AD FALCON API Manual

ALE Mesh Smoothing

Javad Ghorbani

March 26, 2026

Artemis Dev / AD FALCON API | artemisdvlp.com

Contents

1 ALE Mesh Smoothing

This page describes FALCON's Arbitrary Lagrangian-Eulerian (ALE) mesh smoothing capability for maintaining mesh quality in large deformation analyses.

1.1 Overview

In finite element simulations involving large deformations, the computational mesh can become severely distorted, leading to:

- Negative Jacobian determinants (inverted elements)
- Poor element aspect ratios
- Inaccurate stress/strain integration
- Solver convergence failures

FALCON's ALE mesh smoother addresses these issues by periodically relocating interior nodes to improve element quality while preserving the material stress state at Gauss points. This allows simulations to proceed beyond the point where a purely Lagrangian approach would fail.

1.1.1 Key Features

- **Multiple smoothing algorithms**: Pure Laplacian, Weighted Laplacian, and Smart Laplacian methods
- **Optional quality-triggered activation**: When `@@QualityTriggeredOnly: Yes`, smoothing runs only when quality thresholds are violated
- **Boundary preservation**: Exterior boundaries, contact surfaces, and rigid body interfaces are handled correctly
- **Stress preservation**: Cauchy stress at Gauss points is preserved during mesh relocation
- **Autoinc-safe validation**: If smoothing cannot maintain positive Gauss Jacobians, the mesh is reverted and the substep is marked failed (so autoinc can retry with smaller Δt when enabled)
- **Fully coupled analysis support**: Compatible with hydro-mechanical analyses

---

1.2 Theoretical Background

1.2.1 ALE Formulation

The ALE method decouples the mesh motion from the material motion. In a purely Lagrangian formulation, mesh nodes move with the material:

$$\mathbf{x}_{\text{mesh}} = \mathbf{x}_{\text{material}}$$

In the ALE approach, mesh nodes can move independently:

$$\mathbf{x}_{\text{mesh}} = \mathbf{x}_{\text{material}} + \mathbf{u}_{\text{ALE}}$$

where $\mathbf{u}_{\text{ALE}}$ is the mesh relocation displacement computed by the smoothing algorithm.

1.2.2 Laplacian Smoothing

The core smoothing algorithm relocates each interior node to a weighted average of its neighbors' positions:

$$\mathbf{x}_i^{\text{new}} = (1-\omega)\mathbf{x}_i^{\text{old}} + \omega \sum_{j \in \mathcal{N}_i} w_{ij}\mathbf{x}_j$$

where:

- $\omega \in (0, 1]$ is the relaxation factor
- $\mathcal{N}_i$ is the set of nodes adjacent to node i
- w_{ij} are the neighbor weights (normalized to sum to 1)

The algorithm iterates until convergence or a maximum iteration count is reached.

In FALCON, the neighbor weights are always taken from the selected `@@WeightingScheme` (even for `PureLaplacian`). The difference between smoothing methods is primarily *how candidate moves are accepted and stabilized*, not whether weights are used.

1.2.3 Weighting Schemes

FALCON implements four weighting schemes:

Uniform weighting:

$$w_{ij} = \frac{1}{|\mathcal{N}_i|}$$

Inverse distance weighting:

$$w_{ij} = \frac{1/d_{ij}}{\sum_{k \in \mathcal{N}_i} 1/d_{ik}}, \quad d_{ij} = \|\mathbf{x}_j - \mathbf{x}_i\|$$

Area-based weighting (accounts for element sizes):

$$w_{ij} = \frac{A_{ij}}{\sum_{k \in \mathcal{N}_i} A_{ik}}$$

where A_{ij} is the total area of elements sharing edge (i, j).

Quality-based weighting (favors neighbors of poor quality elements):

$$w_{ij} = \frac{W_{ij}}{\sum_{k \in \mathcal{N}_i} W_{ik}}$$

where W_{ij} is computed by accumulating contributions from all elements e that contain both nodes i and j:

$$W_{ij} = \sum_{e \in \mathcal{E}_{ij}} (1 + q_e), \quad q_e = 1 - \max(0, \min(1, J_e))$$

Here, $\mathcal{E}_{ij}$ is the set of elements containing both nodes, J_e is the scaled Jacobian of element e, and q_e is the quality deficiency factor. Elements with low scaled Jacobian (poor quality) contribute higher weights, directing smoothing effort toward distorted regions.

