AD FALCON API Manual

Explicit and IMEX Time Integration

Javad Ghorbani

March 26, 2026

Contents

1 Explicit and IMEX Time Integration

FALCON supports two non-Newton time integration modes for **Dynamic** analyses:

- **Explicit**: fully explicit **mechanical-only** update using lumped mass (no PW/PA).
- **IMEX** (implicit–explicit): explicit solid update coupled with an implicit flow solve for PW/PA (one-step θ-method; `@@Theta`/`@@FlowTheta`, θ = 1 is backward Euler).

These modes are most commonly used with **ModernAutoInc** adaptive substepping.

1.1 Overview

1.1.1 Explicit (lumped)

- Uses a **lumped mass** for displacement DOFs.
- Uses a predictor–corrector (**modified Euler / Heun**) update with an embedded local error estimate.

1.1.2 IMEX

- Advances solid (displacements) explicitly with **modified Euler / Heun**.
- Advances flow (pore pressures) implicitly with a one-step θ**-method** (`@@Theta` / `@@FlowTheta`, $\theta \in [0.5,1.0]$; default θ = 0.6; θ = 1: backward Euler).
- Solves the flow system twice per substep (**predictor/corrector**) to form a coupling-aware error estimate.

1.2 Syntax

Time integration is configured inside `% Step Definitions` (per step). Keys are written as `@@<Key>: <Value>`.

1.2.1 Example: Explicit with ModernAutoInc

```
% Step Definitions
@Step 1:
  @@StartStep: 0
  @@SimMode: Dynamic
  @@StepTime: 1.0

  @@TimeIntegration: Explicit
  @@ModernAutoInc: Yes
```

```
  @@InitialStepIncrement: 1e-4
  @@MinTimeStep: 1e-8
  @@MaxTimeStep: 1e-3
  @@ErrorTarget: 1e-3
  @@MaxRetry: 10
%%%
```

1.2.2 Example: IMEX with ModernAutoInc

```
% Step Definitions
@Step 1:
  @@StartStep: 0
  @@SimMode: Dynamic
  @@StepTime: 1.0

  @@TimeIntegration: IMEX
  @@ModernAutoInc: Yes

  @@InitialStepIncrement: 1e-4
  @@MinTimeStep: 1e-8
  @@MaxTimeStep: 1e-3
  @@ErrorTarget: 1e-3
  @@MaxRetry: 10
%%%
```

Step keys are case-insensitive; the spellings below are the canonical forms used in this manual.

1.2.3 Additional Explicit/IMEX Controls

Key	Default	Applies To	Meaning
@@Theta / @@ FlowTheta	0.6	IMEX	Flow θ-method parameter ($\theta \in [0.5, 1]$; $\theta = 1$: backward Euler).

Key	Default	Applies To	Meaning
@@MassSolve / @@ExplicitMassSolve / @@DisplacementMassSolve	LumpedDiagonal	Explicit, IMEX	How displacement acceleration $\mathbf{a}$ is obtained in the mass solve: LumpedDiagonal (default; invert diag(M)), RowSum, DiagonalConsistent (alias), Direct, CG.
@@MassSolveTol / @@ExplicitMassSolveTol	1e-10	Explicit, IMEX	CG tolerance for the displacement mass solve (used only when @@MassSolve: CG).
@@MassSolveMaxIters / @@ExplicitMassSolveMaxIters	500	Explicit, IMEX	Max CG iterations for the displacement mass solve (used only when @@MassSolve: CG).

