Bifurcation data analysis workflow

I have written several scripts to automate the bifurcation image analysis workflow. However, the scripts are mixed in a pool of many other scripts and frequently, I have to think the process through again in my head to decide which ones to use, in what order. To save this effort for the future, I document the workflow here.

0. Generate crop_data and mask with ImageJ

1. Crop channel

Generate tiffstacks of channels from raw videos.

```
python crop_channel.py crop_data nd2Dir
```

2. Remove background

Remove the light artifact by subtracting the median of the stack.

```
python remove_background.py img_folder
```

3. PIV

Apply ImageJ PIV macro using headless mode.

```
python ij_piv.py img_folder
```

4. Rename ImageJ PIV data files

ImageJ macro generates files like "_1.txt", which does not give correct order when sorting based on filename. We therefore rename the files, and also remove excess columns from the data. The remaining data are saved as "{:05d}.csv".

```
python rename_ij_piv.py piv_folder
```

5. Apply mask

Add a "mask" column to .csv data, based on input mask image. In fact, mask can be applied to the compactPIV data, in which case the mask only needs to be computed once. The current implementation still has a lot of redundant computation, and can be optimized.

```
python apply_mask piv_folder mask_dir
```

6. Wrap PIV data

Wrap a bunch of csv files into a .mat file.

```
python wrap_piv.py piv_folder
```

7. Flow rate

Compute flow rate from PIV.

```
python flowrate.py main_piv_folder flowrate_folder dt
```