Identification of statistical parameters of FIPs

Hugo Lancery, Mathieu Klingler

March 31, 2023

Abstract

Microstructure-scale interactions involving crystallographic texture (orientation and disorientation distributions), distributions of grain shape and size, nearest neighbor grains/phases, etc. in polycrystals can be simulated using the Crystal Plasticity Finite Element Method (CPFEM). Digital statistical volume element (SVE) instantiations that comprise a significant number of grains are analyzed by CPFEM to compute fatigue indicator parameters (FIP) which are used as surrogate measures of the driving force for fatigue crack formation within the first grain or nucleant phase.

1 Data import and computation

First let's import properly all the data (.csv): 1000 FIP randoms,200 FIP X dir. and 200 FIP Z dir. We begin with all the importation needed :

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import os
```

Now let's import and build our data as an array for optimal python usage with numpy:

```
def data_in_line(dir_path):
    """ Return the data in one big matrix """
    csv_files_random = [f for f in os.listdir(
        dir_path) if f.endswith('.csv')]
    data = []
    for file in csv_files_random:
        file_path = os.path.join(dir_path, file)
        df = pd.read_csv(file_path).values.tolist()[:200]
        i = 0
        for i in range(len(df), 282):
            df.append([0.0])
        data.append(df)
    data_final = (np.array(data)[:, :, 0])
    return data_final
```

Wich will return a 2D Matrix with a shape such as $(1000, 282)$ for the test database. We will now compute the function to calculate our FIP and linearize the CDF:

```
def main(data_re):
    def ecdf(data):
        """ Compute ECDF """
        x = np.sort(data)
        n = x.size
        y = np.arange(1, n+1) / n
        return (x, y)

    x, Fx = ecdf(data_re)
    linearized_Fx = -np.log(-np.log(Fx))

    return x,linearized_Fx
```

2 Identification of μ and σ

Let's use the data base of X and Z direction FIPs for our training part.

In this Section, statistical parameters in the extrapolation formulations are identified for the prediction of the maximum FIPs in a large volume. This requires fitting the FIPs in the probability plot. In Fig. 1, the random dataset of FIPs collected from 200 SVEs (containing 1 x 282) are plotted on the Gumbel probability plot along with linear fits estimated via least squares regression.

The statistical parameters and are identified using different training dataset: the random, basal and transverse textures.

data	μ	σ
Random	3.87e-3	3.85e-4
Transverse(X)	-2.15e-3	5.68e-4
Transverse(Z)	2.89e-3	5.1e-4

SWITCH PAGE

2.1 Identification of μ and σ in Gumbel distribution in random data base

We begin to import the data in 2D, then we convert it into a 1-array dimension list to apply the ecdf to it. We get in return the CDF that we linearized and FIP values, wich allow us to plot the prediction affine curves to the maximum plastic strain and stress in polycrystals.

```
dir_path = 'C:/Users/.../FIP_data/FIP_each_SVE_random'
data_2D=data_in_line(dir_path)
size_x,size_y=np.shape(data_2D)
new_size_x=int(size_x/n)
x_max=[]
Fx_lin_max=[]

"""We convert the matrix into a 1D shape, to calculate the ecdf."""
data_1D = np.reshape(data_2D,(1,size_y*size_x))
"""ECDF function wich returns FIP and CDF linearized."""
x_rand,Fx_lin_rand=main(data_1D)
x_rand=x_rand[0][:-1]
Fx_lin_rand=Fx_lin_rand[:-1]

"""We seperate the data to use only the max's values."""
"""We will study the FIP data inclueded to get the mu and sigma"""
itemindex = np.where(x_rand < 0.006)[0][-1]
x0 = x_rand[itemindex]
Fip0 = Fx_lin_rand[itemindex]
excluded_x, excluded_linearized_Fx = x_rand[:itemindex], Fx_lin_rand[:itemindex]
included_x, included_linearized_Fx= x_rand[itemindex:], Fx_lin_rand[itemindex:]

