EPFL

CIVIL 449: Nonlinear Analysis of Structures

School of Architecture, Civil & Environmental Engineering
Civil Engineering Institute

Iterative Techniques for Nonlinear Analysis

Prof. Dr. Dimitrios Lignos, Dr. Diego Heredia
EPFL, ENAC, IIC, RESSLab

Objectives of today's lecture

- To introduce:
  - Iterative techniques to solve systems of nonlinear equations
  - Incremental approach to equilibrium
  - Load-displacement constraint methods:
    - Load control
    - Displacement control
    - Arc-length control

Motivation

- Finite element equilibrium equation for static (linear) analysis:

$$\mathbf{F}_{ext} = \mathbf{K}_{structure} \cdot \mathbf{v}$$

- When considering geometric (and material) nonlinearities, the stiffness $\mathbf{K}$ becomes a function of the displacements $\mathbf{v}$ of the structure. When considering nonlinearities, the equilibrium equation for static analysis is written,

$$\mathbf{F}_{ext} = \mathbf{K}_{structure}(\mathbf{v}) \cdot \mathbf{v}$$

- We need an iterative solution procedure to solve the equilibrium equations

- The imposed external forces and/or displacement are applied incrementally

Nonlinear equilibrium equations

- The basic problem in a general nonlinear analysis is to find the state of equilibrium of a body corresponding to the applied loads. If the externally loads are applied incrementally, the equilibrium condition of a system can be formulated as follows:

$$\mathbf{F}_{unb}^{n} = \mathbf{F}_{unb}(\mathbf{v}^{n}) = \mathbf{F}_{int}(\mathbf{v}^{n}) - \mathbf{F}_{ext}^{n} = \mathbf{0}$$

Where the superscript n referrers to the load step and $\mathbf{F}_{unb}^{n}$ denotes the unbalanced load vector

- The response calculations are performed incrementally by dividing the total applied load or displacement into several increments

- In an incremental solution, it is assumed that the solution for the step $n-1$ is known and that the solution for the step n is required

- This equilibrium condition can be solved using different iterative solution techniques

Newton-Raphson scheme

- The most frequently used scheme for the solution of nonlinear equations is the Newton-Raphson iteration. The equilibrium condition of the system at the step n is given by

$$\mathbf{F}_{unb}^{n} = \mathbf{F}_{unb}(\mathbf{v}^{n}) = \mathbf{F}_{int}(\mathbf{v}^{n}) - \mathbf{F}_{ext}^{n} = \mathbf{0}$$

- Assume that in the iterative solution, the quantity $\mathbf{v}^{n,i-1}$is known. The superscript i is the iteration counter for the Newton-Raphson scheme, and is initialized with $\mathbf{v}^{n,1} = \mathbf{v}^{n-1}$, where $\mathbf{v}^{n-1}$ denotes the previous converged step

- Using a Taylor series expansion and neglecting the higher-order terms gives

$$\mathbf{F}_{unb}^{n}(\mathbf{v}^{n}) \approx \mathbf{F}_{unb}^{n}\left(\mathbf{v}^{n,i-1}\right) + \left.\frac{\partial \mathbf{F}_{unb}}{\partial \mathbf{v}}\right|_{\mathbf{v}^{n,i-1}} \cdot \left(\mathbf{v}^{n} - \mathbf{v}^{n,i-1}\right) = \mathbf{0}$$

With

$$\left.\frac{\partial \mathbf{F}_{unb}}{\partial \mathbf{v}}\right|_{\mathbf{v}^{n,i-1}} = \left.\frac{\partial \mathbf{F}_{int}}{\partial \mathbf{v}}\right|_{\mathbf{v}^{n,i-1}} \text{ and } \mathbf{F}_{unb}\left(\mathbf{v}^{n,i-1}\right) = \mathbf{F}_{int}^{n,i-1} - \mathbf{F}_{ext}^{n}$$

Here, the external loads are assumed to be deformation independent

Newton-Raphson scheme (2)

- The Taylor series expansion can be rewritten as

$$\left.\frac{\partial \mathbf{F}^{int}}{\partial \mathbf{v}}\right|_{\mathbf{v}^{n,i-1}} \cdot \left(\mathbf{v}^{n} - \mathbf{v}^{n,i-1}\right) = \mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i-1}$$

- Therefore, the increment in displacement $\Delta\mathbf{v}^{i} = \mathbf{v}^{n} - \mathbf{v}^{n,i-1}$ can be computed by solving

$$\mathbf{K}_{structure}^{n,i-1}\Delta\mathbf{v}^{i} = \mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i-1} \quad (1)$$

Where $\mathbf{K}_{structure}^{n+1,i-1}$ is the current tangent stiffness matrix of the structure

$$\mathbf{K}_{structure}^{n,i-1} = \left.\frac{\partial \mathbf{F}^{int}}{\partial \mathbf{v}}\right|_{\mathbf{v}^{n,i-1}}$$

- Equation (1) can be solved using various classic approaches from linear algebra to handle systems of equations of the form $\boldsymbol{Ax} = \boldsymbol{b}$ (Gaussian elimination, LU decomposition, Cholesky decomposition, iterative methods like the Jacobi, Gauss-Seidel, Conjugate Gradient methods)

Newton-Raphson scheme (3)

- Finally, the displacement at iteration i is given by

$$\mathbf{v}^{n,i} = \mathbf{v}^{n,i-1} + \Delta\mathbf{v}^{i}$$

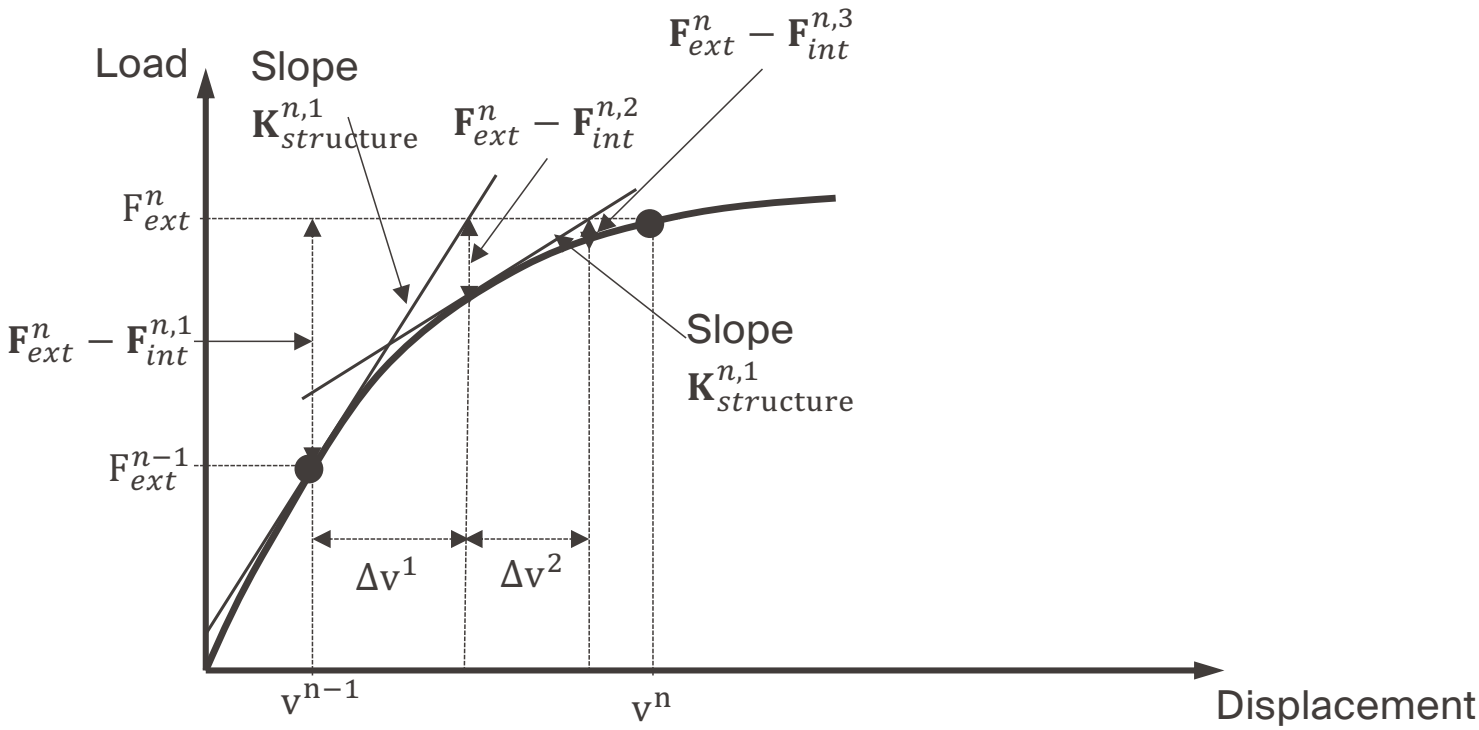
- At the first iteration of the Newton-Raphson iterative procedure, the following quantities are initialized:

$$\mathbf{K}^{n,1}_{structure} = \mathbf{K}^{n-1}_{structure},\ \mathbf{F}^{n,1}_{int} = \mathbf{F}^{n-1}_{int} \text{ and } \mathbf{v}^{n,1} = \mathbf{v}^{n-1}$$

- The iterations are carried out until appropriate convergence criteria (discussed later) are satisfied

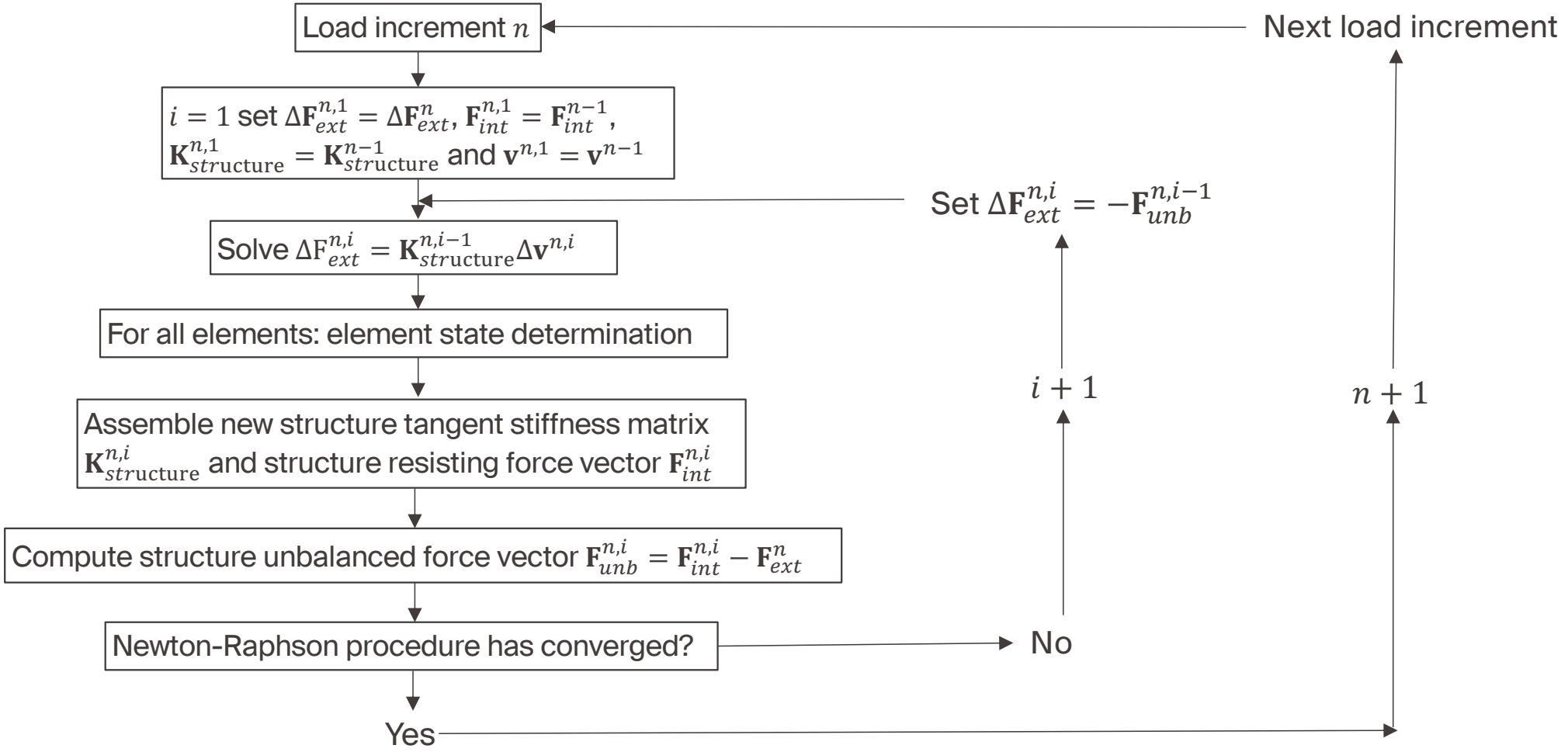
Newton-Raphson scheme (4)

