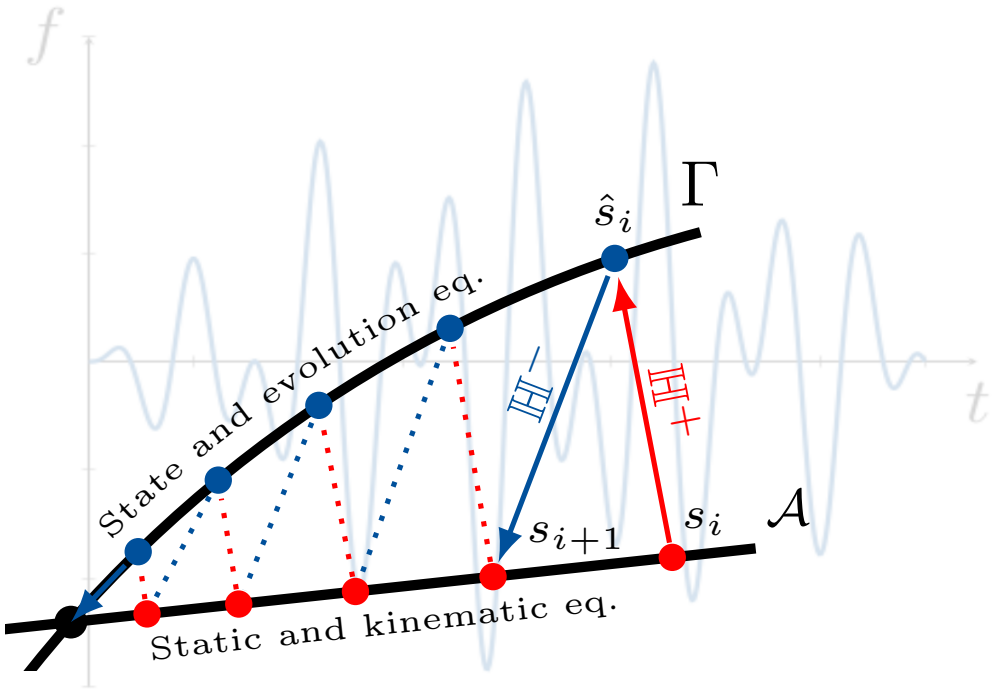
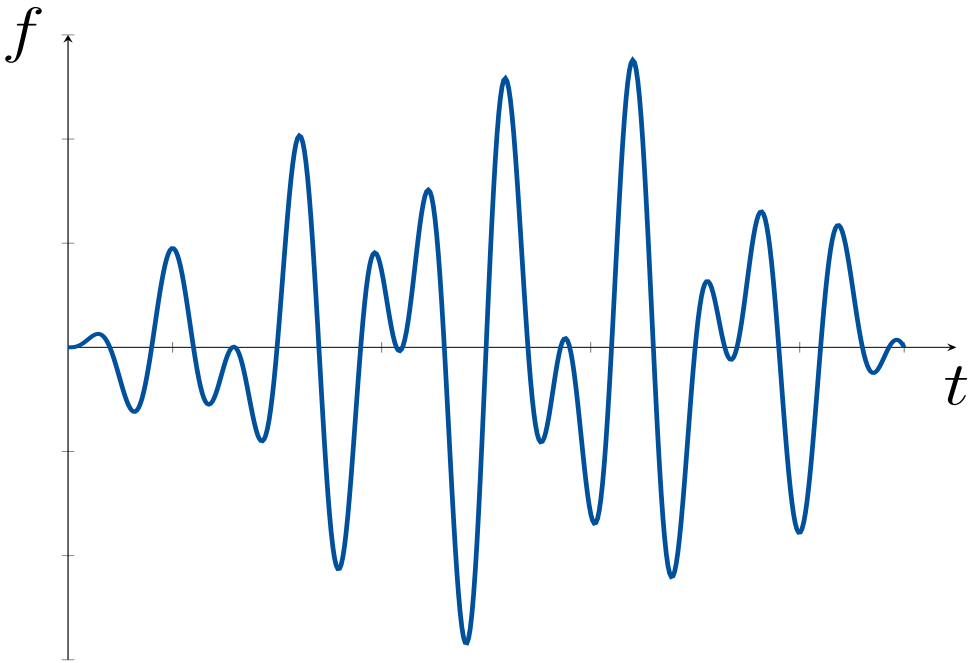
A Semi-incremental Model Order Reduction Approach for Fatigue Damage Computations

Shadi Alameddin
IBNM, Leibniz Universität Hannover
06. February 2020



IRTG-1627
DFG Deutsche Forschungsgemeinschaft

A Semi-incremental Model Order Reduction Approach for Fatigue Damage Computations

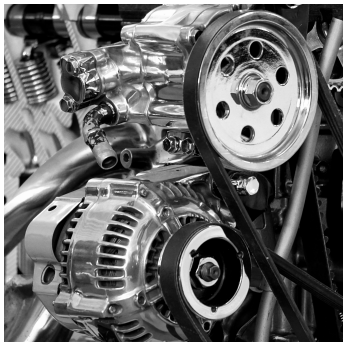


Shadi Alameddin
IBNM, Leibniz Universität Hannover
06. February 2020

IRTG-1627
DFG Deutsche Forschungsgemeinschaft

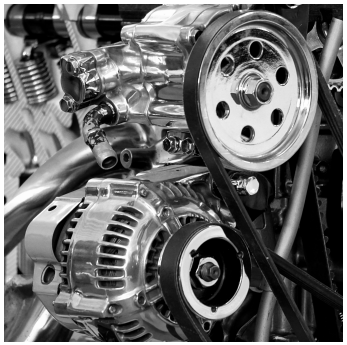
Fatigue damage

- Fluctuating loads


Fatigue damage

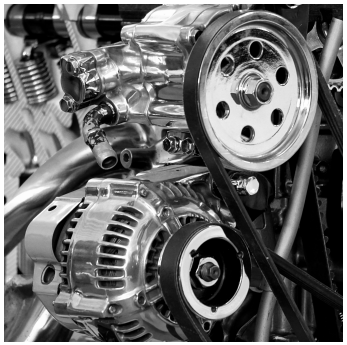
- Fluctuating loads


- Large number of load cycles
  thousands or millions
- Material degradation
  lower load carrying capacity
- Continuum damage model
  damage evolution in rate form
- Computational expense
  memory and time

Fatigue damage

- Fluctuating loads


- Large number of load cycles
  thousands or millions
- Material degradation
  lower load carrying capacity
- Continuum damage model
  damage evolution in rate form
- Computational expense
  memory and time

Model order reduction (MOR) techniques

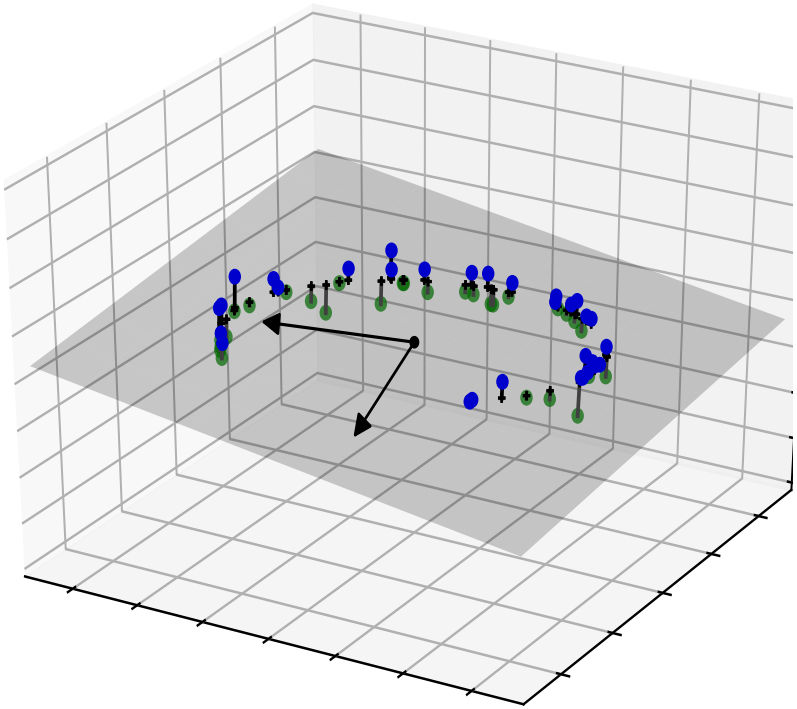
Outline

Outline

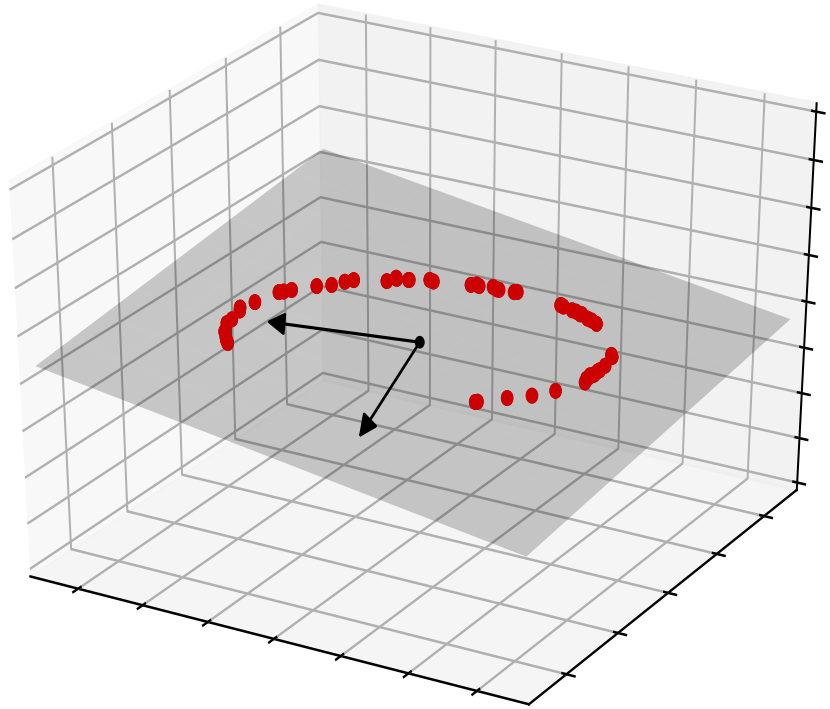
Dimensionality Reduction

Projection-based approach

Solutions lie close to a low-dimensional subspace



High-dimensional space



2D subspace

Mechanical problem

Infinitesimal strain and isothermal quasi-static settings

Admissibility equations (**AD**)

Static admissibility

$$\nabla \cdot \boldsymbol{\sigma} + \boldsymbol{b} = \boldsymbol{0} \qquad \text{in } \Omega \times \mathcal{T}$$

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \bar{\boldsymbol{t}} \qquad \text{on } \partial\Omega_{\mathrm{N}} \times \mathcal{T}$$

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^{\mathsf{T}}$$

Kinematic admissibility

$$\boldsymbol{\varepsilon} = \nabla^{\mathrm{S}} \boldsymbol{u} \qquad \text{in } \Omega \times \mathcal{T}$$

$$\boldsymbol{u} = \bar{\boldsymbol{u}} \qquad \text{on } \partial\Omega_{\mathrm{D}} \times \mathcal{T}$$

Nonlinear constitutive model (**CM**)

State equations

$$\mathcal{F} = f(\psi(\boldsymbol{\varepsilon}, \mathcal{I}))$$

Evolution equations

$$\dot{\mathcal{I}} = g(\phi^*(\boldsymbol{\sigma}, \mathcal{F}))$$

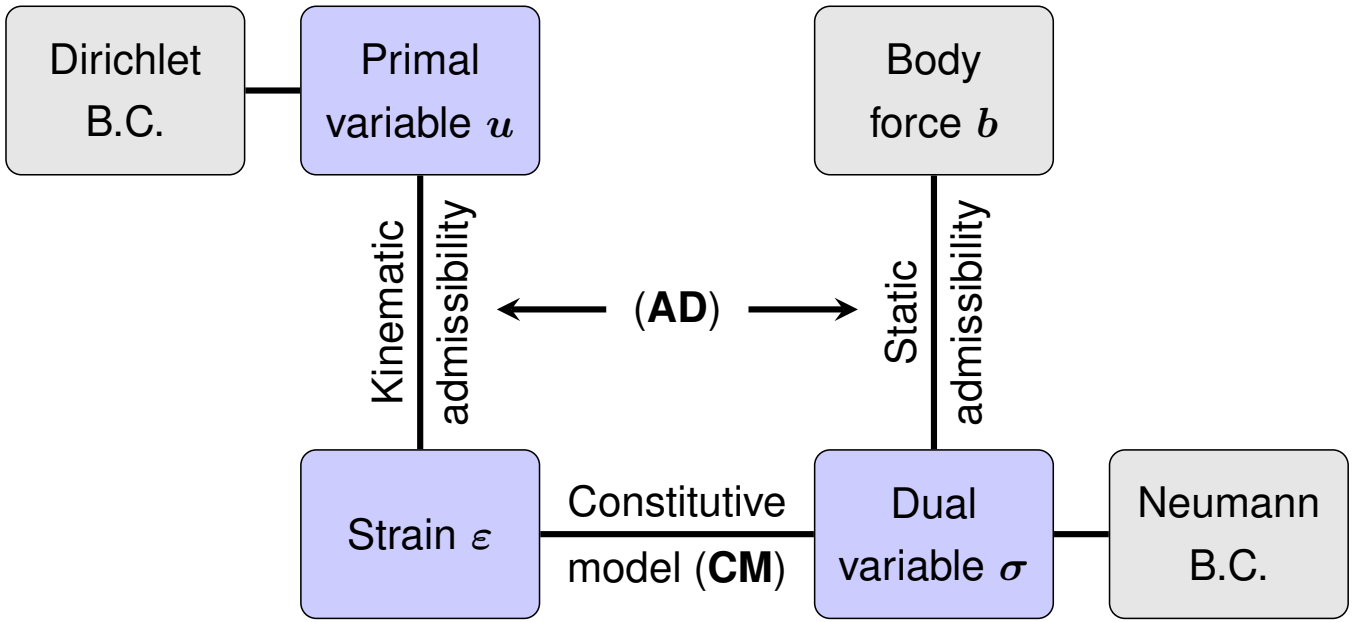
[GSM by Halphen & Nguyen 1975]

LATIN vs. incremental schemes

LATIN: large time increment

An **incremental** scheme:

Solve



for each time step consecutively.