1.2.4 Smart Laplacian

The Smart Laplacian method enhances basic Laplacian smoothing by:

1. Computing a candidate position using weighted averaging (Laplacian target)
2. Applying adaptive relaxation and per-node displacement limits
3. Performing a local line search (step halving) to ensure all *connected* elements keep positive Gauss point Jacobians after midpoint recomputation
4. Accepting the move only if a local quality penalty improves; otherwise rejecting the move (no “damage then rollback”)

This prevents the algorithm from degrading already-good elements while improving poor ones.

1.2.5 Stress Handling in ALE

During smoothing, the mesh is relocated while the material state remains Lagrangian. This deferred stress correction procedure (Benson, 1989; Benson, 1992), let the constitutive integration in the next Lagrangian step pull the stress back onto the yield surface.

In Updated Lagrangian analyses, ALE mesh motion updates the *nodal coordinates* (mesh geometry) without introducing artificial material displacement; stresses/history remain defined at Gauss points and are advanced by the constitutive model in the subsequent step.

Equilibrium Re-establishment

After the geometry update, the strain-displacement matrix $\mathbf{B}$ reflects the new mesh configuration, and equilibrium is re-established through:

$$\mathbf{F}_{\text{int}} = \int_{\Omega} \mathbf{B}^T \boldsymbol{\sigma} \, dV$$

1.3 Mesh Quality Metrics

FALCON uses several metrics to assess element quality:

1.3.1 Minimum/Maximum Angles

For triangular and quadrilateral elements, the internal angles are computed. Quality thresholds are:

- **Minimum angle**: Elements with angles below this are flagged (default: 15°)
- **Maximum angle**: Elements with angles above this are flagged (default: 165°)

1.3.2 Aspect Ratio

The ratio of the longest edge to the shortest edge:

$$\text{AR} = \frac{\max(L_{\text{edges}})}{\min(L_{\text{edges}})}$$

Ideal elements have AR ≈ 1. Elements with AR exceeding the threshold are flagged.

1.3.3 Jacobian Determinant

The determinant of the Jacobian matrix at Gauss points:

$$J = \det\left(\frac{\partial \mathbf{x}}{\partial \xi}\right)$$

where ξ are the natural coordinates. Positive J indicates a valid element; $J \leq 0$ indicates inversion.

1.3.4 Scaled Jacobian

The **scaled Jacobian** is a normalized quality metric that compares the actual element area to an ideal element with the same characteristic dimension. It ranges from 0 (degenerate) to 1 (ideal shape):

For triangular elements:

$$J_{\text{scaled}} = \frac{A_{\text{actual}}}{A_{\text{ideal}}} = \frac{A}{\frac{\sqrt{3}}{4} L_{\max}^2}$$

where A is the actual element area and $L_{\max}$ is the longest edge length. The ideal area is that of an equilateral triangle with edge length $L_{\max}$.

For quadrilateral elements:

$$J_{\text{scaled}} = \frac{A_{\text{actual}}}{A_{\text{ideal}}} = \frac{A}{\frac{1}{2} D_{\text{avg}}^2}$$

where $D_{\text{avg}} = \frac{1}{2}(D_1 + D_2)$ is the average diagonal length. The ideal area is that of a square with the same average diagonal.

Interpretation: - $J_{\text{scaled}} = 1.0$: Ideal element (equilateral triangle or square) - $J_{\text{scaled}} > 0.5$: Good quality element - $J_{\text{scaled}} < 0.1$: Poor quality, triggers smoothing (default threshold) - $J_{\text{scaled}} \leq 0$: Inverted element (invalid)

The scaled Jacobian is used in quality-triggered smoothing and in the quality-based weighting scheme to identify and prioritize problematic elements.

