FFT-Based Crystal Plasticity for Polycrystalline Systems

From Atoms to Algorithms:
Physical Model • Mathematical Formulation • GPU Implementation

Santiago Garcia Botero

University of Texas at San Antonio

February 16, 2026

Roadmap

This talk covers three tightly coupled layers:

I. **Physical Model** — What happens inside a metal when you deform it? Crystals, dislocations, slip systems, thermal activation, hardening.

II. **Mathematical Formulation** — How do we turn physics into equations? Equilibrium PDE, Lippmann–Schwinger, Green's operator, Fourier methods, stress control.

III. **Numerical Implementation** — How do we solve it fast and correctly? FFT algorithms, CG acceleration, Anderson mixing, discrete derivatives, GPU batching, EVPFFT.

IV. **Validation & Results** — Does it work? Homogeneous test, polycrystal benchmarks, plasticity fields.

Goal: A self-contained GPU-accelerated crystal plasticity simulator with an interactive GUI.

Outline

Why Crystal Plasticity?

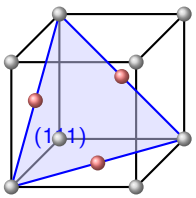
Engineering metals (steel, aluminium, nickel superalloys, titanium) are **polycrystalline**: thousands of tiny single crystals (grains) bonded together, each with a different lattice orientation.

The engineering problem:

- How does a turbine blade deform under load at 850 °C?
- Where will fatigue cracks nucleate?
- How does sheet metal thin during stamping?

Why not just FEA with J2 plasticity?

- J2 knows nothing about grain structure.
- Misses anisotropy, texture, hot-spot localisation.
- Crystal plasticity resolves *every grain*.



FCC unit cell with (111) slip plane highlighted in blue.

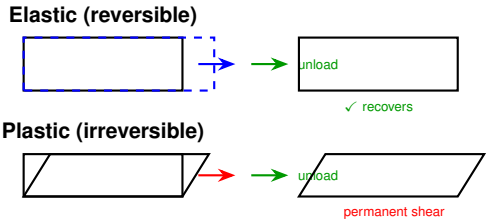
Elastic vs. Plastic Deformation

Elastic deformation (reversible):

- Stretching of atomic bonds.
- Remove load $\rightarrow$ shape fully recovers.
- Governed by Hooke's law: $\boldsymbol{\sigma} = \boldsymbol{C} : \varepsilon$.

Plastic deformation (irreversible):

- Motion of **dislocations** — line defects in the crystal lattice.
- Remove load $\rightarrow$ permanent shape change remains.
- Occurs on specific crystallographic planes and directions.

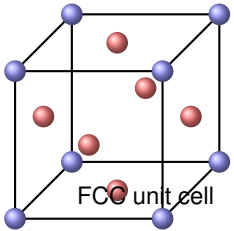


The FCC Crystal Structure

Most engineering metals (Cu, Al, Ni, Au, γ-Fe) have the **face-centered cubic (FCC)** structure.

Key facts:

- 4 atoms per unit cell.
- Densest-packed planes: $\{111\}$ family (4 planes).
- Densest-packed directions: $\langle 110 \rangle$ family.
- Dislocations move most easily on dense planes in dense directions.



Anisotropy:
Even a single crystal has directional stiffness:

$$A = \frac{2C_{44}}{C_{11} - C_{12}}$$

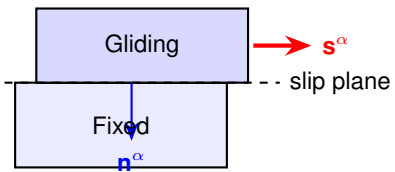
$A = 1$ would be isotropic. Cu has $A = 3.21$ — very anisotropic!

	C_{11} (GPa)	C_{12} (GPa)	C_{44} (GPa)
Cu	168.4	121.4	75.4
Al	107.3	60.9	28.3
Ni	246.5	147.3	124.7

Slip Systems: Where Plastic Deformation Happens

A **slip system** α is a pair: slip plane normal $\mathbf{n}^\alpha$ and slip direction $\mathbf{s}^\alpha$.

FCC metals have 12 slip systems:
4 planes $\{111\} \times$ 3 directions $\langle 110 \rangle$



The amount of slip on system α is γ^α.

The 12 FCC slip systems:

α	Plane **n**	Dir **s**
1–3	(111)	$[01\bar{1}]$, $[\bar{1}01]$, $[1\bar{1}0]$
4–6	$(\bar{1}11)$	$[01\bar{1}]$, $[101]$, $[\bar{1}\bar{1}0]$
7–9	$(1\bar{1}1)$	$[011]$, $[\bar{1}01]$, $[110]$
10–12	$(11\bar{1})$	$[011]$, $[101]$, $[\bar{1}10]$

4 close-packed $\{111\}$ planes
$\times$ 3 directions each = 12 systems.

The Schmid Tensor: Projecting Stress onto Slip

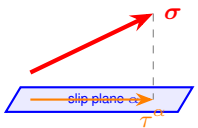
Not all stress drives slip equally. The **resolved shear stress** on system α is the projection of the full stress tensor onto the slip plane:

$$\boxed{\tau^\alpha = \boldsymbol{\sigma} : \boldsymbol{P}^\alpha = \sigma_{ij}\, P_{ij}^\alpha}$$

where the **Schmid tensor** is:

$$\boldsymbol{P}^\alpha = \tfrac{1}{2}\left(\mathbf{s}^\alpha \otimes \mathbf{n}^\alpha + \mathbf{n}^\alpha \otimes \mathbf{s}^\alpha\right)$$

- $\tau^\alpha > 0$: forward slip.
- $\tau^\alpha < 0$: backward slip.
- $|\tau^\alpha|$ small $\rightarrow$ system inactive.



The RSS is the component of stress resolved onto the slip direction.

How Much Slip? The Flow Rule

Question: Given τ^α, how fast does dislocation glide occur?

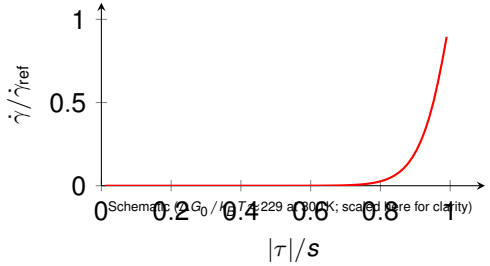
Dislocation motion requires overcoming an **energy barrier** ΔG. At finite temperature T, thermal fluctuations help:

$$\dot{\gamma}^\alpha = \dot{\gamma}_{\text{ref}} \exp\left[-\frac{\Delta G_0}{k_B T}\left(1 - \left(\frac{|\tau^\alpha|}{s^\alpha}\right)^p\right)^q\right] \operatorname{sgn}(\tau^\alpha)$$

This is the **Kocks–Argon–Ashby thermally-activated flow rule**, implemented as the OTIS model in our code.

