{\partial W}{\partial C_{ij}} = 2 \frac{\partial W}{\partial I_1} \frac{\partial I_1}{\partial C_{ij}} + 2 \frac{\partial W}{\partial I_2} \frac{\partial I_2}{\partial C_{ij}} + 2 \frac{\partial W}{\partial I_3} \frac{\partial I_3}{\partial C_{ij}}$$

Solution Steps

1. Define the strain energy density function for the hyperelastic material in terms of a deformation variable (for this research the Right Cauchy Green Deformation Tensor)
2. Calculate the derivatives of the strain energy density function with respect to the invariants of the deformation variable it is written as a function of
3. Calculate the derivatives of the invariants with respect to the deformation tensor
4. Plug everything into Second Piola-Kirchhoff stress
5. Convert into Cauchy stress (if desired)

Hyperelastic Constitutive Models

St. Venant-Kirchhoff Hyperelastic Material Model: the simplest hyperelastic material model, an extension of the linear elastic material model to the nonlinear regime.

$$W(\mathbf{E}) = \frac{\lambda}{2} [\text{tr}(\mathbf{E})]^2 + \mu \text{tr}(\mathbf{E}^2)$$

where λ and μ are the Lamé constants and $\mathbf{E}$ is the Green Strain Tensor.

Neo-Hookean Hyperelastic Material Model: similar to Hooke's law, but can be used for predicting the nonlinear stress-strain behavior of materials undergoing large deformations.

$$W(C) = \mu_1(I_1 - 3) + D_1(J - 1)^2$$

where, $\mu_1 = \frac{G}{2}$, G is the shear modulus, $D_1 = \frac{K}{2}$, K is the bulk modulus, and I_1 and I_2 are the first and second invariants of the Right Cauchy Green deformation tensor and J is the determinant of the deformation gradient.

Mooney-Rivlin Hyperelastic Material Model: hyperelastic material model where the strain energy density function is a linear combination of two invariants of the right Cauchy Green deformation tensor.

$$W(\mathbf{C}) = \mu_1(I_1 - 3) + \mu_2(I_2 - 3) + D_1(J - 1)^2$$

where, $\mu_1 = \frac{G}{2}$, G is the shear modulus, $E = 6(\mu_1 + \mu_2)$, E is Young's modulus, $D_1 = \frac{K}{2}$, K is the bulk modulus, and I_1 and I_2 are the first and second invariants of the Right Cauchy Green deformation tensor and J is the determinant of the deformation gradient.

*Note for incompressibility, $J = 1$, therefore incompressible Neo-Hookean and Mooney-Rivlin material models reduce the strain energy density equations above by dropping the $(J - 1)$ terms. Also note that if $\mu_2 = 0$, a Mooney-Rivlin material model reduces to a Neo-Hookean. Although hyperelastic materials are generally considered to be incompressible, in reality they are only nearly incompressible and the most accurate material models will include compressibility. Also, a compressible material model helps to avoid the numerical problems inherent in incompressible formulations (Poisson's ratio = 0.5).

Ogden Hyperelastic Material Model: used to describe the non-linear stress-strain behavior of complex materials such as rubbers, polymers, and biological tissue. The Ogden strain energy density function is written in terms of the principal stretches as:

$$W(\lambda_1, \lambda_2, \lambda_3) = \sum_{p=1}^N \frac{\mu_p}{\alpha_p} (\lambda_1^{\alpha_p} + \lambda_2^{\alpha_p} + \lambda_3^{\alpha_p} - 3)$$

where N , μ_p , and α_p are material constants and λ_j are the principal stretches.

*Note for $N = 1$ and $\alpha_1 = 1$, the Ogden model reduces to a Neo-Hookean material and when $N = 2$, $\alpha_1 = 2$, and $\alpha_2 = -2$, the Ogden model reduces to a Mooney-Rivlin material.

Yeoh Hyperelastic Material Model: uses higher order terms in I_1 to account for the departure from neo-Hookean/Gaussian behavior at large stretches.

$$W(\mathbf{C}) = C_{10}(I_1 - 3) + C_{20}(I_1 - 3)^2 + C_{30}(I_1 - 3)^3$$

where C_{10} , C_{20} , and C_{30} are material constants and I_1 is the first invariant of the Right Cauchy Green deformation tensor.

Limitations of Hyperelastic Material Models: Most material models allow the analyst to describe only a subset of the structural properties of hyperelastic materials.

1. The stress strain functions in the model are stable. They do not change with repetitive loading. The material model does not differentiate between a 1st time strain and a 100th time straining of the part under analysis.

2. There is no provision to alter the stress strain description in the material model based on the maximum strains experienced.
3. The stress strain function is fully reversible so that increasing strains and decreasing strains use the same stress strain function. Loading and unloading the part under analysis is the same.
4. The models treat the material as perfectly elastic meaning that there is no provision for permanent strain deformation. Zero stress is always zero strain.

Sources and further reading:

<http://www.ewp.rpi.edu/hartford/~ernesto/S2015/FWLM/TermProject/Misulia/Altidis2005-ANSYSUsersGroup-HyperelasticMaterials.pdf>

<http://www.umich.edu/~bme332/ch6consteqelasticity/bme332consteqelasticity.htm>

<http://www.brown.edu/Departments/Engineering/Courses/En221/Notes/Elasticity/Elasticity.htm>

<http://www.colorado.edu/engineering/cas/courses.d/NFEM.d/NFEM.Ch08.d/NFEM.Ch08.pdf>

<http://sameradeeb.srv.ualberta.ca/constitutive-laws/hyperelastic-materials/>

http://biomechanics.stanford.edu/me338/me338_project01.pdf

https://en.wikipedia.org/wiki/Strain_energy_density_function

Derivation of Equations for the explicit form of the final equations for reduced order models incorporating an incompressible Mooney-Rivlin hyperelastic material model

The deformation gradient tensor F_{ij} defined in terms of the Kronecker delta δ_{ij} and displacement field u_i , a function of the position vector in the reference configuration X_j .

$$F_{ij} = \delta_{ij} + \frac{\partial u_i}{\partial X_j}$$

The Right Cauchy Green deformation tensor, C_{ij} , is defined as:

$$C_{ij} = F_{li} F_{lj}$$

$$C_{ij} = \left(\delta_{li} + \frac{\partial u_l}{\partial X_i} \right) \left(\delta_{lj} + \frac{\partial u_l}{\partial X_j} \right)$$

$$C_{ij} = \delta_{li} \delta_{lj} + \delta_{li} \frac{\partial u_l}{\partial X_j} + \delta_{lj} \frac{\partial u_l}{\partial X_i} + \frac{\partial u_l}{\partial X_i} \frac{\partial u_l}{\partial X_j}$$

Using the substitution property of the Kronecker delta, where if a subscript on the Kronecker delta matches a subscript on what it is operating on, replace the matching subscript with the other subscript and drop the Kronecker delta.

Example:

$$\delta_{ij} u_j = u_i$$

Yields:

$$C_{ij} = \delta_{ij} + \frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_l}{\partial X_i} \frac{\partial u_l}{\partial X_j}$$

Strain energy, Ψ , for **incompressible Mooney-Rivlin materials** (1. Nam-Ho Kim: FEA for Nonlinear Elastic Problems, 2. University of Wisconsin: BME 615 Hyperelasticity Strain Energy Density, 3. Brown University School of Engineering: Continuum Mechanics, 4. University of Colorado: Review of Continuum Mechanics Field Equations):

$$\Psi(\mathbf{C}) = \mu_1(I_1 - 3) + \mu_2(I_2 - 3)$$

Where, μ_1 and μ_2 are material constants (Nam-Ho Kim: FEA for Nonlinear Elastic Problems):

$$\mu_1 = \frac{G}{2}$$

$$E = 6(\mu_1 + \mu_2)$$

Where E and G are Young's modulus and shear modulus of the material, respectively. Also, I_1 and I_2 are the first and second invariants, respectively, of the Right Cauchy Green deformation tensor.

Adopting a hyperelastic material model between the Right Cauchy Green deformation tensor and second Piola-Kirchhoff stress, S_{ij} , tensors gives:

$$S_{ij} = 2 \frac{\partial \Psi}{\partial C_{ij}} = 2 \frac{\partial \Psi}{\partial I_1} \frac{\partial I_1}{\partial C_{ij}} + 2 \frac{\partial \Psi}{\partial I_2} \frac{\partial I_2}{\partial C_{ij}} + 2 \frac{\partial \Psi}{\partial I_3} \frac{\partial I_3}{\partial C_{ij}}$$

Note that the strain energy equation used in this derivation isn't a function of I_3 , therefore the third term in the second Piola-Kirchhoff stress goes to zero.

Also (Bonet and Wood):

$$\frac{\partial I_1}{\partial C_{ij}} = \frac{\partial C_{kk}}{\partial C_{ij}} = \delta_{ij}$$

$$\frac{\partial I_2}{\partial C_{ij}} = \frac{\partial C_{kl} C_{kl}}{\partial C_{ij}} = 2C_{ij}$$

And:

$$\frac{\partial \Psi}{\partial I_1} = \frac{\partial}{\partial I_1} [\mu_1 (I_1 - 3) + \mu_2 (I_2 - 3)] = \mu_1$$

$$\frac{\partial \Psi}{\partial I_2} = \frac{\partial}{\partial I_2} [\mu_1 (I_1 - 3) + \mu_2 (I_2 - 3)] = \mu_2$$

Therefore:

$$S_{ij} = 2 \frac{\partial \Psi}{\partial I_1} \frac{\partial I_1}{\partial C_{ij}} + 2 \frac{\partial \Psi}{\partial I_2} \frac{\partial I_2}{\partial C_{ij}}$$

$$S_{ij} = 2\mu_1 \delta_{ij} + 2\mu_2 2C_{ij}$$

$$S_{ij} = 2\mu_1 \delta_{ij} + 4\mu_2 C_{ij}$$

$$S_{ij} = 2\mu_1 \delta_{ij} + 4\mu_2 \left(\delta_{ij} + \frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_i}{\partial X_i} \frac{\partial u_j}{\partial X_j} \right)$$

With S_{ij} as defined above, the weak formulation of the geometric (and material!) nonlinear problem is to find the displacement field u such that:

$$\underbrace{\int_{\Omega_0} \rho_0 v_i \ddot{u}_i dX}_{\mathbf{A}} + \underbrace{\int_{\Omega_0} \frac{\partial v_i}{\partial X_k} (F_{ij} S_{jk}) dX}_{\mathbf{B}} = \underbrace{\int_{\Omega_0} \rho_0 v_i b_i^0 dX}_{\mathbf{C}} + \underbrace{\int_{\partial\Omega_0^t} v_i t_i^0 ds}_{\mathbf{D}}$$

Where the assumed displacement field takes the form:

$$u_i(X, t) = \sum_{n=1}^M q_n(t) U_i^{(n)}(X)$$

And:

$$v_i = U_i^{(m)}$$

The components of this equation are labeled **A**, **B**, **C**, and **D** for breaking up the sections:

Looking at the terms inside the integral **A**:

$$\rho_0 v_i \ddot{u}_i$$

Yields:

$$\begin{aligned} &= \rho_0 U_i^{(m)} \frac{\partial^2 (q_n(t) U_i^{(n)}(X))}{\partial t^2} \\ &= \rho_0 U_i^{(m)} U_i^{(n)} \ddot{q}_n \end{aligned}$$