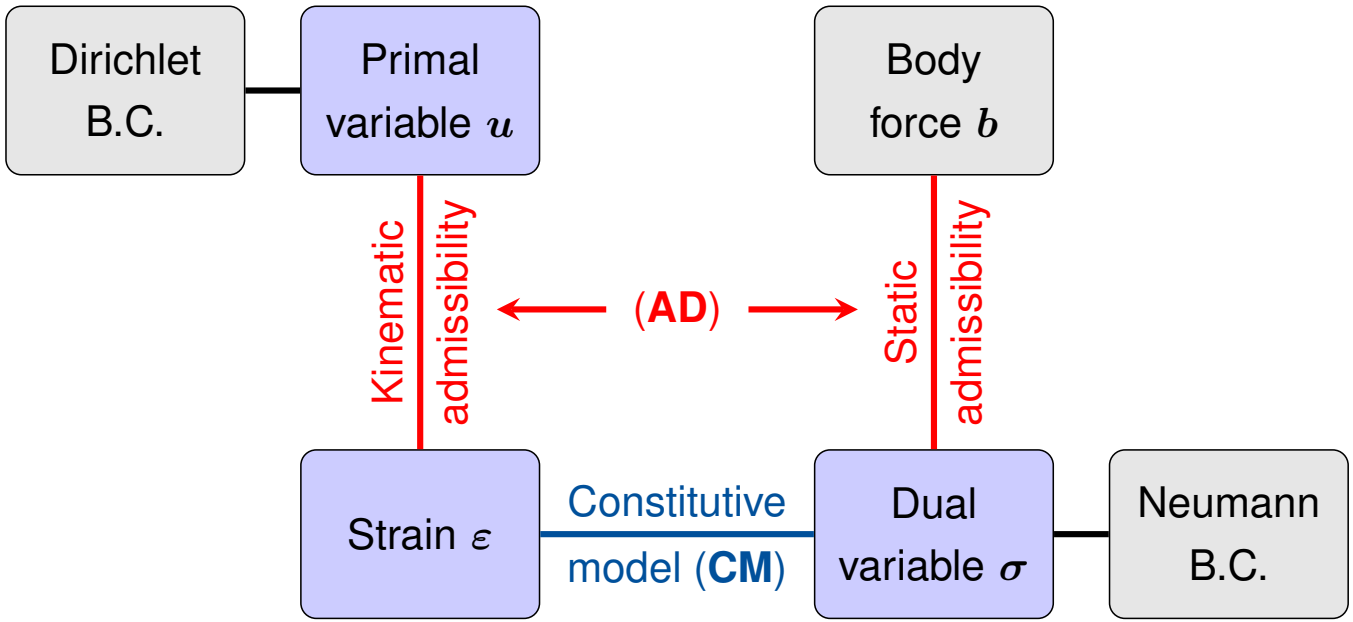
[de Souza Neto 2008]

LATIN vs. incremental schemes

LATIN: large time increment

A **LATIN** scheme:

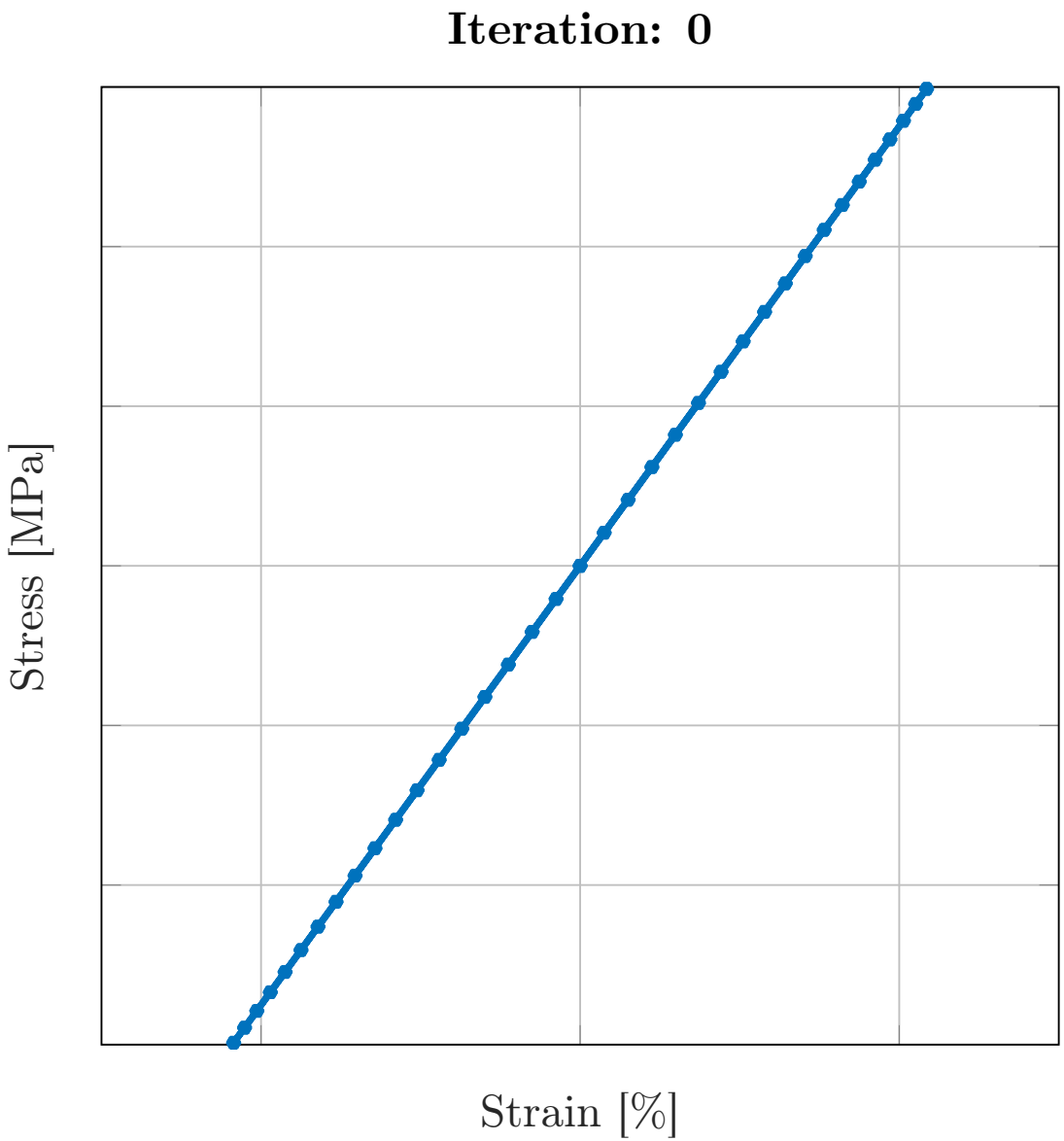
Solve iteratively (**AD**) and (**CM**)



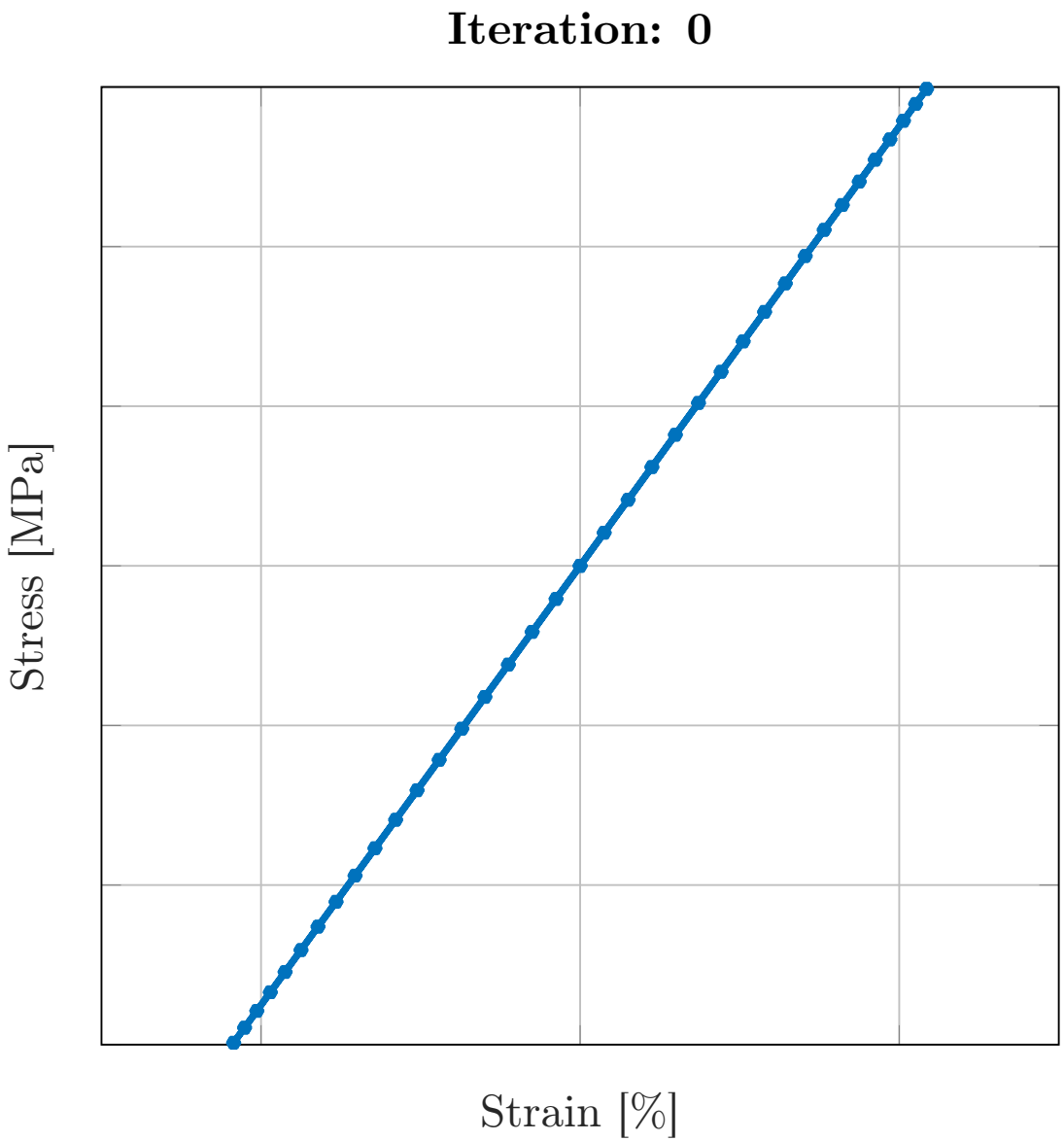
for all time steps simultaneously.

[Ladevèze 1989,1999; Néron et al 2012]

What does simultaneously mean?



What does simultaneously mean?



