**伴随算法说明文档**

**获取伴随的流程**

1. 对原始数据进行预处理；
2. 对预处理后的的数据进行插值；
3. 对插值后的数据添加航向；
4. 将添加航向后的所有船只的数据按时刻分类；
5. 对按时刻分类得到的数据进行聚类取交，然后获取伴随。

**详细流程说明：**

1. 对原始数据进行预处理，即**pretreat.py**：

对原始数据进行预处理，包括修正时间戳（把科学计数法的时间戳改成整数）、去除时间戳重复的数据、按时间顺序排序、添加船号。

1. 对预处理后的的数据进行插值，即**interp1d.py**：

使用插值算法，将每条船的时间戳对齐。比如有a、b两条船，a在2、3、5、8、9、11时刻有数据，b在1、3、4、6、8、10时刻有数据，通过插值算法，我们可以生成a在2，4，6，8，10时刻的数据，同时也生成b在2，4，6，8，10时刻的数据，这样方便获取每条船在同一时刻的坐标。

1. 对插值后的数据添加航向，即**add\_cog.py**：

根据同一条船相邻两个时刻的坐标计算航向。

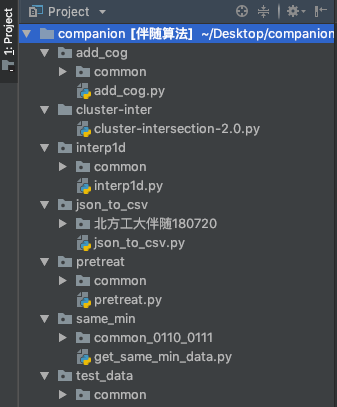
1. 将添加航向后的所有船只的数据按时刻分类，即**same\_min.py**：

将所有船只的数据按时刻分类，一个时刻为一个类，并且将这个时刻所有的船只数据都保存在同一个文件中。比如在在第二步插值和第三步添加航向后，我们得到了a、b、c三条船在2、4、6、8、10这五个时刻的数据，这时我们生成一个2.csv文件，其中包括a、b、c三条船在2这个时刻的数据，然后生成4.csv，其中包括a、b、c三条船在4这个时刻的数据，以此类推，然后生成n.csv，其中包括a、b、c三条船在n这个时刻的数据。

1. 对按时刻分类得到的数据进行聚类取交，然后获取伴随，即**cluster-inter.py**：

我们对第四步获得的数据进行处理，通过不断的聚类取交来获取伴随。比如、对2这个时刻的所有船只的数据2.csv使用DBSCAN聚类算法可以得到多个簇，然后根据航向和船之间距离对这多个簇进行筛选，筛选后的簇形成第一个候选集，同理得到4.csv的候选集，两个候选集取交，可以得到一个新的候选集，然后再获取6.csv的候选集，与之前得到的新的候选集取交，得到一个最新的候选集，以此类推，当候选集中中的某一个簇持续时间足够长的时候我们就认为这个簇中的船只是伴随。

**文件树形图：**



各个文件（夹）说明

1. test\_data/common/：

用来测试的原始数据

1. pretreat/pretreat.py：

用于对最一开始得到的数据进行预处理，需要设置三个参数，注意在windows操作系统下的文件路径写法与Mac操作系统的略微不一样，下图所示是Mac的写法，还有，不要忘记最后的反斜杠

pretreat.py参数说明：

|  |  |  |
| --- | --- | --- |
| 参数 | 意义 | 取值 |
| files\_dir | 原始数据的路径 | ‘../test\_data/common/’ |
| save\_dir | 修改后的数据的保存路径 | ‘common/’ |
| time\_index | 数据中时间戳在第几列 | 0 |

以上取值可以根据实际情况及需求进行修改或调整。

1. pretreat/common/

通过预处理后得到的数据文件。

1. interp1d/interp1d.py

对预处理后的的数据进行插值。首先把根据相邻两个时间戳之差是否大于3600将时间戳进行分段，新生成的每段的时间戳除了第一个和最后一个，每个时间戳数据都是300的整数倍，如果相邻两个时间戳大于某个值则在这两个时间戳之间不进行插值。比如预处理后的时间戳是这样的：

500，690，901，1111，5000，5201，5401，5609，

那么得到的新的时间戳是这样的：

500，600，900，1111，5000，5100，5400，5609，

interp1d.py参数说明：

|  |  |  |
| --- | --- | --- |
| 参数 | 意义 | 取值 |
| files\_dir | 预处理后的数据路径 | ‘../pretreat/common/’ |
| save\_dir | 修改后的数据的保存路径 | ‘common/’ |
| gap\_time | 隔多少秒插入一条数据 | 300 |
| no\_interp | 相邻时间戳之差大于no\_interp就不再这两个时间戳之间插入数据 | 3600 |
| time\_index | 数据中时间戳在第几列 | 1 |
| lon\_index | 数据中经度在第几列 | 2 |
| lat\_index | 数据中纬度在第几列 | 3 |

以上取值可以根据实际情况及需求进行修改或调整。

1. interp1d/common

通过插值算法后得到的数据文件

1. add\_cog/add\_cog.py

根据同一条船相邻两个时刻的坐标计算航向。

|  |  |  |
| --- | --- | --- |
| 参数 | 意义 | 取值 |
| files\_dir | 插值后的数据路径 | ‘../interp1d/common/’ |
| save\_dir | 添加航向后的数据的保存路径 | ‘common/’ |
| lon\_index | 数据中经度在第几列 | 2 |
| lat\_index | 数据中纬度在第几列 | 3 |

以上取值可以根据实际情况及需求进行修改或调整。

1. add\_cog/common/

添加航向后的数据文件.

