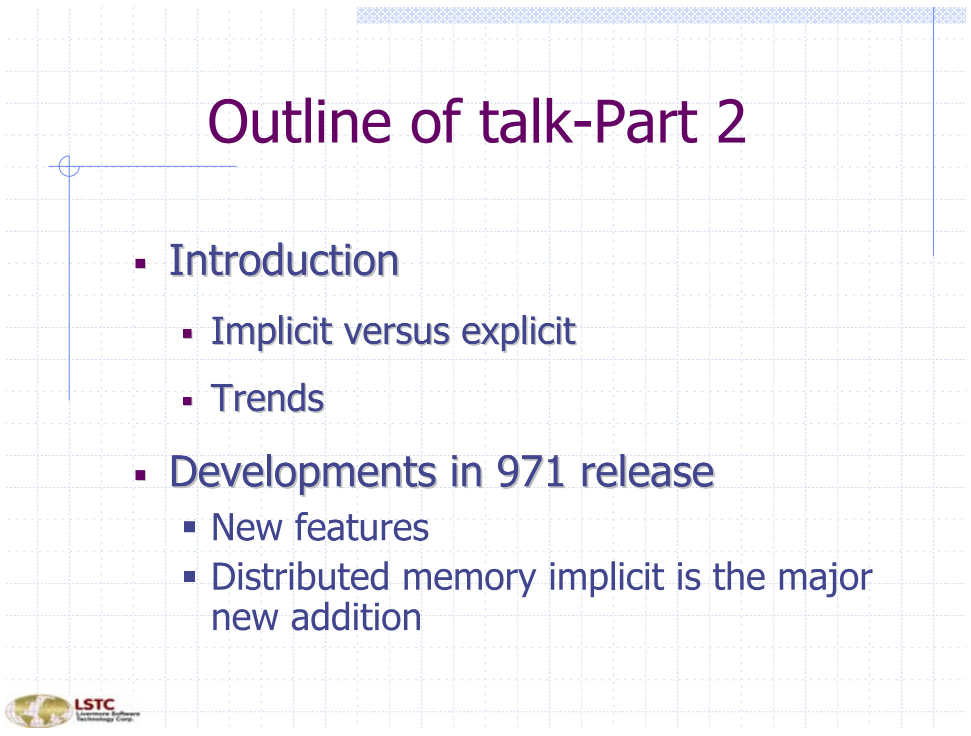
Recent developments in LS-DYNA

Part II

3rd LS-DYNA Forum 2004

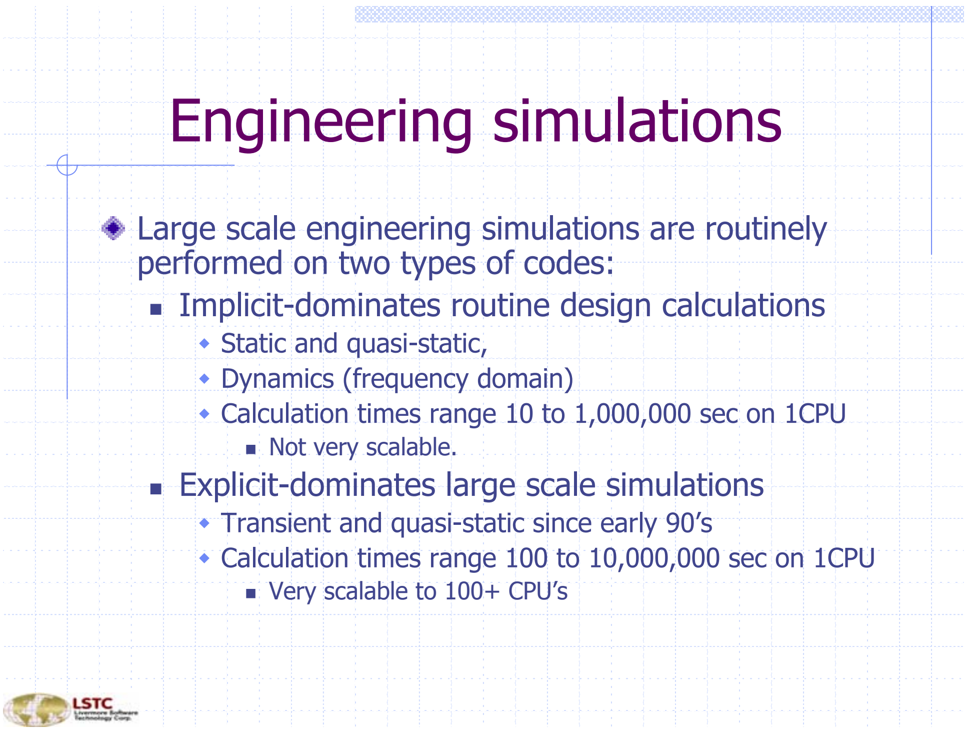
October 15

J. O. Hallquist
Livermore Software Technology Corporation



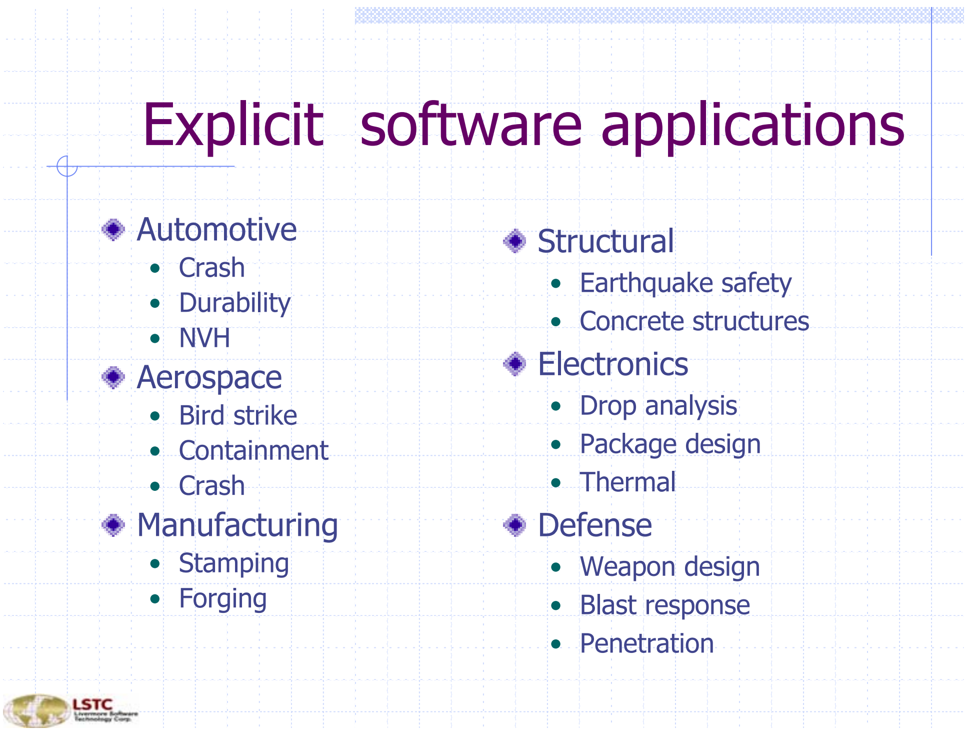
Outline of talk-Part 2

- Introduction
  - Implicit versus explicit
  - Trends
- Developments in 971 release
  - New features
  - Distributed memory implicit is the major new addition



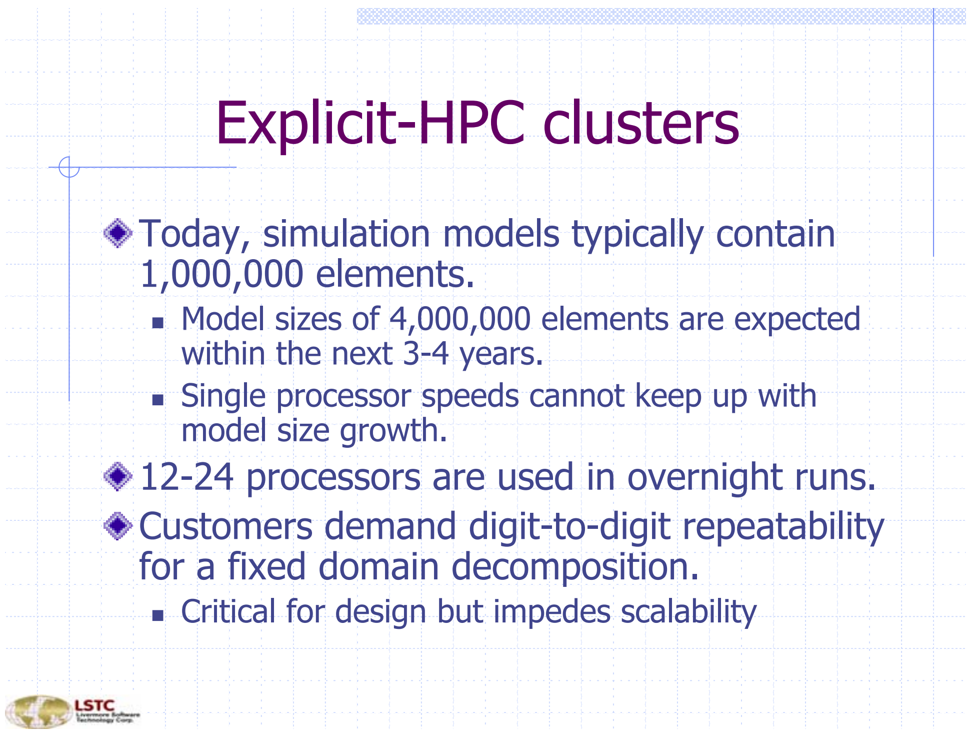
Engineering simulations

- Large scale engineering simulations are routinely performed on two types of codes:
  - Implicit-dominates routine design calculations
    - Static and quasi-static,
    - Dynamics (frequency domain)
    - Calculation times range 10 to 1,000,000 sec on 1CPU
      - Not very scalable.
  - Explicit-dominates large scale simulations
    - Transient and quasi-static since early 90's
    - Calculation times range 100 to 10,000,000 sec on 1CPU
      - Very scalable to 100+ CPU's



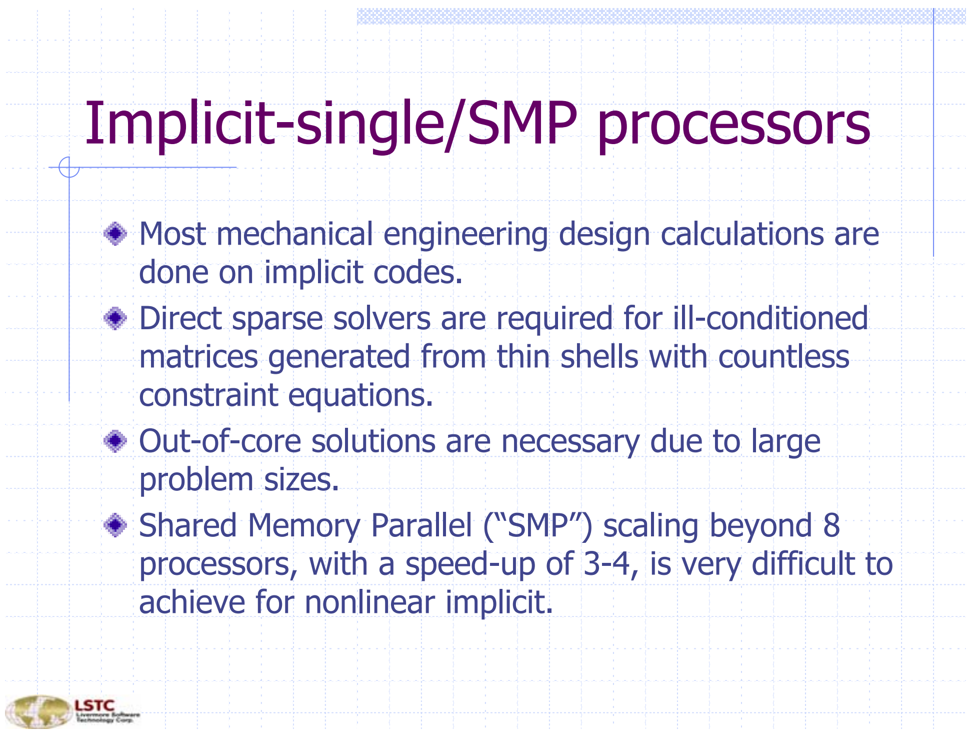
Explicit software applications

- Automotive
  - Crash
  - Durability
  - NVH
- Aerospace
  - Bird strike
  - Containment
  - Crash
- Manufacturing
  - Stamping
  - Forging
- Structural
  - Earthquake safety
  - Concrete structures
- Electronics
  - Drop analysis
  - Package design
  - Thermal
- Defense
  - Weapon design
  - Blast response
  - Penetration



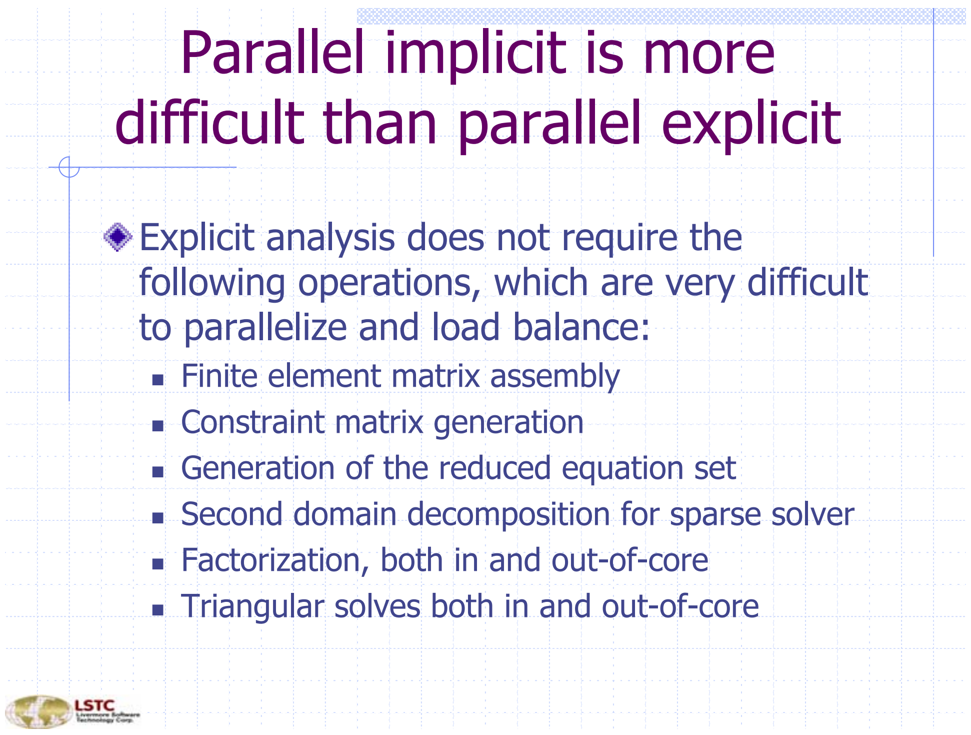
Explicit-HPC clusters

- Today, simulation models typically contain 1,000,000 elements.
  - Model sizes of 4,000,000 elements are expected within the next 3-4 years.
  - Single processor speeds cannot keep up with model size growth.
- 12-24 processors are used in overnight runs.
- Customers demand digit-to-digit repeatability for a fixed domain decomposition.
  - Critical for design but impedes scalability



Implicit-single/SMP processors

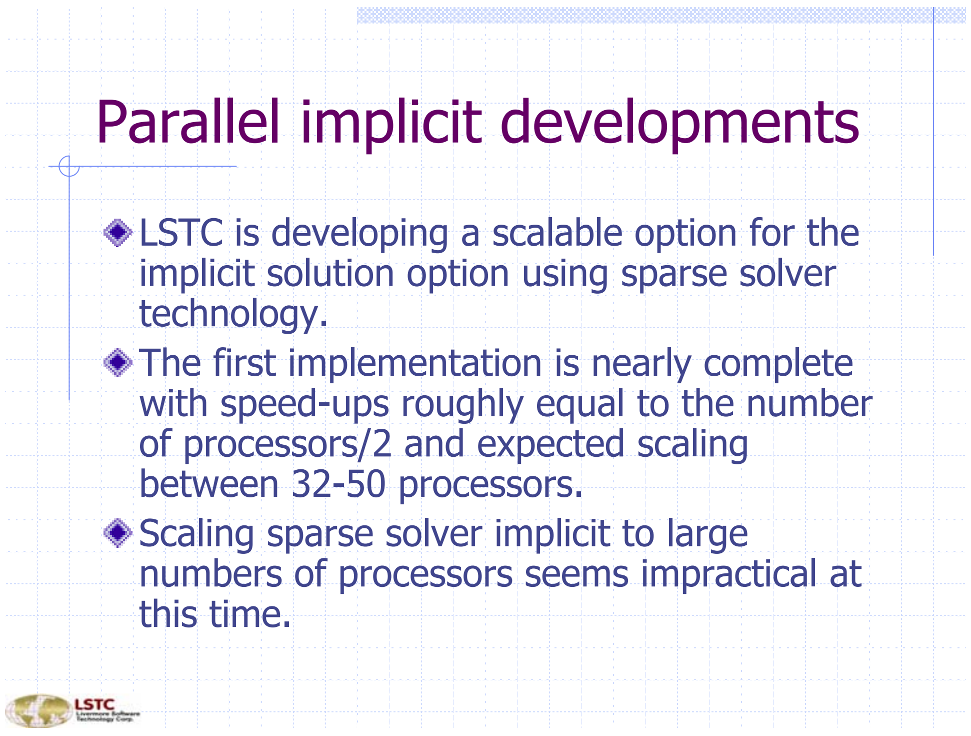
- Most mechanical engineering design calculations are done on implicit codes.
- Direct sparse solvers are required for ill-conditioned matrices generated from thin shells with countless constraint equations.
- Out-of-core solutions are necessary due to large problem sizes.
- Shared Memory Parallel ("SMP") scaling beyond 8 processors, with a speed-up of 3-4, is very difficult to achieve for nonlinear implicit.