"""Then we calculate our affine coefficient "a" and "b"."""
a_rand, b_rand = np.polyfit(included_x, included_linearized_Fx, 1)
sigma_rand = 1/a_rand
mu_rand = -sigma_rand*b_rand
```

This code gives us this result after ploting:

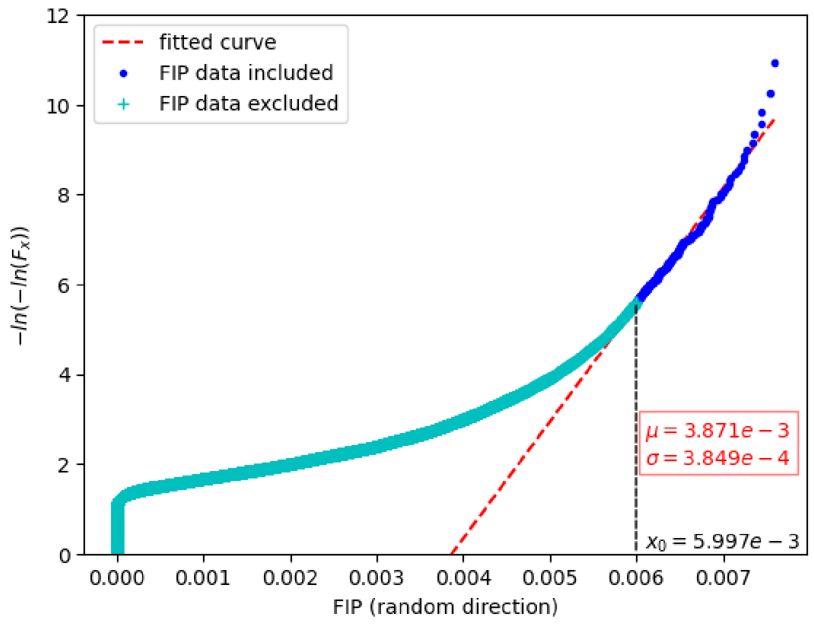

Figure 1: probability plot in random direction

2.2 Identification of μ and σ in Gumbel distribution in X direction

We begin to import the data in 2D, then we convert it into a 1-array dimension list to apply the ecdf to it. We get in return the CDF that we linearized and FIP values, wich allow us to plot the prediction affine curves to the maximum plastic strain and stress in polycrystals.

```
dir_path = 'C:/Users/.../FIP_data/FIP_each_SVE_X_basal_transverse'
data_2D=data_in_line(dir_path)
size_x,size_y=np.shape(data_2D)
new_size_x=int(size_x/n)
x_max=[]
Fx_lin_max=[]

"""We convert the matrix into a 1D shape, to calculate the ecdf."""
data_1D = np.reshape(data_2D,(1,size_y*size_x))
"""ECDF function wich returns FIP and CDF linearized."""
x_Z,Fx_lin_Z=main(data_1D)
x_Z=x_Z[0][:-1]
Fx_lin_Z=Fx_lin_Z[:-1]

"""We seperate the data to use only the max's values."""
"""We will study the FIP data inclueded to get the mu and sigma"""
itemindex = np.where(x_Z < 0.006)[0][-1]
x0 = x_Z[itemindex]
Fip0 = Fx_lin_Z[itemindex]
excluded_x, excluded_linearized_Fx = x_Z[:itemindex], Fx_lin_Z[:itemindex]
included_x, included_linearized_Fx= x_Z[itemindex:], Fx_lin_Z[itemindex:]

"""Then we calculate our affine coefficient "a" and "b"."""
a_Z, b_Z = np.polyfit(included_x, included_linearized_Fx, 1)
sigma_Z = 1/a_Z
mu_Z = -sigma_Z*b_Z
```

This code gives us this result after ploting:

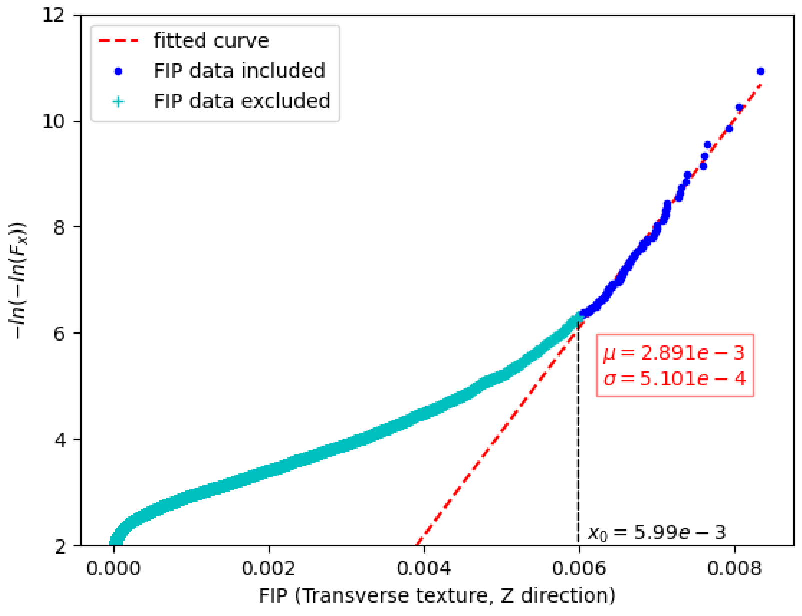

Figure 2: probability plot Transverse textures loading in X direction

2.3 Identification of μ and σ in Gumbel distribution in Z direction

We begin to import the data in 2D, then we convert it into a 1-array dimension list to apply the ecdf to it. We get in return the CDF that we linearized and FIP values, wich allow us to plot the prediction affine curves to the maximum plastic strain and stress in polycrystals.