Looking at the terms inside the integral **B**:

$$\begin{aligned}
\frac{\partial v_i}{\partial X_k} (F_{ij} S_{jk}) &= \frac{\partial v_i}{\partial X_k} \left[\left(\delta_{ij} + \frac{\partial u_i}{\partial X_j} \right) \left(2\mu_1 \delta_{jk} + 4\mu_2 \left(\delta_{jk} + \frac{\partial u_j}{\partial X_k} + \frac{\partial u_k}{\partial X_j} + \frac{\partial u_l}{\partial X_j} \frac{\partial u_l}{\partial X_k} \right) \right) \right] \\
&= \frac{\partial v_i}{\partial X_k} \left[\delta_{ij} 2\mu_1 \delta_{jk} + \delta_{ij} 4\mu_2 \delta_{jk} + \delta_{ij} 4\mu_2 \frac{\partial u_j}{\partial X_k} + \delta_{ij} 4\mu_2 \frac{\partial u_k}{\partial X_j} + \delta_{ij} 4\mu_2 \frac{\partial u_l}{\partial X_j} \frac{\partial u_l}{\partial X_k} + \frac{\partial u_i}{\partial X_j} 2\mu_1 \delta_{jk} + \frac{\partial u_i}{\partial X_j} 4\mu_2 \delta_{jk} + \frac{\partial u_i}{\partial X_j} 4\mu_2 \frac{\partial u_j}{\partial X_k} \right. \\
&\quad \left. + \frac{\partial u_i}{\partial X_j} 4\mu_2 \frac{\partial u_k}{\partial X_j} + \frac{\partial u_i}{\partial X_j} 4\mu_2 \frac{\partial u_l}{\partial X_j} \frac{\partial u_l}{\partial X_k} \right] \\
&= \frac{\partial v_i}{\partial X_k} \left[2\mu_1 \delta_{ik} + 4\mu_2 \delta_{ik} + 4\mu_2 \frac{\partial u_i}{\partial X_k} + 4\mu_2 \frac{\partial u_k}{\partial X_i} + 4\mu_2 \frac{\partial u_l}{\partial X_i} \frac{\partial u_l}{\partial X_k} + 2\mu_1 \frac{\partial u_i}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_j}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_k}{\partial X_j} \right. \\
&\quad \left. + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_l}{\partial X_j} \frac{\partial u_l}{\partial X_k} \right] \\
&= \frac{\partial v_i}{\partial X_k} \left[(2\mu_1 + 4\mu_2) \delta_{ik} + (2\mu_1 + 8\mu_2) \frac{\partial u_i}{\partial X_k} + 4\mu_2 \frac{\partial u_k}{\partial X_i} + 4\mu_2 \frac{\partial u_l}{\partial X_i} \frac{\partial u_l}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_j}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_k}{\partial X_j} \right. \\
&\quad \left. + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_l}{\partial X_j} \frac{\partial u_l}{\partial X_k} \right] \\
&= \frac{\partial v_i}{\partial X_k} (2\mu_1 + 4\mu_2) \delta_{ik} \\
&\quad + \frac{\partial v_i}{\partial X_k} \left[(2\mu_1 + 8\mu_2) \frac{\partial u_i}{\partial X_k} + 4\mu_2 \frac{\partial u_k}{\partial X_i} + 4\mu_2 \frac{\partial u_l}{\partial X_i} \frac{\partial u_l}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_j}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_k}{\partial X_j} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_l}{\partial X_j} \frac{\partial u_l}{\partial X_k} \right] \\
&= (2\mu_1 + 4\mu_2) \frac{\partial v_k}{\partial X_k} \\
&\quad + \frac{\partial v_i}{\partial X_k} \left[(2\mu_1 + 8\mu_2) \frac{\partial u_i}{\partial X_k} + 4\mu_2 \frac{\partial u_k}{\partial X_i} + 4\mu_2 \frac{\partial u_l}{\partial X_i} \frac{\partial u_l}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_j}{\partial X_k} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_k}{\partial X_j} + 4\mu_2 \frac{\partial u_i}{\partial X_j} \frac{\partial u_l}{\partial X_j} \frac{\partial u_l}{\partial X_k} \right]
\end{aligned}$$

*What if i,j instead of k,k? symmetric?

Substituting in the assumed displacement field and v_i and noting that the subscript on q and the superscript on U must match each other and be different letters for multiplying terms, finally yields:

$$\begin{aligned}
&= (2\mu_1 + 4\mu_2) \frac{\partial U_k^{(m)}}{\partial X_k} \\
&\quad + \frac{\partial U_i^{(m)}}{\partial X_k} \left[(2\mu_1 + 8\mu_2) \frac{\partial q_n(t) U_i^{(n)}(X)}{\partial X_k} + 4\mu_2 \frac{\partial q_n(t) U_k^{(n)}(X)}{\partial X_i} + 4\mu_2 \frac{\partial q_n(t) U_l^{(n)}(X)}{\partial X_i} \frac{\partial q_p(t) U_l^{(p)}(X)}{\partial X_k} \right. \\
&\quad + 4\mu_2 \frac{\partial q_n(t) U_i^{(n)}(X)}{\partial X_j} \frac{\partial q_p(t) U_j^{(p)}(X)}{\partial X_k} + 4\mu_2 \frac{\partial q_n(t) U_i^{(n)}(X)}{\partial X_j} \frac{\partial q_p(t) U_k^{(p)}(X)}{\partial X_j} \\
&\quad \left. + 4\mu_2 \frac{\partial q_s(t) U_i^{(s)}(X)}{\partial X_j} \frac{\partial q_n(t) U_l^{(n)}(X)}{\partial X_j} \frac{\partial q_p(t) U_l^{(p)}(X)}{\partial X_k} \right]
\end{aligned}$$

$$\begin{aligned}
&= (2\mu_1 + 4\mu_2) \frac{\partial U_k^{(m)}}{\partial X_k} \\
&\quad + \frac{\partial U_i^{(m)}}{\partial X_k} \left[(2\mu_1 + 8\mu_2) q_n \frac{\partial U_i^{(n)}}{\partial X_k} + 4\mu_2 q_n \frac{\partial U_k^{(n)}}{\partial X_i} + 4\mu_2 q_n \frac{\partial U_l^{(n)}}{\partial X_i} q_p \frac{\partial U_l^{(p)}}{\partial X_k} + 4\mu_2 q_n \frac{\partial U_i^{(n)}}{\partial X_j} q_p \frac{\partial U_j^{(p)}}{\partial X_k} \right. \\
&\quad \left. + 4\mu_2 q_n \frac{\partial U_i^{(n)}}{\partial X_j} q_p \frac{\partial U_k^{(p)}}{\partial X_j} + 4\mu_2 q_s \frac{\partial U_i^{(s)}}{\partial X_j} q_n \frac{\partial U_l^{(n)}}{\partial X_j} q_p \frac{\partial U_l^{(p)}}{\partial X_k} \right] \\
&= (2\mu_1 + 4\mu_2) \frac{\partial U_k^{(m)}}{\partial X_k} + (2\mu_1 + 8\mu_2) \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_i^{(n)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_k^{(n)}}{\partial X_i} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_l^{(n)}}{\partial X_i} q_p \frac{\partial U_l^{(p)}}{\partial X_k} \\
&\quad + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_i^{(n)}}{\partial X_j} q_p \frac{\partial U_j^{(p)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_i^{(n)}}{\partial X_j} q_p \frac{\partial U_k^{(p)}}{\partial X_j} \\
&\quad + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} q_n \frac{\partial U_l^{(n)}}{\partial X_j} q_p \frac{\partial U_l^{(p)}}{\partial X_k}
\end{aligned}$$

1. Terms on single q :

$$\begin{aligned}
&(2\mu_1 + 8\mu_2) \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_i^{(n)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_k^{(n)}}{\partial X_i} \\
&= q_n \left((2\mu_1 + 8\mu_2) \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_k^{(n)}}{\partial X_i} \right)
\end{aligned}$$

2. Terms on double qq :

$$\begin{aligned}
&4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_l^{(n)}}{\partial X_i} q_p \frac{\partial U_l^{(p)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_i^{(n)}}{\partial X_j} q_p \frac{\partial U_j^{(p)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_n \frac{\partial U_i^{(n)}}{\partial X_j} q_p \frac{\partial U_k^{(p)}}{\partial X_j} \\
&= 4\mu_2 q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_l^{(n)}}{\partial X_i} \frac{\partial U_l^{(p)}}{\partial X_k} + \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \frac{\partial U_j^{(p)}}{\partial X_k} + \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \frac{\partial U_k^{(p)}}{\partial X_j} \right) \\
&= 4\mu_2 q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_l^{(n)}}{\partial X_i} \frac{\partial U_l^{(p)}}{\partial X_k} + \frac{\partial U_j^{(p)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} + \frac{\partial U_k^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \right)
\end{aligned}$$

3. Term on triple qqq :

$$\begin{aligned}
&4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} q_n \frac{\partial U_l^{(n)}}{\partial X_j} q_p \frac{\partial U_l^{(p)}}{\partial X_k} \\
&= 4\mu_2 q_s q_n q_p \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(s)}}{\partial X_j} \frac{\partial U_l^{(n)}}{\partial X_j} \frac{\partial U_l^{(p)}}{\partial X_k}
\end{aligned}$$

Looking at the terms inside the integral **C**:

$$\rho_0 v_i b_i^0$$

Substituting:

$$v_i = U_i^{(m)}$$

Yields:

$$= \rho_0 U_i^{(m)} b_i^0$$

Looking at the terms inside the integral **D**:

$$v_i t_i^0$$

Substituting:

$$v_i = U_i^{(m)}$$

Yields:

$$= U_i^{(m)} t_i^0$$

Finally, combining all bolded equations yields Equation (14):

$$M_{ij} \ddot{q}_j + D_{ij} \dot{q}_j + K_{ij}^{(1)} q_j + K_{ijl}^{(2)} q_j q_l + K_{ijkp}^{(3)} q_j q_l q_p = F_i$$

where

$$M_{mn} = \int_{\Omega_0} \rho_0 U_i^{(m)} U_i^{(n)} dX$$

$$K_{mn}^{(1)} = \int_{\Omega_0} (2\mu_1 + 8\mu_2) \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_k^{(n)}}{\partial X_i} dX$$

$$K_{mnp}^{(2)} = \int_{\Omega_0} 4\mu_2 \left(\frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_l^{(n)}}{\partial X_i} \frac{\partial U_l^{(p)}}{\partial X_k} + \frac{\partial U_j^{(p)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} + \frac{\partial U_k^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \right) dX$$

$$K_{msnp}^{(3)} = 4\mu_2 \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(s)}}{\partial X_j} \frac{\partial U_l^{(n)}}{\partial X_j} \frac{\partial U_l^{(p)}}{\partial X_k} dX$$

$$F_i = \int_{\Omega_0} (\rho_0 U_i^{(m)} b_i^0 - (2\mu_1 + 4\mu_2) \frac{\partial U_k^{(m)}}{\partial X_k}) dX + \int_{\partial\Omega_0^t} U_i^{(m)} t_i^0 ds$$

Note $D_{ij} \dot{q}_j$ has been added to match Mignolet's Equation (14). Look to his explanation.

Equation numbers referenced are from the paper.

The deformation gradient tensor F_{ij} defined in terms of the Kronecker delta δ_{ij} and displacement field u_i , a function of the position vector in the reference configuration X_j .

Equation (1):

$$F_{ij} = \delta_{ij} + \frac{\partial u_i}{\partial X_j}$$

Green strain tensor, E_{ij} , Equation (2):

$$\begin{aligned} E_{ij} &= \frac{1}{2} (F_{ki} F_{kj} - \delta_{ij}) \\ E_{ij} &= \frac{1}{2} \left[\left(\delta_{ki} + \frac{\partial u_k}{\partial X_i} \right) \left(\delta_{kj} + \frac{\partial u_k}{\partial X_j} \right) - \delta_{ij} \right] \\ E_{ij} &= \frac{1}{2} \left(\delta_{ki} \delta_{kj} + \delta_{kj} \frac{\partial u_k}{\partial X_i} + \delta_{ki} \frac{\partial u_k}{\partial X_j} + \frac{\partial u_k}{\partial X_i} \frac{\partial u_k}{\partial X_j} - \delta_{ij} \right) \end{aligned}$$

Using the substitution property of the Kronecker delta, where if a subscript on the Kronecker delta matches a subscript on what it is operating on, replace the matching subscript with the other subscript and drop the Kronecker delta.