Parallel implicit is more difficult than parallel explicit

- Explicit analysis does not require the following operations, which are very difficult to parallelize and load balance:
  - Finite element matrix assembly
  - Constraint matrix generation
  - Generation of the reduced equation set
  - Second domain decomposition for sparse solver
  - Factorization, both in and out-of-core
  - Triangular solves both in and out-of-core

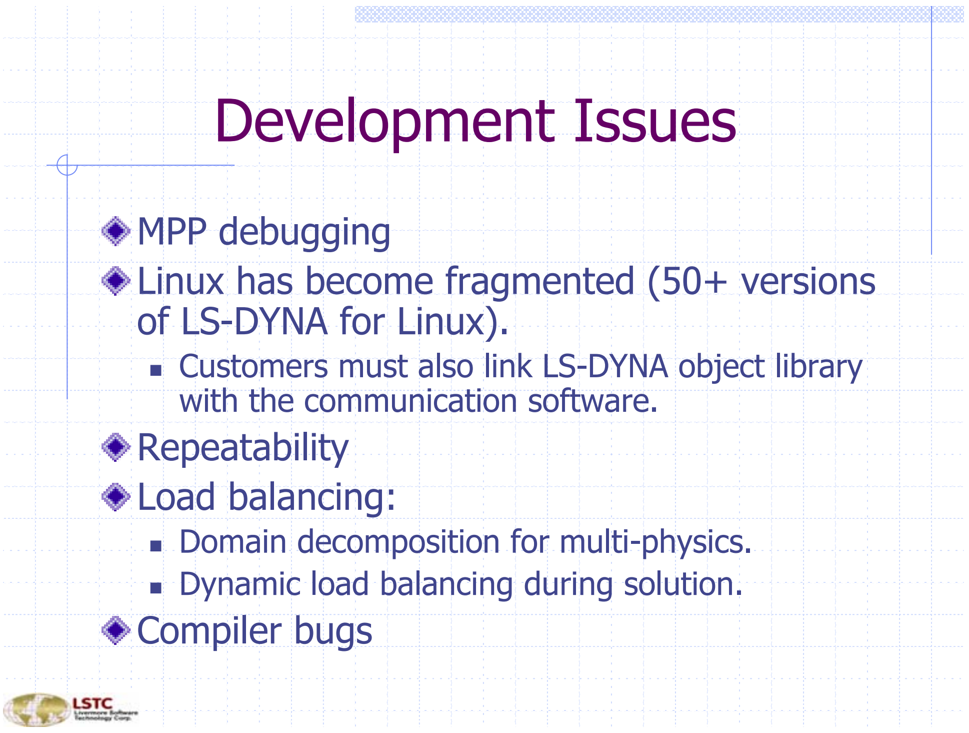
LSTC Livermore Software Technology Corp.



Parallel implicit developments

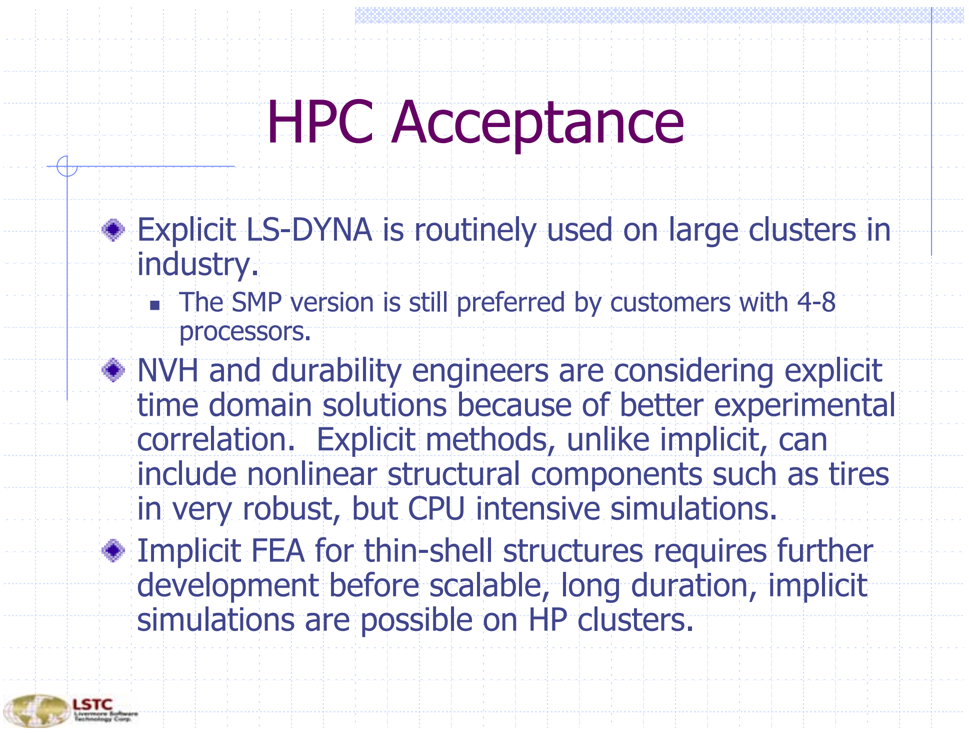
- LSTC is developing a scalable option for the implicit solution option using sparse solver technology.
- The first implementation is nearly complete with speed-ups roughly equal to the number of processors/2 and expected scaling between 32-50 processors.
- Scaling sparse solver implicit to large numbers of processors seems impractical at this time.

LSTC Livermore Software Technology Corp.



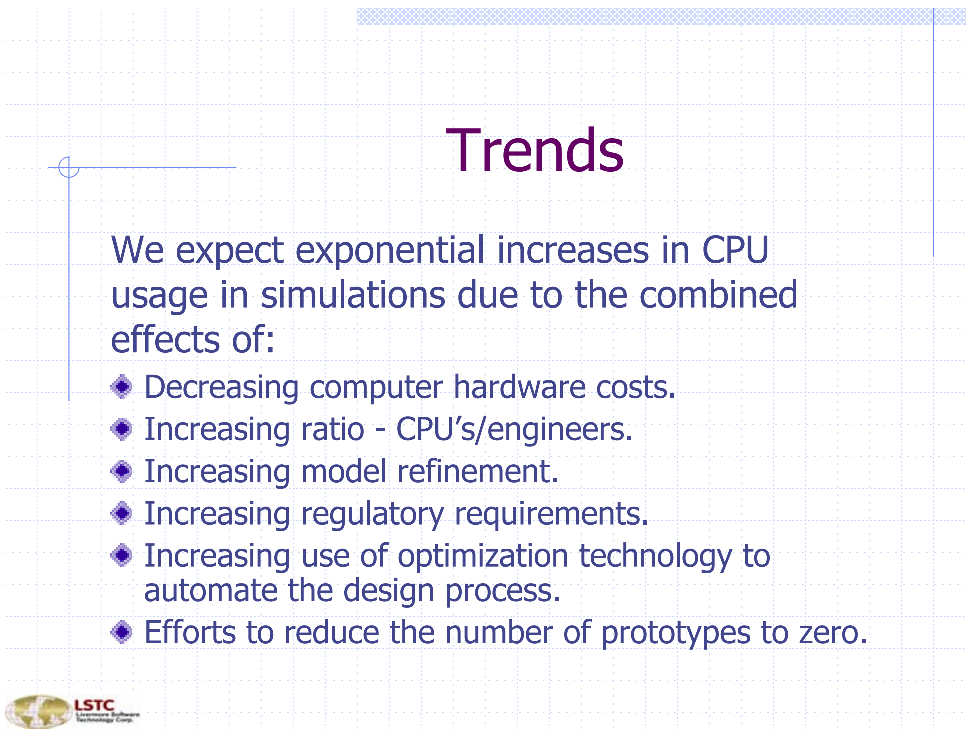
Development Issues

- MPP debugging
- Linux has become fragmented (50+ versions of LS-DYNA for Linux).
  - Customers must also link LS-DYNA object library with the communication software.
- Repeatability
- Load balancing:
  - Domain decomposition for multi-physics.
  - Dynamic load balancing during solution.
- Compiler bugs



HPC Acceptance

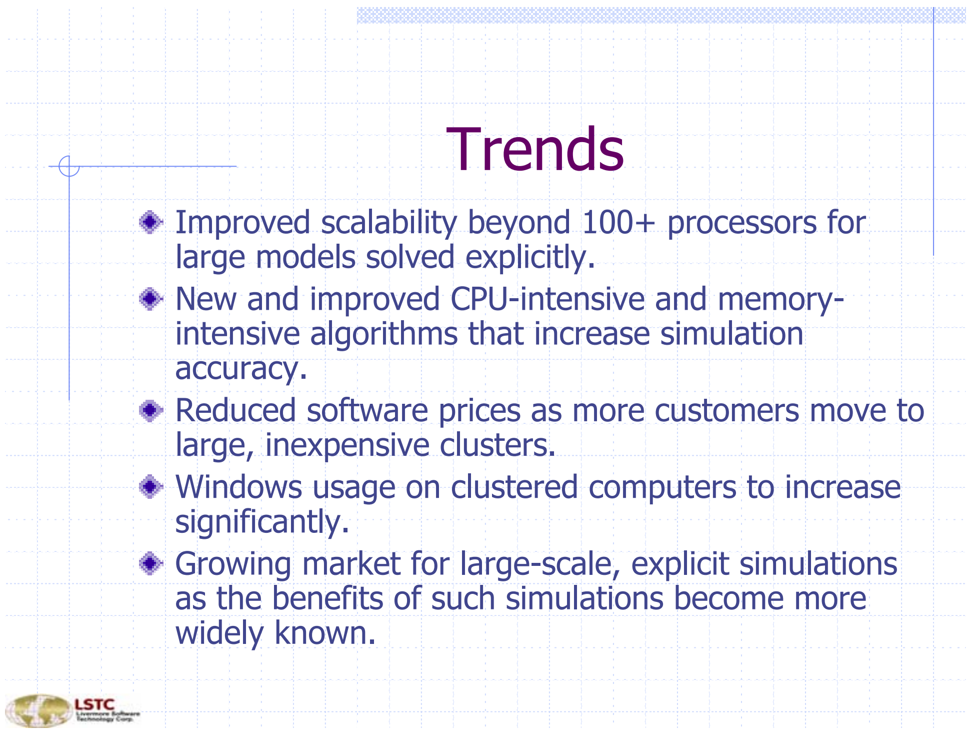
- Explicit LS-DYNA is routinely used on large clusters in industry.
  - The SMP version is still preferred by customers with 4-8 processors.
- NVH and durability engineers are considering explicit time domain solutions because of better experimental correlation. Explicit methods, unlike implicit, can include nonlinear structural components such as tires in very robust, but CPU intensive simulations.
- Implicit FEA for thin-shell structures requires further development before scalable, long duration, implicit simulations are possible on HP clusters.



Trends

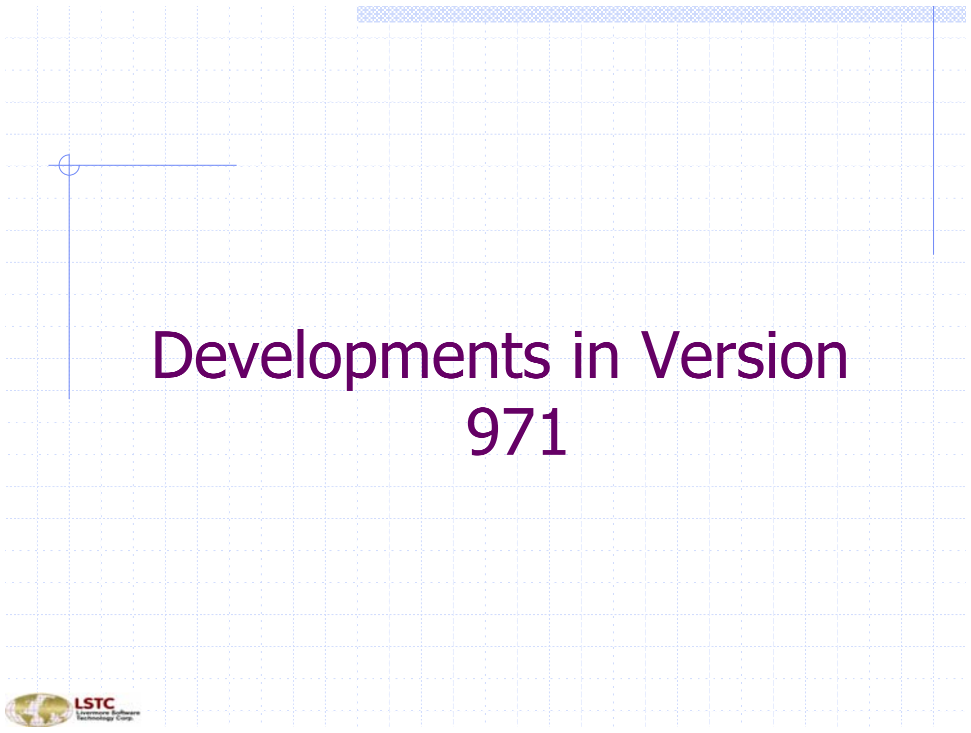
We expect exponential increases in CPU usage in simulations due to the combined effects of:

- Decreasing computer hardware costs.
- Increasing ratio - CPU's/engineers.
- Increasing model refinement.
- Increasing regulatory requirements.
- Increasing use of optimization technology to automate the design process.
- Efforts to reduce the number of prototypes to zero.



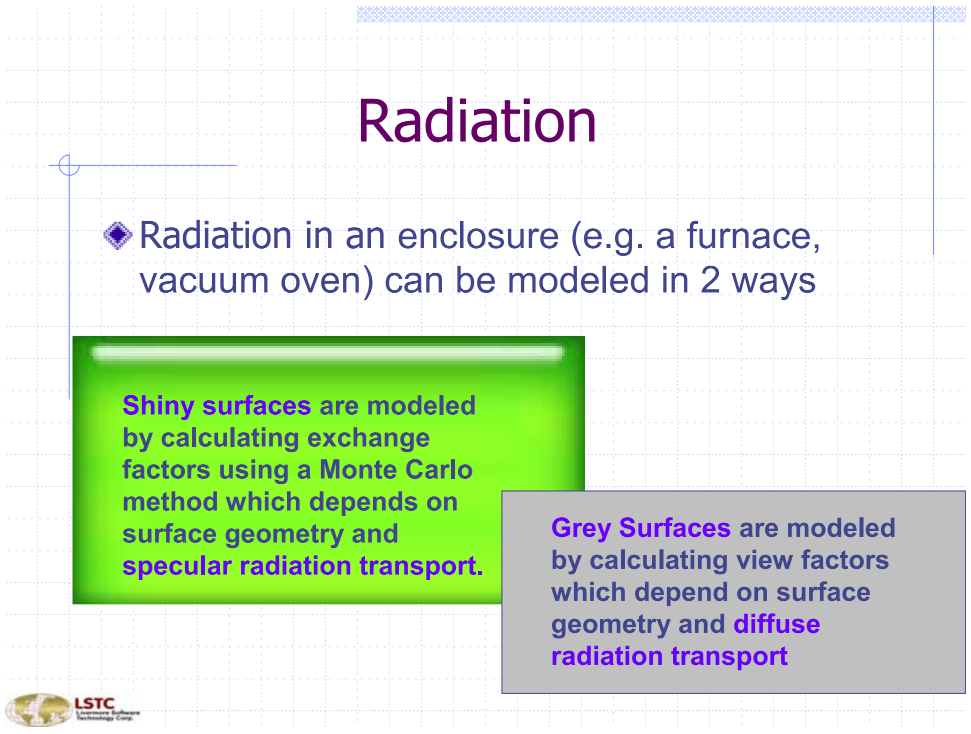
Trends

- Improved scalability beyond 100+ processors for large models solved explicitly.
- New and improved CPU-intensive and memory-intensive algorithms that increase simulation accuracy.
- Reduced software prices as more customers move to large, inexpensive clusters.
- Windows usage on clustered computers to increase significantly.
- Growing market for large-scale, explicit simulations as the benefits of such simulations become more widely known.



