FoMpy Documentation

Release 0.2.0

Gabriel Espiñeira

Mar 12, 2019

CONTENTS:

CHAPTER

ONE

GETTING STARTED

FoMPy is an effective tool that extracts the main figures of merit (FoM) of a semiconductor's IV curve and provides useful statistical parameters for variability studies. It includes several methods to extract the threshold voltage.

In the figure below the used can see the basic workflow behind the FoMpy library:

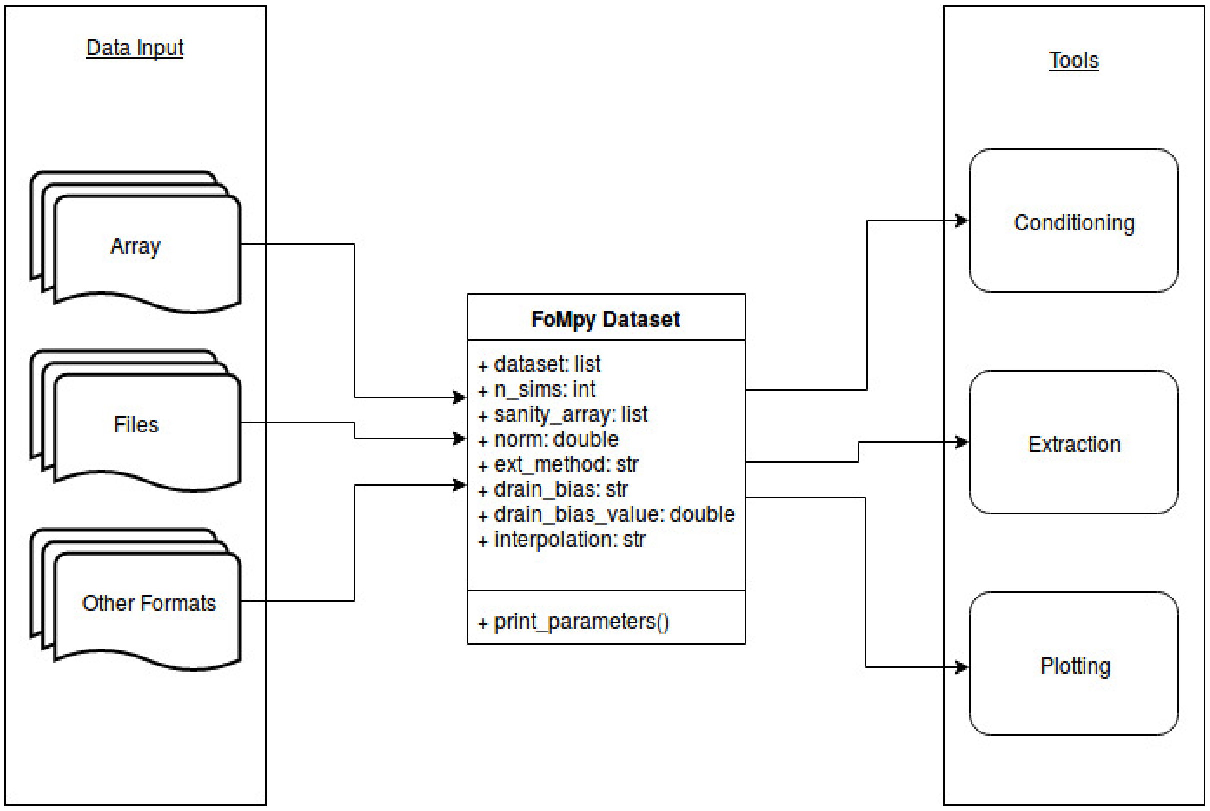

Fig. 1: Diagram showing the basic methodology of the FoMpy library. After loading the data into a FoMpy Dataset, using the various tools implemented in the library, the user is able to process and extract important parameters from a given dataset.

1.1 Getting FoMpy

The source code of FoMpy can be downloaded from https://gitlab.citius.usc.es/gabriel.espineira/FoMPy/ and is purely intended to run as a library for Linux systems (Debian, Ubuntu, FreeBSD, . . .). FoMpy uses several external libraries that need to be installed in order to be able to use the full functionality of the tool. A list of these libraries can be seen below:

setuptools>=21.0.0

numpy>=1.10.0

scipy>=0.8.0

matplotlib>=3.0.2

First you need to have installed pip on your system. Open up a terminal and type:

```
sudo apt update

sudo apt install python3-pip
```

Once the installation is complete, verify the installation by checking the pip version:

```
pip3 --version
```

The use of virtual enviroments is highly encouraged. In order to use them run the following commands in a terminal:

```
sudo apt install python3-venv
python3 -m venv .venv
source .venv/bin/activate
```

Note that as of this moment you're inside a virtual enviroment (Notice (.venv) $ in the terminal) with a limited version of python and therefore you will have to install all the packages you need (including the ones mentioned above because they are installed in the system, not the virtual enviroment).

To check the version available of a package on your system type in a terminal:

```
python3 -c "import numpy; print(numpy.version.version)"
```

and the version that is currently installed of numpy on your system will be printed.

Via pip (recommended)

Run the following command in a terminal:

```
pip3 install --extra-index-url https://test.pypi.org/simple/ fompy
```

and check the library is installed by importing it from a **python3 terminal**:

```
import fompy
```

Unless an error comes up, FoMpy is now installed on your virtual enviroment.

1.2 Quick Start

In this section the user can learn the most basic yet powerful commands implemented in the FoMpy library. In order to do so either start by reading the basic commands or download and try the exampled provided in the repository explained at the end of this page.

1.2.1 Basic commands

A bunch of useful FoMpy commands are now provided. Supported tools include fompy.extract, fompy.plot or fompy.savetotxt. Here are some quick examples of the core capabilities of FoMpy:

In order to load a FoMpy Dataset run inside a **python3 terminal**:

```
import fompy
```

FoMpy implements an importing tools that allows the user to extract the data from various sources (from a file, an array stored in memory, a Sentaurus output file, etc). Inside the folder './data/' the user has to store all simulations in individual folders (i.e. './data/sim_1/current_file_1.txt', './data/sim_2/current_file_2.txt', etc):

```
path_data = './data'
fds = fompy.dataset(path_data, parser=fompy.JCJB)
```

Note that the defined path has to point to the parent directory of the folders containing our single curve files.

After running this, a Fompy Dataset is created and the IV curves are stored inside it. They can be accessed by calling the dataset attribute:

```
print(fds.dataset)
```

Now that the Fompy Dataset has been implemented several other parameters can be defined like the number of simulations (fds.n_sims) or a value for normalizing the curves (fds.norm)., the default extraction method (fds.ext_method), the drain bias for the ensemble of curves (fds.drain_bias), the drain bias value (fds.drain_bias_value) and the default interpolation method (fds.interpolation). All these parameters can be defined/updated like the following example (Note that some of them will be defined automatically, like the number of simulations, once the IV curves are loaded):

```
fds.drain_bias_value = 0.66
```

Also a predefined function can be called in order to print the current value of the attributes of the selected Fompy Dataset:

```
fds.print_parameters()
```

The most important capability of Fompy is that it allows the user to extract the most common figures of merit (LATEX FOM) of a semiconductor's IV curve using different methodologies. In order to extract these FoM the user has to call the function extract. The following example extracts the threshold voltage values (LATEX VTH) of the curves in the Fompy Dataset:

```
vth_array = fompy.extract(fds, fom = 'vth')
```

and write the results to a file:

```
fompy.savetotxt('./results_vth.txt', 'vth', vth_array)
```

Note that since no extraction method has been defined the library uses the second derivative method ('SD') as a default. This can be changed to oterh commonly used methods like the constant current method, the third derivative or the linear extrapolation (See further instructions on how to choose this in the full documentation).