Example:

$$\delta_{ij} u_j = u_i$$

Therefore,

$$\begin{aligned} E_{ij} &= \frac{1}{2} \left(\delta_{ij} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_i}{\partial X_j} + \frac{\partial u_k}{\partial X_i} \frac{\partial u_k}{\partial X_j} - \delta_{ij} \right) \\ E_{ij} &= \frac{1}{2} \left(\frac{\partial u_j}{\partial X_i} + \frac{\partial u_i}{\partial X_j} + \frac{\partial u_k}{\partial X_i} \frac{\partial u_k}{\partial X_j} \right) \end{aligned}$$

Adopting a linear elastic model between the Green strain and second Piola-Kirchhoff stress, S_{ij} , tensors yields Equation (8):

$$\begin{aligned} S_{ij} &= C_{ijkl} E_{kl} \\ S_{ij} &= \frac{1}{2} C_{ijkl} \left(\frac{\partial u_l}{\partial X_k} + \frac{\partial u_k}{\partial X_l} + \frac{\partial u_r}{\partial X_k} \frac{\partial u_r}{\partial X_l} \right) \end{aligned}$$

Equation (12) with **A**, **B**, **C**, and **D** labels for breaking up the sections:

$$\underbrace{\int_{\Omega_0} \rho_0 v_i \ddot{u}_i dX}_{\mathbf{A}} + \underbrace{\int_{\Omega_0} \frac{\partial v_i}{\partial X_k} (F_{ij} S_{jk}) dX}_{\mathbf{B}} = \underbrace{\int_{\Omega_0} \rho_0 v_i b_i^0 dX}_{\mathbf{C}} + \underbrace{\int_{\partial\Omega_0^t} v_i t_i^0 ds}_{\mathbf{D}}$$

Looking at the terms inside the integral **A**:

$$\rho_0 v_i \ddot{u}_i$$

Substituting assumed displacement field Equation (13):

$$u_i(X, t) = \sum_{n=1}^M q_n(t) U_i^{(n)}(X)$$

And:

$$v_i = U_i^{(m)}$$

Yields:

$$\begin{aligned} &= \rho_0 U_i^{(m)} \frac{\partial^2 (q_n(t) U_i^{(n)}(X))}{\partial t^2} \\ &= \rho_0 U_i^{(m)} U_i^{(n)} \ddot{q}_n \end{aligned}$$

Looking at the terms inside the integral **B**:

$$\begin{aligned} \frac{\partial v_i}{\partial X_k} (F_{ij} S_{jk}) &= \frac{1}{2} \frac{\partial v_i}{\partial X_k} \left[\left(\delta_{ij} + \frac{\partial u_i}{\partial X_j} \right) C_{jklw} \left(\frac{\partial u_w}{\partial X_l} + \frac{\partial u_l}{\partial X_w} + \frac{\partial u_r}{\partial X_l} \frac{\partial u_r}{\partial X_w} \right) \right] \\ &= \frac{1}{2} \frac{\partial v_i}{\partial X_k} \left[\delta_{ij} C_{jklw} \left(\frac{\partial u_w}{\partial X_l} + \frac{\partial u_l}{\partial X_w} + \frac{\partial u_r}{\partial X_l} \frac{\partial u_r}{\partial X_w} \right) + \frac{\partial u_i}{\partial X_j} C_{jklw} \left(\frac{\partial u_w}{\partial X_l} + \frac{\partial u_l}{\partial X_w} + \frac{\partial u_r}{\partial X_l} \frac{\partial u_r}{\partial X_w} \right) \right] \\ &= \frac{1}{2} \frac{\partial v_i}{\partial X_k} \left[C_{iklw} \left(\frac{\partial u_w}{\partial X_l} + \frac{\partial u_l}{\partial X_w} + \frac{\partial u_r}{\partial X_l} \frac{\partial u_r}{\partial X_w} \right) + \frac{\partial u_i}{\partial X_j} C_{jklw} \left(\frac{\partial u_w}{\partial X_l} + \frac{\partial u_l}{\partial X_w} + \frac{\partial u_r}{\partial X_l} \frac{\partial u_r}{\partial X_w} \right) \right] \end{aligned}$$

Substituting assumed displacement field Equation (13):

$$u_i(X, t) = \sum_{n=1}^M q_n(t) U_i^{(n)}(X)$$

And:

$$v_i = U_i^{(m)}$$

And noting that the subscript on q and the superscript on U must match each other and be different letters for multiplying terms, finally yields:

$$\begin{aligned} &= \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} \left[C_{iklw} \left(\frac{\partial q_n(t) U_w^{(n)}(X)}{\partial X_l} + \frac{\partial q_n(t) U_l^{(n)}(X)}{\partial X_w} + \frac{\partial q_n(t) U_r^{(n)}(X)}{\partial X_l} \frac{\partial q_p(t) U_r^{(p)}(X)}{\partial X_w} \right) \right. \\ &\quad \left. + \frac{\partial q_s(t) U_i^{(s)}(X)}{\partial X_j} C_{jklw} \left(\frac{\partial q_n(t) U_w^{(n)}(X)}{\partial X_l} + \frac{\partial q_n(t) U_l^{(n)}(X)}{\partial X_w} + \frac{\partial q_n(t) U_r^{(n)}(X)}{\partial X_l} \frac{\partial q_p(t) U_r^{(p)}(X)}{\partial X_w} \right) \right] \\ &= \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} \left[C_{iklw} \left(q_n \frac{\partial U_w^{(n)}}{\partial X_l} + q_n \frac{\partial U_l^{(n)}}{\partial X_w} + q_n \frac{\partial U_r^{(n)}}{\partial X_l} q_p \frac{\partial U_r^{(p)}}{\partial X_w} \right) \right. \\ &\quad \left. + q_s \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} \left(q_n \frac{\partial U_w^{(n)}}{\partial X_l} + q_n \frac{\partial U_l^{(n)}}{\partial X_w} + q_n \frac{\partial U_r^{(n)}}{\partial X_l} q_p \frac{\partial U_r^{(p)}}{\partial X_w} \right) \right] \end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} q_n \frac{\partial U_w^{(n)}}{\partial X_l} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} q_n \frac{\partial U_l^{(n)}}{\partial X_w} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} q_n \frac{\partial U_r^{(n)}}{\partial X_l} q_p \frac{\partial U_r^{(p)}}{\partial X_w} \\
&\quad + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} q_n \frac{\partial U_w^{(n)}}{\partial X_l} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} q_n \frac{\partial U_l^{(n)}}{\partial X_w} \\
&\quad + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} q_n \frac{\partial U_r^{(n)}}{\partial X_l} q_p \frac{\partial U_r^{(p)}}{\partial X_w}
\end{aligned}$$

1. Terms on single q :

$$\begin{aligned}
&\frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} q_n \frac{\partial U_w^{(n)}}{\partial X_l} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} q_n \frac{\partial U_l^{(n)}}{\partial X_w} \\
&= q_n \left(\frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_w^{(n)}}{\partial X_l} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_l^{(n)}}{\partial X_w} \right) \\
&= q_n \left(\frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{ikwl} \frac{\partial U_l^{(n)}}{\partial X_w} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_l^{(n)}}{\partial X_w} \right)
\end{aligned}$$

Using Equation (9):

$$C_{ikwl} = C_{iklw}$$

Yields:

$$\begin{aligned}
&= q_n \left(\frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_l^{(n)}}{\partial X_w} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_l^{(n)}}{\partial X_w} \right) \\
&= q_n \left(\frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_l^{(n)}}{\partial X_w} \right)
\end{aligned}$$

Finally, substituting $w=p$ to match Equation 16:

$$= q_n \left(\frac{\partial U_i^{(m)}}{\partial X_k} C_{iklp} \frac{\partial U_l^{(n)}}{\partial X_p} \right)$$

2. Terms on double qq :

$$\begin{aligned}
&\frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} q_n \frac{\partial U_r^{(n)}}{\partial X_l} q_p \frac{\partial U_r^{(p)}}{\partial X_w} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} q_n \frac{\partial U_w^{(n)}}{\partial X_l} + \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} q_n \frac{\partial U_l^{(n)}}{\partial X_w} \\
&= \frac{1}{2} q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} \right) + \frac{1}{2} q_s q_n \left(\frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} \frac{\partial U_w^{(n)}}{\partial X_l} + \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} \frac{\partial U_l^{(n)}}{\partial X_w} \right)
\end{aligned}$$

substituting $s=p$

$$= \frac{1}{2} q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} \right) + \frac{1}{2} q_p q_n \left(\frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(p)}}{\partial X_j} C_{jklw} \frac{\partial U_w^{(n)}}{\partial X_l} + \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(p)}}{\partial X_j} C_{jklw} \frac{\partial U_l^{(n)}}{\partial X_w} \right)$$

Using rules from CIVE 662 HW1 #3:

$$u_i v_j S_{jk} w_k = w_k S_{kj} v_j u_i$$

And using transpose rule of a fourth-order tensor:

$$(A_{ijkl})^T = A_{klij}$$

Yields:

$$= \frac{1}{2} q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} \right) + \frac{1}{2} q_p q_n \left(\frac{\partial U_w^{(n)}}{\partial X_l} C_{lwjk} \frac{\partial U_i^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} + \frac{\partial U_l^{(n)}}{\partial X_w} C_{lwjk} \frac{\partial U_i^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} \right)$$

Finally, some algebraic manipulations to match the indices in equations given by Mignolet.

Combine terms on $q_n q_p$:

$$= \frac{1}{2} q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} + \frac{\partial U_w^{(n)}}{\partial X_l} C_{lwjk} \frac{\partial U_i^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} + \frac{\partial U_l^{(n)}}{\partial X_w} C_{lwjk} \frac{\partial U_i^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} \right)$$

Switch n, m and p in second term (dummy indices, therefore switching doesn't affect the summation):

$$= \frac{1}{2} q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_k} C_{iklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} + \frac{\partial U_w^{(p)}}{\partial X_l} C_{lwjk} \frac{\partial U_i^{(m)}}{\partial X_j} \frac{\partial U_i^{(n)}}{\partial X_k} + \frac{\partial U_l^{(n)}}{\partial X_w} C_{lwjk} \frac{\partial U_i^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} \right)$$

Using Equation (9) on second term:

$$C_{lwjk} = C_{wljk}$$

Switch indices on all terms to match C_{ijkl} (just switching letters, not order):

$$= \frac{1}{2} q_n q_p \left(\frac{\partial U_i^{(m)}}{\partial X_j} C_{ijkl} \frac{\partial U_r^{(n)}}{\partial X_k} \frac{\partial U_r^{(p)}}{\partial X_l} + \frac{\partial U_i^{(p)}}{\partial X_j} C_{ijkl} \frac{\partial U_r^{(m)}}{\partial X_k} \frac{\partial U_r^{(n)}}{\partial X_l} + \frac{\partial U_i^{(n)}}{\partial X_j} C_{ijkl} \frac{\partial U_r^{(p)}}{\partial X_k} \frac{\partial U_r^{(m)}}{\partial X_l} \right)$$

3. Term on triple qqq :

$$\begin{aligned} & \frac{1}{2} \frac{\partial U_i^{(m)}}{\partial X_k} q_s \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} q_n \frac{\partial U_r^{(n)}}{\partial X_l} q_p \frac{\partial U_r^{(p)}}{\partial X_w} \\ &= \frac{1}{2} q_s q_n q_p \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(s)}}{\partial X_j} C_{jklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} \\ &= \frac{1}{2} q_s q_n q_p \frac{\partial U_i^{(m)}}{\partial X_j} \frac{\partial U_i^{(s)}}{\partial X_k} C_{kjlw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} \end{aligned}$$

Using Equation (9):

$$\begin{aligned} & C_{kjlw} = C_{jklw} \\ &= \frac{1}{2} q_s q_n q_p \frac{\partial U_i^{(m)}}{\partial X_j} \frac{\partial U_i^{(s)}}{\partial X_k} C_{jklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} \end{aligned}$$

Looking at the terms inside the integral $\mathbf{C}$:

$$\rho_0 v_i b_i^0$$

Substituting:

$$v_i = U_i^{(m)}$$

Yields:

$$= \rho_0 U_i^{(m)} b_i^0$$

Looking at the terms inside the integral **D**:

$$v_i t_i^0$$

Substituting:

$$v_i = U_i^{(m)}$$

Yields:

$$= U_i^{(m)} t_i^0$$

Finally, combining all bolded equations yields Equation (14):

$$M_{ij} \ddot{q}_j + D_{ij} \dot{q}_j + K_{ij}^{(1)} q_j + K_{ijl}^{(2)} q_j q_l + K_{ijkp}^{(3)} q_j q_l q_p = F_i$$

where

$$M_{mn} = \int_{\Omega_0} \rho_0 U_i^{(m)} U_i^{(n)} dX$$

$$K_{mn}^{(1)} = \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklp} \frac{\partial U_l^{(n)}}{\partial X_p} dX$$

$$K_{mnp}^{(2)} = \frac{1}{2} [\hat{K}_{mnp}^2 + \hat{K}_{pmn}^2 + \hat{K}_{nmp}^2]$$

$$\hat{K}_{mnp}^{(2)} = \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_j} C_{ijkl} \frac{\partial U_r^{(n)}}{\partial X_k} \frac{\partial U_r^{(p)}}{\partial X_l} dX$$

$$K_{msnp}^{(3)} = \frac{1}{2} \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_j} \frac{\partial U_i^{(s)}}{\partial X_k} C_{jklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} dX$$

$$F_i = \int_{\Omega_0} \rho_0 U_i^{(m)} b_i^0 dX + \int_{\partial \Omega_0^t} U_i^{(m)} t_i^0 ds$$

Note $D_{ij} \dot{q}_j$ has been added to match Mignolet's Equation (14). Look to his explanation.