Developments in Version 971

LSTC Livermore Software Technology Corp.



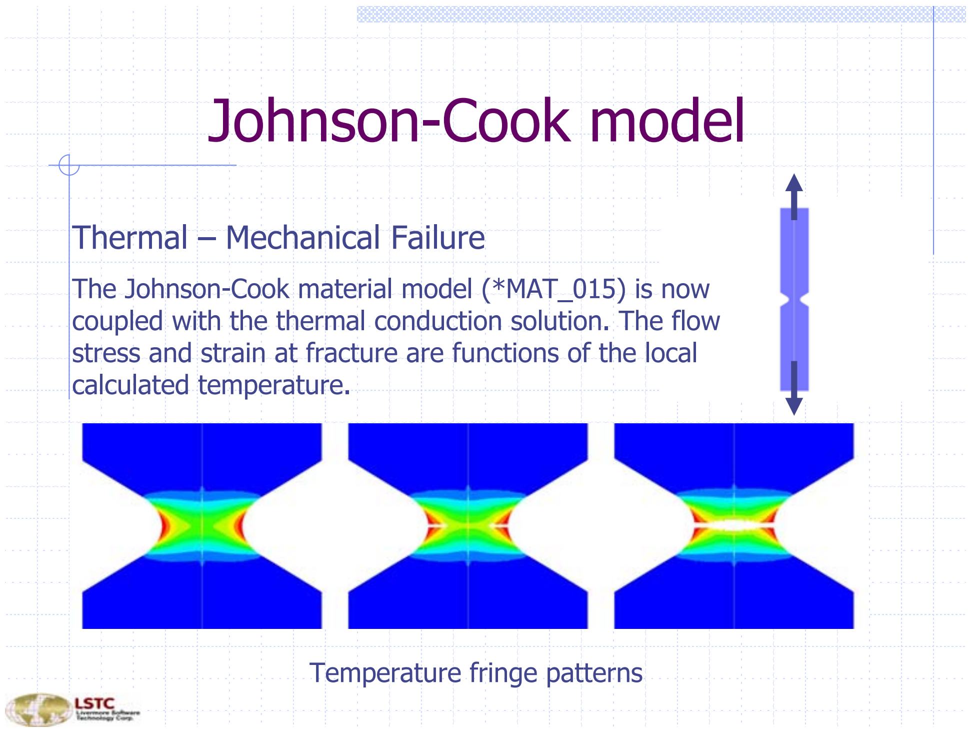
Radiation

- Radiation in an enclosure (e.g. a furnace, vacuum oven) can be modeled in 2 ways

Shiny surfaces are modeled by calculating exchange factors using a Monte Carlo method which depends on surface geometry and specular radiation transport.

Grey Surfaces are modeled by calculating view factors which depend on surface geometry and diffuse radiation transport

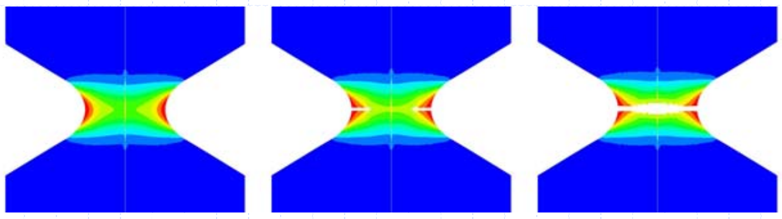
LSTC Livermore Software Technology Corp.



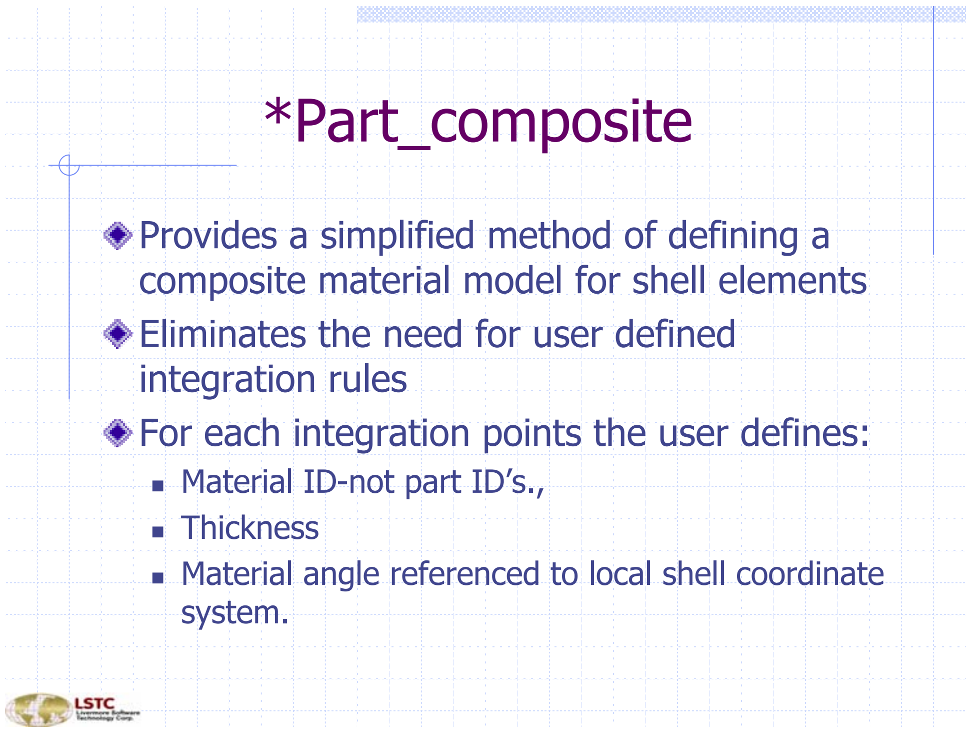
Johnson-Cook model

Thermal – Mechanical Failure

The Johnson-Cook material model (*MAT_015) is now coupled with the thermal conduction solution. The flow stress and strain at fracture are functions of the local calculated temperature.

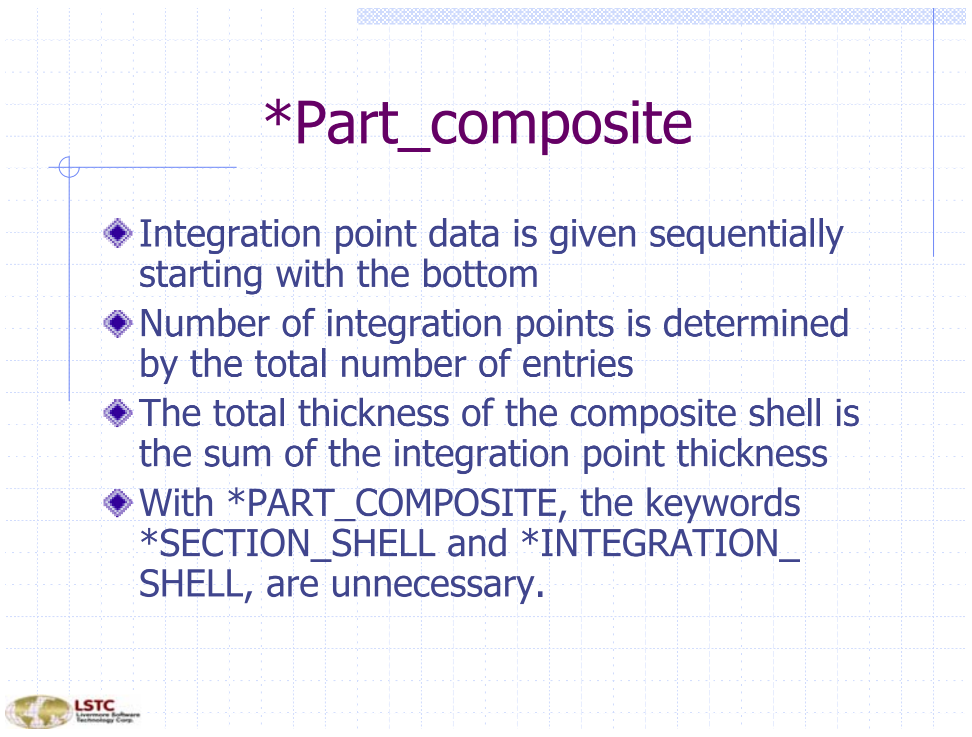


Temperature fringe patterns



*Part_composite

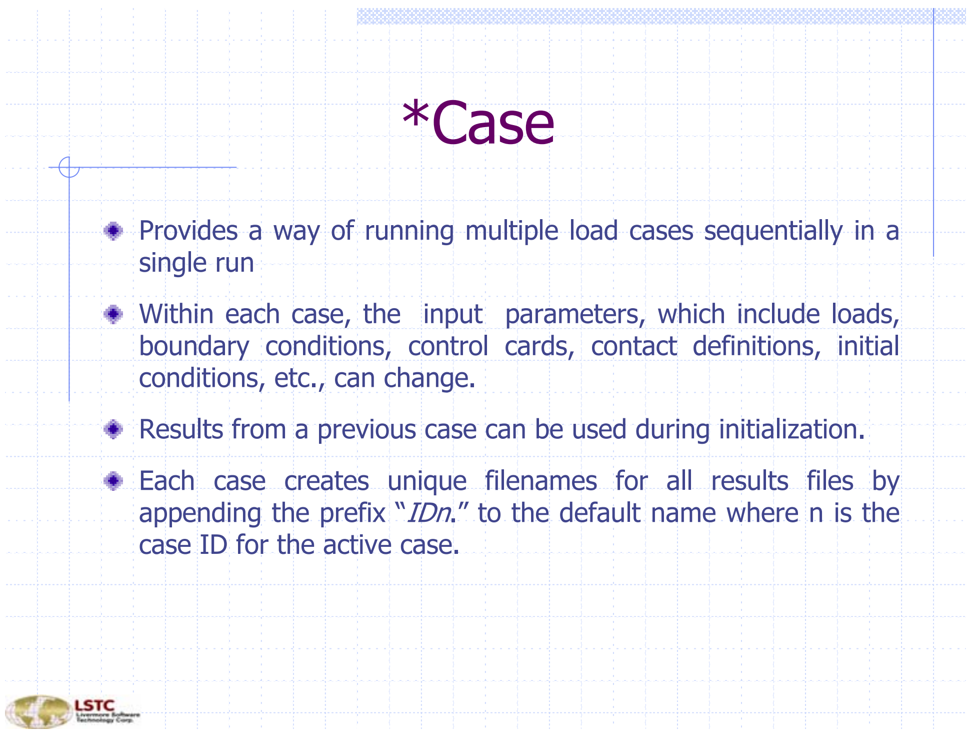
- Provides a simplified method of defining a composite material model for shell elements
- Eliminates the need for user defined integration rules
- For each integration points the user defines:
  - Material ID-not part ID's.,
  - Thickness
  - Material angle referenced to local shell coordinate system.



*Part_composite

- Integration point data is given sequentially starting with the bottom
- Number of integration points is determined by the total number of entries
- The total thickness of the composite shell is the sum of the integration point thickness
- With *PART_COMPOSITE, the keywords *SECTION_SHELL and *INTEGRATION_ SHELL, are unnecessary.

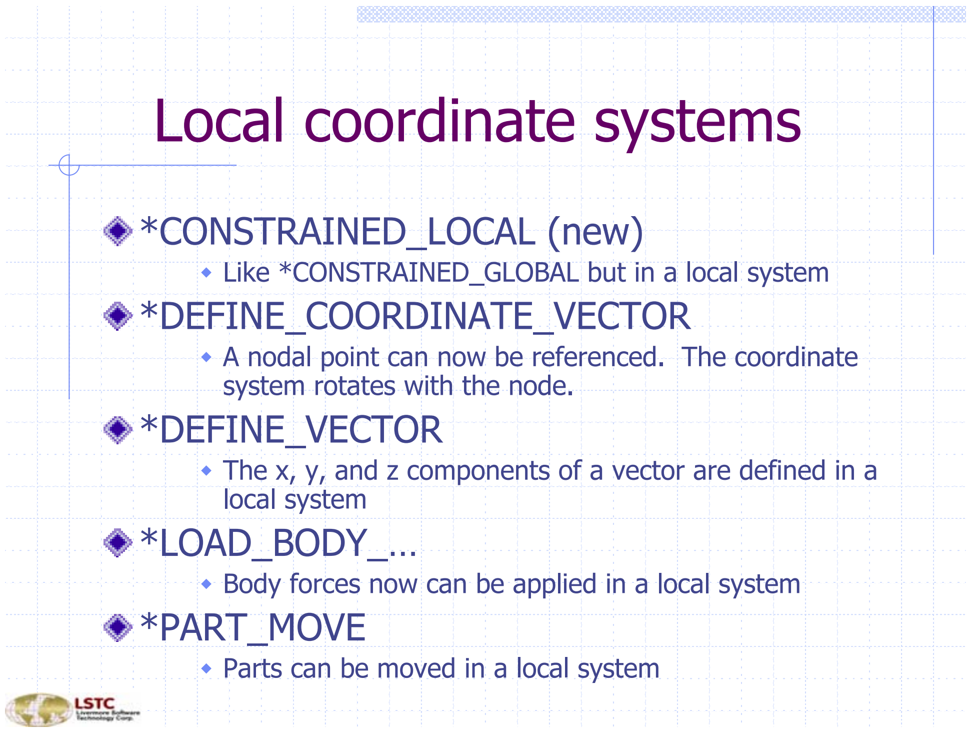
LSTC Livermore Software Technology Corp.



*Case

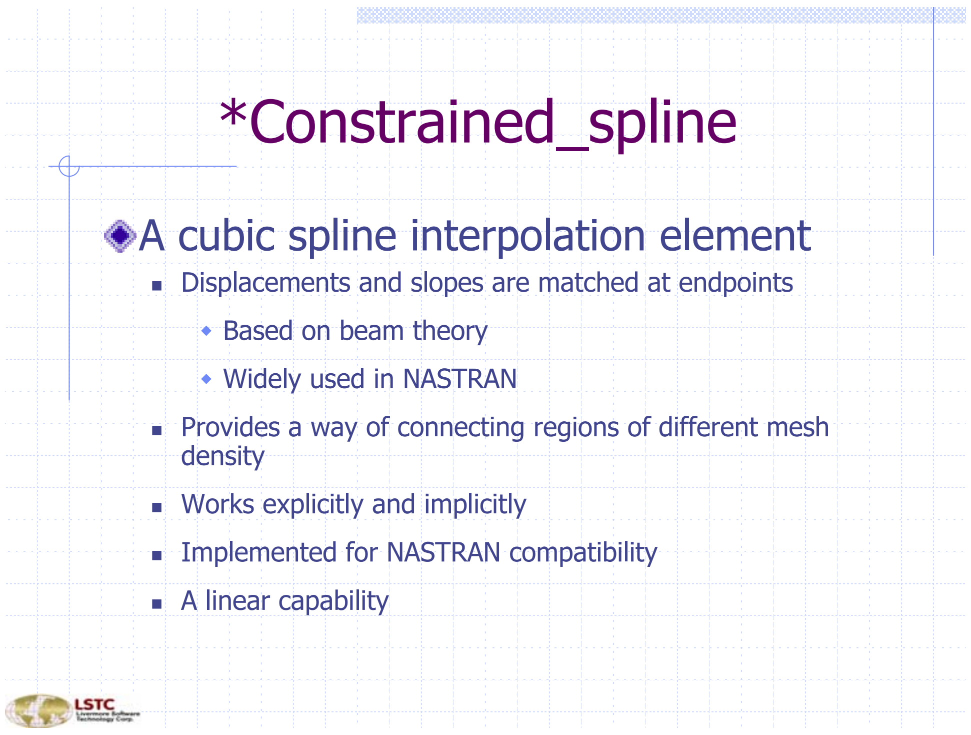
- Provides a way of running multiple load cases sequentially in a single run
- Within each case, the input parameters, which include loads, boundary conditions, control cards, contact definitions, initial conditions, etc., can change.
- Results from a previous case can be used during initialization.
- Each case creates unique filenames for all results files by appending the prefix "*IDn.*" to the default name where n is the case ID for the active case.

