An Overview of
Fiber Orientation Tools

`http://github.com/charlestucker3/Fiber-Orientation-Tools`

Charles L. Tucker III
Department of Mechanical Science and Engineering
University of Illinois at Urbana-Champaign
1206 W. Green St.
Urbana, IL 61801

February 22, 2022

This document summarizes **Fiber Orientation Tools**, a set of MATLAB functions for modeling flow-induced fiber orientation in discontinuous fiber composites, and for predicting the resulting mechanical properties.

The tools accompany the book *Fundamentals of Fiber Orientation: Description, Measurement and Prediction* by C. L. Tucker III (Hanser, Munich, 2022), and references to sections, figures and equations indicate items in the book.

MATLAB live scripts that demonstrate the use of various tools are described first, followed by a list of the functions in the toolkit, organized by category. To see the details of any function, type `help` followed by the function name in the MATLAB command window.

1 Live Scripts with Example Calculations

The live scripts are arranged here according to the chapters of the book.

Chapter 2. Describing Fiber Orientation and Length

OrientationDistributionFunctions.mlx creates 3-D orientation distribution functions $\psi(\mathbf{p})$ using the Jeffery distribution function, as in Fig. 2.5. Uses `A2F` and `drawPsi`.

OrientationTensorExamples.mlx calculates orientation tensors for various combinations of $\mathbf{p}$ vectors. Uses `p2A` and follows the examples in Sections 2.3.1 and 2.3.5.

EigenvaluesEigenvectors.mlx finds eigenvalues and eigenvectors of a second-order orientation tensor, and compares the standard MATLAB function `eig` with the `eigsort` function from this toolkit.

ReconstructDiscreteDistributionFcn.mlx shows how to find a set of orientation vectors $\mathbf{p}^i$ and weighting factors f_i to form a discrete approximation of an orientation distribution function, using the Jeffery distribution function, Eqn. (2.103). Uses `A2F` and `matchA`. See also **A2Faccuracy.mlx**, below.

A2Faccuracy.mlx explains how to control the accuracy of `A2F`, which finds the deformation gradient tensor $\mathbf{F}$ that will transform an initially isotropic orientation state to a given second-order tensor $\mathbf{A}$, using the deformation form of Jeffery's equation. `F2A` does the reverse calculation.

Chapter 3. Measuring Fiber Orientation and Length

PlanarSectionMeasurement.mlx uses `thetaphi2A` to compute orientation information for data from a planar section. Follows Example 3.1.1.

Chapter 4. Flow Orientation of Single Fibers

JefferyFiber.Motionmlx calculates the motion of a single fiber following Jeffery's equation using `pDot` and `ode45` (Section 4.2.3). Illustrates Jeffery orbits, as in Fig. 4.14.

JefferyDeformationForm.mlx illustrates the use of `changep`, which implements the deformation form of Jeffery's equation. Also uses `randomfibers` to generate a set of $\mathbf{p}$ vectors that are randomly distributed in all directions, and `p2A` to see the initial and final orientation tensors.

Chapter 5. Flow Orientation of Groups of Fibers

JefferyTensorEqn.mlx illustrates the numerical solution of the orientation tensor equation when every fiber follows Jeffery's equation. This is the example from Section 5.1.2 and it uses `AdotJeffQuad`.

SolvePlanarDistnFcn.mlx shows how to use `solvePsi2D` to solve for the planar orientation distribution function $\psi_\phi(\phi, t)$. See Sections 5.2.1 and 5.3.1. This script produces Fig. 5.5(a), and illustrates the control of 'wiggles' in the finite difference solution using power-law upwinding and/or grid refinement.

Solve3DdistnFcn.mlx shows how to use `solvePsi3D` to solve for the 3-D orientation distribution function $\psi(\mathbf{p}, t)$. See Sections 5.2.2 and 5.3.2. This script produces the tensor history in Fig. 5.7, and shows how to display distribution functions as in Fig. 5.6. This script also shows how to calculate $\psi(\mathbf{p}, t)$ for anisotropic rotary diffusion models, using `solvePsiARD` (Section 5.6).

OrientationTensorEqns.mlx uses `Adot2` together with `ode45` to solve the different tensor equations to model fiber orientation. The script shows solutions to the Folgar-Tucker equation (Sections 5.3 and 5.4), anisotropic rotary diffusion (ARD) models (Section 5.6), and slow kinetics models (Section 5.7).

FitCI.mlx illustrates the use of `fitCI` to find the interaction coefficient C_I from the Folgar-Tucker model that produces a given value of A_{11} in steady-state simple shear flow. Also shows that the choice of closure approximation affects the value of C_I.

FitARDparams.mlx uses `fitARD` to obtain the ARD parameters that give desired steady-state values of A_{11} and A_{33} in simple shear flow. Also demonstrates that the steady-state orientation for a given ARD model is independent of the kinetic parameter κ when the RSC or RPR models are also used.