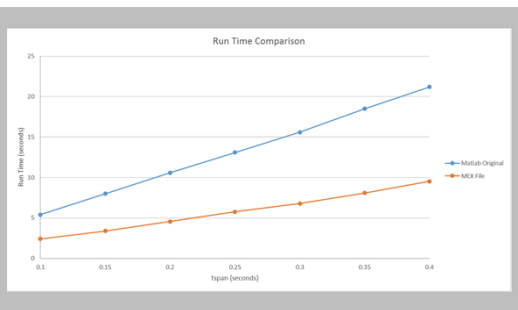
$$\int_{\Omega_0} \rho_0 v_i \ddot{u}_i dX + \int_{\Omega_0} \frac{\partial v_i}{\partial X_k} (F_{ij} S_{jk}) dX = \int_{\Omega_0} \rho_0 v_i b_i^0 dX + \int_{\partial\Omega_0^f} v_i t_i^0 ds$$

$$S_{ij} = C_{ijkl} E_{kl}$$

$$F_{ij} = \delta_{ij} + \frac{\partial u_i}{\partial X_j}$$

$$S_{ij} = 2 \frac{\partial \Psi}{\partial C_{ij}} = 2 \frac{\partial \Psi}{\partial I_1} \frac{\partial I_1}{\partial C_{ij}} + 2 \frac{\partial \Psi}{\partial I_2} \frac{\partial I_2}{\partial C_{ij}} + 2 \frac{\partial \Psi}{\partial I_3} \frac{\partial I_3}{\partial C_{ij}}$$

$$M_{ij} \ddot{q}_j + D_{ij} \dot{q}_j + K_{ij}^{(1)} q_j + K_{ijl}^{(2)} q_j q_l + K_{ijkp}^{(3)} q_j q_l q_p = F_i$$



Kirsten's Summer at Sandia

Kirsten Peterson

Sandia National Laboratories

Presented at 1556 Group Meeting on July 20, 2016.



*Exceptional
service
in the
national
interest*



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.

My Background

- Graduated high school: Plato, MO
May 2008
- B.S. Civil Engineering: Embry Riddle Aeronautical University
May 2012
- M.S. Civil Engineering: Colorado State University
May 2014
 - Thesis: The Mechanics of Plastic-Aluminum Composite I-Beams
- Ph.D. Civil Engineering: Colorado State University
2017
 - Dissertation: Discrete Element Model of Damaged Soil Particles and Post-Fire Ground Treatments for Protection of Critical Transportation Structures

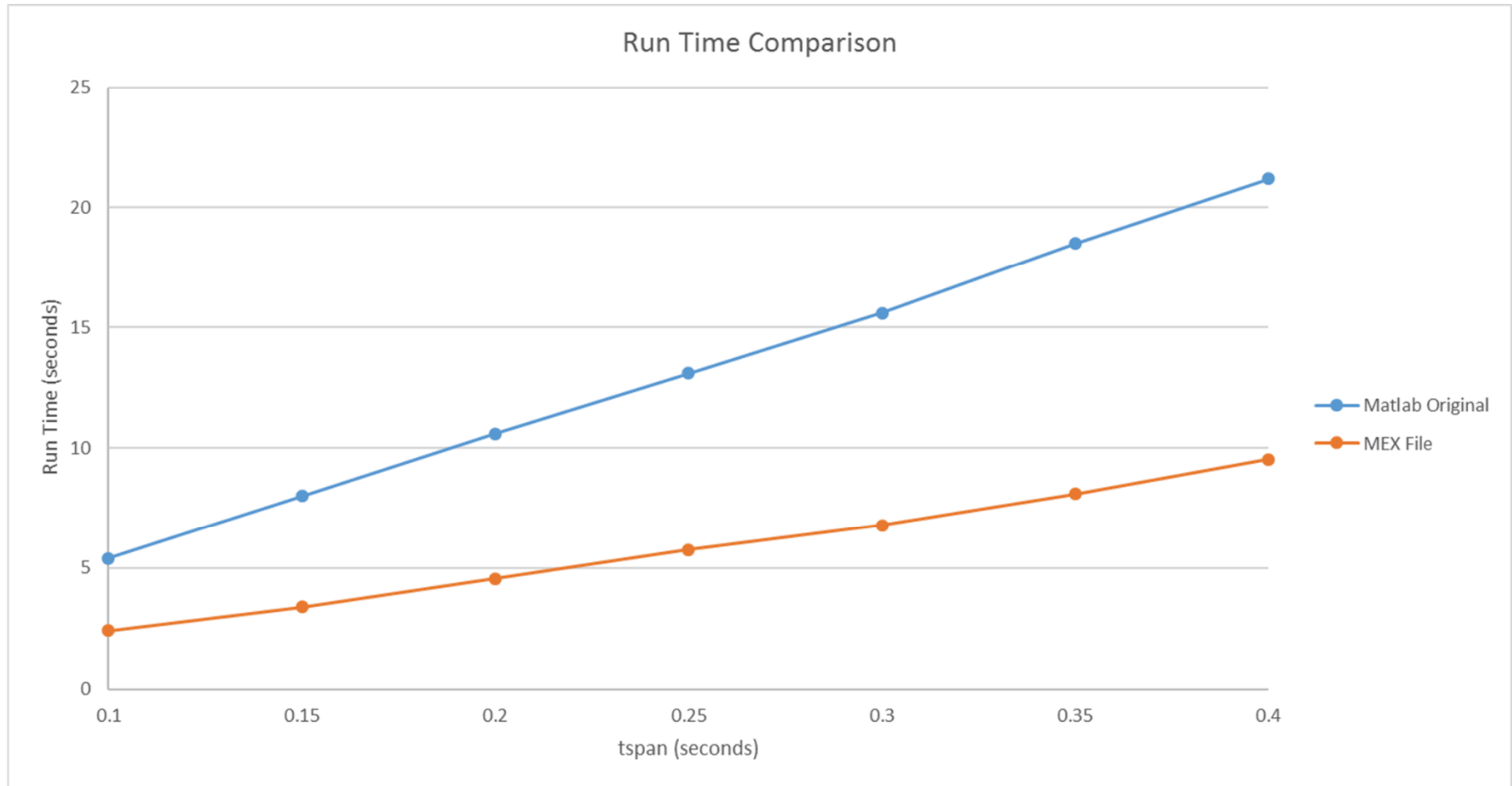
Summer Work Overview

- MEX Files
 - Introduction
 - Run time comparisons for linear viscoelastic reduced order models (ROMs)
- Derive equations for ROMs incorporating material nonlinearity
 - Show Mignolet et. al. findings for linear elastic material
 - Derive ROM equations for hyperelastic material models
 - Mooney-Rivlin model
 - Neo-Hookean model

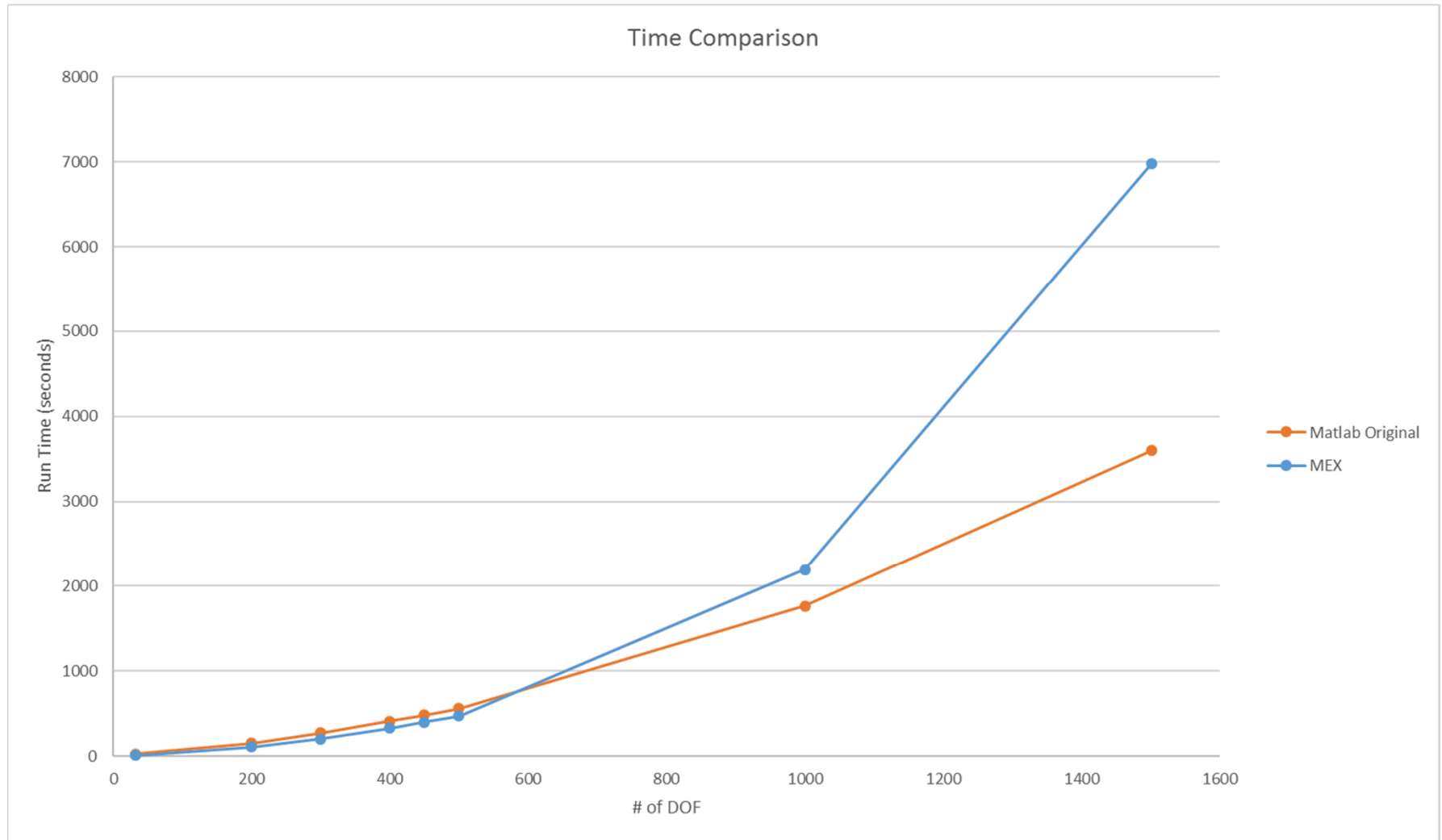
MEX Files Introduction

- What are they?
 - Dynamically linked subroutines that the MATLAB interpreter loads and executes
- MEX files allow you to call your own C, C++, or Fortran subroutines from the MATLAB command line and .m scripts as if they were built-in functions
- Why are they useful?
 - Repurpose functions or subroutines you already have written to be used from Matlab
 - Use a better tool for computationally expensive Matlab functions
- Tutorial for MEX files written in C++
 - Get from Kevin or Rob if you're interested (or from me by August 4th)

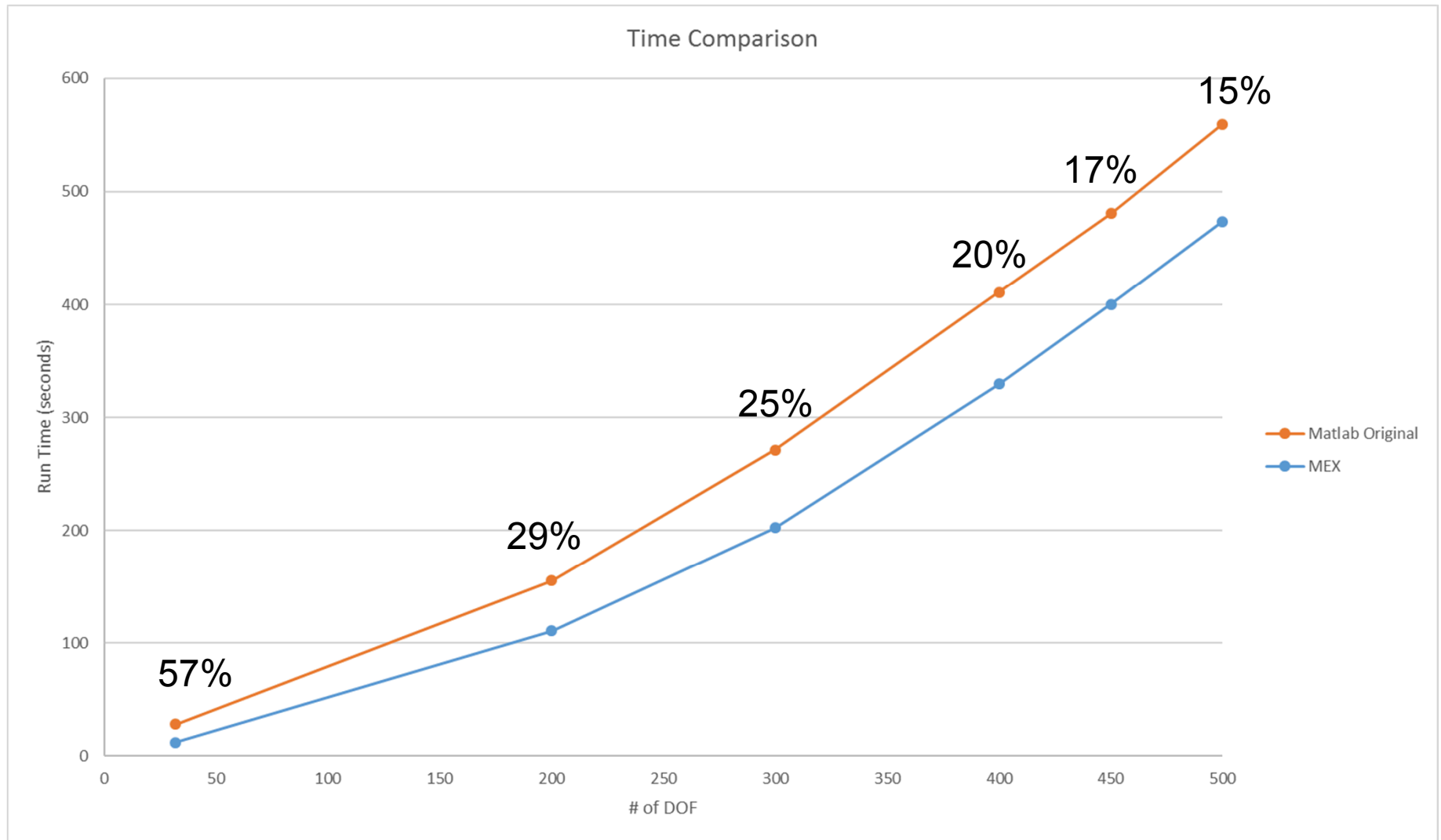
Runtime Results – 32 DOF system



Runtime Results cont...