1.3 Semi-Discrete Coupled Equations (What Is Being Time-Integrated?)

After spatial discretization (finite elements), FALCON advances a semi-discrete system in time. In a coupled dynamic (solid + pore) setting, a common abstract form is:

1.3.1 Solid momentum balance (dynamic)

Let:

- $\mathbf{u}(t)$ = displacement DOFs
- $\mathbf{v}(t) = \dot{\mathbf{u}}(t)$ = velocity DOFs
- $\mathbf{a}(t) = \ddot{\mathbf{u}}(t)$ = acceleration DOFs

A generic semi-discrete momentum balance can be written as:

$$\mathbf{M}\,\mathbf{a} + \mathbf{C}_{uu}\,\mathbf{v} + \mathbf{f}_{\text{int}}(\mathbf{u}, \ldots) + \mathbf{f}_{\text{c}}(\mathbf{u}, \mathbf{v}, \ldots) = \mathbf{f}_{\text{ext}}(t) \tag{1}$$

Where:

- $\mathbf{M}$ is the (consistent) mass matrix; Explicit/IMEX use a **lumped diagonal** version in the

update.

- $\mathbf{C}_{uu}$ represents damping/coupling terms that depend on velocity.
- $\mathbf{f}_{\text{int}}$ is the internal force vector (from stress integration). For IMEX, this may also include flow-related internal contributions.
- $\mathbf{f}_{\text{c}}$ is the contact (and other constraint) force contribution.
- $\mathbf{f}_{\text{ext}}$ is the assembled external force vector at the current time.

1.3.2 Flow (pore pressure) evolution

FALCON represents pore pressures as an **initial (baseline) field + excess (incremental) field**:

$$p_w^{\text{total}} = p_w^0 + p_w^{\text{excess}}, \qquad p_a^{\text{total}} = p_a^0 + p_a^{\text{excess}} \tag{2a}$$

The `% Initial Assignments` block (`@PW:` / `@PA:`) sets the baseline fields p^0 (stored as `InitialPoreWaterPressure` / `InitialPoreAirPressure`). The solver evolves the excess fields p^{excess}. FALCON does not infer baseline pore pressures from the current/excess field.

1.3.3 Boundary conditions and output conventions (important)

- The `PW` / `PA` DOFs represent **excess** pore pressures (p^{excess}), not total pressure.
- A drained boundary is typically specified as `PW = 0` (excess), while the total pore pressure at that boundary is still $p^{\text{total}} = p^0 + 0$.
- In GiD output:
  - `PW` / `PA` are written as **total** pressures ($p^0 + p^{\text{excess}}$).
  - `EXPW` is written as **excess** pore water pressure (p^{excess}).

Let $\mathbf{p}(t)$ denote pore-pressure DOFs (e.g. PW, PA). A typical semi-discrete flow equation can be written as:

$$\mathbf{C}_{pp}\,\dot{\mathbf{p}} + \mathbf{K}_{pp}\,\mathbf{p} = \mathbf{f}_p(t) - \mathbf{C}_{pu}\,\mathbf{v} \tag{2}$$

Where:

- $\mathbf{C}_{pp}$ is a "capacity / compressibility" (time) operator.
- $\mathbf{K}_{pp}$ is a conductivity / diffusion operator.
- $\mathbf{C}_{pu}\,\mathbf{v}$ represents the coupling of the solid rate into the flow equation.

In FALCON, IMEX treats (2) implicitly with a one-step θ-method (θ = 1 recovers backward Euler) for stability. Explicit does not support pore pressure DOFs (PW/PA).

1.4 Explicit Time Integration (Modified Euler / Heun)

Explicit advances the **solid only** with an explicit predictor–corrector integrator.

Define a substep size Δt. Consider the state at time t_n:

- $\mathbf{u}^n$, $\mathbf{v}^n$

1.4.1 Mass lumping

Explicit uses a diagonal "lumped" mass matrix for efficient element-wise updates:

- $\mathbf{M}_\ell = \mathrm{diag}(m_i)$ (lumped mass for displacement DOFs)

  This makes the explicit acceleration updates component-wise.