LATIN solution algorithm

- Elastic initialisation $\{(\boldsymbol{\varepsilon}_i, \mathcal{I}_i), (\boldsymbol{\sigma}_i, \mathcal{F}_i) \mid i = 0\}$
- Evaluate **(CM)** at the local stage

$$\text{given}(\boldsymbol{\sigma}_i, \mathcal{F}_i) \qquad \text{seek}(\boldsymbol{\varepsilon}_{\text{local}}, \mathcal{I}_{\text{local}}, \mathcal{F}_{\text{local}})$$

- Transfer data using affine relation (search direction eq.)

$$(\boldsymbol{\sigma}_{\text{global}} - \boldsymbol{\sigma}_{\text{local}}) - \alpha\, \mathbb{C} : (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}}) = \mathbf{0} \qquad \alpha \in]0, 1]$$

- Solve **(AD)** to get $(\boldsymbol{\sigma}_{\text{global}}, \boldsymbol{\varepsilon}_{\text{global}})$ at the global stage
- Iterate until convergence with an energy error indicator

$$\xi^2 = \frac{1+\alpha^2}{2T} \int_{\Omega \times \mathcal{T}} (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}}) : \mathbb{C} : (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}})\ \mathrm{d}\Omega\, \mathrm{d}t$$

LATIN solution algorithm

- Elastic initialisation $\{(\boldsymbol{\varepsilon}_i, \mathcal{I}_i), (\boldsymbol{\sigma}_i, \mathcal{F}_i) \mid i = 0\}$
- Evaluate **(CM)** at the local stage

$$\text{given}(\boldsymbol{\sigma}_i, \mathcal{F}_i) \qquad \text{seek}(\boldsymbol{\varepsilon}_{\text{local}}, \mathcal{I}_{\text{local}}, \mathcal{F}_{\text{local}})$$

- Transfer data using affine relation (search direction eq.)

$$(\boldsymbol{\sigma}_{\text{global}} - \boldsymbol{\sigma}_{\text{local}}) - \alpha\, \mathbb{C} : (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}}) = \mathbf{0} \qquad \alpha \in]0, 1]$$

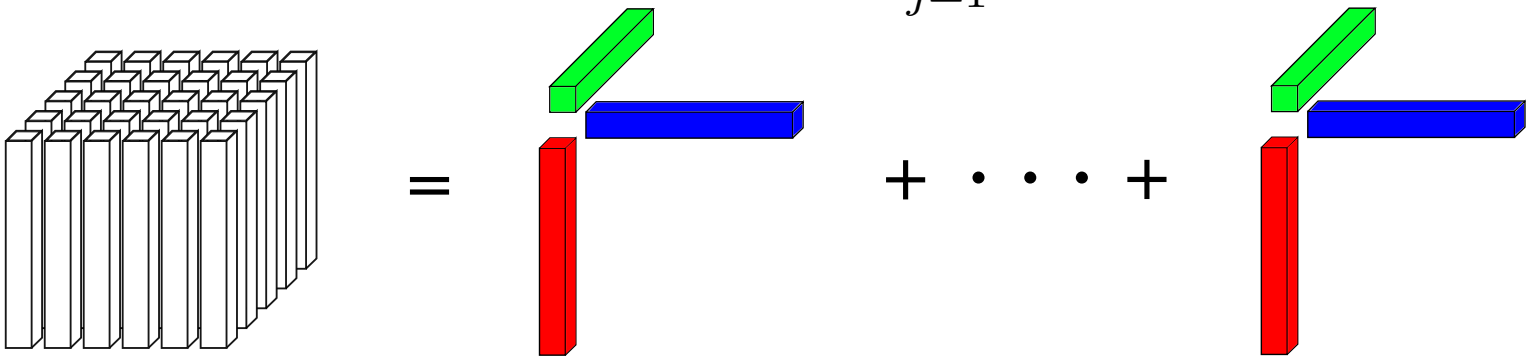
- Solve **(AD)** to get $(\boldsymbol{\sigma}_{\text{global}}, \boldsymbol{\varepsilon}_{\text{global}})$ at the global stage
- Iterate until convergence with an energy error indicator

$$\xi^2 = \frac{1+\alpha^2}{2T} \int_{\Omega \times \mathcal{T}} (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}}) : \mathbb{C} : (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}})\ \mathrm{d}\Omega\, \mathrm{d}t$$

Proper Generalised Decomposition

as set of linear subspaces

- Low-rank approximation $\boldsymbol{u}(\boldsymbol{x}, t, \bullet) = \sum_{j=1}^{N} \boldsymbol{v}_j(\boldsymbol{x}) \circ \boldsymbol{\lambda}_j(t) \circ \boldsymbol{\gamma}(\bullet)$



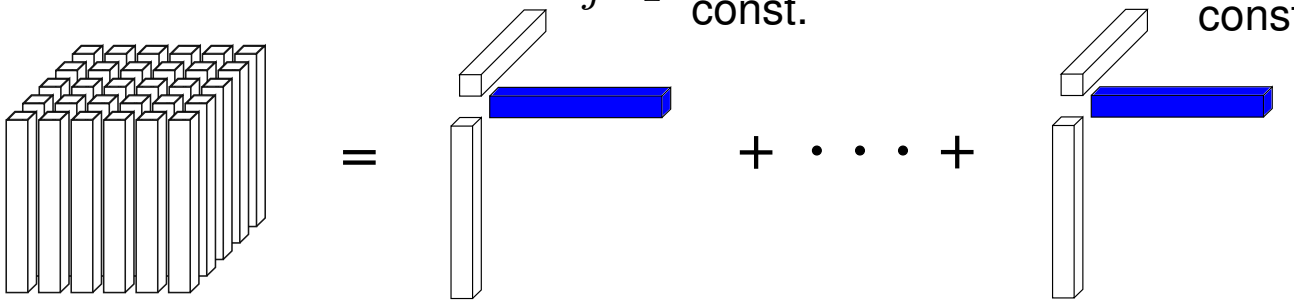
[Chinesta and Ladevèze 2014]

Proper Generalised Decomposition

as set of linear subspaces

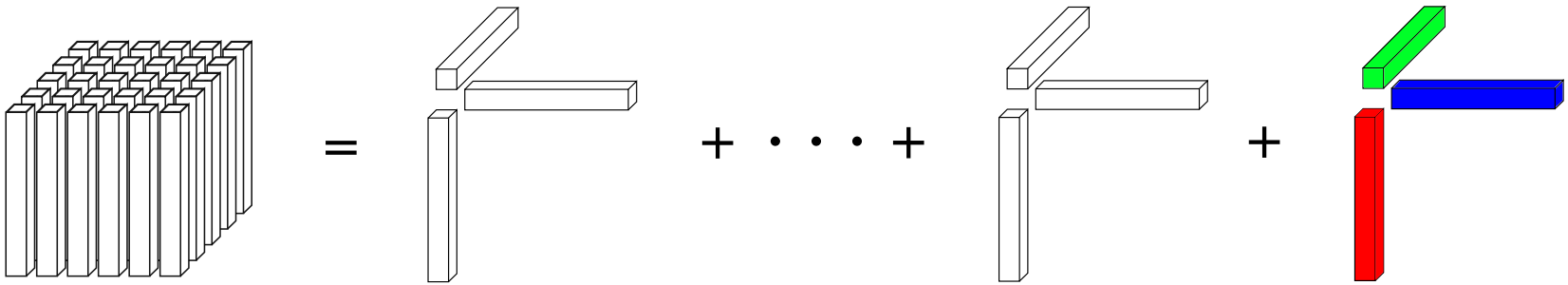
- Update of μ previously generated modes

$$\Delta \boldsymbol{u}_{i+1}(\boldsymbol{x}, t, \bullet) = \sum_{j=1}^{\mu} \underbrace{\boldsymbol{v}_j(\boldsymbol{x})}_{\text{const.}} \circ \Delta \boldsymbol{\lambda}_j(t) \circ \underbrace{\boldsymbol{\gamma}_j(\bullet)}_{\text{const.}}$$



- Enrichment to μ previously generated modes

$$\Delta \boldsymbol{u}_{i+1}(\boldsymbol{x}, t, \bullet) = \boldsymbol{v}_{\mu+1}(\boldsymbol{x}) \circ \boldsymbol{\lambda}_{\mu+1}(t) \circ \boldsymbol{\gamma}_{\mu+1}(\bullet)$$



Alternated directions algorithm

Galerkin finite element discretisation

- Temporal update step

$$\underline{\underline{\tilde{A}}}\,\underline{\underline{\tilde{\Lambda}}}^{\mathsf{T}} = \underline{\underline{\tilde{B}}} \qquad \underline{\underline{\tilde{A}}} \in \mathbb{R}^{\mu \times \mu} \qquad \underline{\underline{\tilde{B}}} \in \mathbb{R}^{\mu \times n_t}$$

$$\underline{\underline{\tilde{\Lambda}}} = [\Delta\underline{\lambda}_1, \cdots, \Delta\underline{\lambda}_\mu]$$

- Spatial problem, with homogeneous boundary conditions

$$\underline{\underline{K}}\,\underline{v} = \underline{f} \qquad \underline{\underline{K}} \in \mathbb{R}^{n \times n} \qquad \underline{v}, \underline{f} \in \mathbb{R}^{n}$$

Temporal problem, with zero initial conditions

$$a\,\underline{\lambda} = \underline{b} \qquad a \in \mathbb{R} \qquad \underline{\lambda}, \underline{b} \in \mathbb{R}^{n_t}$$

Outline

Challenges in a PGD framework

- Efficient variable amplitude and frequency scenarios

$$\int_{\Omega} \bullet \, \mathrm{d}\Omega \qquad \int_{\mathcal{T}} \bullet \, \mathrm{d}t$$

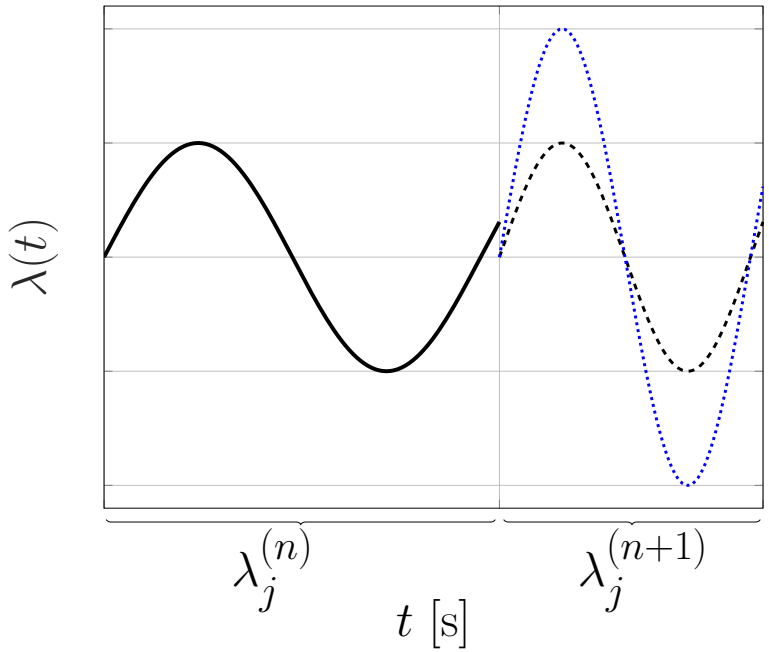
- The fast increase in the number of modes

$$\boldsymbol{u}(\boldsymbol{x}, t) = \sum_{j=1}^{\mu} \boldsymbol{v}_j(\boldsymbol{x}) \circ \boldsymbol{\lambda}_j(t) \text{ with large } \mu$$

- The evaluation of the error indicator and the local stage

Semi-incremental extension

- The temporal domain is divided into cycles
- Cycles are simulated consecutively



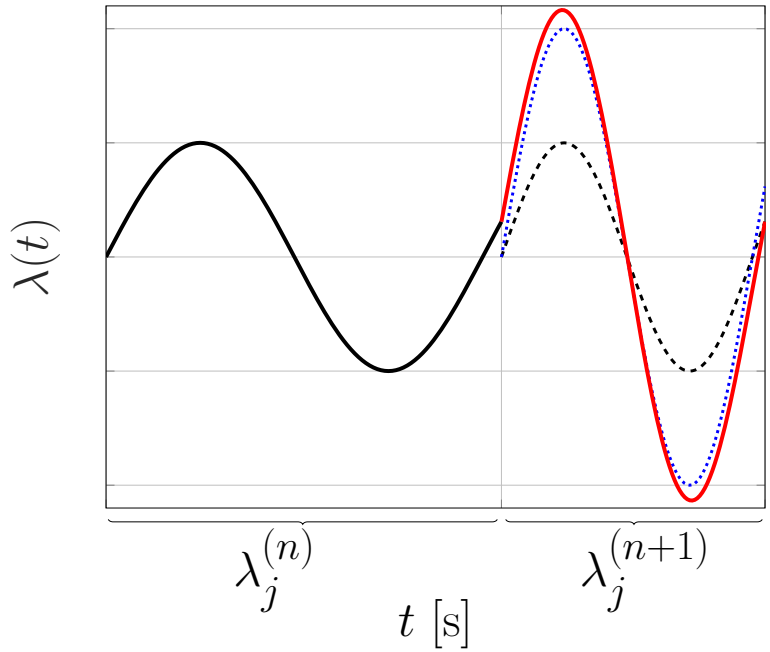
- Already generated $\{\lambda_j(t)\}_{j=1}^{\mu}$ are scaled to $\tilde{t} \in [0, 1]$
- Scaled back to the temporal coordinate of the current cycle