```
dir_path = 'C:/Users/.../FIP_data/FIP_each_SVE_Z_basal_transverse'
data_2D=data_in_line(dir_path)
size_x,size_y=np.shape(data_2D)
new_size_x=int(size_x/n)
x_max=[]
Fx_lin_max=[]

"""We convert the matrix into a 1D shape, to calculate the ecdf."""
data_1D = np.reshape(data_2D,(1,size_y*size_x))

"""ECDF function wich returns FIP and CDF linearized."""
x_X,Fx_lin_X=main(data_1D)
x_X=x_X[0][:-1]
Fx_lin_X=Fx_lin_X[:-1]

"""We seperate the data to use only the max's values."""
"""We will study the FIP data inclueded to get the mu and sigma"""
itemindex = np.where(x_X < 0.0025)[0][-1]
x0 = x_X[itemindex]
Fip0 = Fx_lin_X[itemindex]
excluded_x, excluded_linearized_Fx = x_X[:itemindex], Fx_lin_X[:itemindex]
included_x, included_linearized_Fx= x_X[itemindex:], Fx_lin_X[itemindex:]

"""Then we calculate our affine coefficient "a" and "b"."""
a_X, b_X = np.polyfit(included_x, included_linearized_Fx, 1)
sigma_X = 1/a_X
mu_X = -sigma_X*b_X
```

This code gives us this result after ploting:

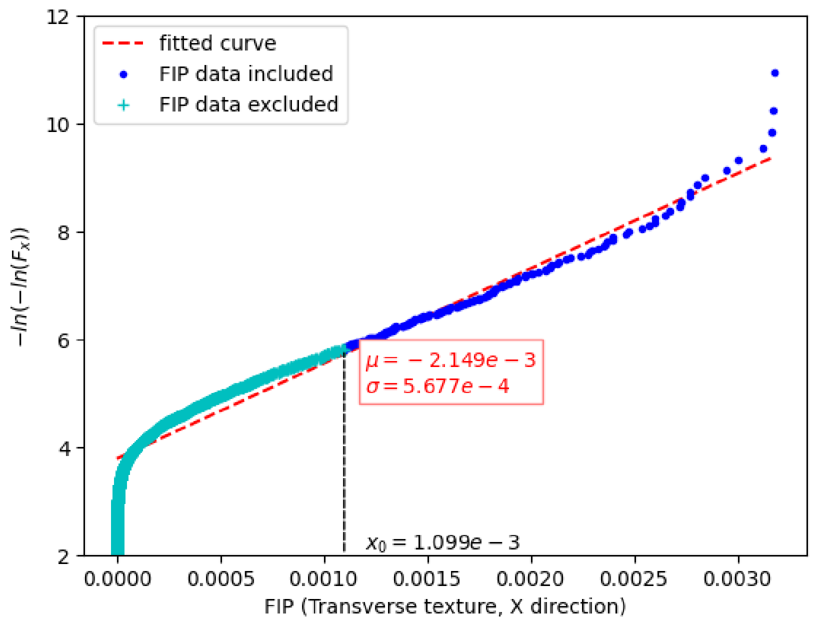

Figure 3: probability plot Transverse textures loading in Z direction

3 Prediction of the CDF of the max FIP_s $F_{X_{max}}$

3.1 Random texture data

Let's use the data base of random FIPs divided in a 200 group for our training part and 800 for the testing part.
We begin to import the data in 3D, then we convert it into a 2D-array dimension list to apply the ecdf to it with a loop for each of our group of SVEs. We get in return the CDF that we linearized and FIP values, wich allow us verify the prediction affine curves to the maximum plastic strain and stress in polycrystals.