Runtime Results cont...



Show Mignolet et. al. findings for linear elastic material

- Deformation Gradient

$$F_{ij} = \delta_{ij} + \frac{\partial u_i}{\partial X_j}$$

- Green Strain Tensor

$$E_{ij} = \frac{1}{2}(F_{ki}F_{kj} - \delta_{ij})$$

- 2nd Piola-Kirchhoff Stress

$$S_{ij} = C_{ijkl}E_{kl}$$

- Weak formulation

$$\int_{\Omega_0} \rho_0 v_i \ddot{u}_i dX + \int_{\Omega_0} \frac{\partial v_i}{\partial X_k} (F_{ij} S_{jk}) dX = \int_{\Omega_0} \rho_0 v_i b_i^0 dX + \int_{\partial\Omega_0^t} v_i t_i^0 ds$$

- Assumed displacement field

$$u_i(X, t) = \sum_{n=1}^M q_n(t) U_i^{(n)}(X)$$

- Weighting function

$$v_i = U_i^{(m)}$$

- “After some algebraic manipulations”

$$M_{ij}\ddot{q}_j + D_{ij}\dot{q}_j + K_{ij}^{(1)}q_j + K_{ijl}^{(2)}q_jq_l + K_{ijkp}^{(3)}q_jq_lq_p = F_i$$

$$M_{mn} = \int_{\Omega_0} \rho_0 U_i^{(m)} U_i^{(n)} dX$$

$$K_{mn}^{(1)} = \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklp} \frac{\partial U_l^{(n)}}{\partial X_p} dX$$

$$K_{mnp}^{(2)} = \frac{1}{2} [\hat{K}_{mnp}^2 + \hat{K}_{pmn}^2 + \hat{K}_{nmp}^2]$$

$$\hat{K}_{mnp}^{(2)} = \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_j} C_{ijkl} \frac{\partial U_r^{(n)}}{\partial X_k} \frac{\partial U_r^{(p)}}{\partial X_l} dX$$

$$K_{msnp}^{(3)} = \frac{1}{2} \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_j} \frac{\partial U_i^{(s)}}{\partial X_k} C_{jklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} dX$$

$$F_i = \int_{\Omega_0} \rho_0 U_i^{(m)} b_i^0 dX + \int_{\partial\Omega_0^t} U_i^{(m)} t_i^0 ds$$

Derive ROM equations for hyperelastic material models

- 2nd Piola-Kirchhoff Stress
$$S_{ij} = 2 \frac{\partial \Psi}{\partial C_{ij}} = 2 \frac{\partial \Psi}{\partial I_1} \frac{\partial I_1}{\partial C_{ij}} + 2 \frac{\partial \Psi}{\partial I_2} \frac{\partial I_2}{\partial C_{ij}} + 2 \frac{\partial \Psi}{\partial I_3} \frac{\partial I_3}{\partial C_{ij}}$$

- Strain Energy (Mooney-Rivlin)
$$\Psi(\mathbf{C}) = \mu_1(I_1 - 3) + \mu_2(I_2 - 3)$$

- Right Cauchy Green Deformation Tensor
$$C_{ij} = F_{li} F_{lj}$$

- With:
$$\frac{\partial I_1}{\partial C_{ij}} = \frac{\partial C_{kk}}{\partial C_{ij}} = \delta_{ij}$$
$$\frac{\partial I_2}{\partial C_{ij}} = \frac{\partial C_{kl} C_{kl}}{\partial C_{ij}} = 2C_{ij}$$

- Yields:
$$S_{ij} = 2\mu_1 \delta_{ij} + 4\mu_2 \left(\delta_{ij} + \frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_l}{\partial X_i} \frac{\partial u_l}{\partial X_j} \right)$$

$$K_{mn}^{(1)} = \int_{\Omega_0} (2\mu_1 + 8\mu_2) \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_k} + 4\mu_2 \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_k^{(n)}}{\partial X_i} dX$$

VS.
$$K_{mn}^{(1)} = \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_k} C_{iklp} \frac{\partial U_l^{(n)}}{\partial X_p} dX$$

$$K_{mnp}^{(2)} = \int_{\Omega_0} 4\mu_2 \left(\frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_l^{(n)}}{\partial X_i} \frac{\partial U_l^{(p)}}{\partial X_k} + \frac{\partial U_j^{(p)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} + \frac{\partial U_k^{(p)}}{\partial X_j} \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(n)}}{\partial X_j} \right) dX$$

VS.
$$K_{mnp}^{(2)} = \frac{1}{2} [\hat{K}_{mnp}^2 + \hat{K}_{pmn}^2 + \hat{K}_{npm}^2]$$

$$\hat{K}_{mnp}^{(2)} = \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_j} C_{ijkl} \frac{\partial U_r^{(n)}}{\partial X_k} \frac{\partial U_r^{(p)}}{\partial X_l} dX$$

$$K_{msnp}^{(3)} = 4\mu_2 \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_k} \frac{\partial U_i^{(s)}}{\partial X_j} \frac{\partial U_l^{(n)}}{\partial X_j} \frac{\partial U_l^{(p)}}{\partial X_k} dX$$

VS.
$$K_{msnp}^{(3)} = \frac{1}{2} \int_{\Omega_0} \frac{\partial U_i^{(m)}}{\partial X_j} \frac{\partial U_i^{(s)}}{\partial X_k} C_{jklw} \frac{\partial U_r^{(n)}}{\partial X_l} \frac{\partial U_r^{(p)}}{\partial X_w} dX$$

Last couple weeks

- Formalize Mooney-Rivlin and Neo-Hookean derivations
- Publication rough draft
- MEX file in Fortran

QUESTIONS?

MEX Files in Matlab Tutorial

This tutorial describes how to write MEX files in C++ coding language, compile them, and call them from the Matlab command line and/or a .m script. This tutorial borrows some explanations from online Matlab documentation found here:

http://www.mathworks.com/help/matlab/matlab_external/introducing-mex-files.html

http://www.mathworks.com/help/matlab/matlab_external/standalone-example.html

What are MEX files?

These programs, called binary MEX files, are dynamically linked subroutines that the MATLAB interpreter loads and executes.

Why use MEX files?

MEX files allow you to call your own C, C++, or Fortran subroutines from the MATLAB command line and .m scripts as if they were built-in functions.

How to write a MEX file in C++

This section will describe how to create a C++ source MEX file. Text highlighted in yellow is required text in the code and text highlighted in red will need to be edited by the user.

1. Add desired header section:

```
/*  
* This is a MEX file for MATLAB.  
*/
```

2. Add the C++ header file, mex.h, containing the MATLAB API function declarations:

```
#include "mex.h"
```

3. Create Gateway Routine: Every C++ program has a main() function. MATLAB uses the gateway routine, mexFunction, as the entry point to the function. Add the following mexFunction code:

```
/* The gateway function */  
void mexFunction(int nlhs, mxArray *plhs[],  
                 int nrhs, const mxArray *prhs[])  
{
```

with the mexFunction parameters described here:

Parameter	Description
<u>nlhs</u>	Number of output (left-side) arguments, or the size of the <u>plhs</u> array.
<u>plhs</u>	Array of output arguments.
<u>nrhs</u>	Number of input (right-side) arguments, or the size of the <u>prhs</u> array.
<u>prhs</u>	Array of input arguments.

4. Declare variables for computational routine:

```
/* variable declarations here */
```

5. Verify MEX File Input and Output Parameters:

```
if(nrhs != # of Inputs) {
    mexErrMsgTxt("Incorrect number of inputs!");
}

if(nlhs != # of Outputs) {
    mexErrMsgTxt("Incorrect number of outputs!");
}
```

Note: there are other verifications you can perform, such as verifying the inputs are of correct type (scalar, double, row vector, etc.). However, the code syntax isn't included here. It can easily be found online.

6. Read the input data:

To read a scalar input, use the `mxGetScalar` function:

```
InputVariableName = mxGetScalar(prhs[0]);
```

Use the `mxGetPr` function to point to the input matrix data:

```
InputVariableName = mxGetPr(prhs[1]);
```

Continue using `mxGetScalar` and `mxGetPr` until all input data has been read in:

```
InputVariableName = mxGetPr(prhs[2]);
InputVariableName = mxGetScalar(prhs[3]);
InputVariableName = mxGetPr(prhs[4]);
InputVariableName = mxGetPr(prhs[5]);
InputVariableName = mxGetScalar(prhs[etc...]);
etc...
```

7. Assign variables to the outputs:

```
/* create the output matrices */  
plhs[0] = mxCreateDoubleMatrix(nrows,ncols,mxREAL);  
plhs[1] = mxCreateDoubleMatrix(nrows,ncols,mxREAL);  
plhs[2] = mxCreateDoubleMatrix(nrows,ncols,mxREAL);  
plhs[etc...] = mxCreateDoubleMatrix(nrows,ncols,mxREAL);  
etc...
```

```
/* get a pointer to the real data in the output matrices */  
OutputVariableName = mxGetPr(plhs[0]);  
OutputVariableName = mxGetPr(plhs[1]);  
OutputVariableName = mxGetPr(plhs[2]);  
OutputVariableName = mxGetPr(plhs[etc...]);  
etc...
```

8. Code your MEX file. This is the longest step of writing your MEX file. Write your C++ code here, or make a call to a function you already have written.
9. Close Gateway Function Routine:

```
// End Mex gateway routine
```

10. Save the MEX file!

How to compile a MEX file

Now that the MEX file is written and saved, it is ready to be compiled. Compile the MEX file from the Matlab command window with:

```
mex MEXfilename.C
```

If everything is correct in the MEX file Matlab will tell you that it completed successfully. For example:

```
>> mex romviscoSB.C  
Building with 'gcc'.  
MEX completed successfully.  
>>
```

How to call a MEX file

MEX files can be called from either the Matlab command window or a .m script. To call a MEX file, use the name of the file, without the file extension:

```
[OutVariable1,OutVariable2,etc...] = MEXfilename(InputVariable1,InputVariable2,etc...);
```

Other useful functions:

mxGetN

Use the `mxGetN` function to get the size of an input matrix:

```
VariableName = mxGetN(prhs[1]);
```

Call to a Matlab function from MEX file

```
/* Define needed variables for mexCallMATLAB "matlabfilename.m" */
const char *matlabfilename; //(NOTE: do not put .m extension here)
mxArray *prhs2[NumberOfInputs], *plhs2[NumberOfOutputs];
// Define inputs
prhs2[0] = mxCreateDoubleMatrix(nrows,ncols,mxREAL);
prhs2[1] = mxCreateDoubleMatrix(nrows,ncols,mxREAL);
prhs2[etc...] = mxCreateDoubleMatrix(nrows,ncols,mxREAL);
// Assign inputs
InputVariableName = mxGetPr(prhs2[0]);
InputVariableName = mxGetPr(prhs2[1]);
InputVariableName = mxGetPr(prhs2[etc...]);
// Call matlab .m file from MEX file
mexCallMATLAB(NumberOfOutputs, plhs2, NumberOfInputs, prhs2, "matlabfilename");
// Assign outputs
OutputVariableName = (double *)mxGetPr(plhs2[0]);
OutputVariableName = (double *)mxGetPr(plhs2[1]);
OutputVariableName = (double *)mxGetPr(plhs2[etc...]);
```

Note to Users

Depending on your local Matlab account settings and compiler, it may be necessary for you to run MEX files only from Matlab on your blade account. If you get the following error when trying to compile your MEX file:

```
>> mex MEXfilename.C
```


Warning: Xcode appears to be installed, but it has not been configured. Run "osascript -e 'do shell script "xcode-select -switch

/Applications/Xcode.app" with administrator privileges"' to configure Xcode. You must be an administrator of this computer to complete this action.