1. same\_min/get\_same\_min\_data.py

|  |  |  |
| --- | --- | --- |
| 参数 | 意义 | 取值 |
| files\_dir | 添加了航向的的数据的路径 | ‘../add\_cog/common/’ |
| save\_dir | 添加航向后的数据的保存路径 | ‘common/’ |
| lon\_index | 数据中经度在第几列 | 2 |
| lat\_index | 数据中纬度在第几列 | 3 |
| sta\_time | 起始时间 | ‘2017-01-10 00:00:00’ |
| end\_time | 结束时间 | ‘2017-01-11 00:00:00’ |
| interval | 间隔（单位：min） | 5 |
| time\_index | 数据中时间在第几列 | 1 |

以上取值可以根据实际情况及需求进行修改或调整。

1. same\_min/common\_0110\_0111

这个文件夹中的文件是1月10号到1月11号之间所有船只在同一时刻的数据，一个时刻的数据为一个文件

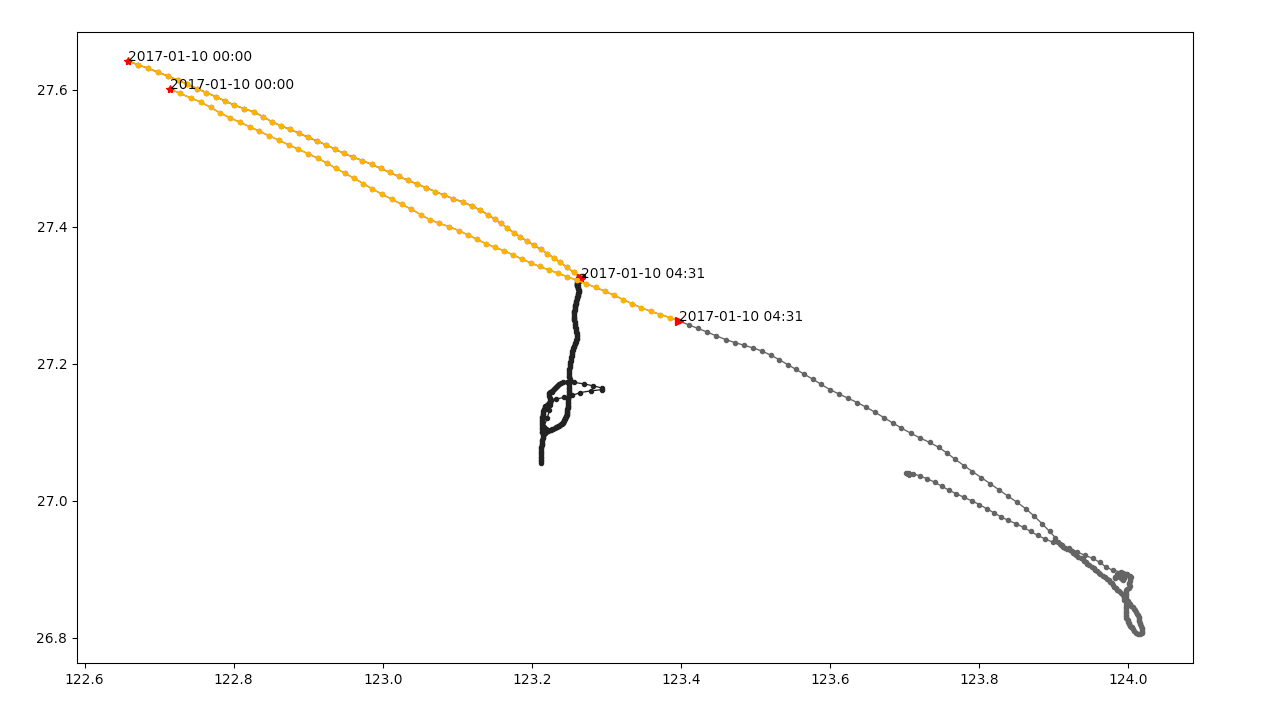
1. cluster-inter/cluster-intersection-2.0.py