1.4 Syntax

ALE mesh smoothing is configured using the `% ALE Configuration` section:

```
% ALE Configuration
@@Enabled: Yes
@@Method: SmartLaplacian
@@WeightingScheme: InverseDistance
...
%%%
```

Notes:

- Section headers such as `% ALEConfiguration` are also accepted.
- Directive keys are case-insensitive and accept one or more leading `@` characters (for example, `@Enabled:` and `@@Enabled:` are equivalent).
- A space after `:` is optional (`@@Method:SmartLaplacian` and `@@Method: SmartLaplacian` both work).
- Lists such as `@@FixedNodes` and `@@SlidingNodes` accept IDs separated by spaces/commas/semicolons and allow ranges (`10-20` or `10:20`). `@@ActiveSteps` accepts a list of integers (no ranges) or the keyword `All`.
- `@@Method` and `@@WeightingScheme` values are case-insensitive (for example, `smart`, `SmartLaplacian`).
- Common aliases: `@@Method`/`@@SmoothingMethod`, `@@RelaxationFactor`/`@@Omega`, `@@ConvergenceTolerance`/`@@Tolerance`, `@@JacobianThreshold`/`@@ScaledJacobianThreshold`, `@@MaxNodeDisplacement`/`@@MaxDisp`, `@@WriteDebugOutput`/`@@Debug`.

```
% ALE Configuration
@@Enabled: Yes
@@Method: SmartLaplacian
@@WeightingScheme: InverseDistance
@@RelaxationFactor: 0.3
@@MaxIterations: 50
@@MinAngleThreshold: 15.0
@@MaxAngleThreshold: 165.0
@@AspectRatioThreshold: 5.0
@@JacobianThreshold: 0.1
@@FixExteriorBoundary: Yes
@@FixContactNodes: Yes
@@FixRigidNodes: Yes
@@QualityTriggeredOnly: No
@@AdaptiveRelaxation: Yes
@@MaxNodeDisplacement: 0.05
```

```
@@MaxCumulativeDisplacement: 0.5
@@SmoothingFrequency: 1
@@ActiveSteps: 1 2 3
%%%
```

1.5 Configuration Parameters

1.5.1 Core Parameters

Parameter	Type	Default	Description
`@@Enabled`	Yes/No	No	Enable or disable ALE mesh smoothing
`@@Id`	Integer	-1	Configuration ID (for multi-zone setups)
`@@Method`	String	SmartLaplacian	Smoothing algorithm (see below). Alias: `@@SmoothingMethod`.
`@@WeightingScheme`	String	InverseDistance	Neighbor weighting scheme (see below)
`@@RelaxationFactor`	Double	0.5	Relaxation parameter $\omega \in (0, 1]$. Alias: `@@Omega`.
`@@MaxIterations`	Integer	50	Maximum smoothing iterations per call
`@@MinIterations`	Integer	3	Minimum smoothing iterations (even if converged)
`@@ConvergenceTolerance`	Double	1.0e-6	Convergence threshold for smoothing (max node displacement). Alias: `@@Tolerance`.

Smoothing Methods:

Method	Description
`PureLaplacian`	Laplacian averaging with relaxation and displacement limits (no quality-based acceptance)
`WeightedLaplacian`	Same behavior as `PureLaplacian` in the current release (reserved for future differentiation)

Method	Description
`SmartLaplacian`	Quality-aware Laplacian with adaptive relaxation and local line-search acceptance (recommended)

Weighting Schemes:

Scheme	Description
`Uniform`	Equal weights for all neighbors
`InverseDistance`	Weights inversely proportional to distance
`AreaBased`	Weights based on adjacent element areas
`QualityBased`	Weights based on element quality (higher weight for poor quality elements)

1.5.2 Quality Thresholds

Parameter	Type	Default	Description
`@@MinAngleThreshold`	Double	15.0	Minimum acceptable interior angle (degrees)
`@@MaxAngleThreshold`	Double	165.0	Maximum acceptable interior angle (degrees)
`@@AspectRatioThreshold`	Double	5.0	Maximum acceptable aspect ratio
`@@JacobianThreshold`	Double	0.1	Minimum acceptable scaled Jacobian. Alias: `@@ScaledJacobianThreshold`.

Elements violating any threshold are flagged as poor quality and trigger smoothing (if `QualityTriggeredOnly: Yes`).

