A C++ package to fit mobility function upon ParaDiS results

Chaoming Yang

1. Installation

Prerequisite packages:

Armadillo: A C++ library for linear algebra & scientific computing <http://arma.sourceforge.net/>

After installed Armadillo, put its installed path in the Makefile

CINCLUDE =

CLINK =

For include and link the library.

Then under the src folder, simply do

make -j 4 all

will build an executable file named **ftDD.exe**

2. Usage

2.1 Run the program

The current code implements linear mobility function



Fitting variables are and .

Please note: The code use the above equation, and there is no precipitate properties such as precipitate shapes, or precipitate densities, because we do not know what correlation is. More details are discussed in section 3;

A typical running command is

*ftDD.exe -p parameter\_filename -i jobkey\_filename*

Where parameter\_filenameand jobkey\_filename are input files

**A. parameter\_filename** specifies the job type and some materials parameters, Example of a parameter\_filename

“ptype ma

burg 2.86e-10

taylor 0.40824829046386307

mu 2.7e10

mob 2384.59

alp 0.23709

bta 8719600.252960484

”

Explantion of the example parameter\_filename file

1. “ptype ma” specifies “ptype” tells the specific types of job, “ma” means fit **m**obility and **a**lpha

Here are all options

“a” means only fit alpha, mobility is fixed, and beta is zero.

“b” means only fit beta, mobility and alpha are fixed.

“mab” is to fit **m**obility and **a**lpha and **b**eta.

“0” is fit nothing, simply do one step calculation. (this is for checking results)

2. “burg 2.86e-10

taylor 0.40824829046386307

mu 2.7e10”

They are materials constants.

3 “mob 2384.59

alp 0.23709

bta 8719600.252960484”

They are initial guess of three variable values, **mob**ilty, **alp**ha and **bet**a, and they are not necessary to be close to final value.

**B. jobkey\_filename** specifies the folder of ParaDiS results that we want to fit.

To fit ParaDiS Results are in xxx\_results, then specify xxx in the **jobkey\_filename**

An Example of **jobkey\_filename:**

“AluPrecA

“

Then the code will read flux and properties in AluPrecA\_resultsand run the fitting.

We can fit **several runs together**.

For example, we use jobkey\_filename:

“ AluPrecA AluPrecB AluPrecC AluPrecD AluPrecE

”

Then we fit parameters using data from AluPrecA\_results, AluPrecB\_results, … together.

Please be sure that when we specify which ParaDiS data to fit, we use proper “ptype” described above.

Outputs:

The program outputs two files **check\_value** file and **variable\_result** file

**A. Check\_value** file is chk{xxx}.txt where {xxx} is the prefix of ParaDiS results file “xxx\_results”.

The data contains five columns, they are

time, stress, density, velocity and **fitted velocity**

So that you can plot the results based on chk{xxx}.txt to see how good is your fitting.

**B. variable\_result** is the file that stores the fitted variables, a typical one has one line as

“Error = 480.36 mob = 1367.61 alpha = 0.0605381 beta = 7.60314e+13”

You may work on the above formula to calculate units.

2.2 Use the program to do fitting based on Al Precipitate fitting.

Step 1. Fit pure Al to determine **M**obility and Alpha.

In **parameter\_filename,** use ptype “ma” to fit **m**obility and **a**lpha.

In **jobkey\_filename**, you can fit them together by specify “Alue4 Alue5 Alue6”

Or you can fit them individually.

Each time you will get a res.txt and chkxxx.txt so you can check your fitting.

Step 2. Fit Al-Precipitate to get Beta.

In **parameter\_filename,** use ptype “b” to fit **m**obility and **a**lpha.

In **jobkey\_filename**, you can fit them together by specify “AluPrec1 AluPrec2 AluPrec3 AluPrec4 AluPrec5” to fit all t-type precipitates or “AluPrecA AluPrecB AluPrecC AluPrecD AluPrecE” to fit all alpha precipitates. Or you can fit them individually.

3. Notes.

1. We only fit flow stress, that means we discard portion of data that is not flow regime. How many we discard depends on the specific data and how good the fitting actually is.

To determine starting from where is flow regime, we specify it in function *prepData* in file ftDDinit.cpp.

The code piece is “

void ftDD::prepData(const string& key) {

// start fitting from the ns data points (only do flow stress)

**int start\_point = 1200; //**

…

”

This means that starting from 1200th line, we consider it’s flow stress regime.

It’s not guaranteed that starting from 1200 will be flow stress, please check it and change it according to specific ParaDiS results.

2. The code use the above equation, there is no precipitate properties such as precipitate shapes, or precipitate densities, because we do not know what correlation is.

Therefore, the “beta” variable absorb all information about precipitates.

The next step is to find how beta is related to exact precipitates.

For example, beta may be linear to volume fraction. Or beta may be linear to density that is defined by this  where V is volume of precipitates, and r is precipitate radius. Or beta may be relate to precipitate shape.