Parameters:

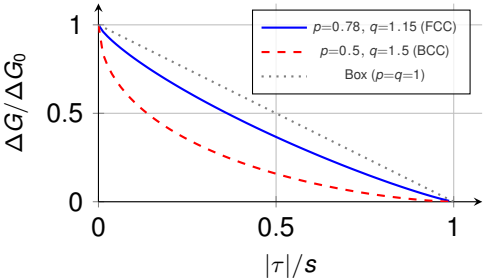
- $\dot{\gamma}_{\text{ref}} = 10^7\ \text{s}^{-1}$
- $\Delta G_0 = 9.5 \times 10^{-19}\ \text{J}$
- $p = 0.78$, $q = 1.15$ (barrier shape)
- s^α: slip resistance (evolves)

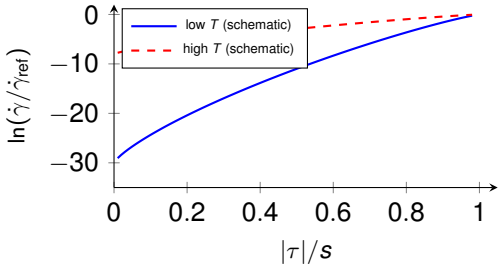


Schematic ($\Delta G_0 / k_B T \approx 229$ at 300 K; scaled here for clarity)

Energy Barrier and Temperature Sensitivity

The barrier shape (p, q) controls **rate sensitivity**:





Higher $T \rightarrow$ activation at lower τ/s.
Coefficients scaled for visual clarity;
actual $\Delta G_0/k_B T \approx 229$ (300 K), ≈ 61 (1123 K).

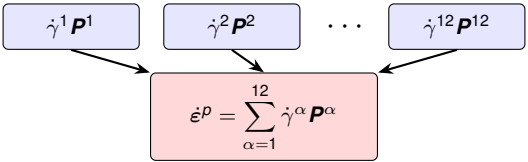
- At low τ/s: barrier is high $\rightarrow$ negligible slip.
- Near $\tau \approx s$: barrier vanishes $\rightarrow$ rapid slip.
- Higher T: thermal energy $k_B T$ helps overcome the barrier at lower stress.

Plastic Strain from Multiple Slip Systems

The total **plastic strain rate** is the sum of contributions from all active systems:

$$\dot{\varepsilon}^p = \sum_{\alpha=1}^{12} \dot{\gamma}^\alpha \, \boldsymbol{P}^\alpha$$

Each system contributes a simple shear along its Schmid tensor. The superposition produces **complex, general deformation**.



Each system contributes a simple shear.
Their sum gives the full plastic strain tensor.

Hardening: Why Metals Get Stronger

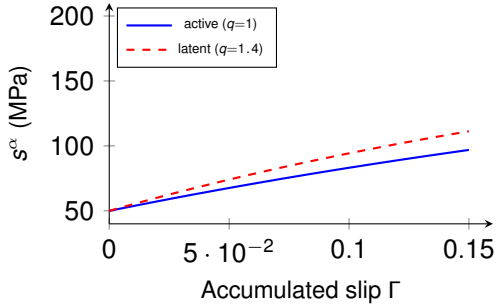
As plastic deformation proceeds, **dislocation density increases**. Dislocations interact and impede each other $\rightarrow$ the crystal **hardens**.

Voce hardening law [1]:

$$\dot{s}^{\alpha} = \sum_{\beta} h_{\alpha\beta} \, |\dot{\gamma}^{\beta}|, \qquad h_{\alpha\beta} = q_{\alpha\beta} \, h_0 \Big(1 - \frac{s^{\beta}}{s_s}\Big)$$

Parameter	Symbol	Cu value
Initial resistance	τ_0	50 MPa
Saturation resistance	s_s	200 MPa
Initial hardening rate	h_0	500 MPa
Latent hardening ratio	q_{lat}	1.4

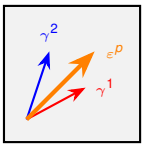
Self-hardening ($\alpha{=}\beta$): $q = 1$.
Latent ($\alpha{\neq}\beta$): $q = 1.4$ (cross-slip interactions).



Latent hardening is stronger: inactive systems harden faster!

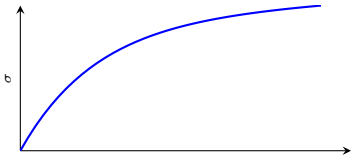
Multi-Slip and Emergent Macroscopic Behavior

Multi-slip:
Under general loading, multiple slip systems activate simultaneously. Their combined effect produces the observed macroscopic deformation.



Two systems combine to produce general shear

Emergent behavior:
Many microscopic slip events (discrete) produce smooth macroscopic stress-strain curves (continuum).



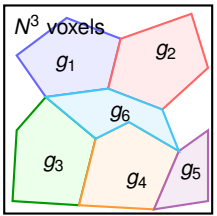
Discrete slip → smooth macroscopic curve

Polycrystalline Microstructure

A real metal contains 10^3–10^6 grains. We represent a **Representative Volume Element (RVE)** on a regular $N \times N \times N$ grid.

Voronoi tessellation (periodic):

1. Seed N_g random points in $[0, 1)^3$.
2. Replicate to 3^3 periodic images.
3. Assign each voxel to nearest seed (KD-tree).
4. Random Bunge Euler angles $(\varphi_1, \Phi, \varphi_2)$ using SO(3) Haar measure.



Each grain g has rotation $\mathbf{R}_g$ that rotates stiffness and slip systems into the sample frame:

$$C_{ijkl}^{\text{sample}} = R_{ip} R_{jq} R_{kr} R_{ls}\, C_{pqrs}^{\text{crystal}}$$

Local Frames and Slip System Rotation

Each grain has its own crystallographic frame. To compute the resolved shear stress, we rotate each slip system into the **sample frame**:

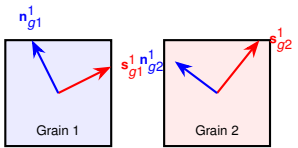
$$\mathbf{s}_g^\alpha = \mathbf{R}_g\, \mathbf{s}^\alpha_{\text{crystal}}, \qquad \mathbf{n}_g^\alpha = \mathbf{R}_g\, \mathbf{n}^\alpha_{\text{crystal}}$$

The Schmid tensor in the sample frame becomes:

$$P^\alpha_{g,ij} = \tfrac{1}{2}\left(s^\alpha_{g,i}\, n^\alpha_{g,j} + n^\alpha_{g,i}\, s^\alpha_{g,j}\right)$$

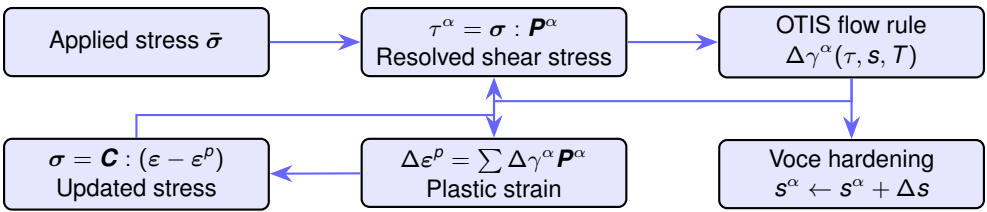
This means:
Two grains under the *same* stress tensor will activate *different* slip systems with *different* magnitudes.