FoMpy also has built-in several plotting capabilities to be able to check the extraction results. A simple plot of the threshold voltage with the 'SD' method and the second derivative of the curve goes as follows:

```
fompy.plot(fds, fom = 'vth', save_plot='./vth_plots/sd/')
```

Note that the plots have been saved to the path './vth_plots/sd/', keeping the indexing of the curves as stored in the Fompy Dataset.

CHAPTER

TWO

IMPORTING YOUR DATA

2.1 From a file

In order to load a FoMpy Dataset the user needs to import fompy and specifically dataset tool to load the values from a file:

```
dataset(path, parser, save_to_file, interval , exclude)
```

where the arguments of the function are:

- str path: path to the parent directory containing the folders where the IV curves are.
- fn parser: function defining the format of the curves.
- str save_to_file: path where the user wants to save the FoMpy Dataset.
- [int, int] interval: list containing the start and end index of the curves to import.
- int exclude: Index of a curve that the user wishes to exclude from the Fompy Dataset.

By default only the path and the parser have to be defined. Additional functionalities are implemented, like saving all the IV curves to a single file with a header indicating the curve id. Morevover the user is able to define an interval of curves to load into the dataset, discarding the rest or define single curves from the dataset.

If additional parameters of the dataset want to be defined, i.e. a normalization value for all the curves or a default extraction method, the user can simple access these dataset attributes doing `fds.norm = 100`.

An example of how to load the curves to a FoMpy Dataset can be seen below:

```
import fompy

path_file = './data/sim_FinFET_vd_high/'

path_to_save='./sim_data.dat'

fds = fompy.dataset(path_to_file, parser='JCJB', save_to_file = path_to_save)

fds.norm = 35.8
fds.ext_method = 'SD'
fds.drain_bias = 'high'
fds.print_parameters()

>>   Number of simulations:  5
>>   Normalization value:  35.8
>>   Extraction method:  'SD'
>>   Drain Bias label:  'high'
>>   Drain Bias value:  [0.7]
```

CHAPTER

THREE

EXTRACTING THE FOM

3.1 Threshold voltage (Vth)

Generally, when performing variability studies, the parameters that characterize the $mathrm{V_{T}}$ statistical distribution (e.g. standard deviation, mean value) are used as the main criteria to assess the impact of a variability source on the device's performance in the sub-threshold region cite{TEDNATFINER}. However, a question that may arise is how independent the statistical results are from the selected $mathrm{V_{T}}$ extraction method. For this reason, in this work we present a complete comparison of four commonly used extraction techniques (SD, CC, LE and TD), with the aim of establishing the impact that $mathrm{V_{T}}$ extraction methods have on statistical variability studies. FoMpy has included four popular extraction methodologies found in the literature, a straightforward constant current, a geometrical linear extrapolation and two transconductance-based methods like the second and third derivatives.

The second derivative method, also called transconductance change method, is one of the most popular used methods. It evaluates $\mathrm{V_{TH}}$ at the $\mathrm{V_G}$ value where the derivative of the transconductance ($g_m = dI_D/dV_G$) is maximum. In Figure ?, the IV curve for the device is plotted along with the $\mathrm{V_{TH}}$ extracted value where the red dotted line crosses the x-axis and the second derivative curve with the blue dashed line. A drawback of this method is its sensitivity to noise and error, as it acts as a high pass filter on the measured data. One way of solving it, is to apply additional smoothing techniques or numerical fitting.

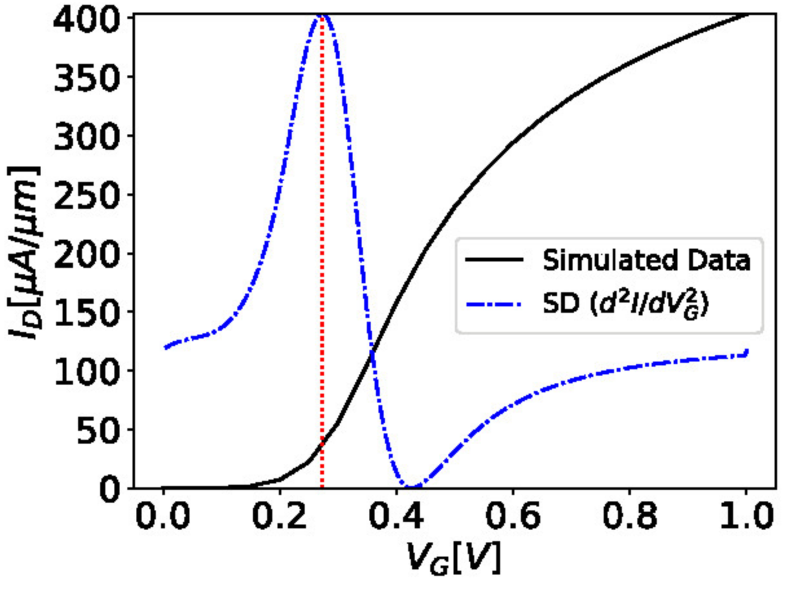

Fig. 1: Second derivative (SD) method implementation (blue dashed line). The red dotted line indicates the position of $\mathrm{V_{TH}}$.

Traditionally the constant current method has been the preferred technique used to obtain the threshold voltage value in variability studies cite{ORTIZ} cite{TEDNATFINER} cite{IEDM_vt_normal_CC} due to its simplicity and speed. This method determines $mathrm{V_{T}}$ at a critical user-defined value of the drain current ($mathrm{I_{Dcc}}$) where the transition point from linear to saturation regime happens cite{Schroder_book}. A general criteria is to consider $mathrm{I_{Dcc}}= W_{eff}/L_{G}times I_{0}$, being $W_{eff} = 2 times H_{CH}+W_{CH}$ the effective perimeter of the Si channel cite{w_effective}, and $mathrm{I_{0}}$ a constant current level cite{CC_crit_fins} defined by the user depending on the studied device. A graphical depiction of this method can be seen in Fig.~ref{fig:cc_method}.

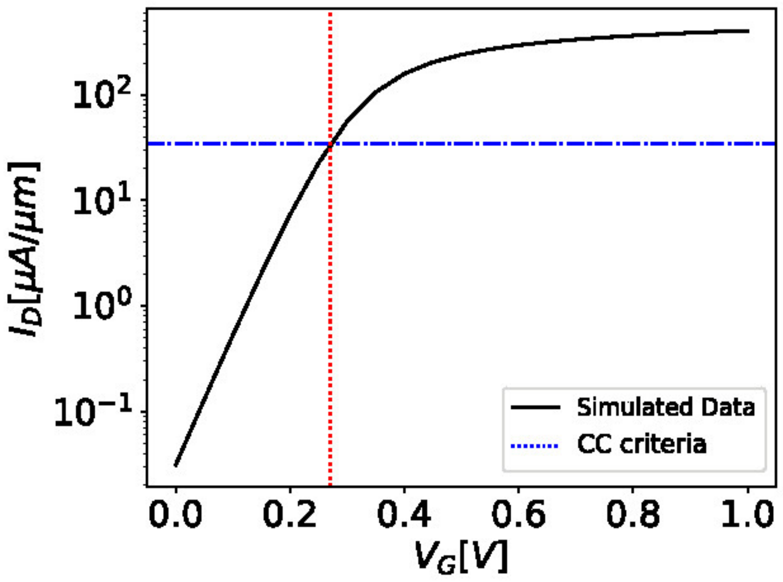

Fig. 2: Constant current (CC) criteria applied to the FinFET device with a value of $I_{Dcc}=34.6~mathrm{mu A/mu m}$ (blue dashed line).