LSTC Livermore Software Technology Corp.



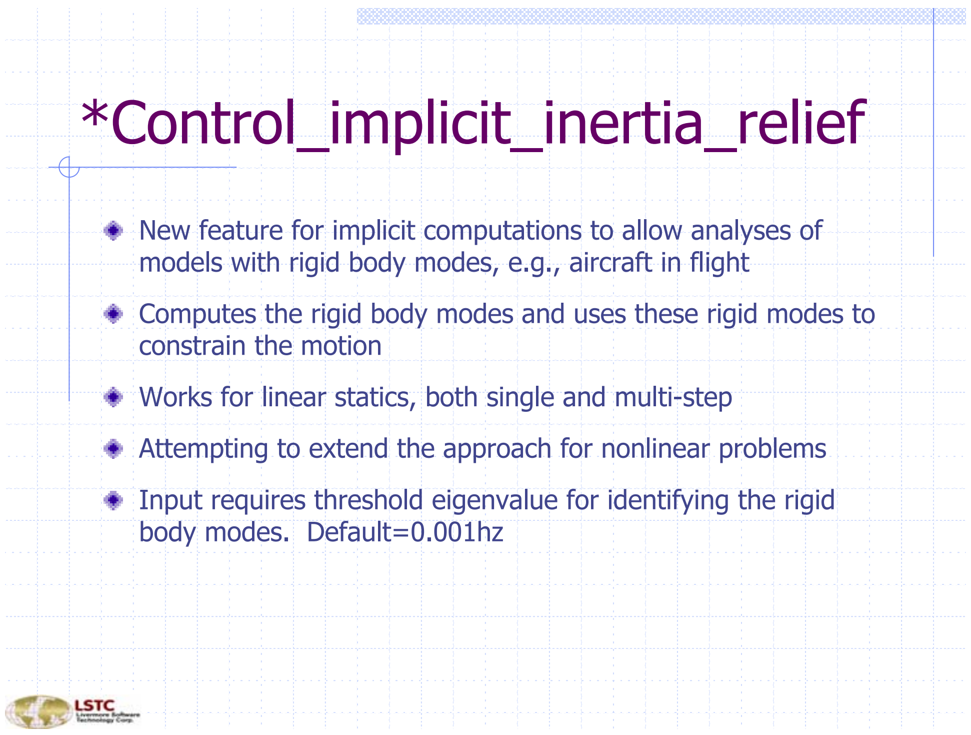
Local coordinate systems

- *CONSTRAINED_LOCAL (new)
  - Like *CONSTRAINED_GLOBAL but in a local system
- *DEFINE_COORDINATE_VECTOR
  - A nodal point can now be referenced. The coordinate system rotates with the node.
- *DEFINE_VECTOR
  - The x, y, and z components of a vector are defined in a local system
- *LOAD_BODY_...
  - Body forces now can be applied in a local system
- *PART_MOVE
  - Parts can be moved in a local system



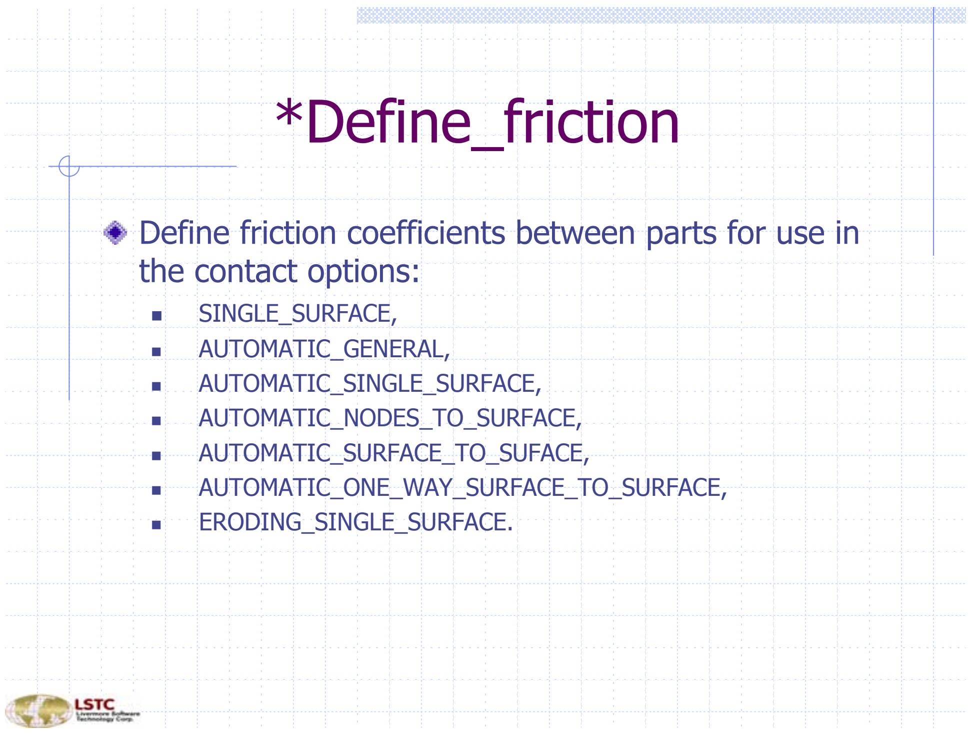
*Constrained_spline

- A cubic spline interpolation element
  - Displacements and slopes are matched at endpoints
    - Based on beam theory
    - Widely used in NASTRAN
  - Provides a way of connecting regions of different mesh density
  - Works explicitly and implicitly
  - Implemented for NASTRAN compatibility
  - A linear capability



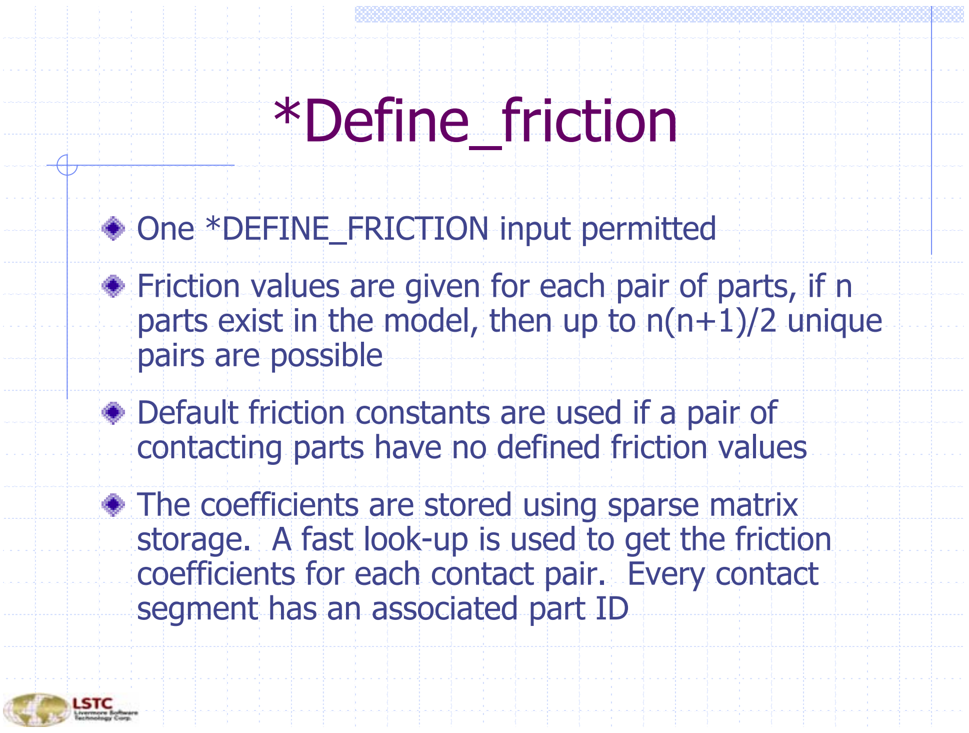
*Control_implicit_inertia_relief

- New feature for implicit computations to allow analyses of models with rigid body modes, e.g., aircraft in flight
- Computes the rigid body modes and uses these rigid modes to constrain the motion
- Works for linear statics, both single and multi-step
- Attempting to extend the approach for nonlinear problems
- Input requires threshold eigenvalue for identifying the rigid body modes. Default=0.001hz



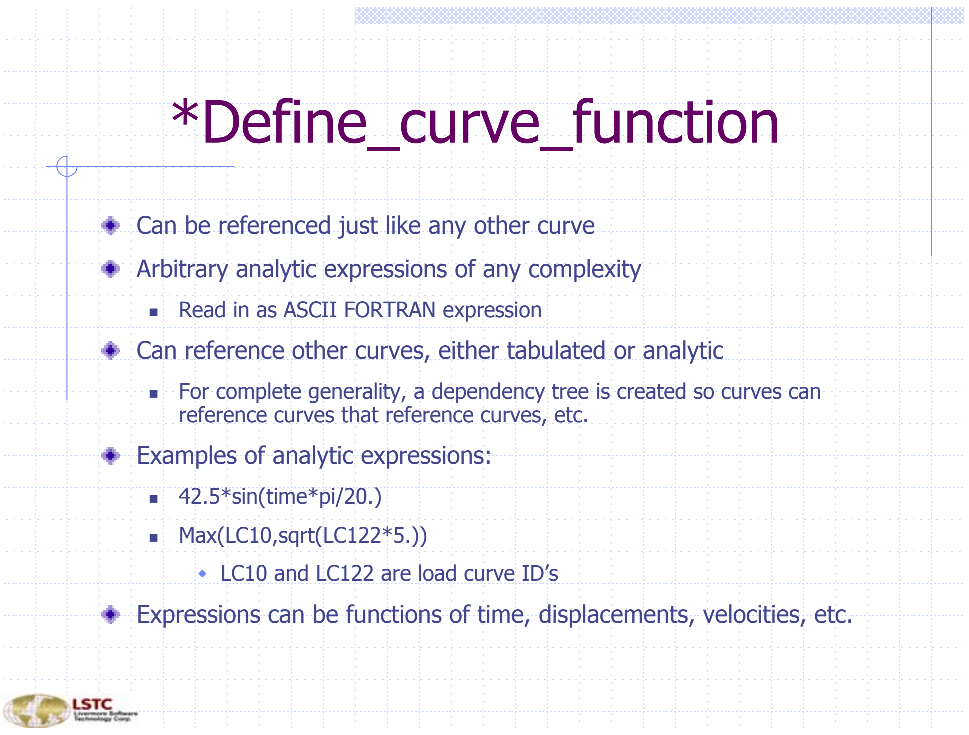
*Define_friction

- Define friction coefficients between parts for use in the contact options:
  - SINGLE_SURFACE,
  - AUTOMATIC_GENERAL,
  - AUTOMATIC_SINGLE_SURFACE,
  - AUTOMATIC_NODES_TO_SURFACE,
  - AUTOMATIC_SURFACE_TO_SUFACE,
  - AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE,
  - ERODING_SINGLE_SURFACE.



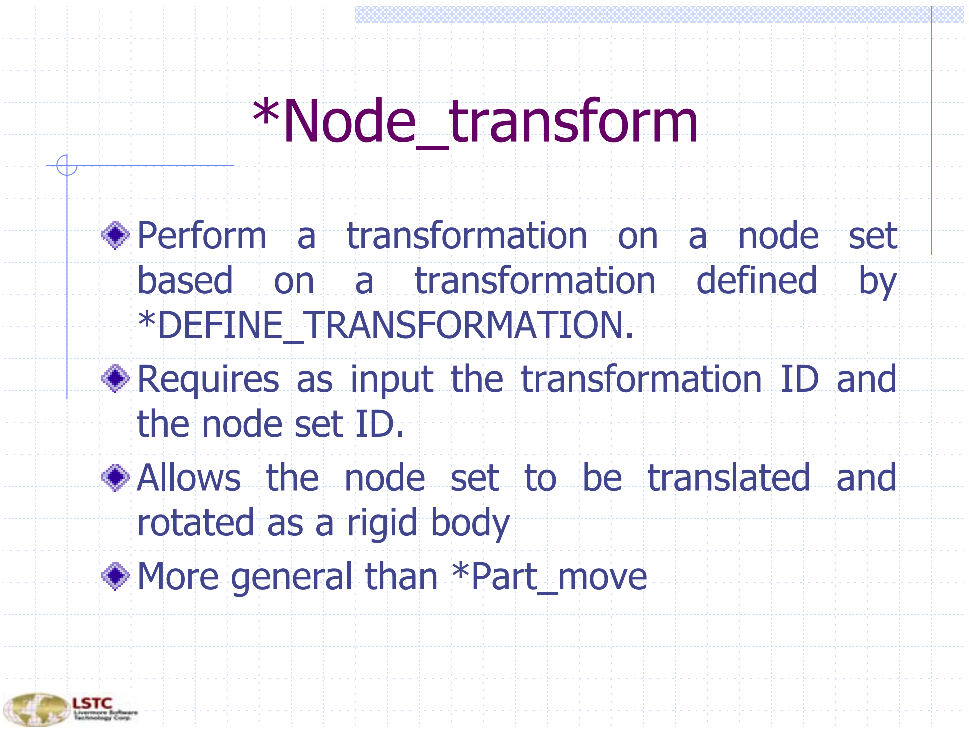
*Define_friction

- One *DEFINE_FRICTION input permitted
- Friction values are given for each pair of parts, if n parts exist in the model, then up to n(n+1)/2 unique pairs are possible
- Default friction constants are used if a pair of contacting parts have no defined friction values
- The coefficients are stored using sparse matrix storage. A fast look-up is used to get the friction coefficients for each contact pair. Every contact segment has an associated part ID



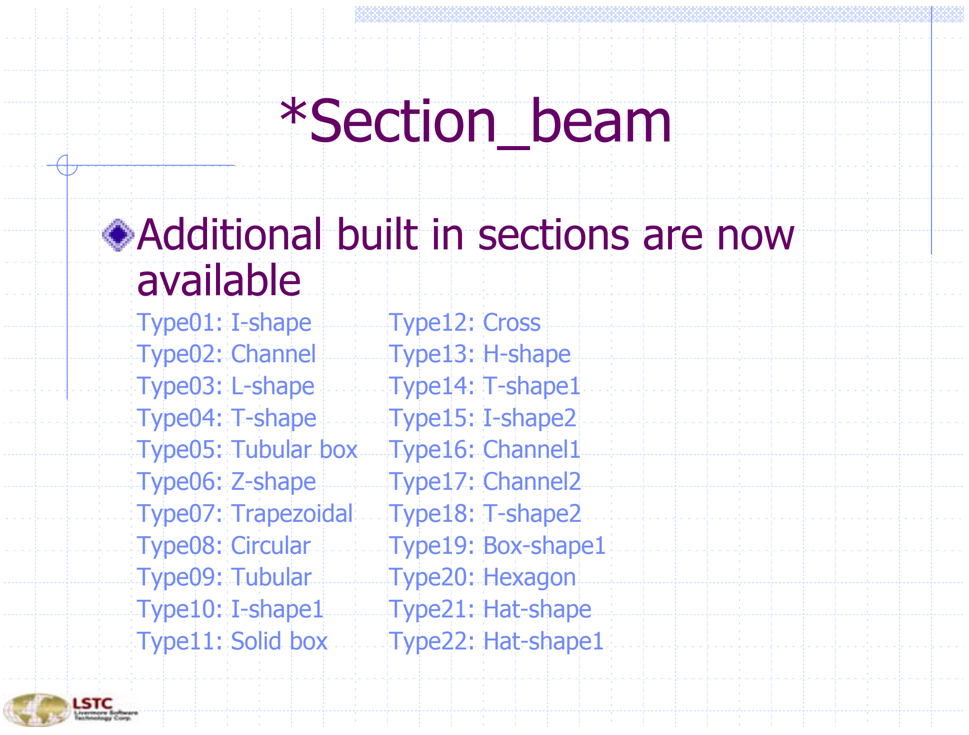
*Define_curve_function

- Can be referenced just like any other curve
- Arbitrary analytic expressions of any complexity
  - Read in as ASCII FORTRAN expression
- Can reference other curves, either tabulated or analytic
  - For complete generality, a dependency tree is created so curves can reference curves that reference curves, etc.
- Examples of analytic expressions:
  - 42.5*sin(time*pi/20.)
  - Max(LC10,sqrt(LC122*5.))
    - LC10 and LC122 are load curve ID's
- Expressions can be functions of time, displacements, velocities, etc.