Error using mex

No supported compiler or SDK was found. For options, visit <http://www.mathworks.com/support/compilers/R2014b/maci64.html>.

Then you don't have the required compiler on your computer and you need to connect to your blade account.

Debugging

MEX files don't print to Matlab's command window live (as the calculations are being made). Instead, everything that gets printed to the screen in the MEX file code is stored and printed to Matlab's command window after the MEX file sends the results back to Matlab. Therefore, if the MEX file crashes and causes a segmentation fault in Matlab, it can be difficult to pinpoint exactly what line in the code is causing the crash. The following lines of code will allow you to create an output file and write to it even if the MEX file crashes and you haven't closed the output file yet.

Near beginning of code:

```
// Create output file
```

```
FILE *name = fopen("outputfilename.txt","w"); //w is for write only
```

Where you want to see if it's crashing:

```
// Write to output file
```

```
fprintf(name, "Your code is doing great and made it to this line. \n");
```

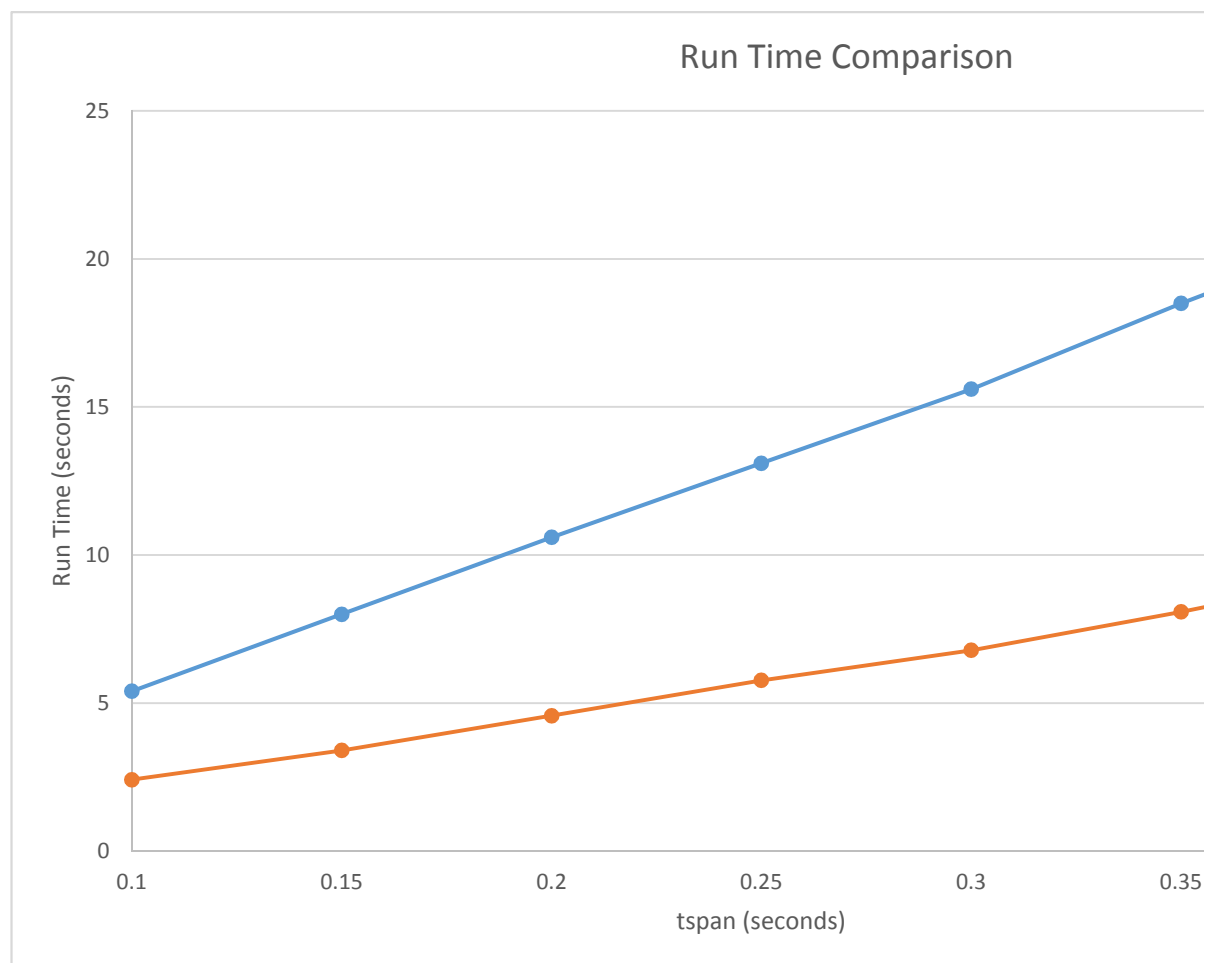
```
fflush(name);
```

Make sure the "fflush" command is written directly after your "fprintf" command, because "fflush" pushes the printed text to the output file from the buffer. If the code crashes before "fflush" then anything written since the last "fflush" won't be in the output file and it will be harder to track down the problem.

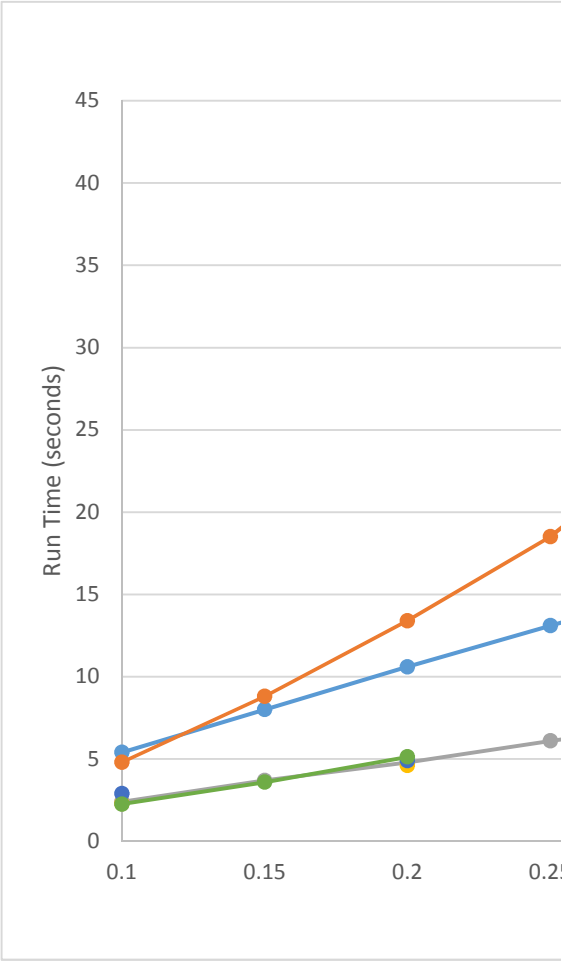
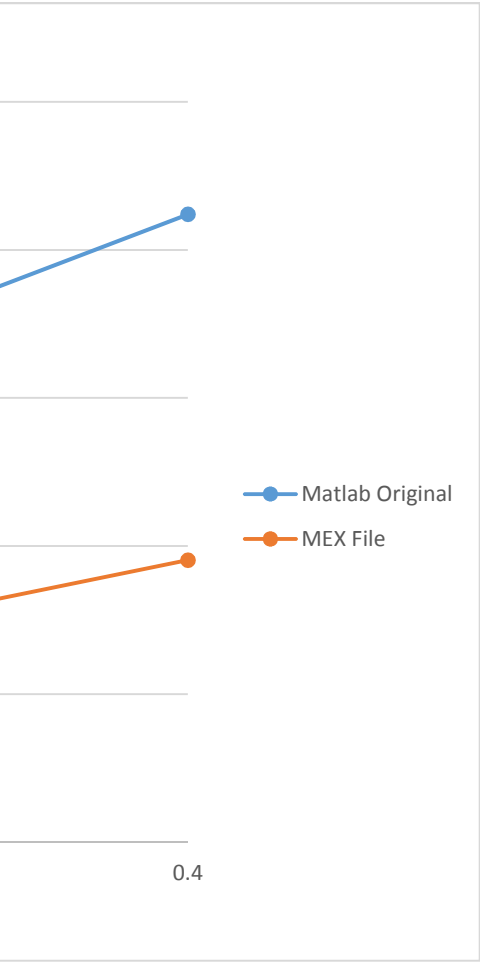
Number of time steps		Matlab original	Mex w/ .m call	Mex w/ C++ cal	romviscoSBts.C
	tspan	run time (s)	run time (s)	run time (s)	run time (s) (tskip=1)
20000	0.1	5.4	4.8	2.4	2.4
30000	0.15	8.0	8.8	3.7	3.4
40000	0.2	10.6	13.4	4.8	4.6
50000	0.25	13.1	18.5	6.1	5.8
60000	0.3	15.6	25.5	7.3	6.8
70000	0.35	18.5	32.5	8.8	8.1
80000	0.4	21.2	41.7	10.1	9.5
	0.5			12.4	

Initial time
3.3 - 3.5 seconds

*viscolnt C++ ca *viscolnt and t_ext C++ calls



Mex w/ new storage (romviscoSBts.C)	Mex calling to C++ f_ext once before time stepping
run time (s) (tskip=100)	no difference compared to Mex w/ C++ call
2.3	2.6
3.6	
5.1	

