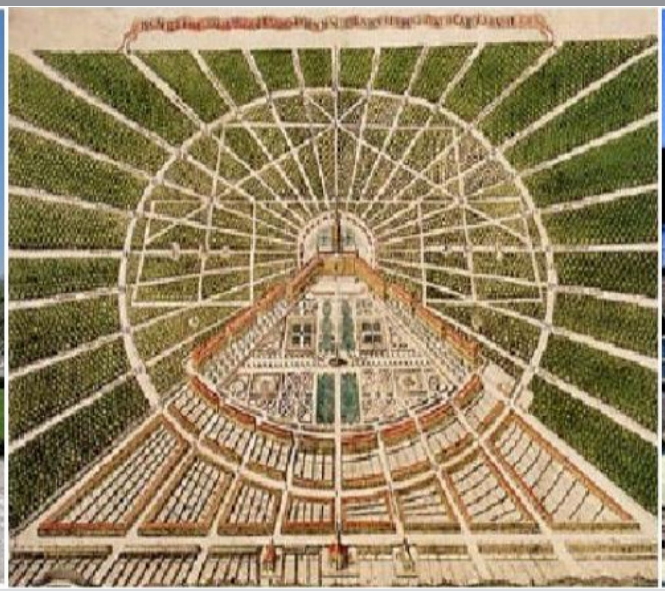
Karlsruhe Institute of Technology

Université franco-allemande
Deutsch-Französische Hochschule

Matti Schneider

# Faster primal solvers

Introduction to FFT-based numerical methods for the homogenization of random materials



# Overview

1. Accelerated gradient methods
2. Newton - CG
3. Adaptive parameter selection
4. Summary and conclusions

# Overview

1. Accelerated gradient methods
2. Newton - CG
3. Adaptive parameter selection
4. Summary and conclusions

# Previously ...

sought:

$$\langle w(\cdot,\varepsilon)\rangle_Q \longrightarrow \min_{\varepsilon=\bar{\varepsilon}+\nabla^s u} \quad \Longleftrightarrow \quad \operatorname{div} \mathsf{S}(\cdot,\bar{\varepsilon}+\nabla^s u) = 0, \quad \mathsf{S} \equiv \frac{\partial w}{\partial \varepsilon}$$

basic scheme:

$$\varepsilon^{k+1} = \bar{\varepsilon} - \Gamma^0 : \left(\mathsf{S}(\cdot,\varepsilon^k) - \mathbb{C}^0 : \varepsilon^k\right)$$

coincides with (projected) gradient descent

$$x^{k+1} = P_A(x^k - s^k \nabla f(x^k))$$

# Convergence criterion?

$$\varepsilon^{k+1} = \bar{\varepsilon} - \Gamma^0 : \left( \mathsf{S}(\cdot, \varepsilon^k) - \mathbb{C}^0 : \varepsilon^k \right)$$

know:

$$\varepsilon^k = \bar{\varepsilon} + \Gamma^0 : \mathbb{C}^0 : \varepsilon^k \qquad \left( \quad \Leftarrow P_A(P_A(x)) = P_A(x) \quad \right)$$

# Convergence criterion?

$$\varepsilon^{k+1} = \varepsilon^{k} - \Gamma^{0} : \mathsf{S}(\cdot, \varepsilon^{k})$$

know:

$$\varepsilon^{k} = \bar{\varepsilon} + \Gamma^{0} : \mathbb{C}^{0} : \varepsilon^{k} \qquad \left( \quad \Leftarrow P_A(P_A(x)) = P_A(x) \quad \right)$$

# Convergence criterion?

$$\varepsilon^{k+1} - \varepsilon^{k} = -\Gamma^{0} : \mathsf{S}(\cdot, \varepsilon^{k})$$

# Convergence criterion?

$$\varepsilon^{k+1} - \varepsilon^{k} = -\frac{1}{\alpha_0} \Gamma : \mathsf{S}(\cdot, \varepsilon^{k})$$

# Convergence criterion?

$$\alpha_0 \left( \varepsilon^{k+1} - \varepsilon^k \right) = -\Gamma : \mathsf{S}(\cdot, \varepsilon^k)$$

# Convergence criterion?

$$\alpha_0 \left\| \varepsilon^{k+1} - \varepsilon^{k} \right\| = \left\| \Gamma : \mathsf{S}(\cdot, \varepsilon^{k}) \right\|$$

# Convergence criterion?

$$\frac{\alpha_0 \left\| \varepsilon^{k+1} - \varepsilon^k \right\|}{\|\bar{\sigma}^k\|} = \frac{\left\| \Gamma : \mathsf{S}(\cdot, \varepsilon^k) \right\|}{\|\bar{\sigma}^k\|} \overset{!}{\leq} \texttt{tol}$$

Why?

- dimension-free
- independent of algorithm, i.e., $\alpha_0$
- readily computable

# Final basic scheme

**Algorithm 1** Basic scheme $(\bar{\varepsilon}, \texttt{maxit}, \texttt{tol}, \alpha_0)$

$\varepsilon \leftarrow \varepsilon^0$ ▷ $\varepsilon^0 \equiv \bar{\varepsilon}$ or via extrapolation
$\texttt{res} \leftarrow 1$
$\bar{\sigma} \leftarrow 0$
$k \leftarrow 0$
**while** $k < \texttt{maxit}$ **and** $\texttt{res} > \texttt{tol}$ **do**
    $k \leftarrow k + 1$
    $\xi \leftarrow \varepsilon$
    $\varepsilon \leftarrow \mathsf{S}(\cdot, \varepsilon) - \alpha_0\, \varepsilon$ ▷ estimate $\alpha_\pm$
    $\bar{\sigma} \leftarrow \langle \varepsilon \rangle_Q + \alpha_0\, \bar{\varepsilon}$ ▷ $\langle \varepsilon \rangle_Q = \hat{\varepsilon}(0)$
    $\varepsilon \leftarrow \bar{\varepsilon} - 1/\alpha_0\, \Gamma : \varepsilon$ ▷ use FFT & favorite discretization
    $\texttt{res} \leftarrow \alpha_0\, \|\varepsilon - \xi\| / \|\bar{\sigma}\|$ ▷ update $\alpha_0$ afterwards
**end while**
**return** $\varepsilon, \bar{\sigma}, \texttt{res}$ ▷ Requires two strain fields

[H. Moulinec and P. Suquet, Comptes Rendus de l'Académie des Sciences, 1994]

[H. Moulinec and P. Suquet, CMAME, 1998]

# Last time

$$\alpha_- \leq \lambda \leq \alpha_+ \quad \forall x, \xi \, \forall \lambda \in \texttt{Eig}\left( \frac{\partial \mathrm{S}}{\partial \varepsilon}(x, \xi) \right)$$

implies

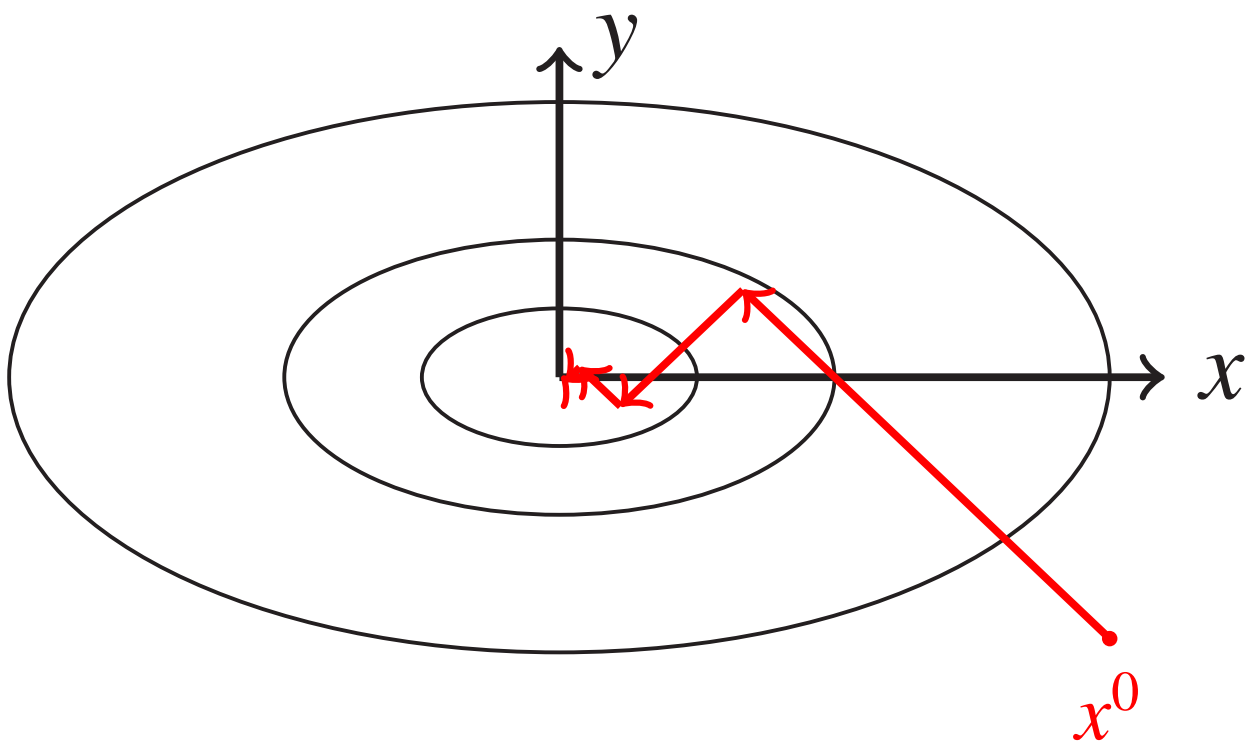
$$\| \varepsilon^{k+1} - \varepsilon^* \|_{L^2} \leq \left( \frac{\kappa - 1}{\kappa + 1} \right) \| \varepsilon^k - \varepsilon^* \|_{L^2}$$

for

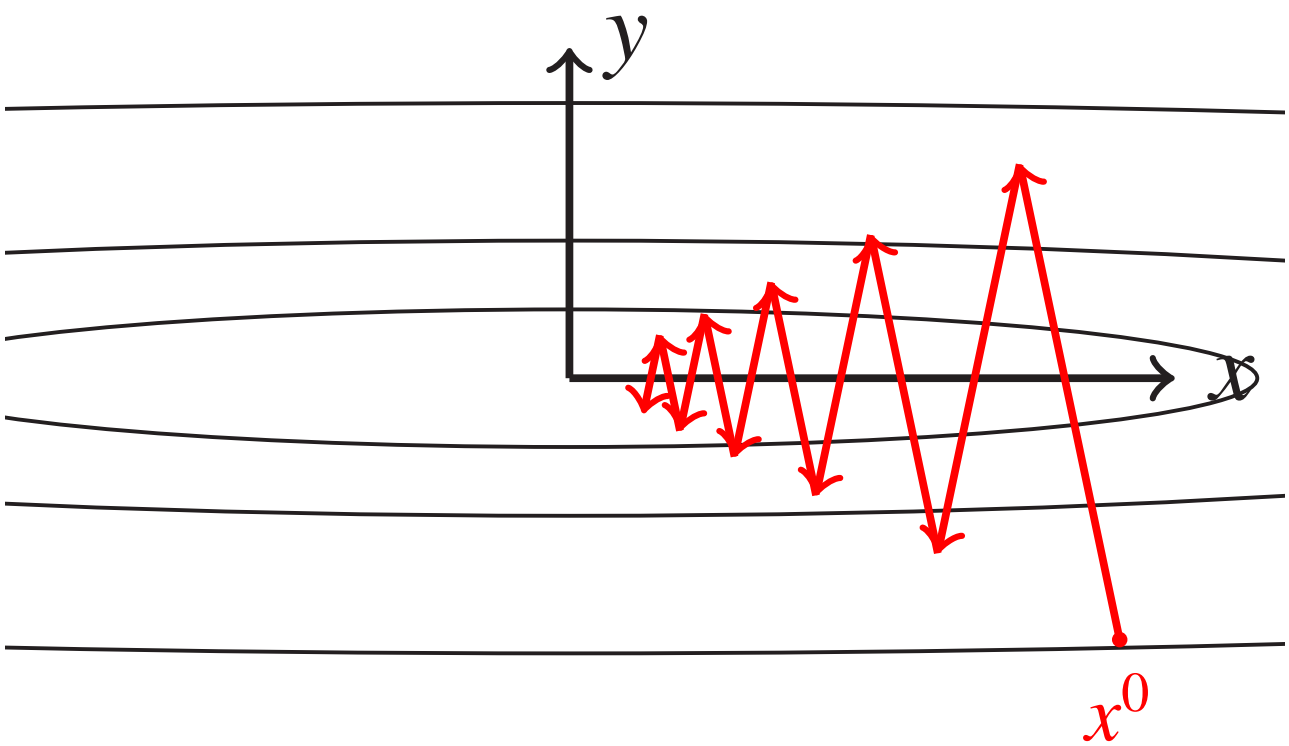
$$\alpha_0 = \frac{\alpha_- + \alpha_+}{2} \quad \text{and} \quad \kappa = \frac{\alpha_+}{\alpha_-}$$

$\Rightarrow$ # iterations $\propto \kappa$

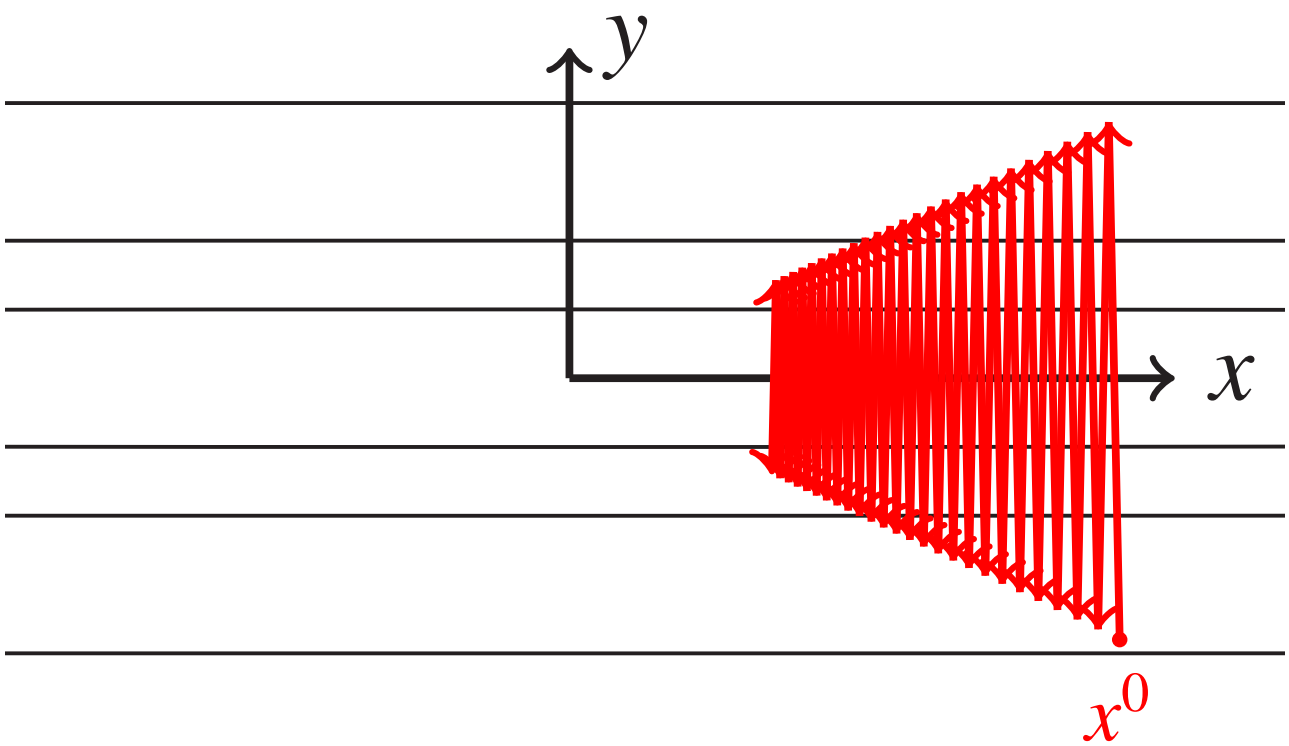
- What if $\kappa \gg 1$? $\nearrow$ $\kappa = 2$



- What if $\kappa \gg 1$? $\nearrow \kappa = 10$



- What if $\kappa \gg 1$? $\nearrow \kappa = 100$



# Remedy - fast gradient methods

- augment (projected) gradient descent ↗ **heavy-ball method** (Polyak)

  [B. Polyak, USSR Computational Mathematics and Mathematical Physics, 1964]

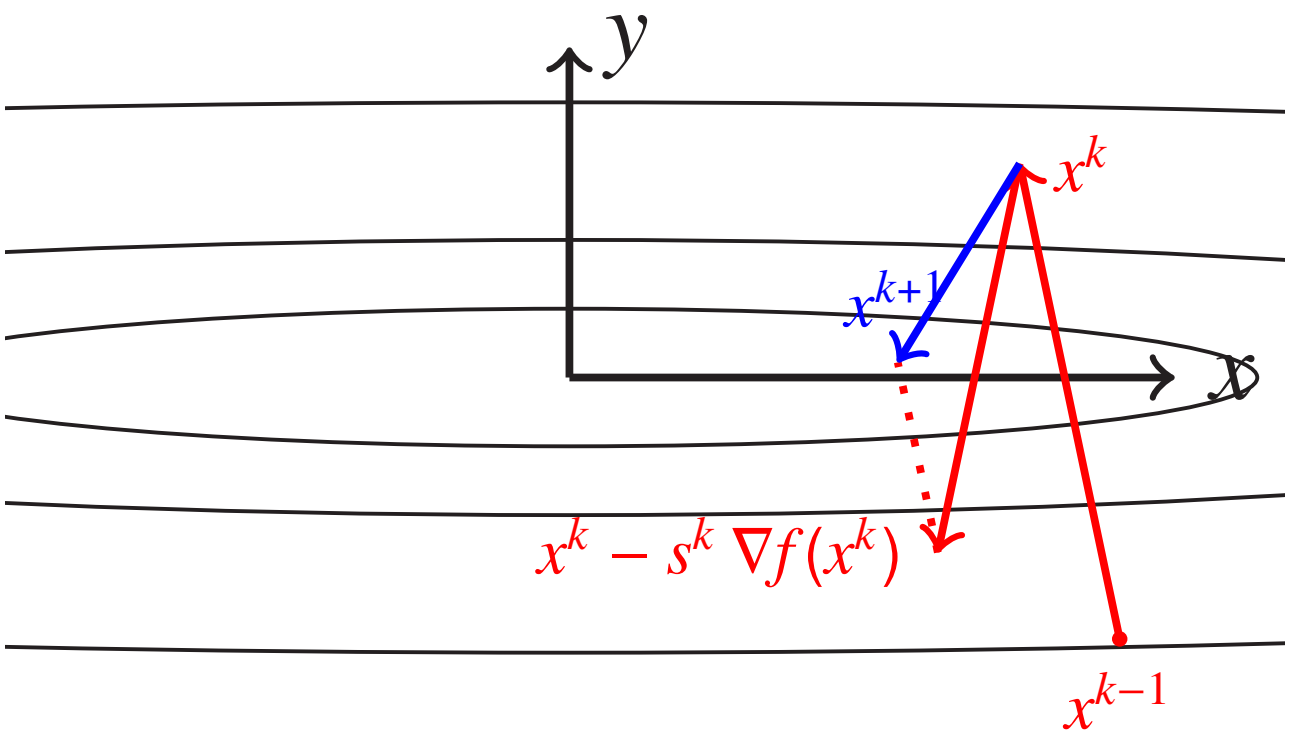
$$x^{k+1} = P_A(x^k - s^k \nabla f(x^k) + \beta^k (x^k - x^{k-1}))$$