$\rightarrow$ **Stress and strain heterogeneity** within the polycrystal.



Same system α=1, different orientations in each grain

Part I Summary: The Physical Picture



Every voxel in the RVE runs this loop independently. The challenge: how to ensure **equilibrium** and **compatibility** across the entire microstructure?

Problem Statement: Equilibrium on an RVE

Given a periodic RVE with spatially varying stiffness $\boldsymbol{C}(\mathbf{x})$ and prescribed macroscopic stress $\bar{\boldsymbol{\sigma}}$, find the local fields satisfying:

$$\begin{aligned}
\text{Equilibrium:} \quad & \nabla \cdot \boldsymbol{\sigma}(\mathbf{x}) = \mathbf{0} && \text{(E1)} \\
\text{Constitutive:} \quad & \boldsymbol{\sigma}(\mathbf{x}) = \boldsymbol{C}(\mathbf{x}) : \varepsilon(\mathbf{x}) && \text{(E2)} \\
\text{Compatibility:} \quad & \varepsilon = \mathrm{sym}(\nabla \mathbf{u}), \quad \mathbf{u} \text{ periodic} && \text{(E3)} \\
\text{Average stress:} \quad & \langle \boldsymbol{\sigma} \rangle = \bar{\boldsymbol{\sigma}} && \text{(E4)}
\end{aligned}$$

Key difficulty: $\boldsymbol{C}(\mathbf{x})$ varies *sharply* between grains — each grain has a rotated anisotropic stiffness.

This is the same PDE that FEM solves — we solve it differently (spectrally), which gives us massive speed advantages on regular grids.

The Reference Medium Trick

Idea: Decompose $\boldsymbol{C}(\mathbf{x})$ into a *known* homogeneous reference $\boldsymbol{C}^0$ plus a perturbation:

$$\boldsymbol{C}(\mathbf{x}) = \boldsymbol{C}^0 + \Delta\boldsymbol{C}(\mathbf{x})$$

The stress can then be split:

$$\boldsymbol{\sigma} = \underbrace{\boldsymbol{C}^0 : \boldsymbol{\varepsilon}}_{\text{reference}} + \underbrace{\Delta\boldsymbol{C} : \boldsymbol{\varepsilon}}_{\text{polarization } \boldsymbol{\tau}}$$

How to choose $\boldsymbol{C}^0$?

Mode	Formula	When to use
`mean` (Voigt)	$\boldsymbol{C}^0 = \langle \boldsymbol{C}(\mathbf{x}) \rangle$	Default, safe
`contrast_aware`	Geometric mean of per-voxel K, μ	High-contrast composites

Lippmann–Schwinger Equation

Substituting the decomposition into equilibrium and using the **Green's function** of the reference medium gives the **Lippmann–Schwinger integral equation** [2, 3]:

$$\boxed{\varepsilon(\mathbf{x}) = \bar{\varepsilon} - \left(\boldsymbol{\Gamma}^0 * \boldsymbol{\tau}\right)(\mathbf{x})}$$

where:

- $\boldsymbol{\Gamma}^0$ is the **periodic Green's operator** of $\boldsymbol{C}^0$.
- $\boldsymbol{\tau} = \Delta\boldsymbol{C} : \varepsilon$ is the **polarization stress**.
- $*$ denotes spatial convolution $\rightarrow$ **pointwise multiplication in Fourier space**.

This is the key equation. Everything that follows is about solving it efficiently.

The Green's Operator in Fourier Space

For an **isotropic** reference (λ_0, μ_0), at wave direction $\mathbf{n} = \xi/|\xi|$:

Acoustic tensor:

$$A_{ik} = \mu_0\, \delta_{ik} + (\lambda_0 + \mu_0)\, n_i n_k$$

Analytic inverse:

$$A^{-1}_{ik} = \frac{\delta_{ik}}{\mu_0} - \frac{\lambda_0 + \mu_0}{\mu_0(\lambda_0 + 2\mu_0)}\, n_i n_k$$

Green's operator (Fourier space):

$$b_k = n_j\, \hat{\tau}_{kj}$$

$$\hat{u}_i = A^{-1}_{ik}\, b_k$$

$$\hat{\varepsilon}'_{ij} = \tfrac{1}{2}(n_i \hat{u}_j + n_j \hat{u}_i)$$

Applied pointwise at each frequency ξ.

Implementation: Pre-compute A^{-1} and $\mathbf{n}$ at all N^3 frequencies $\rightarrow$ stored as $(N, N, N, 3, 3)$ arrays. No Voigt notation needed.

Discrete Derivative Schemes

The wave vector $\hat{k}_i(\xi)$ encodes the spatial derivative $\partial/\partial x_i$. Different discretizations are possible:

Scheme	Wave vector	Reference	Notes
`continuous`	$\hat{k}_i = \xi_i$	Classical	Exact for smooth fields
`finite_difference`	$\hat{k}_i = \frac{N}{\pi} \sin(\frac{\pi \xi_i}{N})$	Willot 2015 [4]	FD-consistent
`rotated`	$\hat{k}_i = N \sin(\frac{2\pi \xi_i}{N})$	Willot & Pellegrini	Best for composites

Why this matters: The continuous scheme can produce Gibbs-like ringing at sharp grain boundaries. The rotated scheme reduces oscillations and improves convergence for high-contrast materials.

In our code: `derivative_scheme='continuous'` is the default; switch to `'rotated'` for composites with stiffness contrast $> 5\times$.