1.5.3 Boundary Handling

Parameter	Type	Default	Description
`@@FixExteriorBoundary`	Yes/No	Yes	Fix exterior boundary *corner/principal* nodes (quadratic mid-edge nodes are recomputed)
`@@FixContactNodes`	Yes/No	Yes	Fix nodes involved in contact pairs

Parameter	Type	Default	Description
`@@FixRigidNodes`	Yes/No	Yes	Fix nodes belonging to rigid bodies
`@@AllowBoundarySliding`	Yes/No	No	Reserved (sliding is controlled by `@@SlidingNodes`)

Corner nodes are always fixed to preserve domain topology.

Quadratic mid-edge nodes (T6/Q8 midside nodes) are not smoothed directly. After smoothing corner nodes, mid-edge nodes are recomputed as geometric midpoints of their adjacent corners to maintain element consistency. As a result, mid-edge boundary nodes are not treated as fixed by `FixExteriorBoundary`; explicitly fixing mid-edge nodes can over-constrain the element.

When `@@FixContactNodes` / `@@FixRigidNodes` are enabled, the fixed set is updated based on the *current* contact pairs and rigid body membership (nodes are not permanently “stuck” as fixed if those sets change).

1.5.4 Node Lists (Optional)

Parameter	Type	Default	Description
`@@FixedNodes`	List	—	Explicitly fix these node IDs in addition to automatic fixing
`@@SlidingNodes`	List	—	Boundary node IDs allowed to slide tangentially along the boundary (if not fixed)

1.5.5 Displacement Limits

Parameter	Type	Default	Description
`@@MaxNodeDisplacement`	Double	0.01	Maximum displacement per node *per smoothing iteration* (fraction of local characteristic element size). Alias: `@@MaxDisp`.

Parameter	Type	Default	Description
`@@MaxCumulativeDisplacement`	Double	0.1	Maximum accumulated mesh motion per simulation step (absolute distance; tracked as the sum of per-call max node displacement).

These limits prevent excessive mesh motion that could cause numerical instabilities.

> Implementation note (behavioral): FALCON additionally applies a conservative hard cap on the per-iteration node move based on local characteristic element size, so increasing `@@MaxNodeDisplacement` above that cap may not increase the actual move per iteration.

1.5.6 Activation Control

Parameter	Type	Default	Description
`@@QualityTriggeredOnly`	Yes/No	No	Only smooth when quality thresholds are violated
`@@AdaptiveRelaxation`	Yes/No	Yes	Adjust relaxation based on local quality
`@@MinRelaxation`	Double	0.1	Minimum adaptive relaxation factor
`@@MaxRelaxation`	Double	0.9	Maximum adaptive relaxation factor
`@@QualityWeight`	Double	0.5	Reserved (not used in current implementation)
`@@SmoothingFrequency`	Integer	1	Smooth every N substeps (1 = every substep)
`@@ActiveSteps`	List	All	Simulation step IDs where ALE is active

1.5.7 State Variable Handling

Stresses and history variables are handled internally using a deferred stress correction approach (Benson, 1989; Benson, 1992): after mesh relocation, no explicit stress repair is performed; instead, the subsequent Lagrangian constitutive update restores consistency by returning the stress state to the yield surface.

1.5.8 Advanced Parameters

Parameter	Type	Default	Description
`@@SmoothBeforeAssembly`	Yes/No	Yes	Parsed for compatibility; smoothing is currently applied after coordinate updates (see Workflow)
`@@SmoothAfterConvergence`	Yes/No	No	Parsed for compatibility; not currently used
`@@UpdateGaussPointData`	Yes/No	Yes	Update Gauss point shape function caches after smoothing
`@@ConservativeFieldTransfer`	Yes/No	Yes	Parsed for compatibility; not currently used

1.5.9 Contact-Specific Parameters

Parameter	Type	Default	Description
`@@ContactZoneBuffer`	Double	0.1	Thickness of the “contact zone” around contact pairs (fraction of local characteristic element size)
`@@PreserveContactNormals`	Yes/No	Yes	Constrain contact-zone boundary motion to be tangential (prevents normal-direction motion that would change contact normals)
`@@ContactPairIds`	List	All	Optional: apply contact-zone handling only to these contact pair IDs

When `@@PreserveContactNormals: Yes`, FALCON defines a *contact zone* consisting of nodes on the selected contact pairs and nearby nodes within `@@ContactZoneBuffer`. Nodes in this zone that are not fixed are restricted to *slide tangentially* (no motion normal to the local contact boundary), improving robustness of smoothing near contact without altering contact normals.