[Bhattacharyya et al 2018; Fritzen et al 2018]

Semi-incremental extension

- Continuity of the temporal modes is an issue
- Temporal modes are vertically scaled and shifted via

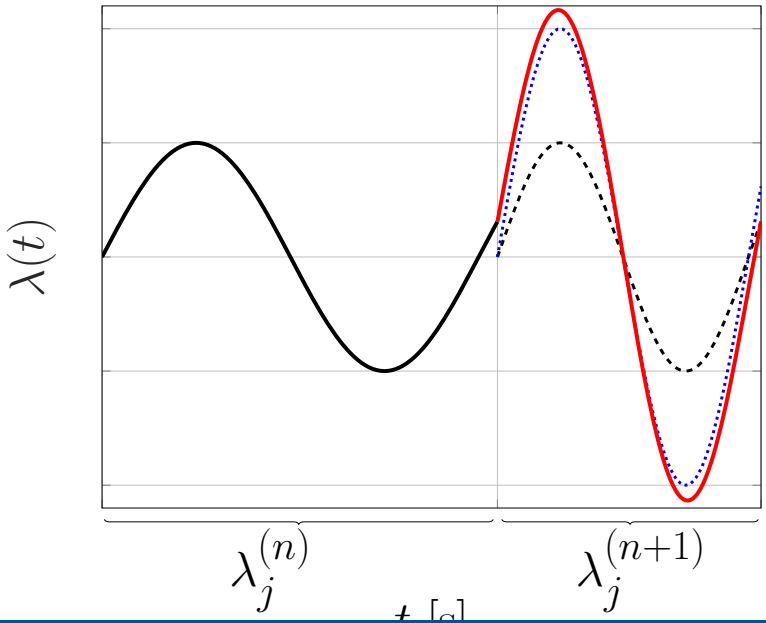
$$\tilde{\lambda}_j^{(n+1)}(\tilde{t}) = m\,\tilde{\lambda}_j^{(n)}(\tilde{t}) + g\,\tilde{t} + h \quad \text{with I.C. and B.C.}$$



Semi-incremental extension

- Continuity of the temporal modes is an issue
- Temporal modes are vertically scaled and shifted via

$$\tilde{\lambda}_j^{(n+1)}(\tilde{t}) = m\,\tilde{\lambda}_j^{(n)}(\tilde{t}) + g\,\tilde{t} + h \quad \text{with I.C. and B.C.}$$



Variable loading / integration over confined domains 

SVD Compression of PGD

- Large number of modes
- Slow temporal update step

$$\underline{\underline{\tilde{A}}}\,\underline{\underline{\tilde{\Lambda}}}^{\mathsf{T}} = \underline{\underline{\tilde{B}}}$$

$$\underline{\underline{\tilde{A}}} \in \mathbb{R}^{\mu\times\mu} \qquad \underline{\underline{\tilde{\Lambda}}} = [\Delta\underline{\lambda}_1, \cdots, \Delta\underline{\lambda}_\mu] \qquad \underline{\underline{\tilde{B}}} \in \mathbb{R}^{\mu\times n_t}$$

- Enrichment with a Gram-Schmidt scheme is not optimal
- SVD is optimal but expensive to compute

SVD Compression of PGD

- Given a solution $\underline{\underline{U}} = \underline{\underline{V}}\ \underline{\underline{\Lambda}}^{\mathsf{T}} \in \mathbb{R}^{n \times n_t}$ with

$$\underline{\underline{V}} = [\underline{v}_1, \cdots, \underline{v}_\mu] \in \mathbb{R}^{n \times \mu} \qquad \underline{\underline{\Lambda}} = [\underline{\lambda}_1, \cdots, \underline{\lambda}_\mu] \in \mathbb{R}^{n_t \times \mu}$$

- Exploit the outer product format of PGD

$$\underline{\underline{V}} = \underline{\underline{Q}}_v\ \underline{\underline{R}}_v \qquad \underline{\underline{\Lambda}} = \underline{\underline{Q}}_\lambda\ \underline{\underline{R}}_\lambda$$

- Compute SVD of a small matrix

$$\underline{\underline{T}} = \underline{\underline{R}}_v\ \underline{\underline{R}}_\lambda^{\mathsf{T}} \in \mathbb{R}^{\mu \times \mu} \qquad \underline{\underline{T}} \approx \underline{\underline{\Theta}}\ \underline{\underline{S}}\ \underline{\underline{W}}^{\mathsf{T}}$$

- Solution with a compressed basis

$$\underline{\underline{U}} \approx \underline{\underline{Q}}_v\ \underline{\underline{\Theta}}\quad \underline{\underline{S}}\quad \underline{\underline{W}}^{\mathsf{T}}\underline{\underline{Q}}_\lambda^{\mathsf{T}}$$

[Bebendorf 2008]

SVD Compression of PGD

- Given a solution $\underline{\underline{U}} = \underline{\underline{V}}\ \underline{\underline{\Lambda}}^{\mathsf{T}} \in \mathbb{R}^{n \times n_t}$ with

$$\underline{\underline{V}} = [\underline{v}_1, \cdots, \underline{v}_\mu] \in \mathbb{R}^{n \times \mu} \qquad \underline{\underline{\Lambda}} = [\underline{\lambda}_1, \cdots, \underline{\lambda}_\mu] \in \mathbb{R}^{n_t \times \mu}$$

- Exploit the outer product format of PGD

$$\underline{\underline{V}} = \underline{\underline{Q}}_v\ \underline{\underline{R}}_v \qquad \underline{\underline{\Lambda}} = \underline{\underline{Q}}_\lambda\ \underline{\underline{R}}_\lambda$$

- Compute SVD of a small matrix

$$\underline{\underline{T}} = \underline{\underline{R}}_v\ \underline{\underline{R}}_\lambda^{\mathsf{T}} \in \mathbb{R}^{\mu \times \mu} \qquad \underline{\underline{T}} \approx \underline{\underline{\Theta}}\ \underline{\underline{S}}\ \underline{\underline{W}}^{\mathsf{T}}$$

- Solution with a compressed basis

$$\underline{\underline{U}} \approx \underline{\underline{Q}}_v\ \underline{\underline{\Theta}}\ \underline{\underline{S}}\ \underline{\underline{W}}^{\mathsf{T}}\ \underline{\underline{Q}}_\lambda^{\mathsf{T}}$$

Non-demanding optimal decomposition

[Bebendorf 2008]

What is left?

- $\boldsymbol{\varepsilon}_{\text{global}}$ is approximated using PGD
- $\boldsymbol{\varepsilon}_{\text{local}}$ is NOT!

$$(\boldsymbol{\sigma}_{\text{global}} - \boldsymbol{\sigma}_{\text{local}}) - \alpha\, \mathbb{C} : (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}}) = \mathbf{0}$$

- Low-rank approximation of $\boldsymbol{\varepsilon}_{\text{local}}$
- Randomised SVD (RSVD) algorithm

- Dedicated arithmetic toolbox to handle operations on PGD

Stress decomposition

Algebra on Low-rank Matrices

$$\underline{\underline{A}} = \underline{\underline{V}}_A \, \underline{\underline{S}}_A \, \underline{\underline{\Lambda}}_A{}^{\mathsf{T}} \qquad\qquad \underline{\underline{B}} = \underline{\underline{V}}_B \, \underline{\underline{S}}_B \, \underline{\underline{\Lambda}}_B{}^{\mathsf{T}}$$

- Addition

$$\underline{\underline{C}} = \underline{\underline{A}} + \underline{\underline{B}} = \begin{bmatrix} \underline{\underline{V}}_A & \underline{\underline{V}}_B \end{bmatrix} \begin{bmatrix} \underline{\underline{S}}_A & 0 \\ 0 & \underline{\underline{S}}_B \end{bmatrix} \begin{bmatrix} \underline{\underline{\Lambda}}_A & \underline{\underline{\Lambda}}_B \end{bmatrix}^{\mathsf{T}} = \underline{\underline{V}}_C \, \underline{\underline{S}}_C \, \underline{\underline{\Lambda}}_C{}^{\mathsf{T}}$$

- Multiplication

$$\underline{\underline{A}} \, \underline{x} = \underline{\underline{V}}_A \, \underline{\underline{S}}_A \, \left(\underline{\underline{\Lambda}}_A{}^{\mathsf{T}} \, \underline{x} \right)$$

- Deviatoric Operation

$$\underline{\underline{A}}_{\mathrm{dev}} = \underline{\underline{P}}_{\mathrm{dev}} \, \underline{\underline{V}}_A \, \underline{\underline{S}}_A \, \underline{\underline{\Lambda}}_A{}^{\mathsf{T}} = \left(\underline{\underline{P}}_{\mathrm{dev}} \, \underline{\underline{V}}_A \right) \, \underline{\underline{S}}_A \, \underline{\underline{\Lambda}}_A{}^{\mathsf{T}}$$

Stress decomposition

Algebra on Low-rank Matrices

$$\underline{\underline{A}} = \underline{\underline{V}}_A \, \underline{\underline{S}}_A \, \underline{\underline{\Lambda}}_A{}^{\mathsf{T}} \qquad\qquad \underline{\underline{B}} = \underline{\underline{V}}_B \, \underline{\underline{S}}_B \, \underline{\underline{\Lambda}}_B{}^{\mathsf{T}}$$

- Addition

$$\underline{\underline{C}} = \underline{\underline{A}} + \underline{\underline{B}} = \begin{bmatrix} \underline{\underline{V}}_A & \underline{\underline{V}}_B \end{bmatrix} \begin{bmatrix} \underline{\underline{S}}_A & 0 \\ 0 & \underline{\underline{S}}_B \end{bmatrix} \begin{bmatrix} \underline{\underline{\Lambda}}_A & \underline{\underline{\Lambda}}_B \end{bmatrix}^{\mathsf{T}} = \underline{\underline{V}}_C \, \underline{\underline{S}}_C \, \underline{\underline{\Lambda}}_C{}^{\mathsf{T}}$$

- Multiplication

$$\underline{\underline{A}} \, \underline{x} = \underline{\underline{V}}_A \, \underline{\underline{S}}_A \, \left(\underline{\underline{\Lambda}}_A{}^{\mathsf{T}} \, \underline{x} \right)$$

- Deviatoric Operation

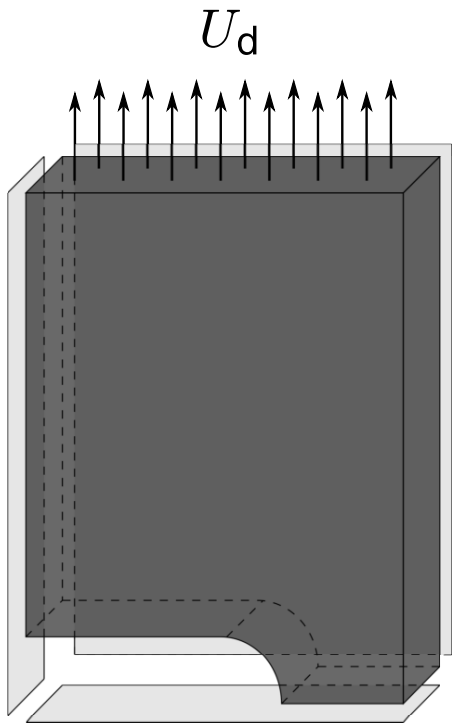
$$\underline{\underline{A}}_{\mathrm{dev}} = \underline{\underline{P}}_{\mathrm{dev}} \, \underline{\underline{V}}_A \, \underline{\underline{S}}_A \, \underline{\underline{\Lambda}}_A{}^{\mathsf{T}} = \left(\underline{\underline{P}}_{\mathrm{dev}} \, \underline{\underline{V}}_A \right) \, \underline{\underline{S}}_A \, \underline{\underline{\Lambda}}_A{}^{\mathsf{T}}$$

Most operations exploit the PGD decomposition

Outline

Numerical results

- A plate subjected to cyclic loading (Cr-Mo steel at $580^{\circ}\mathrm{C}$)



Viscoplastic viscodamage material

- Stress–strain

$$\boldsymbol{\sigma} = (1 - D)\, \mathbb{C} : \boldsymbol{\varepsilon}^{\mathrm{e}} \qquad \dot{\boldsymbol{\varepsilon}}^{\mathrm{p}} = \dot{\lambda}\, \boldsymbol{N}$$

- Kinematic hardening

$$\boldsymbol{\beta} = \frac{2}{3}\, c\, \boldsymbol{\alpha} \qquad \dot{\boldsymbol{\alpha}} = \dot{\lambda} \left(\tilde{\boldsymbol{N}} - \frac{3\, a}{2\, c}\, \boldsymbol{\beta} \right)$$

- Isotropic hardening

$$R = R_{\infty}\, (1 - \mathrm{e}^{-b\, r}) \qquad \dot{r} = \dot{\lambda}$$

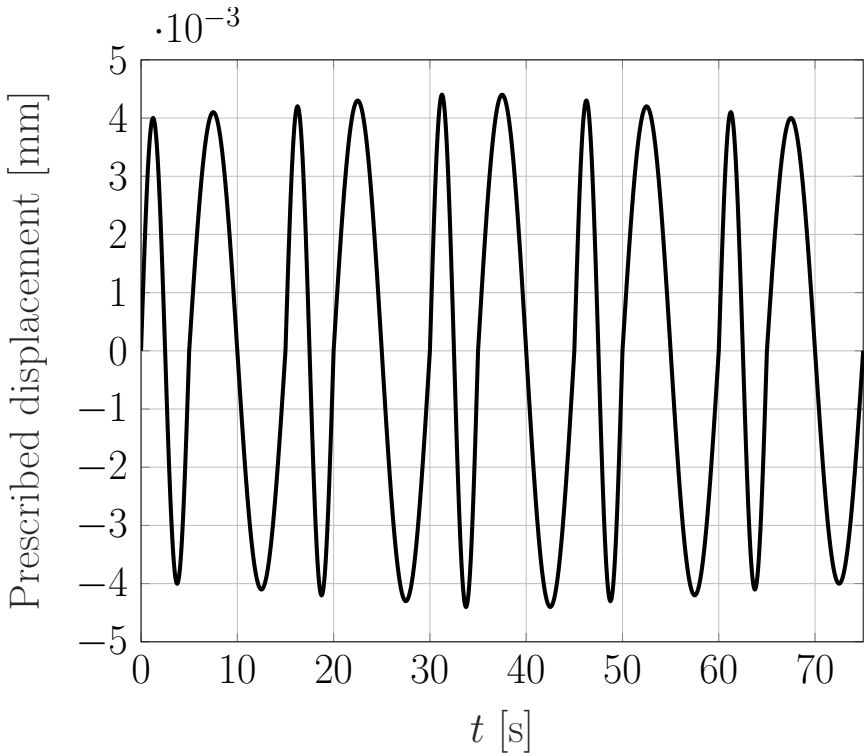
- Damage

$$Y = \frac{1}{2}\, \boldsymbol{\varepsilon}^{\mathrm{e}} : \mathbb{C} : \boldsymbol{\varepsilon}^{\mathrm{e}} \qquad \dot{D} = \frac{\dot{\lambda}}{1 - D} \left(\frac{Y}{S} \right)^{s}$$

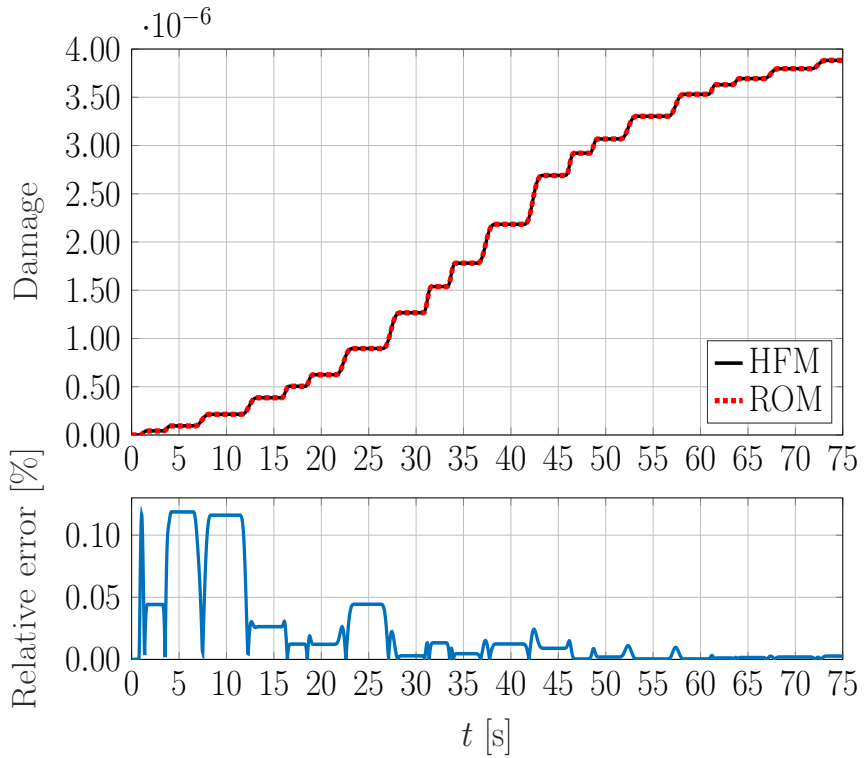
Model verification

$1884 \cdot 200 \cdot 10$ DOF

With respect to a modified Newton-Raphson scheme



Prescribed displacement

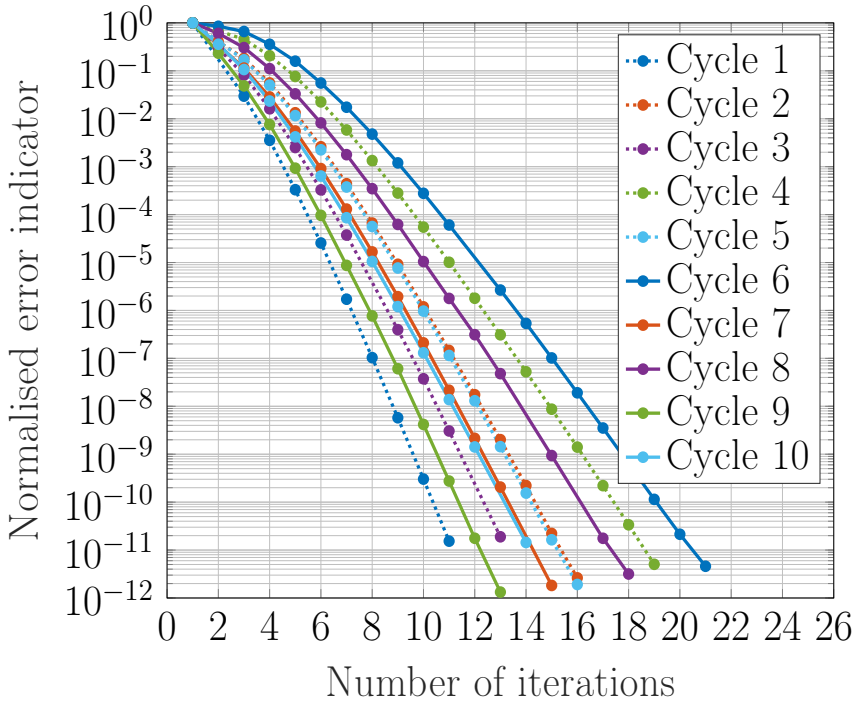


Damage evolution

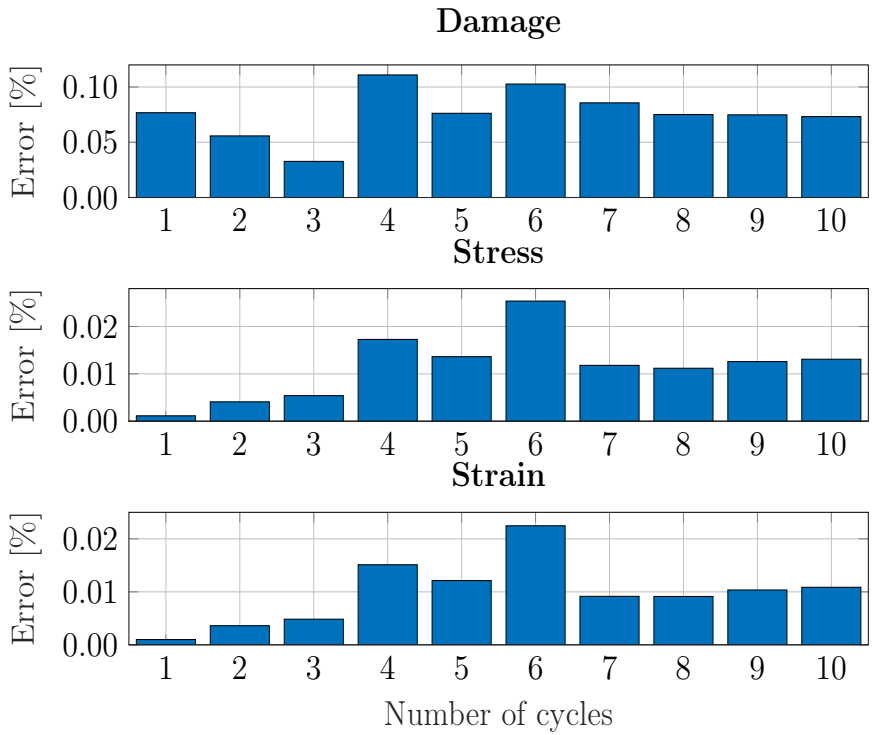
Model verification

$$\|\boldsymbol{e}\|^2_{\Omega\times\mathcal{T}} = \frac{1}{T\,|\Omega|}\int_{\mathcal{T}}\int_{\Omega} \boldsymbol{e}:\boldsymbol{e}\;\mathrm{d}\Omega\,\mathrm{d}t$$

(norm in Bochner space)



