A Semi-incremental Scheme for Fatigue Damage Computations

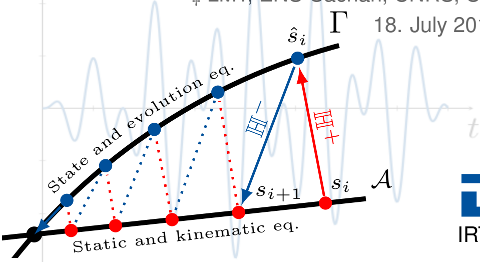
S. Alameddin[†] , A. Fau[†],

U. Nackenhorst[†], D. Néron[‡], P. Ladevèze[‡]

† IBNM, Leibniz Universität Hannover

‡ LMT, ENS Cachan, CNRS, Université Paris Saclay

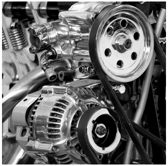
18. July 2019



Deutsche Forschungsgemeinschaft
IRTG-1627

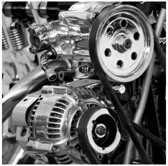
Fatigue damage

Fluctuating loads


Fatigue damage

Fluctuating loads



Material degradation

lower load carrying capacity

Large number of load cycles

thousands or millions

Continuum damage model

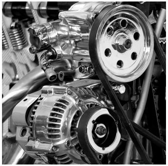
$\dot{D} = \mathrm{d}D/\,\mathrm{d}t$

Computational expense

memory and time

Fatigue damage

Fluctuating loads


Material degradation

lower load carrying capacity

Large number of load cycles

thousands or millions

Continuum damage model

$\dot{D} = \mathrm{d}D / \mathrm{d}t$

Computational expense

memory and time

Model order reduction (MOR) techniques

Outline

1. State of the art
2. Reduced order model for cyclic loading
3. Challenges and workarounds
4. Numerical examples
5. Conclusions and current research

Outline

MOR for nonlinear problems

Short-term plastic/damage computations

- Proper Orthogonal Decomposition (POD)
  [Kerfriden, 2012; Ryckelynck, 2011].
- Proper Generalised Decomposition (PGD)
  [Vitse, 2016; Bhattacharyya, 2017].

Long-term fatigue computations

- Temporal homogenisation
  [Fish and Yu, 2002; Devulder, 2010]
- (Modified) jump cycle approach
  [Desmorat, 2005; Bhattacharyya, 2018]
- Space-time finite element method
  [Bhamare, 2014; Fritzen, 2018]

MOR for nonlinear problems

Short-term plastic/damage computations

- Proper Orthogonal Decomposition (POD)
  - [Kerfriden, 2012; Ryckelynck, 2011].
- Proper Generalised Decomposition (PGD)
  - [Vitse, 2016; Bhattacharyya, 2017].

Long-term fatigue computations

- Temporal homogenisation
  - [Fish and Yu, 2002; Devulder, 2010]
- (Modified) jump cycle approach
  - [Desmorat, 2005; Bhattacharyya, 2018]
- Space-time finite element method
  - [Bhamare, 2014; Fritzen, 2018]

PGD in a modified LATIN framework

Outline

Mechanical problem

Infinitesimal strain and isothermal quasi-static settings

Admissibility equations (**AD.**)

Static admissibility

$$\nabla\cdot\boldsymbol{\sigma} + \boldsymbol{b} = \boldsymbol{0} \quad \text{in } \Omega \times [0,T]$$

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

Kinematic admissibility

$$\boldsymbol{\varepsilon} = \nabla^{\mathrm{s}}\boldsymbol{u} \quad \text{in } \Omega \times [0,T]$$

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

Nonlinear constitutive model (**CM.**)

State equations

$$\boldsymbol{\sigma} = f(\psi(\boldsymbol{\varepsilon}, \boldsymbol{\varepsilon}^{\mathrm{p}}, \boldsymbol{q}))$$

$$\boldsymbol{Q} = g(\psi(\boldsymbol{\varepsilon}, \boldsymbol{\varepsilon}^{\mathrm{p}}, \boldsymbol{q}))$$