- for a single degree of freedom system:



Newton-Raphson scheme (5)

- The following flow chart summarizes the process of the solution for the structure state determination



Newton-Raphson scheme

- If the current solution iterate $\left(\mathbf{v}^{n,i}, \mathbf{F}_{ext}^{n,i}\right)$ is sufficiently close to the solution $(\mathbf{v}^{n}, \mathbf{F}_{ext}^{n})$ and if the consistent tangent stiffness matrix does not change abruptly, the convergence of the Newton-Raphson scheme **is quadratic**

- Quadratic convergence is guaranteed if the exact consistent tangent stiffness matrix is used. This requires that

$$\mathbf{K}_{structure}^{n,i-1} = \left.\frac{\partial \mathbf{F}^{int}}{\partial \mathbf{v}}\right|_{\mathbf{v}^{n,i-1}}$$

- If the current solution iterate $\left(\mathbf{v}^{n,i}, \mathbf{F}_{ext}^{n,i}\right)$ is not sufficiently close to the solution $(\mathbf{v}^{n}, \mathbf{F}_{ext}^{n})$ and/or if the tangent stiffness matrix is not consistent and/or changes abruptly, the Newton-Raphson scheme may diverge

- In an effective finite element program, the exact tangent stiffness matrix will be used, if possible; hence, the primary procedure for reaching convergence (if convergence difficulties are encountered) is to **decrease the magnitude of the load step**

Modified Newton scheme

- In the Newton-Raphson iteration, the major computational cost per iteration is the calculation and factorization of the tangent stiffness matrix. Since these calculations can be expensive for large-order systems, the use of a modification of the full Newton-Raphson algorithm can be effective

- One modification could be to use the initial stiffness matrix $\mathbf{K}^{1}_{structure}$ to solve

$$\mathbf{K}^{1}_{structure}\Delta\mathbf{v}^{i} = \mathbf{F}^{n}_{ext} - \mathbf{F}^{n,i-1}_{int}$$

In this case, only the initial stiffness matrix $\mathbf{K}^{1}_{structure}$ needs to be computed and factorized, thus avoiding the computational cost of recomputing and factorizing the structure stiffness matrix

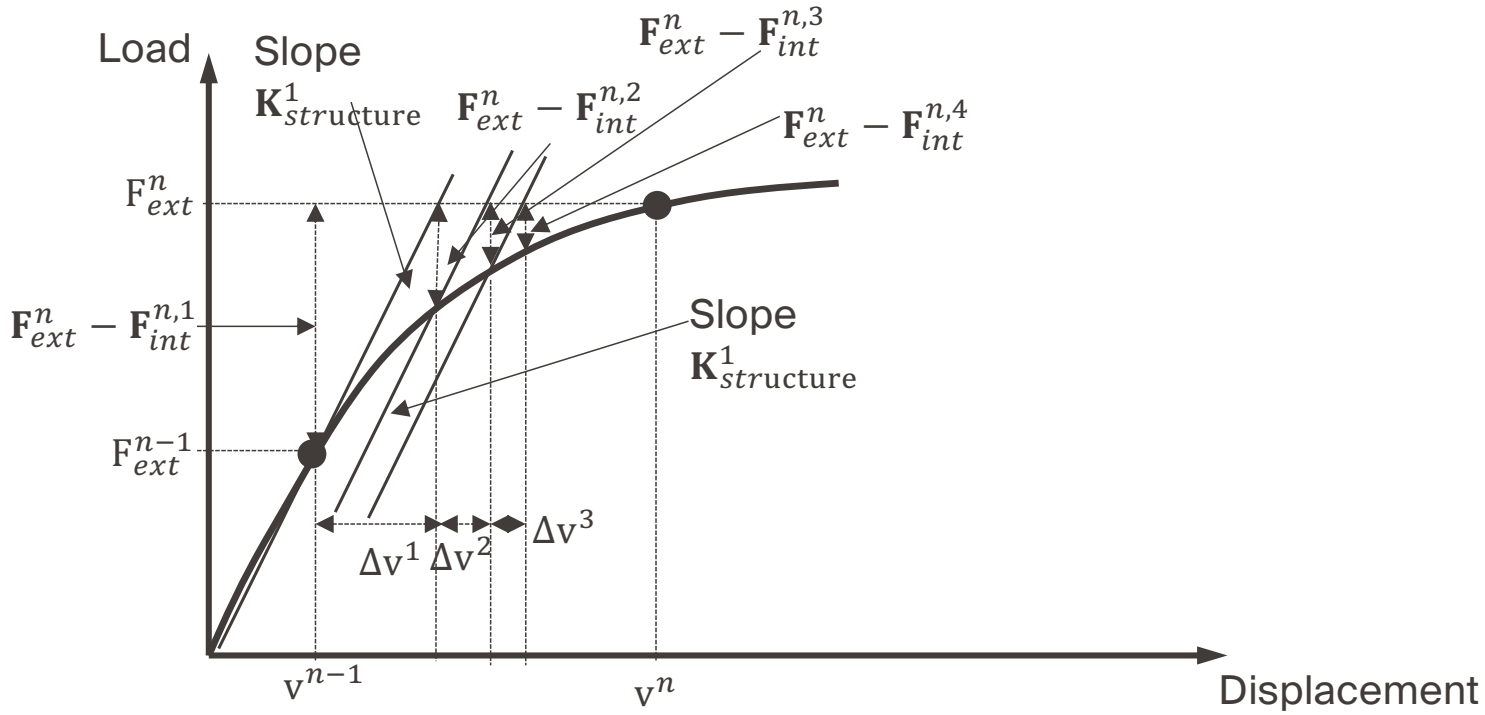
- Another approach could be to use a stiffness matrix $\mathbf{K}^{n^*}_{structure}$ to solve

$$\mathbf{K}^{n^*}_{structure}\Delta\mathbf{v}^{i} = \mathbf{F}^{n}_{ext} - \mathbf{F}^{n,i-1}_{int}$$

Where $\mathbf{K}^{n^*}_{structure}$ is updated only at several converged steps n^* (for example $n^* = 1,5,10,20$ ...)

Modified Newton scheme (2)

- The following figure illustrates the process of the solution when used for a single degree of freedom system. In this example the initial structure tangent stiffness matrix $\mathbf{K}^1_{structure}$ is used to iterate



Example: Newton schemes

- For a single degree of freedom system, consider the following load step $n = 2$, with:

$$F_{ext}^{n=2} = 10,\ F_{int}^{n} = 4 + 2\sqrt{v^n} \text{ and } v^{n=1} = 1$$

Compare the Newton-Raphson and the modified Newton with initial tangent to compute $v^{n=2}$

- At each iteration i of the Newton schemes, the tangent stiffness matrix is given by

$$K_{structure}^{n,i} = \frac{\partial F_{int}^{n,i}}{\partial v^{n,i}} = \frac{1}{\sqrt{v^{n,i}}}$$

- The iterations of the Newton schemes are obtained by solving

$$K_{structure}^{n,i-1}\Delta v^i = F_{ext}^{n} - F_{int}^{n,i-1} \rightarrow \frac{1}{\sqrt{v^{n,i-1}}}\Delta v^i = 10 - 4 - 2\sqrt{v^{n,i-1}}$$

And

$$v^{n,i} = v^{n,i-1} + \Delta v^i$$

And using $K_{structure}^{n,i-1} = \frac{1}{\sqrt{v^{n,i-1}}}$ for the Newton-Raphson iterations and $K_{structure}^{n,i-1} = K_{structure}^{1} = \frac{1}{\sqrt{v^{n,i}}} = 1$ for the modified Newton with initial tangent

Example: Newton schemes (2)

- The following quantities are initialized for the first iteration of the Newton schemes:

$$v^{2,1} = v^1 = 1,\ K^{2,1}_{structure} = 1 \text{ and } F^{2,1}_{int} = 4 + 2\sqrt{v^{2,1}} = 6$$

- The table summarizes the iterations of both iterative schemes

	Newton-Raphson			**Modified Newton (Initial tangent)**		
	$K^{2,i-1}_{structure}$	$F^2_{ext} - F^{2,i-1}_{int}$	$v^{2,i}$	$K^{2,i-1}_{structure}$	$F^2_{ext} - F^{2,i-1}_{int}$	$v^{2,i}$
$i = 2$	1.00	4	5	1.00	4	5
$i = 3$	0.45	1.53	8.42	1.00	1.53	6.53
$i = 4$	0.34	0.20	8.99	1.00	0.89	7.42
$i = 5$	0.33	0.00	9.00	1.00	0.55	7.97
$i = 6$	0.33	0.00	9.00	1.00	0.35	8.32
⋮						
$i = 19$	0.33	0.00	9.00	1.00	0.00	9.00

Example: Newton schemes (3)

- The Newton-Raphson scheme converges in 4 iterations
- The modified Newton scheme with the initial tangent converges in 18 iterations
- Slower convergence rate; no need to compute the consistent tangent stiffness at every iteration

The Broyden-Fletcher-Goldfarb-Shanno method

- The BFGS (Broyden-Fletcher-Goldfarb-Shanno) method is a particular instance of quasi-Newton methods (known as matrix update methods). These involve updating the coefficient matrix (i.e., its inverse) to provide a secant approximation to the stiffness matrix from iteration $i - 1$ to i

- **Step 1:** Evaluate a displacement vector increment

$$\Delta\bar{\mathbf{v}} = \left(\mathbf{K}_{structure}^{n,i-1}\right)^{-1}\left(\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i-1}\right)$$

This displacement vector defines a 'direction' for the actual displacement vector

- **Step 2:** Perform a line search in the direction $\Delta\bar{\mathbf{v}}$ to satisfy 'equilibrium' in this direction. For this line search, the following displacement vector is evaluated:

$$\mathbf{v}^{n,i} = \mathbf{v}^{n,i-1} + \beta\Delta\bar{\mathbf{v}}$$

Where β is a scalar multiplier determined by the line search. The unbalanced loads corresponding to this displacement vector $\left(\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i}\right)$ are computed

The Broyden-Fletcher-Goldfarb-Shanno method (2)

- The parameter β is varied until the unbalanced loads in the direction $\Delta\bar{\mathbf{v}}$, as defined by the inner product $\Delta\bar{\mathbf{v}}\left(\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i}\right)$ is small. This condition is satisfied when, for a convergence tolerance tol, the following equation is satisfied:

$$\Delta\bar{\mathbf{v}}\left(\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i}\right) \leq tol \cdot \Delta\bar{\mathbf{v}}\left(\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i-1}\right)$$

The final value of $\mathbf{v}^{n,i}$ is obtained for the β for which this condition is satisfied. The following quantities are then computed:

$$\boldsymbol{\delta}^{i} = \mathbf{v}^{n,i} - \mathbf{v}^{n,i-1} \text{ and } \boldsymbol{\gamma}^{i} = \mathbf{F}_{int}^{n,i} - \mathbf{F}_{int}^{n,i-1}$$

The variation of the parameter β (i.e. the line search) can be performed using different approaches such as the bisection method. This step is not mandatory and β can be set to 1 and no line search is performed

- **Step 3**: The coefficient matrix is updated using

$$\left(\mathbf{K}_{structure}^{n,i}\right)^{-1} = \left(\mathbf{A}^{\mathrm{i}}\right)^{\mathrm{T}}\left(\mathbf{K}_{structure}^{n,i-1}\right)^{-1}\mathbf{A}^{i}$$

With

$$\mathbf{A}^{i} = \mathbf{I} + \mathbf{v}^{\mathrm{i}}\left(\mathbf{w}^{\mathrm{i}}\right)^{\mathrm{T}}$$

The Broyden-Fletcher-Goldfarb-Shanno method (3)