1.4.2 Solid predictor (Euler / symplectic-style)

Compute an acceleration estimate from the current-time force balance (schematically):

$$\mathbf{a}^n = \mathbf{M}_\ell^{-1}\Big(\mathbf{r}_u(\mathbf{u}^n, \mathbf{v}^n, t_n)\Big) \tag{3}$$

Then form the predictor:

$$\mathbf{v}^{\text{pred}} = \mathbf{v}^n + \Delta t\, \mathbf{a}^n \tag{4}$$

$$\mathbf{u}^{\text{pred}} = \mathbf{u}^n + \Delta t\, \mathbf{v}^{\text{pred}} \tag{5}$$

1.4.3 Pore pressures with Explicit

Explicit is not supported when PW/PA DOFs are present. Use IMEX (recommended) or Implicit.

1.4.4 Corrector (Heun)

Evaluate forces again at the predictor state to obtain $\mathbf{a}^{\text{pred}}$.

Then Heun (trapezoidal) correction for the solid is:

$$\mathbf{v}^{n+1} = \mathbf{v}^n + \frac{\Delta t}{2}\Big(\mathbf{a}^n + \mathbf{a}^{\text{pred}}\Big) \tag{8}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \frac{\Delta t}{2}\Big(\mathbf{v}^n + \mathbf{v}^{\text{pred}}\Big) \tag{9}$$

For pore pressures, use IMEX Section 4.

1.4.5 What the "error estimate" means

Modified Euler / Heun naturally provides an embedded estimate of the local truncation error by comparing predictor vs. corrector states:

- $\Delta\mathbf{u}_{\text{est}} = \mathbf{u}^{n+1} - \mathbf{u}^{\text{pred}}$
- $\Delta\mathbf{v}_{\text{est}} = \mathbf{v}^{n+1} - \mathbf{v}^{\text{pred}}$

FALCON converts these differences into a **dimensionless** scalar `err_est` used by ModernAutoInc (see Section 6).

1.5 IMEX Time Integration (Explicit Solid + Implicit Flow)

IMEX uses the same explicit predictor–corrector update for the **solid**, but treats the **flow equation** implicitly with a one-step θ-method (θ = 1 is backward Euler).

1.5.1 Solid predictor/corrector (same idea as Explicit)

IMEX advances $\mathbf{u}$ and $\mathbf{v}$ with the same Heun structure as in (4)–(9).

1.5.2 θ-method for flow (Backward Euler is θ = 1)

Applying a one-step θ-method to (2) (using a chosen solid rate $\mathbf{v}^\star$ and $\theta \in [0.5, 1]$) yields:

$$\mathbf{C}_{pp}\,\frac{\mathbf{p}^{n+1}-\mathbf{p}^{n}}{\Delta t} + \theta\,\mathbf{K}_{pp}\,\mathbf{p}^{n+1} + (1-\theta)\,\mathbf{K}_{pp}\,\mathbf{p}^{n} = \mathbf{f}_p(t_{n+1}) - \theta\,\mathbf{C}_{pu}\,\mathbf{v}^\star - (1-\theta)\,\mathbf{C}_{pu}\,\mathbf{v}^{n} \tag{11}$$

Rearranging:

$$\left(\theta\,\mathbf{K}_{pp} + \frac{1}{\Delta t}\mathbf{C}_{pp}\right)\mathbf{p}^{n+1} = \mathbf{f}_p(t_{n+1}) - \theta\,\mathbf{C}_{pu}\,\mathbf{v}^\star - (1-\theta)\,\mathbf{C}_{pu}\,\mathbf{v}^{n} + \left(\frac{1}{\Delta t}\mathbf{C}_{pp} - (1-\theta)\mathbf{K}_{pp}\right)\mathbf{p}^{n} \tag{12}$$

This is a linear system for $\mathbf{p}^{n+1}$ with operator:

$$\mathbf{A}(\Delta t) = \theta\,\mathbf{K}_{pp} + \frac{1}{\Delta t}\mathbf{C}_{pp} \tag{13}$$

1.5.3 Predictor/corrector flow solves (coupling-aware)

To remain consistent with the Heun error estimate, IMEX performs two flow solves using two different solid-rate inputs:

- **Predictor flow solve:** use $\mathbf{v}^\star = \mathbf{v}^{\text{pred}}$ to compute $\mathbf{p}^{\text{pred}}$.
- **Corrector flow solve:** use $\mathbf{v}^\star = \mathbf{v}^{n+1}$ to compute $\mathbf{p}^{n+1}$.