If `@@FixContactNodes: Yes`, contact-pair nodes are fully fixed, so normal preservation applies only to other (non-fixed) boundary nodes in the contact zone.

1.5.10 Debug and Output

Parameter	Type	Default	Description
`@@WriteDebugOutput`	Yes/No	No	Enable debug file output via `@@DebugOutputPath`. Alias: `@@Debug`.
`@@TrackMeshMotion`	Yes/No	No	Parsed for compatibility; not currently used
`@@DebugOutputPath`	String	""	Path to debug output file

Debug output can also be configured from the `% Debug` section using `@ALE: /path/to/ale_debug.txt`. If provided, it overrides `@@DebugOutputPath`. When using `@@DebugOutputPath`, set `@@WriteDebugOutput: Yes`.

The debug CSV output also includes per-node flags `IsContact`, `IsContactZone`, `IsContactSlide`, and `CanSlide` to verify contact-zone behavior.

1.6 Algorithm Workflow

The ALE mesh smoothing algorithm integrates with FALCON's simulation loop as follows:

1.6.1 Quality Assessment

At the end of each converged substep (after the coordinate update in Updated Lagrangian analyses):

1. Compute quality metrics for all elements
2. Identify elements violating thresholds
3. If `QualityTriggeredOnly: Yes` and no violations, skip smoothing
4. Otherwise, proceed to smoothing

1.6.2 Smoothing Iteration

For each smoothing iteration (up to `MaxIterations`):

1. For each movable *principal* node (corner node) in the active region:
    - Compute weighted average of neighbor positions (Laplacian target)
    - Apply adaptive relaxation and displacement limits
    - Apply a local line search (step halving) and accept only if:
        - All connected elements keep positive Gauss point Jacobians after midpoint recomputation
        - Local quality penalty improves

2. Check for convergence (maximum node displacement below tolerance)

When `QualityTriggeredOnly: Yes` is enabled, `SmartLaplacian` focuses smoothing on elements that violate thresholds and their immediate neighborhood, reducing unnecessary mesh motion elsewhere.

1.6.3 Validation

After smoothing:

1. Recompute midpoint nodes (for quadratic elements)
2. Update element local coordinates
3. Validate all Gauss point Jacobians are positive
4. If validation fails:
   - Optionally apply a conservative global backoff of the overall mesh motion and re-check validity
   - If still invalid, revert to the pre-smoothing configuration
   - Signal retry to autoinc system (smaller Δt)

1.6.4 Stress Handling

If smoothing succeeds, the geometry (B-matrix) reflects the new mesh configuration and internal forces are recomputed.

Stress and internal variables are advanced using the deferred stress correction procedure (Benson, 1989; Benson, 1992); there is no user-configurable remapping/interpolation stage for this feature. For robustness and accuracy, keep mesh relocations bounded using `MaxNodeDisplacement` and `MaxCumulativeDisplacement`.

1.7 Integration with Autoinc

The ALE system is fully integrated with FALCON's automatic time increment controller:

1. **On invalid mesh detection** (mesh already invalid or smoothing cannot maintain positive Gauss Jacobians):
   - ALE signals failure to the step handler
   - Mesh is reverted to pre-smoothing state (when applicable)
   - Autoinc reduces Δt and retries the substep

ALE is primarily *preventive*: it aims to avoid invalid elements rather than “fix” already inverted meshes. If a mesh is already inverted, autoinc time step reduction (and/or remeshing) is required.

This ensures robustness: the simulation either succeeds with good mesh quality or gracefully reduces the time step rather than diverging.

