README for trajectory data files

Tim Tyree

3.17.2021

Any spiral tip trajectory supplied in this project can be reproduced on any commodity computer to machine floating point precision upon request.

1 row = 1 spiral tip location

```
#how I load this data in Python using a pandas.DataFrame instance
import pandas as pd
df = pd.read_csv('path/to/file.csv')
print      (      df.head(50)      ) # prints first 50 rows of trajectory
file
```

Pandas may be installed with a package manager such as brew using

```
$ brew install python
$ pip install pandas
```

Python may be launched from command line using the command `python3`.

Schema of any spiral tip trajectory '*.csv' file:

- t = time (milliseconds) since the start of simulation. Most recordings begin at time t=100 ms.
- n= number of spiral tips present in the computational domain
- x= x-position of spiral tip (pixels)
- y= y-position of spiral tip (pixels)
- grad_ux= the x-component of \nabla V_\text{mem}
- grad_uy= the y-component of \nabla V_\text{mem}
- grad_vx= the x-component of \nabla \partial_t V_\text{mem}
- grad_vy= the y-component of \nabla \partial_t V_\text{mem}
- frame= the number of frames since start of simulation
- particle= the number identifying individual trajectories. Unique identifier within a given '*.csv' but not across all files.

Note on local electrophysiological states:

Note some trajectory files may contain additional fields that describe the local electrophysiological state. This information can be computed for any spiral tip trajectory.

Note on '*_unwrap.csv' files:

- Spiral tip trajectory '*.csv' files ending with '*_unwrap.csv', has altered spiral tip positions. The altered spiral tip positions have the following properties:
  - Each trajectory beings at (0,0).
  - Each trajectory is continuous (periodic boundary conditions were unwrapped).

Example Usage:

```
import pandas as pd
df = pd.read_csv('path/to/file.csv')
print(df.describe()) # prints mean, median, etc… of trajectory file
x0=float(df[df.particle==4].head(0).x.values[0])
print(x0) # prints the initial x position of particle number 4
xf=float(df[df.particle==4].tail(0).x.values[0])
print(xf) # prints the final x position of particle number 4
print(xf-x0)
```