Stress-Controlled Loading

FFT solvers are natively **strain-driven**: input $\bar{\varepsilon}$, output $\langle \boldsymbol{\sigma} \rangle$. To prescribe **target stress** $\bar{\boldsymbol{\sigma}}$, we wrap the solver in a **Newton–Raphson iteration** on the macroscopic strain:

1. Initial guess: $\bar{\varepsilon}^{(0)} = (\boldsymbol{C}^0)^{-1} : \bar{\boldsymbol{\sigma}}$
2. Solve FFT with $\bar{\varepsilon}^{(k)} \;\rightarrow\; \langle \boldsymbol{\sigma} \rangle^{(k)}$
3. Residual: $\Delta\bar{\boldsymbol{\sigma}} = \bar{\boldsymbol{\sigma}} - \langle \boldsymbol{\sigma} \rangle^{(k)}$
4. **Broyden secant update:**

$$\boldsymbol{S}_{\text{eff}}^{(k+1)} = \boldsymbol{S}_{\text{eff}}^{(k)} + \frac{(\Delta\bar{\varepsilon} - \boldsymbol{S}_{\text{eff}}\Delta\bar{\boldsymbol{\sigma}}) \otimes \Delta\bar{\boldsymbol{\sigma}}}{\Delta\bar{\boldsymbol{\sigma}} \cdot \Delta\bar{\boldsymbol{\sigma}}}$$

5. Update: $\bar{\varepsilon}^{(k+1)} = \bar{\varepsilon}^{(k)} + \boldsymbol{S}_{\text{eff}}\,\Delta\bar{\boldsymbol{\sigma}}$

Result: 4–5 Newton iterations to converge $\|\Delta\bar{\boldsymbol{\sigma}}\|/\|\bar{\boldsymbol{\sigma}}\| < 10^{-3}$. The Broyden update learns the effective compliance as it iterates — much better than a fixed Voigt guess.

Basic Scheme (Moulinec–Suquet 1994)

Fixed-point iteration on the Lippmann–Schwinger equation:

1. Initialize $\varepsilon^{(0)} = \bar{\varepsilon}$
2. **Repeat** until $\|\Delta\varepsilon\|/\|\varepsilon\| < \text{tol}$:
   a. $\boldsymbol{\sigma}^{(n)} = \boldsymbol{C}(\mathbf{x}) : \varepsilon^{(n)}$ (constitutive law)
   b. $\boldsymbol{\tau}^{(n)} = \boldsymbol{\sigma}^{(n)} - \boldsymbol{C}^0 : \varepsilon^{(n)}$ (polarization)
   c. $\hat{\boldsymbol{\tau}}^{(n)} = \mathrm{FFT}[\boldsymbol{\tau}^{(n)}]$ (9 forward FFTs)
   d. $\hat{\varepsilon}' = \boldsymbol{\Gamma}^0(\xi) : \hat{\boldsymbol{\tau}}^{(n)}$ (pointwise, DC $= 0$)
   e. $\varepsilon' = \mathrm{IFFT}[\hat{\varepsilon}']$ (9 inverse FFTs)
   f. $\varepsilon^{(n+1)} = \bar{\varepsilon} - \varepsilon'$

Cost per iteration: $\mathcal{O}(N^3 \log N)$ via FFT vs. $\mathcal{O}(N^6)$ for direct FE assembly.

Problem: Converges slowly for high-contrast materials (many iterations needed).

CG Acceleration (Zeman 2010)

The basic scheme can be slow. CG [5] reformulates as a linear system for $\tilde{\varepsilon}$ (strain fluctuation):

$$\underbrace{\left(\boldsymbol{I} + \boldsymbol{\Gamma}^0 \circ \Delta \boldsymbol{C}\right)}_{\boldsymbol{A}} \tilde{\varepsilon} = -\boldsymbol{\Gamma}^0 \circ (\Delta \boldsymbol{C} : \bar{\varepsilon}) \equiv \mathbf{b}$$

Conjugate Gradient solves $\boldsymbol{A}\tilde{\varepsilon} = \mathbf{b}$ without forming $\boldsymbol{A}$ — only matrix-vector products (2 FFTs each).

Advantages over basic scheme:

- Guaranteed convergence for SPD operators
- Superlinear convergence rate
- Uses $\sim$9 iterations vs. $\sim$14 basic

Benchmark (Cu, 16^3, 8 grains):

Solver	Iters	Time
Basic	14	0.038s
CG	9	0.023s
Basic+AA(5)	10	0.046s

Anderson Acceleration

For the basic scheme, we optionally apply **Anderson mixing** [6] to accelerate convergence: Given the last m iterates $\{x_{k-m+1}, \ldots, x_k\}$ and their fixed-point evaluations $\{G(x_{k-m+1}), \ldots, G(x_k)\}$:

$$x_{k+1} = G(x_k) - (\Delta \mathbf{G})\, \theta^*, \qquad \theta^* = \arg\min_{\theta} \|F_k - \Delta \mathbf{F}\, \theta\|^2$$

where $F_k = G(x_k) - x_k$ is the fixed-point residual, $\Delta \mathbf{F}$ collects the last m residual differences, and $\Delta \mathbf{G} = \Delta \mathbf{X} + \Delta \mathbf{F}$ (differences of the fixed-point map outputs).

- Window $m = 3$–5 is typical.
- Tikhonov regularization for numerical robustness.
- Periodic restart every 50 iterations.
- **Effect:** $\sim$30% fewer iterations (14 $\rightarrow$ 10 for polycrystalline Cu).

GPU Acceleration: Batched FFT

The FFT solver spends $> 60\%$ of runtime in forward/inverse FFTs. We use **CuPy** to run all FFTs on the GPU:

Batched strategy:

1. Reshape tensor $(N, N, N, 3, 3) \rightarrow (9, N, N, N)$.
2. Run 9 independent 3D FFTs as a batch.
3. Reshape back to $(N, N, N, 3, 3)$.

CPU alternative: `use_rfft=True` uses real-input FFT (`rfftn`/`irfftn`), which halves memory on the last axis.

```
def fft_3x3(field, use_rfft=False):
    if HAS_GPU:
        N = field.shape[0]
        f9 = field.reshape(N,N,N,9)
        f9 = f9.transpose(3,0,1,2)
        out = cp.empty_like(f9,
                  dtype=cp.
                      complex128)
        for c in range(9):
            out[c] = cpfft.fftn(f9[c
                ])
        return out.transpose(
            1,2,3,0).reshape(
            N,N,N,3,3)
```

All constitutive operations (stiffness contraction, strain updates) also run on GPU via CuPy's NumPy-compatible API.

EVPFFT: Coupling Plasticity with FFT

The **elasto-viscoplastic FFT** solver [7, 8] couples the plastic constitutive update with the FFT equilibrium solver:

1. Divide total loading into N_{inc} increments.
2. At each increment:
   - **Outer Newton loop:** stress control ($\bar{\varepsilon}$ updated until $\langle \boldsymbol{\sigma} \rangle \approx \bar{\boldsymbol{\sigma}}$).
   - **Inner FFT loop:** fixed-point equilibrium.
   - At each FFT iteration, every voxel evaluates:
     - $\tau^{\alpha} = \boldsymbol{\sigma} : \boldsymbol{P}^{\alpha} \;\rightarrow\;$ OTIS flow rule $\;\rightarrow\; \Delta\gamma^{\alpha}$
     - $\Delta\varepsilon^{p} = \sum_{\alpha} \Delta\gamma^{\alpha} \boldsymbol{P}^{\alpha}$
     - $\boldsymbol{\sigma} = \boldsymbol{C} : (\varepsilon - \varepsilon^{p}_{\text{frozen}} - \Delta\varepsilon^{p})$
   - After convergence: commit ε^{p}, update hardening.