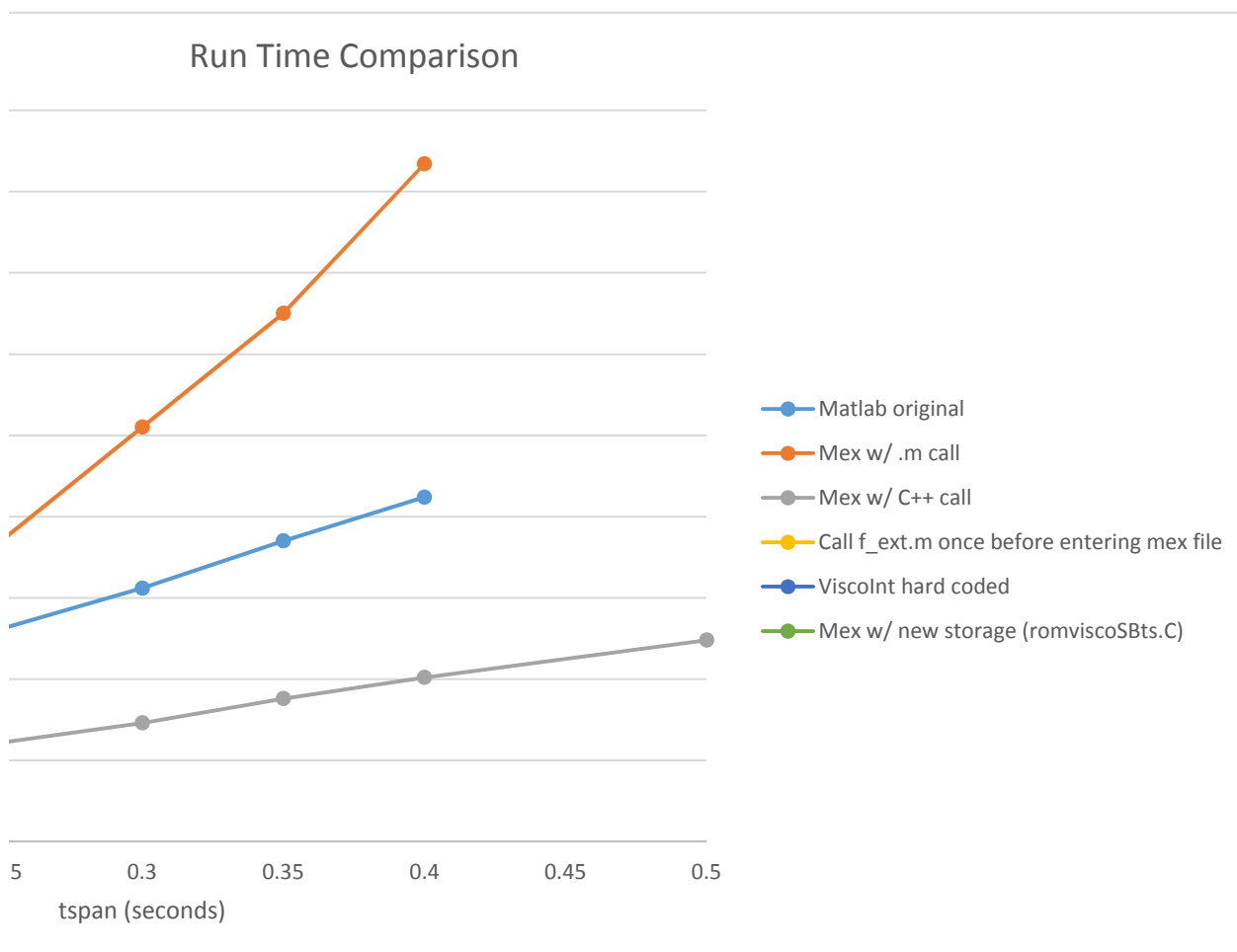


Call f_ext.m once before entering mex file Viscolnt hard coded

*Changes initial time to 8.3 seconds for tsp code file: romviscolnt.C

2.3 2.9

4.6 4.9



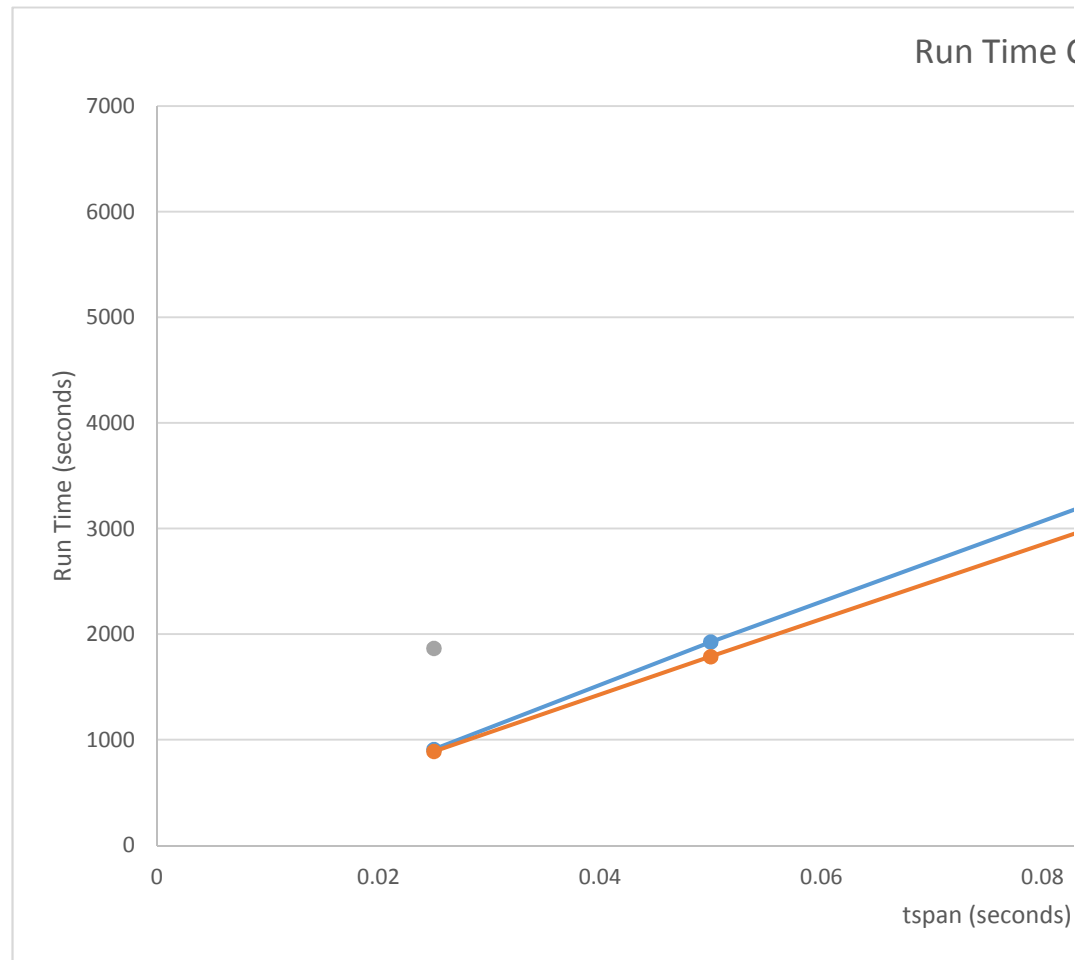
# of time steps	tspan	Matlab original	Matlab w/ changes	Matlab w/ changes	Mex w/ .m call
		run time (s)	tskip = 100	tskip = 200	run time (s)
25000	0.025	910	890	903	
50000	0.05	1927	1788	*wasn't always hitting peaks	
100000	0.1	3835	3559		
150000	0.15	5831	5330		

63.90833333

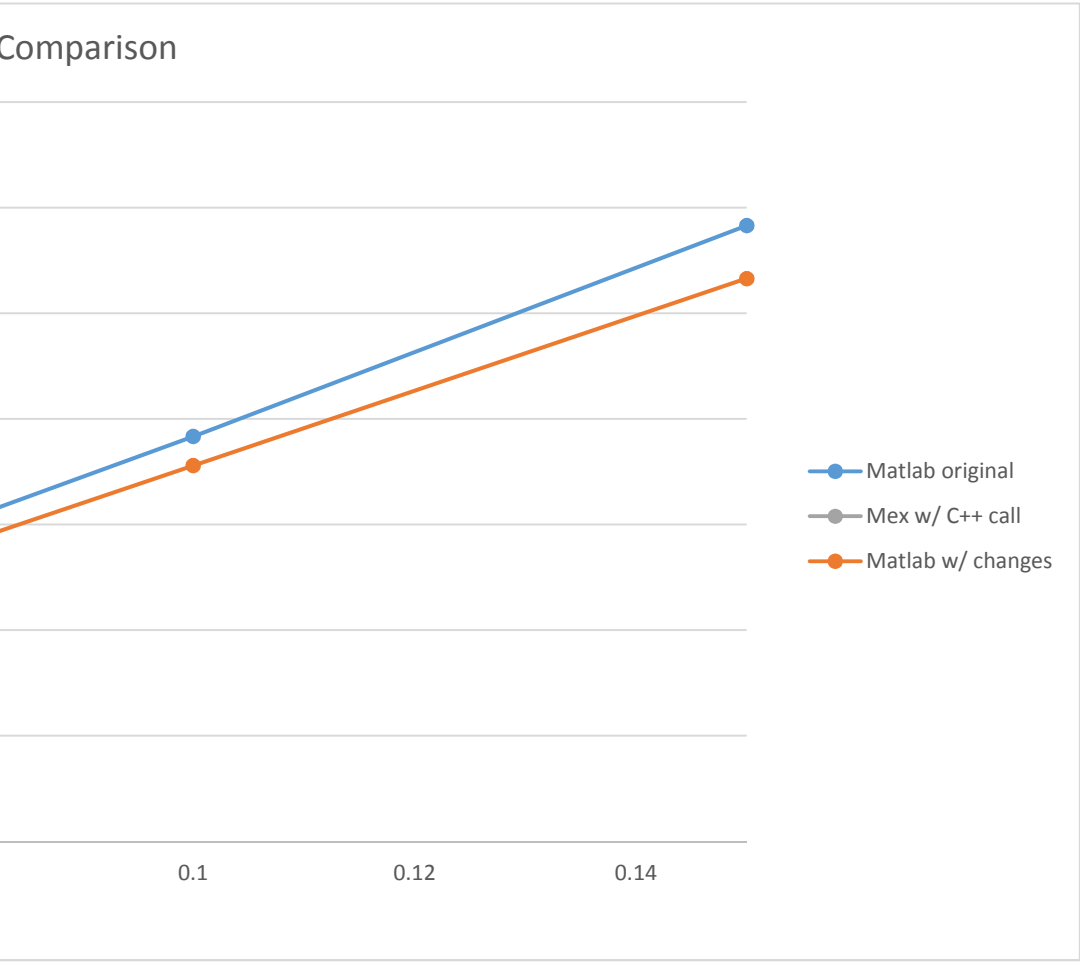
4.594166667

Initial time

N/A - Rob sent me reduced matrices

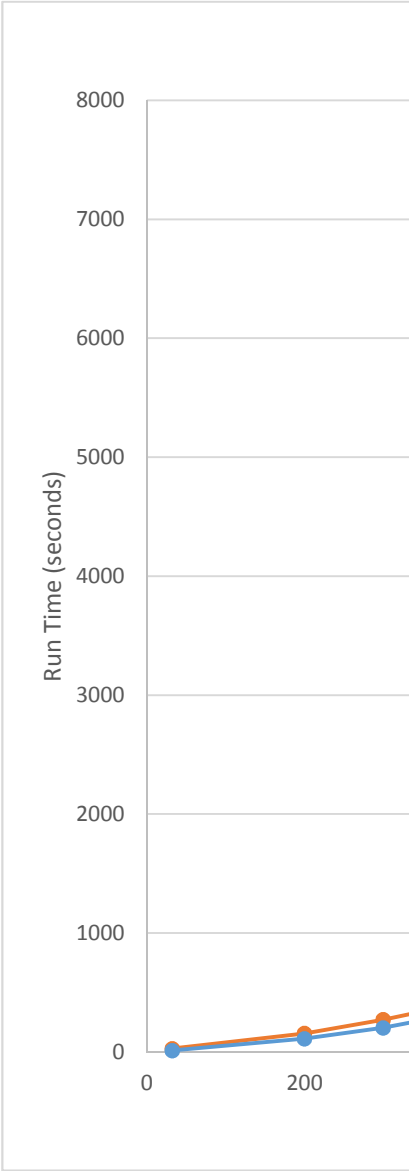


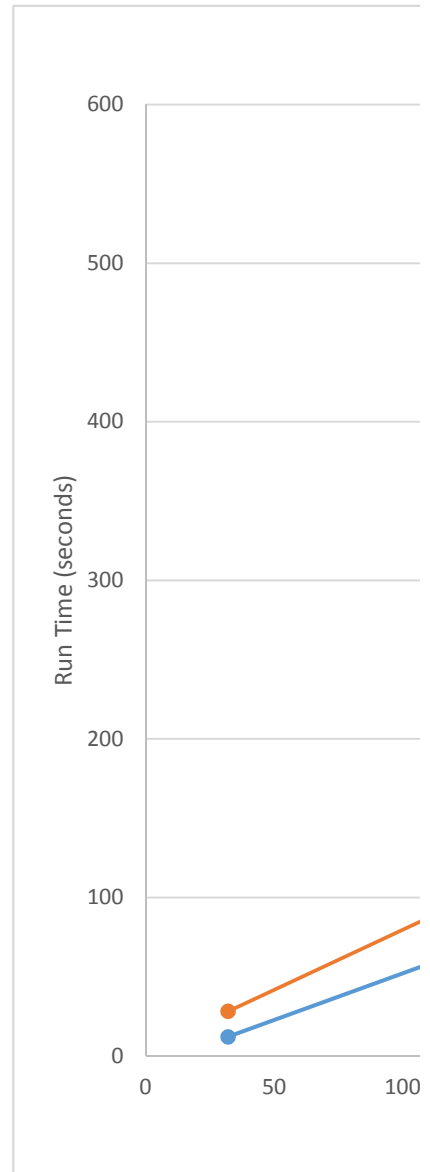
romviscoSBtsts.C (tskip=100)	romviscoSBts.C (tskip=100)	Mex w/ C++ call
run time (s)	run time (s)	run time (s)
		1867.9
6992	6541	