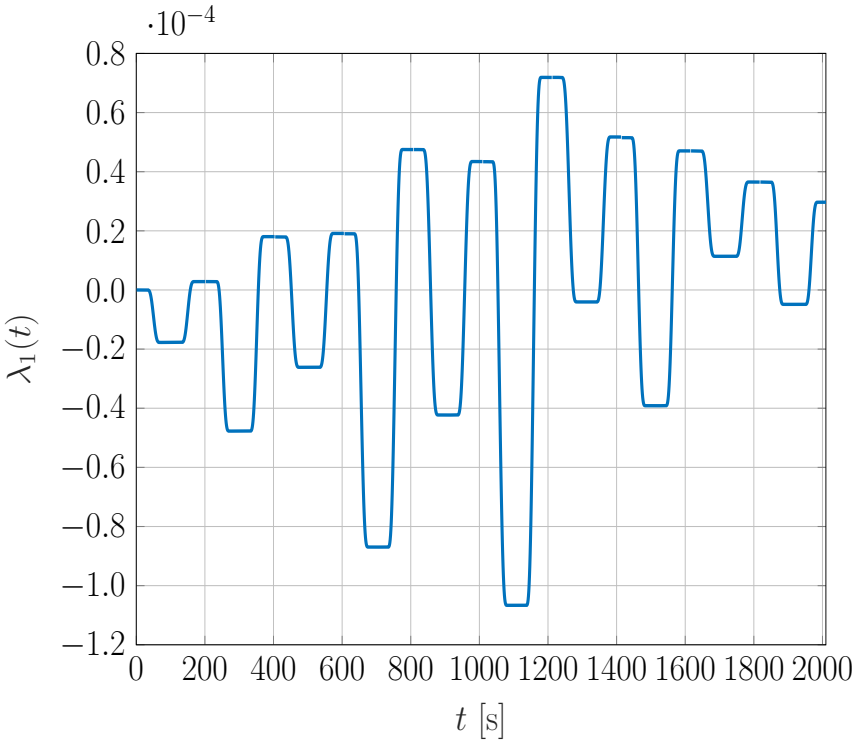
Error indicator

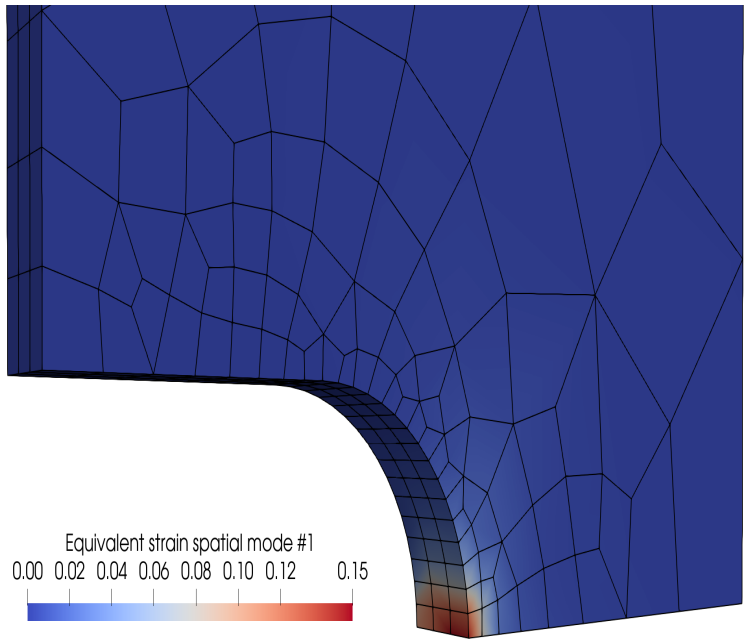


Space-time average relative error

Model verification

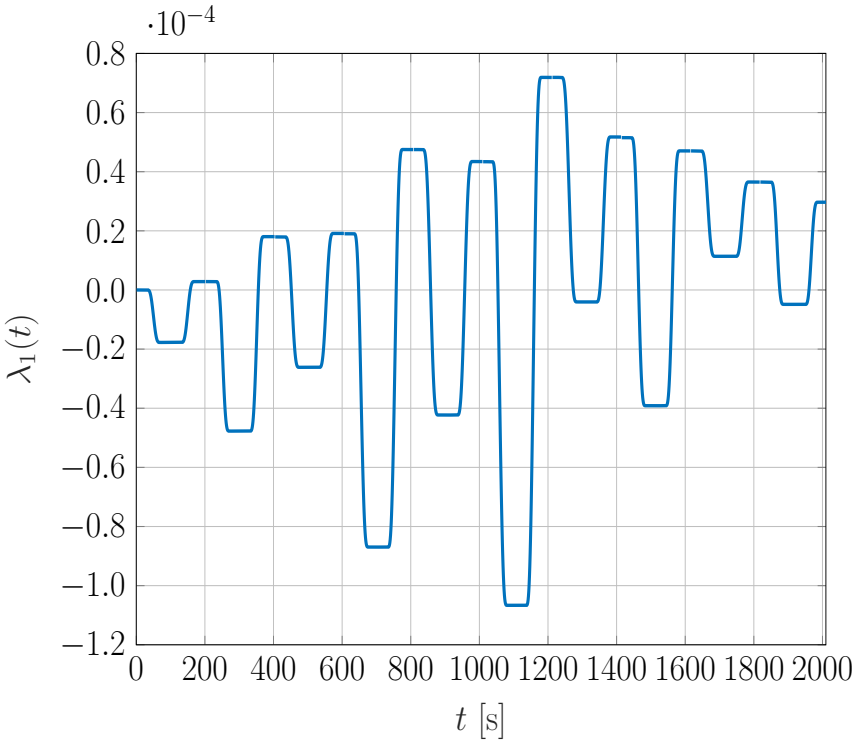
The first temporal and spatial modes

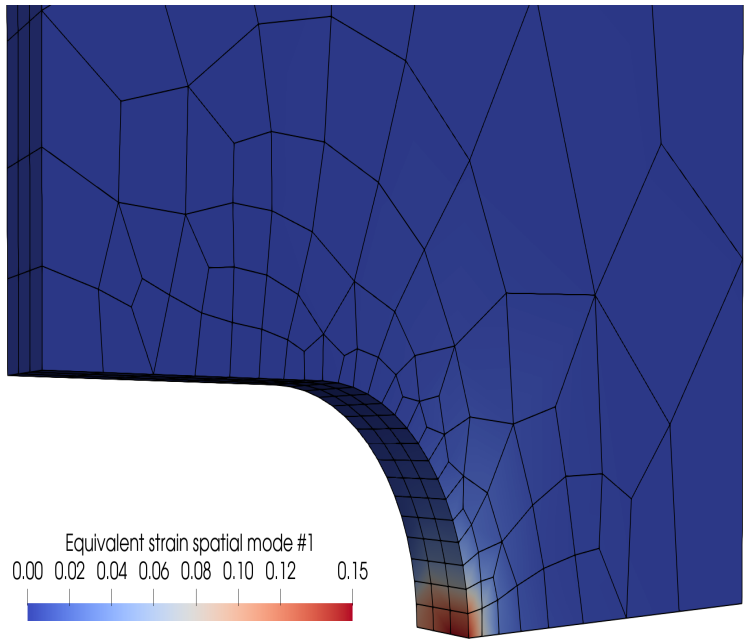




Model verification

The first temporal and spatial modes



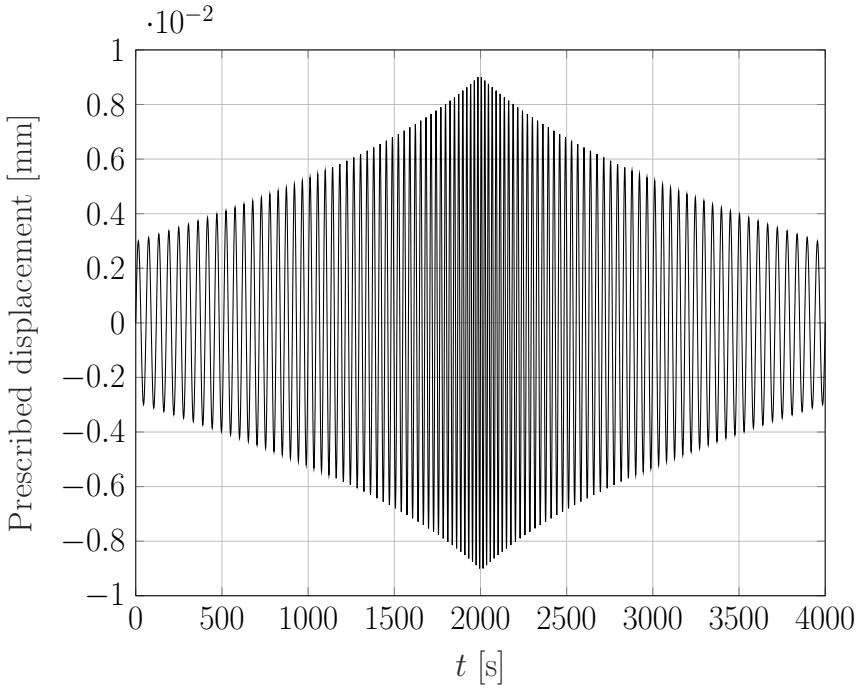


12 modes / Speedup factor ~ 25

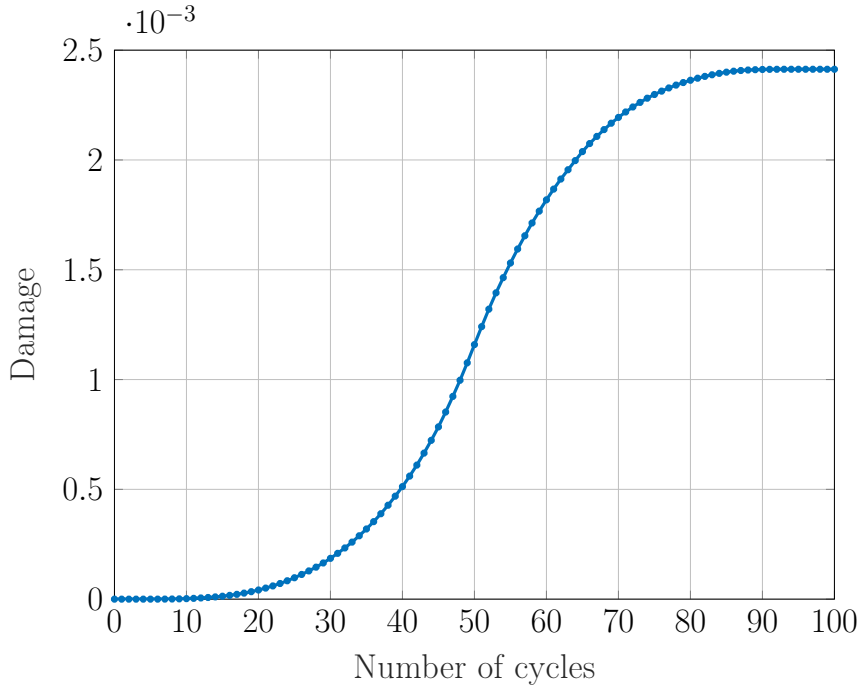
Variable amplitudes & frequencies

Amplitudes: $[30, 90] \cdot 10^{-4}\,\mathrm{mm}$

Time periods: $[20, 60]\,\mathrm{sec}$



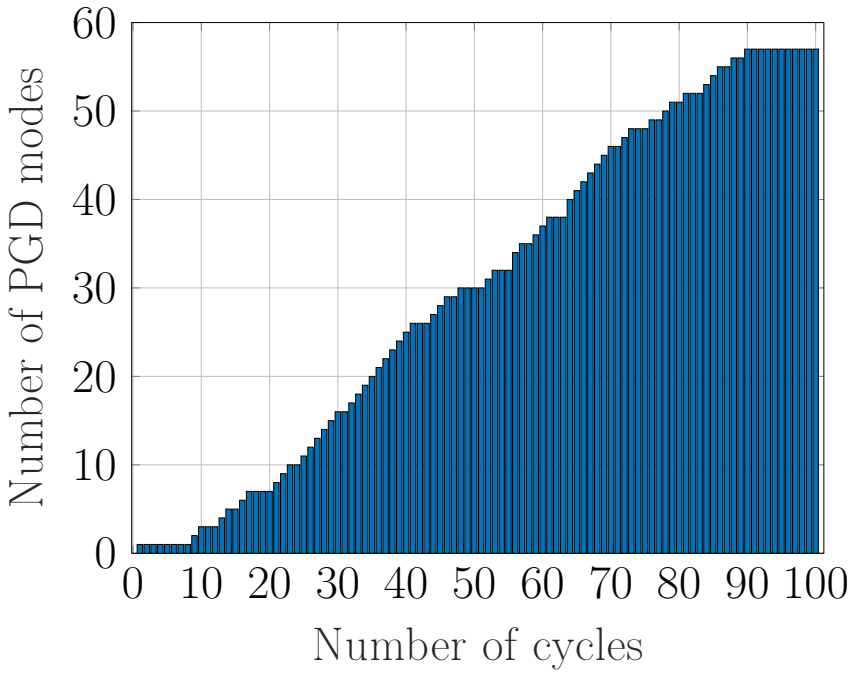
Prescribed displacement



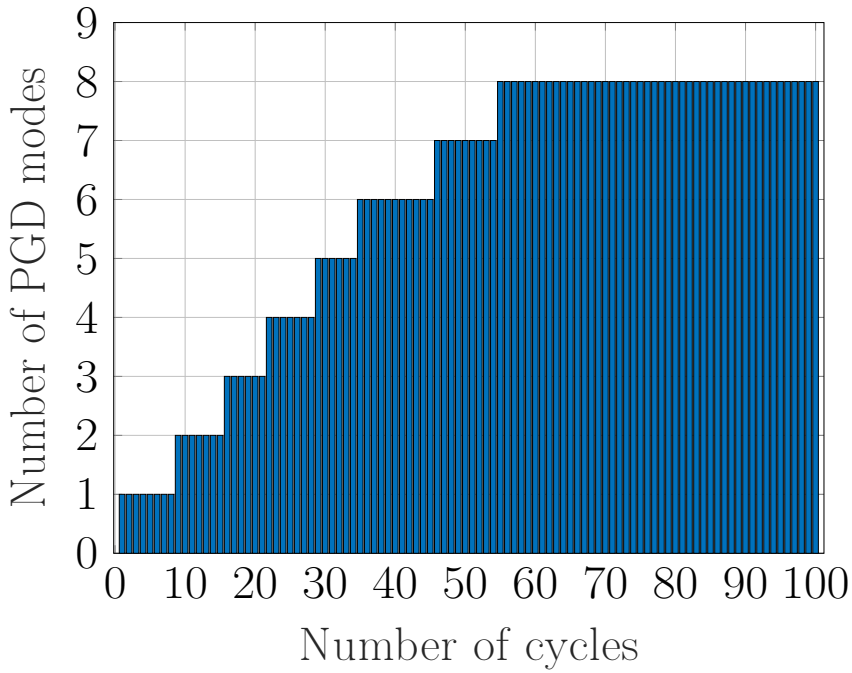
Damage evolution

Variable amplitudes & frequencies

The growth of the ROB using an SVD scheme



Gram-Schmidt

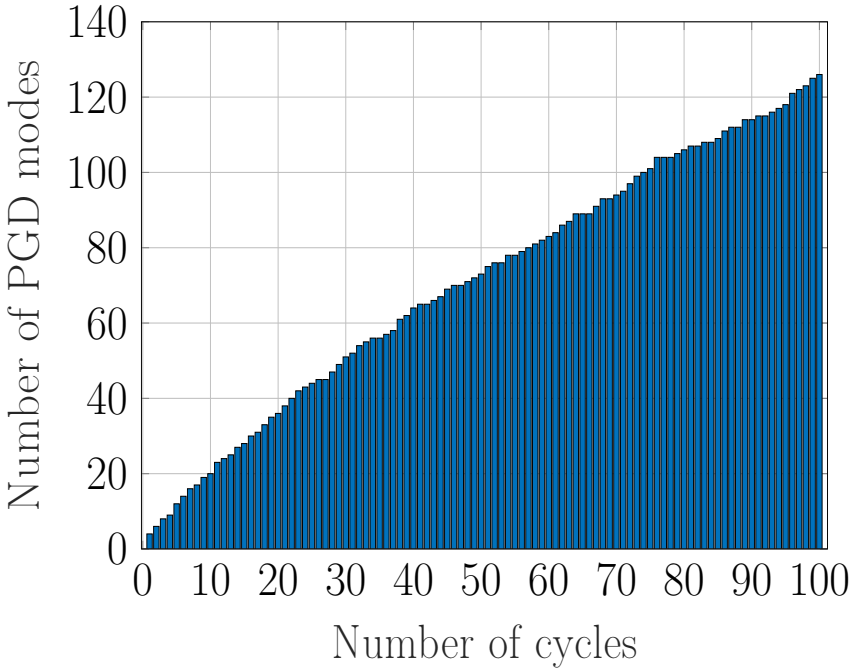


SVD scheme

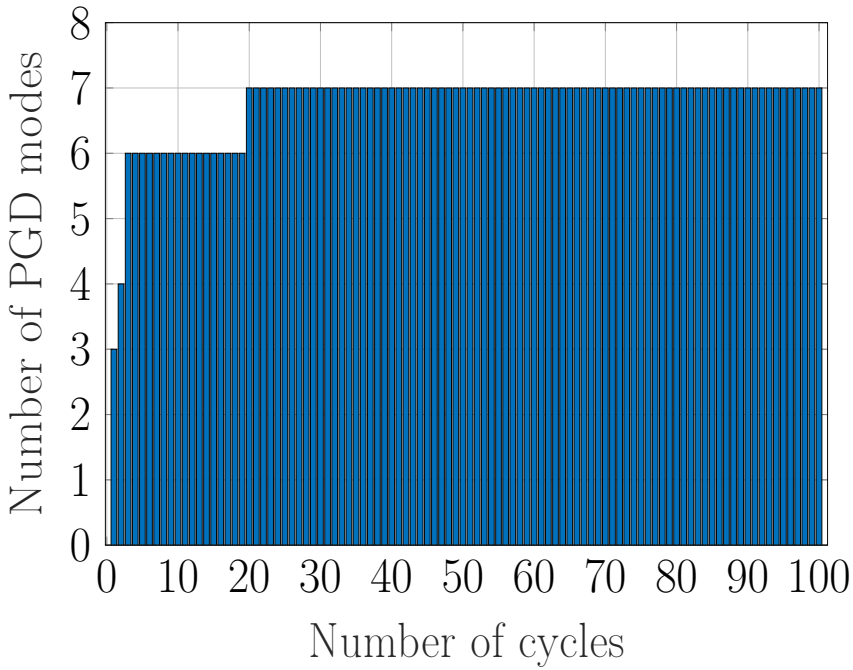
Different ortho. schemes

$50,547 \cdot 33 \cdot 100$ DOF

In case of random loading and fine discretisation



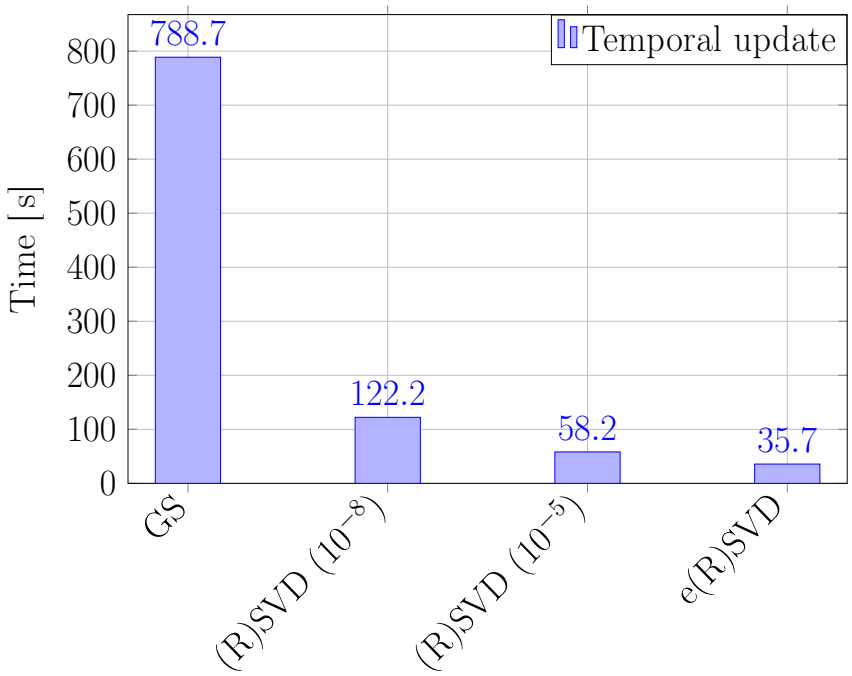
Gram-Schmidt



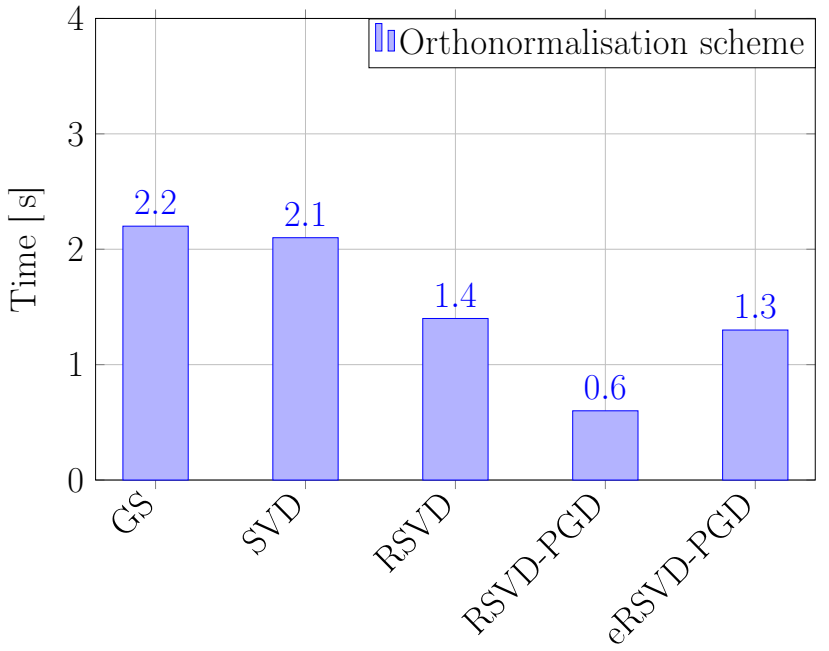
SVD scheme

Different ortho. schemes

The required time to update and orthonormalise the modes



Temporal update

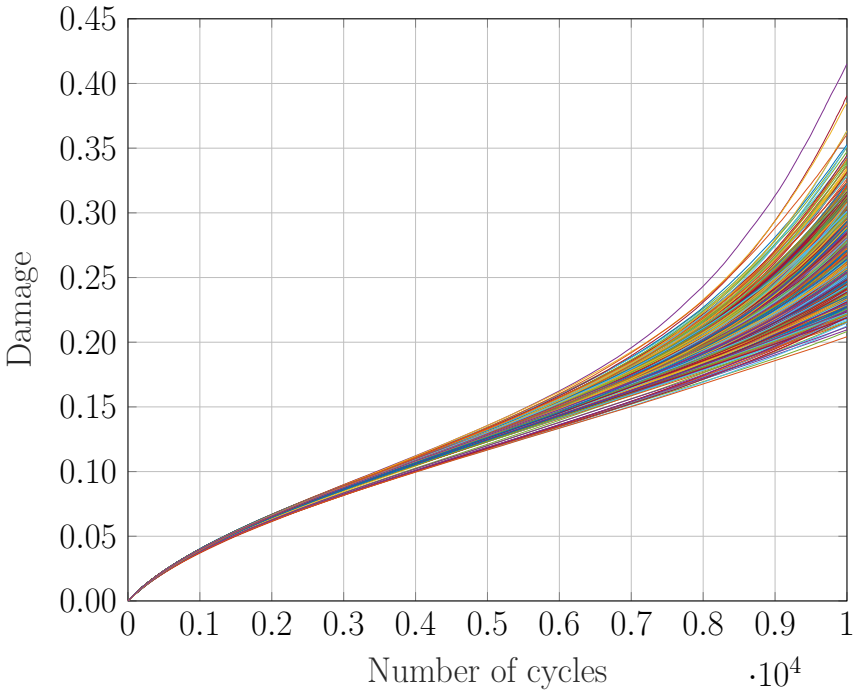


Orthonormalisation step

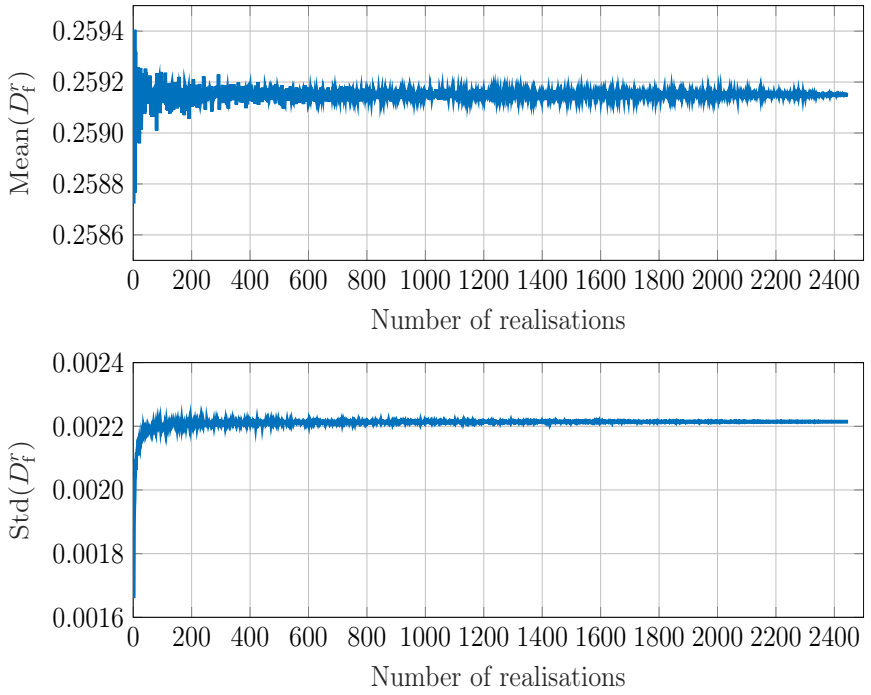
Random amplitude loading

$1884 \cdot 41 \cdot 10^4$ DOF

- 10^4 cycles, uniform distribution in $[53, 56] \cdot 10^{-2}\,\mathrm{mm}$



Different damage realisations

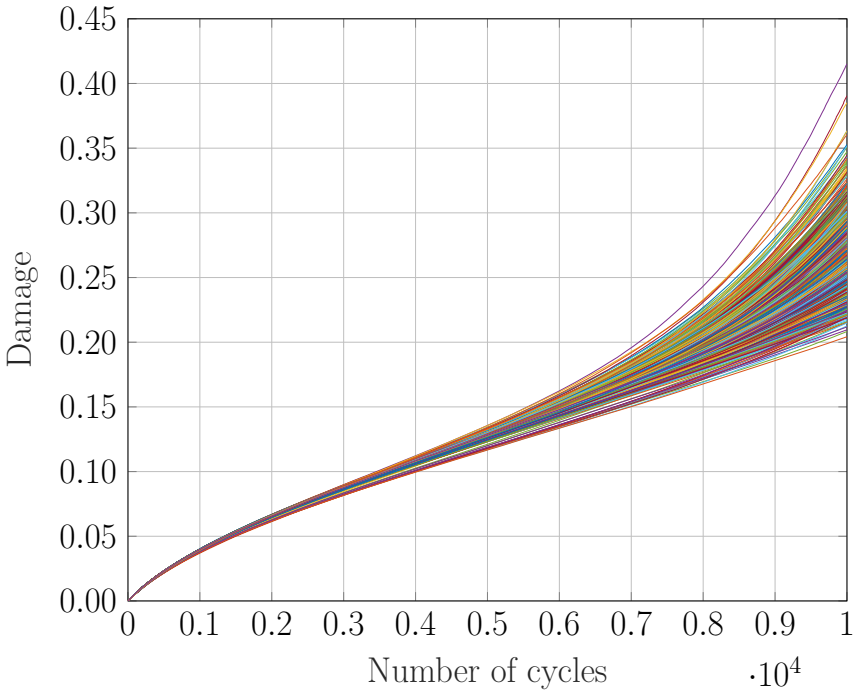


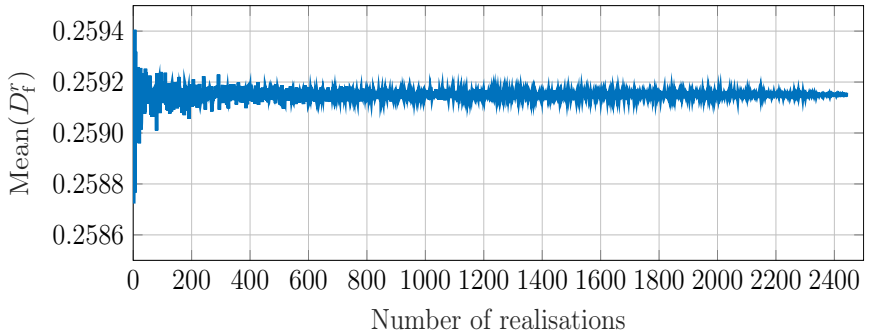
The mean and SD of D_f

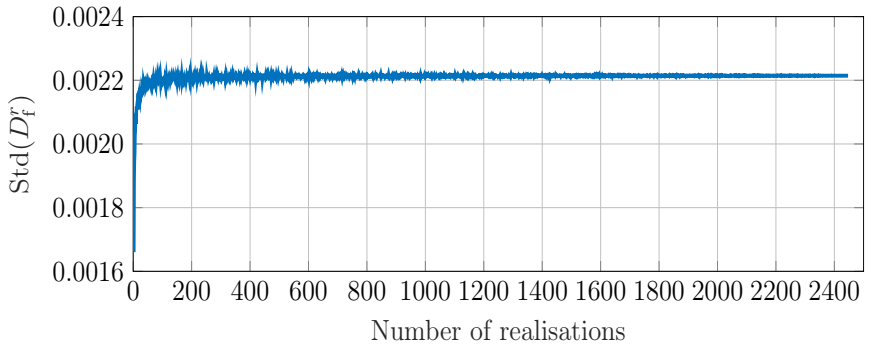
Random amplitude loading

$1884 \cdot 41 \cdot 10^4$ DOF

- 10^4 cycles, uniform distribution in $[53, 56] \cdot 10^{-2}\,\mathrm{mm}$







Critical damage value $D_\mathrm{c} = 0.3$

Probability of failure $P_\mathrm{f} = 5.4\%$

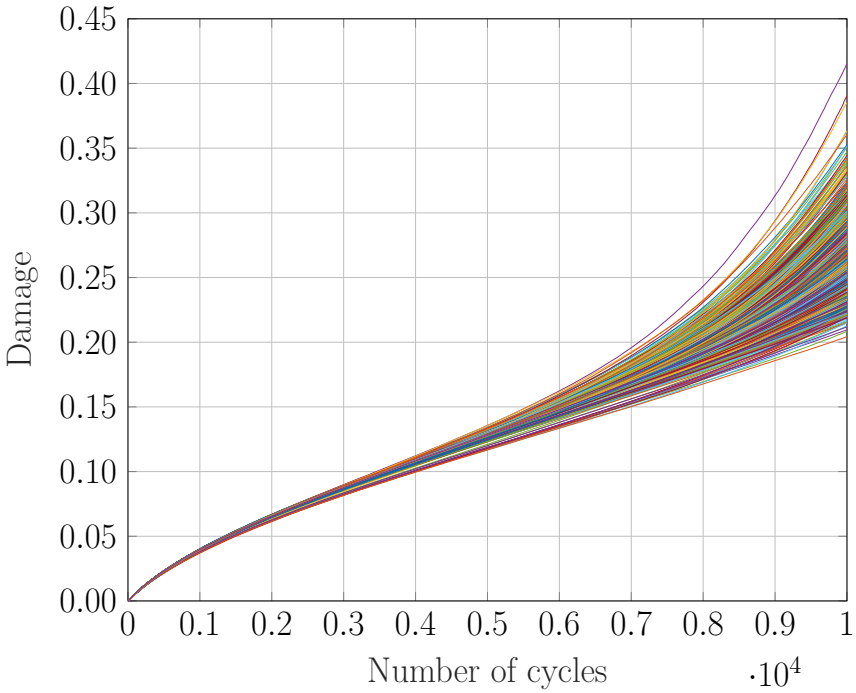
Requirements:

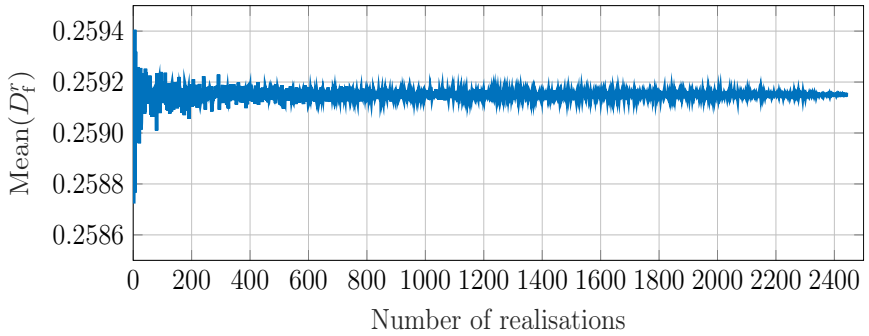
$[15 - 35]\,\mathrm{min}$ and $[1 - 1.5]\,\mathrm{GB}$

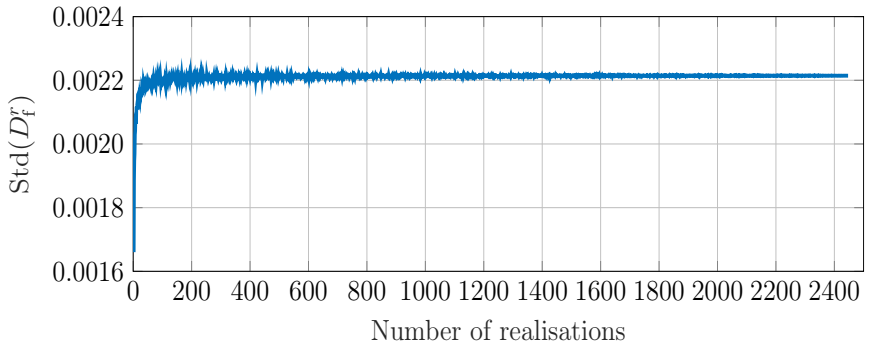
Random amplitude loading

$1884 \cdot 41 \cdot 10^4$ DOF

- 10^4 cycles, uniform distribution in $[53, 56] \cdot 10^{-2}\,\mathrm{mm}$







Critical damage value $D_\mathrm{c} = 0.3$

Probability of failure $P_\mathrm{f} = 5.4\%$

Requirements:

$[15 - 35]\,\mathrm{min}$ and $[1 - 1.5]\,\mathrm{GB}$

Approx. 10 modes / Time-saving factors $50 \sim 100$