*Node_transform

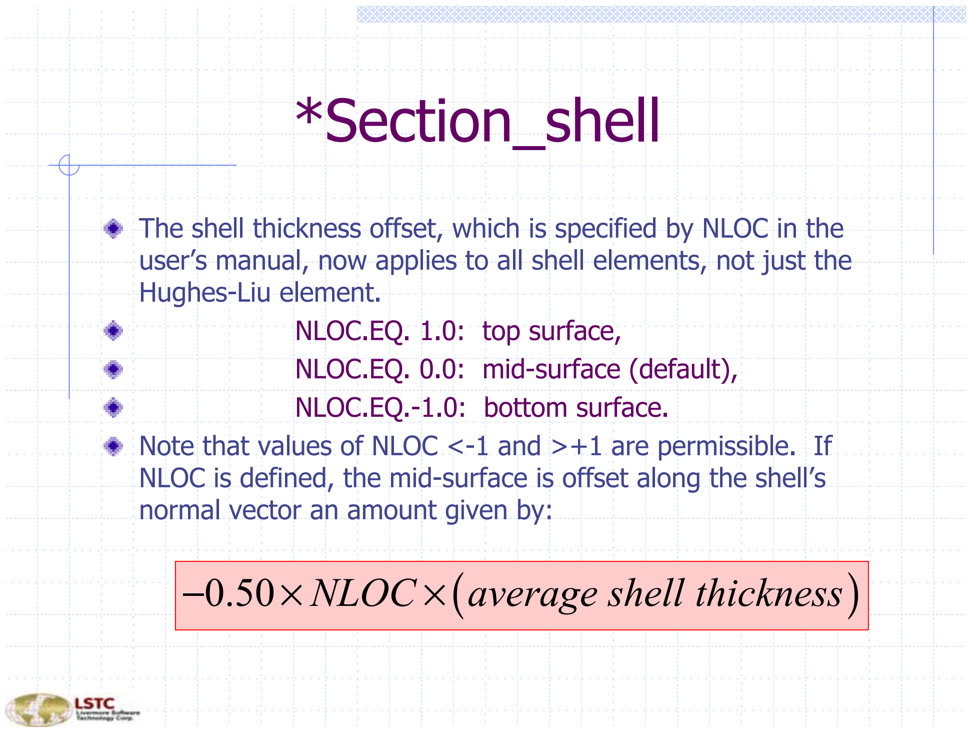
- Perform a transformation on a node set based on a transformation defined by *DEFINE_TRANSFORMATION.
- Requires as input the transformation ID and the node set ID.
- Allows the node set to be translated and rotated as a rigid body
- More general than *Part_move



*Section_beam

- Additional built in sections are now available

Type01: I-shape	Type12: Cross
Type02: Channel	Type13: H-shape
Type03: L-shape	Type14: T-shape1
Type04: T-shape	Type15: I-shape2
Type05: Tubular box	Type16: Channel1
Type06: Z-shape	Type17: Channel2
Type07: Trapezoidal	Type18: T-shape2
Type08: Circular	Type19: Box-shape1
Type09: Tubular	Type20: Hexagon
Type10: I-shape1	Type21: Hat-shape
Type11: Solid box	Type22: Hat-shape1

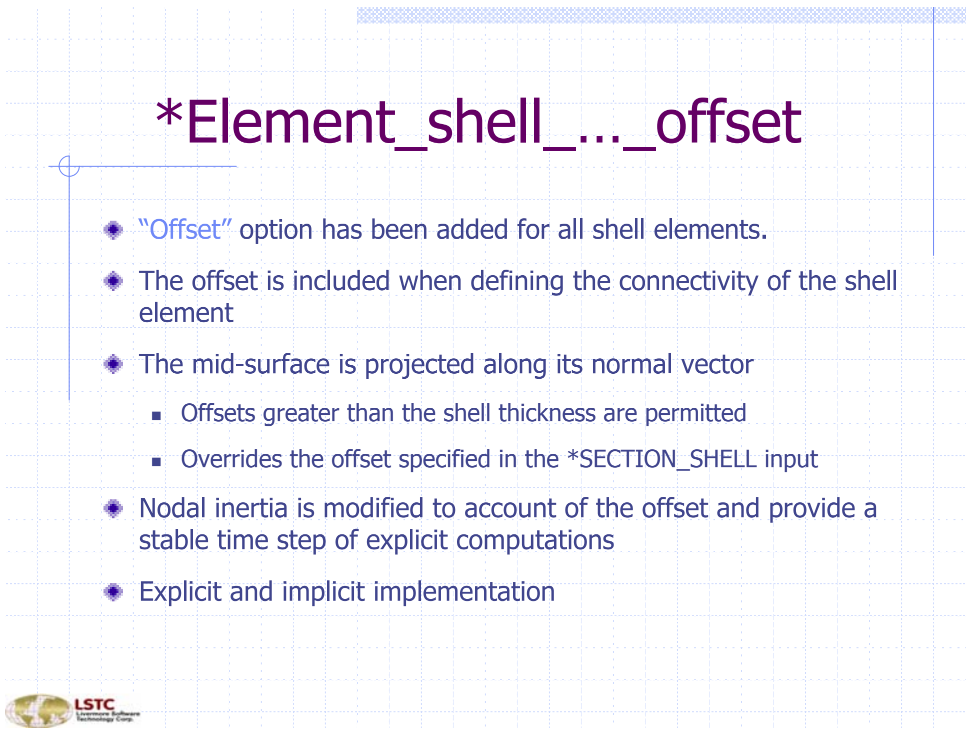


*Section_shell

- The shell thickness offset, which is specified by NLOC in the user's manual, now applies to all shell elements, not just the Hughes-Liu element.
- NLOC.EQ. 1.0: top surface,
- NLOC.EQ. 0.0: mid-surface (default),
- NLOC.EQ.-1.0: bottom surface.
- Note that values of NLOC <-1 and >+1 are permissible. If NLOC is defined, the mid-surface is offset along the shell's normal vector an amount given by:

$$\boxed{-0.50 \times NLOC \times (average\ shell\ thickness)}$$

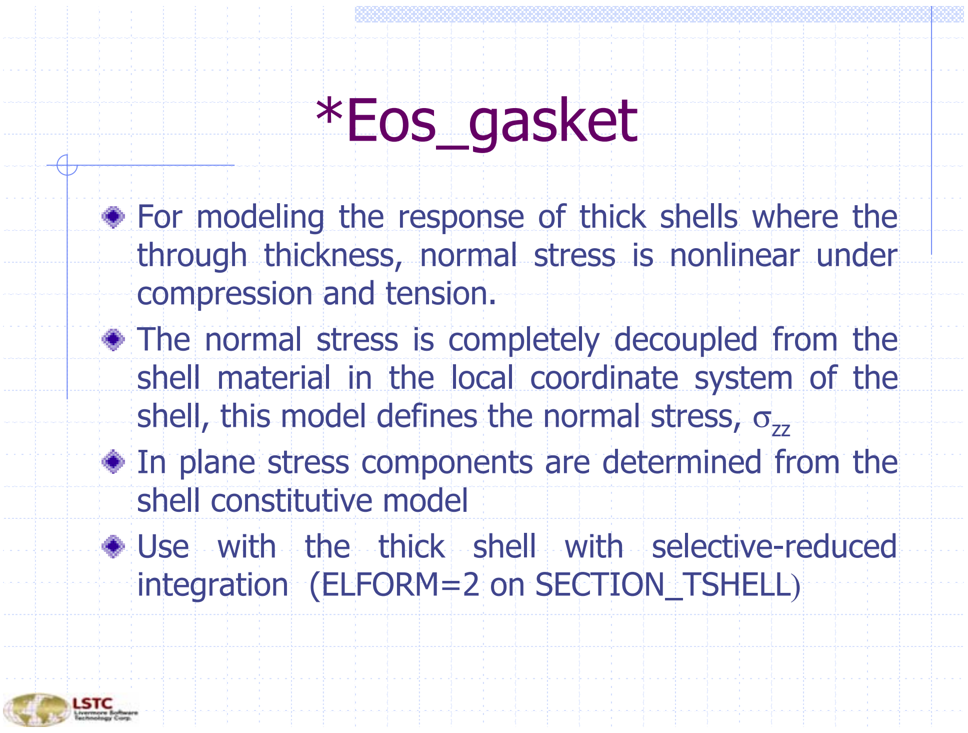
LSTC Livermore Software Technology Corp.



*Element_shell_..._offset

- "Offset" option has been added for all shell elements.
- The offset is included when defining the connectivity of the shell element
- The mid-surface is projected along its normal vector
  - Offsets greater than the shell thickness are permitted
  - Overrides the offset specified in the *SECTION_SHELL input
- Nodal inertia is modified to account of the offset and provide a stable time step of explicit computations
- Explicit and implicit implementation

LSTC Livermore Software Technology Corp.

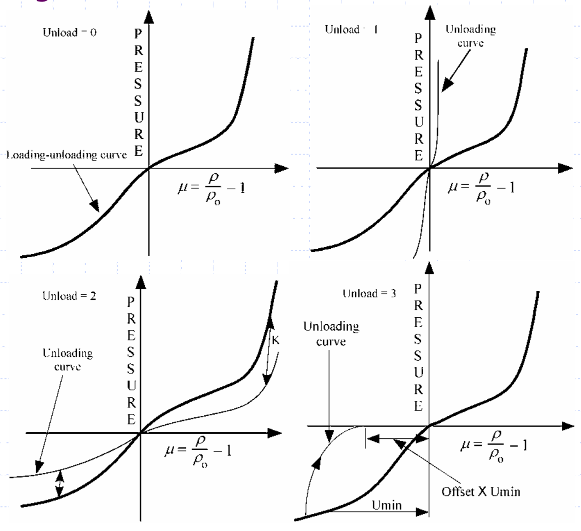


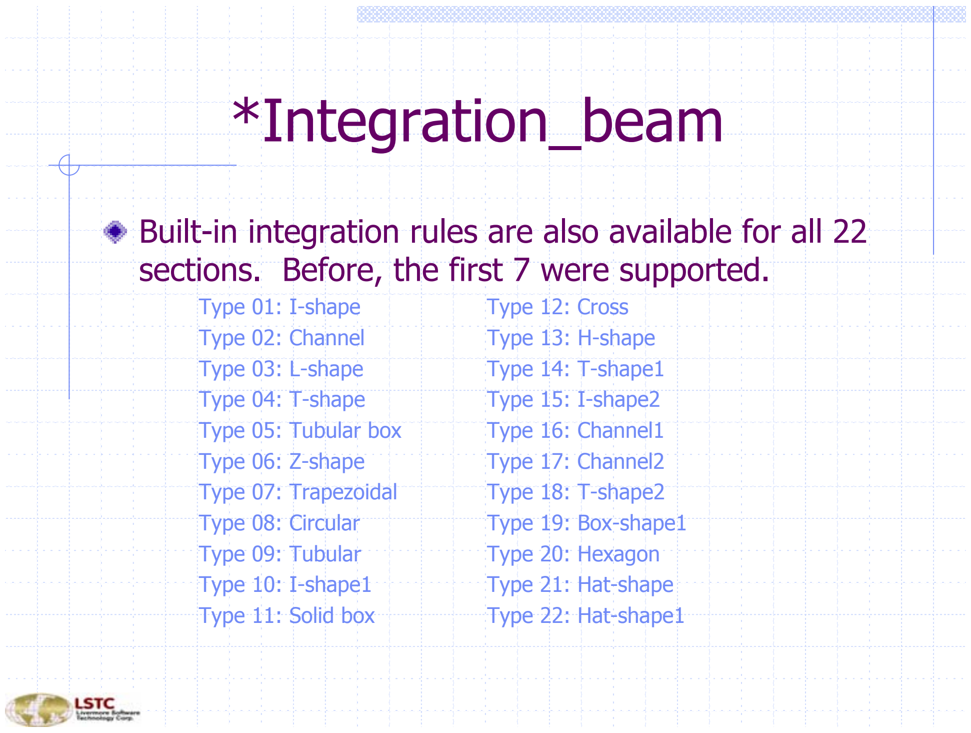
*Eos_gasket

- For modeling the response of thick shells where the through thickness, normal stress is nonlinear under compression and tension.
- The normal stress is completely decoupled from the shell material in the local coordinate system of the shell, this model defines the normal stress, σ_{zz}
- In plane stress components are determined from the shell constitutive model
- Use with the thick shell with selective-reduced integration (ELFORM=2 on SECTION_TSHELL)

*Eos_gasket

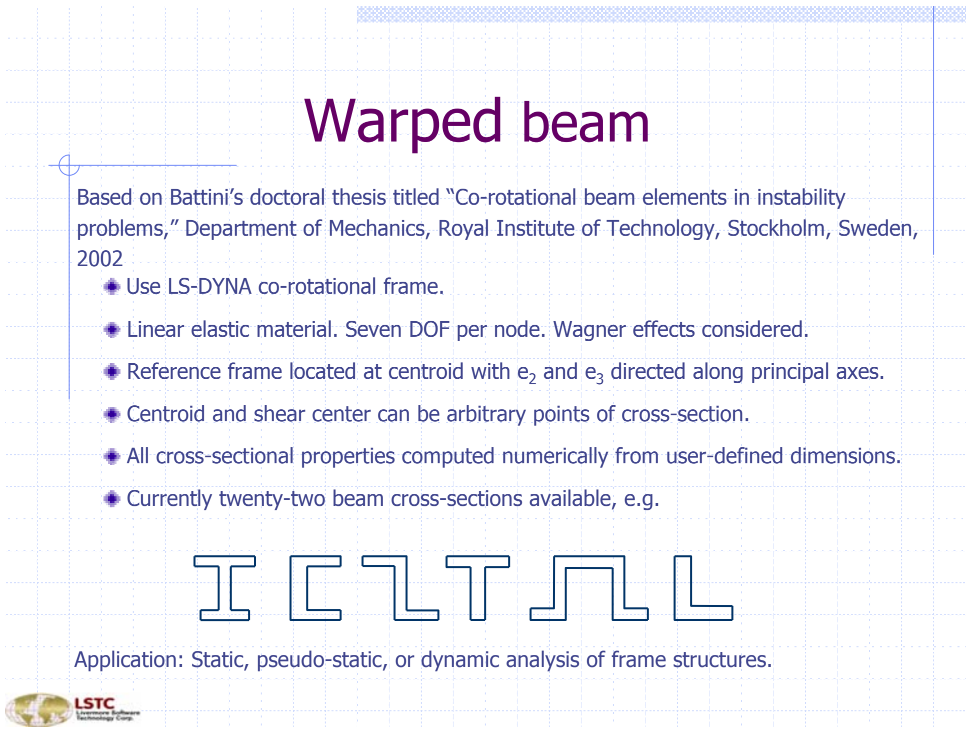
Loading and unload behaviors for normal stress





*Integration_beam

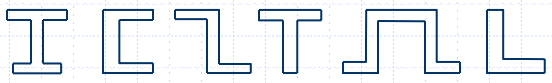
- Built-in integration rules are also available for all 22 sections. Before, the first 7 were supported.
  - Type 01: I-shape
  - Type 02: Channel
  - Type 03: L-shape
  - Type 04: T-shape
  - Type 05: Tubular box
  - Type 06: Z-shape
  - Type 07: Trapezoidal
  - Type 08: Circular
  - Type 09: Tubular
  - Type 10: I-shape1
  - Type 11: Solid box
  - Type 12: Cross
  - Type 13: H-shape
  - Type 14: T-shape1
  - Type 15: I-shape2
  - Type 16: Channel1
  - Type 17: Channel2
  - Type 18: T-shape2
  - Type 19: Box-shape1
  - Type 20: Hexagon
  - Type 21: Hat-shape
  - Type 22: Hat-shape1



Warped beam

Based on Battini's doctoral thesis titled "Co-rotational beam elements in instability problems," Department of Mechanics, Royal Institute of Technology, Stockholm, Sweden, 2002

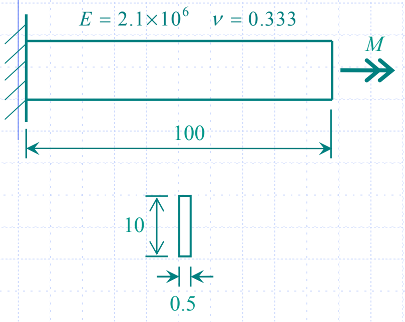
- Use LS-DYNA co-rotational frame.
- Linear elastic material. Seven DOF per node. Wagner effects considered.
- Reference frame located at centroid with e_2 and e_3 directed along principal axes.
- Centroid and shear center can be arbitrary points of cross-section.
- All cross-sectional properties computed numerically from user-defined dimensions.
- Currently twenty-two beam cross-sections available, e.g.