The difference $\mathbf{p}^{n+1} - \mathbf{p}^{\text{pred}}$ contributes to the overall embedded error estimate.

1.6 Requirements and Limitations

Item	Explicit	IMEX
Supported `@@SimMode:`	`Dynamic` only	`Dynamic` only
Uses Newton iterations	No	No (flow uses a linear solve)

Item	Explicit	IMEX
Works with @@ModernAutoInc: Yes	Recommended	Recommended
Works with @@ModernAutoInc: No (fixed increments)	Supported (fixed Δt; no rejection by err_est)	Supported (fixed Δt; no rejection by err_est)
Coupled / fully-coupled (PW/PA present)	Not supported (throws)	Supported
Contact (Penalty formulation)	Supported (assembled each substep)	Supported (assembled each substep)
Contact (Augmented Lagrangian)	Not supported (throws)	Not supported (throws)

Notes:

- If @@SimMode is not Dynamic, FALCON aborts with an error.
- In fixed-increment mode (@@ModernAutoInc: No), Explicit/IMEX still compute an embedded error estimate, but do not use it to accept/reject substeps.
- SeepageFace active-set outer iterations are disabled in Dynamic mode.
- ContactFormulation=AugmentedLagrangian is not supported with Explicit/IMEX; FALCON aborts to avoid incorrect results.
- **Static/Consolidation:** Explicit/IMEX are not supported because they rely on dynamic state (lumped mass update for the solid). Use @@TimeIntegration: Implicit for quasi-static/static and consolidation steps with large Δt.
- **Matrix assembly (implementation):**
  - Explicit assembles M, C, and C21 (plus contact/internal forces), but does **not** assemble or solve with K for the solid. Stresses/strains are still computed to form F_internal.
  - IMEX also assembles K because the flow solve uses an operator of the form A = θ K_pp + (C_pp/Δt); it still does not solve a global Newton system for the solid.

1.7 Adaptive Substepping Metric (@@ErrorTarget) for Explicit/IMEX

For Explicit and IMEX, @@ErrorTarget is treated as a **dimensionless local time-integration error target** (not a Newton residual norm).

1.7.1 Definition of err_est

For each substep, Explicit/IMEX computes a **predictor** state (Euler) and a **corrector** state (Heun). The embedded error is based on the predictor–corrector difference:

- $\Delta \mathbf{u} = \mathbf{u}^{\text{corr}} - \mathbf{u}^{\text{pred}}$
- $\Delta \mathbf{v} = \mathbf{v}^{\text{corr}} - \mathbf{v}^{\text{pred}}$

FALCON reduces these vectors into an infinity-norm error indicator over DOF groups (displacements vs. pore pressures, and their rates). For IMEX, the pore-pressure part is included; for Explicit it is absent.

Each group uses a mixed absolute/relative scaling of the form

$$\|\Delta x\|_\infty \le \text{atol} + \text{rtol}\, s,$$

where:

- `rtol` is the current adaptive target (starts from `@@ErrorTarget`).
- `atol` is an absolute floor used to keep denominators nonzero (internally `max(@@Safety MinDenominator, 1e-30)`).
- `s` is a scale based on the current magnitude of the group (defined below).

In terms of the scalar that is printed, FALCON reports

$$\text{err\_est} = \max_g \left(\text{rtol}\, \frac{\|\Delta x_g\|_\infty}{\text{atol} + \text{rtol}\, s_g} \right), \qquad s_g = \max(\|x_g^n\|_\infty, \|x_g^{\text{corr}}\|_\infty, 10^{-30}).$$

with groups g spanning displacement/velocity (and, for IMEX, pore pressure and pore rate).