Evolution equations

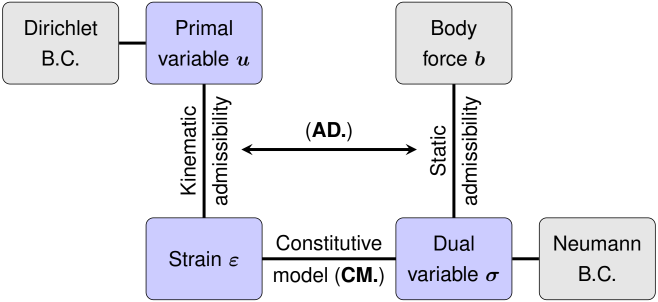
$$\dot{\boldsymbol{\varepsilon}}^{\mathrm{p}} = \hat{f}(\phi(\boldsymbol{\sigma}, \boldsymbol{Q}))$$

$$\dot{\boldsymbol{q}} = \hat{g}(\phi(\boldsymbol{\sigma}, \boldsymbol{Q}))$$

LATIN vs. incremental schemes

An **incremental** scheme:

Solve iteratively

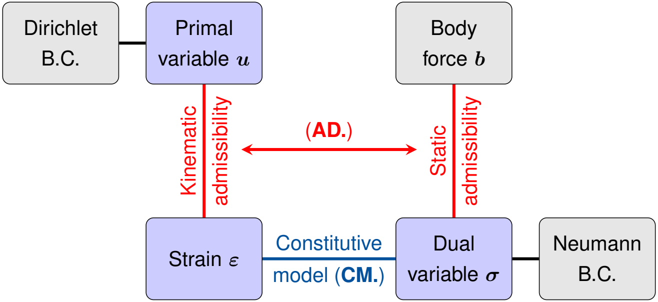


for each time step consecutively.

LATIN vs. incremental schemes

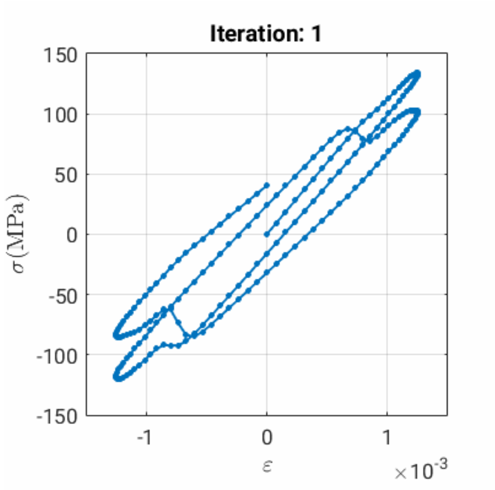
A **LATIN** scheme:

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



for all time steps simultaneously.

What does simultaneously mean?



LATIN solution algorithm

Start with an elastic initialisation s_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.)

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

Iterate until convergence with an energy error indicator

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

LATIN solution algorithm

Start with an elastic initialisation s_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

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

Static admissibility

Weak form at iteration $i+1$

$$\int_{\Omega\times\mathcal{I}} \nabla^{s}\boldsymbol{w}^{*} : \boldsymbol{\sigma}^{(i+1)}_{\text{global}} \, \mathrm{d}\Omega\,\mathrm{d}t = \int_{\Omega\times\mathcal{I}} \boldsymbol{w}^{*}\cdot\boldsymbol{b} \, \mathrm{d}\Omega\,\mathrm{d}t + \int_{\partial\Omega_N\times\mathcal{I}} \boldsymbol{w}^{*}\cdot\bar{\boldsymbol{t}} \, \mathrm{d}S\,\mathrm{d}t$$

Correction $\Delta\bullet^{(i+1)}_{\text{global}} = \bullet^{(i+1)} - \bullet^{(i)}$

$$\int_{\Omega\times\mathcal{I}} \nabla^{s}\boldsymbol{w}^{*} : \Delta\boldsymbol{\sigma}^{(i+1)}_{\text{global}} \, \mathrm{d}\Omega\,\mathrm{d}t = 0$$

Static admissibility and global search direction

$$\int_{\Omega\times\mathcal{I}} \nabla^{s}\boldsymbol{w}^{*} : \mathbb{H}^{-} : \Delta\boldsymbol{\varepsilon}^{(i+1)}_{\text{global}} \, \mathrm{d}\Omega\,\mathrm{d}t = -\int_{\Omega\times\mathcal{I}} \nabla^{s}\boldsymbol{w}^{*} : \underbrace{\hat{\boldsymbol{f}}_{\text{local}}}_{\text{known}} \, \mathrm{d}\Omega\,\mathrm{d}t$$

Low-rank approximation

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

Enrichment to (μ) previously generated modes

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

POD-like update of (μ) previously generated modes

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

Alternated directions algorithm

Galerkin finite element discretisation

Spatial problem, with homogeneous boundary conditions:

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

Temporal problem, with zero initial conditions:

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

What is actually solved?

Choosing the search directions $\mathbb{H}^-$ as

$$\mathbb{H}^- = \alpha\, \mathbb{C} \qquad \alpha \in]0,1]$$

Enrichment step

$$\gamma\, \underline{\underline{K}}\, \underline{v} = \underline{f} \qquad \gamma \in \mathbb{R} \qquad \underline{\underline{K}} \in \mathbb{R}^{n \times n} \qquad \underline{v}, \underline{f} \in \mathbb{R}^{n} \qquad \text{with B.C.}$$

$$a\, \underline{\lambda} = \underline{b} \qquad a \in \mathbb{R} \qquad \underline{\lambda}, \underline{b} \in \mathbb{R}^{n_t} \qquad \text{with I.C.}$$

Temporal update step

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

Outline

Challenges in a PGD framework

The cost of integration over all generalised coordinates

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

The fast increase in the number of modes

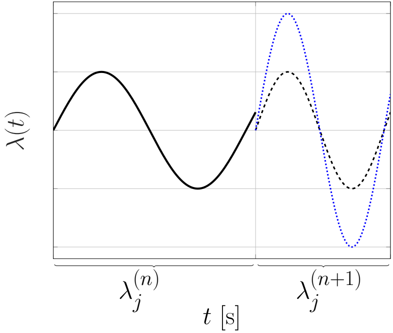
$$\boldsymbol{u}(\boldsymbol{x}, t) = \sum_{j=1}^{\mu} \boldsymbol{v}_j(\boldsymbol{x}) \circ \boldsymbol{\lambda}_j(t) \text{ with large } \mu \qquad ☹$$

Efficient variable amplitude and frequency simulations

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 $\tilde{t} \in [0,1]$

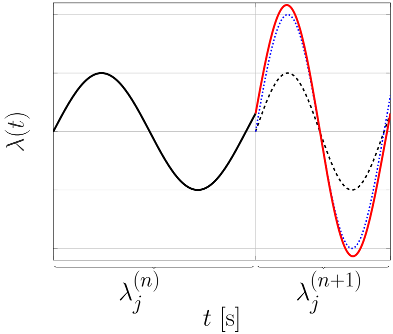
Scaled back to the temporal coordinate of the current cycle

Semi-incremental extension

Continuity of the temporal modes is an issue

Temporal modes are vertically scaled and shifted via

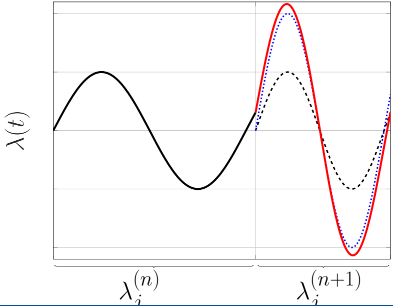
$$\tilde{\lambda}_j^{(n+1)}(\tilde{t}) = m\,\tilde{\lambda}_j^{(n)}(\tilde{t}) + g\,\tilde{t} + h \quad \text{with I.C. and B.C.}$$



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 🙂

Large number of modes

Slow temporal update step

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

PGD 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

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

Solution with a compressed basis

$$\underline{\underline{U}} \approx \underline{\underline{Q}}_v\ \underline{\underline{\hat{V}}} \quad \underline{\underline{\hat{S}}} \quad \underline{\underline{\hat{\Lambda}}}^{\mathsf{T}} \underline{\underline{Q}}_\lambda^{\mathsf{T}}$$

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

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

Solution with a compressed basis

$$\underline{\underline{U}} \approx \underline{\underline{Q}}_v\ \underline{\underline{\hat{V}}}\ \underline{\underline{\hat{S}}}\ \underline{\underline{\hat{\Lambda}}}^{\mathsf{T}} \underline{\underline{Q}}_\lambda^{\mathsf{T}}$$

Non-demanding optimal decomposition 🙂

Is this efficient?

A low-rank approximation of $\varepsilon_{\text{global}}$

What about $\boldsymbol{\sigma}_{\text{global}}$, the local stage and the error indicator?

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

What is missing is PGD of $\varepsilon_{\text{local}}$

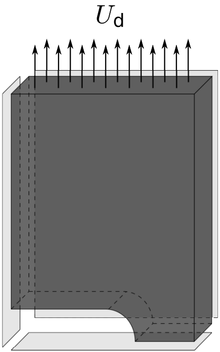
Randomised SVD (RSVD) algorithm

A dedicated arithmetic toolbox to handle operations on PGD

Outline

Numerical results

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



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

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

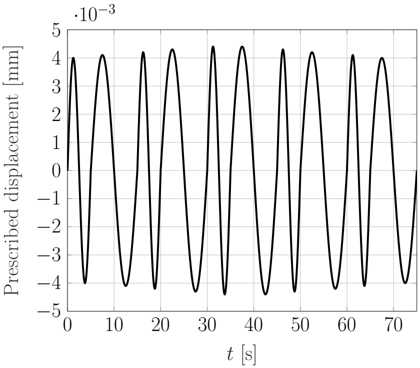
Damage

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

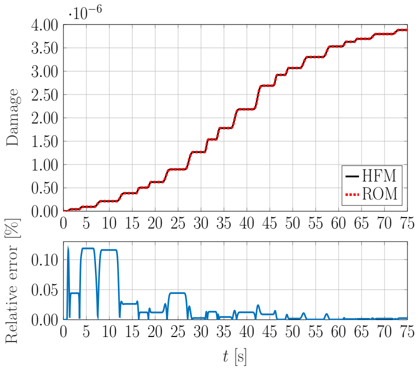
Model verification

$1884 \cdot 41 \cdot 10$ DOF

With respect to a modified Newton-Raphson scheme



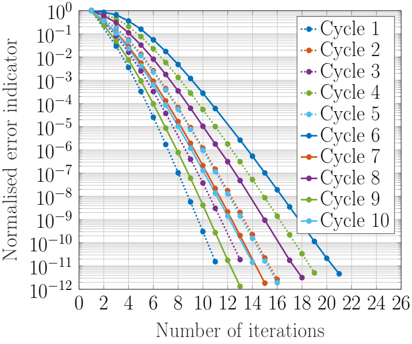
Prescribed displacement



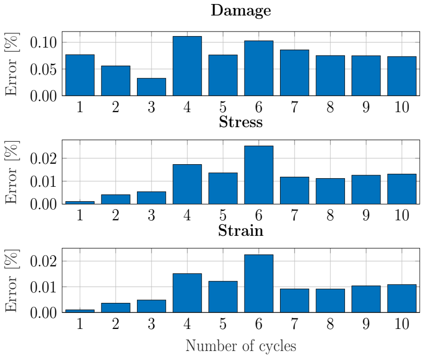
Damage evolution

Model verification

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



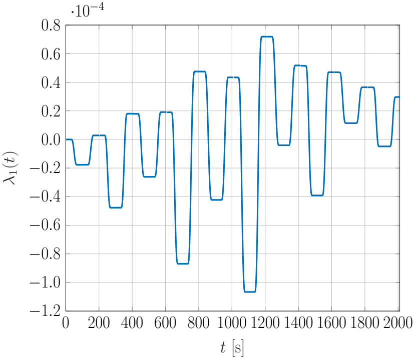
Error indicator

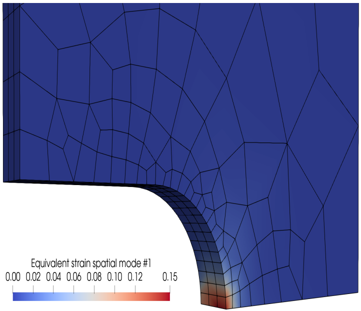


Space-time average relative error

Model verification

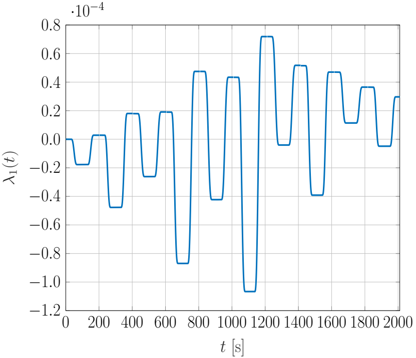
The first temporal and spatial modes

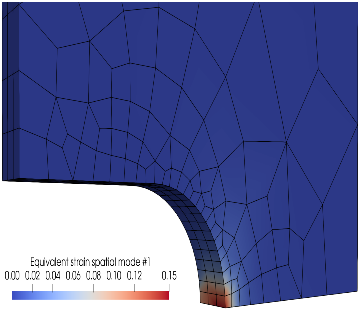




Model verification

The first temporal and spatial modes



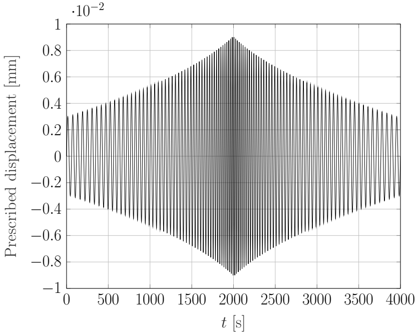


12 modes / Speedup factor ~ 25

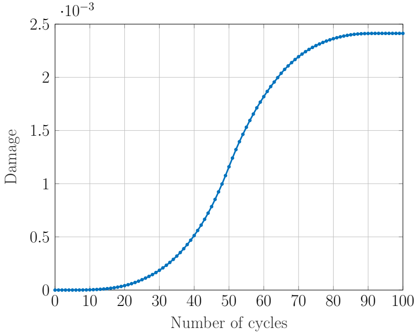
Variable amplitudes & frequencies

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

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



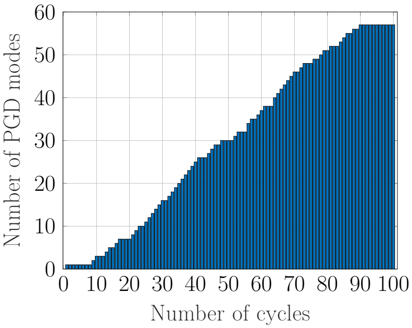
Prescribed displacement



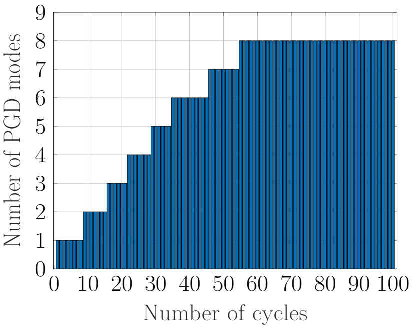
Damage evolution

Variable amplitudes & frequencies

The growth of the ROB using an SVD scheme



Gram-Schmidt

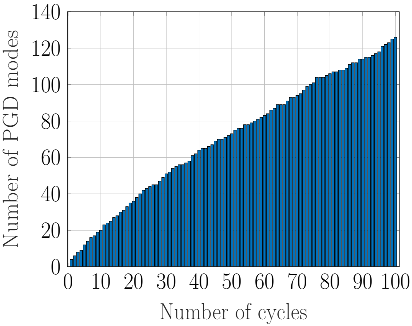


SVD scheme

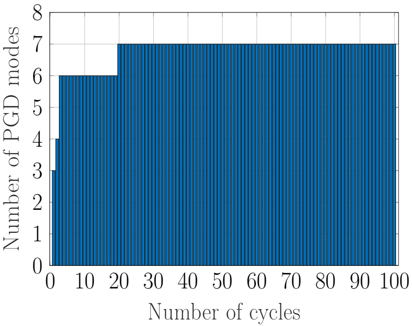
Different ortho. schemes

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

In case of random loading and fine discretisation



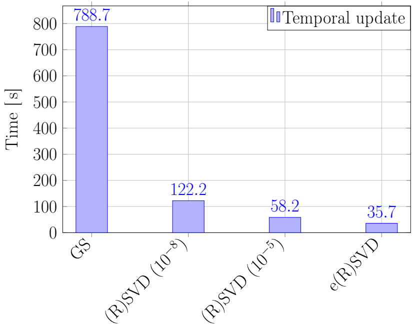
Gram-Schmidt



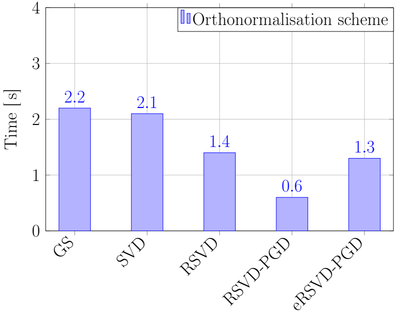
SVD scheme

Different ortho. schemes

The required time to update and orthonormalise the modes



Temporal update

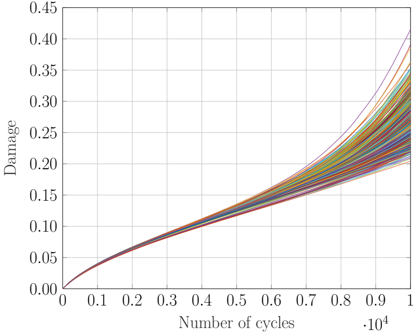


Orthonormalisation step

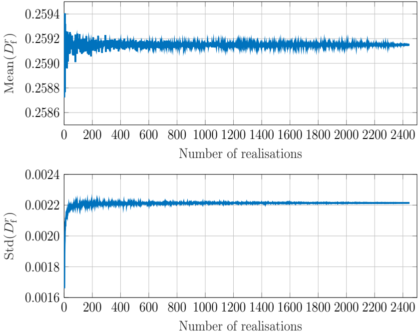
Random amplitude loading

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

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



Different damage realisations

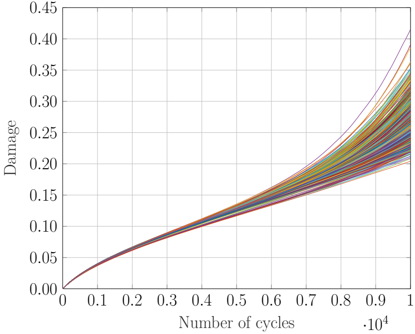


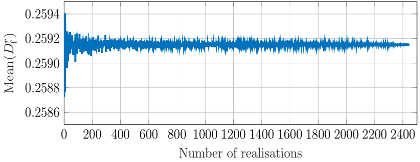
The mean and STD of D_f

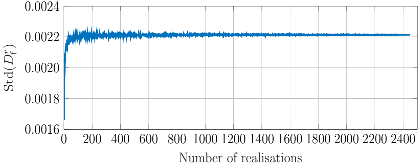
Random amplitude loading

$1884 \cdot 41 \cdot 10^4$ 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 $P_\mathrm{f} = 5.4\%$

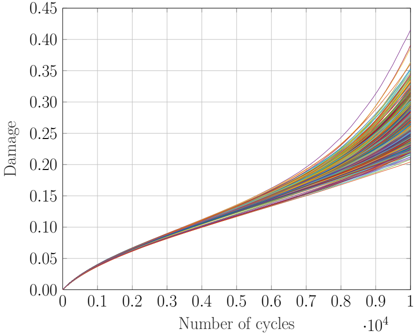
Requirements:

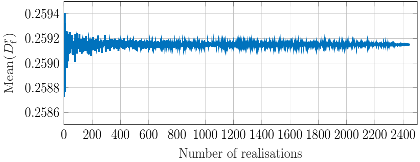
$[15 - 35]\,\mathrm{min}$ and $[1 - 1.5]\,\mathrm{GB}$

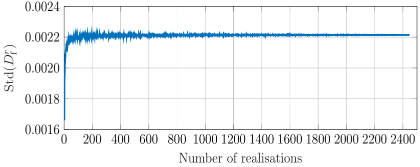
Random amplitude loading

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

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







Critical damage value $D_c = 0.3$

Probability of failure $P_f = 5.4\%$

Requirements:

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

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

Outline

1. State of the art
2. Reduced order model for cyclic loading
3. Challenges and workarounds
4. Numerical examples
5. **Conclusions and current research**

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 research

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 research

Reducing the cost of evaluating the constitutive model

Thank you for your attention