- The vectors $\mathbf{v}^{\mathrm{i}}$ and $\mathbf{w}^{\mathrm{i}}$ are computed from the known nodal point forces and displacements using

$$\mathbf{v}^{\mathrm{i}} = -\left(\frac{\left(\boldsymbol{\delta}^{i}\right)^{T}\boldsymbol{\gamma}^{i}}{(\boldsymbol{\delta}^{i})^{T}\mathbf{K}_{structure}^{n,i-1}\boldsymbol{\delta}^{i}}\right)\mathbf{K}_{structure}^{n,i-1}\boldsymbol{\delta}^{i} - \boldsymbol{\gamma}^{i}$$

And

$$\mathbf{w}^{i} = \frac{\boldsymbol{\delta}^{i}}{(\boldsymbol{\delta}^{i})^{T}\boldsymbol{\gamma}^{i}}$$

- The line search approach presented for the BFGS method could also be used in the Newton-Raphson and modified Newton approaches

- With the line search performed within an iteration i, the expense of the iteration increases, but fewer iterations may be needed for convergence. Also, the line search may prevent divergence of the iterations, and in practice, this increased robustness is the major reason why a line search can be effective

Example: The BFGS Method

- Let's consider the previous example: for a single degree of freedom system, assume the following load step $n = 2$, with:

$$F_{ext}^{n=2} = 10,\ F_{int}^{n} = 4 + 2\sqrt{v^n} \text{ and } v^{n=1} = 1$$

Perform the iteration of the BFGS method to compute $v^{n=2}$. Omit the line searches in the solution (i.e., use $\beta = 1$)

- The iterations of the BFGS scheme are obtained by solving

$$v^{n,i} = v^{n,i-1} + \Delta\bar{v}$$

With

$$\Delta\bar{v} = \left(K_{structure}^{n,i-1}\right)^{-1}\left(F_{ext}^{n} - F_{int}^{n,i-1}\right) = \left(\frac{1}{\sqrt{v^{n,i-1}}}\right)^{-1}\left(10 - 4 - 2\sqrt{v^{n,i-1}}\right)$$

Example: The BFGS Method

- The following quantities are initialized for the first iteration of the BFGS scheme:

$$v^{2,1} = v^1 = 1,\ K^{2,1}_{structure} = 1 \text{ and } F^{2,1}_{int} = 4 + 2\sqrt{v^{2,1}} = 6$$

- The table below summarizes the iterations of the BFGS scheme:

i	$v^{2,i-1}$	$\Delta\bar{v} = \delta^i$	$v^{2,i}$	γ^i	$\left(K^{2,i}_{structure}\right)^{-1}$
2	1.00	4.00	5.00	2.47	1.62
3	5.00	2.47	7.47	0.99	2.48
4	7.47	1.32	8.80	0.45	2.85
5	8.80	0.19	8.99	0.06	2.98
6	8.99	0.01	9.00	0.00	3.00

- Convergence is achieved after 5 iterations

Convergence criteria

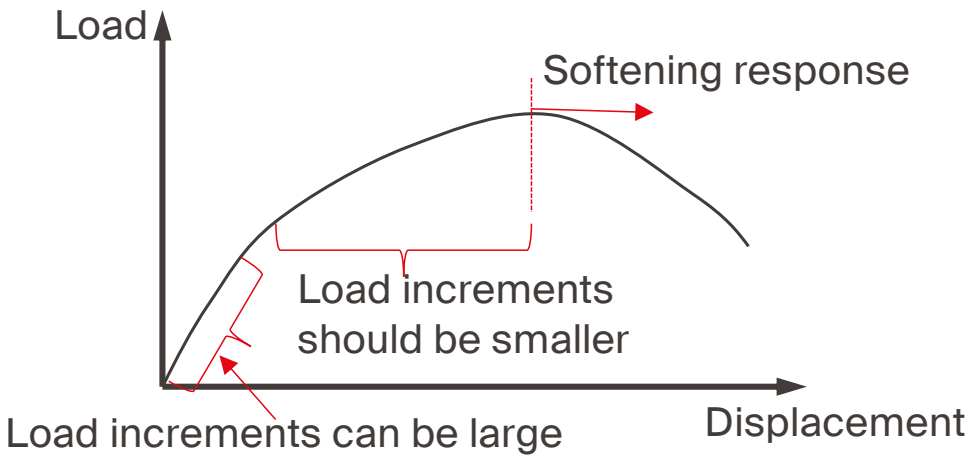
- The iterative schemes are performed until a convergence criterion is satisfied. Typically, this criterion is satisfied when a certain quantity g reaches a predefined threshold. Mathematically,

$$g \leq tol$$

Where tol is an arbitrary tolerance value

- The following quantities are typically used to assess convergence at an iteration i:
  - The norm of the unbalanced force: $\left\|\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i}\right\| \leq tol$
  - The relative norm of the unbalanced force: $\frac{\left\|\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i}\right\|}{\mathbf{F}_{ext}^{n}} \leq tol$
  - The norm of the displacement increment: $\left\|\Delta\mathbf{v}^{n,i}\right\| \leq tol$
  - The relative norm of the displacement increment: $\frac{\left\|\Delta\mathbf{v}^{n,i}\right\|}{\left\|\mathbf{v}^{n-1}\right\|} \leq tol$
  - The energy increment: $\left(\Delta\mathbf{v}^{n,i}\right)^{T}\left(\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i}\right) \leq tol$
  - The relative energy increment: $\frac{\left(\Delta\mathbf{v}^{n,i}\right)^{T}\left(\mathbf{F}_{ext}^{n} - \mathbf{F}_{int}^{n,i}\right)}{\left(\mathbf{v}^{n-1}\right)^{T}\left(\mathbf{F}_{ext}^{n}\right)} \leq tol$
  - Fixed number of iterations: A predetermined number of iterations are performed, after which the final state is assumed to be converged. **This approach does not guarantee convergence**

Load-Displacement-Constraint methods



- To compute the load-displacement response, initially relatively large load increments can be used, but as the peak load is approached, the load increments shall be smaller
- At the peak point, the stiffness matrix is singular (i.e., the slope of the load-displacement curve is zero)
- In the softening path of the load-displacement curve, a special solution procedure is required which allows for a decrease in load. An increase in displacement shall be used in this case

Load-Displacement-Constraint methods (2)

- For this purpose, a load-displacement-constraint method can be used
- The idea is to introduce a load multiplier that increases (or decreases) the intensity of the applied loads, to obtain fast convergence per load step, to traverse the peak point and evaluate the softening response
- A basic assumption is that the load vector varies proportionally during the response calculation. With this assumption, the governing equation at a load step n is given by

$$\mathbf{F}_{int}^{n} - \lambda^{n}\bar{\mathbf{F}}_{ext} = \mathbf{0}$$

Where,

- λ^{n} is a scalar load multiplier, unknown (to be determined);
- $\bar{\mathbf{F}}_{ext}$ is the reference load vector for the $nDOFs$ degrees of freedom of the finite element model;
- $\mathbf{F}_{int}^{n}$ is the usual structure resisting force vector.

NOTE: The value of the load multiplier can increase or decrease, and the increment per step should in general also change, depending on the structural response characteristics

Load-Displacement-Constraint methods (3)

- In an incremental-iterative procedure, the loads and displacements are updated by adding the incremental changes from the current iteration to the values obtained in the previous converged step $(n-1)$:

$$\mathbf{v}^{n,i} = \mathbf{v}^{n-1} + \Delta\mathbf{v}^{n,i}$$
$$\mathbf{F}_{ext}^{n,i} = \mathbf{F}_{ext}^{n-1} + \Delta\mathbf{F}_{ext}^{n,i}$$

- On each iteration i for an increment n, the incremental updates are computed by adding the contribution of the previous iteration and the iterative updates at the current iteration:

$$\Delta\mathbf{v}^{n,i} = \Delta\mathbf{v}^{n,i-1} + \delta\mathbf{v}^{n,i}$$
$$\Delta\,\mathbf{F}_{ext}^{n,i} = \Delta\mathbf{F}_{ext}^{n,i-1} + \delta\mathbf{F}_{ext}^{n,i}$$

- The unbalanced force vector is then given by

$$\mathbf{F}_{unb}^{n,i} = \mathbf{F}_{int}^{n,i} - \left(\mathbf{F}_{ext}^{n-1} + \Delta\mathbf{F}_{ext}^{n,i}\right)$$

Where the internal force $\mathbf{F}_{int}^{n,i}$ corresponds to a displacement $\left(\mathbf{v}^{n-1} + \Delta\mathbf{v}^{n,i}\right)$

- Finally, the equilibrium equation to be solved at the i^{th} iteration of the n^{th} increment is given by

$$\mathbf{K}_{structure}^{n,i-1}\delta\mathbf{v}^{n,i} = \mathbf{F}_{ext}^{n,i} - \mathbf{F}_{int}^{n,i-1}$$

Where the stiffness matrix $\mathbf{K}_{structure}^{n,i-1}$ corresponds to the solution schemes previously discussed

Load-Displacement-Constraint methods (4)

- Introducing the load multiplier λ and the reference load vector $\bar{\mathbf{F}}_{ext}$, the governing equations are rewritten:

$$\mathbf{F}_{ext}^{n,i} = \mathbf{F}_{ext}^{n-1} + \Delta\mathbf{F}_{ext}^{n,i-1} + \delta\lambda^{n,i}\bar{\mathbf{F}}_{ext}$$
$$\mathbf{K}_{structure}^{n,i-1}\delta\mathbf{v}^{n,i} = -\mathbf{F}_{unb}^{n,i-1} + \delta\lambda^{n,i}\bar{\mathbf{F}}_{ext}$$

- The unknowns are the vector of displacement increments $\delta\mathbf{v}^{n,i}$ and the load multiplier increment $\delta\lambda^{n,i}$. This system of equation represents $nDOFs$ equations and $nDOFs + 1$ unknowns. Therefore, an additional equation is required to solve the system

- The additional equation is a constraint equation between $\delta\mathbf{v}^{n,i}$ and $\delta\lambda^{n,i}$ of the form

$$\left(\boldsymbol{a}^{n,i}\right)^T\delta\mathbf{v}^{n,i} + b^{n,i}\delta\lambda^{n,i} = c^{n,i}$$

Where $\boldsymbol{a}^{n,i}, b^{n,i}$ and $c^{n,i}$ are parameters to be defined later depending on the control method (load, displacement or arc-length control)

Load-Displacement-Constraint methods (5)

- The system can be written as follows:

$$\mathbf{K}_{structure}^{n,i-1}\delta\mathbf{v}^{n,i} - \delta\lambda^{n,i}\bar{\mathbf{F}}_{ext} = -\mathbf{F}_{unb}^{n,i-1}$$

$$\left(\boldsymbol{a}^{n,i}\right)^{T}\delta\mathbf{v}^{n,i} + b^{n,i}\delta\lambda^{n,i} = c^{n,i}$$

- And the system can be written in a matrix form,

$$\begin{bmatrix} \mathbf{K}_{structure}^{n,i-1} & -\bar{\mathbf{F}}_{ext} \\ \left(\boldsymbol{a}^{n,i}\right)^{T} & b^{n,i} \end{bmatrix} \begin{bmatrix} \delta\mathbf{v}^{n,i} \\ \delta\lambda^{n,i} \end{bmatrix} = - \begin{bmatrix} \mathbf{F}_{unb}^{n,i-1} \\ -c^{n,i} \end{bmatrix}$$

Load-Displacement-Constraint methods (6)

- The system of equations,

$$\begin{bmatrix} \mathbf{K}_{structure}^{n,i-1} & -\bar{\mathbf{F}}_{ext} \\ \left(\boldsymbol{a}^{n,i}\right)^T & b^{n,i} \end{bmatrix} \begin{bmatrix} \delta\mathbf{v}^{n,i} \\ \delta\lambda^{n,i} \end{bmatrix} = - \begin{bmatrix} \mathbf{F}_{unb}^{n,i-1} \\ -c^{n,i} \end{bmatrix}$$

can be solved using the iterative schemes discussed previously. However, this approach may not be optimal because the augmented stiffness matrix on the left-hand side is neither symmetric nor banded, unlike the original structure stiffness matrix. This increases the computational cost associated with storing and inverting the matrix

- Another approach to solve the same system is as follows:
  - From the first equation of the linear system:

$$\mathbf{K}_{structure}^{n,i-1}\delta\mathbf{v}^{n,i} - \bar{\mathbf{F}}_{ext}\delta\lambda^{n,i} = -\mathbf{F}_{unb}^{n,i-1} \quad \rightarrow \delta\mathbf{v}^{n,i} = \left(\mathbf{K}_{structure}^{n,i-1}\right)^{-1}\left(\bar{\mathbf{F}}_{ext}\delta\lambda^{n,i} - \mathbf{F}_{unb}^{n,i-1}\right)$$

$$\delta\mathbf{v}_p^{n,i} = \left(\mathbf{K}_{structure}^{n,i-1}\right)^{-1}\bar{\mathbf{F}}_{ext} \quad \text{and} \quad \delta\mathbf{v}_r^{n,i} = -\left(\mathbf{K}_{structure}^{n,i-1}\right)^{-1}\mathbf{F}_{unb}^{n,i-1}$$

  - We can now decompose the displacement iteration update as follows:

$$\delta\mathbf{v}^{n,i} = \delta\lambda^{n,i}\delta\mathbf{v}_p^{n,i} + \delta\mathbf{v}_r^{n,i}$$

$\delta\lambda^{n,i}$ is computed from the constraint equation after substituting $\delta\mathbf{v}^{n,i}$ with the expression above

Load-Displacement-Constraint methods (7)

- The iterative update in the load multiplier is then computed by solving the constraint equation:

$$\delta\lambda^{n,i} = \frac{c^{n,i} - \left(\boldsymbol{a}^{n,i}\right)^T \delta\mathbf{v}_r^{n,i}}{\left(\boldsymbol{a}^{n,i}\right)^T \delta\mathbf{v}_p^{n,i} + b^{n,i}}$$

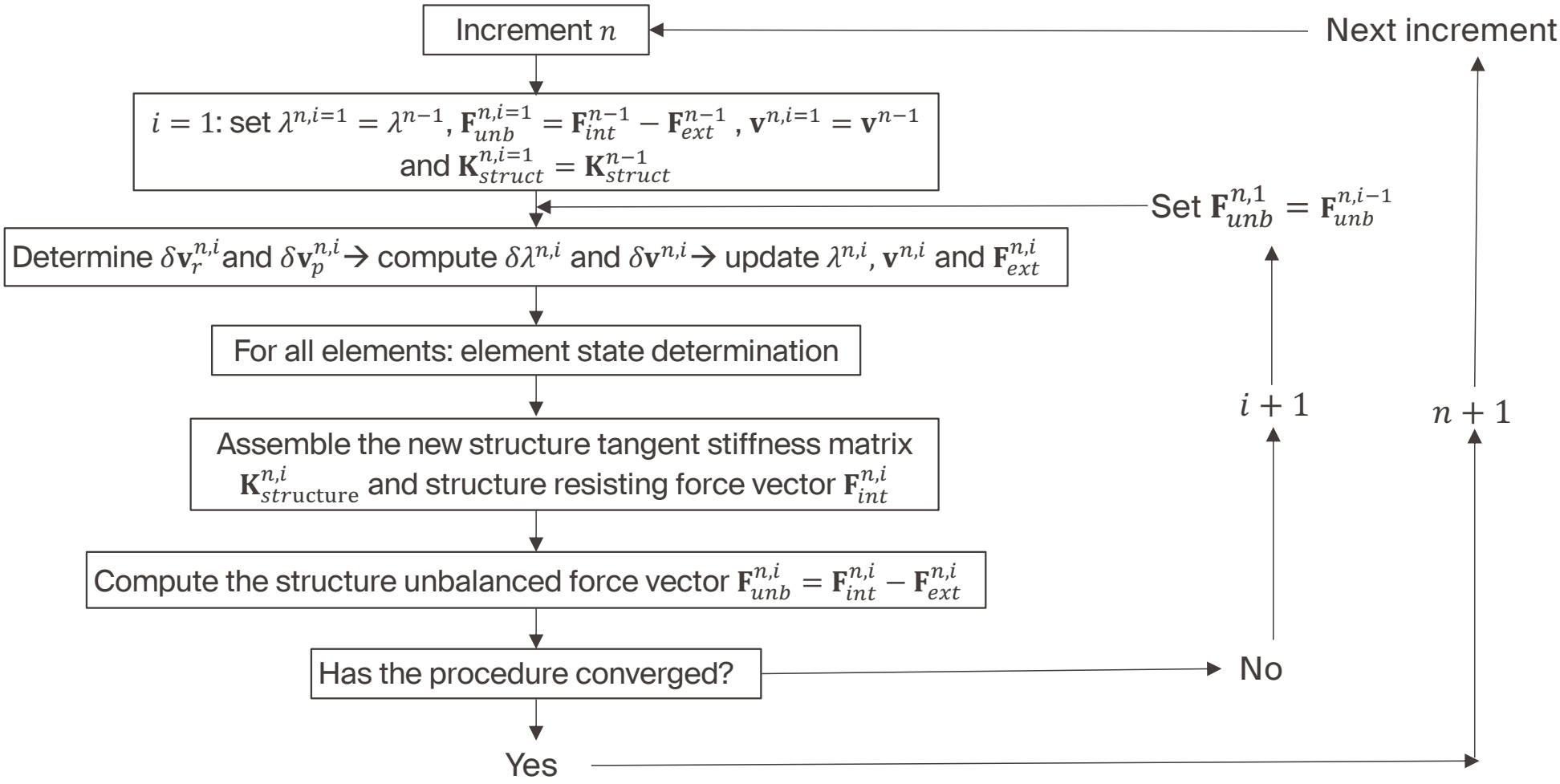
- The benefit of using this approach is that at each iteration, the Jacobian of the problem corresponds to the stiffness matrix $\mathbf{K}_{structure}^{n,i-1}$, which is banded and symmetric, enabling the use of optimized solvers when inverting and storing the matrix when solving for the unknown displacements

- NOTE: from linear algebra, a banded matrix is a sparse matrix whose non-zero entries are confined to a diagonal band, comprising the main diagonal and zero or more diagonals on either side

$$\mathrm{A} = \begin{bmatrix} a_{00} & a_{01} & a_{02} & 0 & 0 & 0 & 0 & 0 \\ a_{10} & a_{11} & a_{12} & a_{13} & 0 & 0 & 0 & 0 \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} & 0 & 0 & 0 \\ 0 & a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & 0 & 0 \\ 0 & 0 & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & 0 \\ 0 & 0 & 0 & a_{53} & a_{54} & a_{55} & a_{56} & a_{57} \\ 0 & 0 & 0 & 0 & a_{64} & a_{65} & a_{66} & a_{67} \\ 0 & 0 & 0 & 0 & 0 & a_{75} & a_{76} & a_{77} \end{bmatrix}$$

Load-Displacement-Constraint methods (8)

- The flow chart gives the solution process for the general load-displacement constraint methods



Load control

- In load control, the vector of applied external forces is known at every step. The load multiplier is therefore also known. The external forces are computed at the first iteration $(i = 1)$ of each increment n and held constant in the remaining iterations,

- The constraint equation between $\delta \mathbf{v}^{n,i}$ and $\delta \lambda^{n,i}$ is given by the following:

$$\mathbf{0}^T \cdot \delta \mathbf{v}^{n,i} + 1 \cdot \delta \lambda^{n,i} = \begin{cases} \Delta \bar{\lambda}^n \text{ if } i = 1 \\ 0 \text{ else} \end{cases}$$

Where $\Delta \bar{\lambda}^n$ denotes the prescribed increment in load multiplier

- In this case, the constants defining the general form of the constraints are given by

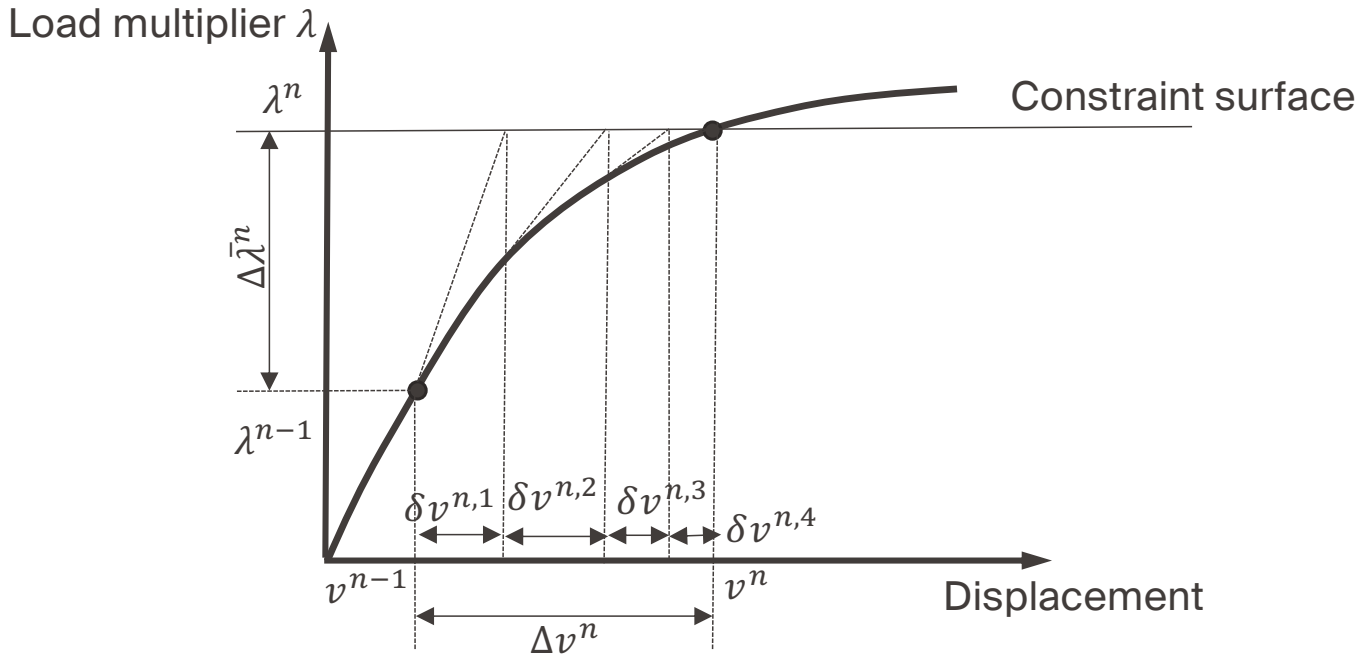
$$\boldsymbol{a}^{n,i} = \mathbf{0},\ b^{n,i} = 1 \text{ and } c^{n,i} = \begin{cases} \Delta \bar{\lambda}^n \text{ if } i = 1 \\ 0 \text{ else} \end{cases}$$

- Therefore, for iteration i of the increment n, the iterative update to the load multiplier $\delta \lambda^{n,i}$ is given by

$$\delta \lambda^{n,i} = \frac{c^{n,i} - \left(\boldsymbol{a}^{n,i}\right)^T \delta \mathbf{v}_r^{n,i}}{(\boldsymbol{a}^{n,i})^T \delta \mathbf{v}_p^{n,i} + b^{n,i}} = \begin{cases} \Delta \bar{\lambda}^n \text{ if } i = 1 \\ 0 \text{ else} \end{cases}$$

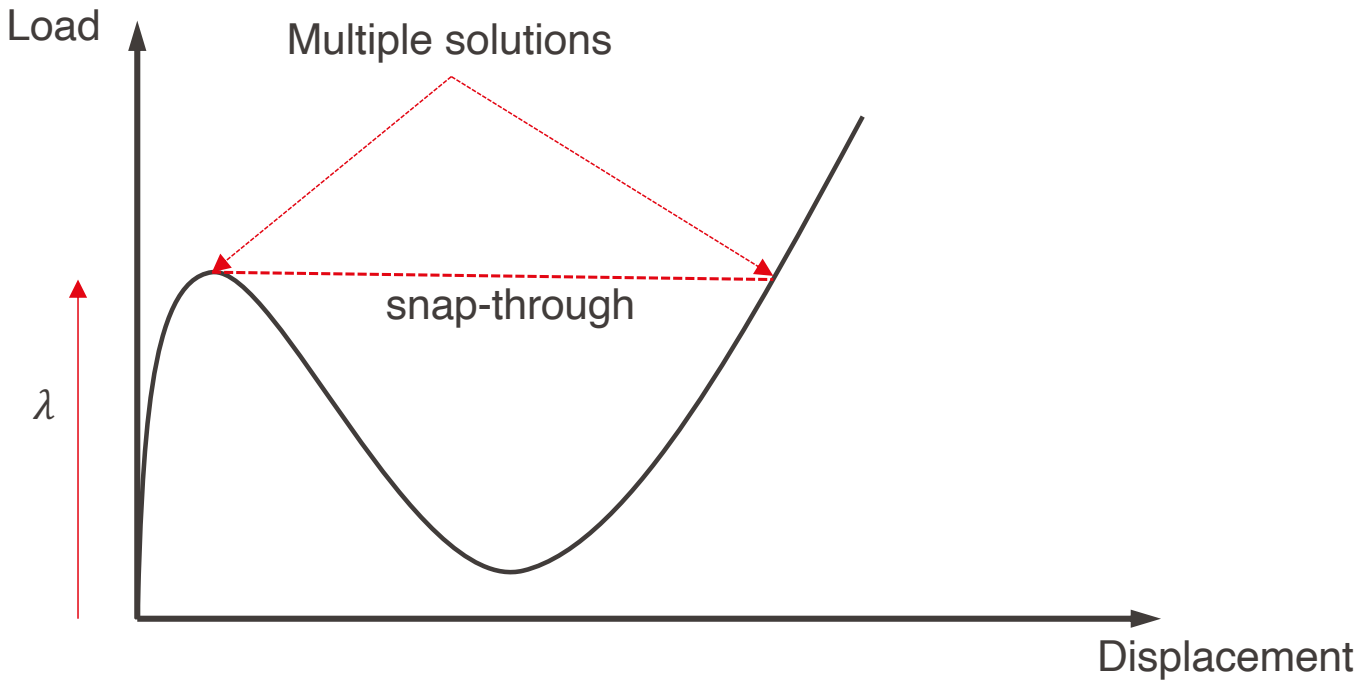
Load control (2)

- The following figure schematically illustrates the solution process for a step, in load control

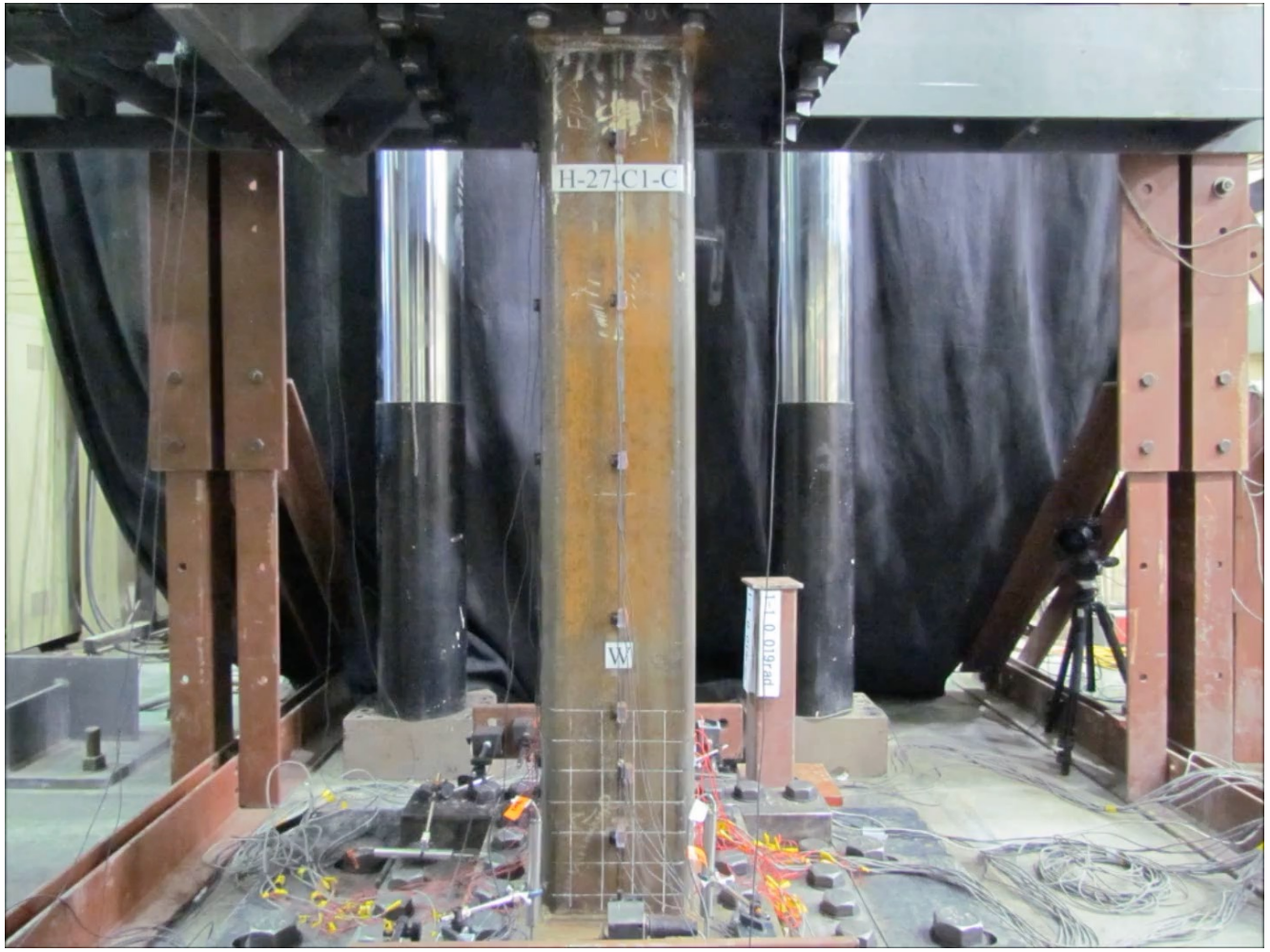


Load control (3)

- Load control fails to converge in a load-displacement equilibrium path which exhibits snap-through

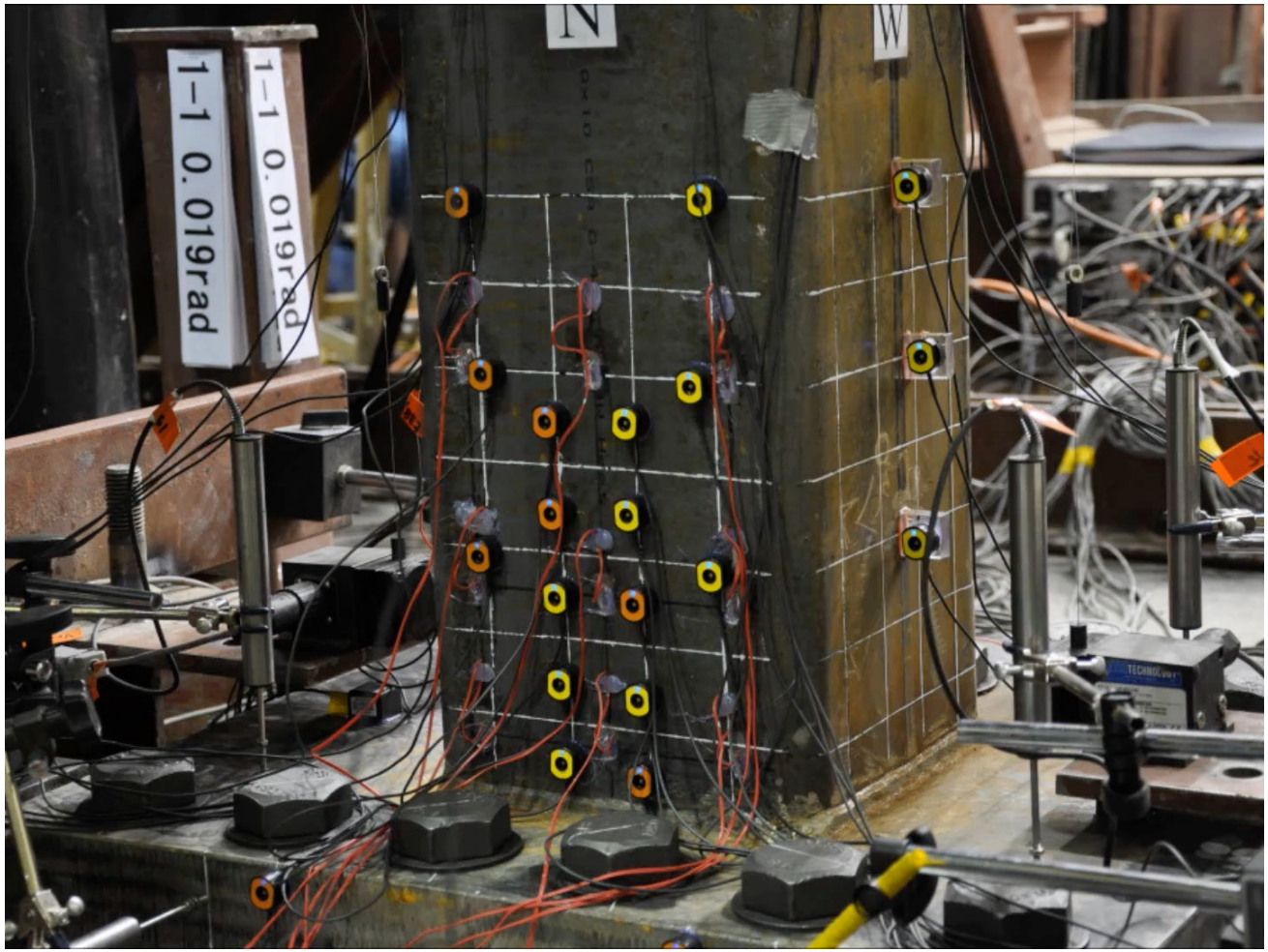


Load control (4): Instability of steel columns



Source: Suzuki and Lignos (2021)

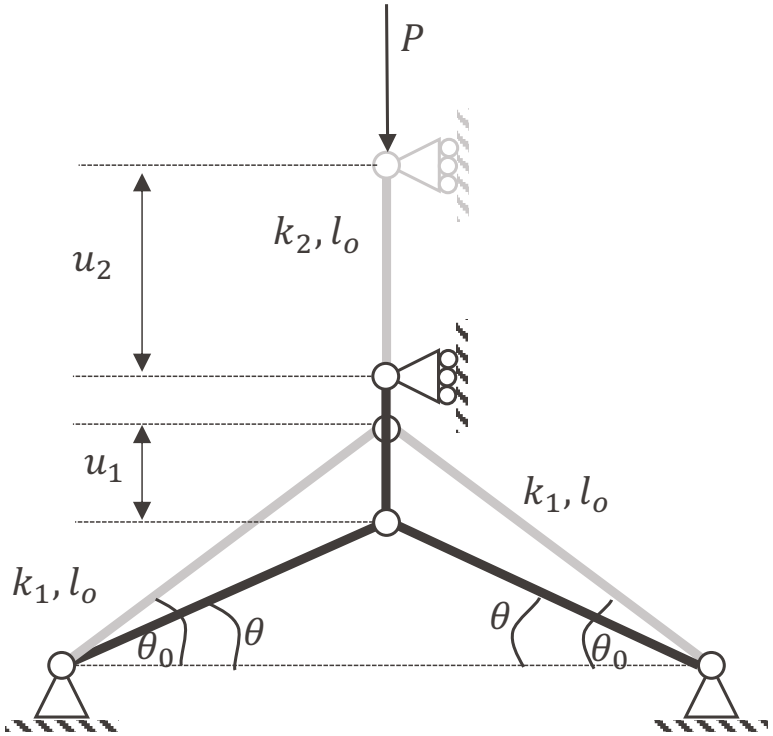
Load control (5): Instability of steel columns



Source: Suzuki and Lignos (2021)

Example: Load control

- Consider the following example:



- Three truss elements, each with the same undeformed length l_u have axial stiffness $k_j = \frac{EA_j}{l_o}$

Example: Load control (2)

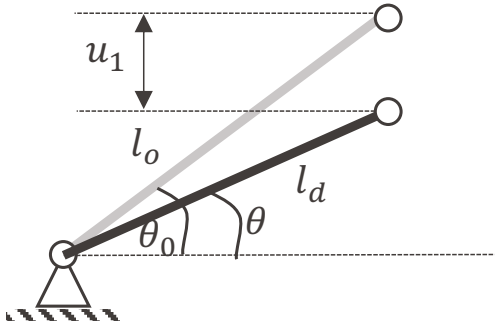
- The deformation of the vertical truss element is given by $e_2 = \frac{u_2 - u_1}{l_o} = a_2 - a_1$
  - where a_1 and a_2 are the normalized displacements
- The internal force N_2 inside the vertical truss element is given by $N_2 = k_2 l_o (a_2 - a_1)$
- The internal force N_1 inside the inclined truss elements is given by $N_1 = k_1 \, (l_o - l_d)$ where l_d is the deformed length of the element
- By considering force equilibrium at both extremities of the vertical truss element, the following relations are obtained:

$$N_2 = 2N_1 \sin(\theta)$$
$$N_2 = P$$

NOTE: here the equalities are in terms of absolute value (not vectoral). This is also the reason why we have used $l_o - l_d > 0$ in the definition of N_1(and not $l_d - l_o < 0$)

Example: Load control (3)

- Consider the following figure



- Using Pythagoras theorem, we have

$$l_d^2 = (l_o \sin(\theta_0) - u_1)^2 + (l_o \cos(\theta_0))^2 \rightarrow \frac{l_d}{l_o} = \sqrt{1 - \frac{2u_1}{l_o}\sin(\theta_0) + \left(\frac{u_1}{l_o}\right)^2}$$

$$\sin(\theta) = \frac{l_o \sin(\theta_0) - u_1}{l_d}$$

- Thus

$$N_1 \sin(\theta) = k_1 (l_o - l_d) \frac{l_o \sin(\theta_0) - u_1}{l_d} = k_1 \left(\frac{l_o}{l_d} - 1\right)(l_o \sin(\theta_0) - u_1) = k_1 l_o \left(\frac{1}{\sqrt{B(a_1)}} - 1\right)(\sin(\theta_0) - a_1)$$

Where

$$B(a_1) = 1 - 2a_1 \sin(\theta_0) + a_1^2$$

Example: Load control (4)

- The force equilibrium conditions can be rewritten as follows

$$\begin{cases} 2k_1 l_o \left(\dfrac{1}{\sqrt{B(a_1)}} - 1 \right) (\sin(\theta_0) - a_1) - k_2 l_o (a_2 - a_1) \\ k_2 l_o (a_2 - a_1) - P = 0 \end{cases}$$

- Defining the load multiplier as $\lambda = \dfrac{P}{2k_1 l_o}$ and the stiffness ratio between the truss members as $w = \dfrac{k_2}{k_1}$, the above system of equations can be rewritten as

$$\begin{cases} \left(\dfrac{1}{\sqrt{B(a_1)}} - 1 \right) (\sin(\theta_0) - a_1) - w(a_2 - a_1) \\ w(a_2 - a_1) - \lambda = 0 \end{cases}$$

Example: Load control (5)

- The system of equations has the form $\mathbf{F}_{int}(\mathbf{v}) - \lambda\bar{\mathbf{F}}_{ext}$ with

$$\mathbf{F}_{int}(\mathbf{v}) = \mathbf{F}_{int}(a_1, a_2) = \begin{bmatrix} F_1(a_1 a_2) \\ F_2(a_1 a_2) \end{bmatrix} = \begin{bmatrix} \left(\frac{1}{\sqrt{B(a_1)}} - 1\right)(\sin(\theta_0) - a_1) - w(a_2 - a_1) \\ w(a_2 - a_1) \end{bmatrix}$$

And

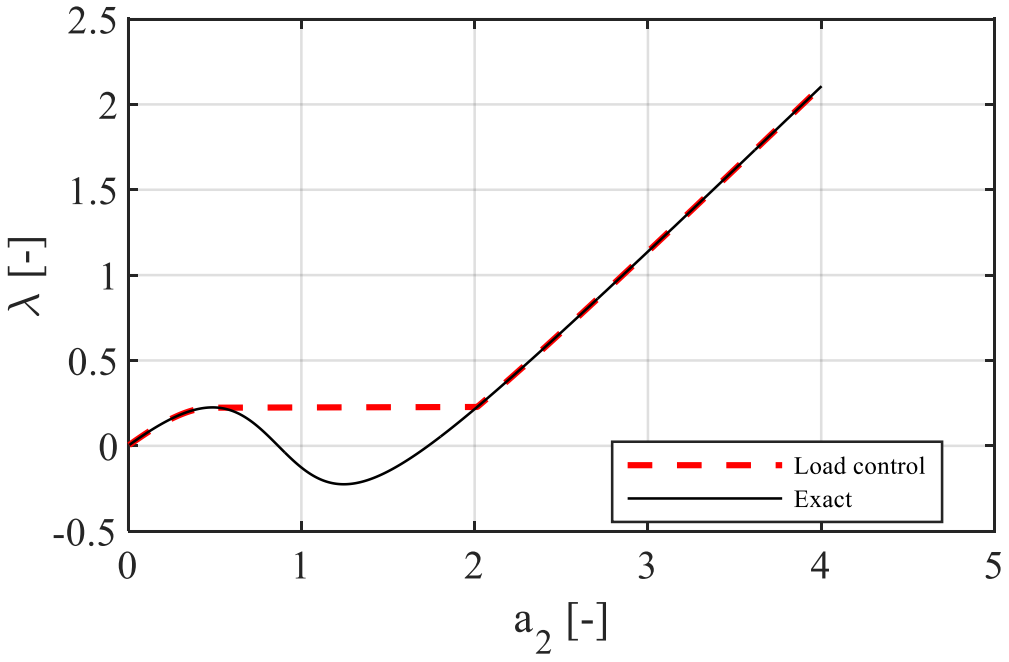
$$\bar{\mathbf{F}}_{ext} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- The structure stiffness matrix (i.e. the Jacobian) of the problem is given by

$$\mathbf{K}_{structure} = \begin{bmatrix} \frac{\partial F_1}{\partial a_1} & \frac{\partial F_1}{\partial a_2} \\ \frac{\partial F_1}{\partial a_2} & \frac{\partial F_2}{\partial a_2} \end{bmatrix} = \begin{bmatrix} (1+w) - \frac{(1-\sin(\theta_0)^2)}{B^{\frac{3}{2}}} & -w \\ -w & w \end{bmatrix}$$

Example: Load control (6)

- The system is solved for $w = \frac{k_2}{k_1} = 50$ using load control with a Newton-Raphson algorithm



- The load control algorithm cannot capture the snap-trough response

Displacement control

- Displacement control is an efficient method in solving systems which exhibit snap-through behavior. The value of the displacement at a specific dof is prescribed in this case.

- The constraint equation between $\delta \mathbf{v}^{n,i}$ and $\delta \lambda^{n,i}$ is therefore given by

$$\delta \mathbf{v}_q^{n,i} + 0 \cdot \delta \lambda^{n,i} = \begin{cases} \Delta \bar{\mathrm{v}}^n \text{ if } i = 1 \\ 0 \text{ else} \end{cases}$$

Where the subscript q denotes the selected dof q at which the control displacement is imposed and $\Delta \bar{\mathrm{v}}^n$ denotes the prescribed increment in displacement

- In this case, the constants defining the general form of the constraints are given by

$$\boldsymbol{a}^{n,i} = [0, 0, \dots, 1, \dots, 0, 0]^T,\ b^{n,i} = 0 \text{ and } c^{n,i} = \begin{cases} \Delta \bar{\mathrm{v}}^n \text{ if } i = 1 \\ 0 \text{ else} \end{cases}$$

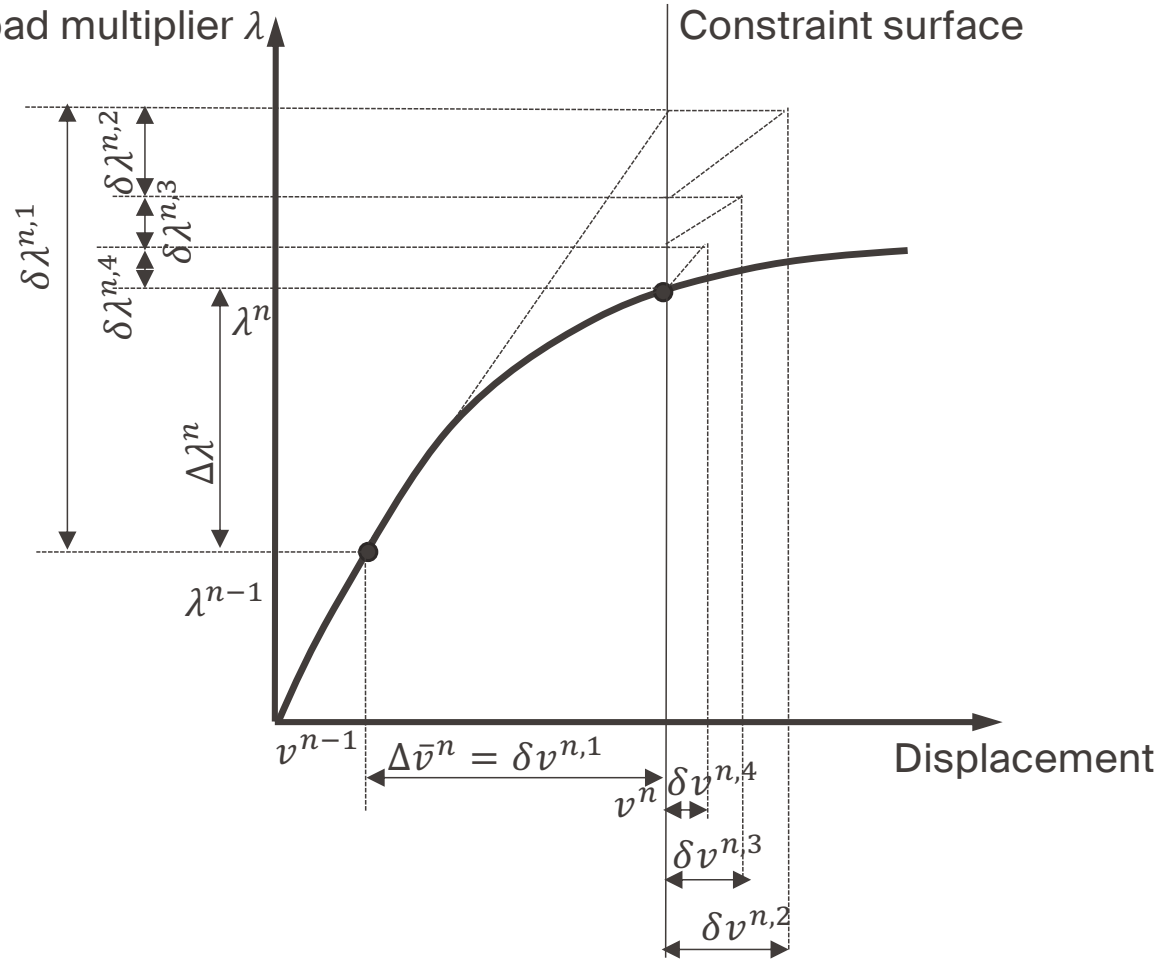
Where the where the entry 1 in $\boldsymbol{a}^{n,i}$ corresponds to the dof q

- Therefore, for iteration i of the increment n, the iterative update to the load multiplier $\delta \lambda^{n,i}$ is given by

$$\delta \lambda^{n,i} = \frac{c^{n,i} - \left(\boldsymbol{a}^{n,i}\right)^T \delta \mathbf{v}_r^{n,i}}{(\boldsymbol{a}^{n,i})^T \delta \mathbf{v}_p^{n,i} + b^{n,i}} = \begin{cases} \dfrac{\Delta \bar{\mathrm{v}}^n}{\delta \mathbf{v}_p^{n,i}} \text{if } i = 1 \ (note, \delta \mathbf{v}_r^{n,1} = \mathbf{0}) \\ -\dfrac{\delta \mathbf{v}_r^{n,i}}{\delta \mathbf{v}_p^{n,i}} \text{ else} \end{cases}$$

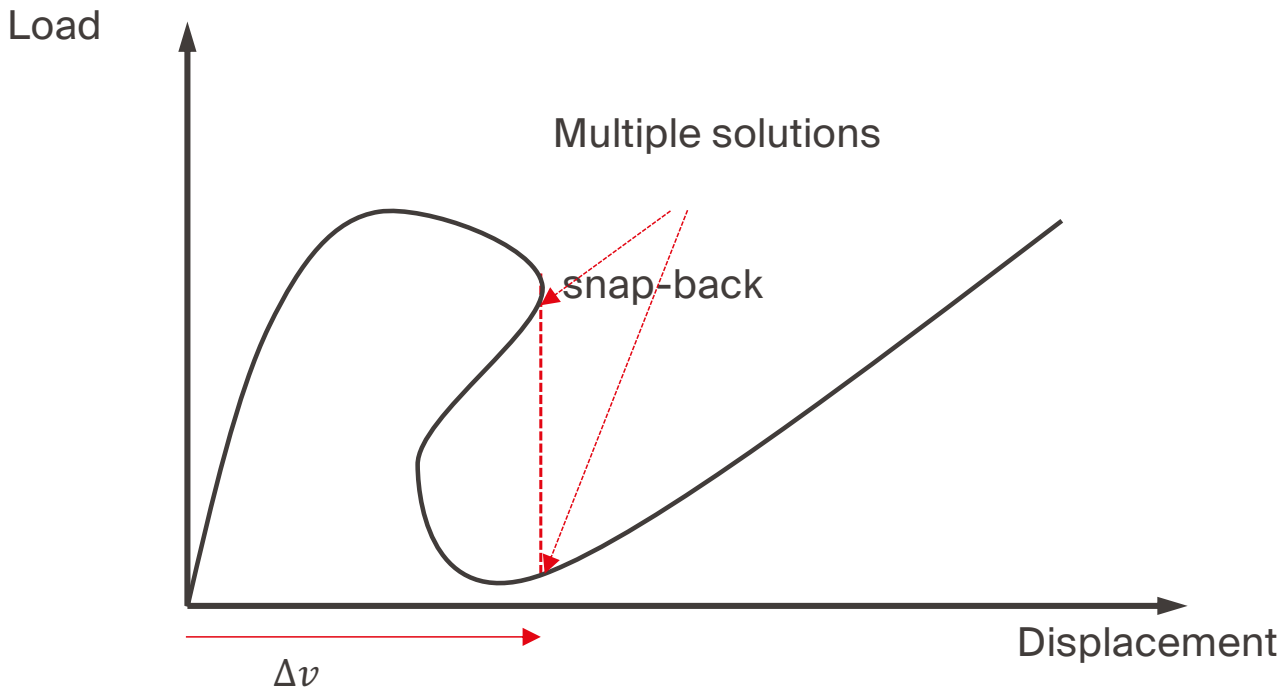
Displacement control (2)

- The following figure schematically illustrates the solution process for a step in displacement control



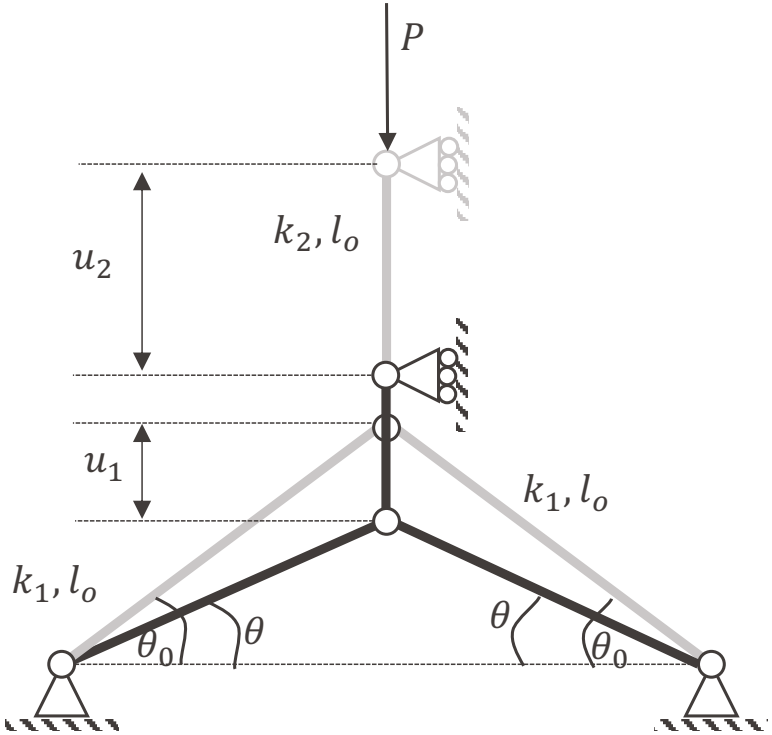
Displacement control (3)

- Displacement control fails to converge in a load-displacement which exhibits snap-back



Example: Displacement control

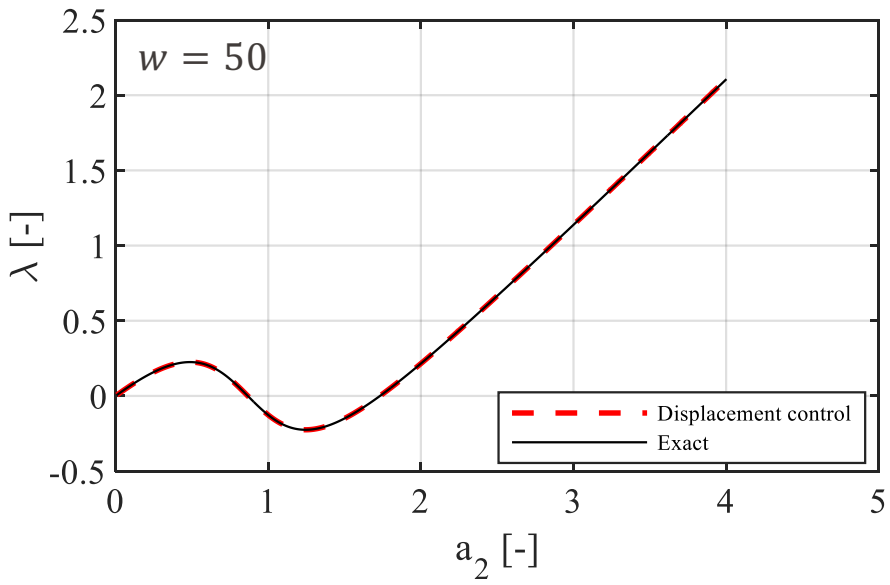
- Consider the previous example shown below

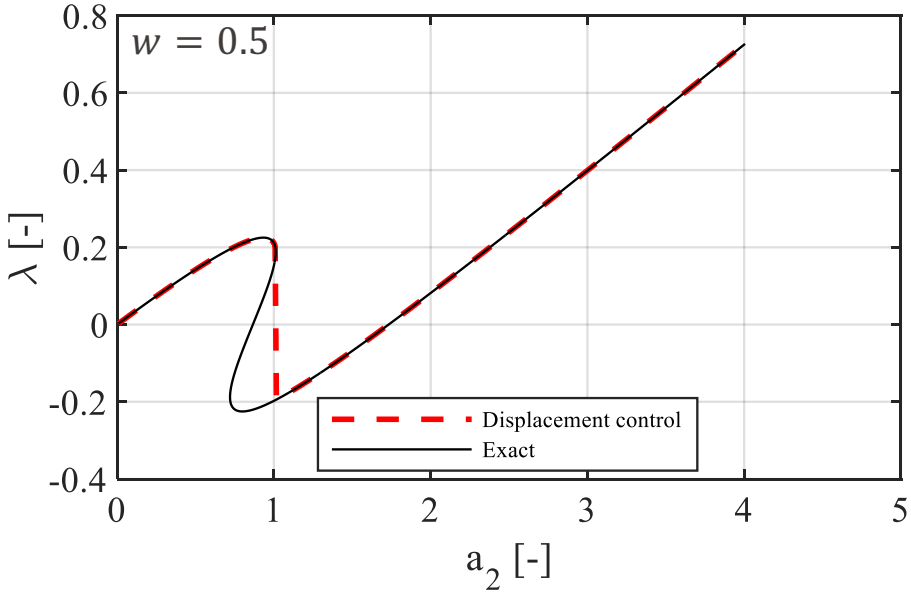


- Determine the equilibrium path using displacement control and Newton-Raphson for $w = \frac{k_2}{k_1} = 50$ and $w = 0.5$
- Use a_2 as the controlling dof for the displacement control algorithm

Example: Displacement control (2)

- The following results are obtained:





- Displacement control can capture snap-through responses but cannot capture snap-back responses

Arc-Length control

- Arc-length control (Riks 1979) is an efficient method in solving systems, which exhibit snap-back behaviour. The method postulates a simultaneous increment in both displacement $\Delta\mathbf{v}$ and load multiplier $\Delta\lambda$. The method is based on the constraint of each increment n to a prescribed arc-length of length $\Delta\bar{l}^n$. Mathematically, this constraint can be written as:

$$\left(\Delta\mathbf{v}^{n,i}\right)^T\Delta\mathbf{v}^{n,i} + \psi^2\left(\Delta\lambda^{n,i}\right)^2 = \left(\Delta\bar{l}^n\right)^2$$

Where $\Delta\bar{l}^n$ is the prescribed arc length for the step n; ψ is a user-defined scalar parameter. When $\psi = 1$, the method is called spherical arc-length method (the constraint corresponds to a sphere of radius $\Delta\bar{l}^n$). Similarly, when $\psi = 0$, the method is called cylindrical arc-length method. For arbitrary ψ, the constraint is geometrically interpreted as a hyper-ellipse in the multidimensional displacement-load space $(\Delta\mathbf{v} - \lambda)$

- An initial arc-length iteration is determined by the constraint equation and the following iterations lie on the constrained surface created by the arc. The constraint equation can therefore be rewritten using the iteration updates instead of the increment update:

$$\left(\delta\mathbf{v}^{n,i=1}\right)^T\delta\mathbf{v}^{n,i} + \psi^2\delta\lambda^{n,i=1}\delta\lambda^{n,i} = \left(\Delta l^{n,i}\right)^2$$

Where $\Delta l^{n,i}$ denotes the value of the arc length for iteration i

Arc-Length control (2)

- The constraint equation between $\delta\mathbf{v}^{n,i}$ and $\delta\lambda^{n,i}$ is therefore given by

$$\left(\delta\mathbf{v}^{n,i=1}\right)^T \cdot \delta\mathbf{v}^{n,i} + \psi^2 \delta\lambda^{n,i=1} \cdot \delta\lambda^{n,i} = \begin{cases} \left(\Delta\bar{l}^n\right)^2 \text{ if } i = 1 \\ 0 \text{ else} \end{cases}$$

- In this case, the constants defining the general form of the constraints are given by

$$\boldsymbol{a}^{n,i} = \delta\mathbf{v}^{n,i=1} = \delta\lambda^{n,i=1} \cdot \delta\mathbf{v}_p^{n,i=1},\ b^{n,i} = \psi^2 \delta\lambda^{n,i=1} \text{ and } c^{n,i} = \begin{cases} \left(\Delta\bar{l}^n\right)^2 \text{ if } i = 1 \\ 0 \text{ else} \end{cases}$$

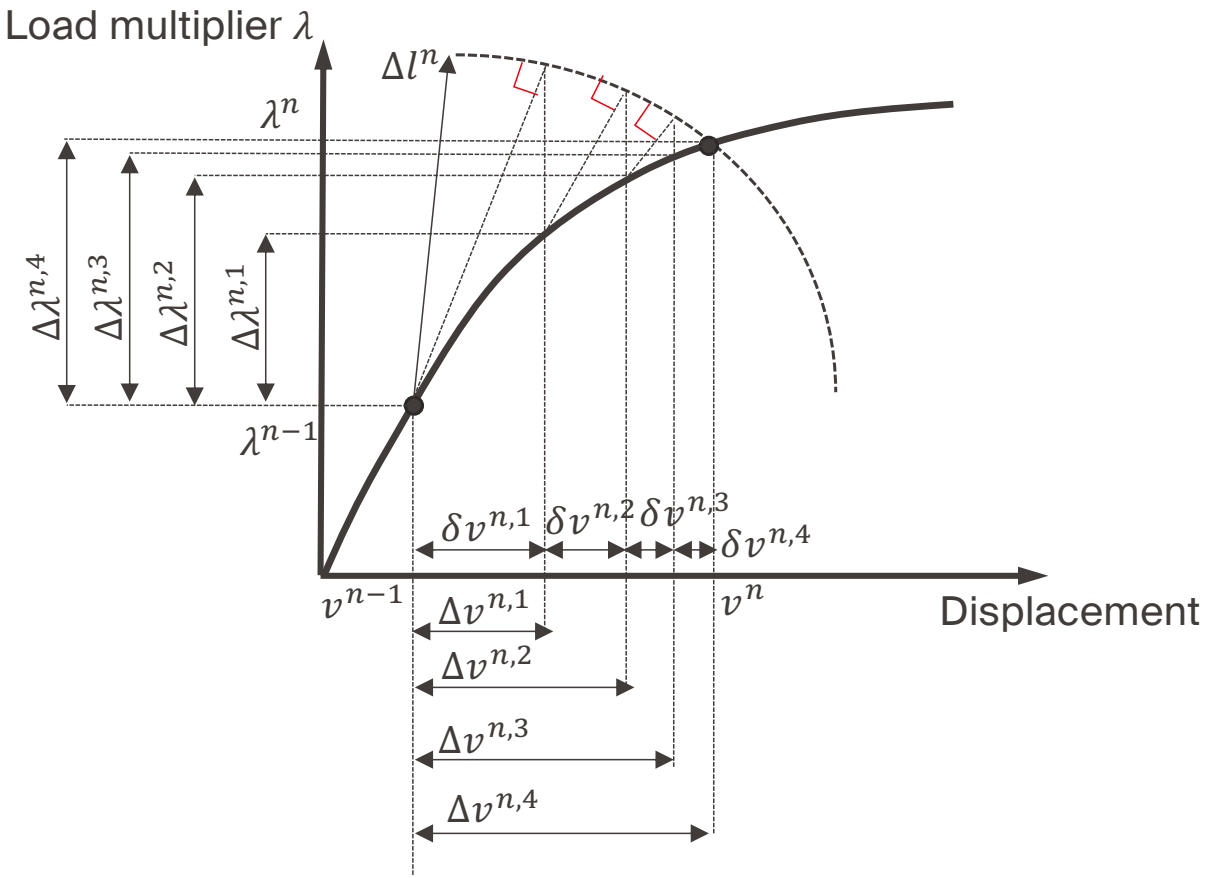
- Therefore, for iteration i of the increment n, the iterative update to the load multiplier $\delta\lambda^{n,i}$ is given by