```
dir_path='C:/Users/.../FIP_data/FIP_each_SVE_random'
data_2D=data_in_line(dir_path)
size_x,size_y=np.shape(data_2D)
n=25
new_size_x=int(size_x/n)
x_max=[]
Fx_lin_max=[]

"""We convert the matrix into a 3D shape."""
data_3D = np.reshape(data_2D,(new_size_x,n,size_y))

"""We convert the matrix into a 2D shape to then apply our main function."""
data_3D_2D = np.reshape(data_3D,(new_size_x,size_y*n))
data_3D_max=[]
for i in range(0, new_size_x):
    data_3D_max.append(data_3D_2D[i,:].max())
x,Fx_lin=main(data_3D_max)
```

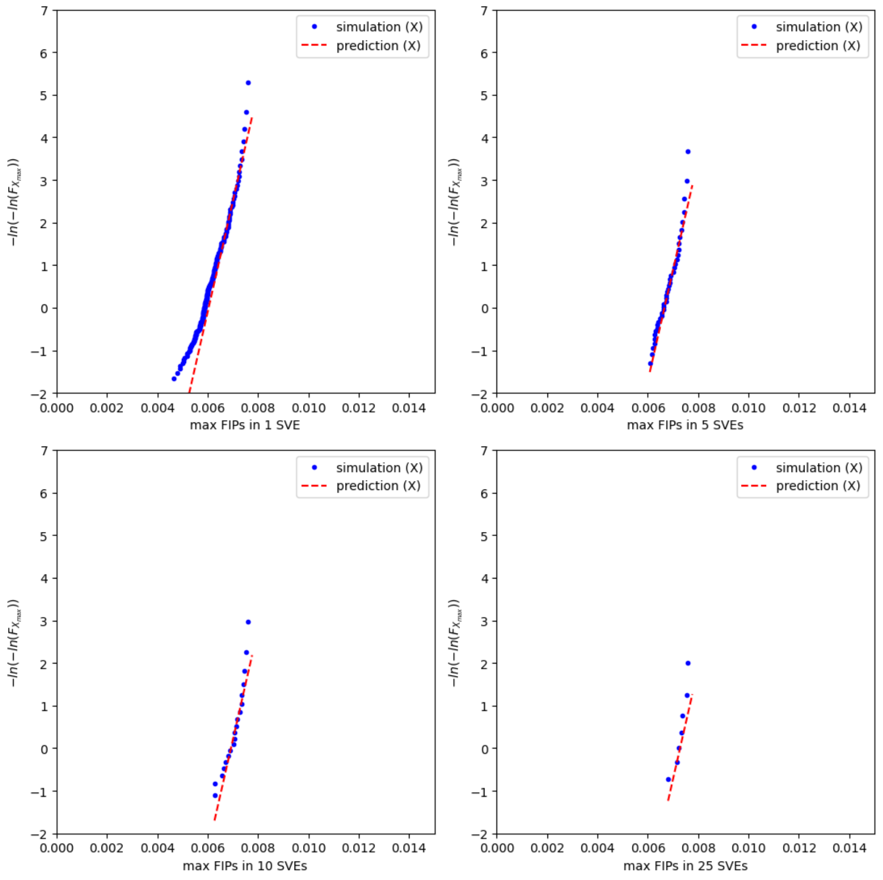

Figure 4: Gumbel EVD probability plot of the maxFIPs in 1 SVE, SVEs, 25 SVEs and 50 SVEs.

3.2 Basal/Transverse (X and Z direction) texture

Let's use the data base of X and Z direction FIPs for our training part.
We begin to import the data in 3D, then we convert it into a 2D-array dimension list to apply the ecdf to it with a loop for each of our group of SVEs. We get in return the CDF that we linearized and FIP values, wich allow us verify the prediction affine curves to the maximum plastic strain and stress in polycrystals.