The linear extrapolation method is another widely used technique based on the quadratic law cite{Schroder_book}. Ortiz et al. state that, using this method $mathrm{V_{T}}$ is obtained as the gate voltage axis intercept of the tangent of the IV characteristics at its maximum first derivative (slope) point cite{ORTIZ} (see Figure ref{fig:le_method}).

The third derivative method consists on choosing the $mathrm{V_{T}}$ where the third derivative of the current ($d^{3}I_{D}/dV^{3}_{G}$) has a maximum (see Figure ref{fig:td_method}). This extraction method disagrees with the SD method inherently as the values extracted from the maximums and minimums of the function always fall to the sides of the ones obtained with the SD method. Beside this, successive differentiation amplifies the noise, hence increasing the inestability, making it less reliable. To reduce this induced error fitting and smoothing techniques can be applied.

An example of how to extract the Vth from the curves in a FoMpy Dataset can be seen below:

```
import fompy

path_file = './data/sim_FinFET_vd_high/'

path_to_save='./sim_data.dat'

fds = fompy.dataset(path_to_file, parser='JCJB', save_to_file = path_to_save)

#VTH EXTRACTION

vth_array = fompy.extract(fds, fom = 'vth')
```

(continues on next page)

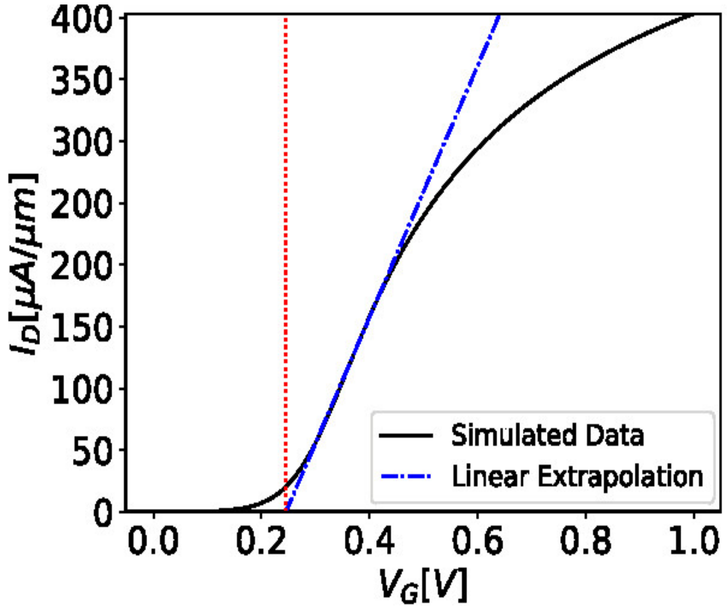

Fig. 3: Linear Extrapolation (LE) method showing the extrapolated line (blue dashed line) intersecting with the x-axis at \$V_{T}\$ (red dotted line).

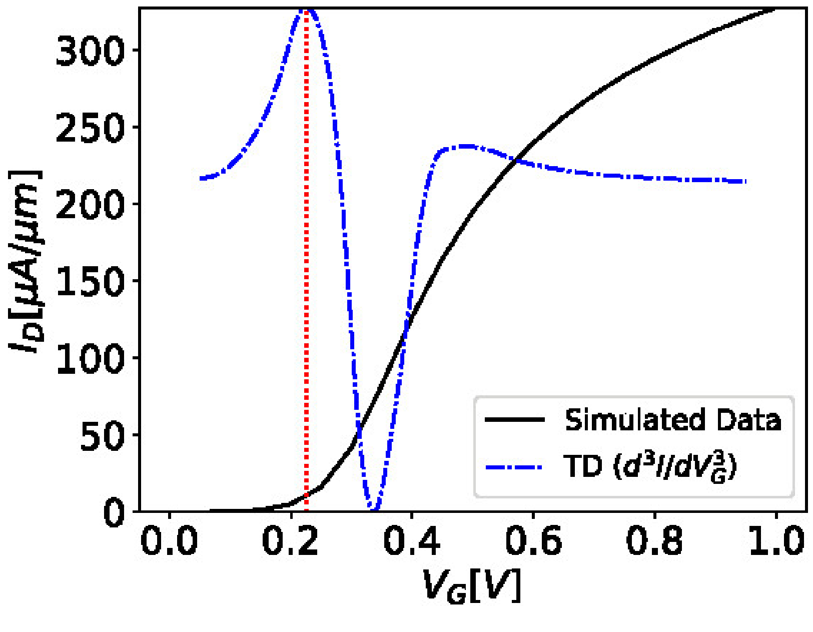

Fig. 4: Third derivative (TD) method (blue dashed line) plot showing the extracted \$mathrm{V_{T}}\$ at the \$mathrm{V_{G}}\$ point where the third derivative is maximum (red dotted line).

(continued from previous page)

```
print(vth_array)

vth_array_cc = fompy.extract(fds, fom = 'vth', method = 'CC', cc_criteria=1.5e-6)
print(vth_array_cc)

vth_array_td = fompy.extract(fds, fom = 'vth', method = 'TD')
print(vth_array_td)

vth_array_le = fompy.extract(fds, fom = 'vth', method = 'LE')
print(vth_array_le)
```

3.2 Sub-threshold Slope (SS)

An example of how to extract the SS from the curves in a FoMpy Dataset can be seen below:

```
#SS EXTRACTION

ss_array = fompy.extract(fds1 = fds, fom = 'ss')
print(ss_array)

ss_array = fompy.extract(fds1 = fds, fom = 'ss', vg_start = 0.05)
print(ss_array)

ss_array = fompy.extract(fds1 = fds, fom = 'ss', vg_start = 0.05, vg_end = 0.2)
print(ss_array)

ss_array = fompy.extract(fds1 = fds, fom = 'ss', vg_end = 0.2)
print(ss_array)
```

3.3 On-Off Current (Ion-Ioff)

An example of how to extract the Ion and Ioff from the curves in a FoMpy Dataset can be seen below:

```
#IOFF EXTRACTION

ioff_array_0 = fompy.extract(fds, fom = 'ioff')
print(ioff_array_0)
ioff_array = fompy.extract(fds, fom = 'ioff', vg_ext = 0.2)
print(ioff_array)

#ION EXTRACTION

ion_array_vg_defined = fompy.extract(fds, fom = 'ion',vg_ext = 0.7)
print(ion_array_vg_defined)

ion_array_default_SD = fompy.extract(fds, fom = 'ion')
print(ion_array_default_SD)

ion_array_LE = fompy.extract(fds, fom = 'ion',method = 'LE')
print(ion_array_LE)
```

(continues on next page)

(continued from previous page)

```
ion_array_TD = fompy.extract(fds, fom = 'ion',method = 'TD')
print(ion_array_TD)
```