1.8 Supported Element Types

ALE mesh smoothing in FALCON currently supports 2D elements:

Element Type	Nodes	Description
Linear Triangle (T3)	3	3-node triangular element
Quadratic Triangle (T6)	6	6-node triangular element with midside nodes
Linear Quadrilateral (Q4)	4	4-node quadrilateral element
Quadratic Quadrilateral (Q8)	8	8-node quadrilateral element with midside nodes

For quadratic elements, midpoint nodes are automatically recomputed after smoothing corner nodes to maintain element geometry consistency.

> **Note**: ALE mesh smoothing is currently limited to 2D elements. If `@@Enabled: Yes` is used with any 3D elements present, FALCON rejects the input (data format error) and stops.

1.9 Examples

1.9.1 Example 1: Basic Configuration for Large Deformation

```
% ALE Configuration
@@Enabled: Yes
@@Method: SmartLaplacian
@@WeightingScheme: InverseDistance
@@RelaxationFactor: 0.3
@@MaxIterations: 50
@@MinAngleThreshold: 20.0
@@MaxAngleThreshold: 140.0
@@AspectRatioThreshold: 4.0
@@JacobianThreshold: 0.1
@@FixExteriorBoundary: Yes
@@FixContactNodes: Yes
@@FixRigidNodes: Yes
@@QualityTriggeredOnly: No
@@AdaptiveRelaxation: Yes
@@MaxNodeDisplacement: 0.01
@@MaxCumulativeDisplacement: 0.1
```

```
@@SmoothingFrequency: 1
@@ActiveSteps: 1
%%%
```

Interpretation: Smart Laplacian smoothing with moderate quality thresholds and fixed exterior boundary. With `QualityTriggeredOnly: No`, smoothing runs every substep; set `QualityTriggeredOnly: Yes` to smooth only when thresholds are violated.

1.9.2 Example 2: Conservative Settings for Stiff Problems

```
% ALE Configuration
@@Enabled: Yes
@@Method: WeightedLaplacian
@@WeightingScheme: AreaBased
@@RelaxationFactor: 0.1
@@MaxIterations: 100
@@MinAngleThreshold: 10.0
@@MaxAngleThreshold: 160.0
@@AspectRatioThreshold: 6.0
@@JacobianThreshold: 0.05
@@FixExteriorBoundary: Yes
@@FixContactNodes: Yes
@@FixRigidNodes: Yes
@@QualityTriggeredOnly: No
@@AdaptiveRelaxation: Yes
@@MaxNodeDisplacement: 0.01
@@MaxCumulativeDisplacement: 0.3
@@SmoothingFrequency: 1
@@ActiveSteps: 1
%%%
```

Interpretation: More conservative settings with fixed exterior boundary, lower relaxation, and tighter displacement limits. Suitable for problems with sensitive boundary conditions.

1.9.3 Example 3: Aggressive Smoothing for Severe Deformation

```
% ALE Configuration
@@Enabled: Yes
@@Method: SmartLaplacian
@@WeightingScheme: InverseDistance
@@RelaxationFactor: 0.5
@@MaxIterations: 30
```

```
@@MinAngleThreshold: 30.0
@@MaxAngleThreshold: 120.0
@@AspectRatioThreshold: 3.0
@@JacobianThreshold: 0.2
@@FixExteriorBoundary: Yes
@@FixContactNodes: Yes
@@FixRigidNodes: Yes
@@QualityTriggeredOnly: No
@@AdaptiveRelaxation: Yes
@@MaxNodeDisplacement: 0.1
@@MaxCumulativeDisplacement: 2.0
@@SmoothingFrequency: 1
@@ActiveSteps: 1 2
%%%
```

Interpretation: Aggressive smoothing with tight quality thresholds (triggering smoothing more often), higher relaxation, and larger displacement limits. `QualityTriggeredOnly: No` means smoothing runs every substep regardless of current quality. Use for problems with very severe mesh distortion.

