iFlow Quick User Guide

Yoeri Dijkstra & Ronald Brouwer

July 3, 2017

This user guide provides a short introduction to iFlow using the Scheldt River tutorial.

1 Installation

The iFlow modelling framework is written in Python, which is a free, open source programming language. In order to run iFlow you need to install Python 2.7 and a number of additional packages that enhance Python's functionality. We recommend to use one of the many freely available a pre-built Python bundles, which combine Python with the most commonly used packages. Though many of these should work, we have experience with the following:

Python bundle	Notes
Anaconda 2.4.1	Recommended bundle
Anaconda 4.0..	Newer version than Anaconda 2.4.1. The code has not yet been extensively tested with the newer matplotlib (v2) included in this bundle, but should work with this.
Canopy 1.5.2	Similar to Anaconda, but uses a different version of the plotting packages, which is not compatible with the standard plotting tools in iFlow.

These packages have been tested on all platforms.

iFlow can be run using either the command line and any code editing program or any IDE. An IDE provides an integrated environment for code development, running and debugging and is therefore recommended. We recommend using PyCharm, which is a free IDE especially developed for Python programs.

2 Starting iFlow

To use iFlow from the command line[1], navigate to the iFlow directory by typing `cd` followed by the path to the iFlow main directory and press enter. If this is on a different disk than C:, e.g. D:, type `D:` and press enter to move to the disk first. Type `python main.py` to run iFlow. To run iFlow from an IDE, please run the script `main.py` in the iFlow main directory.

The iFlow main menu should appear on the screen. The main menu should look like the text below

[1] On a Windows system, in the start menu, type `cmd` and press enter to start the command line.

```
iFlow version 2.4

No working directory set. Now working from the iFlow root directory.
Enter cwd to change the working directory.

Please enter the path to a new input file:
```

iFlow works using a main directory for its source files and a working directory for specific projects. One can use a working directory for input and output files and own pieces of code (e.g. for plotting results). By default, the working directory is the same as the main directory. It is recommended to make a different separate directory for each of your iFlow projects. Here we will go to the directory with the Scheldt River tutorial. To change the working directory type `cwd` in the menu and press enter. Next type or copy the full absolute path to the Scheldt River tutorial folder (folder named `GettingStartedScheldt`) and press enter.

To run a simulation, enter the path to an iFlow input file. Type `input/Scheldt_base.txt` and press enter. iFlow then runs the input file and shows its progress on the screen. This may look like the text below.

```
Call stack was built successfully
** Call stack **
1 analytical2DV.Geometry2DV
2 numerical2DV.RegularGrid
3 analytical2DV.turbulence.TurbulenceUniform
4 semi_analytical2DV.hydro.HydroLead tide, river
5 analytical2DV.salinity_diagnostic.SaltHyperbolicTangent
6 semi_analytical2DV.hydro.HydroFirst adv, stokes, nostress, baroc, tide, river
7 semi_analytical2DV.sediment.SedDynamic erosion, sedadv, noflux
8 plot.Plot_base_model

** Run **
Running turbulence model Uniform
Running module HydroLead
Running module SaltHyperbolicTangent
Running module HydroFirst
Running module Sediment
Plotting
```

Under `** Call stack **` iFlow lists the modules that it will use in the simulation. Modules are separate components of the model each with their specific task. The above model will make the geometry of the estuary, build a grid and determine the turbulence related parameters. Next it computes the hydrodynamics at leading order, sets the salinity profile, computes the first order hydrodynamics and the sediment dynamics. Finally, the last module shows plots of the result. Under `** Run **` some of the modules show that they are running or show their progress otherwise.

The next time that iFlow is started, the working directory is set to the last used working directory. Also, a list with the five most recently used input files will be shown in the starting menu. This will look like

```
iFlow version 2.4

Current working directory set to ..../GettingStartedScheldt.
Enter cwd to change the working directory.

Please choose an input file from the list of recent files:
1 input/Scheldt_base.txt
or enter the path to a new input file:
```

To run one of the input files in the list, type the corresponding number (here 1) and press enter. Alternatively, to use the last input file, simply press enter without entering the number.