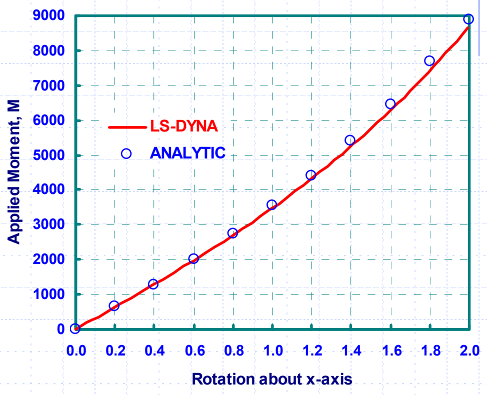


Application: Static, pseudo-static, or dynamic analysis of frame structures.

Warped beam

1. Non-linear Torsion (Goyet)

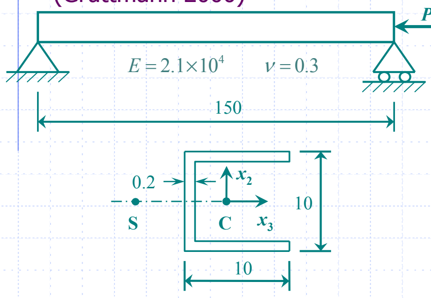


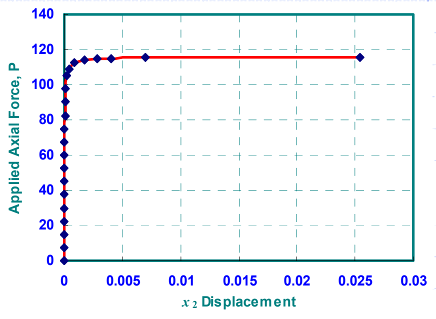


# of Elements	Explicit Elapsed Time	Implicit Elapsed Time
20	121 seconds	2 seconds

Warped beam

2. Lateral Torsional Buckling (Gruttmann 2000)



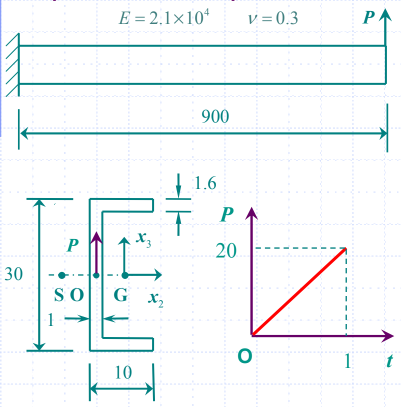


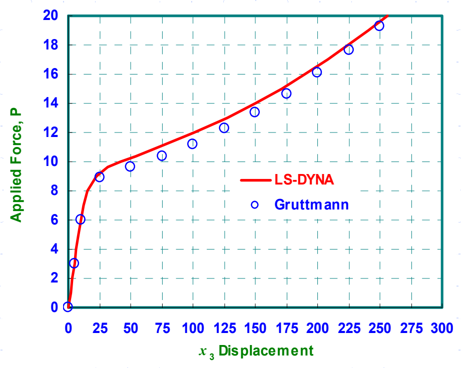
Buckling is initialized by a small tip torque $10^{-5}P$

Lateral Torsional Buckling Load		Flexural Buckling Load (For Comparison)	Pure Torsional Buckling Load (For Comparison)
LS-DYNA	Theory		
115.50	115.54	594.06	125.1

Warped beam

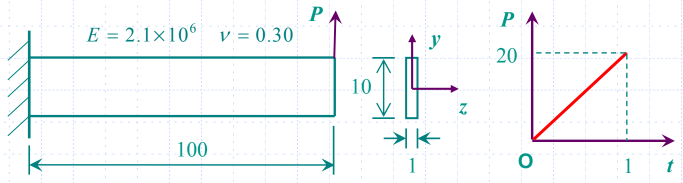
- 3. Channel-Section (Gruttmann 2000)





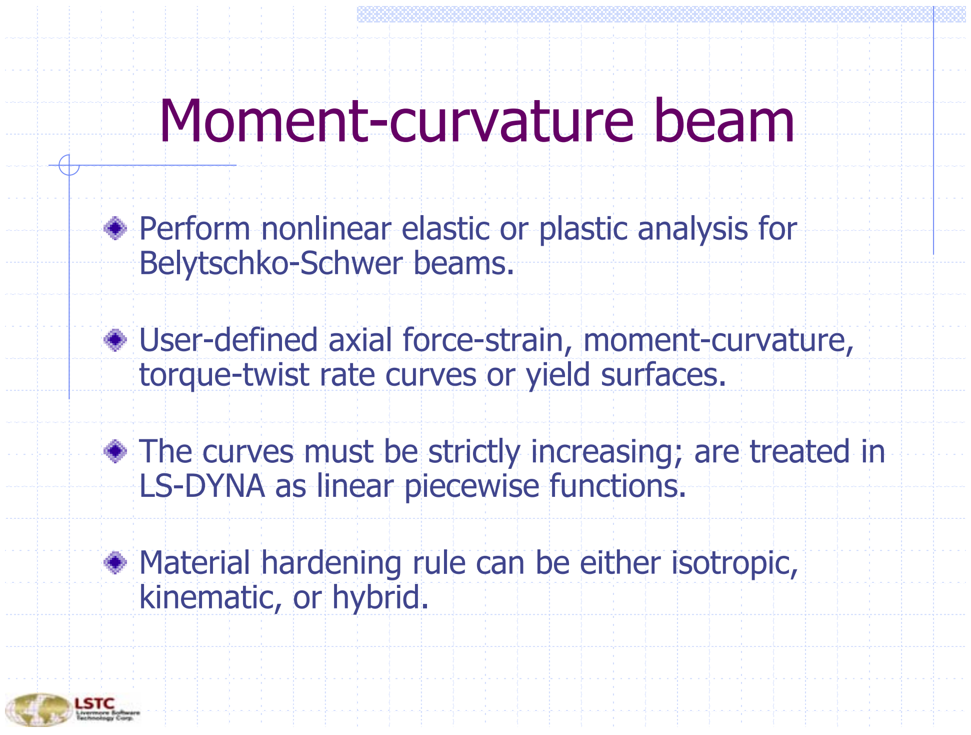
Warped beam

- 4. Lateral Buckling (Battini 2002)



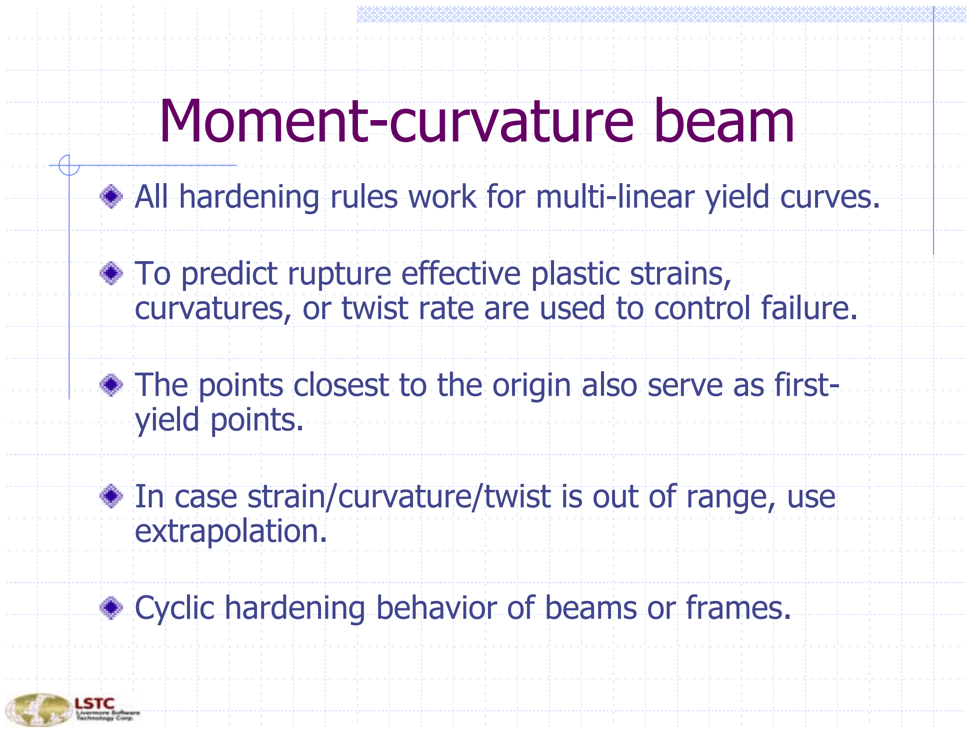
Note: *P* is applied at the centroid of cross-section.

Lateral Buckling Load	
LS-DYNA	Battini (2002)
9.08	9.25



Moment-curvature beam

- Perform nonlinear elastic or plastic analysis for Belytschko-Schwer beams.
- User-defined axial force-strain, moment-curvature, torque-twist rate curves or yield surfaces.
- The curves must be strictly increasing; are treated in LS-DYNA as linear piecewise functions.
- Material hardening rule can be either isotropic, kinematic, or hybrid.



Moment-curvature beam

- All hardening rules work for multi-linear yield curves.
- To predict rupture effective plastic strains, curvatures, or twist rate are used to control failure.
- The points closest to the origin also serve as first-yield points.
- In case strain/curvature/twist is out of range, use extrapolation.
- Cyclic hardening behavior of beams or frames.

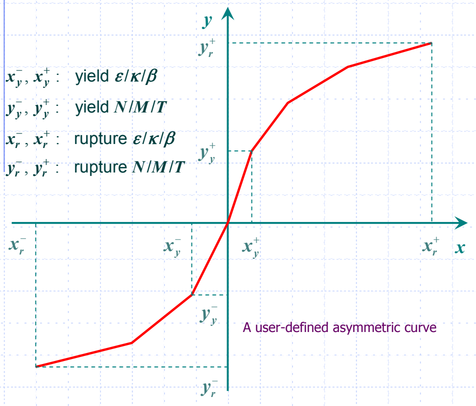
Moment-curvature beam

x_y^-, x_y^+ : yield $\varepsilon/\kappa/\beta$

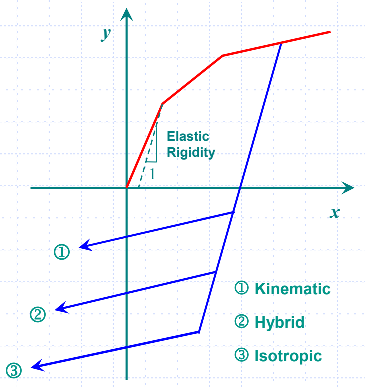
y_y^-, y_y^+ : yield $N/M/T$

x_r^-, x_r^+ : rupture $\varepsilon/\kappa/\beta$

y_r^-, y_r^+ : rupture $N/M/T$



A user-defined asymmetric curve



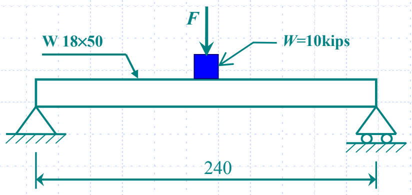
① Kinematic

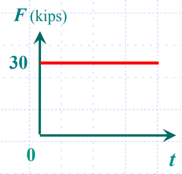
② Hybrid

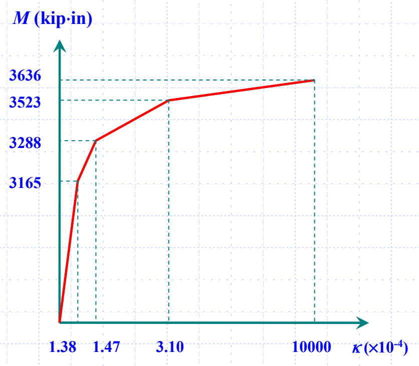
③ Isotropic

Moment-curvature beam

1. Steel I-Shape Beam

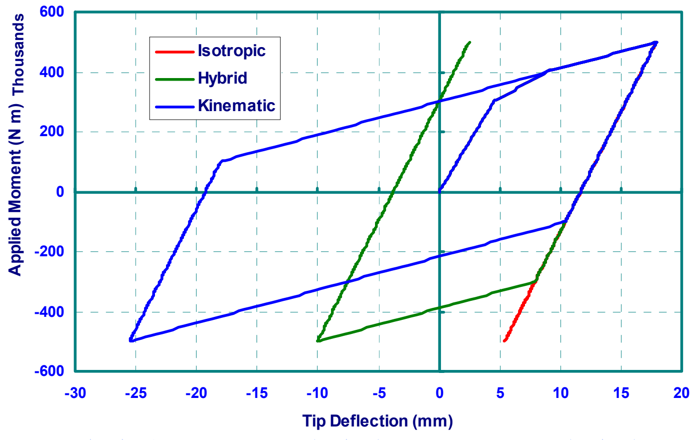






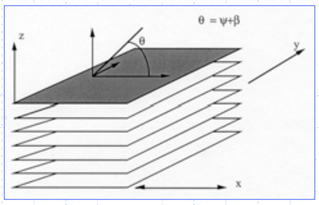
Symmetric Moment-Curvature Curve

Moment-curvature beam



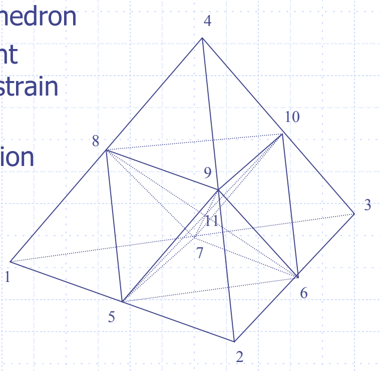
*Integration_shell

- Material types can change from integration point to integration point through the shell thickness.
  - Mix orthotropic, elastic-plastic, and viscoelastic materials
  - New input option to control failure when material types are mixed
    - EQ.0: Element is deleted when the layers which include failure, fail.
    - EQ.1: Element failure cannot occur since some layers do not have a failure option.