Outline

Conclusion and future research

- Efficient semi-incremental scheme for damage problems
- Can handle variable amplitude and frequency loadings
- Provides a minimal expansion of PGD
- Most operations are done over decomposed QoI
- Open-source code: `gitlab.com/shadialameddin/romfem`

Conclusion and future research

- Efficient semi-incremental scheme for damage problems
- Can handle variable amplitude and frequency loadings
- Provides a minimal expansion of PGD
- Most operations are done over decomposed QoI
- Open-source code: `gitlab.com/shadialameddin/romfem`

Current tasks

- Improve the documentation of the code
- Reducing the cost of evaluating the constitutive model

Conclusion and future research

- Efficient semi-incremental scheme for damage problems
- Can handle variable amplitude and frequency loadings
- Provides a minimal expansion of PGD
- Most operations are done over decomposed QoI
- Open-source code: `gitlab.com/shadialameddin/romfem`

Current tasks

- Improve the documentation of the code
- Reducing the cost of evaluating the constitutive model

Thank you for your attention

Alternated directions algorithm

- Spatial problem, with homogeneous boundary conditions:

$$\int_\Omega \nabla^{\mathrm{s}} \boldsymbol{v}^* : \left(\int_{\mathcal{T}} \lambda \, \mathbb{H}^- \, \lambda \, \mathrm{d}t \right) : \nabla^{\mathrm{s}} \boldsymbol{v} \; \mathrm{d}\Omega = - \int_\Omega \nabla^{\mathrm{s}} \boldsymbol{v}^* : \left(\int_{\mathcal{T}} \lambda \, \hat{\boldsymbol{f}} \; \mathrm{d}t \right) \mathrm{d}\Omega$$

$$\underline{\underline{K}} \, \underline{v} = \underline{f} \qquad \underline{\underline{K}} \in \mathbb{R}^{n \times n} \qquad \underline{v}, \underline{f} \in \mathbb{R}^n$$

Temporal problem, with zero initial conditions:

$$\int_{\mathcal{T}} \lambda^* \left(\int_\Omega \nabla^{\mathrm{s}} \boldsymbol{v} : \mathbb{H}^- : \nabla^{\mathrm{s}} \boldsymbol{v} \, \mathrm{d}\Omega \right) \lambda \; \mathrm{d}t = - \int_{\mathcal{T}} \lambda^* \left(\int_\Omega \nabla^{\mathrm{s}} \boldsymbol{v} : \hat{\boldsymbol{f}} \; \mathrm{d}\Omega \right) \mathrm{d}t$$

$$a \, \underline{\lambda} = \underline{b} \qquad a \in \mathbb{R} \qquad \underline{\lambda}, \underline{b} \in \mathbb{R}^{n_t}$$

Alternated directions algorithm

- Temporal update step