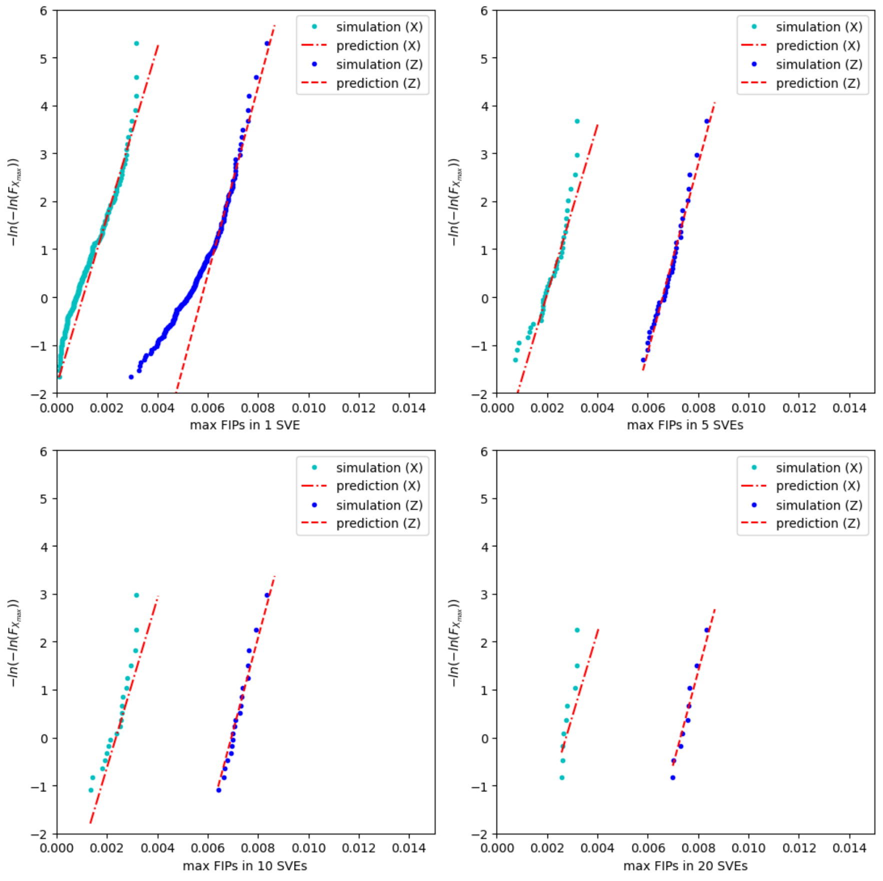

Figure 5: Gumbel EVD probability plot of the maxFIPs in 1 SVE, SVEs, 25 SVEs and 50 SVEs.

4 Validation of the prediction

4.1 Study of the Mmax value

We find that the differences among these three cases become more pronounced when a larger sample size n is considered, the differences of Mmax between transverse (X,Z) and random textures are 0.1 ×10 3 and 0.5 × 10 3 for n=100 SVEs

We can see the prediction is accurate but still has a small error, Furthermore we can assume that our random texture data base is composed of many transverse X textures since the curves are closer one to each other.

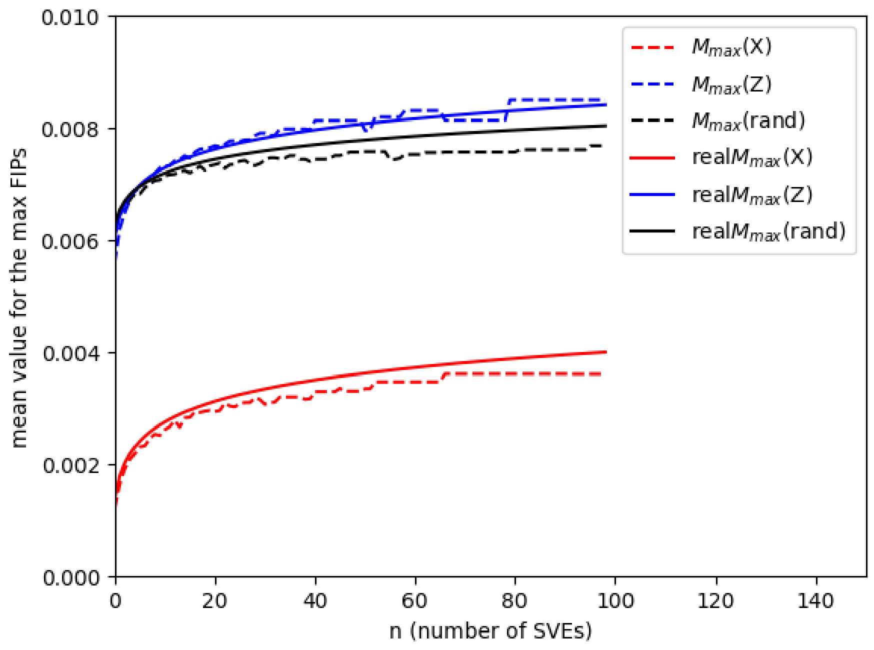

Figure 6: Prediction of mean value M_{max} for the maximum FIPs in terms of the corresponding target sample sizes n up to 100 SVEs.