3.4 DIBL

An example of how to extract the DIBL from the curves in a FoMpy Dataset can be seen below:

```
#DIBL EXTRACTION
fds_hdb = fompy.dataset(path_file_high, parser='JCJB')
fds_ldb = fompy.dataset(path_file_low, parser='JCJB')
fds_hdb._drain_bias_value = 0.7
fds_ldb._drain_bias_value = 0.05

dibl_array = fompy.extract(fds_hdb, fds_ldb, fom = 'dibl')
print(dibl_array)
```

CHAPTER

FOUR

PLOTTING FIGURES

An example of how to plot the different FoM from the curves in a FoMpy Dataset can be seen below:

```
fompy.plot(fds, fom = 'vth', save_plot='./vth_plots/sd/')

fompy.plot(fds, fom = 'vth',method = 'LE')

fompy.plot(fds, fom = 'ioff', vg_ext = 0.2)

fompy.plot(fds, fom = 'ion', vg_ext = 0.5)
fompy.plot(fds, fom = 'ion')
fompy.plot(fds, fom = 'ion', method='TD')

fompy.plot(fds, fom = 'ss') (TODO vg_ext)

fompy.plot(fds_hdb, fds_ldb, fom = 'dibl')
```

CHAPTER

FIVE

MODULES

5.1 aux.py

This module includes several auxiliar routines used during the usage of FoMpy.

Example

In order find the closest index between a given value and a array of values, the function find_closest is used. For example during the extraction of IOFF:

```
voltage_index = find_closest(x_interp, vg_ext)
parameter.append(y_interp[voltage_index])
```

In order calculate the discrete derivate of a given curve the function get_diff is used as follows (second derivative method extraction):

```
d2 = get_diff(curve, order = 2)
lower_bound=find_closest(d2[:,0],10*d2[1,0])
higher_bound=find_closest(d2[2:,0],0.8*d2[-1,0])
warning_interval_limit_low = int(np.round(lower_bound*1.5))
warning_interval_limit_high = int(np.round(higher_bound*0.8))
vt_temp = np.argmax(d2[lower_bound:higher_bound,1])
```

where these the maximum value of the second derivative is found leaving out values outside a defined interval where the derivatives can be too noisy.

Finally, the function checkPath is used mostly in FoMpy to check if a directory exists and if it does not it creates it when saving either a plot or extracted results.

`fompy.aux.`**`checkPath`**`(`*cwd*`)`

Function that checks if a given path exists, and if it doesn't, it creates it.

Parameters **`cwd`** (*`str`*) – Path that the user wishes to check if it exists.

`fompy.aux.`**`find_closest`**`(`*A*, *target*`)`

Function that finds the closest item inside an array A to a target value. It returns the index of A with the closest value to the target ($A[idx] \approx target$).

Parameters

- **`A`** (*`array_like`*) – Array containing all the values to be compared. It usually is an array of currents of a FoMpy dataset.
- **`target`** (*`float`*) – Target value to be found inside the array A.

Returns **idx** – Index of the item contained in the array A with a value closest to the target.

Return type int

`fompy.aux.`**`get_diff`**(*arr*, *order*, *type='central'*)

Function for calculating the n-order derivative of a discrete array

Parameters

- **order** (*int*) – Order indicating the type of derivative to compute
- **type** (*str*) – The type of derivate can either be calculated in a forward way, central or backward, depending on the evaluation coordinates of the derivative function (see https://en.wikipedia.org/wiki/Finite_difference)

5.2 conditioning.py

This module includes the routines used to precondition the input data before any extraction is done. It includes routines for interpolating the data points, normalizing the data and filtering points that are too noisy.

Example

In order to normalize data the following steps have to be executed either in a script or python3 terminal:

```
import fompy
path_file_JCJB = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB)
norm_value = 35.8/10**9
fompy.normalize(fds, norm_value)
```

In order to filter data the following steps have to be executed either in a script or python3 terminal:

```
import fompy
path_file_JCJB = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB)
fompy.filter(fds, theta_crit = 1.52)
```

Additionally, by default FoMpy uses the cubic spline interpolation in order to extract accurately the FoMs. If the user wishes to change the interpolation methods, there are several implemented and tested methods that can be selected. The list of available methods includes: 'akima','pchip' or 'linear'. In order to change the interpolation method the following steps have to be executed either in a script or python3 terminal:

```
import fompy
path_file_JCJB = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB)
fds.interpolation = 'linear'
```

`fompy.conditioning.`**`angle_wrt_0`**(*x*, *y*)

Function that calculates the polar angle of a point with respect to the origin of coordinates. It returns the angle in degrees and it is assummed that the angle is contained in the 1rst cuadrant.

Parameters

- **x** (*float*) – The x-axis coodinate of the point.
- **y** (*float*) – The y-axis coodinate of the point.

Returns theta – Angle between the given point and the origin of coordinates in degrees. It is assummed that the angle is contained in the 1rst cuadrant.

Return type float

class `fompy.conditioning.`**`filter_tool`**

Class for filtering noisy data from a semiconductor's IV curve.

...

`polar_filter` (*fds*, *theta_crit*)

Class method that filters data from a semiconductor's IV curve checking the increase in the angle of the points with respect to the origin of an IV curve and removes all the increments that go above defined a threshold.

Parameters

- **`fds`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor's IV curve.
- **`theta_crit`** (*`float`*) – Threshold value for the biggest allowed increase in the angle between to consecutive points of an IV curve.

class `fompy.conditioning.`**`interpolator`**

Class containing several tested interpolation strategies that can be used in a semiconductor's IV curve. For further documentation go to https://docs.scipy.org/doc/scipy/reference/interpolate.html

...

`spline_interpol` (*x*, *y*, *n*, *d*, *s*)

Interpolate data with a piecewise cubic polynomial which is twice continuously differentiable [1]. The result is represented as a PPoly instance with breakpoints matching the given data. The natural boundary condition is selected by default.

Parameters

- **`x`** (*`array_like, shape (n,)`*) – 1-d array containing values of the independent variable.
- **`y`** (*`array_like`*) – Array containing values of the dependent variable. It can have arbitrary number of dimensions, but the length along axis must match the length of x. Values must be finite.
- **`n`** (*`int`*) – Length of the output interpolated array.
- **`d`** (*`int`*) – Degree of the smoothing spline. Must be <= 5. Default is k=3, a cubic spline.
- **`s`** (*`float`*) – Positive smoothing factor used to choose the number of knots.

`akima_interpol` (*x*, *y*, *n*)

Fit piecewise cubic polynomials, given vectors x and y. The interpolation method by Akima uses a continuously differentiable sub-spline built from piecewise cubic polynomials. The resultant curve passes through the given data points and will appear smooth and natural.

Parameters

- **`x`** (*`array_like, shape (n,)`*) – 1-D array of monotonically increasing real values.
- **`y`** (*`array_like`*) – N-D array of real values. The length of y along the first axis must be equal to the length of x.
- **`n`** (*`int`*) – Length of the output interpolated array.

pchip_interpol (*x*, *y*, *n*)

Convenience function for pchip interpolation. xi and yi are arrays of values used to approximate some function f, with yi = f(xi). The interpolant uses monotonic cubic splines to find the value of new points x and the derivatives there.