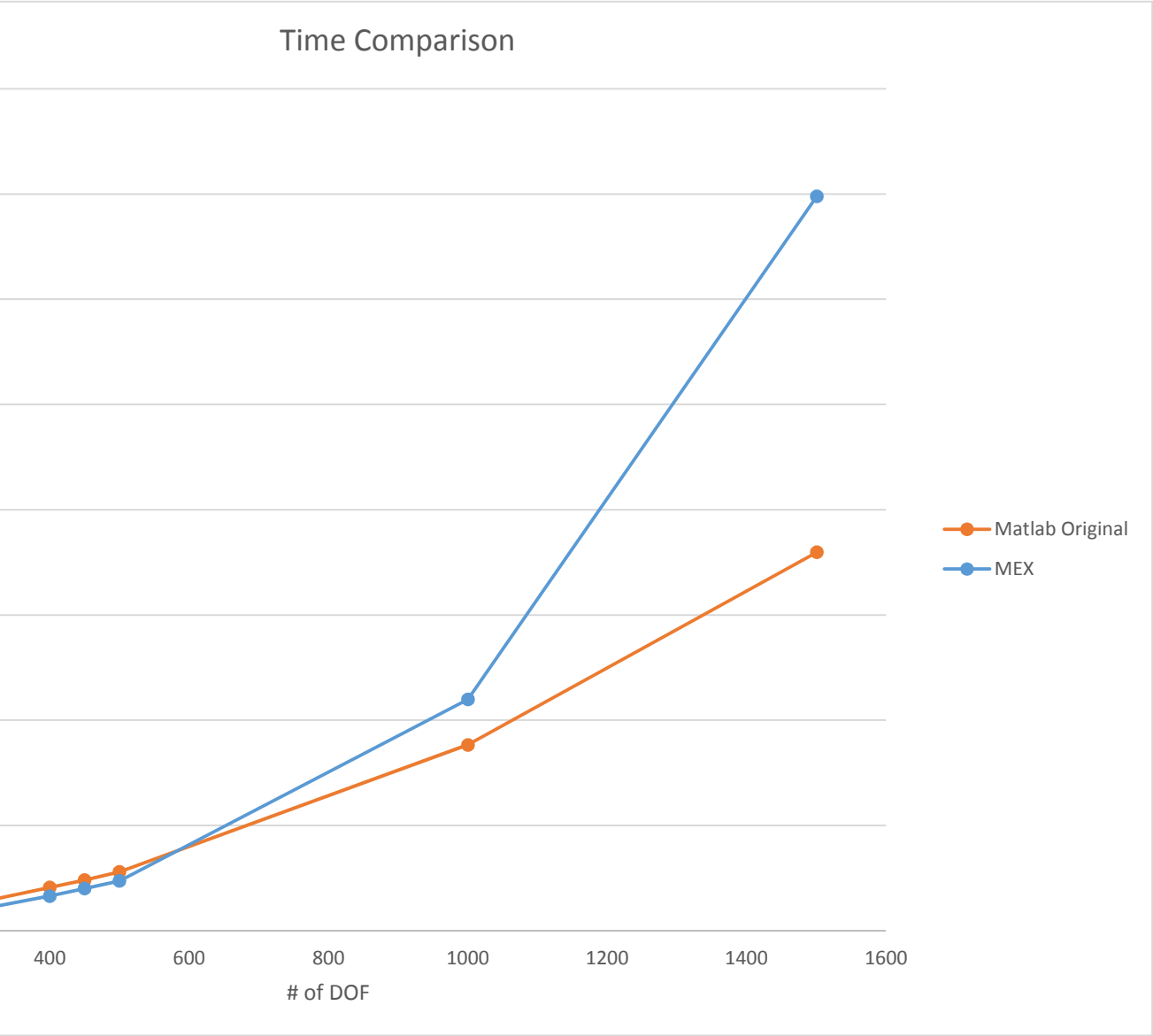
DOF	dt	Matlab original		Matlab w/ central_diff_visco_klp		romviscoSBtst
		cost_pre	run time (s)	cost_pre	run time (s)	cost_pre
32	5.00E-06		5.4			
32	1.00E-06	11.05	28.2	9.4	28.2	8.9
200	1.00E-06	105.31	155.2			101.0
200	3.00E-06	104.57	53.7	104.9	53.3	
200	5.00E-06					
300	1.00E-06	132.51	271.4			133.6
400	1.00E-06	128.76	411.1			128.0
450	1.00E-06	108.39	480.7			294.7
500	1.00E-06	125.66	559.7			122.1
500	3.00E-06	127.02	187.7			
1000	1.00E-06	217.43	1766.2	214.6	1820.5	205.6
1501	1.00E-06	394.65	3596.7			384.5

tspan = 0.1

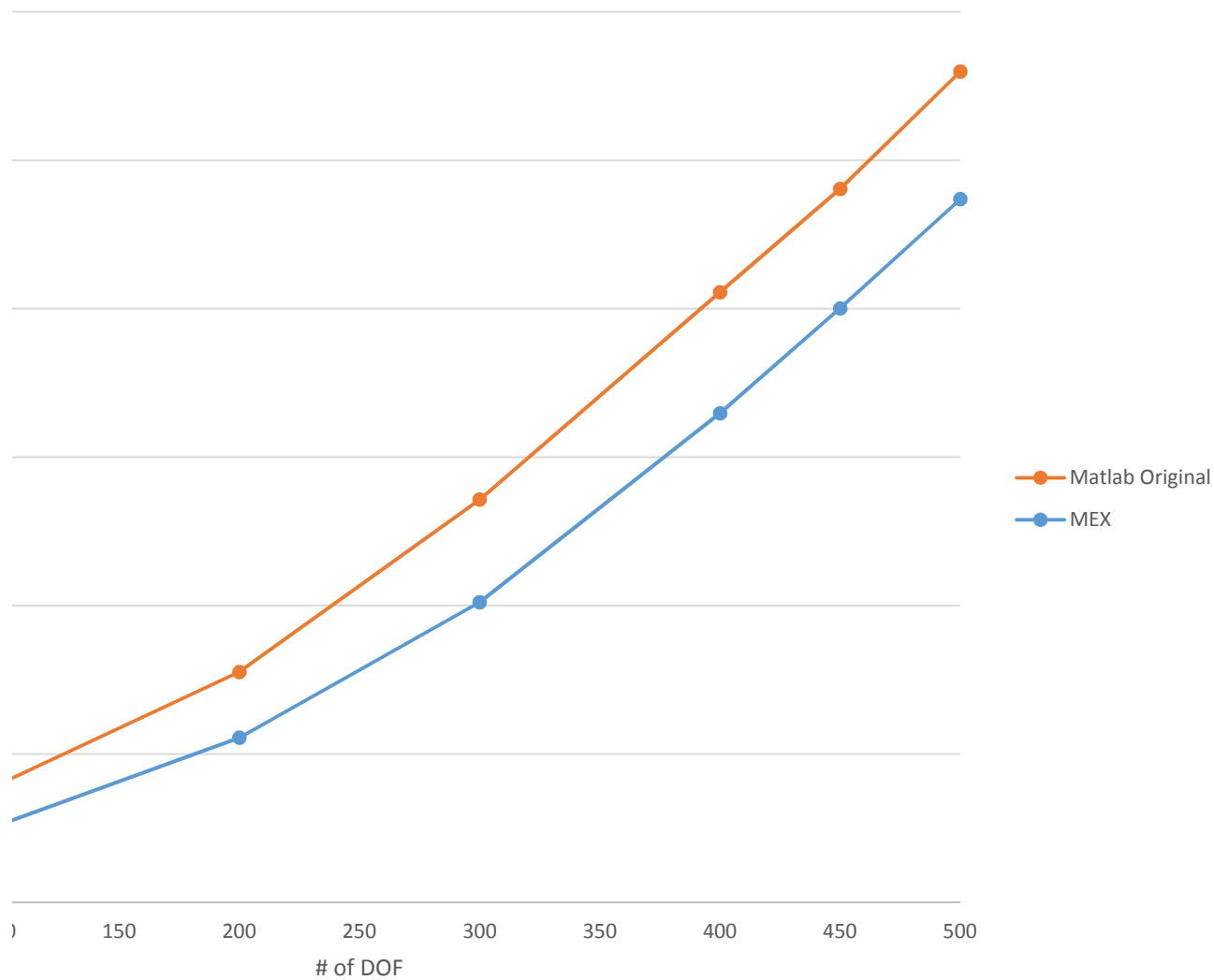




s.C (tskip=100)	romviscoSBts.C (tskip=100)		
run time (s)	cost_pre	run time (s)	
11.9	9.7999	12.1787	57%
110.8	102.0058	110.9478	29%
203.6	134.6604	202.2792	25%
318.6	130.1851	329.5607	20%
389.9	113.8297	400.2096	17%
458.0	122.4663	473.7717	15%
2182.2	215.1870	2197.2000	
7117.4	389.6928	6980.3000	



Time Comparison



romviscoC.C Instructions

Introduction

romviscoC.C is a MEX file written in C++ coding language. It is a computational function used to integrate a reduced order model using the explicit central difference method. Comments throughout the code describe where the calculations come from (relative to central_diff_visco.m).

Compile MEX file

To compile romviscoC.C, type the following command in the Matlab command window:

```
>> mex romviscoC.C  
Building with 'gcc'.  
MEX completed successfully.
```

The same message shown above should be written in the command window if this step is completed successfully. The MEX file cannot be run until it is successfully compiled.

Run MEX file

To run romviscoC.C, the following line (with user's variable names) is written in the .m script:

```
[time,q,qd]=romviscoC(invM_red,K_red,tspan,dt,q0,qd0,Kvg_red,Kvk_red,fshear,taushear,F_red,mag,omega,fbulk,taubulk,tskip);
```

with Inputs (Ndof=number of degrees of freedom, NProny=number of Prony terms):

- invM_red = inverse of the reduced mass matrix (Ndof x Ndof)
- K_red = reduced K matrix (Ndof x Ndof)
- tspan = (1 x 2) row vector with beginning time and end time (example: [0.0,0.1])
- dt = time increment for each time step (scalar)
- q0 = initial displacements (Ndof x 1)
- qd0 = initial velocities (Ndof x 1)
- Kvg_red = reduced shear K matrix (Ndof x Ndof)
- Kvk_red = reduced bulk K matrix (Ndof x Ndof)
- fshear = Prony pre-factors for shear (NProny x 1)
- taushear = time constants for shear (NProny x 1)
- F_red = reduced force shape (Ndof x 1)
- mag = external force magnitude (scalar)
- omega = frequency of haversine function (scalar)
- fbulk = Prony pre-factors for bulk (NProny x 1)
- taubulk = time constants for bulk (NProny x 1)

- `tskip` = tells MEX file how often to save output variables (will save every “`tskip`”th value) – when `tskip=1`, values every time step are saved

and Outputs (`nts`=number of time steps):

- `time` = row vector of time values ($1 \times nts/tskip$)
- `q` = matrix of displacements for every degree of freedom at every saved time step ($nts/tskip \times Ndof$)
- `qd` = matrix of velocities for every degree of freedom at every saved time step ($nts/tskip \times Ndof$)

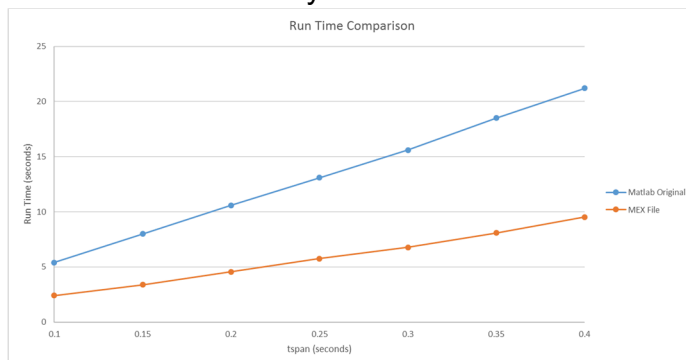
Options

`romviscoC.C` is written with several options imbedded within the code. The user must comment/uncomment accordingly (instructions in code) when implementing a certain option.

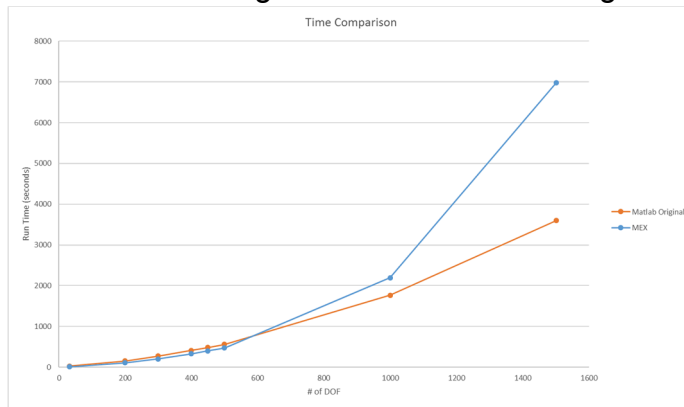
- Force subfunction options: the user has the option to call `f_ext.C` every time step, `f_once.C` once inside MEX file before starting time stepping, `f_extOnce.m` once before entering the MEX file, or `f_ext.m` every time step (from MEX file back to Matlab).
 - Default is calling `f_ext.C` every time step.
- Output format options: the user has the option either write results to pointers directly sent back to Matlab or to write outputs to `.txt` files for later post-processing.
 - Default is writing to pointers that are directly sent back to Matlab.
- Output accelerations option: `qdd` can also be output from the MEX file.
 - Default is that accelerations are NOT output.

Time Saving Results and MEX File Limitations

- Run time is linear as you increase the total time span (as expected)



- MEX file is no longer efficient after 500 degrees of freedom



- Time savings below 500 degrees of freedom:

