Exploring Position Dependent Property/Microstructure Relationships in Polycrystal Materials via Probabilistic Graphical Model Techniques

Presenter: Peng Chen
Advisor: Nicholas Zabaras
Materials Process Design and Control Laboratory
Sibley School of Mechanical and Aerospace Engineering
101 Frank H. T. Rhodes Hall, Cornell University
Ithaca, NY 14853-3801

Work supported by NSF, AFOSR and an OSD/AFOSR MURI on UQ
Motivation

Undirected graphical model
- Nonparametric representation
- Graph learning
  - Gaussian model
  - Nonparametric model
- Belief propagation
  - Nonparametric belief propagation
  - Loop belief propagation

Application to Multiscale disk forging problem
- Graphical representation
- Microstructure representation

Numerical Examples
- Final properties prediction
- Final texture prediction
- Initial texture prediction (inverse problem)

Future Plan
Give a probabilistic prediction

Simulation process

Motivation

Materials Process Design and Control Laboratory
An undirected graph, or pairwise Markov Random Field (MRF) is defined by a set of nodes $V$, and a corresponding set of edges $E$.

In this work, we consider an undirected graph $G$ with structure given as the right one, which contains two levels, the input variables $X = \{x_i; i \in \mathcal{V}\}$ and the output variables $Y = \{y_i; i \in \mathcal{V}\}$.

The joint distribution can be factorized as products of all the pairwise potential functions,

$$p(X, Y) = \frac{1}{Z} \prod_{(i,j) \in E} \psi_{i,j}(y_i, y_j) \prod_{i \in \mathcal{V}} \phi_i(x_i, y_i)$$
For graphical models with continuous variables, the most common way to represent the potential function is using Gaussian functions. Let \( \mathcal{N}(x; \mu, \Sigma) \) denotes a normalized Gaussian density with mean \( \mu \) and covariance \( \Sigma \), evaluated at \( x \).

For the problems that we are interested in (i.e. material problems), the stochastic relationship between different variables are so complicated that cannot be simply modeled by the Gaussian functions.

Hence, in this work, all the potential functions are designed to be nonparametric, that is, Gaussian mixture function, then the potential functions take the form:

\[
\psi_{i,j}(y_i, y_j) = \sum_{m=1}^{M} \omega^{(m)} \mathcal{N} \left( |y_i - y_j|; \mu^{(m)}, \Sigma^{(m)} \right)
\]

\[
\varphi_i(x_i, y_i) = \sum_{m=1}^{M} \omega^{(m)} \mathcal{N} \left( [x_i, y_i]; \mu_{xy}^{(m)}, \Lambda^{(m)} \right)
\]

where \( \omega^{(m)} \) is the weight associated with the \( i \)th kernel.
Graphical Model Learning

- Gaussian Model

In the Gaussian model, all the potential functions are modeled by Gaussian functions \( \mathcal{N}(x; \mu, \Sigma) \), the unknown parameters are \( \mu and \Sigma \).

Given a set of training data, the log likelihood function is:

\[
\log L(X; \mu, \Sigma) = \sum_{n=1}^{N} \log \mathcal{N}(x_n; \mu, \Sigma)
\]

Take the derivative w.r.t \( \mu and \Sigma \) respectively, set them equal to zero, then we can obtain the maximum likelihood estimators as:

\[
\hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n
\]

\[
\hat{\Sigma} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \hat{\mu})(x_n - \hat{\mu})^T
\]
Nonparametric Model

In the Nonparametric model, all the potential functions are modeled as Gaussian mixture, \( \sum_{k=1}^{K} \omega_k \mathcal{N}(x_n; \mu_k, \Sigma_k) \), the unknown parameters are \( \omega_k, \mu_k \) and \( \Sigma_k \).

Given a set of training data, the log likelihood function is:

\[
\log p(X; \omega, \mu, \Sigma) = \sum_{n=1}^{N} \log \left( \sum_{k=1}^{K} \omega_k \mathcal{N}(x_n; \mu_k, \Sigma_k) \right)
\]

The unknown parameters are learned by Expectation-Maximization algorithm:

1. Initialize the means \( \mu_k \), covariance \( \Sigma_k \) and mixing coefficient \( \omega_k \)
2. E step: Evaluate the responsibilities using the current parameter values
   \[
   \gamma(z_{nk}) = \frac{\omega_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \omega_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}
   \]
3. M step: Re-estimate the parameters using the current responsibilities
   \[
   \mu_{k}^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n, \quad \omega_{k}^{new} = \frac{N_k}{N}, \quad \Sigma_{k}^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_{k}^{new})(x_n - \mu_{k}^{new})^T
   \]
   where \( N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \)
Inference problem

Inference problem is typical problem that predict the conditional distributions of hidden variables in the graphical model via belief propagation method.

Markov Blanket for \( y_{i,j} \)

What we have:
- all the parameters associated with the graph from the learning process
- a new coming input

What we want to know:
- the conditional distribution for \( y \) given a new \( x \)
The marginal for $p(y_{ij})$ can be calculated by

$$p(y_{i,j}) \propto \varphi_{i,j}(x_{i,j}, y_{i,j}) \prod_{y_{k,l} \in \Gamma(y_{i,j})} m_{y_{k,l}}(y_{i,j})$$

The message update function is

$$m_{i-1,j}(y_{i,j}) \propto \int \psi(y_{i,j}, y_{i-1,j}) \varphi_{i-1,j}(x_{i-1,j}, y_{i-1,j}) \prod_{(p,q) \in \Gamma(y_{i-1,j}) \setminus y_{ij}} m_{p,q}(y_{i-1,j}) dy_{i-1,j}$$
Nonparametric Belief Propagation

Algorithm: NBP update of the nonparametric message $m_{k,l}(y_{i,j})$ sent from node k,l to neighboring node i,j. (Erik B. Sudderth, 2010)

Given input messages $m_{p,q}(y_{k,l}) = \{\mu_{p,q}^{(i)}, \Lambda_{p,q}^{(i)}, \omega_{p,q}^{(i)}\}_L$ for each $(p,q) \in \Gamma(y_{i-1,j}) \setminus y_{ij}$, construct an output message $m_{k,l}(y_{i,j})$ as follows:

1. Determine the marginal influence $\zeta(y_{k,l})$:
   $$\zeta(y_{k,l}) = \int \psi(y_{k,l}, y_{i,j}) dy_{i,j}$$

2. Draw N independent samples from the product
   $$y_{kl}^{(r)} \sim \zeta(y_{k,l}) \psi_{k,l}(x_{k,l}, y_{k,l}) \prod_{(p,q) \in \Gamma(y_{k,l}) \setminus y_{ij}} m_{p,q}(y_{k,l})$$

GMRC is first adopted to reduce the product and then exact sampling is applied

3. For each $y_{kl}^{(r)}$, use importance sampling to sample from
   $$y_{i,j}^{(r)} \sim \psi(y_{i,j}, y_{k,l} = y_{kl}^{(r)})$$

   set $\omega_{i,j}^{(r)}$ as the importance weight and choose $\Lambda_{i,j}^{(r)}$ by “rule of thumb” theorem

4. treat $m_{k,l}(y_{i,j})$ as $m_{k,l}(y_{i,j}) \propto \sum_{i=1}^{N} \omega_{i,j}^{(r)} \mathcal{N}(y_{i,j}^{(r)}, \Lambda_{i,j}^{(r)})$, use GMRC to reduce it to L component

Remember in our case, all the beliefs are Gaussian Mixture. The Belief update of belief equation in the previous slide is then defined by a product of mixtures.

As illustrated in the following figure, the product of $d$ Gaussian mixtures, each containing $L$ components, is itself a mixture of $L^d$ Gaussians. While in principle this belief update could be performed exactly, the exponential growth in the number of mixture components quickly becomes intractable.

A product of three mixtures of $L = 4$ 1D Gaussians results in a Gaussian mixture with $4^3 = 64$ components.
Basic Algorithm

\[ f(x) = \sum_{i=1}^{N} \omega_i \cdot \mathcal{N}(x; \mu_i, \Sigma_i) \]

reduce

\[ f(x) = \sum_{j=1}^{L} \omega_j \cdot \mathcal{N}(x; \mu_j, \Sigma_j) \]

I. Preprocessing
Apply any top-down reduce method to compute an initial clustering. The simplest is merging two mixture components with the smallest dissimilarity by

\[
\omega_m = \omega_i + \omega_j \\
\mu_m = \frac{1}{\omega_m} \left( \omega_i \mu_i + \omega_j \mu_j \right) \\
\Sigma_m = \frac{1}{\omega_m} \left( \omega_i \Sigma_i + \omega_j \Sigma_j + \frac{\omega_i \omega_j}{\omega_m} (\mu_i - \mu_m)(\mu_j - \mu_m)^T \right)
\]

II. K-means Clustering
- start from the above initial cluster arrangement
- for each \( i \in \{1, ..., L\} \), let cluster \( C_i \) be the set of sites in S closer to centers \( c_i \) than to any other center \( c_j \) with \( j \neq i \) f.a. \( i \in \{1, ..., N\} \)
- for each \( i \in \{1, ..., L\} \), set \( c_i \) to be the center of mass of all sites in cluster \( C_i \)
- Repeat the above process until convergence
An efficient way to compute local marginals:

- Pick $y_{nn}$ as root node
- Compute and propagate messages from $y_{11}$ to the root node $y_{nn}$, storing received messages at every node.
- Compute and propagate messages from the root node $y_{nn}$ to $y_{11}$, storing received messages at every node.
- Compute the product of received messages at each node for which the marginal is required, and normalize if necessary.
Application to Multiscale Disk Forging Problem

Simulation process

Multiscale Forging simulation

Probabilistic Graphical model representation

Macro properties

Initial Reduced Microstructure representation

final Reduced Microstructure representation

Give a probabilistic prediction for a new microstructure input
Each integration point corresponds to a small cell in the graphical model.

The initial micro nodes are the low dimensional representation for the initial microstructure, the Macro nodes represent the macroscopic properties after the forging process; the Micro scale nodes represent the low dimensional representation for the final texture;

The dashed line in the graphical model denotes the hidden correlations between variables, which in our case being reflected by other potential function (dash line).
- At each macro scale point, due to the randomness of the microstructure, we will have a set of (microstructure) samples (realizations).

- One can construct a reduced order stochastic model for each macroscopic point (e.g. PCA/KPCA).

- Without model reduction technique, it will result in a huge dimension of the random space and thus this problem becomes computationally intractable.
Consider a set of D dimensional data

\[ \{ x_n; n = 1, \ldots, N \} \]

The mean is

\[ \bar{x} = E[x] = \frac{1}{N} \sum_{n=1}^{N} x_n \]

The covariance is

\[ C = \frac{1}{N} \sum_{n=1}^{N} (x_n - \bar{x})(x_n - \bar{x})^T \]

Find the Eigenvalue and Eigenvector for the covariance matrix C

\[ Cv_i = \lambda_i v_i \]

Sort the Eigenvalue \( \lambda_i \), and take k eigenvectors corresponding to the largest k Eigenvalues to assemble a new Matrix A, then we can reduce the original data set to K dimension via matrix A as

\[ y_n = A_{KxD} \left( x_n - \bar{x} \right) \]

where \( y_n \) is of k dimension
The initial texture field is obtained from a pre-processing step. In the pre-processing step, a set of raw ingots, which have the same grain size and orientations everywhere but different surfaces represented by the Bezier curves, are used as an input to the deterministic flat-die forging process. Since all workpieces go through distinct deformation processes due to their unique surface shapes, the resultant microstructures will vary from point to point and from sample to sample. Totally 1000 training input data are sampled from the above process.

In this work, assume each microstructure has 20 Grains, which means one microstructure realization has 60 dimension, and we used a 12x8 2D grid, and 4 gauss points in each element, so the corresponding graphical model should have 384 macroscopic nodes.

The dimensionality of microstructure representation is reduced to 5 by PCA.

A comparison of Gaussian Model and Nonparametric Model is shown in the following slides.
Strain field Prediction

Strain field Prediction Given a new initial Texture:

Each component of the Nonparametric model:

- **mean**
- **std**
- **weight**
Stress field Prediction

Given a new initial Texture:

True

Gaussian Model

Nonparametric Model

Each component of the Nonparametric model:

mean

std

weight
Young’s Modulus field Prediction

Young’s modulus field Prediction Given a new initial Texture:

Each component of the Nonparametric model:

- **mean**
- **std**
- **weight**
Shear Modulus field Prediction

Shear Modulus field Prediction Given a new initial Texture:

Each component of the Nonparametric model:

- **mean**
- **std**
- **weight**
Reconstruction for the Marginals

Comparison of the reconstructed marginal stress distribution by Gaussian model and the nonparametric model and the true marginal stress distribution at three randomly chosen points.
Convergence of the loopy belief propagation algorithm for the forging example.

\[ \eta = \frac{1}{ML} \sum_{i,j} \sum_{l=1}^{L} \left\{ \| \mu_{ij}^{(l)(t+1)} - \mu_{ij}^{(l)(t)} \|^2 + \| \sigma_{ij}^{(l)(t+1)} - \sigma_{ij}^{(l)(t)} \|^2 \right\}. \]
**Eigen plots for the textures**

Initial microstructure:

![Graph](image1)

*The largest 20 Eigenvalues of the microstructure data at a random macroscopic point obtained by PCA method.*

final microstructure:

![Graph](image2)

*The largest 20 Eigenvalues of the final texture data at a random macroscopic point obtained by PCA method.*
Final Texture Prediction Given a new initial Texture:
Final Texture Prediction

Orientation density function prediction at a random point:

Pole figure for the Initial texture

Pole figure for the final texture

Pole figure for the reconstructed final texture by Gaussian model

Pole figure for the reconstructed final texture by the Nonparametric model
Final Texture Prediction

Orientation density function prediction at a random point:

Pole figure for the Initial texture

Pole figure for the final texture

Pole figure for the reconstructed final texture by Gaussian model

Pole figure for the reconstructed final texture by the Nonparametric model
Initial Texture Prediction

Initial Texture Prediction Given Final Properties

Desired
Gaussian Model
Nonparametric Model

Gaussian Model
Nonparametric Model
Initial Texture Prediction

Initial Texture Prediction Given Final Texture:

- Gaussian Model
- Nonparametric Model
Future Plan

Given: a set of observation \( \mathbf{D} = \{D_1, \ldots, D_m\} \)

where \( m \) is the number of observation, \( D_i \) is one observation for all the variables

Known condition: all the variables in the graphical model are correlated

Objective: find the hidden variables such that the resultant graphical model can be interpreted by the minimum independent parameters
Thank you!