Parameters

- **x** (*array_like, shape (n,)*) – A 1-D array of monotonically increasing real values. x cannot include duplicate values (otherwise f is overspecified)
- **y** (*array_like*) – A 1-D array of real values. y's length along the interpolation axis must be equal to the length of x. If N-D array, use axis parameter to select correct axis.
- **n** (*int*) – Length of the output interpolated array.

lin_interpol (*x*, *y*, *n*)

Interpolate a 1-D function.x and y are arrays of values used to approximate some function f: y = f(x). This class returns a function whose call method uses interpolation to find the value of new points.

Parameters

- **x** (*array_like, shape (n,)*) – A 1-D array of real values.
- **y** (*array_like*) – A N-D array of real values. The length of y along the interpolation axis must be equal to the length of x.
- **n** (*int*) – Length of the output interpolated array.

class fompy.conditioning.**normalizer**

Normalizer class.

...

normalize (*fds*, *norm*)

Class method used for normalizing the currents a semiconductor's IV curve.

Parameters

- **fds** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve.
- **norm** (*float*) – float value used to normalize the currents contained in the FoMpy dataset IV curves.

5.3 fds.py

This module includes the routines used to import data into a FoMpy Dataset. Some useful examples showing how to import the data using these functions can be seen below:

Example

If the parser is not defined, the default parser is used, where a simple voltage-current file is assumed to be defined as input:

```
import fompy
path_file = './data/default/'
fds = fompy.dataset(path_file)
```

or:

```
import fompy
path_file = './data/default/'
fds = fompy.dataset(path_file,'data*',  parser=fompy.default)
```

Moreover, another types of parsers can be selected:

```
import fompy
path_file_JCJB = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB)

path_file_mc = './data/mc_data/'
fds = fompy.dataset(path_file_mc, parser=fompy.MC)

(TODO)ADD ATLAS
```

If the user wishes to remove several IV curves included in the parent folder, two options called 'interval' and 'exclude' can be passed, so the indexes defined are removed from the FoMpy Dataset:

```
import fompy
path_file_JCJB = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB, exclude=[5,6])
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB, interval=[0,8])
```

`fompy.fds.`**`ATLAS`**(*fds*)

(TODO)

`class` `fompy.fds.`**`FompyDataset`**(***kwargs*)

Class containing the simulated IV curves and their parameters.

. . .

`dataset`

List containing all IV curves

Type array: double[]

`n_sims`

Number of simulated IV curves

Type int

`sanity_array`

Array of ones by default. If a simulation has failed, its index is converted to zero.

Type array: double[]

`norm`

Normalization value applied to the IV curve

Type double

`ext_method`

FompyDataset default method used to extrad the figures of merit

Type str

`drain_bias_label`
Either high or low.

Type str

`drain_bias_value`
Drain voltage value used to simulate the IV curves.

Type double

`interpolation`
IV curve interpolation method. Can either be ‘cubic_spline’, ‘akima’,’pchip’ or ‘linear’.

Type str

`filter`
IV curve filter method. So far only the polar filter has been implemented (‘polar_filter’).

Type str

`print_parameters``()`
Prints all the non-array attributes.

`fompy.fds.`**`JCJB`**`(`*fds*, *path*, *path_subdirs*, *path_filenames*, *interval*, *exclude*`)`
Function that imports the simulated data from a JCJB file and stores it into a FoMpy Dataset.

Parameters

- **`fds`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor’s IV curve.
- **`path`** (*`str`*) – Parent path where the simulations are stored
- **`path_subdirs`** (*`str`*) – List of alphabetically sorted subdirectories found inside the parent directory.
- **`path_filenames`** (*`str`*) – List of alphabetically sorted files found inside the parent directory.
- **`interval`** (*`array_like`*) – List of two int values: start(index of the first simulation to load into the Fompy Dataset) and end(index of the last simulation to load into the Fompy-Dataset)
- **`exclude`** (*`array_like`*) – Index values of simulations to exclude.

Returns **fds** – Structure of data containing the FoMpy Dataset.

Return type FoMpy Dataset

`fompy.fds.`**`MC`**`(`*fds*, *path*, *filename_user*, *path_subdirs*, *path_filenames*, *interval*, *exclude*, *skiprows*, *comments*`)`
Function that imports the simulated data from a MC set of files and stores it into a FoMpy Dataset.

Parameters

- **`fds`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor’s IV curve.
- **`path`** (*`str`*) – Parent path where the simulations are stored
- **`path_subdirs`** (*`str`*) – List of alphabetically sorted subdirectories found inside the parent directory.
- **`path_filenames`** (*`str`*) – List of alphabetically sorted files found inside the parent directory.

- **`interval`** (*`array_like`*) – List of two int values: start(index of the first simulation to load into the Fompy Dataset) and end(index of the last simulation to load into the Fompy-Dataset)
- **`exclude`** (*`array_like`*) – Index values of simulations to exclude.
- **`skiprows`** (*`int`*) – Number of rows to skip at the begining of a file. 0 rows are skipped by default.
- **`comments`** (*`str`*) – All the lines starting with this character are considered comments. '#' is used by default.

Returns fds – Structure of data containing the FoMpy Dataset.

Return type FoMpy Dataset

class `fompy.fds.`**`daoArray`**(***kwargs*)

(TODO)

class `fompy.fds.`**`daoFile`**(*parser=None*)

Data Acces Object used to import from a set of files the simulated IV curves to a FompyDataset.

...

`parser`

Type of parser, user-defined, that imports the simulated IV curves in a specific format.

Type str

`load`(*path*, *parser=None*, *interval=None*, *exclude=None*, *skiprows=None*, *comments=None*)

`load`(*path*, *filename_user = None*, *parser = None*, *interval = None*, *exclude = None*, *skiprows = None*, *comments = None*)

Class method that extracts V_{TH} of a semiconductor's IV curve.

Parameters

- **`path`** (*`array_like, shape (n,)`*) – 1-d array containing values of the independent variable.
- **`parser`** (*`function`*) – Function that implements how the data is imported to a Fompy Dataset. The list of available functions includes: (TODO)'default', 'JCJB', 'MC' and 'ATLAS'.
- **`interval`** (*`array_like`*) – List of two int values: start(index of the first simulation to load into the Fompy Dataset) and end(index of the last simulation to load into the Fompy-Dataset)
- **`exclude`** (*`array_like`*) – Index values of simulations to exclude.
- **`skiprows`** (*`int`*) – Number of rows to skip at the begining of a file. 0 rows are skipped by default.
- **`comments`** (*`str`*) – All the lines starting with this character are considered comments. '#' is used by default.

class `fompy.fds.`**`dataDAO`**

Data Acces Object Interface used to import the simulated IV curves to a FompyDataset.

`fompy.fds.`**`default`**(*fds*, *path*, *path_subdirs*, *path_filenames*, *interval*, *exclude*, *skiprows*, *comments*)

(TODO)Function that imports the simulated data from a given file and stores it into a FoMpy Dataset. The format of this file is asummed to have comments starting with '#', without a header and two columns, one for the voltage and one for the currents, separated by ' ' delimiters.

Parameters

- **fds** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve.
- **path** (*str*) – Parent path where the simulations are stored
- **path_subdirs** (*str*) – List of alphabetically sorted subdirectories found inside the parent directory.
- **path_filenames** (*str*) – List of alphabetically sorted files found inside the parent directory.
- **interval** (*array_like*) – List of two int values: start(index of the first simulation to load into the Fompy Dataset) and end(index of the last simulation to load into the Fompy-Dataset)
- **exclude** (*array_like*) – Index values of simulations to exclude.
- **skiprows** (*int*) – Number of rows to skip at the begining of a file. 0 rows are skipped by default.
- **comments** (*str*) – All the lines starting with this character are considered comments. '#' is used by default.