$$\delta\lambda^{n,i} = \frac{c^{n,i} - \left(\boldsymbol{a}^{n,i}\right)^T \delta\mathbf{v}_r^{n,i}}{(\boldsymbol{a}^{n,i})^T \delta\mathbf{v}_p^{n,i} + b^{n,i}} = \begin{cases} \pm \dfrac{\Delta\bar{l}^n}{\sqrt{\left(\delta\mathbf{v}_p^{n,i=1}\right)^T \delta\mathbf{v}_p^{n,i=1} + \psi^2}} \text{if } i = 1 (note, \delta\mathbf{v}_r^{n,1} = \mathbf{0}) \\ -\dfrac{\left(\delta\mathbf{v}^{n,i=1}\right)^T \delta\mathbf{v}_r^{n,i}}{(\delta\mathbf{v}^{n,i=1})^T \delta\mathbf{v}_p^{n,i} + \psi^2 \delta\lambda^{n,i=1}} \text{ else} \end{cases}$$

The $\pm$ sign in the first iteration arises from the quadratic nature of the constraint equation. It is possible that both roots may be admissible for the initial iterative update of the load multiplier $\delta\lambda^{n,i=1}$. The selection of the appropriate root will be discussed in a few slides

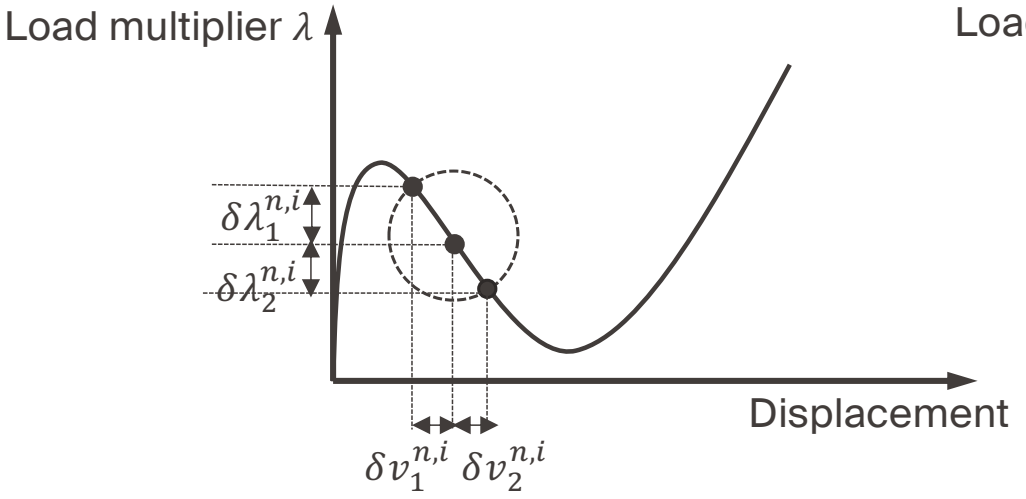
Arc-Length control (3)

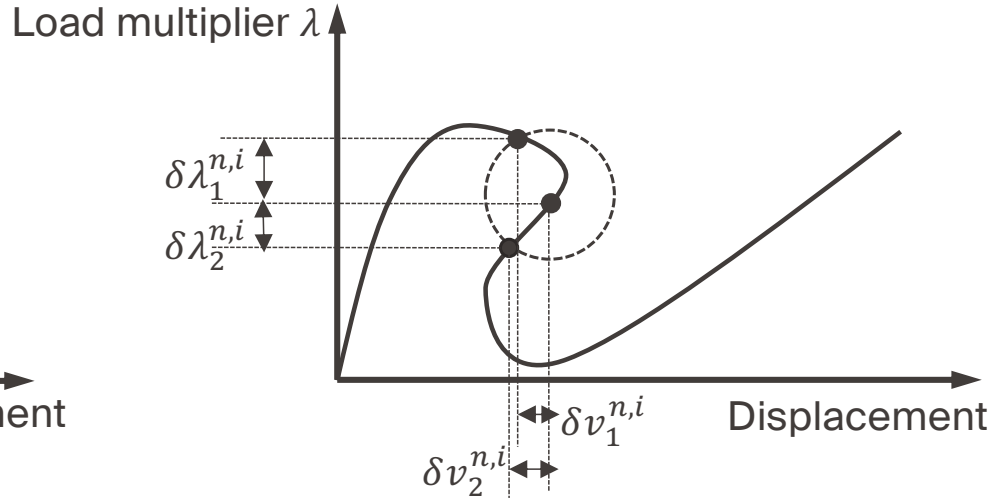
- The following figure schematically illustrates the solution process for a given step with arc-length control



Arc-Length control (4)

- As previously observed, the drawback of the method is related to the two distinct solutions $(\delta \mathbf{v}_1^{n,i}, \delta\lambda_1^{n,i})$ and $(\delta \mathbf{v}_2^{n,i}, \delta\lambda_2^{n,i})$ that may be admissible for the first iteration $i = 1$ of an increment n

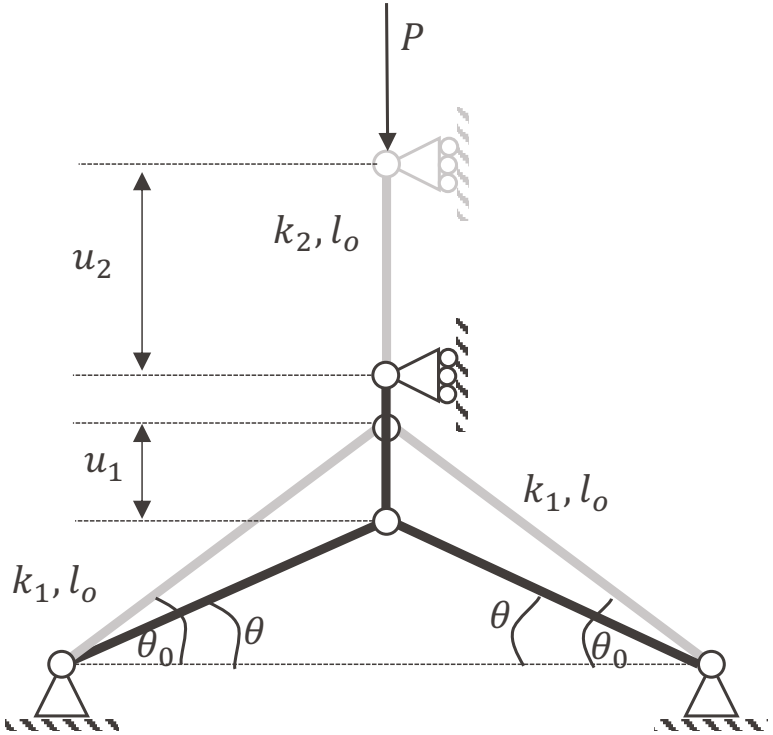




- One approach to selecting the appropriate root is to choose the one that has the same sign as the determinant of the current structure stiffness matrix

Example: Arc-Length control

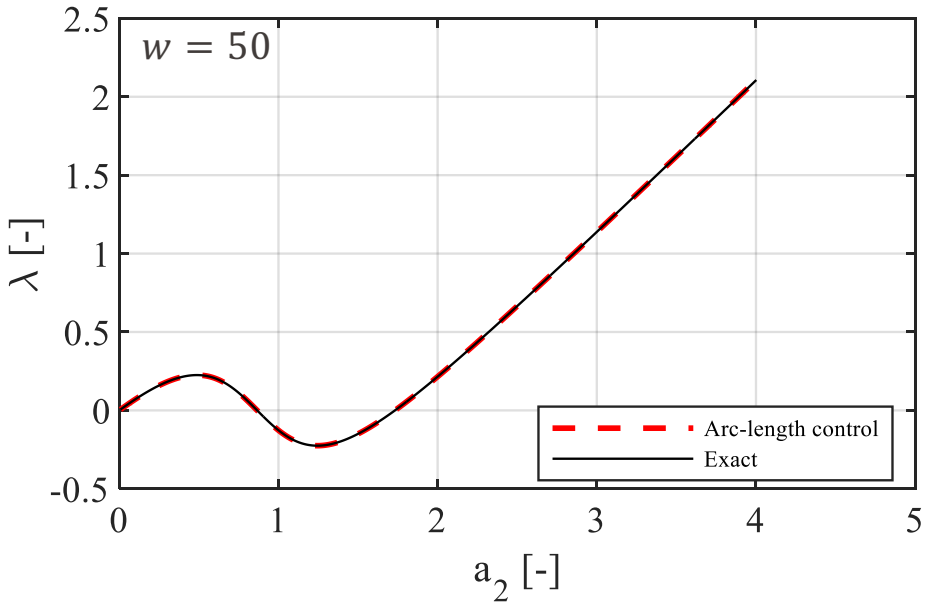
- Consider the previous example shown below

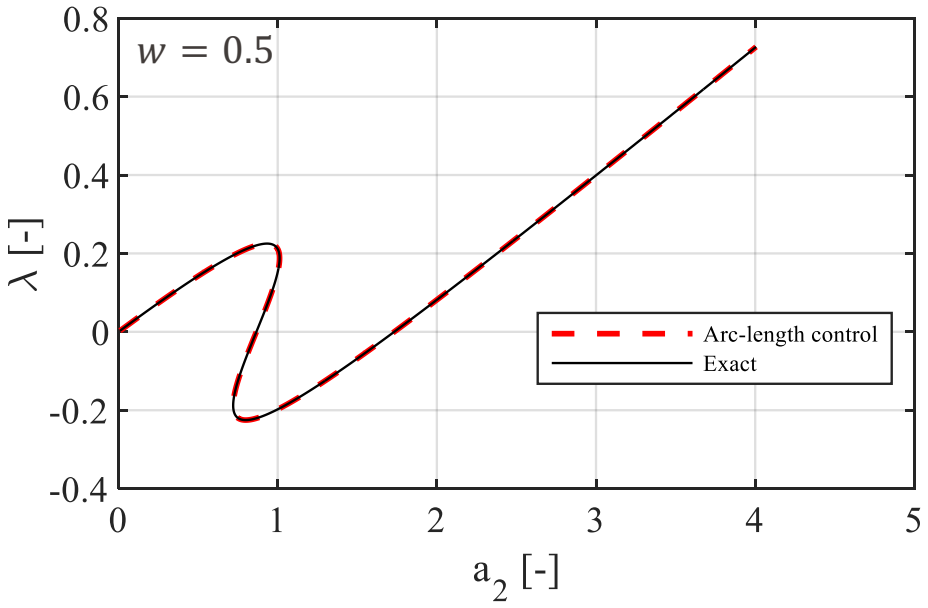


- Determine the equilibrium path using the arc-length control and Newton-Raphson iterations for $w = \frac{k_2}{k_1} = 50$ and $w = 0.5$. Use the cylindrical arc-length (i.e. $\psi = 0$) with a constant arc length $\Delta\bar{l}$

Example: Arc-Length control (2)

- The following results are obtained:





- Arc-length control can capture both snap-through and snap-back responses