# Remedy - fast gradient methods

- augment (projected) gradient descent ↗ **heavy-ball method** (Polyak)

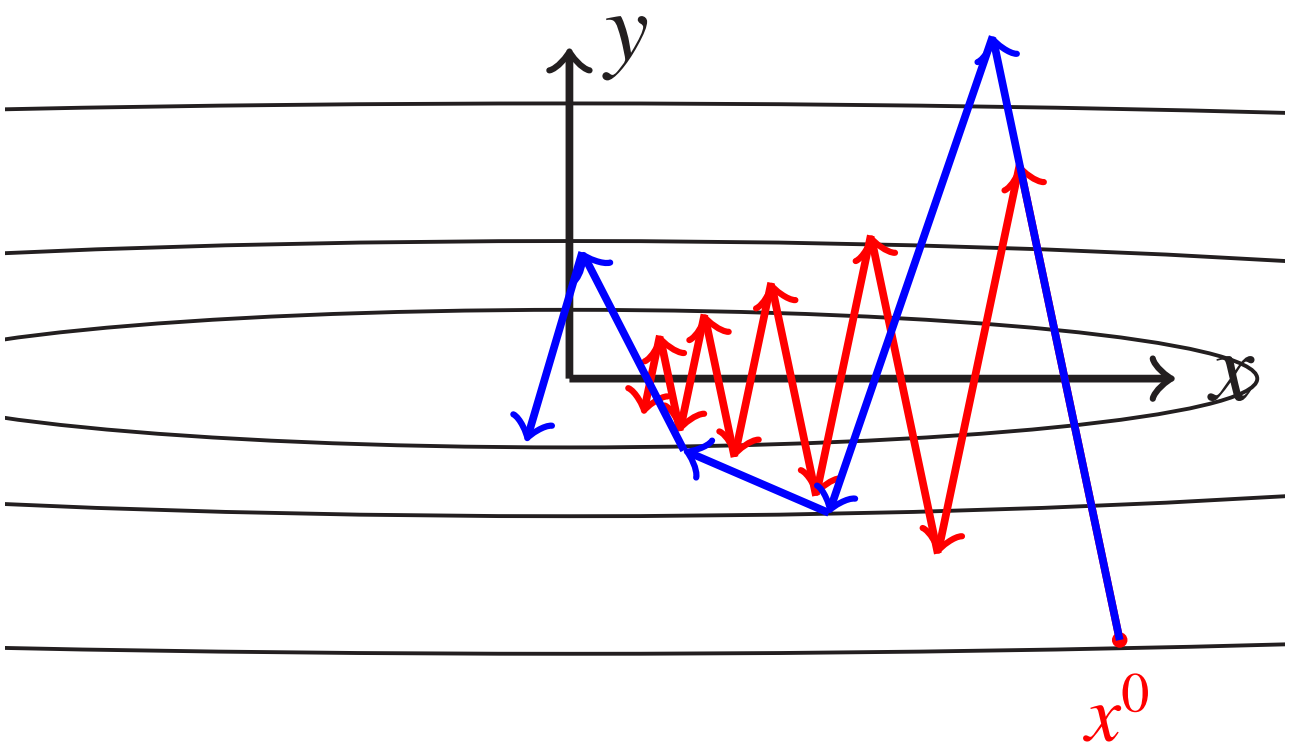
  [B. Polyak, USSR Computational Mathematics and Mathematical Physics, 1964]

$$x^{k+1} = P_A(x^k - s^k \nabla f(x^k) + \beta^k (x^k - x^{k-1}))$$



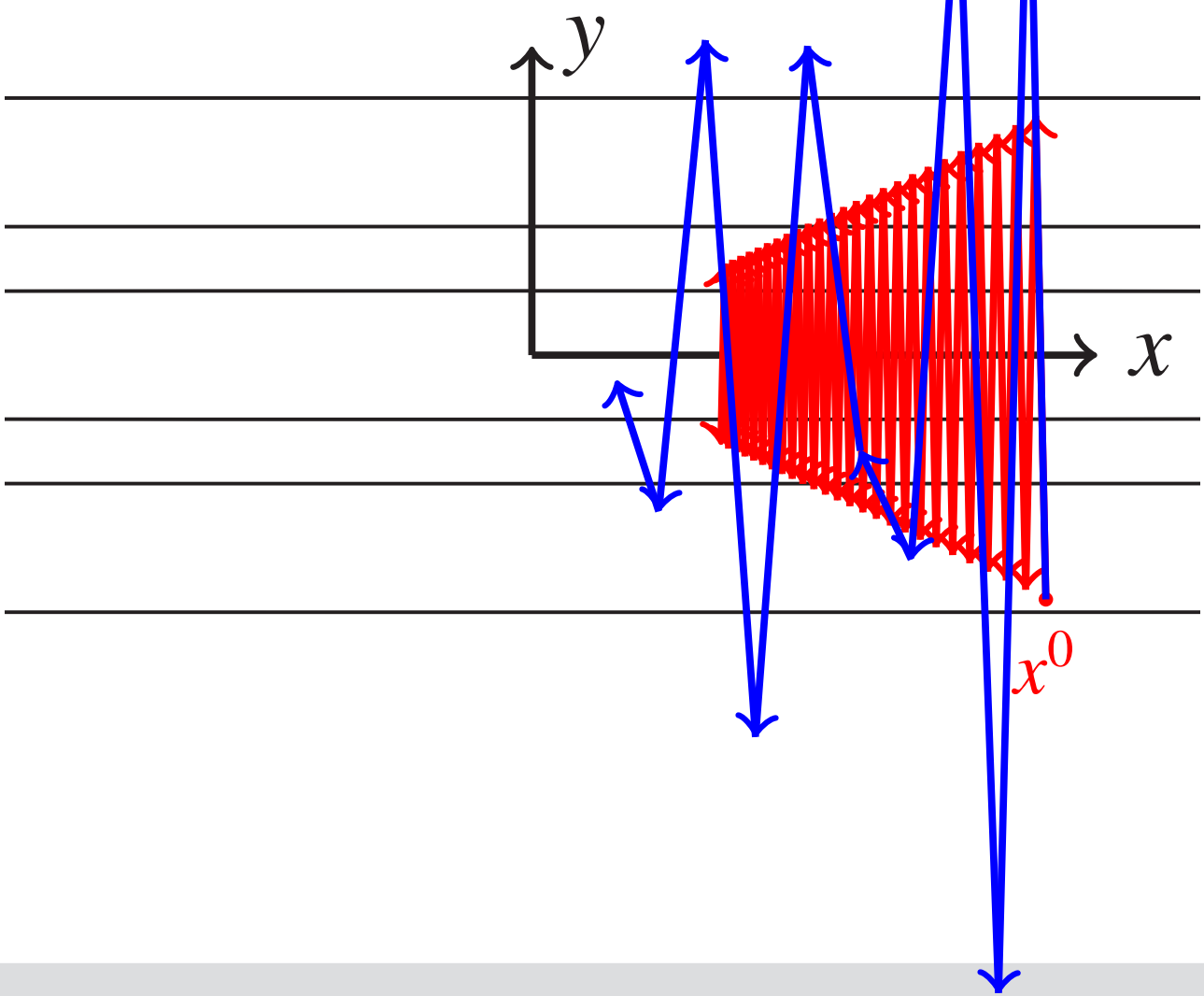
# Gradient vs fast gradient

- What if $\kappa \gg 1$? $\nearrow$ $\kappa = 10$



# Gradient vs fast gradient

- What if $\kappa \gg 1$? $\nearrow$ $\kappa = 100$



# Heavy ball for micromechanics

$$\varepsilon^{k+1} = \bar{\varepsilon} - \Gamma^0 : \left(\mathsf{S}(\cdot, \varepsilon^k) - \mathbb{C}^0 : \varepsilon^k\right) + \beta_k \left(\varepsilon^k - \varepsilon^{k-1}\right)$$

with

$$\alpha_- \leq \lambda \leq \alpha_+ \quad \forall x, \xi \, \forall \lambda \in \mathtt{Eig}\left(\frac{\partial \mathsf{S}}{\partial \varepsilon}(x, \xi)\right)$$

best rate for (linear elasticity)

$$\alpha_0 = \left(\frac{\sqrt{\alpha_-} + \sqrt{\alpha_+}}{2}\right)^2 \quad \text{and} \quad \beta^k = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \quad \text{with} \quad \kappa = \frac{\alpha_+}{\alpha_-}$$

$\Rightarrow$ # iterations $\propto \sqrt{\kappa}$

# Heavy ball for micromechanics

$$\varepsilon^{k+1} = \bar{\varepsilon} - \Gamma^0 : \left(\mathsf{S}(\cdot, \varepsilon^k) - \mathbb{C}^0 : \varepsilon^k\right) + \beta_k \left(\varepsilon^k - \varepsilon^{k-1}\right)$$

with

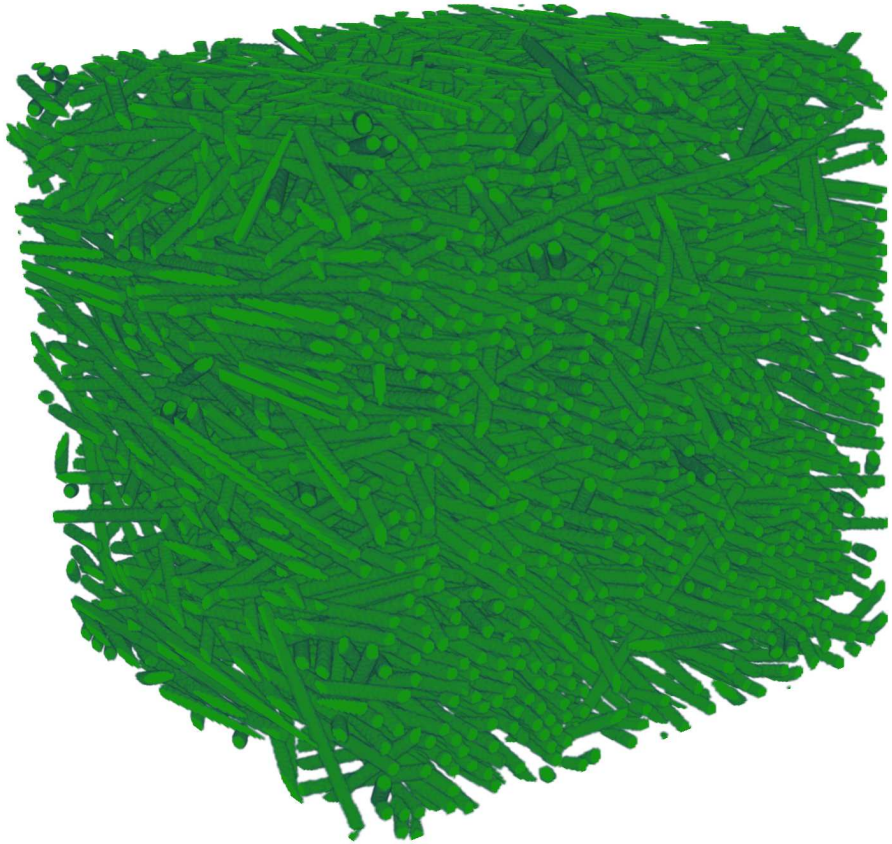
$$\alpha_- \leq \lambda \leq \alpha_+ \quad \forall x, \xi \, \forall \lambda \in \mathtt{Eig}\left(\frac{\partial \mathsf{S}}{\partial \varepsilon}(x, \xi)\right)$$

best rate for (linear elasticity)

$$\alpha_0 = \left(\frac{\sqrt{\alpha_-} + \sqrt{\alpha_+}}{2}\right)^2 \quad \text{and} \quad \beta^k = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \quad \text{with} \quad \kappa = \frac{\alpha_+}{\alpha_-}$$

$\Rightarrow$ # iterations $\propto \sqrt{\kappa}$

# Practical performance - setup

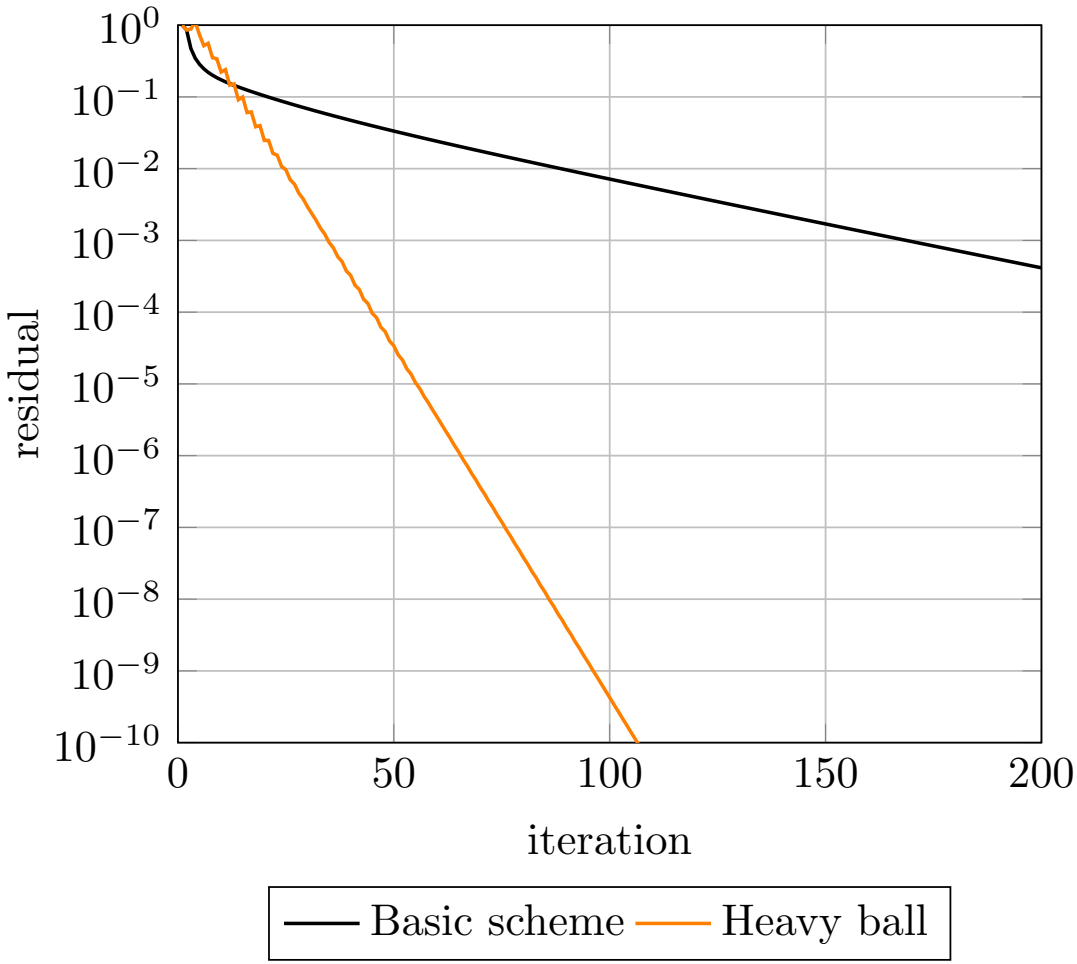


- glass-fiber reinforced PA, $256^3$
- $\phi = 20\%$, $r_a = 30$,
  $A = \mathrm{diag}(0.8, 0.1, 0.1)$
- generated by SAM

  [MS, Computational Mechanics, 2017]
- $E_{\mathrm{Fiber}} = 72$ GPa, $\nu_{\mathrm{Fiber}} = 0.22$,
  $E_{\mathrm{PA}} = 2.1$ GPa, $\nu_{\mathrm{PA}} = 0.3$
- uniaxial extension in $e_1$

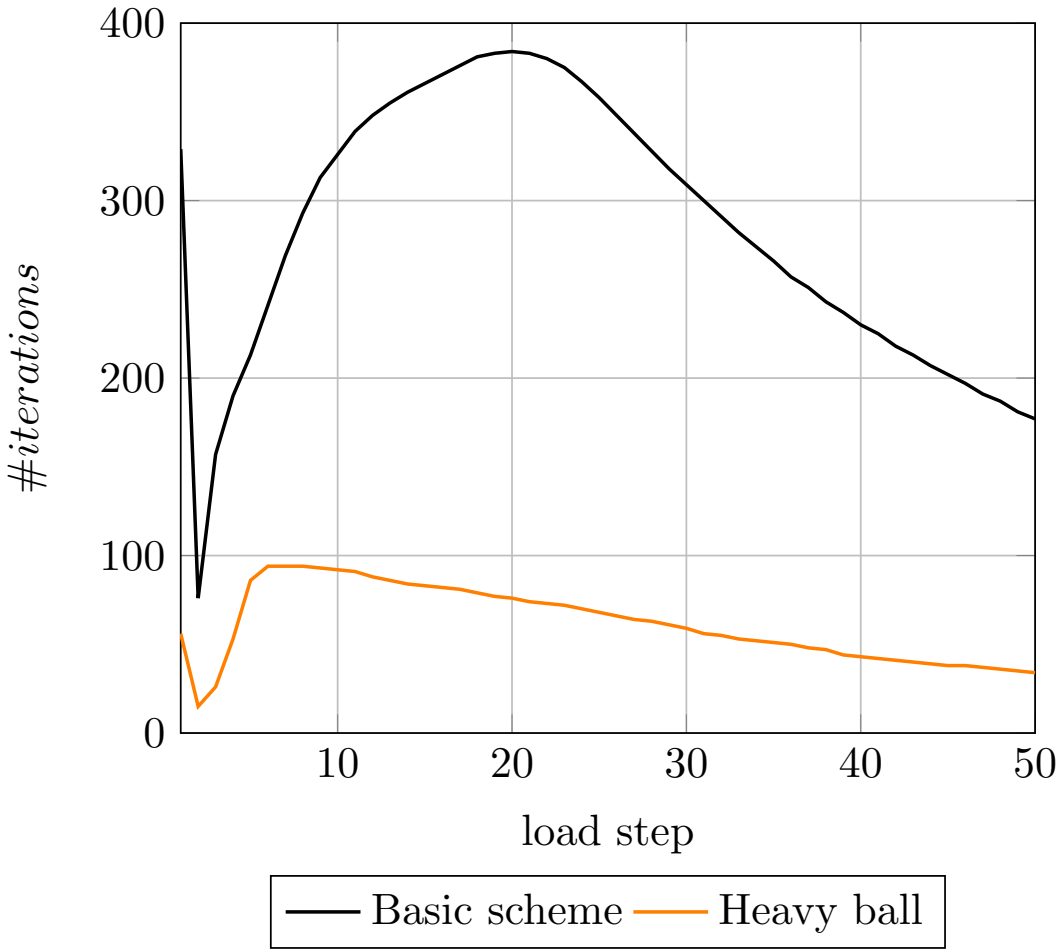
[MS, International Journal for Numerical Methods in Engineering, 2019]

# Practical performance - linear elasticity




[MS, International Journal for Numerical Methods in Engineering, 2019]

# Practical performance - vM plasticity




[MS, International Journal for Numerical Methods in Engineering, 2019]

- von Mises elastoplastic matrix
- 5% uniaxial extension in x
- $\texttt{tol} = 10^{-5}$

average it.

| | Basic | Heavy ball |
|---|---|---|
| $128^3$ | 284.08 | 64.64 |
| $256^3$ | 382.7 | 61.58 |

# Implementation

$$\varepsilon^{k+1} = \bar{\varepsilon} - \Gamma^0 : \left(\mathsf{S}(\cdot, \varepsilon^k) - \mathbb{C}^0 : \varepsilon^k\right) + \beta_k \left(\varepsilon^k - \varepsilon^{k-1}\right)$$

- rewriting (for implementation)

$$\varepsilon^{k+1} = \bar{\varepsilon} - \Gamma^0 : \left(\mathsf{S}(\cdot, \varepsilon^k) - \mathbb{C}^0 : \varepsilon^k - \beta_k \, \mathbb{C}^0 : \left(\varepsilon^k - \varepsilon^{k-1}\right)\right)$$

- rewriting (for residual)

$$\varepsilon^{k+1} - \varepsilon^k = -\Gamma^0 : \mathsf{S}(\cdot, \varepsilon^k) + \beta_k \left(\varepsilon^k - \varepsilon^{k-1}\right)$$

$$\|\varepsilon^{k+1} - \varepsilon^k\|^2 = \|\Gamma^0 : \mathsf{S}(\cdot, \varepsilon^k)\|^2 - \frac{2\beta_k}{\alpha_0} \left(\mathsf{S}(\cdot, \varepsilon^k), \varepsilon^k - \varepsilon^{k-1}\right)_{L^2} + \beta_k^2 \, \|\varepsilon^k - \varepsilon^{k-1}\|_{L^2}^2$$

## **Algorithm 2** Heavy Ball $(\bar{\varepsilon}, \texttt{maxit}, \texttt{tol}, \alpha_0, \beta)$

$[\varepsilon, \xi] \leftarrow [\varepsilon^0, \varepsilon^0]$ ▷ $\varepsilon^0 \equiv \bar{\varepsilon}$ or via extrapolation

$[\texttt{res}, \bar{\sigma}, k, p] \leftarrow [1, 0, 0, 0]$

**while** $k < \texttt{maxit}$ **and** $\texttt{res} > \texttt{tol}$ **do**

  $k \leftarrow k + 1$

  $p_{\texttt{old}} \leftarrow p$

$$\begin{bmatrix} \varepsilon \\ \xi \\ p \\ q \end{bmatrix} \leftarrow \begin{bmatrix} \mathsf{S}(\cdot, \varepsilon) - \alpha_0 \, \varepsilon - \alpha_0 \beta \, (\varepsilon - \xi) \\ \varepsilon \\ \|\varepsilon - \xi\|^2 \\ (\mathsf{S}(\cdot, \varepsilon), \varepsilon - \xi)_{L^2} \end{bmatrix}$$ ▷ estimate $\alpha_\pm$

  $\bar{\sigma} \leftarrow \langle \varepsilon \rangle_Q + \alpha_0 \, \bar{\varepsilon}$ ▷ $\langle \varepsilon \rangle_Q = \hat{\varepsilon}(0)$

  $\varepsilon \leftarrow \bar{\varepsilon} - 1/\alpha_0 \, \Gamma : \varepsilon$ ▷ use FFT & favorite discretization

  $\texttt{res} \leftarrow (p + 2\beta q/\alpha_0 - \beta^2 \, p_{\texttt{old}})/\|\bar{\sigma}\|$ ▷ update $\alpha_0, \beta$ afterwards

**end while**

**return** $\varepsilon, \bar{\sigma}, \texttt{res}$ ▷ Requires two strain fields

[F. Ernesti and MS, CMAME, 2020]

# Overview

1. Accelerated gradient methods

2. Newton - CG

3. Adaptive parameter selection

4. Summary and conclusions

# Linear conjugate gradient method

- goal: $\frac{1}{2}x^T Ax - b^T x \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \overset{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$

- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

**Algorithm 3** Linear CG $(A, b, x^0, \mathrm{maxit}, \mathtt{tol})$

1: $g \leftarrow Ax^0 - b$
2: $d \leftarrow -g$
3: $\mathtt{r} \leftarrow \|g\|$
4: $k \leftarrow 0$
5: **while** $k < \mathrm{maxit}$ **and** $\mathtt{r} > \mathtt{tol}$ **do**
6: $\quad k \leftarrow k + 1$
7: $\quad \mathtt{r}_{\mathrm{old}} \leftarrow \mathtt{r}$
8: $\quad z \leftarrow Ad$
9: $\quad \alpha \leftarrow \mathtt{r}^2 / d^T z$
10: $\quad x \leftarrow x + \alpha\, d$
11: $\quad g \leftarrow g + \alpha\, z$
12: $\quad \mathtt{r} \leftarrow \|g\|$
13: $\quad \gamma \leftarrow \mathtt{r}^2 / \mathtt{r}^2_{\mathrm{old}}$
14: $\quad d \leftarrow -g + \gamma\, d$
15: **end while**
16: **return** $x, \mathtt{r}$

[M. Hestenes and E. Stiefel, Journal of Research of the National Bureau of Standards, 1952]

# Linear conjugate gradient method

- goal: $\frac{1}{2}x^T Ax - b^T x \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \overset{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$

- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

**Algorithm 3** Linear CG $(A, b, x^0, \mathrm{maxit}, \mathtt{tol})$

1: $g \leftarrow Ax^0 - b$
2: $d \leftarrow -g$
3: $\mathtt{r} \leftarrow \|g\|$
4: $k \leftarrow 0$
5: **while** $k < \mathrm{maxit}$ **and** $\mathtt{r} > \mathtt{tol}$ **do**
6: $\quad k \leftarrow k + 1$
7: $\quad \mathtt{r}_{\mathrm{old}} \leftarrow \mathtt{r}$
8: $\quad z \leftarrow Ad$
9: $\quad \alpha \leftarrow \mathtt{r}^2 / d^T z$
10: $\quad x \leftarrow x + \alpha\, d$
11: $\quad g \leftarrow g + \alpha\, z$
12: $\quad \mathtt{r} \leftarrow \|g\|$
13: $\quad \gamma \leftarrow \mathtt{r}^2 / \mathtt{r}^2_{\mathrm{old}}$
14: $\quad d \leftarrow -g + \gamma\, d$
15: **end while**
16: **return** $x, \mathtt{r}$

[M. Hestenes and E. Stiefel, Journal of Research of the National Bureau of Standards, 1952]

# Linear conjugate gradient method

- goal: $\frac{1}{2} x^T A x - b^T x \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \overset{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$

- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

**Algorithm 3** Linear CG $(A, b, x^0, \text{maxit}, \texttt{tol})$

$g \leftarrow Ax^0 - b$
$d \leftarrow -g$
$\texttt{r} \leftarrow \|g\|$
$k \leftarrow 0$
**while** $k < \text{maxit}$ **and** $\texttt{r} > \texttt{tol}$ **do**
    $k \leftarrow k + 1$
    $\texttt{r}_{\text{old}} \leftarrow \texttt{r}$
    $z \leftarrow Ad$
    $\alpha \leftarrow \texttt{r}^2 / d^T z$
    $x \leftarrow x + \alpha\, d$
    $g \leftarrow g + \alpha\, z$
    $\texttt{r} \leftarrow \|g\|$
    $\gamma \leftarrow \texttt{r}^2 / \texttt{r}^2_{\text{old}}$
    $d \leftarrow -g + \gamma\, d$
**end while**
**return** $x, \texttt{r}$

[M. Hestenes and E. Stiefel, Journal of Research of the National Bureau of Standards, 1952]

# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \overset{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$

- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

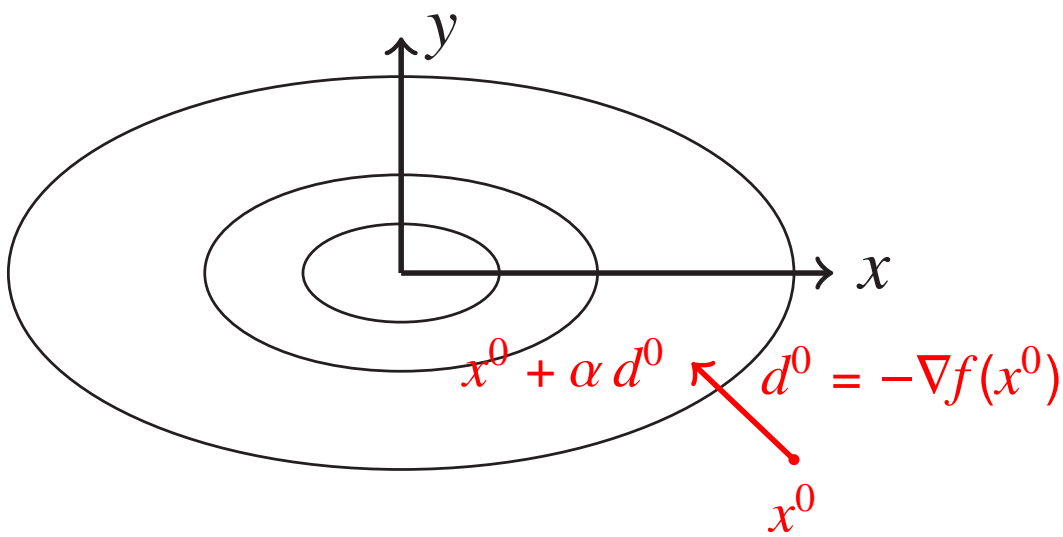
# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$




- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

# Linear conjugate gradient method

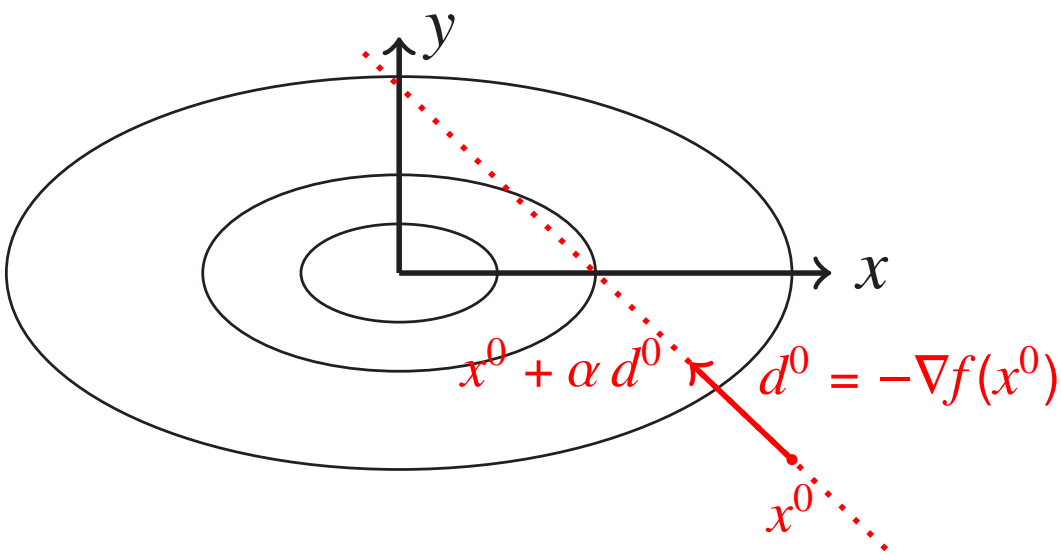
- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$

- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method



# Linear conjugate gradient method

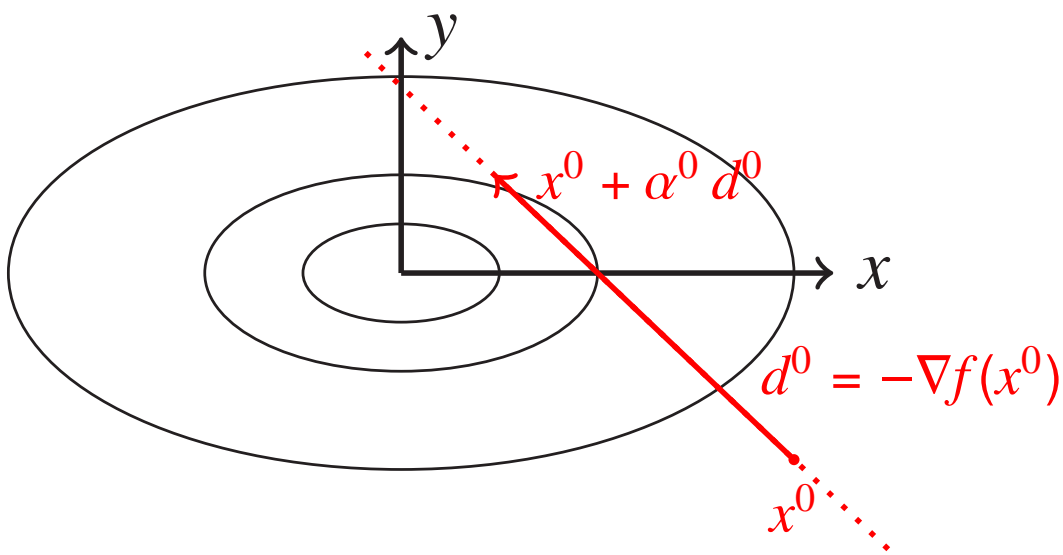
- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$

- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method



# Linear conjugate gradient method

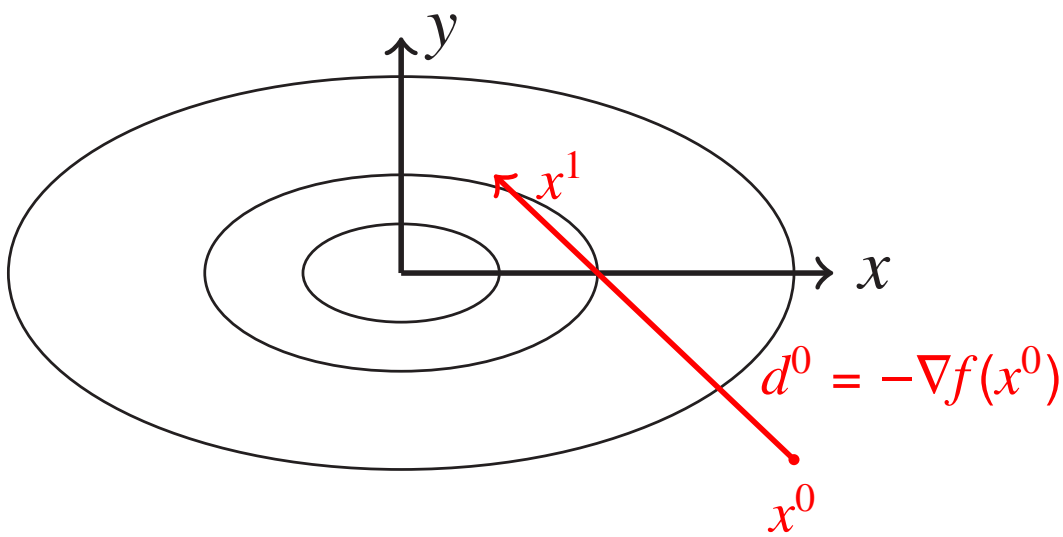
- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$

- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method



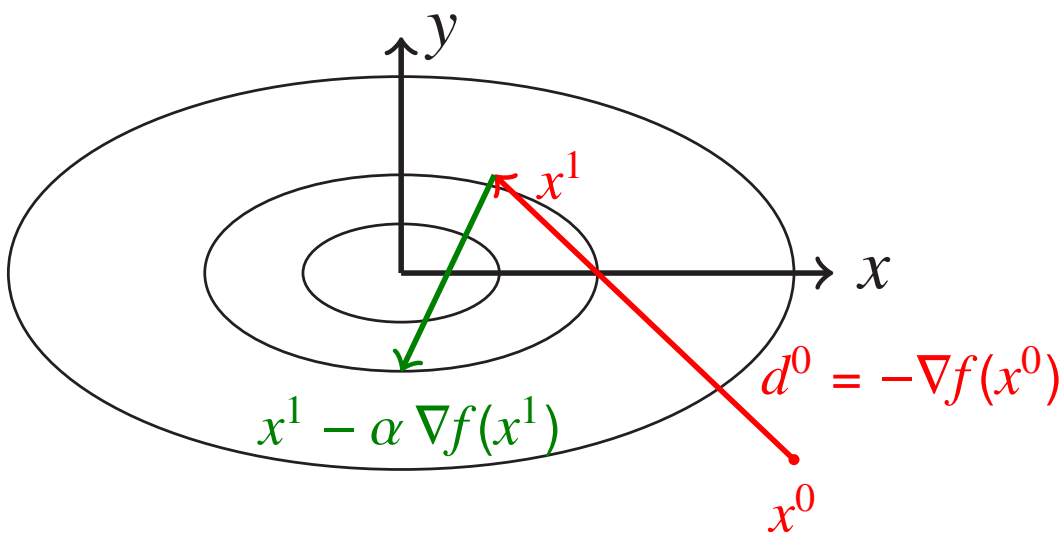
# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$




- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

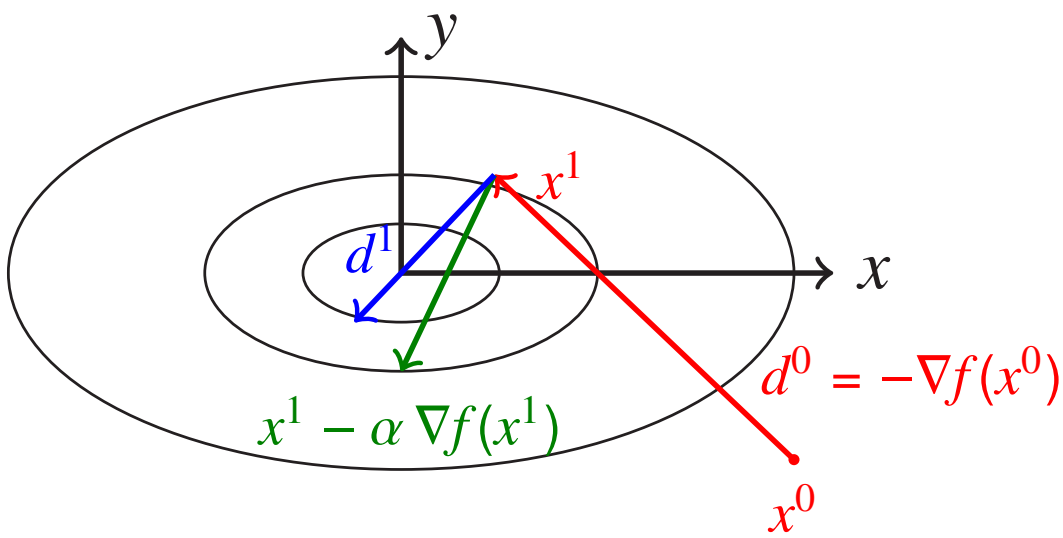
# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$




- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

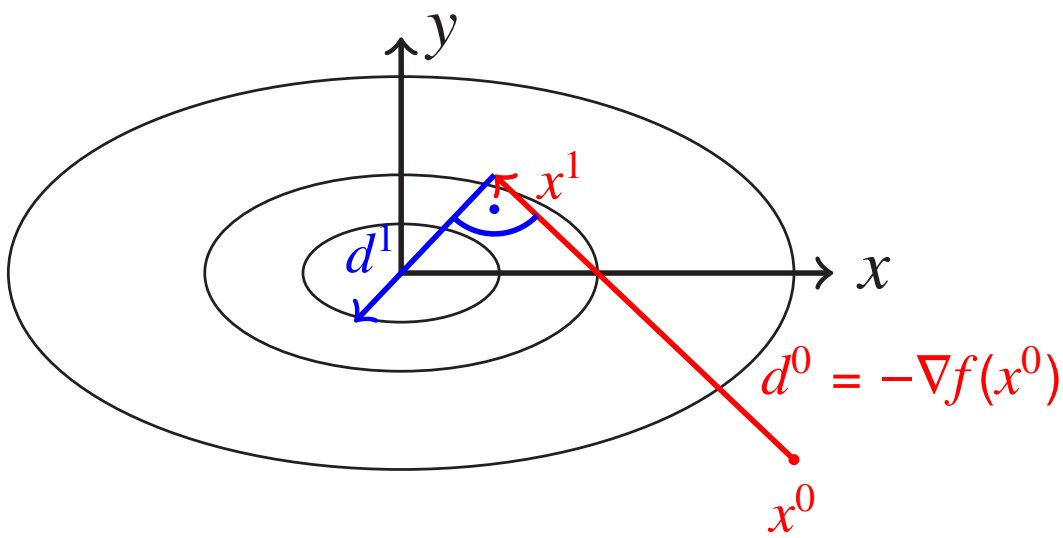
# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$




- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

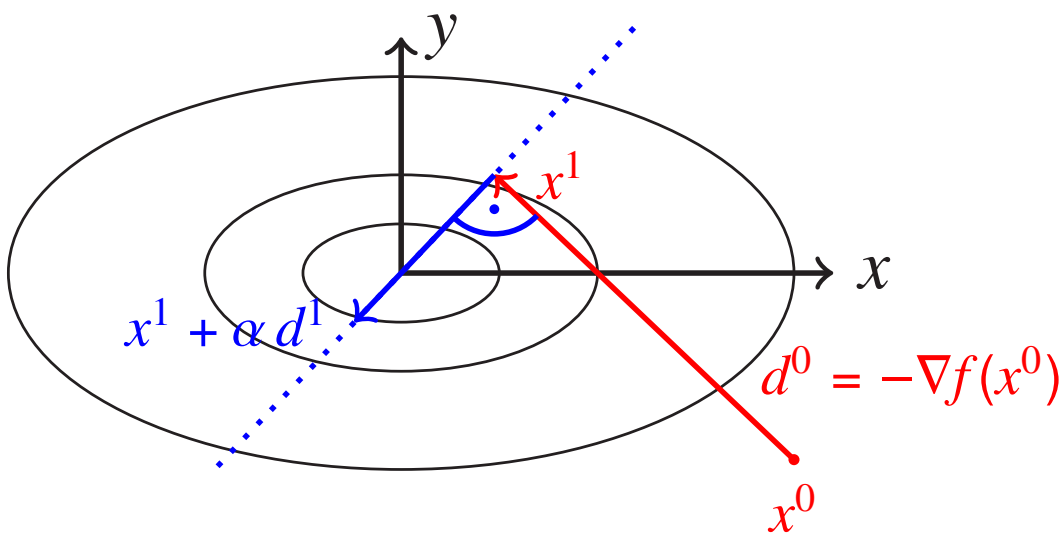
# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \overset{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$




- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

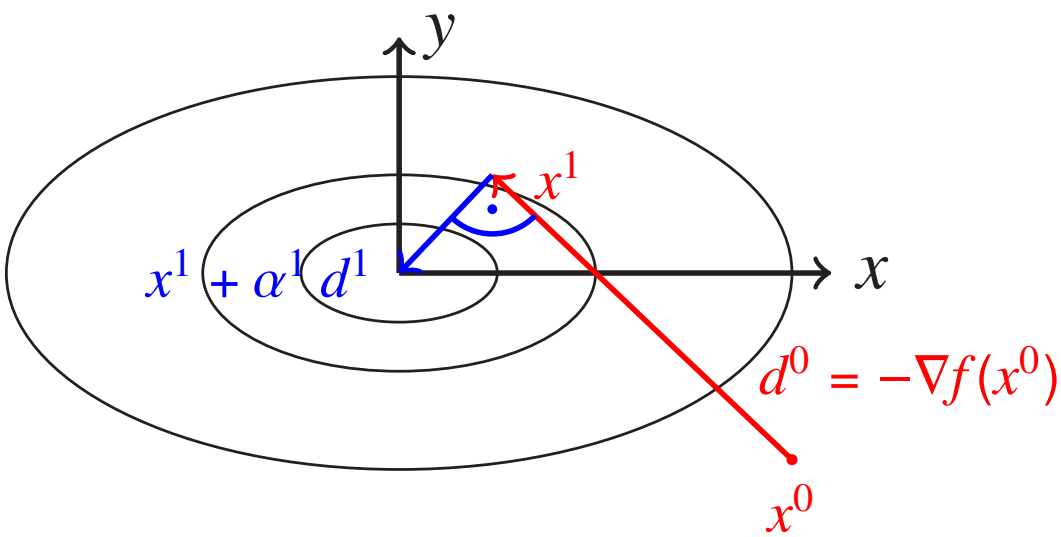
# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$




- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

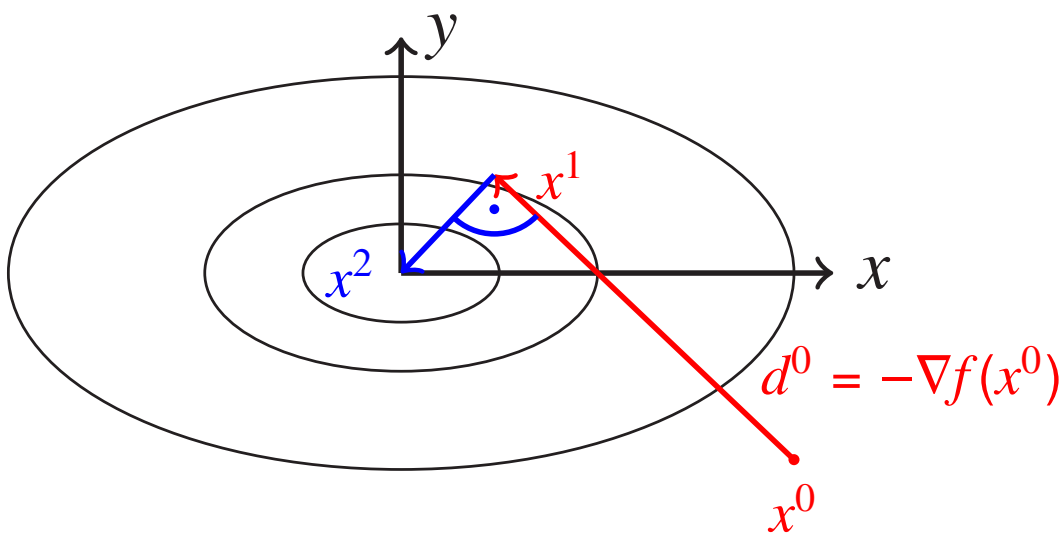
# Linear conjugate gradient method

- goal: $f(x) \longrightarrow \min_x$
- utilize

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$
$$x^{k+1} = x^k + \alpha^k\, d^k$$

- with

$$\alpha^k \stackrel{!}{=} \operatorname{argmin} f(x^k + \alpha\, d^k)$$
$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2$$




- ↗ Hestenes-Stiefel (1952)
- **optimal** Krylov method

# Linear CG & FFT

- goal:
  $\langle(\bar{\varepsilon} + \nabla^s u) : \mathbb{C} : (\bar{\varepsilon} + \nabla^s u)\rangle_Q \to \min$
- introduced by Brisard-Dormieux (2010) and Zeman et al. (2010)
- $\texttt{res} = \|\Gamma : \mathbb{C} : \varepsilon\|/\|\bar{\sigma}\|$
- requires four fields, vs. two fields for basic/heavy ball

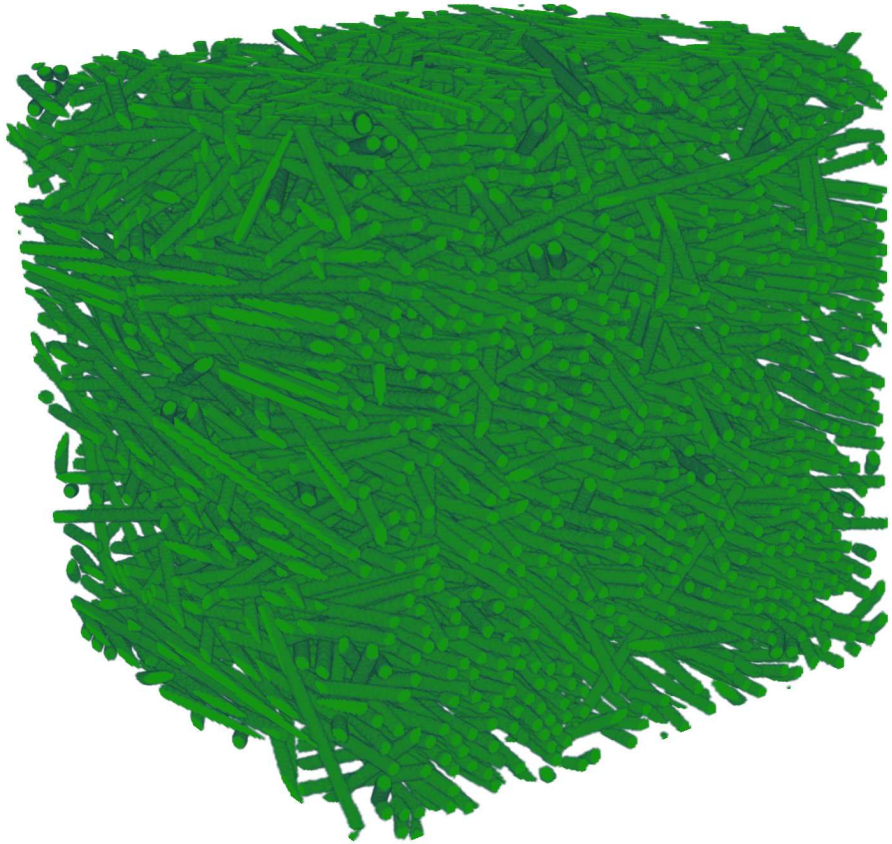
**Algorithm 4** Linear CG $(\mathbb{C}, \bar{\varepsilon}, \text{maxit}, \texttt{tol})$

$G \leftarrow \Gamma : \mathbb{C} : \bar{\varepsilon}$ ▷ Compute $\bar{\sigma} = \langle \mathbb{C} : \bar{\varepsilon} \rangle_Q$
$D \leftarrow -G$
$[\mathtt{r}, \mathtt{res}] \leftarrow [\|G\|, \|G\|/\|\bar{\sigma}\|]$
$k \leftarrow 0$
**while** $k < \text{maxit}$ **and** $\mathtt{res} > \mathtt{tol}$ **do**
  $k \leftarrow k + 1$
  $\mathtt{r}_{\text{old}} \leftarrow \mathtt{r}$
  $Z \leftarrow \Gamma : \mathbb{C} : D$ ▷ $[\triangle\bar{\sigma}, \hat{Z}(0)] \leftarrow [\hat{Z}(0), 0]$
  $\alpha \leftarrow \mathtt{r}^2/(D, Z)_{L^2}$
  $\varepsilon \leftarrow \varepsilon + \alpha D$
  $\bar{\sigma} \leftarrow \bar{\sigma} + \alpha \triangle\bar{\sigma}$
  $\mathtt{res} \leftarrow r/\|\bar{\sigma}\|$
  $G \leftarrow G + \alpha Z$
  $\mathtt{r} \leftarrow \|G\|$
  $\gamma \leftarrow \mathtt{r}^2/\mathtt{r}^2_{\text{old}}$
  $D \leftarrow -G + \gamma D$
**end while**
$\varepsilon \leftarrow \varepsilon + \bar{\varepsilon}$
**return** $\varepsilon, \bar{\sigma}$

[S. Brisard and L. Dormieux, Computational Materials Science, 2010]

[J. Zeman, J. Vondřejc, J. Novák, and I. Marek, Journal of Computational Physics, 2010]

# Practical performance - setup

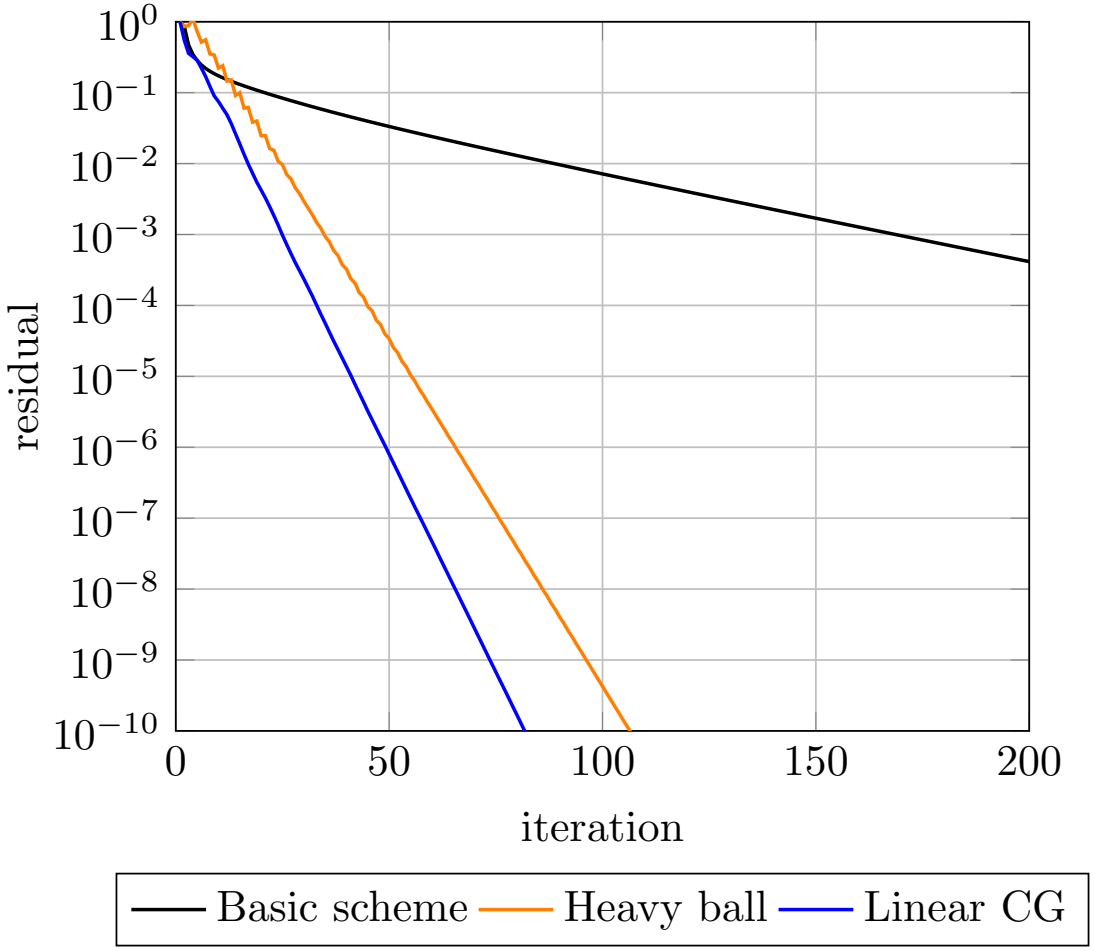


- glass-fiber reinforced PA, $256^3$
- $\phi = 20\%$, $r_a = 30$,
  $A = \mathrm{diag}(0.8, 0.1, 0.1)$
- generated by SAM

  [MS, Computational Mechanics, 2017]
- $E_{\mathrm{Fiber}} = 72$ GPa, $\nu_{\mathrm{Fiber}} = 0.22$,
  $E_{\mathrm{PA}} = 2.1$ GPa, $\nu_{\mathrm{PA}} = 0.3$
- uniaxial extension in $e_1$

[MS, International Journal for Numerical Methods in Engineering, 2019]

# Practical performance - linear elasticity




[MS, International Journal for Numerical Methods in Engineering, 2019]

# Newton's method

- solve

$$G(x) \stackrel{!}{=} 0 \quad \text{via} \quad G(x+d) \approx G(x) + G'(x)\, d \stackrel{!}{=} 0$$

  via

$$x^{k+1} = x^k + \alpha^k\, d^k, \quad G'(x^k)\, d^k = -G(x^k)$$

- locally quadratic convergence ($\alpha^k = 1$) under
  - non-degeneracy of root
  - technical smoothness conditions at root
  - exact computation of $d^k$
- global convergence via globalization strategy, i.e., via backtracking with $\alpha^k \in (0, 1]$

$$\|G(x^k + \alpha^k d^k\| \leq (1-\delta)\, \|G(x^k)\|, \quad \alpha^k = (1-\rho)^m, \quad \rho, \delta \in (0,1)$$

# Newton's method in optimization

- solve

$$\nabla f(x) \stackrel{!}{=} 0 \quad \text{via} \quad \nabla f(x+d) \approx \nabla f(x) + \nabla^2 f(x)\, d \stackrel{!}{=} 0$$

via

$$x^{k+1} = x^k + \alpha^k\, d^k, \qquad \nabla^2 f(x^k)\, d^k = -\nabla f(x^k)$$

# Newton-CG method

- solve

$$\nabla f(x) \stackrel{!}{=} 0 \quad \text{via} \quad \nabla f(x+d) \approx \nabla f(x) + \nabla^2 f(x)\, d \stackrel{!}{=} 0$$

via

$$x^{k+1} = x^k + \alpha^k\, d^k, \quad \nabla^2 f(x^k)\, d^k = -\nabla f(x^k) \quad \leftarrow \quad \text{solved with CG}$$

# Newton-CG method

- solve

$$\nabla f(x) \stackrel{!}{=} 0 \quad \text{via} \quad \nabla f(x+d) \approx \nabla f(x) + \nabla^2 f(x)\, d \stackrel{!}{=} 0$$

via

$$x^{k+1} = x^k + \alpha^k\, d^k, \quad \nabla^2 f(x^k)\, d^k = -\nabla f(x^k) \quad \leftarrow \quad \text{solved with CG}$$

# Newton-CG method

- solve

$$\nabla f(x) \stackrel{!}{=} 0 \quad \text{via} \quad \nabla f(x+d) \approx \nabla f(x) + \nabla^2 f(x)\, d \stackrel{!}{=} 0$$

  via

$$x^{k+1} = x^k + \alpha^k\, d^k, \quad \nabla^2 f(x^k)\, d^k = -\nabla f(x^k) \quad \leftarrow \quad \text{solved with CG}$$

- under same assumptions as Newton provided, if

$$\|\nabla^2 f(x^k)\, d^k + \nabla f(x^k)\| \leq \texttt{const}\, \|\nabla f(x^k)\|^p,$$

  will converge with rate $p \in [1, 2]$
- in practice: $p = 1$, $\texttt{const} = 0.1$
- global convergence via globalization strategy

# Newton-CG & FFT

- solve

$$\operatorname{div} \mathsf{S}(\cdot, \bar{\varepsilon} + \nabla^s (u + v)) \approx \operatorname{div} \mathsf{S}(\cdot, \bar{\varepsilon} + \nabla^s u) + \operatorname{div} \frac{\mathsf{S}}{\partial \varepsilon}(\cdot, \bar{\varepsilon} + \nabla^s u) : \nabla^s v \stackrel{!}{=} 0$$

via

$$u^{k+1} = u^k + \alpha^k v^k,$$

$$\operatorname{div} \frac{\mathsf{S}}{\partial \varepsilon}(\cdot, \bar{\varepsilon} + \nabla^s u^k) : \nabla^s v^k = -\operatorname{div} \mathsf{S}(\cdot, \bar{\varepsilon} + \nabla^s u^k) \quad \leftarrow \quad \text{solved with CG}$$

# Newton-CG & FFT

- solve

$$\operatorname{div} \mathsf{S}(\cdot, \bar{\varepsilon} + \nabla^s (u + v)) \approx \operatorname{div} \mathsf{S}(\cdot, \bar{\varepsilon} + \nabla^s u) + \operatorname{div} \frac{\mathsf{S}}{\partial \varepsilon}(\cdot, \bar{\varepsilon} + \nabla^s u) : \nabla^s v \stackrel{!}{=} 0$$

  via

$$\varepsilon^{k+1} = \varepsilon^k + \alpha^k \xi^k,$$

$$\xi^k = -\Gamma^0 : \left[ \frac{\mathsf{S}}{\partial \varepsilon}(\cdot, \varepsilon^k) : \xi^k - \mathbb{C}^0 : \xi^k + \mathsf{S}(\cdot, \varepsilon^k) \right] \quad \leftarrow \quad \text{solved with CG}$$

  [L. Gélébart and R. Mondon-Cancel, Computational Materials Science, 2013]

- careful with residuals & forcing

  [D. Wicht, MS and T. Böhlke, International Journal for Numerical Methods in Engineering, 2020]

# Newton-CG & FFT

- solve

$$\operatorname{div} \mathsf{S}(\cdot, \bar{\varepsilon} + \nabla^s (u + v)) \approx \operatorname{div} \mathsf{S}(\cdot, \bar{\varepsilon} + \nabla^s u) + \operatorname{div} \frac{\mathsf{S}}{\partial \varepsilon}(\cdot, \bar{\varepsilon} + \nabla^s u) : \nabla^s v \stackrel{!}{=} 0$$

  via

$$\varepsilon^{k+1} = \varepsilon^k + \alpha^k \xi^k,$$

$$\xi^k = -\Gamma^0 : \left[ \frac{\mathsf{S}}{\partial \varepsilon}(\cdot, \varepsilon^k) : \xi^k - \mathbb{C}^0 : \xi^k + \mathsf{S}(\cdot, \varepsilon^k) \right] \quad \leftarrow \quad \text{solved with CG}$$

- memory: 1 ($\varepsilon$) + 4 (CG) + 3.5 (tangent) = 8.5 strain fields
- only **linear convergence** in practice

# Synopsis first two parts

- trade memory vs speed
- high-memory versions: CG / Newton-CG
- low-memory solvers: basic / fast gradient
- **Why** are CG and Newton-CG faster??

# Connections

$$d^k = -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1}$$

CG: $$x^{k+1} = x^k + \alpha^k\, d^k$$

Heavy ball: $$x^{k+1} = x^k - s^k\, \nabla f(x^k) + \beta^k\, (x^k - x^{k-1})$$

# Connections

$$
\begin{aligned}
& & d^k &= -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1} \\
\text{CG:} & & x^{k+1} &= x^k + \alpha^k\, d^k \\
& & d^k &= \frac{1}{\alpha^k}\left(x^{k+1} - x^k\right)
\end{aligned}
$$

$$
\text{Heavy ball:} \quad x^{k+1} = x^k - s^k\, \nabla f(x^k) + \beta^k\, (x^k - x^{k-1})
$$

# Connections

$$
\begin{aligned}
& & d^k &= -\nabla f(x^k) + \gamma^{k-1}\, d^{k-1} \\
\text{CG:} & & x^{k+1} &= x^k + \alpha^k\, d^k \\
& & d^{k-1} &= \frac{1}{\alpha^{k-1}} \left( x^k - x^{k-1} \right)
\end{aligned}
$$

$$
\text{Heavy ball:} \quad x^{k+1} = x^k - s^k\, \nabla f(x^k) + \beta^k\, (x^k - x^{k-1})
$$

# Connections

$$d^k = -\nabla f(x^k) + \frac{\gamma^{k-1}}{\alpha^{k-1}} \left( x^k - x^{k-1} \right)$$

CG: $$x^{k+1} = x^k + \alpha^k \, d^k$$

Heavy ball: $$x^{k+1} = x^k - s^k \, \nabla f(x^k) + \beta^k \, (x^k - x^{k-1})$$

# Connections

$$
\begin{aligned}
& & d^k &= -\nabla f(x^k) + \frac{\gamma^{k-1}}{\alpha^{k-1}} \left( x^k - x^{k-1} \right) \\
\text{CG:} & & x^{k+1} &= x^k + \alpha^k \left( -\nabla f(x^k) + \frac{\gamma^{k-1}}{\alpha^{k-1}} \left( x^k - x^{k-1} \right) \right)
\end{aligned}
$$

$$
\text{Heavy ball:} \quad x^{k+1} = x^k - s^k \nabla f(x^k) + \beta^k \left( x^k - x^{k-1} \right)
$$

# Connections

CG: $x^{k+1} = x^k - \alpha^k \nabla f(x^k) + \frac{\alpha^k \gamma^{k-1}}{\alpha^{k-1}} \left( x^k - x^{k-1} \right)$

Heavy ball: $x^{k+1} = x^k - s^k \nabla f(x^k) + \beta^k \left( x^k - x^{k-1} \right)$

# Connections

CG: $x^{k+1} = x^k - \alpha^k \nabla f(x^k) + \frac{\alpha^k \gamma^{k-1}}{\alpha^{k-1}} \left( x^k - x^{k-1} \right)$

Heavy ball: $x^{k+1} = x^k - s^k \nabla f(x^k) + \beta^k \left( x^k - x^{k-1} \right)$

$\Rightarrow \qquad \alpha^k = s^k \quad \text{and} \quad \frac{\alpha^k \gamma^{k-1}}{\alpha^{k-1}} = \beta^k$

# Connections

CG: $x^{k+1} = x^k - \alpha^k \nabla f(x^k) + \frac{\alpha^k \gamma^{k-1}}{\alpha^{k-1}} \left( x^k - x^{k-1} \right)$

Heavy ball: $x^{k+1} = x^k - s^k \nabla f(x^k) + \beta^k \left( x^k - x^{k-1} \right)$

$\Rightarrow \qquad \alpha^k = s^k \quad \text{and} \quad \frac{\alpha^k \gamma^{k-1}}{\alpha^{k-1}} = \beta^k$

# HB $\Rightarrow$ Linear CG

- plane search

$$(s^k, \beta^k) = \operatorname{argmin}_{s,\beta} \quad f\left(x^k - s\,\nabla f(x^k) + \beta\,(x^k - x^{k-1})\right)$$

- CG = heavy ball with **adaptive parameters**

# HB $\Rightarrow$ Linear CG

- plane search

$$(s^k, \beta^k) = \operatorname{argmin}_{s,\beta} \quad f\left(x^k - s\,\nabla f(x^k) + \beta\,(x^k - x^{k-1})\right)$$

- CG = heavy ball with **adaptive parameters**

# Similarly

$$\begin{aligned} \text{Newton:} \quad & x^{k+1} = x^k - \alpha^k \left[\nabla^2 f(x^k)\right]^{-1} \nabla f(x^k) \\ \text{Gradient method:} \quad & x^{k+1} = x^k - s^k \nabla f(x^k) \end{aligned}$$

↗ Newton is gradient descent with **adaptive (anisotropic) step size**

# Overview

1. Accelerated gradient methods

2. Newton - CG

3. Adaptive parameter selection

4. Summary and conclusions

# Barzilai-Borwein method

- gradient scheme

$$x^{k+1} = x^k - s^k \nabla f(x^k)$$

- 1D secant method:

$$s^k = \frac{x^k - x^{k-1}}{f'(x^k) - f'(x^{k-1})} \approx \frac{1}{f''(x^k)}$$

- higher dimension (Barzilai-Borwein, 1988):

$$s^k = \frac{\|x^k - x^{k-1}\|^2}{\langle \nabla f(x^k) - \nabla f(x^{k-1}), x^k - x^{k-1} \rangle}$$

[J. Barzilai and J. M. Borwein, IMA Journal of Numerical Analysis, 1988]

# Barzilai-Borwein method

$$x^{k+1} = x^k - s^k \nabla f(x^k) \quad \text{with} \quad s^k = \frac{\|x^k - x^{k-1}\|^2}{\langle \nabla f(x^k) - \nabla f(x^{k-1}), x^k - x^{k-1} \rangle}$$

- if

$$\alpha_- \leq \lambda \leq \alpha_+ \quad \forall x \forall \lambda \in \mathtt{Eig}\left(\nabla^2 f(x)\right),$$

then

$$\frac{1}{\alpha_+} \leq s^k \leq \frac{1}{\alpha_-}$$

# Barzilai-Borwein method

$$x^{k+1} = x^k - s^k \nabla f(x^k) \quad \text{with} \quad s^k = \frac{\|x^k - x^{k-1}\|^2}{\langle \nabla f(x^k) - \nabla f(x^{k-1}), x^k - x^{k-1} \rangle}$$

- practical implementation

$$s^k = \frac{\|\nabla f(x^{k-1})\|^2}{\|\nabla f(x^{k-1})\|^2 - \langle \nabla f(x^k), \nabla f(x^{k-1}) \rangle} s^{k-1}$$

# Barzilai-Borwein method & FFT

$$\varepsilon^{k+1} = \varepsilon^k - s^k \, \Gamma : \mathsf{S}(\cdot, \varepsilon^k) \quad \text{with} \quad \left\langle \varepsilon^0 \right\rangle_Q = \bar{\varepsilon}$$

- practical implementation

$$s^k = \frac{\| \Gamma : \mathsf{S}(\cdot, \varepsilon^{k-1}) \|^2}{\| \Gamma : \mathsf{S}(\cdot, \varepsilon^{k-1}) \|^2 - \left( \mathsf{S}(\cdot, \varepsilon^k), \Gamma : \mathsf{S}(\cdot, \varepsilon^{k-1}) \right)_{L^2}} \, s^{k-1}$$

[MS, International Journal for Numerical Methods in Engineering, 2019]

# Barzilai-Borwein method & FFT

$$\xi^k = \Gamma : S(\cdot, \varepsilon^k)$$
$$\varepsilon^{k+1} = \varepsilon^k - s^k \xi^k \quad \text{with} \quad \left\langle \varepsilon^0 \right\rangle_Q = \bar{\varepsilon}$$

- practical implementation

$$s^k = \frac{\|\xi^{k-1}\|^2}{\|\xi^{k-1}\|^2 - (S(\cdot, \varepsilon^k), \xi^{k-1}))_{L^2}} s^{k-1}$$

[MS, International Journal for Numerical Methods in Engineering, 2019]

# Basic scheme II

**Algorithm 5** Alternative basic scheme $(\bar{\varepsilon}, \texttt{maxit}, \texttt{tol}, \alpha_0)$

$[\varepsilon, \xi] \leftarrow [\varepsilon^0, 0]$ ▷ $\left\langle \varepsilon^0 \right\rangle_Q \overset{!}{=} \bar{\varepsilon}$

$[\texttt{res}, s, k, \texttt{r}] \leftarrow [1, 1/\alpha^0, 0, 1]$

**while** $k < \texttt{maxit}$ **and** $\texttt{res} > \texttt{tol}$ **do**

  $k \leftarrow k + 1$

  $\xi \leftarrow \mathsf{S}(\cdot, \varepsilon)$

  $\bar{\sigma} \leftarrow \langle \xi \rangle_Q$

  $\xi \leftarrow \Gamma : \xi$ ▷ $\hat{\xi}(0) = 0$

  $\varepsilon \leftarrow \varepsilon - s\,\xi$ ▷ use FFT & favorite discretization

  $\texttt{r} \leftarrow \|\xi\|$

  $\texttt{res} \leftarrow \texttt{r}/\|\bar{\sigma}\|$

**end while**

**return** $\varepsilon, \bar{\sigma}, \texttt{res}$ ▷ Requires two strain fields

# Barzilai Borwein basic scheme

**Algorithm 6** Barzilai Borwein basic scheme $(\bar{\varepsilon}, \texttt{maxit}, \texttt{tol}, \alpha_0)$

$[\varepsilon, \xi] \leftarrow [\varepsilon^0, 0]$ ▷ $\left\langle \varepsilon^0 \right\rangle_Q \overset{!}{=} \bar{\varepsilon}$

$[\texttt{res}, s, k, \texttt{r}] \leftarrow [1, 1/\alpha^0, 0, 1]$

**while** $k < \texttt{maxit}$ **and** $\texttt{res} > \texttt{tol}$ **do**

  $k \leftarrow k + 1$

  $\begin{bmatrix} \xi \\ \texttt{p} \end{bmatrix} \leftarrow \begin{bmatrix} \mathsf{S}(\cdot, \varepsilon) \\ (\mathsf{S}(\cdot, \varepsilon), \xi)_{L^2} \end{bmatrix}$

  $\bar{\sigma} \leftarrow \langle \xi \rangle_Q$

  $\xi \leftarrow \Gamma : \xi$ ▷ $\hat{\xi}(0) = 0$

  $s \leftarrow s\,/(1 - \texttt{p}/\texttt{r}^2)$

  $\varepsilon \leftarrow \varepsilon - s\,\xi$ ▷ use FFT & favorite discretization

  $\texttt{r} \leftarrow \|\xi\|$

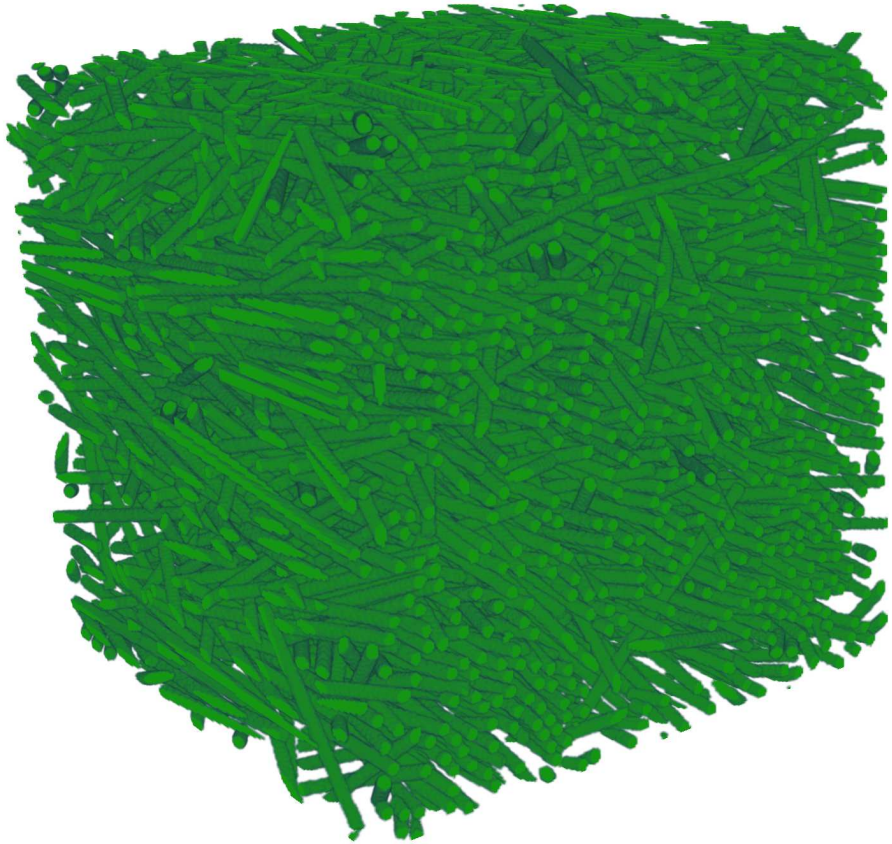
  $\texttt{res} \leftarrow \texttt{r}/\|\bar{\sigma}\|$

**end while**

**return** $\varepsilon, \bar{\sigma}, \texttt{res}$ ▷ Requires two strain fields

[MS, International Journal for Numerical Methods in Engineering, 2019]

# Practical performance - setup

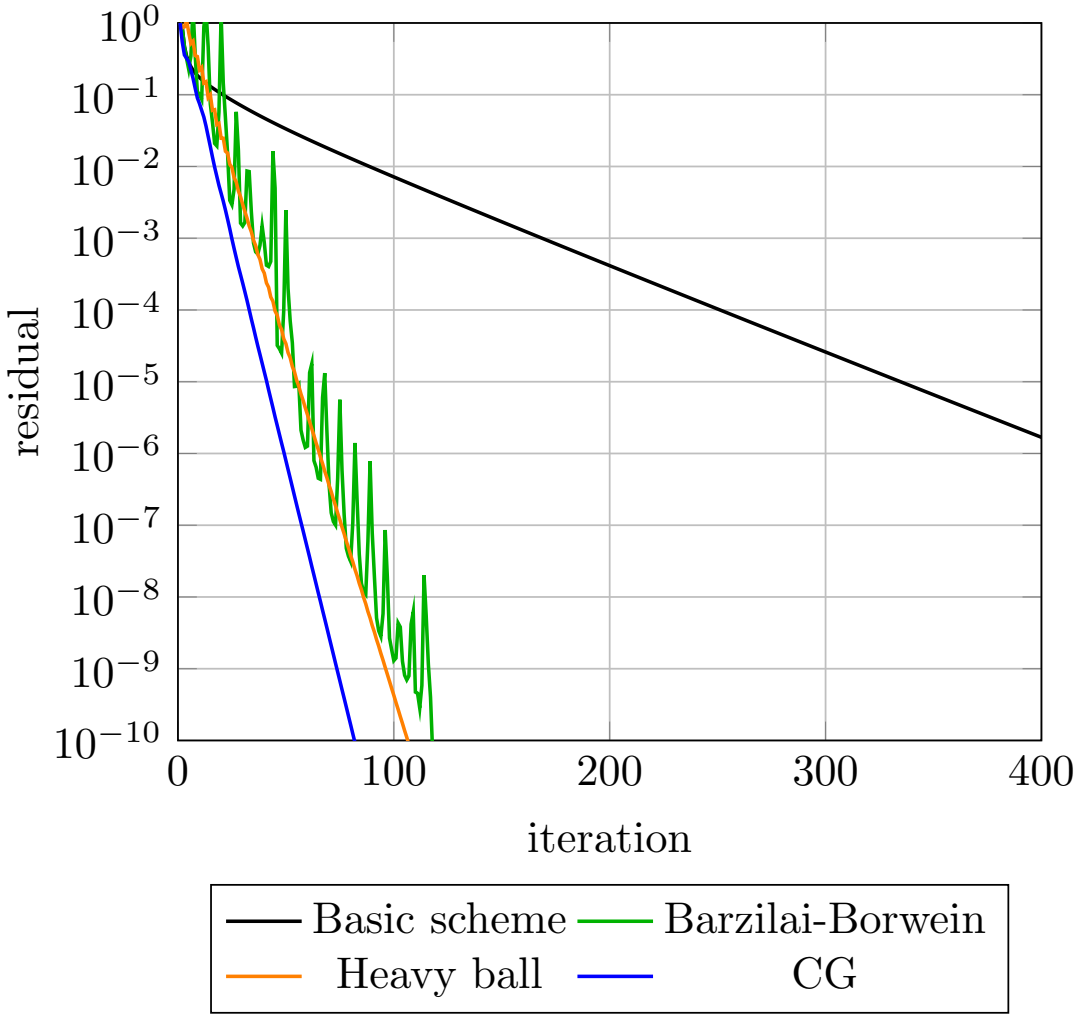


- glass-fiber reinforced PA, $256^3$
- $\phi = 20\%$, $r_a = 30$,
  $A = \mathrm{diag}(0.8, 0.1, 0.1)$
- generated by SAM

  [MS, Computational Mechanics, 2017]
- $E_{\mathrm{Fiber}} = 72$ GPa, $\nu_{\mathrm{Fiber}} = 0.22$,
  $E_{\mathrm{PA}} = 2.1$ GPa, $\nu_{\mathrm{PA}} = 0.3$
- uniaxial extension in $e_1$

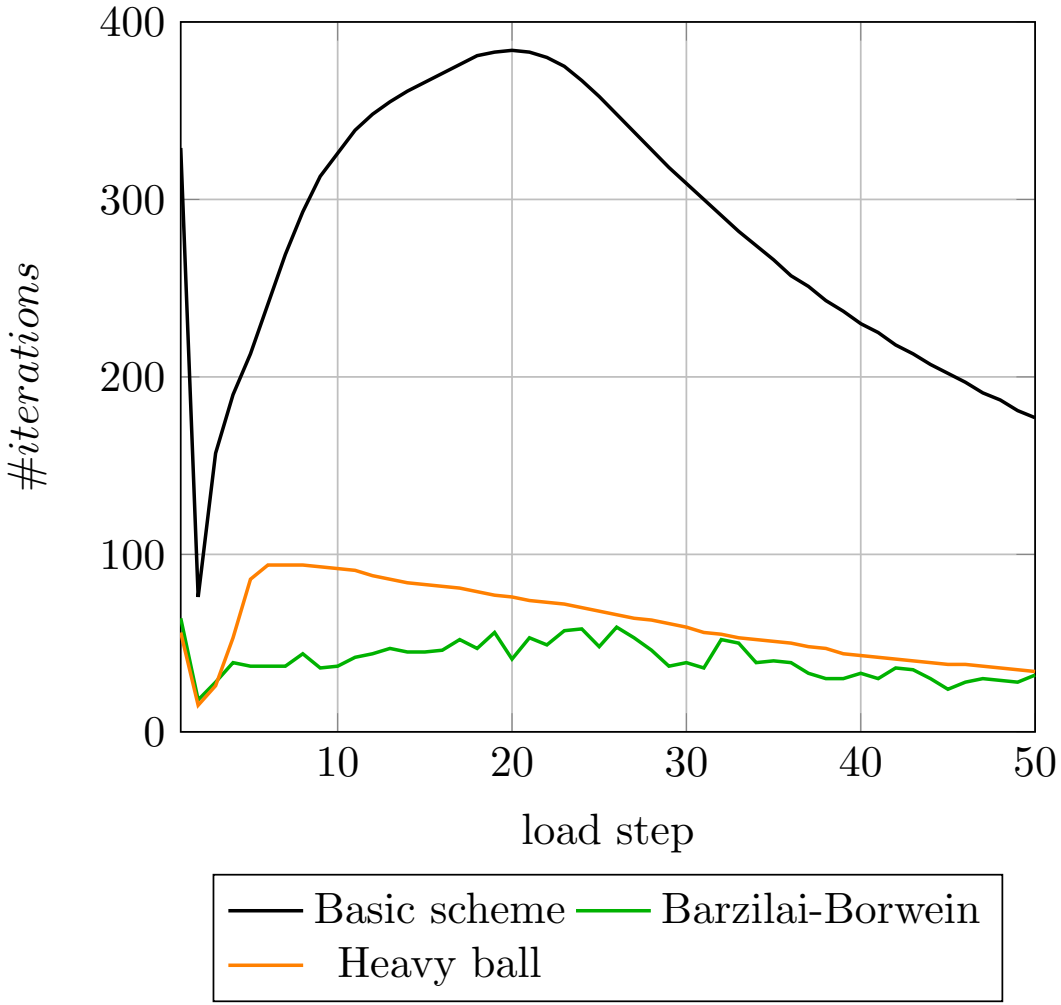
[MS, International Journal for Numerical Methods in Engineering, 2019]

# Practical performance - linear elasticity




[MS, International Journal for Numerical Methods in Engineering, 2019]

# Practical performance - vM plasticity




- von Mises elastoplastic matrix
- 5% uniaxial extension in x

average it.

| | Basic | HB | BB |
|---|---|---|---|
| $128^3$ | 284.08 | 64.64 | 40.58 |
| $256^3$ | 382.7 | 61.58 | 49.6 |

[MS, International Journal for Numerical Methods in Engineering, 2019]

# Synopsis Barzilai-Borwein

- simple to implement
- two strain fields
- no manual update of $\alpha^0$
- no eigenvalue decompositions (!)
- but: no monotonicity

# Heavy ball vs. CG

$$x^{k+1} = x^k - s^k \nabla f(x^k) + \beta^k (x^k - x^{k-1}), \quad \beta^k = s^k \gamma^{k-1} / s^{k-1}$$

- heavy ball: $s^k = \texttt{const}$, $\beta^k = \texttt{const}$, requires **meticulous care**
- CG: line search ↗ **bottleneck**

$$s^k \approx \operatorname{argmin} f(x^k + s\, d^k)$$

and

$$\gamma^k = \|\nabla f(x^{k+1})\|^2 / \|\nabla f(x^k)\|^2 \quad \text{(Fletcher-Reeves, 1964)}$$

[R. Fletcher and C. Reeves, The Computer Journal, 1964]

# Heavy ball + CG

$$x^{k+1} = x^k - s^k \nabla f(x^k) + \beta^k (x^k - x^{k-1})$$

- combine advantages

$$s^k = \texttt{const}$$
$$\beta^k = \|\nabla f(x^k)\|^2 / \|\nabla f(x^{k-1})\|^2$$

# Heavy ball + CG & FFT

$$\varepsilon^{k+1} = \varepsilon^k - s^k \, \Gamma : \mathsf{S}(\cdot, \varepsilon^k) + \beta^k \, (\varepsilon^k - \varepsilon^{k-1})$$

- combine advantages

$$s^k = \texttt{const}$$

$$\beta^k = \|\Gamma : \mathsf{S}(\cdot, \varepsilon^k)\|^2 / \|\Gamma : \mathsf{S}(\cdot, \varepsilon^{k-1})\|^2$$

# Heavy ball + CG & FFT

$$\xi^k = \Gamma : \mathsf{S}(\cdot, \varepsilon^k)$$
$$\varepsilon^{k+1} = \varepsilon^k - s^k \, \xi^k + \beta^k \, (\varepsilon^k - \zeta^k)$$
$$\zeta^{k+1} = \varepsilon^k$$

- combine advantages

$$s^k = \texttt{const}$$
$$\beta^k = \|\xi^k\|^2 / \|\xi^{k-1}\|^2$$

# Nonlinear CG ↗ [MS, Computational Mechanics, 2020]

Université franco-allemande Deutsch-Französische Hochschule

Karlsruhe Institute of Technology

## Algorithm 7 Fletcher-Reeves Nonlinear CG ($\bar{\varepsilon}, \texttt{maxit}, \texttt{tol}, \alpha_0$)

1: $[\varepsilon, \xi, \zeta] \leftarrow [\varepsilon^0, 0, \zeta]$ ▷ $\left\langle \varepsilon^0 \right\rangle_Q \overset{!}{=} \bar{\varepsilon}$

2: $[\texttt{res}, s, k, \texttt{r}, \beta] \leftarrow [1, 1/\alpha^0, 0, 1, 0]$

3: **while** $k < \texttt{maxit}$ **and** $\texttt{res} > \texttt{tol}$ **do**

4: $\quad k \leftarrow k + 1$

5: $\quad \xi \leftarrow \mathsf{S}(\cdot, \varepsilon)$

6: $\quad \bar{\sigma} \leftarrow \langle \xi \rangle_Q$

7: $\quad \xi \leftarrow \Gamma : \xi$ ▷ $\hat{\xi}(0) = 0$

8: $\quad \texttt{r} \leftarrow \|\xi\|$

9: $\quad \beta \leftarrow \texttt{r}^2 \beta$

10: $\quad \begin{bmatrix} \varepsilon \\ \zeta \end{bmatrix} \leftarrow \begin{bmatrix} \varepsilon - s\,\xi + \beta(\varepsilon - \zeta) \\ \varepsilon \end{bmatrix}$

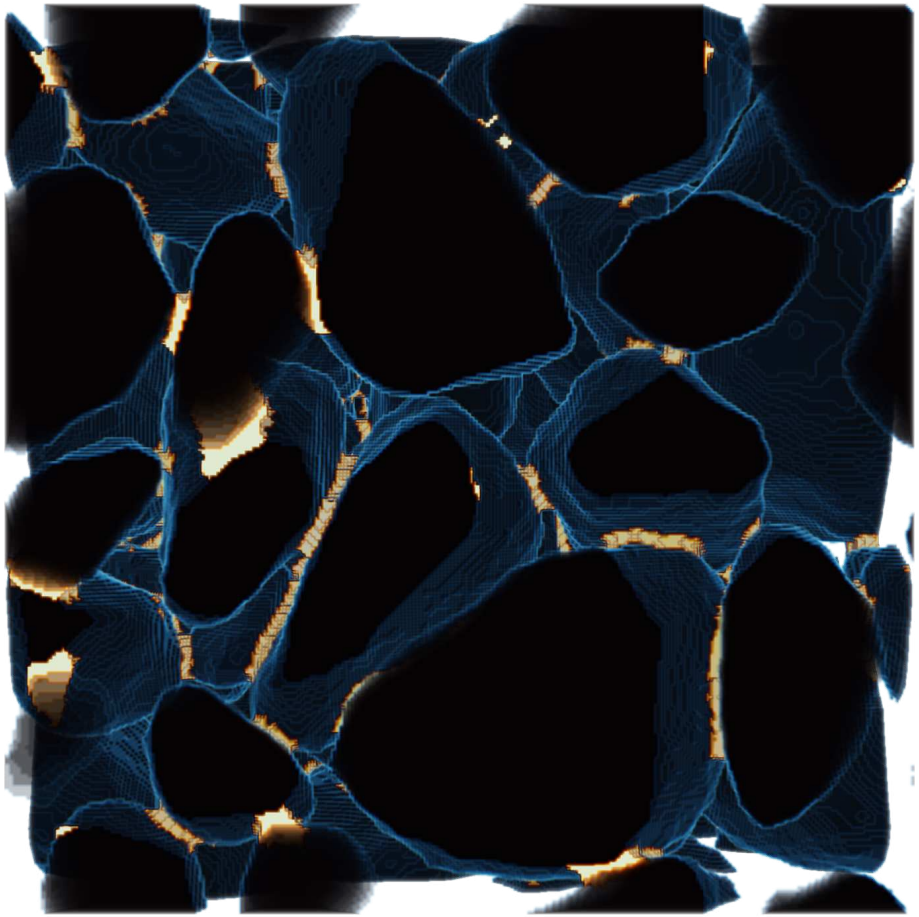
11: $\quad \beta \leftarrow 1/\texttt{r}^2$

12: $\quad \texttt{res} \leftarrow \texttt{r}/\|\bar{\sigma}\|$

13: **end while**

14: **return** $\varepsilon, \bar{\sigma}, \texttt{res}$ ▷ Requires three strain fields

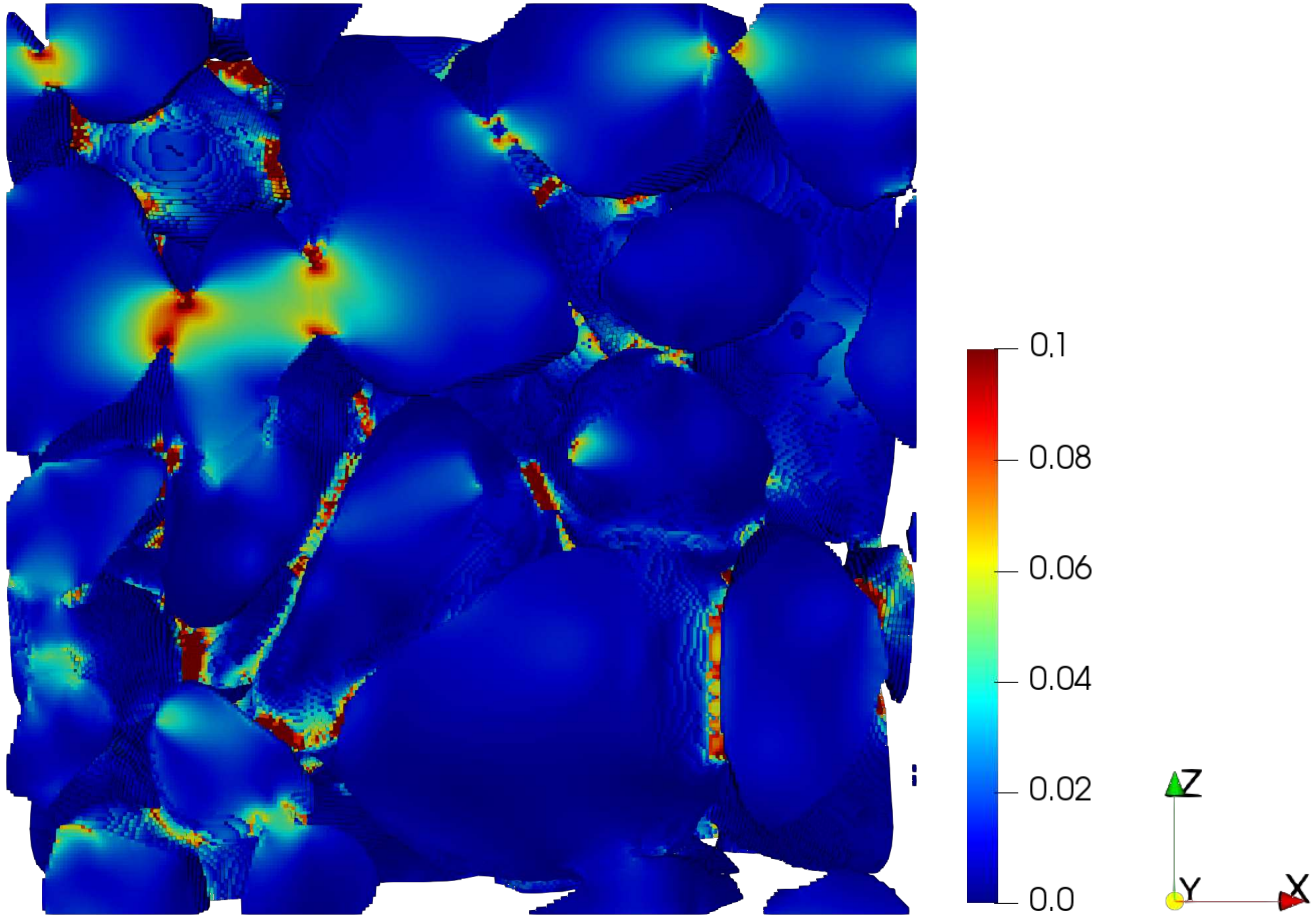
# Practical performance - sandcore



Sand (black) and binder (gold)

↗ [MS, Computational Mechanics, 2020]

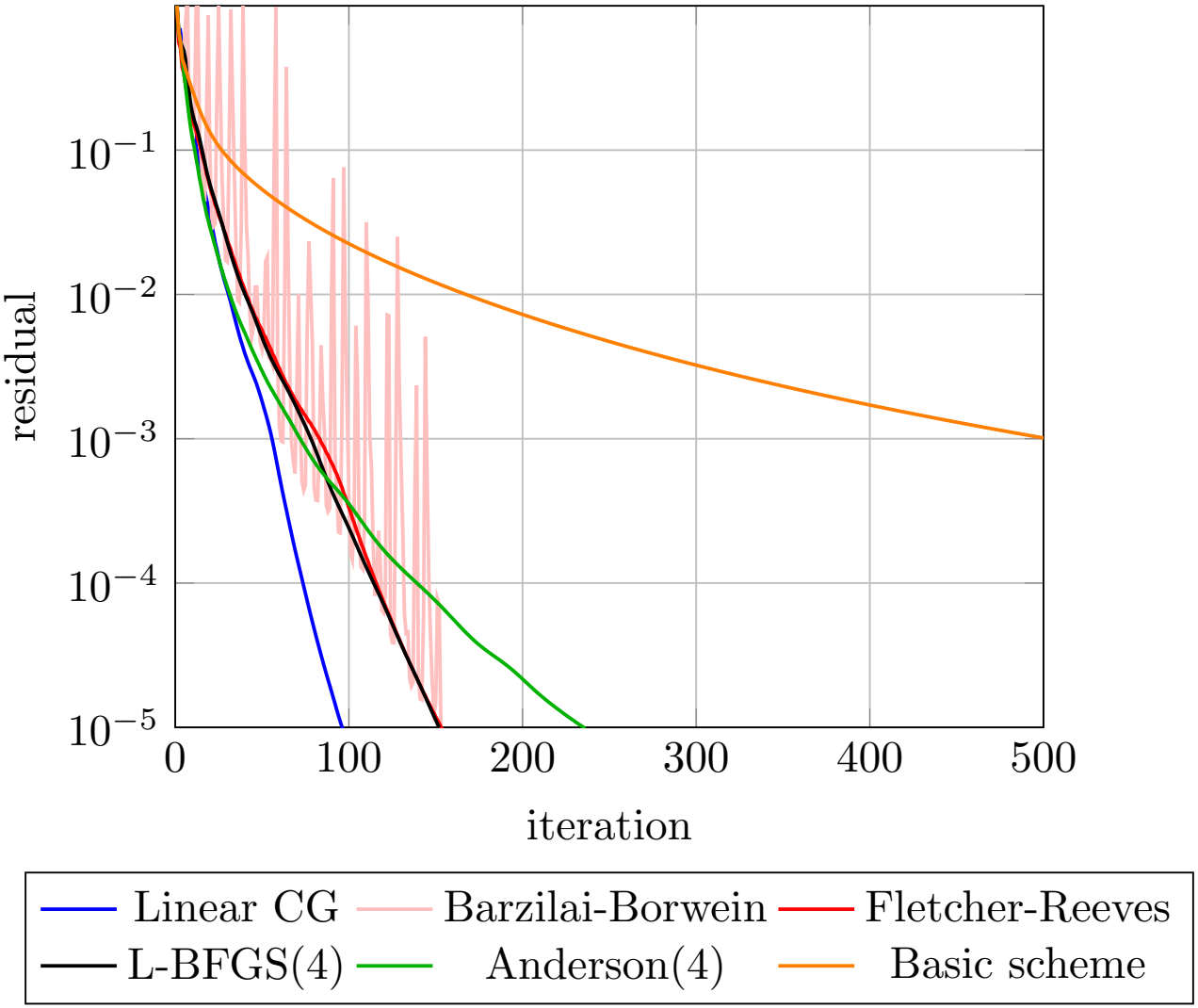
# Practical performance - sandcore




Strain magnitude at 5%, on **staggered grid** [MS-Ospald-Kabel, IJNME, 2016]

↗ [MS, Computational Mechanics, 2020]

# Practical performance - sandcore



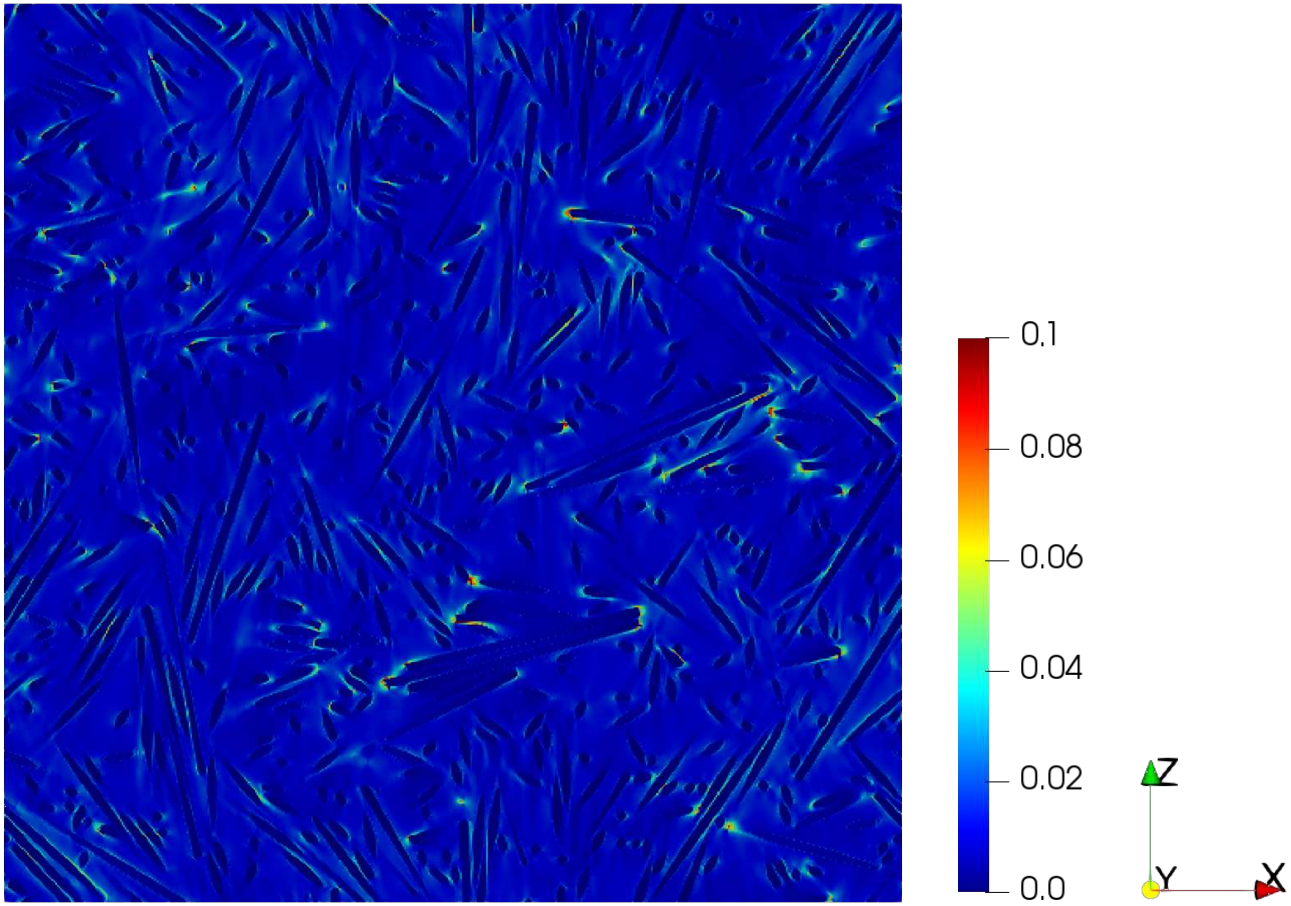

↗ [MS, Computational Mechanics, 2020]

# Practical performance - FRP

Planar isotropic SFRP

↗ [MS, Computational Mechanics, 2020]

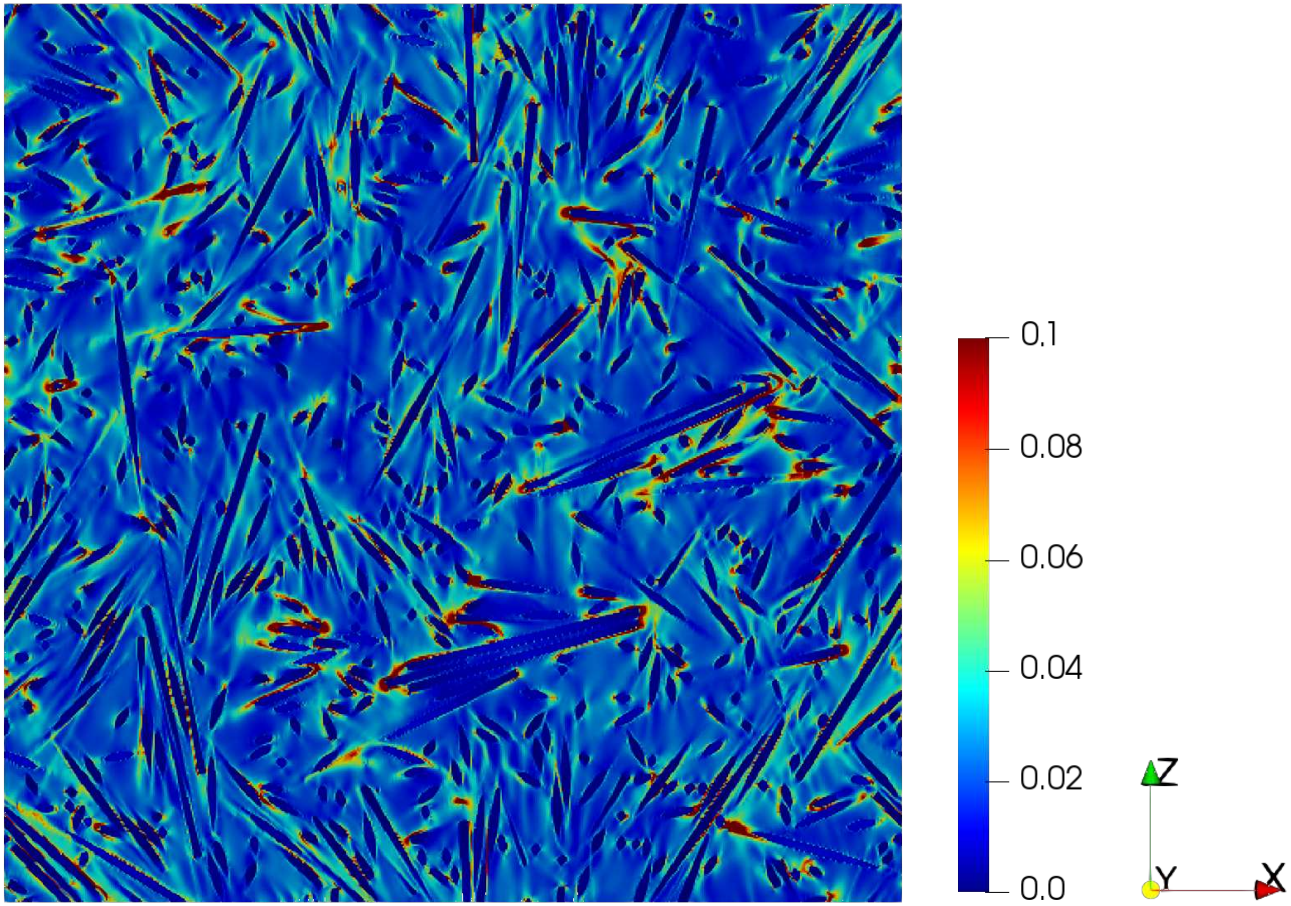
# Practical performance - FRP




Planar isotropic SFRP - strain magnitude @ 1% strain

↗ [MS, Computational Mechanics, 2020]

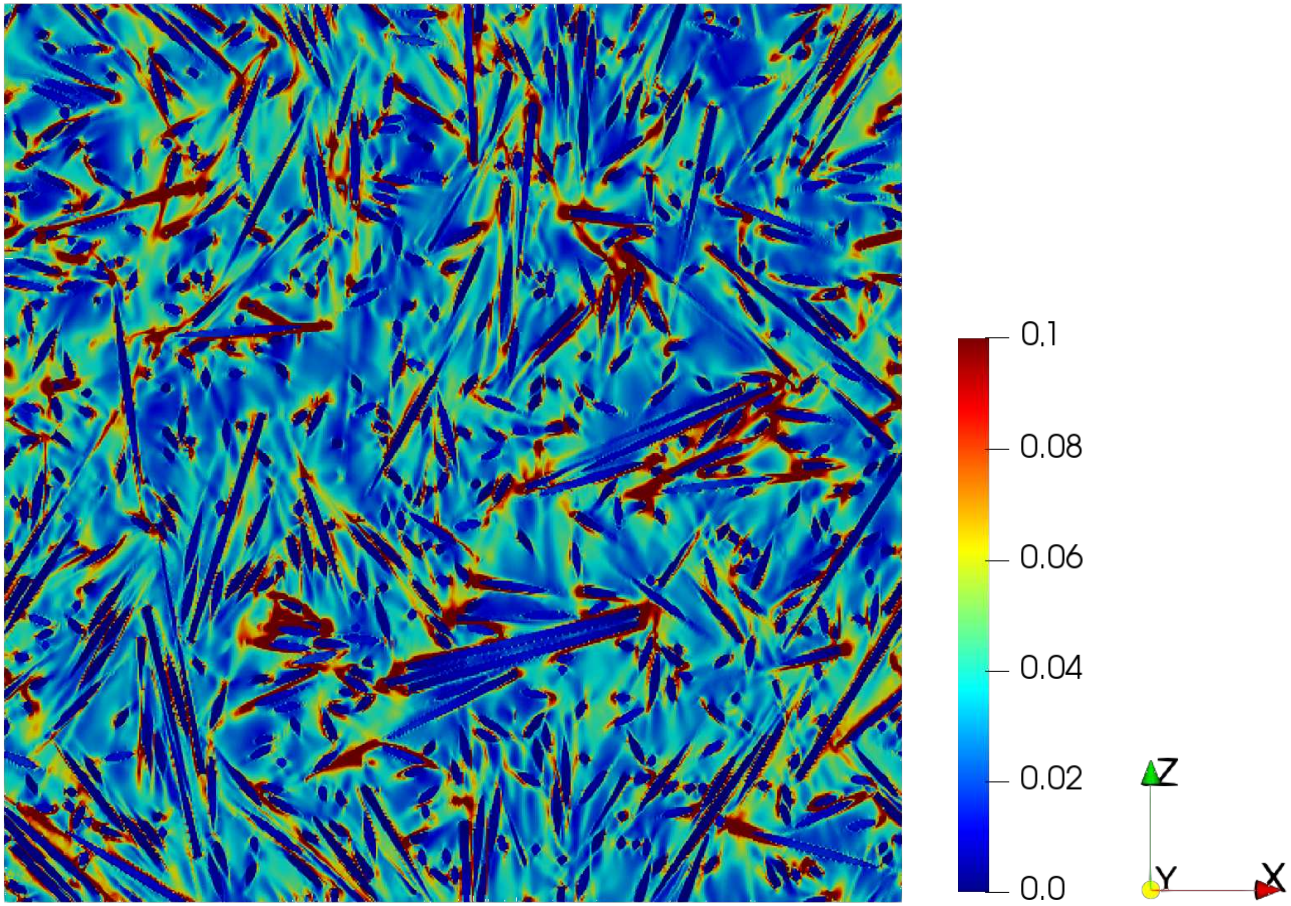
# Practical performance - FRP




Planar isotropic SFRP - strain magnitude @ 2% strain

↗ [MS, Computational Mechanics, 2020]

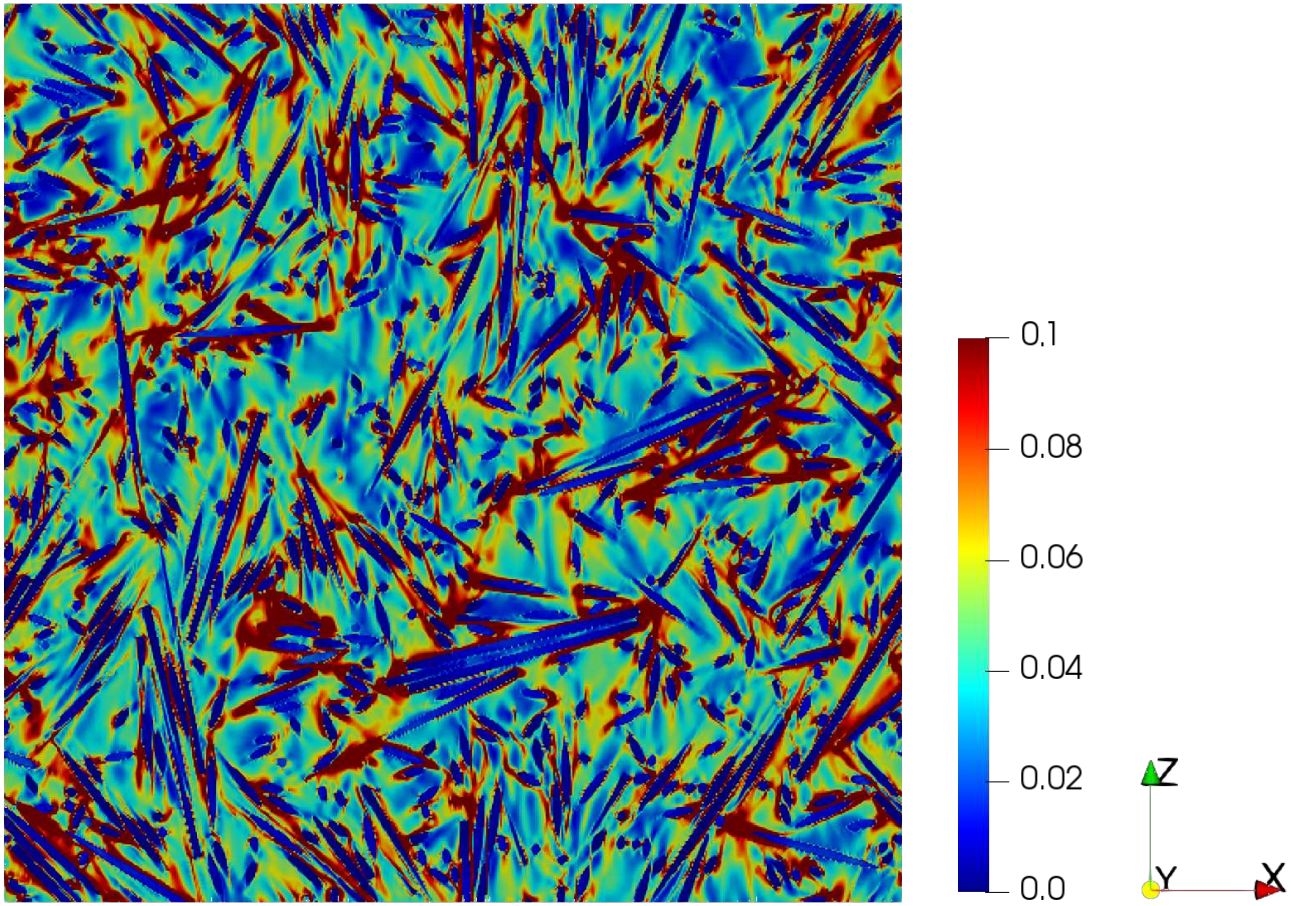
# Practical performance - FRP




Planar isotropic SFRP - strain magnitude @ 3% strain

↗ [MS, Computational Mechanics, 2020]

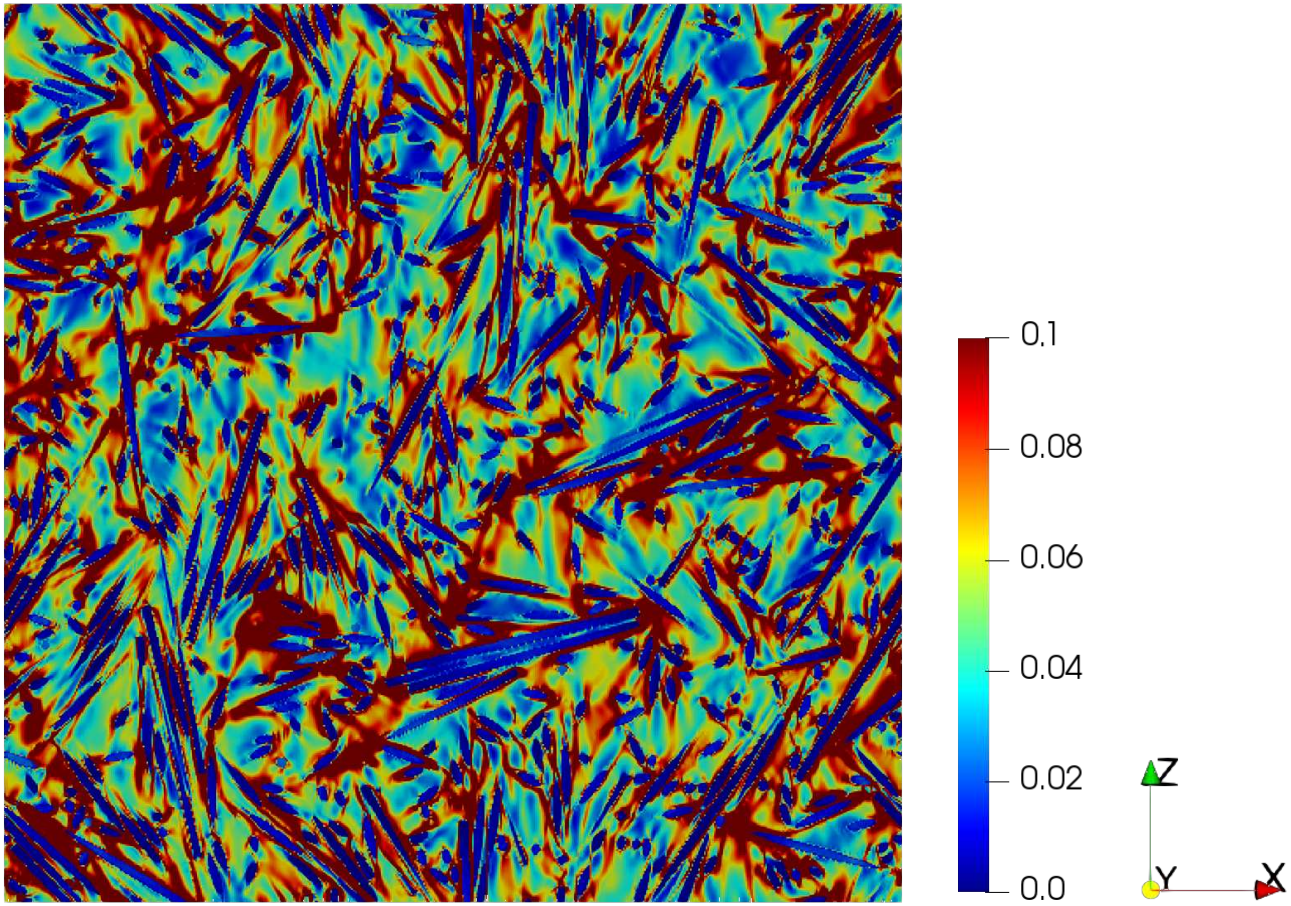
# Practical performance - FRP




Planar isotropic SFRP - strain magnitude @ 4% strain

↗ [MS, Computational Mechanics, 2020]

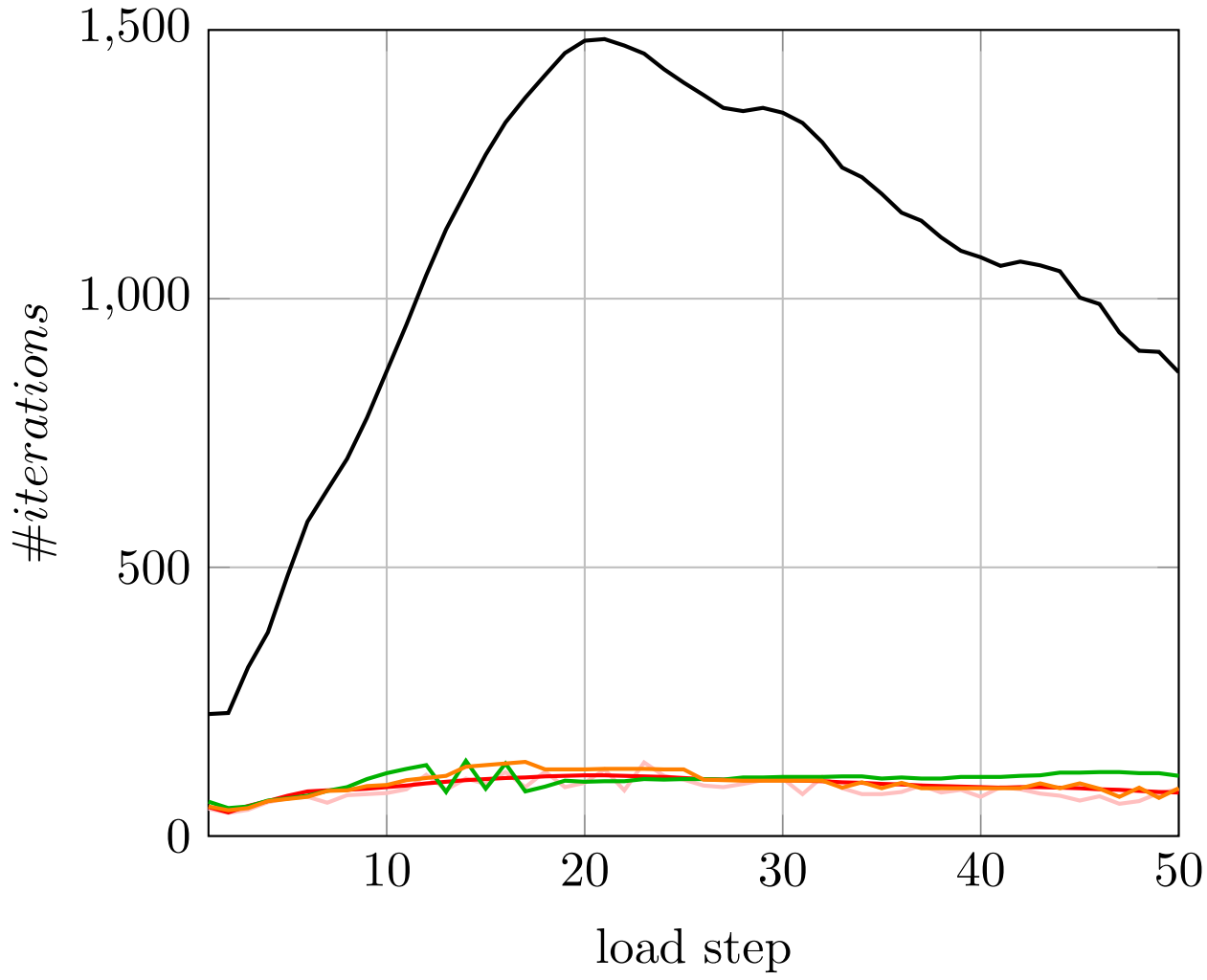
# Practical performance - FRP




Planar isotropic SFRP - strain magnitude @ 5% strain

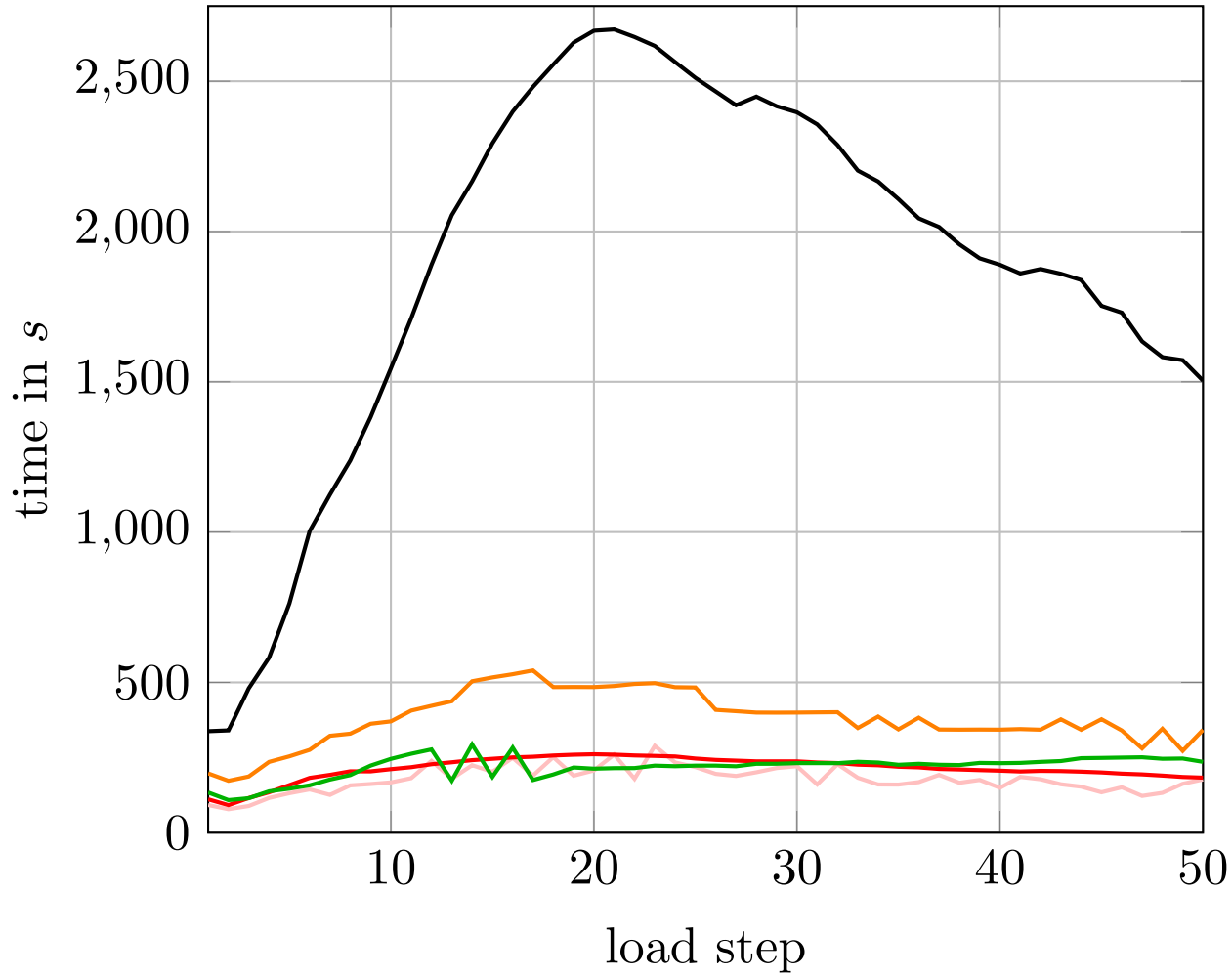
↗ [MS, Computational Mechanics, 2020]

# Practical performance - FRP - iterations




↗ [MS, Computational Mechanics, 2020]

# Practical performance - FRP - run time




↗ [MS, Computational Mechanics, 2020]

# Overview

1. Accelerated gradient methods

2. Newton - CG

3. Adaptive parameter selection

4. Summary and conclusions

# Summary

- speed vs. memory
- **adaptive** parameter selection **!**
- recommendations:

| material law | finite material contrast | with pores |
|---|---|---|
| linear | linear CG | linear CG |
| cheap | BB | BB |
| | polarization | Nonlinear CG |
| expensive | Newton-CG | Newton-CG |
| | Nonlinear CG | Nonlinear CG |

↗ [MS, "A review of nonlinear FFT-based computational homogenization methods", Acta Mechanica, 2021]

- not covered: Anderson (↗ L. Gélébart)
- not covered: polarization schemes (next lecture)

# More details . . .

Acta Mech 232, 2051–2100 (2021)
https://doi.org/10.1007/s00707-021-02962-1

REVIEW AND PERSPECTIVE IN MECHANICS

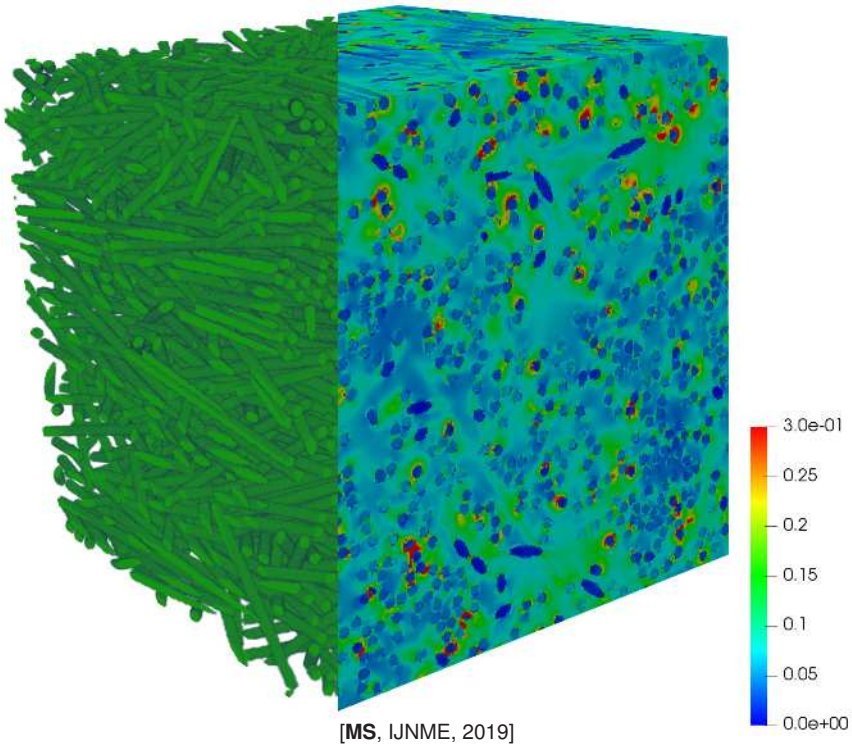
Matti Schneider

## A review of nonlinear FFT-based computational homogenization methods

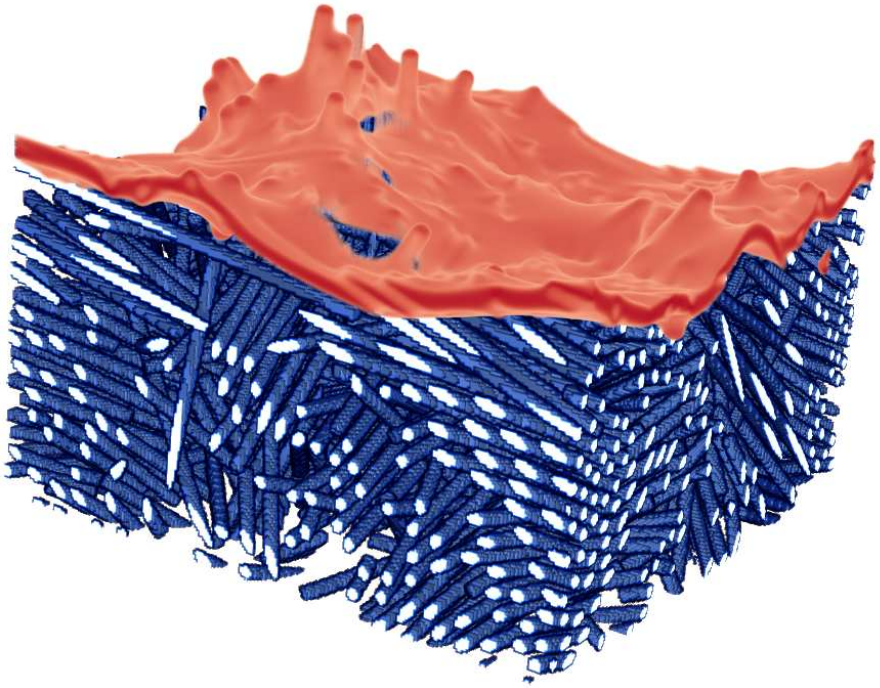
Received: 9 December 2020 / Revised: 18 January 2021 / Accepted: 16 February 2021 / Published online: 24 March 2021


**Abstract** Since their inception, computational homogenization methods based on the fast Fourier transform (FFT) have grown in popularity, establishing themselves as a powerful tool applicable to complex, digitized microstructures. At the same time, the understanding of the underlying principles has grown, in terms of both discretization schemes and solution methods, leading to improvements of the original approach and extending the applications. This article provides a condensed overview of results scattered throughout the literature and guides the reader to the current state of the art in nonlinear computational homogenization methods using the fast Fourier transform.

# The end




[**MS**, IJNME, 2019]



[Ernesti-**MS**-Böhlke, CMAME, 2020]

matti.schneider@kit.edu