Returns **fds** – Structure of data containing the FoMpy Dataset.

Return type FoMpy Dataset

`fompy.fds.`**`exclude_indexes`**(*fds*, *interval*, *exclude*)

Function that generates an array of zeros and ones. If the simulation number 5 has failed, either because the folder is empty, or not enough voltages have been simulated, then fds.sanity_array[4] is set to zero.

Parameters

- **fds** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve.
- **interval** (*array_like*) – List of two int values: start(index of the first simulation to load into the Fompy Dataset) and end(index of the last simulation to load into the Fompy-Dataset)
- **exclude** (*array_like*) – Index values of simulations to exclude.

Returns **fds** – Structure of data containing the sanity array.

Return type FoMpy Dataset

5.4 fom.py

This module includes the routines used to extract several commonly used FoMs in semiconductor simulations.

Example

After the data is imported into a FoMpy Dataset, the user may use several routines implemented in FoMpy to extract the 'vth', 'ioff', 'ion', 'ss', 'ratio', 'power' or 'dibl'. Code examples explaining how to use them can be seen below.

In order to extrac the vth of a given IV curve, the following commands have to be used either in a script or in a python3 command line:

```
import fompy
path_file_high = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB)
vth_array = fompy.extract(fds, fom = 'vth')
```

If a different FoM is needed, the user has to change the keyword fom, from ‘vth’ to another one contained in the list shown before.

Additionally, if the user wants to obtain the DIBL, two different FoMpy Datasets have to be imported:

```
import fompy
path_file_JCJB = './data/sim_FinFET_vd_high/'
path_file_low = './data/sim_FinFET_vd_low/'

fds_hdb = fompy.dataset(path_file_JCJB, parser='JCJB')
fds_ldb = fompy.dataset(path_file_low, parser='JCJB')
fds_hdb.drain_bias_value = 0.7
fds_ldb.drain_bias_value = 0.05
dibl_array = fompy.extract(fds_hdb, fds_ldb, fom = 'dibl')
```

class `fompy.fom.`**`dibl_ext`**

`extraction` (*fds1*, *fds2*, *method=None*)

`extraction` (*fds1*, *fds2*, *method = None*)

Class method that extracts SS of a semiconductor’s IV curve.

Parameters

- **`fds1`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor’s IV curve. Needed for the extraction of any FoM.
- **`fds2`** (*`FoMpy Dataset`*) – additional structure of data containing the most important parameters of a semiconductor’s IV curve. Needed for generating the plot of the calibration and the DIBL.
- **`method`** (*`str`*) – Keyword indicating the desired method of extraction of the FoMs. The list of available methods includes: ‘SD’, ‘CC’, ‘TD’ and ‘LE’. If method is not defined the ‘SD’ is selected by default.

`plot` (*fds1*, *parameter = None*, *method = None*, *cc_crit = None*, *curves = None*, *save = None*, *A=None*, *B=None*)

Class method that plots the extracted V_{TH} values.

Parameters

- **`fds1`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor’s IV curve. Needed for generating the plot of any FoM.
- **`curve_high`** (*`array_like`*) – Array of data containing the IV curves at high drain bias.
- **`curve_low`** (*`array_like`*) – Array of data containing the IV curves at low drain bias.
- **`parameter_vt_high`** (*`float`*) – Voltage value of high drain bias.
- **`parameter_vt_low`** (*`float`*) – Voltage value of low drain bias.

- **`corriente_low`** (*`float`*) – Current at the vth value extracted for the curve at low drain bias.
- **`save_plot`** (*`bool`*) – If True the generated plot is save to the defined path.

`save_results_to_file` (*path*, *parameter*)
Class method that saves the extracted V_{TH} values.

Parameters

- **`path`** (*`str`*) – Defines the path where the extracted results are saved to a file.
- **`parameter`** (*`array_like`*) – Array of extracted FoM values to be saved into the file.

`fompy.fom.`**`interpol`** (*x=None*, *y=None*, *n=None*, *strategy=None*, *d=None*, *s=None*)
Wrapper function for interpolating imported data from a semiconductor's IV curve.

Parameters

- **`x`** (*`array_like, shape (n,)`*) – 1-d array containing values of the independent variable.
- **`y`** (*`array_like`*) – Array containing values of the dependent variable. It can have arbitrary number of dimensions, but the length along axis must match the length of x. Values must be finite.
- **`strategy`** (*`str`*) – Keyword for defining the selected interpolation method: The list of available methods includes: 'akima', 'pchip' and 'linear'.
- **`d`** (*`int`*) – Degree of the smoothing spline. Must be <= 5. Default is k=3, a cubic spline.
- **`s`** (*`float`*) – Positive smoothing factor used to choose the number of knots.

`class` `fompy.fom.`**`ioff_ext`**
Child class of _extractor that obtains the I_{OFF} figure of merit from a semiconductor's IV curve.

`extraction` (*fds1=None*, *vg_ext=None*)

`extraction` (*fds1*, *method=None*, *cc_criteria = None*)
Class method that extracts I_{OFF} of a semiconductor's IV curve.

Parameters

- **`fds1`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for the extraction of any FoM.
- **`vg_ext`** (*`float`*) – Gate voltage value used to calculate IOFF at.

`plot` (*fds1*, *parameter = None*, *method = None*, *cc_crit = None*, *curves = None*, *save = None*, *A=None*, *B=None*)
Class method that plots the extracted V_{TH} values.

Parameters

- **`fds1`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for generating the plot of any FoM.
- **`parameter`** (*`array_like`*) – Array of extracted FoM values to be plotted.
- **`vg_ext`** (*`float`*) – Gate voltage value used to calculate IOFF at.
- **`curves`** (*`array_like`*) – Array of data containing the IV curves.
- **`save_plot`** (*`bool`*) – If True the generated plot is save to the defined path.

save_results_to_file (*path*, *parameter*)
Class method that saves the extracted V_{TH} values.

Parameters

- **path** (*str*) – Defines the path where the extracted results are saved to a file.
- **parameter** (*array_like*) – Array of extracted FoM values to be saved into the file.

class fompy.fom.**ion_ext**
Child class of _extractor that obtains the I_{ON} figure of merit from a semiconductor's IV curve.

extraction (*fds1*, *vg_ext=None*, *vth=None*)

extraction (*fds1*, *vg_ext = None*, *vth = None*)
Class method that extracts I_{ON} of a semiconductor's IV curve.

Parameters

- **fds1** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for the extraction of any FoM.
- **vg_ext** (*float*) – Gate voltage value used to calculate IOFF at.
- **vth** (*array_like*) – Array of vth extracted values used for obtaining ION (using the vth-dependant formula).

plot (*fds1*, *parameter = None*, *method = None*, *cc_crit = None*, *curves = None*, *save = None*, *A=None*, *B=None*)
Class method that plots the extracted V_{TH} values.

Parameters

- **fds1** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for generating the plot of any FoM.
- **parameter** (*array_like*) – Array of extracted FoM values to be plotted.
- **curves** (*array_like*) – Array of data containing the IV curves.
- **parameter_vth** (*array_like*) – Array of extracted vth values, as a method to obtain ION depends on them.
- **vg_ext** (*float*) – Gate voltage value used to calculate IOFF at.
- **save_plot** (*bool*) – If True the generated plot is save to the defined path.

save_results_to_file (*path*, *parameter*)
Class method that saves the extracted V_{TH} values.

