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

Version 1.1
June 20, 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.

Also included are functions from the article "Planar Fiber Orientation: Jeffery, Non-Orthotropic Closures and Reconstructing Distribution Functions," submitted to the *Journal of Non-Newtonian Fluid Mechanics*, June 2022.

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 topically, following the chapters of the book. Scripts related the planar orientation journal article are listed last.

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`.

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

JefferyFiberMotion.mlx calculates the motion of a single fiber following Jeffery's equation using `pDot` and `ode45` (Section 4.2.3). Jeffery orbits are illustrated, 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 determine 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 uses `AdotJeffQuad`.

SolvePlanarDistnFcn.mlx shows how to solve for the planar orientation distribution function $\psi_\phi(\phi, t)$ using `solvePsi2D`. 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 solve for the 3-D orientation distribution function $\psi(\mathbf{p}, t)$ using `solvePsi3D`. 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 tensor equations for flow-induced fiber orientation. The script shows solutions to the Folgar-Tucker equation (Sections 5.3 and 5.4), anisotropic rotary diffusion 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 that produces a given steady-state value of A_{11} in simple shear flow for the Folgar-Tucker model. This script also demonstrates 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 is independent of the kinetic parameter κ when an RSC or RPR model is used together with anisotropic rotary diffusion.

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 the calculations 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 using mean-field theories. 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, and Voigt averages are computed. 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).

LaminatedPlateProperties.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.

Journal article: Planar Orientation Distributions

PlanarDistnExamples.mlx calculates and plots example distribution functions for planar orientation. These include the Jeffery, ER and Bingham distributions, as well as several fourth-order maximum entropy distributions.

CreateNonOrthoClosure.mlx shows how to use `fitNonOrtho2D` to generate data for a variety of flow cases, fit a non-orthotropic (F) closure to the data, and add that closure to `closeA4planar`.

TestPlanarClosure.mlx compares transient results from a distribution function calculation to several different closure approximations. Both the tensor components and the scalar error measure are plotted.

TestPlanarReconstruction.mlx compares different reconstruction strategies for planar distributions (Jeffery; Bingham; ER; and ME4 with the natural, ER, and F closures and the exact fourth-order tensor) against distribution functions calculated using the Folgar-Tucker model. Transient errors are plotted, as well as the calculated and reconstructed distributions at the end time. While the book recommended the Jeffery distribution for planar reconstruction, the paper shows that the fourth-order maximum entropy distribution (ME4) with the best available $\mathbb{A}$ information is a better choice.

2 Functions Listed by Category

2.1 Operations on Orientation Tensors

Starting in version 1.1, all functions operate on either 3-D or 2-D (planar) tensors, except as noted. The functions determine whether to return 2-D or 3-D results based on the dimensions of the input matrices.

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

inv4 finds the tensor inverse of a fourth-order tensor, in 6×6 matrix form for 3-D or 3×3 in 2-D.

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

rotate4 performs a coordinate transformation on 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. In 2-D the matrix is 2×2 and the vector is 3×1.

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 by `matchA` in 3-D, and for 2-D distribution function recovery. In 2-D the matrix is 3×3 and the vector is 5×1.

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. 3-D orientation only.

transisoA returns the full second-order orientation tensor **A** and fourth-order orientation tensor $\mathbb{A}$ for a transversely isotropic orientation state with given values of A_{11} and $\mathbb{A}_{1111}$. 3- D orientation only.

vec2tens converts a symmetric second-order tensor from 6×1 column vector form to 3×3 matrix form. See Section 2.3.5.1. In 2-D the vector is 3×1 and the matrix is 2×2.

vec2tens4 converts a symmetric fourth-order tensor from 15×1 column vector form to 6×6 matrix form. In 2-D the vector is 5×1 and the matrix is 3×3 This is not discussed in the book, but is the inverse of `tens2vec4`. It can be used to convert fourth-order tensor results from `solvePsi3D` and `solvePsiARD` to matrix form.

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 3-D flow-induced orientation. The 2 in the name denotes a second-generation version of this function.

