smam: Statistical Modeling for Animal Movements

Jun Yan, Vladimir Pozdnyakov, Chaoran Hu

1. Introduction

This package is designed for animal movement analysis. We totally implemented three models, Brownian motion with measurement error [4], moving-resting process with embedded Brownian motion [1,2], and moving-resting-hunting process with embedded Brownian motion [3]. These three models can also be fitted with specific parts of dataset, which is called *segment estimation* in this vignette. From the perspective of ecology and animal movement, segment estimation can be hired for seasonal analysis. We also provided toolbox for seasonal analysis within this package, which can subset data according to given date interval.

In this vignette, we play with the GPS dataset of a mountain lion and quantify her movement behavior. First of all, let us take a look at the dataset.

```
library(smam)
head(f109)
```

```
##         date      time centerE centerN
## 1 2009-01-18  1.000000     0.0     0.0
## 2 2009-01-18  2.000000     0.0    -0.1
## 3 2009-01-18  3.016667     0.0    -0.1
## 4 2009-01-23 27.000000    -0.4    -0.4
## 5 2009-01-24 34.983333     0.3    -1.1
## 6 2009-01-25 42.983333    -0.8    -0.3
```

The first column, 'date', is the date when the observation is collected. The second column, 'time', is the corresponding time line of observations. The last two columns, 'centerE' and 'centerN', are the GPS coordinates.

2. Whole dataset analysis

In this section, we are going to show how to use whole dataset to fit BMME, moving-resting model and moving-resting-handling model with R function `fitBMME`, `fitMR` and `fitMRH`. Let's see the standard format of dataset used by these functions first. It should be a `data.frame` whose first column is the observation time and the other columns are the location coordinates, as showing below.

```
dat1 <- f109[, -1]
head(dat1)
```

```
##        time centerE centerN
## 1  1.000000     0.0     0.0
## 2  2.000000     0.0    -0.1
## 3  3.016667     0.0    -0.1
## 4 27.000000    -0.4    -0.4
## 5 34.983333     0.3    -1.1
## 6 42.983333    -0.8    -0.3
```

Now, we can fit the above data by the following code. (We do not show the result, because the estimation process for moving-resting-handling model is time consuming and maybe take several hours.)

```
## fit BMME model
fitBMME(dat1, start = c(0.5, 0.5))

## fit moving-resting model with full likelihood
fitMR(dat1, start = c(10, 0.1, 1), likelihood = "full")

## fit moving-resting model with composite likelihood
fitMR(dat1, start = c(10, 0.1, 1), likelihood = "composite")

## fit moving-resting-handling model (serial computation)
## numThread is less or equal to 1 will call serial computation
fitMRH(dat1, start = c(4, 0.4, 0.1, 1, 0.5), numThreads = 1)

## fit moving-resting-handling model (parallel computation)
## numThread is greater or equal to 2 will call parallel computation
## with given number of threads
fitMRH(dat1, start = c(4, 0.4, 0.1, 1, 0.5), numThreads = 4)
```

We also mention that the sampling functions, `rBMME`, `rMR`, and `rMRH`, are given for BMME model, moving-resting model and moving-resting-handling model.

3. Segment dataset analysis

In this section, we are going to show the usage of segment estimation. In contrast with whole data estimation, the segment estimation only use part of observations. Segment estimation evaluates the likelihood for each segments of observation and sums them together as the likelihood be used in MLE process. So, in order to use segment estimation, we should tell R which part of observations we are going to use by adding additional 0-n column, where n can be any non-zero integer. The 0's in this column means discard this observation and other non-zero integers are the labels for each segments.

```
## the name of segment indicator column can be given by user
batch <- c(rep(0, 30), rep(1, 40), rep(0, 20), rep(2, 30), rep(0, 80))
dat2 <- cbind(dat1, batch) ## the result of dat2 is shown in appendix.
```

Then, when we apply `fitBMME`, `fitMR`, `fitMRH` on `dat2`, only segment 1 (batch = 1) and segment 2 (batch = 2) will be used. Then, we are going to show the code for fit these three models. (Note that, the name of indicator column should be set in `segment`.)

```
## fit BMME model
fitBMME(dat2, start = c(0.5, 0.5), segment = "batch")

## fit moving-resting model with full likelihood
fitMR(dat2, start = c(10, 0.1, 1), segment = "batch", likelihood = "full")

## fit moving-resting model with composite likelihood
fitMR(dat2, start = c(10, 0.1, 1), segment = "batch", likelihood = "composite")

## fit moving-resting-handling model
fitMRH(dat2, start = c(4, 0.4, 0.1, 1, 0.5), segment = "batch", numThreads = 2)
```

4. Prediction of hidden states

The code for prediction of hidden states is still in process. But you can try and explore them with `?fitStateMR` and `?fitStateMRH`.