Parameters

- **path** (*str*) – Defines the path where the extracted results are saved to a file.
- **parameter** (*array_like*) – Array of extracted FoM values to be saved into the file.

class fompy.fom.**ss_ext**

extraction (*fds1*, *vg_start=None*, *vg_end=None*)

extraction (*fds1*, *vg_start = None*, *vg_end = None*)
Class method that extracts SS of a semiconductor's IV curve.

Parameters

- **fds1** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for the extraction of any FoM.
- **vg_start** (*float*) – Gate voltage defining the start of the interval in which the Subthreshold Swing is extracted.
- **vg_end** (*float*) – Gate voltage defining the end of the interval in which the Subthreshold Swing is extracted.

plot (*fds1*, *parameter = None*, *method = None*, *cc_crit = None*, *curves = None*, *save = None*, *A=None*, *B=None*)
Class method that plots the extracted V_{TH} values.

Parameters

- **fds1** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for generating the plot of any FoM.
- **parameter** (*array_like*) – Array of extracted FoM values to be plotted.
- **curves** (*array_like*) – Array of data containing the IV curves.
- **parameter_ss** (*array_like*) – Array of extracted ss values.
- **vg_start** (*float*) – Gate voltage defining the start of the interval in which the Subthreshold Swing is extracted.
- **vg_end** (*float*) – Gate voltage defining the end of the interval in which the Subthreshold Swing is extracted.
- **vg_sd_medio** (*float*) – Value in the middle of the interval between zero and vth extracted with the SD method. It is used only for defining a limit in the plot.
- **save_plot** (*bool*) – If True the generated plot is save to the defined path.

save_results_to_file (*path*, *parameter*)
Class method that saves the extracted V_{TH} values.

Parameters

- **path** (*str*) – Defines the path where the extracted results are saved to a file.
- **parameter** (*array_like*) – Array of extracted FoM values to be saved into the file.

class fompy.fom.**vth_ext**
Child class of _extractor that obtains the V_{TH} figure of merit from a semiconductor's IV curve.

extraction (*fds1*, *method=None*, *cc_criteria=None*)

extraction (*fds1*, *method=None*, *cc_criteria = None*)
Class method that extracts V_{TH} of a semiconductor's IV curve.

Parameters

- **fds1** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for the extraction of any FoM.
- **method** (*str*) – Keyword indicating the desired method of extraction of the FoMs.The list of available methods includes: 'SD', 'CC', 'TD' and 'LE'. If method is not defined the 'SD' is selected by default.

- **`cc_criteria`** (*`float, optional`*) – Float value used for the extraction of several FoMs using the constant current method.

`plot` (*fds1*, *parameter = None*, *method = None*, *cc_crit = None*, *curves = None*, *save = None*, *A=None*, *B=None*)
Class method that plots the extracted V_{TH} values.

Parameters

- **`fds1`** (*`FoMpy Dataset`*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for generating the plot of any FoM.
- **`parameter`** (*`array_like`*) – Array of extracted FoM values to be plotted.
- **`method`** (*`str`*) – Keyword indicating the desired method of extraction of the FoMs. The list of available methods includes: 'SD', 'CC', 'TD' and 'LE'. If method is not defined the 'SD' is selected by default.
- **`cc_criteria`** (*`float`*) – Current criteria used to extract vth with the CC criteria for the fomplot.
- **`curves`** (*`array_like`*) – Array of data containing the IV curves.
- **`save_plot`** (*`bool`*) – If True the generated plot is save to the defined path.
- **`B`** (*`A,`*) – Parameters obtained during the vth LE extraction method used for the plots.

`save_results_to_file` (*path*, *parameter*)
Class method that saves the extracted V_{TH} values.

Parameters

- **`path`** (*`str`*) – Defines the path where the extracted results are saved to a file.
- **`parameter`** (*`array_like`*) – Array of extracted FoM values to be saved into the file.

5.5 plots.py

This module includes the routines used to generate several common figures in semiconductor simulations.

Example

After the FoMs have been extracted, FoMpy includes several useful visualizing routines, commonly used in semiconductor simulations. Code examples explaining how to use them can be seen below.

In order to generate a plot of the FoMs (fomplot):

```
import fompy
path_file_high = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB)
vth_array = fompy.extract(fds, fom = 'vth')
fompy.plot(fds, fom = 'vth')
```

and FoMpy will generate a single graph for each simulation showing the IV curve with its correspondent FoM extraction criteria and the extracted FoM.

Most of the FoMs are plotted the same way, except for the DIBL. This is because, if we want to calculate the DIBL two FoMpy Datasets are needed:

```
import fompy
path_file_JCJB = './data/sim_FinFET_vd_high/'
path_file_low = './data/sim_FinFET_vd_low/'

fds_hdb = fompy.dataset(path_file_JCJB, parser='JCJB')
fds_ldb = fompy.dataset(path_file_low, parser='JCJB')
fds_hdb.drain_bias_value = 0.7
fds_ldb.drain_bias_value = 0.05

fompy.plot(fds_hdb, fds_ldb, fom = 'dibl')
```

Additionally, other types of plots can be generated. The list of available plots includes: ‘iv’, ‘hist’, ‘qq’, ‘varplot’, ‘calib’ and ‘fomplot’. These are some examples:

```
import fompy
path_file_high = './data/sim_FinFET_vd_high/'
fds = fompy.dataset(path_file_JCJB, parser=fompy.JCJB)
vth_array = fompy.extract(fds, fom = 'vth')

fompy.plot(fds, type='hist', parameter=vth_array, bins=5)

fompy.plot(fds, type='qq', parameter=vth_array)

fompy.plot(fds, type='varplot')

path_file_JCJB = './data/sim_FinFET_vd_high/'
path_file_low = './data/sim_FinFET_vd_low/'

fds_hdb = fompy.dataset(path_file_JCJB, parser='JCJB')
fds_ldb = fompy.dataset(path_file_low, parser='JCJB')

(TODO) fompy.calib(fds_hdb, fds_ldb)
```

class `fompy.plots.`**`plotStrategy`**

Abstract class containing several commonly used plots for semiconductor simulations.

class `fompy.plots.`**`plotter`**

Class used for generating common plots in semiconductor simulations. For extensive documentation on how to modify this code go to https://matplotlib.org/tutorials/index.html

`fomplot`(*i*, *fom=None*, *currents=None*, *voltages=None*, *parameter=None*, *method=None*, *cc_criteria=None*, *parameter_vth=None*, *vg_ext=None*, *curve_high=None*, *curve_low=None*, *vth_high=None*, *vth_low=None*, *corriente_low=None*, *save_plot=None*, *A=None*, *B=None*, *vg_start=None*, *vg_end=None*, *vt_sd_medio=None*)

fomplot(i, fom = None, voltages = None, currents = None, parameter = None,method =

vg_ext = None, curve_high = None, curve_low = None, vth_high = None, vth_low = None

Plot the most common figures of merit of a semiconductor’s IV curve.

Parameters

- **fom** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor’s IV curve.
- **currents** (*path or None, optional*) – Path indicating the folder where the user wishes to save the generated plots.

- **voltages** (*path or None, optional*) – Path indicating the folder where the user wishes to save the generated plots.

hist (*bins=None*, *parameter=None*, *save_plot=None*)

hist(bins = None, parameter = None, save_to_file = None):
Plot a histogram.The return value is a tuple (n, bins, patches) or ([n0, n1, ...], bins, [patches0, patches1,...]) if the input contains multiple data.