1.9.4 Example 4: Extreme Deformation (Deferred Stress Correction)

```
% ALE Configuration
@@Enabled: Yes
@@Method: SmartLaplacian
@@WeightingScheme: InverseDistance
@@RelaxationFactor: 0.3
@@MaxIterations: 50
@@MinAngleThreshold: 20.0
@@MaxAngleThreshold: 140.0
@@AspectRatioThreshold: 4.0
@@JacobianThreshold: 0.1
@@FixExteriorBoundary: Yes
@@FixContactNodes: Yes
@@FixRigidNodes: Yes
@@QualityTriggeredOnly: No
@@AdaptiveRelaxation: Yes
@@MaxNodeDisplacement: 0.1
@@MaxCumulativeDisplacement: 3.0
@@SmoothingFrequency: 1
@@ActiveSteps: 1
%%%
```

Interpretation: Configuration for extreme deformation scenarios with larger allowed mesh motion per call. Stress and history variables are handled internally via deferred stress correction (no interpolation/remapping keywords are required). Prefer increasing smoothing frequency and using moderate per-call limits over very large single-step relocations.

1.10 Recommendations

1.10.1 General Guidelines

1. **Start with Smart Laplacian**: The quality-aware algorithm prevents degradation of good elements while improving poor ones.
2. **Use quality-triggered smoothing**: Set `QualityTriggeredOnly: Yes` to avoid unnecessary computational overhead when the mesh is already acceptable.
3. **Choose appropriate thresholds**: Set angle and aspect ratio thresholds based on your element type and expected deformation patterns.
4. **Limit node displacement**: Use `MaxNodeDisplacement` to prevent large single-step jumps that could destabilize the solution.
5. **Fix critical boundaries**: Always use `FixContactNodes: Yes` and `FixRigidNodes: Yes` to preserve contact and rigid body interface integrity.
6. **Keep ALE motion moderate**: Use smaller `MaxNodeDisplacement` and higher `SmoothingFrequency` rather than large single-step node moves.

1.10.2 Troubleshooting

Issue	Possible Cause	Solution
Smoothing fails repeatedly	Mesh too distorted	Reduce time step, use finer mesh, or relax quality thresholds
Stress oscillations after smoothing	Large mesh motion	Reduce `RelaxationFactor` and `MaxNodeDisplacement`
Poor convergence after ALE	Excessive geometry change	Reduce `MaxCumulativeDisplacement`, enable `AdaptiveRelaxation`
Elements still inverted	Quality thresholds too loose	Tighten `MinAngleThreshold` and `JacobianThreshold`

Issue	Possible Cause	Solution
Computation too slow	Smoothing every substep	Increase `SmoothingFrequency` or use `QualityTriggeredOnly: Yes`
Stress field discontinuities	Excessive mesh motion in one call	Reduce `MaxNodeDisplacement`, increase `SmoothingFrequency`, or refine the mesh

1.10.3 When to Avoid ALE

ALE may not be necessary or appropriate for:

- **Small deformation analyses**: Standard Lagrangian formulation is sufficient
- **Problems with complex boundary conditions**: Fixed boundaries may over-constrain the mesh
- **Very stiff materials**: Mesh distortion is minimal
- **3D analyses**: Currently not supported

1.10.4 Notes on Large Mesh Motion

ALE mesh smoothing improves *mesh quality*; it is not a material advection/remap stage. If the analysis requires very large mesh relocations, prefer more frequent, smaller smoothing moves (controlled by `SmoothingFrequency`, `MaxNodeDisplacement`, and `MaxCumulativeDisplacement`) to maintain solution accuracy.

1.11 References

1. Donea, J., Huerta, A., Ponthot, J.-P., & Rodríguez-Ferran, A. (2004). Arbitrary Lagrangian-Eulerian Methods. *Encyclopedia of Computational Mechanics*, Volume 1: Fundamentals.

2. Freitag, L. A., & Knupp, P. M. (2002). Tetrahedral mesh improvement via optimization of the element condition number. *International Journal for Numerical Methods in Engineering*, 53(6), 1377-1391.

3. Field, D. A. (1988). Laplacian smoothing and Delaunay triangulations. *Communications in Applied Numerical Methods*, 4(6), 709-712.

4. Benson, D. (1989). An Efficient, Accurate, Simple ALE Method for Nonlinear Finite Element Programs. *Computer Methods in Applied Mechanics and Engineering*, 72, 305-350.

5. Benson, D. J. (1992). Computational methods in Lagrangian and Eulerian hydrocodes. *Computer Methods in Applied Mechanics and Engineering*, 99(2-3), 235-394.