3 The input file

iFlow input files are simple text files that list the modules (i.e. model components) to be used, the variables to be calculated and the input variables. Open an input file in any code editor to see or change it. On Windows, Notepad suffices. Please do not use text editors such as MS Word to change the input file. An input file may look something like the text below.

```
# Input file for the Scheldt estuary
#
# Date: 03-07-17
# Authors: Y.M. Dijkstra, R.L. Brouwer


## Geometry ##
module  analytical2DV.Geometry2DV
L       160000
B0      type    functions.ExpRationalFunc
        C1      -0.02742e-3 1.8973
        C2      4.9788e-11 -9.213e-6 1
H0      type    functions.Polynomial
        C -2.9013e-24 1.4030e-18 -2.4218e-13 1.7490e-8 -5.2141e-4 15.332

## Grid ##
module     numerical2DV.RegularGrid
xgrid      equidistant     200
zgrid      equidistant     100
fgrid      integer         2

xoutputgrid    equidistant     100
zoutputgrid    equidistant     50
foutputgrid    integer         2

## Hydrodynamics ##
module      semi_analytical2DV.HydroLead semi_analytical2DV.HydroFirst
#module      numerical2DV.HydroLead numerical2DV.HydroFirst
```

```
submodules  river tide baroc adv nostress stokes
A0          0 1.77 0
A1          0 0 0.14
phase0      0 0 0
phase1      0 0 -1.3
Q0          0
Q1          80

## Turbulence ##
module analytical2DV.TurbulenceUniform
Av0amp      0.0367
Av0phase    0.
sf0         0.0048
m           1.
n           0.

## Salinity ##
module      analytical2DV.SaltHyperbolicTangent
#mean conditions Scheldt
ssea        30.
xc          55.e3
xl          26.e3

## Sediment ##
module      semi_analytical2DV.SedDynamic
submodules  erosion noflux sedadv
astar       1
ws0         2.e-3
Kh          100

## Plotting ##
module      plot.Plot_base_model

## Output ##
module          general.Output
path            output/
filename        Scheldt1

requirements    zeta0 u0 zeta1 u1 T F a c0 c1 H B
## End ##
```

The # is the comment tag in Python and is also used in the input file. Everything after # is thus not regarded as code. The input file contains several blocks that start with the keyword `module`. Behind this is the location and name of the module. For example the module `Geometry2DV` is located in the package `analytical2DV` (which is located in the iFlow main directory in the folder `packages`). The set of input variables is listed under each of the modules. A short explanation of the variables is provided at the start of the manuals corresponding to each package. At the

end of the input file, you find the module for writing output and the keyword `requirements`. The `requirements` block lists the variables that need to be calculated and saved by the output module.

4 Sensitivity analysis

The module `general.Sensitivity` allows fully automatic sensitivity analyses over any number and any selection of parameters. To use this module, add the following to the input file.

```
module      general.Sensitivity
loopstyle   permutations
variables   Q1
Q1          np.linspace(0, 300, 40)
```

The module requires the keyword `variables` followed by the names of the input variable(s) to vary. Then each of these variables can be given a range. This range can be given as values separated by a space or as a python numpy code using `np` for numpy, as is done in the above example. Here it takes 40 values between 0 and 300.

The description of the output module `general.Output` needs to change somewhat. For example to

```
## Output ##
module          general.Output
path            output/
filename        out_Q@{'Q1'}
iteratesWith    general.Sensitivity
```

Note the keyword `iteratesWith`, which tells iFlow to write the output for every parameter configuration set by `general.Sensitivity`. Also note the file name. This now contains dynamic naming. The text `@'Q1'` will be automatically replaced by the value of `Q1` in current simulation. This way, each output file for each parameter configuration has a unique name.

5 Quick plotting tools

Plotting is done by special plot modules. An example of a plot module is provided in the tutorial. Please find the subfolder `plot` and the file `Plot_base_model.py`. Open this using any code editor (do not use MS Word or similar programs). You can use iFlow and python to make any type of plot, but iFlow provides some quick visualisation tools that allow you to plot variables with a single line of code. These tools are:

- `lineplot` - plot a variable of choice along one grid axis.
- `contourplot` - plot a colorplot of a variable of choice along two grid axes.
- `transportplot` - plot the contributions to the sediment transport.

Examples of these are given in the plotting module

6 Wanting more?

This tutorial might give you a taste of iFlow, but there is much more you can do. This tutorial should give you enough information to try the model on a different estuary. With some more work, you can also add new functionalities to the model. iFlow is written is such a way that you can easily write and add your own modules for simulation or plotting. To learn more we refer to the manuals with all details on iFlow.

In the future we plan to further extend this 'Getting Started' to a full tutorial on how to work with iFlow and program in iFlow.