Output: full-field stress, strain, plastic strain, slip resistance, and texture evolution at every step.

EVPFFT: Convergence Acceleration

The EVPFFT loop has two levels that both need fast convergence:

Outer (Newton) loop:

- **Broyden secant update**: learns effective compliance from previous Newton steps.
- **Adaptive inner cap**: limit FFT iters to 5 early on, increase to 15 near convergence.
- Result: $\sim$3–5 Newton iters per load step.

Inner (FFT) loop:

- **Full Newton step**: no damping ($\alpha = 1$).
- **GPU acceleration**: constitutive + FFT on GPU.
- 5–15 FFT iters per Newton step.
- Total: $\sim$35 constitutive evaluations per load step (vs. 360 before optimisation).

10$\times$ speedup from Broyden + adaptive cap + full step at 5 GPa loading with OTIS at $T = 1123$ K.

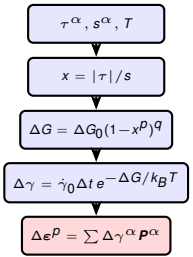
OTIS: The Constitutive Engine

The OTIS flow rule computes $\Delta\gamma^\alpha$ from $(\tau^\alpha, s^\alpha, T, \Delta t)$:

$$\Delta\gamma^\alpha = \dot{\gamma}_{\text{ref}}\ \Delta t\ \exp\left[-\frac{\Delta G_0}{k_B T}\left(1 - \left(|\tau|/s\right)^p\right)^q\right]\ \text{sgn}(\tau^\alpha)$$

Vectorized implementation:

- Input: $(N_{\text{vox}}, 12)$ arrays of τ and s.
- Saturated regime: $|\tau| \geq s \to \Delta\gamma = \dot{\gamma}_{\text{ref}}\,\Delta t$.
- Normal regime: thermally activated exponential.
- Clamped to $|\Delta\gamma| \leq 0.005$ for numerical stability.
- Exponent clipped to ≥ -50 to avoid underflow.



The Analytical OTIS Tangent

For Newton convergence in the constitutive update, we need $\partial \Delta\gamma / \partial \tau$:

$$\frac{\partial \Delta\gamma^{\alpha}}{\partial \tau^{\alpha}} = \dot{\gamma}_{\text{ref}}\, \Delta t \cdot \frac{\Delta G_0}{k_B T} \cdot p \cdot q \cdot \frac{(|\tau|/s)^{p-1}}{s} \cdot \Big(1 - (|\tau|/s)^{p}\Big)^{q-1} \cdot \exp\Big[-\frac{\Delta G_0}{k_B T}\big(1 - (|\tau|/s)^{p}\big)^{q}\Big]$$

Why analytical?

- Exact — no truncation error from finite differences.
- Fast — single-pass pointwise evaluation, fully vectorised.
- Enables consistent tangent stiffness $\boldsymbol{C}^{ep}$ per voxel.

The tangent is the product of four terms from the chain rule:

$$\underbrace{\dot{\gamma}_{\text{ref}}\, \Delta t}_{\text{prefactor}} \cdot \underbrace{\frac{\Delta G_0}{k_B T}}_{\text{thermal}} \cdot \underbrace{pq \frac{(|\tau|/s)^{p-1}}{s}(1-(|\tau|/s)^{p})^{q-1}}_{\text{barrier derivatives}} \cdot \underbrace{\exp[\cdots]}_{\text{exponential}}$$

Physics Invariant Checks (Debug Mode)

We implemented automated **physics sanity checks** that can be enabled with `debug_checks=True`:

Check	What it verifies	Tolerance
`strain_symmetry`	$\max \lvert \boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T \rvert \approx 0$	10^{-10}
`stress_symmetry`	$\max \lvert \boldsymbol{\sigma} - \boldsymbol{\sigma}^T \rvert \approx 0$	10^{-10}
`macro_strain_err`	$\lvert \langle \boldsymbol{\varepsilon} \rangle - \bar{\boldsymbol{\varepsilon}} \rvert / \lvert \bar{\boldsymbol{\varepsilon}} \rvert$	10^{-6}
`macro_stress_err`	$\lvert \langle \boldsymbol{\sigma} \rangle - \bar{\boldsymbol{\sigma}} \rvert / \lvert \bar{\boldsymbol{\sigma}} \rvert$	10^{-4}
`min_energy`	$\min\left(\frac{1}{2}\boldsymbol{\sigma} : \boldsymbol{\varepsilon}\right) \geq 0$	positive

Purpose: Catch bugs in indexing, Voigt conventions, or constitutive updates *before* they corrupt the entire simulation.

Per-Iteration Profiler

The `SolverProfiler` class instruments every phase of the solver with sub-millisecond timing:

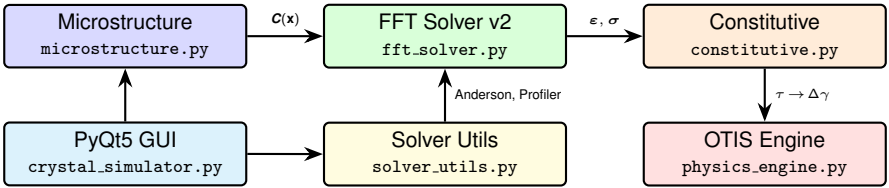
```
from solver_utils import SolverProfiler

prof = SolverProfiler()

with prof.phase('fft'):
    tau_hat = fft_3x3(tau)

with prof.phase('green_op'):
    gamma_hat = apply_green(tau_hat,
                            n_field, Ainv)

with prof.phase('ifft'):
    gamma = ifft_3x3(gamma_hat)

print(prof.summary("Basic scheme"))
```

Example output:

```
Basic scheme (0.038s total)
  fft            0.014s (36.8%)
  constitutive   0.010s (26.3%)
  ifft           0.008s (21.1%)
  green_op       0.004s (10.5%)
  anderson       0.002s  (5.3%)
```

Use case: Identify bottlenecks, validate GPU speedup, compare solver configurations.

Software Architecture



- Pure **Python 3.11** — NumPy + CuPy + PyQt5 + PyVista.
- **GPU:** CuPy-CUDA12x on RTX 5070.
- Full (3×3) tensor arithmetic — no Voigt-notation pitfalls.
- Model-tree GUI with Abaqus-style interaction.