Chapter 6. Suspension Rheology and Flow-Orientation Coupling

FiberSuspensionStress.mlx demonstrates the use of `tauFiber` to find the orientation-dependent stress in a fiber suspension. Shows how to do the calculations used to generate Fig. 6.4.

Chapter 7. Fiber Length Degradation during Processing

FiberLengthModel.mlx explains the dimensionless version of the Phelps-Tucker fiber length model as implemented in `solveFLDstar`, and shows how to set up the calculation for the example in Fig. 7.4.

Chapter 8. Mechanical Properties and Orientation

UnidirectionalProperties.mlx shows how to calculate the stiffness and thermal expansion of composites with unidirectional fiber orientation. The main functions for this are `mori` for the Mori-Tanaka model, `lielens` for the Lienlens/double-inclusion model, and `halpin` for the Halpin-Tsai model. The functions `iso2C`, `C2eng` and `inv4` are also used.

OrientationAveragedProperties.mlx applies orientation averaging to find the properties of a composite with a distribution of fiber orientation. Both stiffness and thermal expansion are considered. The main functions are `oravg4` and `oravg2`. The script reproduces Fig. 8.5(a) showing elastic modulus vs. orientation, and creates a plot of thermal expansion vs. orientation similar to Fig. 8.8(a).

LaminatedPlaneProperties.mlx uses classical lamination theory, as implemented in `Clayer2laminate`, to find the tensile and flexural moduli of an injection molded composite whose orientation varies across the thickness. Partial results from Table 8.4 are reproduced. A summary of the underlying theory is given in **LaminationTheory.pdf** in the documentation folder.

2 Functions Listed by Category

2.1 Operations on Orientation Tensors

eigsort returns the eigenvalues and eigenvectors of a tensor, sorted from largest eigenvalue to smallest.

inv4 finds the tensor inverse of a fourth-order tensor.

p2A converts a set of **p** vectors to second-order and fourth-order orientation tensors, with or without weighting factors.

rotate4 coordinate transformation for a fourth-order tensor. See Section A.4.5.

tens2vec converts a symmetric second-order tensor from 3×3 matrix form to 6×1 column vector (contracted) form. See Section 2.3.5.1.

tens2vec4 converts a symmetric fourth-order tensor from 6×6 matrix form to 15×1 column vector form. This is not discussed in the book, but is used here by `matchA`.

thetaphi2A returns the second-order orientation tensor corresponding to a set of angles (θ, ϕ) measured from planar section data (Section 3.1.1). Either the Bay or Konicek weighting functions can be used.

transisoA returns the full second-order orientation tensor $\mathbf{A}$ and fourth-order orientation tensor $\mathbb{A}$ for a transversely isotropic orientation state with given values of A_{11} and $\mathbb{A}_{1111}$.

vec2tens converts a symmetric second-order tensor from 6×1 column vector form to 3×3 matrix form. See Section 2.3.5.1.

vec2tens4 converts a symmetric fourth-order tensor from 15×1 column vector form to 6×6 matrix form. This is not discussed in the book but is the converse of `tens2vec4`.

2.2 Flow-Induced Orientation Models

Adot2 gives the time derivative $\dot{\mathbf{A}}$ as a function of $\mathbf{A}$, $\mathbf{L}$ and orientation model parameters, for a wide range of orientation models. This is the principal tool used to predict flow-induced orientation.

AdotJeffQuad gives the time derivative $\dot{\mathbf{A}}$ for the Jeffery model using the quadratic closure. Used in Section 5.1.2 and Fig. 5.1.

Asteady finds the steady-state orientation tensor $\mathbf{A}$ for any given $\mathbf{L}$ and any model in `Adot2`.

changep finds a set of current orientation vectors $\mathbf{p}$ using the deformation form of Jeffery's equation, for a given set of initial vectors $\mathbf{p}'$ and deformation gradient tensor $\mathbf{F}$.

closeA4 uses any of several closure approximations to find the fourth-order orientation tensor $\mathbb{A}$ corresponding to a second-order tensor $\mathbf{A}$. See Section 5.4.

fitARD finds the anisotropic rotary diffusion (ARD) model parameters that achieve given steady-state values of A_{11} and A_{33} in 1–3 simple shear flow.

fitCI finds the interaction coefficient C_I for the Folgar-Tucker model that achieves a given steady-state value of A_{11} in simple shear flow.

pDot gives the time derivative $\dot{\mathbf{p}}$ of an orientation vector $\mathbf{p}$ using Jeffery's equation. See Section 4.2.3.

solvePsi2D solves for the orientation distribution function $\psi_\phi(\phi, t)$ for the 2-D version of the Folgar-Tucker model. See Sections 5.2.1 and 5.3.21

solvePsi3D solves for the orientation distribution function $\psi(\theta, \phi, t)$ for the 3-D version of the Folgar-Tucker model. See Sections 5.2.2 and 5.3.2.

solvePsiARD solves for the orientation distribution function $\psi(\theta, \phi, t)$ for 3-D anisotropic rotary diffusion models. See Section 5.6.

tauFiber finds the extra-stress tensor $\boldsymbol{\tau}$ for a fiber suspension, for a given orientation tensor $\mathbf{A}$ and rate of deformation $\mathbf{D}$, Eqn. (6.29).

2.3 Fiber Length Prediction

fldRstar returns a matrix $[R^*]$ used in the non-dimensional version of the Phelps-Tucker fiber length model. This function is not used directly, but is required by `solveFLDstar`.

solveFLDstar solves a non-dimensional version of the Phelps-Tucker fiber length model.

2.4 Mechanical Property Prediction

C2eng finds the engineering constants for a given stiffness tensor $\mathbb{C}$.

Clayer2laminate finds the laminate stiffness matrices $[A]$, $[B]$ and $[D]$ for a laminate, given the stiffness tensor $\mathbb{C}$ for each layer and the layer thicknesses. This is used to compute the tensile and bending properties of a composite where the orientation varies across the thickness; see Section 8.4.5.

diluteEshelby finds the stiffness tensor $\mathbb{C}$ for a dilute composite with unidirectional alignment using Eshelby's equivalent inclusion. This model is primarily of theoretical interest, and it is used in Fig. 8.3.

eng2C converts the engineering constants for an orthotropic material into a stiffness tensor $\mathbb{C}$.

eshtens returns the Eshelby tensor $\mathbb{E}$ for a spheroidal particle in an isotropic matrix. The particle can be prolate (fiber-like), spherical, or oblate (disk-like).

halpin finds the engineering constants and the stiffness tensor $\mathbb{C}$ for a discontinuous fiber composite with unidirectional fibers using the Halpin-Tsai equations.

iso2C finds the stiffness tensor $\mathbb{C}$ for an isotropic material with Young's modulus E and Poisson ratio ν.

lielens returns the stiffness tensor $\mathbb{C}$ and thermal stress tensor $\boldsymbol{\beta}$ for a unidirectional composite using the Lielens/double-inclusion model.

mori returns the stiffness tensor $\mathbb{C}$ and thermal stress tensor $\boldsymbol{\beta}$ for a unidirectional composite using the Mori-Tanaka model.

oravg2 computes the orientation average of a transversely isotropic second-order tensor, Eqn. (8.81).

oravg4 computes the orientation average of a transversely isotropic fourth-order tensor, Eqn. (8.77).

2.5 Reconstruction of Orientation Distribution Functions

A2F find the orientation tensor $\mathbf{A}$ for any deformation gradient tensor $\mathbf{F}$ using the deformation form of Jeffery's equation.

drawPsi draws a sphere colored by the Jeffery orientation distribution function for a given deformation gradient tensor $\mathbf{F}$.

F2A returns the orientation tensor $\mathbf{A}$ for a given deformation gradient tensor $\mathbf{F}$ using Jeffery's model.

matchA, given a set of orientation vectors $\mathbf{p}^i$ and weights f^i, adjusts the weights to exactly match a given second-order or fourth-order orientation tensor, while minimizing the mean square difference between the original and adjusted weights. WARNING: This function can return negative values for some weights, usually when the orientation state is highly aligned. See `RecontructingDiscreteDistributionFcn.mlx` for examples.

2.6 Graphics and Utility Functions

fill3elt draws a triangular mesh in three dimensions, colored according to element values.

fill3mesh draws a triangular mesh in three dimensions, colored according to nodal values.

meshcon builds the edge connectivity information for a triangular mesh, as needed by `refinemesh`.

p2sph converts a set of $\mathbf{p}$ vectors to the angles (θ, ϕ) in a spherical coordinate system.

plot3mesh draws the elements of a triangular mesh in three dimensions.

plot3nodes draws the nodes of a mesh in three dimensions.

randomfibers generates a set of fiber orientation vectors $\mathbf{p}$, randomly oriented in three dimensions.

refinemesh refines a triangular mesh by dividing each initial element into n^2 smaller ones.

sph2p converts a set angles (θ, ϕ) in a spherical coordinate system into unit vectors $\mathbf{p}$.

spheremesh generates different types of triangular meshes on a unit sphere, or on half of a sphere.

sphTriArea finds the area of each triangle in a mesh on the unit sphere. Treats each element as a spherical triangle, rather than a planar triangle.

surfPsi3D colors the surface of a unit sphere according to an orientation distribution function $\psi(\mathbf{p})$. This function is designed to display distribution functions calculated by `solvePsi3D` and `solvePsiARD`.

weightFrac2volFrac converts fiber weight fraction to fiber volume fraction for a two-phase composite.