AdotPlanar gives the time derivative $\dot{\mathbf{A}}$ as a function of $\mathbf{A}$, $\mathbf{L}$ and orientation model parameters for planar orientation.

AdotJeffQuad gives the time derivative $\dot{\mathbf{A}}$ for the 3-D Jeffery model using the quadratic closure. This is a simplified version of `Adot2`, used in Section 5.1.2 and Fig. 5.1.

Asteady finds the steady-state 3-D orientation tensor $\mathbf{A}$ for a given velocity gradient $\mathbf{L}$, for any orientation model in `Adot2`.

AsteadyPlanar finds the steady-state planar orientation tensor $\mathbf{A}$ for a given velocity gradient $\mathbf{L}$, for any orientation model in `AdotPlanar`.

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 3-D closure approximations to find the fourth-order orientation tensor $\mathbb{A}$ corresponding to a second-order tensor $\mathbf{A}$. See Section 5.4.

closeA4planar uses any of several closure approximations to find the planar fourth-order orientation tensor $\mathbb{A}$ corresponding to a planar second-order tensor $\mathbf{A}$. This includes the non-orthotropic F closures.

fitARD finds the 3-D 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 3-D Folgar-Tucker model that achieves a given steady-state value of A_{11} in simple shear flow.

fitNonOrtho2D determines the polynomial coefficients for a non-orthotropic, planar closure (F closure) for a given interaction coefficient and set of flow cases. The utility function `printCoeff` is useful for formatting the results for pasting into `closeA4planar`.

fitOrtho2D determines the polynomial coefficients for an orthotropic, planar closure (D closure) for a given interaction coefficient. The utility function `printCoeff` is useful for formatting the results for pasting into `closeA4planar`.

pDot gives the time derivative $\dot{\mathbf{p}}$ of a 3-D 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 3-D 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 is used in Fig. 8.3.

eng2C converts the engineering constants for an orthotropic material into a stiffness tensor $\mathbb{C}$. See also `iso2C`.

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, stiffness tensor $\mathbb{C}$, and thermal stress tensor $\boldsymbol{\beta}$ 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 3-D orientation average of a transversely isotropic second-order tensor, Eqn. (8.81).

oravg4 computes the 3-D orientation average of a transversely isotropic fourth-order tensor, Eqn. (8.77).

2.5 Reconstruction of Orientation Distribution Functions

A2F finds the 3-D deformation gradient tensor $\mathbf{F}$ for any orientation tensor $\mathbf{A}$ using the deformation form of Jeffery's equation.

F2A returns the 3-D orientation tensor $\mathbf{A}$ for a given deformation gradient tensor $\mathbf{F}$ using Jeffery's model.

fitBingham2D finds the Bingham distribution that matches a given planar second-order orientation tensor $\mathbf{A}$.

fitERdistn2D finds the ellipse radius (ER) distribution that matches a given planar second-order orientation tensor $\mathbf{A}$.

Jefferydistn2D finds the Jeffery distribution that matches a given planar second-order orientation tensor $\mathbf{A}$.

fitMaxEntropy2D finds the fourth-order maximum entropy distribution (ME4) that matches a given planar fourth-order orientation tensor $\mathbb{A}$.

matchA, given a set of 3-D 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 `RecontructDiscreteDistributionFcn.mlx` for examples.

2.6 Graphics and Utility Functions

drawPsi draws a sphere colored by the 3-D Jeffery orientation distribution function for a given deformation gradient tensor $\mathbf{F}$.

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 3-D $\mathbf{p}$ 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.

polyfit2 finds a polynomial function of two variables $\hat{z}(x, y)$ that best fits a set of data $z_i(x_i, y_i)$. Equality constraints can be enforced if desired.

polyfit2 evaluates a polynomial function of two variables $\hat{z}(x, y)$ using coefficients in the form returned by `polyfit2`.

printCoeff formats a vector of polynomial coefficients and prints them to the console. From there, they can then be copied and pasted into a MATLAB code such as `closeA4planar`.

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 of angle pairs (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. Each element is treated 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.