Here g indexes DOF groups; x_g means "take only the components of x that belong to group g" before computing $\|\cdot\|_\infty$. Concretely:

- $g = u_{\text{disp}}$: displacement DOFs (DisX/DisY/DisZ)
- $g = u_{\text{pore}}$: pore-pressure DOFs (PW/PA; IMEX only)
- $g = v_{\text{disp}}$: velocity DOFs (time-derivative of Dis*)
- $g = v_{\text{pore}}$: pore-pressure rates (time-derivative of PW/PA; IMEX only)

When `@@ModernAutoInc: Yes`, a substep is accepted when `err_est <= adaptive Target` (where `adaptiveTarget` starts from `@@ErrorTarget`). In fixed-increment mode (`@@ ModernAutoInc: No`), substeps are not rejected based on `err_est` (it is still computed and reported).

Equivalently: the predictor–corrector jump must be below the allowed tolerance band (absolute floor `atol` plus a relative part `rtol*|x|`).

1.7.2 How this interacts with ModernAutoInc

When `@@ModernAutoInc: Yes`, rejected substeps are retried with a reduced Δt, and successful substeps may increase Δt cautiously.

See Automatic time incrementation for the controller logic and all tuning parameters.

1.8 Optional Stability Guard (Forcing Δt Reduction)

Even if the embedded error estimate is acceptable, an Explicit/IMEX substep can be marked as **failed** to force Δt reduction when solution rates/jumps are too large.

Configure in `% Debug`:

```
% Debug
@SolverEcho: Yes
@SolverEchoEvery: 1
@SolverEchoLevel: 2

# Explicit/IMEX stability guard (set <0 to disable)
@SolverStabilityMaxPdot: 1.0e3   # max ||p_dot||inf
@SolverStabilityMaxDP:   1.0e2   # max ||Δp||inf per substep
@SolverStabilityMaxV:    1.0e1   # max ||v||inf
%%%
```

If any enabled limit is exceeded (or non-finite values appear), the substep is rejected and retried with a smaller Δt.

1.9 Practical Stability Notes (What Sets the "Safe" Δt?)

1.9.1 Solid (wave) stability in explicit dynamics

For linear elastodynamics, explicit schemes have a stability limit linked to the highest natural frequency $\omega_{\max}$:

$$\Delta t \lesssim \frac{2}{\omega_{\max}} \tag{17}$$

In practice this is often expressed as a CFL-type condition based on wave speed and element size:

$$\Delta t \lesssim C\,\frac{h}{c} \tag{18}$$

Where c is the fastest wave speed in the mesh and h is a characteristic element dimension.

1.9.2 Flow (diffusion) stability

For diffusion-like flow, explicit time integration would have a stability restriction roughly of the form $\Delta t \lesssim C\,h^2/\kappa$, where h is an element length scale and κ relates to conductivity/permeability. This can force extremely small Δt in fine meshes or high-permeability regimes. IMEX avoids this by treating the flow equation implicitly with a θ-method ($\theta \geq 0.5$), which is unconditionally stable for linear diffusion.

1.10 Diagnostics

- `@SolverEcho` prints a per-substep summary (including the Explicit/IMEX error metric).
- `@SolverDebug` can write a CSV-style log per substep.

1.10.1 Console Output (IMEX)

When running `@@TimeIntegration: IMEX`, FALCON prints a per-substep progress line to stdout. In that line:

- `fnorm` is labeled as `(err_est)` for IMEX (the embedded predictor/corrector error indicator used by the adaptive controller).
- `p_res_inf` and `p_res_2` are the infinity- and 2-norms of the linear-system residual `r = A p - b` for the accepted (corrector) pressure solve; they are `-1` when no `PW`/`PA` DOFs exist.
- `p_ndof` is the number of active pressure/suction DOFs (`PW`/`PA`) participating in the IMEX implicit θ-method pressure solve (θ = 1 is backward Euler).