Parameters

- **bins** (*int*) – If an integer is given, bins + 1 bin edges are calculated and returned, consistent with numpy.histogram
- **parameter** (*array_like, shape (n,)*) – Input values, this takes either a single array or a sequence of arrays which are not required to be of the same length.
- **save_plot** (*path or None, optional*) – Path indicating the folder where the user wishes to save the generated plots.

iv (*fds*, *save_plot=None*)

iv (*fds*, *save_plot = None*)
Class method that filters data from a semiconductor's IV curve using Gaussian filtering.

Parameters

- **fds** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve.
- **save_plot** (*path or None, optional*) – Path indicating the folder where the user wishes to save the generated plots.

qq (*parameter*, *save_plot=None*)

qq(parameter, save_to_file = None):
Plot a Quantile plot.Plotting positions are converted into quantiles or Z-scores based on a probability distribution

Parameters

- **parameter** (*array_like, shape (n,)*) – Input values, this takes either a single array or a sequence of arrays which are not required to be of the same length.
- **save_plot** (*path or None, optional*) – Path indicating the folder where the user wishes to save the generated plots.

varplot (*fds*, *save_plot=None*)

varplot(self,fds, save_plot = None):
Plot all the IV curves for a common variability source.

Parameters

- **fds** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve.

- **save_plot** (*path or None, optional*) – Path indicating the folder where the user wishes to save the generated plots.

5.6 wrappers.py

This module includes all the wrapper functions used in the FoMpy library. Code examples on how to use them can be seen either in the complete guide, the repository quickstart or at the beginning of each file of the source code.

`fompy.wrappers.`**`dataset`**(*path*, *filename_user=None*, *parser=None*, *save_to_file=None*, *interval=None*, *exclude=None*, *skiprows=None*, *comments=None*)

Wrapper function that creates a FoMpy dataset.

Parameters

- **globstr** (*str*) – Path to the file containing the IV curves
- **parser** (*void*) – Format to read the file
- **save_to_file** (*str*) – Path of the file to store all the FoMpy dataset
- **interval** (*array_like*) – List of two int values: start(index of the first simulation to load into the FompyDataset) and end(index of the last simulation to load into the Fompy-Dataset)
- **exclude** (*array_like*) – Index values of simulations to exclude.
- **skiprows** (*int*) – Number of rows to skip at the begining of a file. 0 rows are skipped by default.
- **comments** (*str*) – All the lines starting with this character are considered comments. '#' is used by default.

Returns Class containing the most important parameters of a semiconductor IV curve

Return type *FompyDataset*

`fompy.wrappers.`**`extract`**(*fds1*, *fds2=None*, *fom=None*, *method=None*, *cc_criteria=None*, *vg_ext=None*, *print_fom=None*, *vg_start=None*, *vg_end=None*)

Wrapper function that extracts the most common figures of merit from a FoMpy dataset.

Parameters

- **fds1** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for the extraction of any FoM.
- **fds2** (*FoMpy Dataset*) – additional structure of data containing the most important parameters of a semiconductor's IV curve. Needed for the extraction of the DIBL.
- **fom** (*str*) – Keyword indicating the desired FoMs to extract. The list of available FoMs includes: 'vth', 'ioff', 'ion', 'ss', 'ratio', 'power' and 'dibl'.
- **method** (*str*) – Keyword indicating the desired method of extraction of the FoMs.The list of available methods includes: 'SD', 'CC', 'TD' and 'LE'. If method is not defined the 'SD' is selected by default.
- **cc_criteria** (*float, optional*) – Float value used for the extraction of several FoMs using the constant current method.
- **vg_ext** (*float*) – Gate voltage value used to calculate a FoM like IOFF or ION.
- **print_fom** (*bool*) – If True all the FoMs are extracted from a FoMpy Dataset (except for the DIBL)

- **vg_start** (*float*) – Gate voltage defining the start of the interval in which the Subthreshold Swing is extracted.
- **vg_end** (*float*) – Gate voltage defining the end of the interval in which the Subthreshold Swing is extracted.

Returns **parameter** – 1-d array containing the extracted FoMs.

Return type array_like

`fompy.wrappers.`**`filter`**(*fds*, *theta_crit*)

Wrapper class for filtering noisy data from a semiconductor's IV curve.

Parameters

- **fds** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve.
- **theta_crit** (*float*) – Threshold value for the biggest allowed increase in the angle between to consecutive points of an IV curve.

`fompy.wrappers.`**`normalize`**(*fds*, *norm*)

Wrapper function that normalizes the currents a semiconductor's IV curve.

Parameters

- **fds** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve.
- **norm** (*float*) – float value used to normalize the currents contained in the FoMpy dataset IV curves.

`fompy.wrappers.`**`plot`**(*fds1*, *fds2=None*, *type=None*, *fom=None*, *parameter=None*, *method=None*, *bins=None*, *cc_criteria=None*, *vg_ext=None*, *vg_start=None*, *vg_end=None*, *save_plot=None*)

Wrapper function that plots the most common figures in semiconductor simulations.

Parameters

- **fds1** (*FoMpy Dataset*) – Structure of data containing the most important parameters of a semiconductor's IV curve. Needed for generating the plot of any FoM.
- **fds2** (*FoMpy Dataset*) – additional structure of data containing the most important parameters of a semiconductor's IV curve. Needed for generating the plot of the calibration and the DIBL.
- **type** (*str*) – Keyword indicating the type of plot to generate. The list of available plots includes: 'iv', 'hist', 'qq', 'varplot', 'calib' and 'fomplot'.
- **fom** (*str*) – Keyword indicating the FoM to be plotted. The list of available methods includes: 'vth', 'ioff', 'ion', 'ss', 'ratio', 'power' and 'dibl'.
- **parameter** (*array_like*) – Array of extracted FoM values to be plotted.
- **method** (*str*) – Keyword indicating the desired method of extraction of the FoMs. The list of available methods includes: 'SD', 'CC', 'TD' and 'LE'. If method is not defined the 'SD' is selected by default.
- **bins** (*int*) – It defines the number of equal-width bins in the given range.
- **cc_criteria** (*float*) – Current criteria used to extract vth with the CC criteria for the fomplot.
- **vg_ext** (*float*) – Gate voltage value used to calculate a FoM like IOFF or ION.

- **`vg_start`** (*`float`*) – Gate voltage defining the start of the interval in which the Subthreshold Swing is extracted.
- **`vg_end`** (*`float`*) – Gate voltage defining the end of the interval in which the Subthreshold Swing is extracted.
- **`save_plot`** (*`bool`*) – If True the generated plot is save to the defined path.

`fompy.wrappers.`**`savetotxt`**(*path*, *fom*, *parameter*)

Wrapper function that saves to a text file the extracted FoMs.

Parameters

- **`path`** (*`str`*) – Defines the path where the extracted results are saved to a file.
- **`fom`** (*`str`*) – Keyword indicating the FoM to be plotted. The list of available methods includes: 'vth', 'ioff', 'ion', 'ss', 'ratio', 'power' and 'dibl'.
- **`parameter`** (*`array_like`*) – Array of extracted FoM values to be plotted.

`fompy.wrappers.`**`version`**()

(TODO)Function that prints the current installed version of FoMpy

PYTHON MODULE INDEX

INDEX