GUI: Interactive Crystal Plasticity Simulator

The `crystal_simulator.py` provides a full interactive interface:

Model Tree (left panel):

- **Material:** (C_{11}, C_{12}, C_{44}) + presets
- **Plasticity:** OTIS params (T, $\dot{\gamma}_{\text{ref}}$, τ_0, s_s, h_0, q_{lat})
- **Microstructure:** N, grains, seed
- **BCs:** Full stress tensor + presets
- **Solver:** Method, tolerances, `derivative_scheme`, `ref_medium_mode`, Anderson m, debug checks
- **Job:** Submit, progress bar

3D Viewer (right panel):

- PyVista 3D rendering
- Field selector: σ_{VM}, σ_{ij}, ε_{ij}, $|\mathbf{u}|$, $\varepsilon^p_{\text{eq}}$, accumulated slip, slip resistance
- Grain boundary overlay
- Loading arrows (red/blue)
- Step-by-step animation
- Dark Fusion theme

Test 1: Homogeneous Medium — Exact in 1 Iteration

Setup: Uniform stiffness $\boldsymbol{C} = \text{const}$ on 8^3 grid. Applied strain: $\varepsilon_{11} = 1\%$.

Expected: Solver should converge in exactly 1 iteration with zero error (no heterogeneity $\rightarrow$ zero polarization).

Derivative scheme	Iterations	$\lvert\langle\sigma\rangle - \sigma_{\text{exact}}\rvert / \lvert\sigma_{\text{exact}}\rvert$	Status
`continuous`	1	0.00	✓
`finite_difference`	1	0.00	✓
`rotated`	1	0.00	✓

Interpretation: All three derivative schemes produce the exact analytical solution for a homogeneous medium. This verifies the Green's operator, FFT, and stress computation.

Test 2: Two-Phase Laminate — Bounds Check

Setup: Layered composite: stiff (Cu) and compliant (0.5× Cu), equal fractions, 16^3 grid.

Expected: Effective stiffness C_{11}^{eff} must lie between Reuss (lower) and Voigt (upper) bounds:

$$C_{\text{Reuss}} \leq C_{\text{eff}} \leq C_{\text{Voigt}}$$

Scheme	C_{eff} (GPa)	Reuss (GPa)	Voigt (GPa)	**Status**
`continuous`	112.27	112.27	126.30	✓
`rotated`	112.27	112.27	126.30	✓

C_{eff} sits exactly at the Reuss bound — consistent with iso-stress conditions in a laminate loaded parallel to the layers.

Test 3: Derivative Scheme Comparison

Setup: Polycrystalline Cu, 8 grains, 16^3 grid, $\varepsilon_{11} = 1\%$, tol$= 10^{-5}$.

Scheme	Solver	Iters	Time (s)	$\mathrm{VM}_{\max}$ (GPa)	Error
continuous	Basic	14	0.037	1.517	5.9×10^{-6}
continuous	CG	9	0.022	1.517	6.5×10^{-6}
finite_difference	Basic	14	0.038	1.521	5.8×10^{-6}
finite_difference	CG	9	0.024	1.521	6.5×10^{-6}
rotated	Basic	14	0.037	1.492	6.0×10^{-6}
rotated	CG	9	0.023	1.492	6.8×10^{-6}

Observations:

- CG consistently faster (9 vs. 14 iters).
- Rotated scheme: slightly lower VM max (smoother near grain boundaries).
- Same iteration count across schemes — contrast is moderate ($A = 3.21$), so all schemes work well.

Test 4: Anderson Acceleration

Setup: Same polycrystal, basic scheme only. Anderson mixing window $m = 0, 3, 5, 8$.

Anderson m	**Iterations**	**Time (s)**	**Final error**
0 (off)	14	0.038	5.87×10^{-6}
3	10	0.037	5.78×10^{-6}
5	10	0.046	3.23×10^{-6}
8	10	0.053	2.70×10^{-6}

Result:

- Anderson reduces iterations by $\sim 30\%$ ($14 \to 10$).
- Final error also improves (better accuracy per iteration).
- Wall time overhead from the mixing step is small.
- Most useful for high-contrast problems where basic scheme converges slowly.

Test 5: Stress-Controlled Solver with Broyden

Setup: Polycrystalline Cu, 16^3, target $\sigma_{11} = 1$ GPa uniaxial.

Newton iter	$\lvert\Delta\bar{\boldsymbol{\sigma}}\rvert/\lvert\bar{\boldsymbol{\sigma}}\rvert$	$\langle\sigma_{11}\rangle$ (MPa)	$\langle\sigma_{22}\rangle$	$\langle\sigma_{33}\rangle$
0	1.3×10^{-1}	923.0	13.3	63.7
1	3.1×10^{-2}	978.4	19.8	1.9
2	5.0×10^{-3}	999.3	0.9	-0.2
3	1.2×10^{-3}	999.8	0.8	-0.6
4	2.1×10^{-4}	1000.0	0.1	0.0

Converged in 5 Newton iterations, total 45 FFT (CG) iterations, 0.124 seconds.

The Broyden secant update quickly learns the effective compliance, achieving quadratic-like convergence. Physics checks: all symmetry, macro strain, energy positivity passed.

Test 6: EVPFFT at High Temperature

Setup: Polycrystalline Cu, 8^3, 4 grains, $\sigma_{11} = 5$ GPa, $T = 1123$ K, OTIS flow rule.

Solver configuration:

- 5 load steps ($0 \rightarrow 5$ GPa)
- Broyden + adaptive inner cap
- $\dot{\gamma}_{\text{ref}} = 10^7$ s^{-1}
- Voce: $\tau_0 = 50$, $s_s = 200$, $h_0 = 500$ MPa

Results:

- Total time: $\sim$0.96 s
- Max plastic strain: $\sim$10.5%
- $\sim$35 constitutive evaluations per step

What the OTIS flow rule captures:

- Thermal activation: at 1123 K, dislocations overcome barriers more easily.
- Rate sensitivity: $\dot{\gamma}$ varies exponentially with τ/s.
- Heterogeneous plastic flow: grains oriented for easy slip deform more.

Output Fields: What the Simulator Produces

At every load step, the simulator provides **full 3D fields** over the N^3 grid:

Elastic fields:

- Von Mises stress σ_{VM}
- Stress components $\sigma_{11}, \sigma_{22}, \ldots$
- Strain components $\varepsilon_{11}, \varepsilon_{22}, \ldots$
- Displacement magnitude $|\mathbf{u}|$
- Hydrostatic pressure

Plastic fields (EVPFFT):

- Equivalent plastic strain $\varepsilon^p_{\text{eq}}$
- Accumulated slip $\sum |\gamma^\alpha|$
- Slip resistance s^α (MPa)
- Per-system slip rates $\dot{\gamma}^\alpha$

All fields are viewable in the 3D GUI with step-by-step animation and grain boundary overlay.

Finite-Strain Kinematics: $\mathbf{F} = \mathbf{F}^e\mathbf{F}^p$

Multiplicative decomposition replaces infinitesimal strain:

$$\mathbf{F} = \mathbf{F}^e\,\mathbf{F}^p, \qquad \mathbf{E}^e = \tfrac{1}{2}\big(\mathbf{F}^{e\top}\mathbf{F}^e - \mathbf{I}\big)$$

Stress measures:

- 2nd Piola–Kirchhoff in intermediate config: $\mathbf{S} = \boldsymbol{C} : \mathbf{E}^e$
- Mandel stress (for resolved shear): $\mathbf{M} = \mathbf{C}^e\mathbf{S}$ (non-symmetric)
- 1st PK for equilibrium: $\mathbf{P} = \mathbf{F}^e\mathbf{S}\mathbf{F}^{p-\top}$

Resolved shear stress:

$$\tau^\alpha = \mathbf{M} : (\hat{\mathbf{s}}^\alpha \otimes \hat{\mathbf{n}}^\alpha)$$

Slip systems remain *in the crystal frame* — $\mathbf{F}^e$ carries the lattice rotation implicitly.

$\mathbf{F}^p$ update: $\mathbf{F}^p_{n+1} = (\mathbf{I} + \mathbf{L}^p\Delta t)\,\mathbf{F}^p_n$ with $\mathbf{L}^p = \sum_\alpha \dot\gamma^\alpha \hat{\mathbf{s}}^\alpha \otimes \hat{\mathbf{n}}^\alpha$

Finite-Strain FFT: Unsymmetrised Green Operator

In finite strain, the displacement gradient $\mathbf{H} = \mathbf{F} - \mathbf{I} = \nabla \mathbf{u}$ is **non-symmetric** (includes rotation).

Lippmann–Schwinger: $\mathbf{H}(\mathbf{x}) = \bar{\mathbf{H}} - \big(\Gamma^0 * \boldsymbol{\tau}\big)(\mathbf{x}), \quad \boldsymbol{\tau} = \mathbf{P} - \boldsymbol{C}^0 : \mathrm{sym}(\mathbf{H})$

	Small strain	**Finite strain**
Unknown field	ε_{ij} (symmetric)	$H_{ij} = F_{ij} - \delta_{ij}$ (non-sym)
Stress	σ_{ij} (Cauchy)	P_{ij} (1st PK)
Green output	$\hat{\varepsilon}'_{ij} = \frac{1}{2}(n_i \hat{u}_j + n_j \hat{u}_i)$	$\hat{H}'_{ij} = \hat{u}_i \, n_j$
Acoustic tensor	$A_{ik} = C^0_{ijkl} n_j n_l$	identical

Only the *output mapping* changes; the acoustic tensor and its inverse are *the same* as small strain.

Kocks–Mecking Dislocation-Density Hardening

Replaces phenomenological Voce with physically-based dislocation density evolution (Kocks & Mecking, 2003):

$$\dot{\rho}^{\alpha} = \left(k_1 \sqrt{\sum \rho} - k_2 \rho^{\alpha}\right) |\dot{\gamma}^{\alpha}|, \qquad s^{\alpha} = s_0 + \alpha_T \, \mu \, b \sqrt{\sum_{\beta} q_{\alpha\beta} \rho^{\beta}}$$

Cu parameters:

- $k_1 = 7 \times 10^8$ m^{-1} (storage)
- $k_2 = 10$ (dynamic recovery)
- $\alpha_T = 0.3$ (Taylor coefficient)
- $b = 2.56 \times 10^{-10}$ m (Burgers vector)
- $\rho_0 = 10^{12}$ m^{-2} (initial SSD density)

Advantages over Voce:

- Tracks physical dislocation density
- Couples naturally with GND
- Temperature dependence via k_2
- Predicts stage II→III transition

Armstrong–Frederick Back-Stress (Bauschinger Effect)

Kinematic hardening introduces a *back-stress* χ^α on each slip system, capturing the Bauschinger effect:

$$\dot{\chi}^\alpha = c_1\,\dot{\gamma}^\alpha - c_2\,\chi^\alpha\,|\dot{\gamma}^\alpha|, \qquad \chi_{\text{sat}} = \frac{c_1}{c_2}$$

- Effective resolved shear stress: $\tau^\alpha_{\text{eff}} = \tau^\alpha - \chi^\alpha$ (fed into OTIS flow rule)
- Cu defaults: $c_1 = 10^9$ Pa, $c_2 = 10 \Rightarrow\ \chi_{\text{sat}} = 100$ MPa
- Armstrong & Frederick (1966): the simplest nonlinear kinematic hardening model that captures load reversal effects
- Essential for cyclic loading / fatigue simulations

Implicit Newton Constitutive Update

Per-voxel Newton–Raphson replaces the explicit constitutive update—all 12 slip systems are coupled simultaneously:

$$R^{\alpha} = \Delta\gamma^{\alpha} - \Delta\gamma^{\alpha}_{\mathrm{OTIS}}\big(\tau^{\alpha}_{\mathrm{eff}}(\Delta\boldsymbol{\gamma}),\, s,\, T,\, \Delta t\big) = 0$$

12×12 Jacobian per voxel:

$$J_{\alpha\beta} = \delta_{\alpha\beta} + \frac{\partial \Delta\gamma^{\alpha}}{\partial \tau^{\alpha}}\, H_{\alpha\beta}, \qquad H_{\alpha\beta} = (\hat{\mathbf{s}}^{\alpha} \otimes \hat{\mathbf{n}}^{\alpha}) : \boldsymbol{C} : (\hat{\mathbf{s}}^{\beta} \otimes \hat{\mathbf{n}}^{\beta})$$

- $H_{\alpha\beta}$: elastic interaction matrix (constant per grain)
- $\partial\Delta\gamma^{\alpha}/\partial\tau^{\alpha}$: analytical OTIS tangent (slide 7)
- Converges in 3–5 Newton iterations (quadratic rate)
- Vectorised via `np.linalg.solve` on $(n_{\mathrm{vox}}, 12, 12)$

GND Hardening from Nye Tensor

Non-local hardening from *geometrically necessary dislocations* (Nye, 1953; Arsenlis & Parks, 1999):

$$\boldsymbol{\alpha} = \operatorname{curl}(\mathbf{F}^p) \qquad \Longleftrightarrow \qquad \hat{\alpha}_{ij} = \varepsilon_{jkl}\,(i\xi_k)\,\hat{F}^p_{il} \quad \text{(spectral)}$$

Projection onto slip systems:

$$\rho^{\alpha}_{\text{edge}} = \tfrac{1}{b}\,\boldsymbol{\alpha} : (\hat{\mathbf{s}}^{\alpha} \otimes \hat{\mathbf{t}}^{\alpha}), \qquad \rho^{\alpha}_{\text{screw}} = \tfrac{1}{b}\,\boldsymbol{\alpha} : (\hat{\mathbf{s}}^{\alpha} \otimes \hat{\mathbf{s}}^{\alpha}), \qquad \rho^{\alpha}_{\text{GND}} = \sqrt{\rho^{2}_{\text{edge}} + \rho^{2}_{\text{screw}}}$$