$$\underline{\underline{\tilde{A}}}\,\underline{\underline{\tilde{\Lambda}}}^{\mathsf{T}} = \underline{\underline{\tilde{B}}} \qquad \underline{\underline{\tilde{A}}} \in \mathbb{R}^{\mu\times\mu} \qquad \underline{\underline{\tilde{B}}} \in \mathbb{R}^{\mu\times n_t}$$

$$\underline{\underline{\tilde{\Lambda}}} = [\Delta\underline{\lambda}_1, \cdots, \Delta\underline{\lambda}_\mu]$$

$$\tilde{A}_{kl} = \alpha \int_\Omega (\underline{\underline{B}}\ \underline{v}_k)^{\mathsf{T}} \underline{\underline{C}}\ (\underline{\underline{B}}\ \underline{v}_l)\ \mathrm{d}\Omega$$

$$\tilde{B}_{kl} = -\int_\Omega (\underline{\underline{B}}\ \underline{v}_k)^{\mathsf{T}} \underline{\hat{\boldsymbol{f}}}_{|t=t_l}\ \mathrm{d}\Omega$$

$$\psi(\boldsymbol{\varepsilon}^{\mathrm{e}}, \boldsymbol{\alpha}, r, D) = \psi^{\mathrm{eD}}(\boldsymbol{\varepsilon}^{\mathrm{e}}, D) + \psi^{\mathrm{p}}(\boldsymbol{\alpha}, r)$$

$$\psi = \frac{1}{2}\,\boldsymbol{\varepsilon}^{\mathrm{e}} : (1-D)\,\mathbb{C} : \boldsymbol{\varepsilon}^{\mathrm{e}} + \frac{1}{3}\,c\,\boldsymbol{\alpha} : \boldsymbol{\alpha} + R_{\infty}\left(r + \frac{\mathrm{e}^{-b\,r}}{b}\right)$$

$$\phi^{*}(\boldsymbol{\sigma}, \boldsymbol{\beta}, R) = \phi^{\mathrm{p}} + \phi^{\beta}$$

$$\phi^{\mathrm{p}} = \frac{k_{\mathrm{p}}}{1+n_{\mathrm{p}}}\left\langle \frac{f^{\mathrm{p}}}{k_{\mathrm{p}}} \right\rangle^{n_{\mathrm{p}}+1} \qquad f^{\mathrm{p}} = \sqrt{3J_2(\boldsymbol{\tau})} - R - \sigma_{\mathrm{y}}$$

$$\phi^{\beta} = \frac{3}{4}\frac{a}{c}\,\boldsymbol{\beta} : \boldsymbol{\beta}$$

$$\phi^{\mathrm{d}} = \frac{S}{(s+1)(1-D)}\left(\frac{Y}{S}\right)^{s+1}$$

LATIN solution algorithm

- Start with an elastic initialisation $s_0 = \{\boldsymbol{\sigma}_0, \boldsymbol{Q}_0, \boldsymbol{\varepsilon}_0, \boldsymbol{q}_0\}$
- Evaluate **(CM)** to get s_{local} (local stage)
- Solve **(AD)** to get s_{global} (global stage)
- Transfer data using affine relations (search direction eq.)

$$(\boldsymbol{\sigma}_{\text{global}} - \boldsymbol{\sigma}_{\text{local}}) - \mathbb{H}^{\mp} : (\boldsymbol{\varepsilon}_{\text{global}} - \boldsymbol{\varepsilon}_{\text{local}}) = \mathbf{0}$$

- Iterate until convergence with an energy error indicator

$$\xi = \frac{\|s_{\text{global}} - s_{\text{local}}\|}{\frac{1}{2}\|s_{\text{global}} + s_{\text{local}}\|}, \quad \|s\|^2 = \frac{1}{2T}\int_{\Omega\times\mathcal{T}} (\boldsymbol{\sigma} : \mathbb{C}^{-1} : \boldsymbol{\sigma} + \boldsymbol{\varepsilon} : \mathbb{C} : \boldsymbol{\varepsilon}) \, \mathrm{d}\Omega \, \mathrm{d}t.$$