Composite tetrahedron

- Based on Belytschko-Guo unpublished paper (1997)
- Ten-node tetrahedron divided into 12 sub-tetrahedrons
- Linear displacement in sub-tetrahedron
- Assumed linear strain, or constant volumetric and linear deviatoric strain over entire tetrahedron
- Implicit and explicit implementation
- SMP-MPP
- Speed ~ fully integrated solid



DYCOS composite failure debonding model

- LS-DYNA's *AUTOMATIC_..._TIEBREAK interface has been enhanced to include the DYCOS debonding model
  - developed jointly by US Navy and Royal Netherlands Navy
  - tiebreak option #7
  - initially bonded interface
  - post-failure: contact with friction and gap opening
- Failure Model
  - elastic - brittle damage (one damage parameter)
  - normal and shear failure stresses
  - softening given by critical energy release rates
  - friction angle adds strength for compressive normal stress

DYCOS composite failure debonding model

- Failure Model
  - Normal, shear tractions computed from secant stiffness

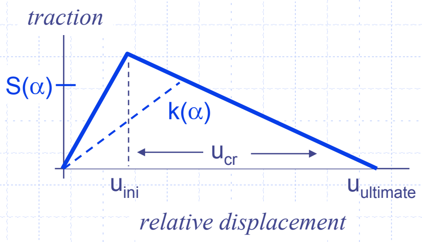
$$\begin{Bmatrix} t_n \\ t_s \end{Bmatrix} = \begin{bmatrix} k_n & 0 \\ 0 & k_s \end{bmatrix} \begin{Bmatrix} u_n \\ u_s \end{Bmatrix}$$

  - Damaged secant stiffness

$$k(\alpha) = \frac{(1-\alpha)u_{ini}}{(\alpha u_{cr} + u_{ini})} k_0$$

  - Damage evolves through yield function

$$f = \left[\frac{\max(t_n,0)}{S_n(\alpha)}\right]^2 + \left[\frac{t_s}{S_s(\alpha)(1-\sin\phi\min(t_n,0))}\right]^2 = 1$$



Mesh-free developments

Mesh-free Basic Features

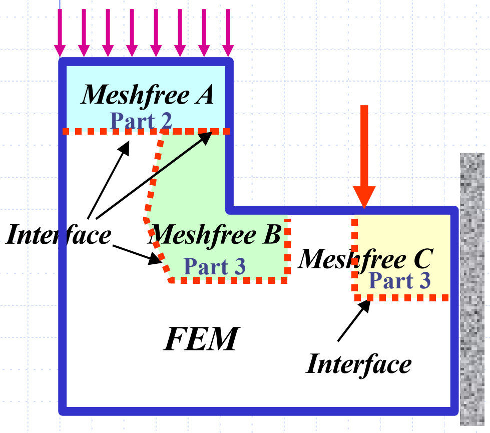
- Higher-order approximation
- Less sensitive to the discretization
- No hourglass control

- Higher CPU cost
- More difficult in theory
- Requires more theoretical developments and refinements

971 developments

- Allow initial meshes based on prisms, tetrahedrons, degenerate solids
- Scalable MPI implementation

Coupled fem/mesh-free



- Start from a regular FE DYNA model
- Divide computational domain into FE and mesh-free zones
- Apply mesh-free approximation in mesh-free zones by defining mesh-free keywords:
  - *CONTROL_EFG
  - *SECTION_SHELL_EFG
  - *SECTION_SOLID_EFG
- Use finite element solver and contact algorithms

Coupled fem/mesh-free

A is the transformation matrix between the general (EFG) and nodal (FEM) displacements

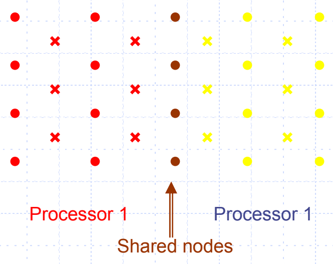
$$\boxed{\mathbf{d}_{\text{FEM}} = \mathbf{A}\mathbf{d}_{\text{EFG}}}$$

$$\mathbf{M}_{\text{FEM}} = \mathbf{A}^{-T}\mathbf{M}_{\text{EFG}}\mathbf{A}^{-1}, \qquad \mathbf{K}_{\text{FEM}} = \mathbf{A}^{-T}\mathbf{K}_{\text{EFG}}\mathbf{A}^{-1}, \qquad \mathbf{R}_{\text{FEM}} = \mathbf{A}^{-T}\mathbf{R}_{\text{EFG}}$$

$$\mathbf{d}_{\text{EFG}} = \mathbf{A}^{-1}\mathbf{d}_{\text{FEM}}$$

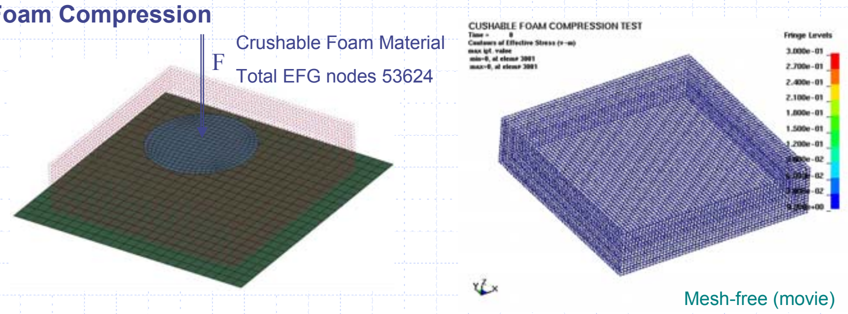
Parallel mesh-free

- SMP
  - Scalability depends on the efficiency of the matrix multiplication with the transformation matrix
- MPP
  - Parallel assembly and factorization of A required. Parallel triangular solves may affect scaling-should be in core for efficiency
  - More communication in an EFG region among processors
    - ✓ More neighbors involve in nodal force computation
    - ✓ Another set of neighbors in transformation



Parallel mesh-free computation

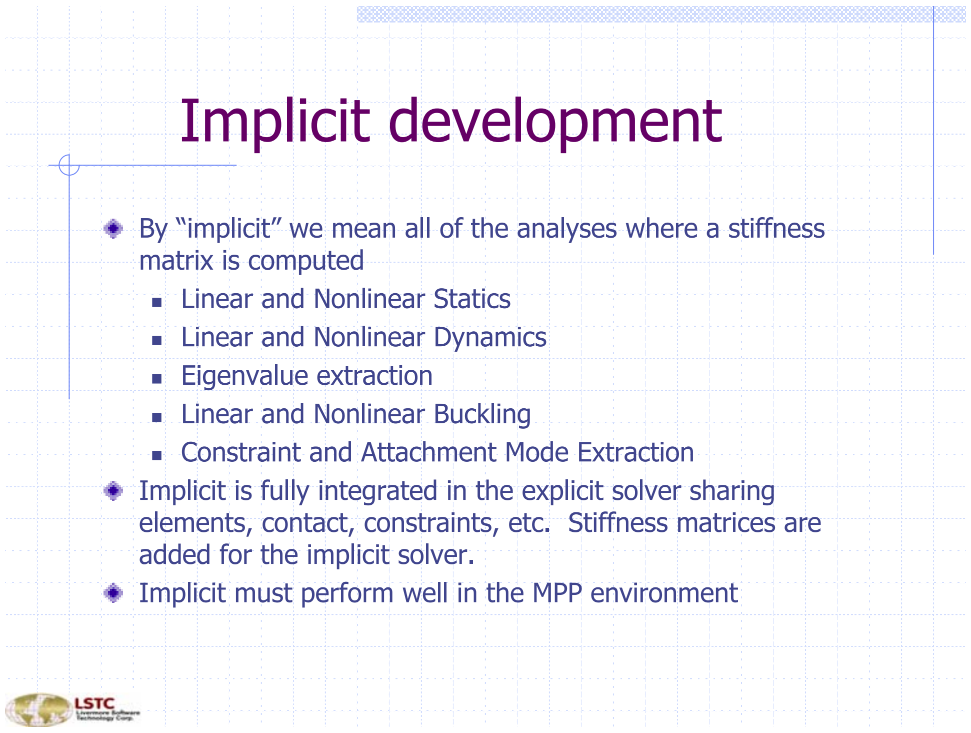
Foam Compression



Mesh-free (movie)

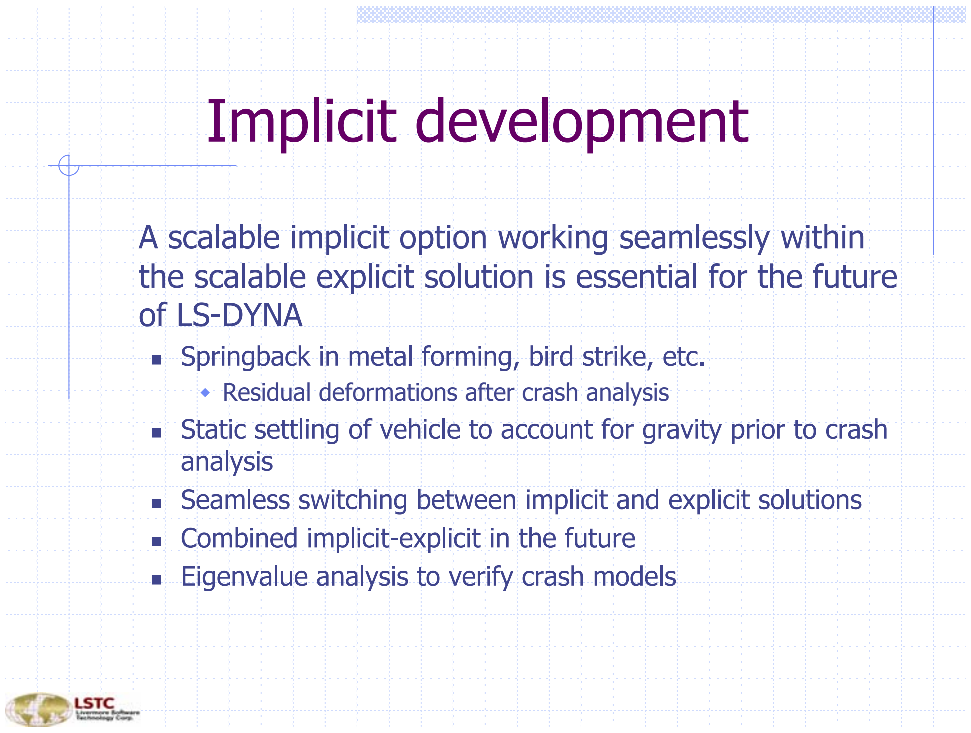
Normalized CPU Time

No. of CPU's	1	2	4	8
SMP	1.00	0.55	0.43	0.32
MPP	0.99	0.52	0.25	0.15



Implicit development

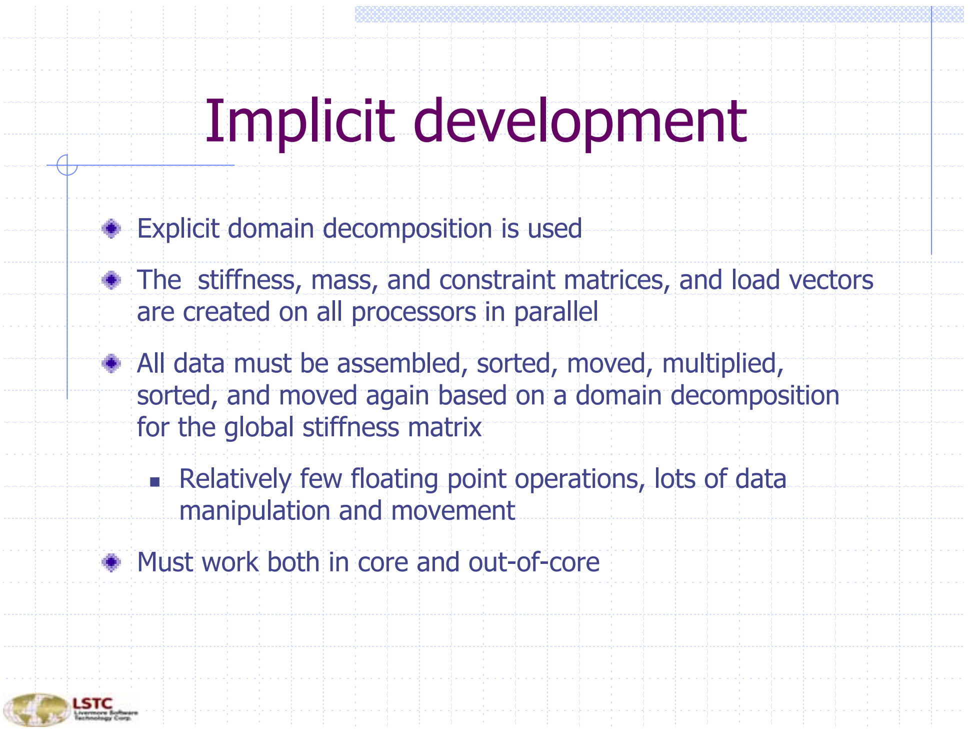
- By "implicit" we mean all of the analyses where a stiffness matrix is computed
  - Linear and Nonlinear Statics
  - Linear and Nonlinear Dynamics
  - Eigenvalue extraction
  - Linear and Nonlinear Buckling
  - Constraint and Attachment Mode Extraction
- Implicit is fully integrated in the explicit solver sharing elements, contact, constraints, etc. Stiffness matrices are added for the implicit solver.
- Implicit must perform well in the MPP environment



Implicit development

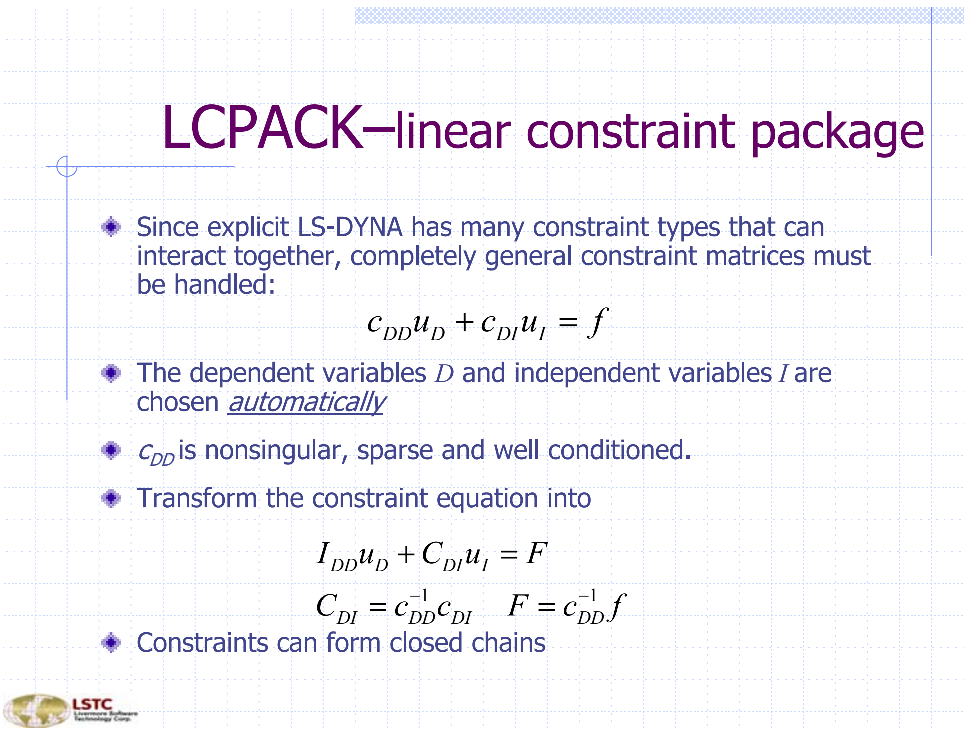
A scalable implicit option working seamlessly within the scalable explicit solution is essential for the future of LS-DYNA

- Springback in metal forming, bird strike, etc.
  - Residual deformations after crash analysis
- Static settling of vehicle to account for gravity prior to crash analysis
- Seamless switching between implicit and explicit solutions
- Combined implicit-explicit in the future
- Eigenvalue analysis to verify crash models



Implicit development

- Explicit domain decomposition is used
- The stiffness, mass, and constraint matrices, and load vectors are created on all processors in parallel
- All data must be assembled, sorted, moved, multiplied, sorted, and moved again based on a domain decomposition for the global stiffness matrix
  - Relatively few floating point operations, lots of data manipulation and movement
- Must work both in core and out-of-core



LCPACK–linear constraint package

- Since explicit LS-DYNA has many constraint types that can interact together, completely general constraint matrices must be handled:

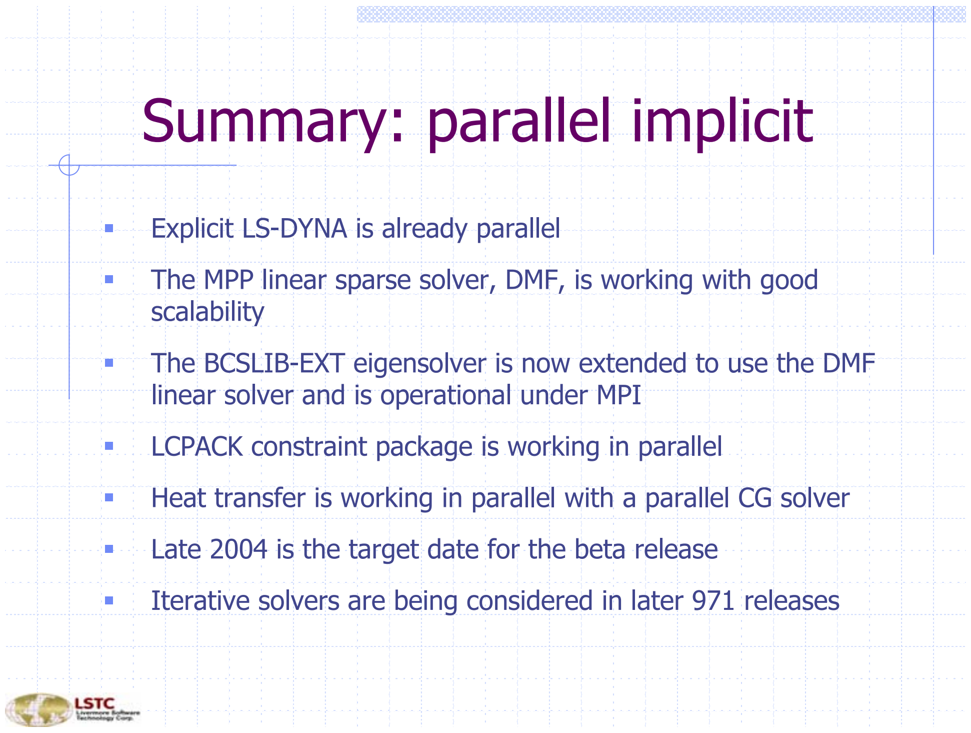
$$c_{DD}u_D + c_{DI}u_I = f$$

- The dependent variables D and independent variables I are chosen *automatically*
- c_{DD} is nonsingular, sparse and well conditioned.
- Transform the constraint equation into

$$I_{DD}u_D + C_{DI}u_I = F$$

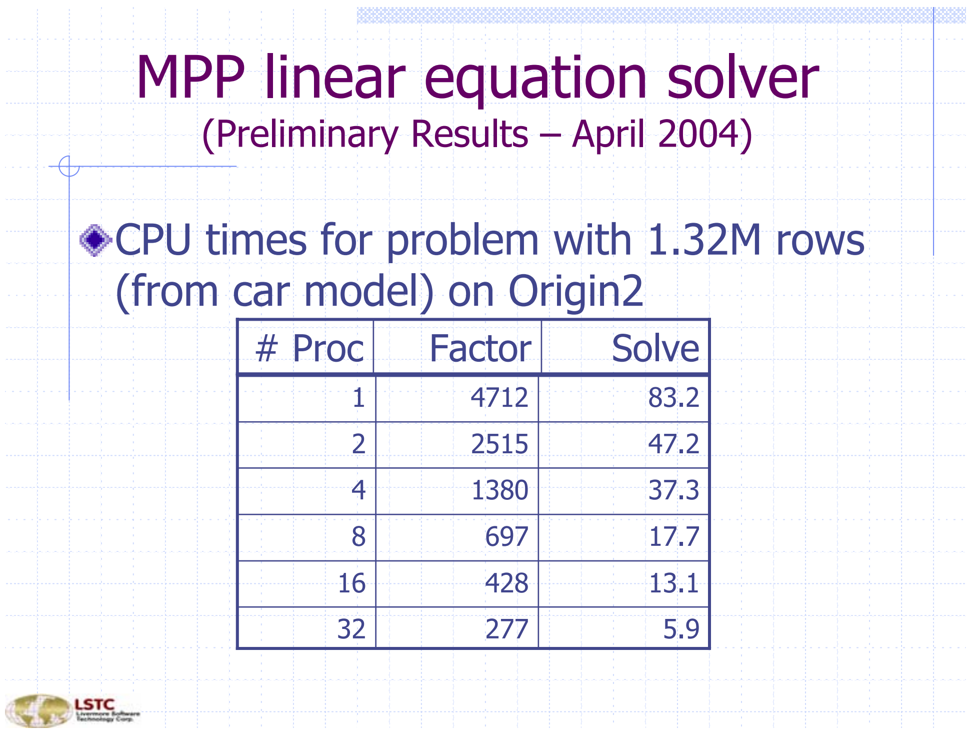
$$C_{DI} = c_{DD}^{-1}c_{DI} \quad F = c_{DD}^{-1}f$$

- Constraints can form closed chains



Summary: parallel implicit

- Explicit LS-DYNA is already parallel
- The MPP linear sparse solver, DMF, is working with good scalability
- The BCSLIB-EXT eigensolver is now extended to use the DMF linear solver and is operational under MPI
- LCPACK constraint package is working in parallel
- Heat transfer is working in parallel with a parallel CG solver
- Late 2004 is the target date for the beta release
- Iterative solvers are being considered in later 971 releases

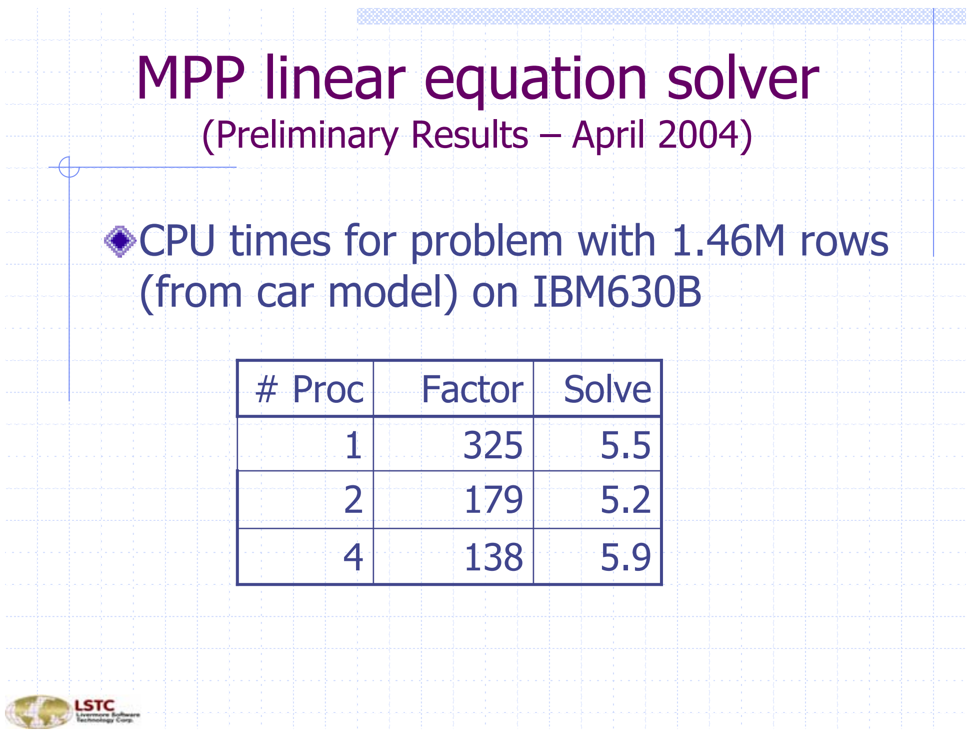


MPP linear equation solver

(Preliminary Results – April 2004)

- CPU times for problem with 1.32M rows (from car model) on Origin2

# Proc	Factor	Solve
1	4712	83.2
2	2515	47.2
4	1380	37.3
8	697	17.7
16	428	13.1
32	277	5.9

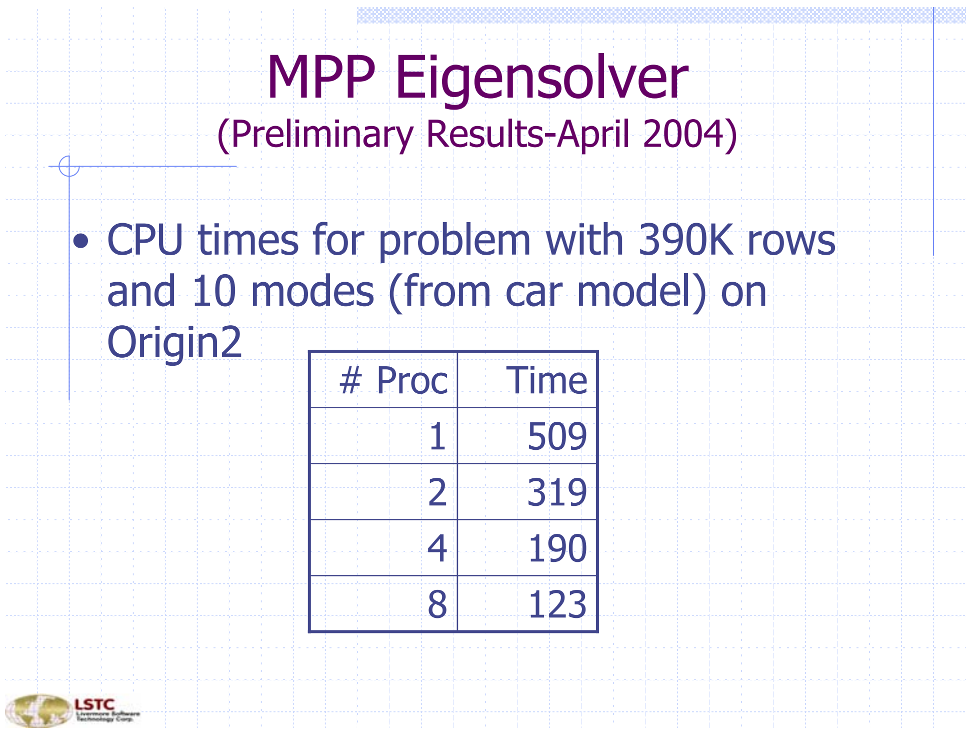


MPP linear equation solver

(Preliminary Results – April 2004)

- CPU times for problem with 1.46M rows (from car model) on IBM630B

# Proc	Factor	Solve
1	325	5.5
2	179	5.2
4	138	5.9

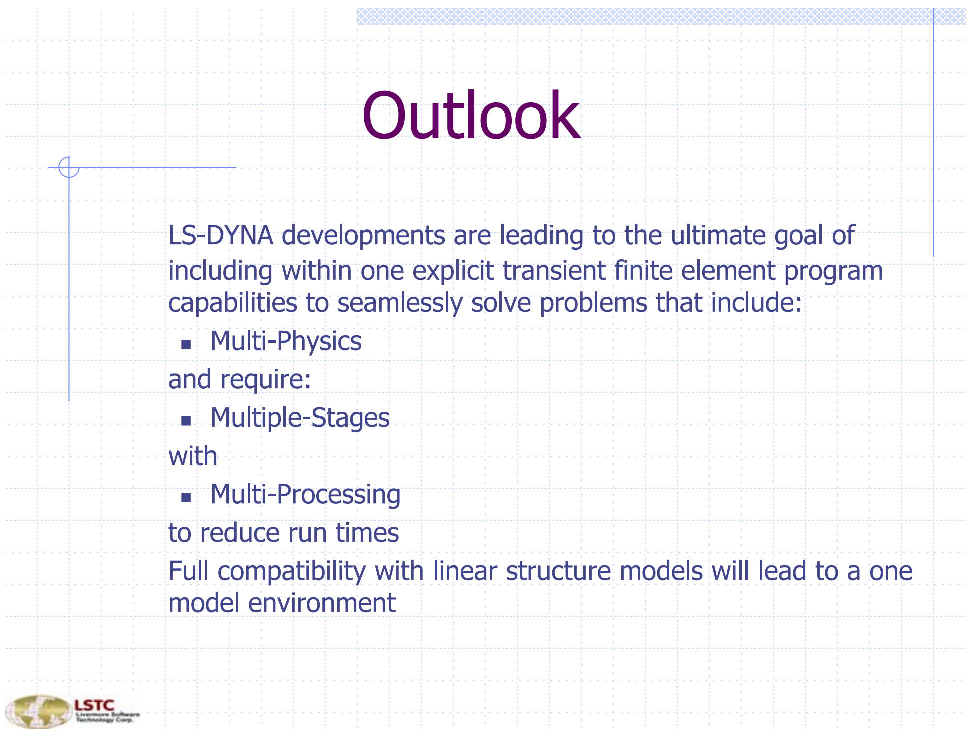


MPP Eigensolver

(Preliminary Results-April 2004)

- CPU times for problem with 390K rows and 10 modes (from car model) on Origin2

# Proc	Time
1	509
2	319
4	190
8	123



Outlook

LS-DYNA developments are leading to the ultimate goal of including within one explicit transient finite element program capabilities to seamlessly solve problems that include:

- Multi-Physics

and require:

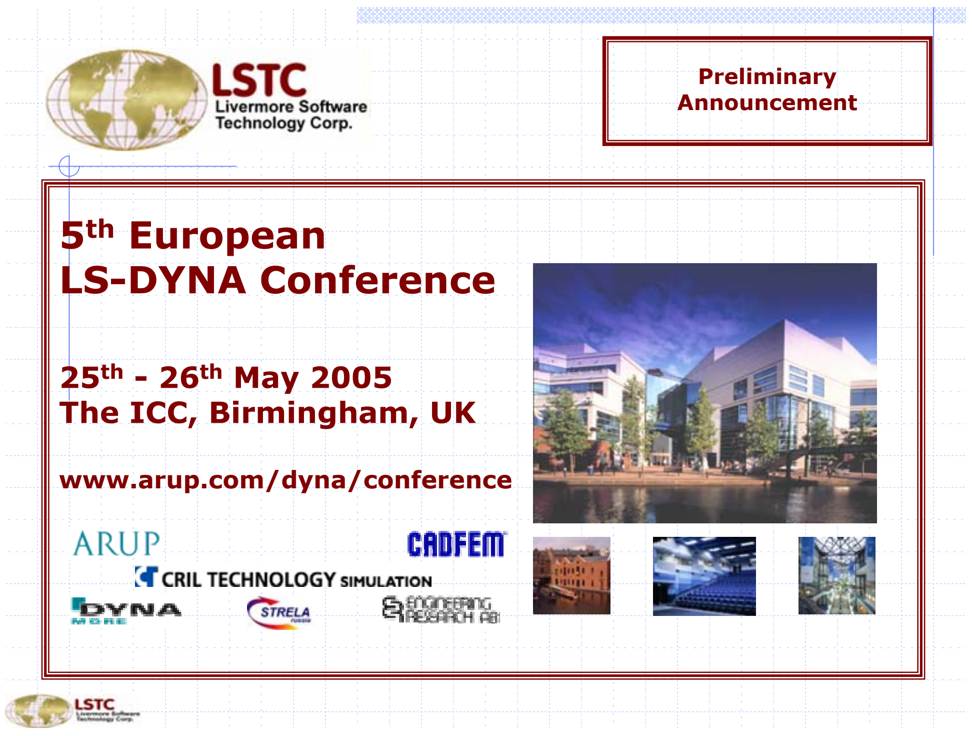
- Multiple-Stages

with

- Multi-Processing

to reduce run times

Full compatibility with linear structure models will lead to a one model environment



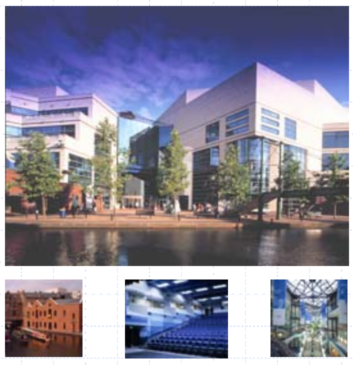
LSTC Livermore Software Technology Corp.

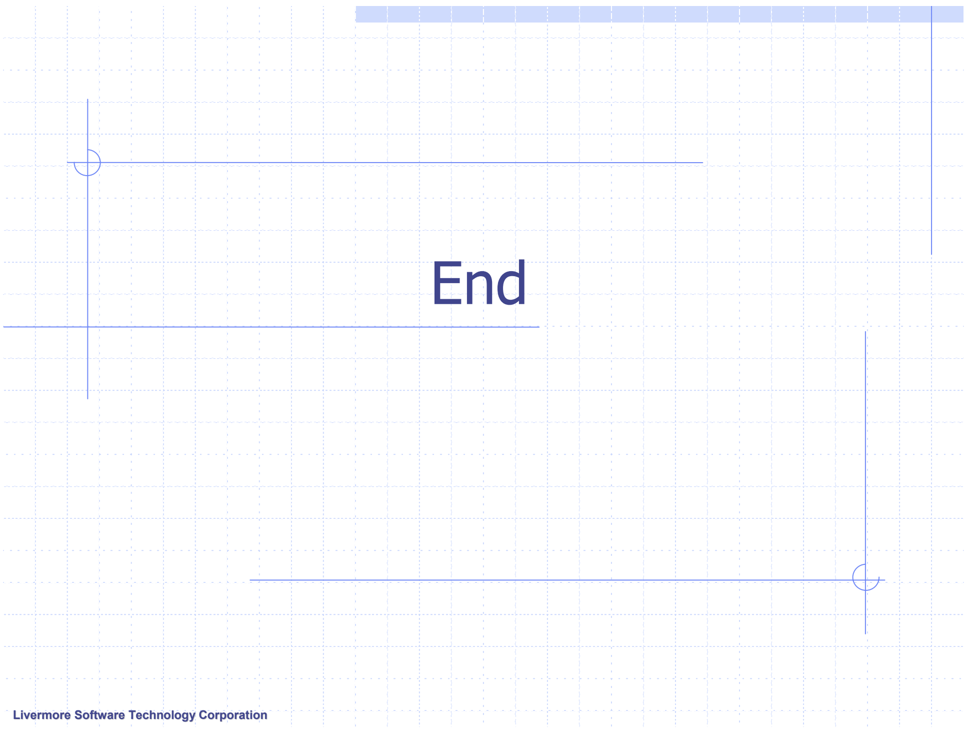
Preliminary Announcement

5th European LS-DYNA Conference

25th - 26th May 2005
The ICC, Birmingham, UK

www.arup.com/dyna/conference





End

Livermore Software Technology Corporation