- GND resistance combines with SSD: $s = s_0 + \alpha_T \mu b \sqrt{\rho_{\text{SSD}} + \rho_{\text{GND}}}$
- Computed via FFT-based spectral curl (no finite-difference stencil)
- $\hat{\mathbf{t}}^{\alpha} = \hat{\mathbf{s}}^{\alpha} \times \hat{\mathbf{n}}^{\alpha}$ is the edge dislocation line direction

Texture Evolution

Crystal orientations **evolve during deformation** via the elastic rotation tensor:

$$\mathbf{F}^e = \mathbf{R}^e\mathbf{U}^e \quad \text{(polar decomposition)}, \qquad \mathbf{R}_{\text{current}} = \mathbf{R}^e\,\mathbf{R}_{\text{initial}}$$

- Per-grain averaged $\mathbf{F}^e$ gives the lattice rotation
- Euler angles updated via $\mathbf{R} \to (\varphi_1, \Phi, \varphi_2)$ (Bunge ZXZ)
- Stiffness and slip systems rotate implicitly through $\mathbf{F}^e$ in the Mandel stress formulation
- Misorientation field visualised in the GUI (degrees from initial orientation)

Validation: At 1% uniaxial strain (200 MPa), maximum misorientation $\approx 0.1^\circ$ — consistent with the elastic regime (large rotations expected only at $> 10\%$ strain).

Summary: What We Built

A **complete, GPU-accelerated crystal plasticity FFT simulator** from scratch in Python:

Physics:

- FCC crystal plasticity (12 slip systems)
- OTIS thermally-activated flow rule
- Voce *and* Kocks–Mecking hardening
- Armstrong–Frederick back-stress
- Single-crystal elastic anisotropy
- **Finite-strain: $\mathbf{F} = \mathbf{F}^e\mathbf{F}^p$**
- **GND hardening (Nye tensor)**
- **Texture evolution tracking**

Mathematics:

- Lippmann–Schwinger equation
- Green's operator (sym + unsym)
- 3 discrete derivative schemes
- Broyden stress control

Numerics:

- Basic + CG solvers
- Anderson acceleration
- Batched GPU FFT (CuPy)
- Implicit Newton (12×12 per voxel)
- Per-phase profiling
- Physics invariant checks

Software:

- PyQt5 model-tree GUI
- PyVista 3D visualisation
- Step-by-step field animation
- 21 display fields (stress, strain, SSD, GND, back-stress, misorientation)
- Pure Python (no Fortran)

Future Work

Near-term:

- Exponential map for $\mathbf{F}^p$ update (preserve $\det \mathbf{F}^p = 1$)
- 64^3–128^3 grids with `pyFFTW`
- HCP crystal structure (basal, prismatic, pyramidal slip)
- Pole figure output and ODF texture analysis

Long-term:

- Phase-field coupling: damage and recrystallisation
- Multi-GPU scaling for large polycrystals (256^3+)
- BCC / twin deformation modes
- CPFEM–FFT hybrid: macro FE with FFT RVE at integration points

References

[1] Carlos N. Tomé, Gilles R. Canova, U. Fred Kocks, Nicholas Christodoulou, and John J. Jonas. The relation between macroscopic and microscopic strain hardening in F.C.C. polycrystals. *Acta Metall.*, 32:1637–1653, 1984.

[2] Hervé Moulinec and Pierre Suquet. A fast numerical method for computing the linear and nonlinear mechanical properties of composites. *C. R. Acad. Sci. Paris, Série II*, 318: 1417–1423, 1994.

[3] Hervé Moulinec and Pierre Suquet. A numerical method for computing the overall response of nonlinear composites with complex microstructure. *Comput. Methods Appl. Mech. Engrg.*, 157:69–94, 1998.

[4] François Willot. Fourier-based schemes for computing the mechanical response of composites with accurate local fields. *C. R. Mécanique*, 343:232–245, 2015.

[5] Jan Zeman, Jaroslav Vondřejc, Jan Novák, and Ivo Marek. Accelerating a FFT-based solver for numerical homogenization of periodic media by conjugate gradients. *Int. J. Numer. Meth. Engng.*, 82:1296–1313, 2010.

References

[6] S. Lucarini and J. Segurado. FFT based approaches in micromechanics: fundamentals, methods and applications. *Modelling Simul. Mater. Sci. Eng.*, 30:023002, 2022.

[7] Ricardo A. Lebensohn. N-site modeling of a 3D viscoplastic polycrystal using Fast Fourier Transform. *Acta Mater.*, 49:2723–2737, 2001.

[8] Ricardo A. Lebensohn, Anand K. Kanjarla, and Philip Eisenlohr. An elasto-viscoplastic formulation based on fast Fourier transforms for the prediction of micromechanical fields in polycrystalline materials. *Int. J. Plasticity*, 32–33:59–69, 2012.

[9] Ricardo A. Lebensohn and Alan Needleman. Numerical implementation of non-local polycrystal plasticity using Fast Fourier Transforms. *J. Mech. Phys. Solids*, 97:333–351, 2016.

Appendix: Stiffness Rotation Formula

The full 4th-order rotation of the stiffness tensor:

$$C_{ijkl}^{\text{sample}} = R_{ip} R_{jq} R_{kr} R_{ls}\, C_{pqrs}^{\text{crystal}}$$

For cubic symmetry, C^{crystal} has only 3 independent constants: (C_{11}, C_{12}, C_{44}). After rotation by Euler angles $(\varphi_1, \Phi, \varphi_2)$:

- The rotation matrix $\mathbf{R}$ is 3×3 (9 elements).
- The contraction involves 81 multiplications per Voigt entry.
- We use the Voigt 6×6 form: $C'_{IJ} = M_{Ii} M_{Jj}\, C_{ij}$ where M is the 6×6 Bond matrix.

Appendix: Voigt Notation Conventions

Voigt ordering: $11 \to 1$, $22 \to 2$, $33 \to 3$, $23 \to 4$, $13 \to 5$, $12 \to 6$.

Strain: $\varepsilon_{\text{Voigt}} = (\varepsilon_{11}, \varepsilon_{22}, \varepsilon_{33}, 2\varepsilon_{23}, 2\varepsilon_{13}, 2\varepsilon_{12})$

Stress: $\sigma_{\text{Voigt}} = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{23}, \sigma_{13}, \sigma_{12})$

Engineering shear factors: $\varepsilon_4 = 2\varepsilon_{23}$ (factor of 2 for shear strains). This ensures $\sigma = C\varepsilon$ works directly with the 6×6 stiffness matrix.

In our code: Internal FFT operations use full (3×3) tensors. Voigt is used only for storage, I/O, and the constitutive contraction. Conversion functions: `strain_v2t`, `strain_t2v`, `stress_v2t`, `stress_t2v`.