cluster-intersection-2.0.py参数设置：

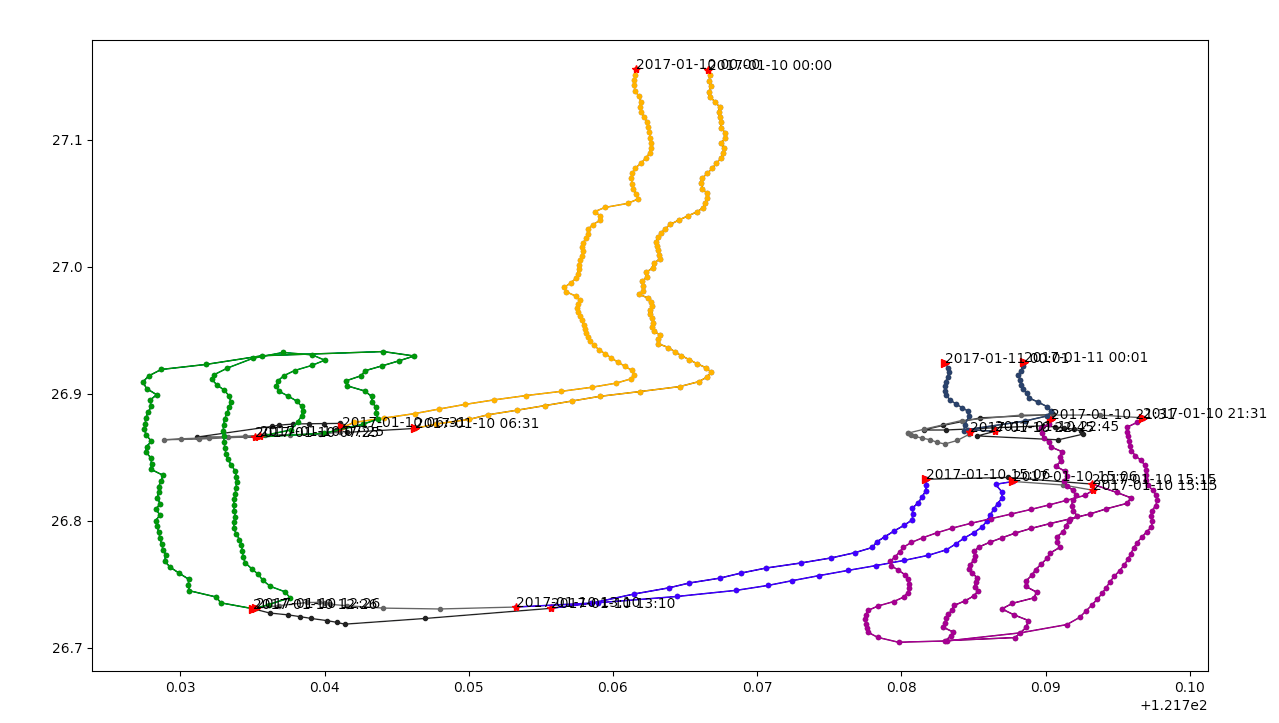
|  |  |  |
| --- | --- | --- |
| 参数 | 意义 | 取值 |
| data\_dir | 同一时刻所有船只的数据的路径 | ‘../same\_min/  common\_0110\_0111/’ |
| draw\_dir | 用于绘图的数据的路径 | ‘common/’ |
| min\_n\_ship | 至少几条船才判断为伴随 | 2 |
| max\_dis | 两个船的距离大于这个数值则判断为非伴随（单位：m） | 1000\*10 |
| max\_dif\_cog | 两只船航向之差大于这个数值则判断为非伴随（单位：度） | 45 |
| min\_duration | 至少持续多久才判断为伴随 | ‘../interp1d/common/’ |
| gap | 数据的时间间隔是多少（由插值算法决定）（单位：s） | 300 |
| colors | 绘制伴随的彩色颜色集 | ['orange', 'green', 'blue', 'purple', '#123456'] |
| full\_track\_colors | 绘制船只在这个时间段的整条轨迹的不同程度的灰色颜色集 | [(0.1, 0.1, 0.1), (0.32, 0.32, 0.32), (0.64, 0.64, 0.64), (0.96, 0.96, 0.96)] |
| id\_index | 数据中船号在第几列 | 0 |
| time\_index | 数据中时间戳在第几列 | 1 |
| lon\_index | 数据中经度在第几列 | 2 |
| lat\_index | 数据中纬度在第几列 | 3 |
| cog\_index | 数据中航向在第几列 | 4 |

以上取值可以根据实际情况及需求进行修改或调整。

**部分伴随图展示及说明：**



在上图中，伴随段的时间从2017-01-10 00:00开始，到2017-01-10 04:21结束，伴随开始的点是红色的五角星，结束的点是红色的三角形，可以明显的看出伴随结束的原因是有一条船有大幅度的转弯，导致这条船的航向与另一条船的航向之差大于我们设置的最小航向差min\_dis\_cog，于是被判断伴随结束。



上图表示，这一天这两只船共有五次伴随，即五个伴随段，每个伴随段都用不同的颜色区分开了每个伴随段依然是从红色的五角星开始，结束于红色的三角形。