References

[1] Yan, J., Chen, Y., Lawrence-Apfel, K., Ortega, I. M., Pozdnyakoc, V., Williams, S., and Meyer, T. (2014) A moving-resting process with an embedded Brownian motion for animal movements. Population Ecology. 56(2): 401–415. doi:10.1007/s10144-013-0428-8

[2] Pozdnyakov, V., Elbroch, L., Labarga, A., Meyer, T., and Yan, J. (2017) Discretely observed Brownian motion governed by telegraph process: estimation. Methodology and Computing in Applied Probability. doi:10.1007/s11009-017-9547-6.

[3] Pozdnyakov, V., Elbroch, L., Hu, C., Meyer, T., and Yan, J. (2018+) On estimation for Brownian motion governed by telegraph process with multiple off states. <arXiv:1806.00849>

[4] Pozdnyakov, V. , Meyer, T. , Wang, Y. and Yan, J. (2014), On modeling animal movements using Brownian motion with measurement error. Ecology, 95: 247-253. doi:10.1890/13-0532.1

Appendix

We show the data for segment estimation here.

```
dat2
```

```
##            time centerE centerN batch
## 1      1.000000     0.0     0.0     0
## 2      2.000000     0.0    -0.1     0
## 3      3.016667     0.0    -0.1     0
## 4     27.000000    -0.4    -0.4     0
## 5     34.983333     0.3    -1.1     0
## 6     42.983333    -0.8    -0.3     0
## 7     50.983333    -0.8    -0.3     0
## 8     58.983333    -0.8    -0.3     0
## 9     66.966667     0.1    -0.4     0
## 10    74.966667    -1.2    -2.8     0
## 11    82.966667    -1.3    -2.9     0
## 12    90.966667    -1.3    -2.9     0
## 13    98.966667    -1.3    -2.9     0
## 14   106.966667    -1.3    -2.9     0
## 15   114.966667    -1.0    -2.7     0
## 16   131.000000    -1.3    -2.9     0
## 17   138.983333     0.4    -3.6     0
## 18   146.983333    -1.1    -3.5     0
## 19   154.966667    -0.7    -4.3     0
## 20   162.966667    -0.7    -4.4     0
## 21   170.966667    -0.7    -4.4     0
## 22   178.966667    -0.7    -4.5     0
## 23   186.966667    -0.7    -4.5     0
## 24   194.966667    -0.7    -4.4     0
## 25   202.966667    -0.7    -4.4     0
## 26   210.966667    -0.7    -4.4     0
## 27   218.966667    -0.5    -4.3     0
## 28   226.983333    -0.7    -4.4     0
## 29   234.966667    -0.7    -4.5     0
## 30   242.966667    -0.4    -4.5     0
## 31   250.966667    -0.2    -4.5     1
## 32   258.966667    -0.3    -4.6     1
## 33   266.966667    -0.5    -5.9     1
## 34   274.966667    -0.5    -5.9     1
## 35   282.966667    -0.7    -4.5     1
## 36   291.000000    -0.3    -4.3     1
## 37   314.966667    -0.4    -4.4     1
## 38   322.966667    -0.4    -3.2     1
```

```
## 39   330.983333     1.7    -0.4     1
## 40   339.000000     2.4     0.5     1
## 41   346.983333     2.4     0.6     1
## 42   354.983333     2.4     0.6     1
## 43   362.966667     2.4     0.6     1
## 44   370.966667     2.4     0.5     1
## 45   378.966667     2.3     0.6     1
## 46   386.966667     2.3     0.6     1
## 47   394.983333     2.4     0.6     1
## 48   402.983333     2.4     0.6     1
## 49   410.983333     2.4     0.6     1
## 50   418.983333     2.3     0.6     1
## 51   426.983333     2.3     0.6     1
## 52   434.983333     2.3     0.6     1
## 53   442.966667     3.8     1.6     1
## 54   450.966667     3.6     0.3     1
## 55   458.983333     3.4     0.7     1
## 56   466.966667     5.3     1.4     1
## 57   474.966667     3.5     0.3     1
## 58   482.983333     3.7     0.6     1
## 59   490.983333     3.5     0.3     1
## 60   507.000000     3.5     0.3     1
## 61   514.966667     4.1     1.0     1
## 62   522.966667     3.5     0.3     1
## 63   530.966667     3.7     0.7     1
## 64   538.983333     3.7     0.7     1
## 65   546.966667     3.5     0.3     1
## 66   554.966667     3.5     0.9     1
## 67   562.966667     3.5     0.9     1
## 68   570.966667     3.4     0.8     1
## 69   586.966667     3.5     0.8     1
## 70   594.966667     3.4     0.8     1
## 71   602.983333     3.6     1.0     0
## 72   610.966667     3.6     1.0     0
## 73   618.966667     3.5     0.8     0
## 74   634.966667     3.6     1.0     0
## 75   642.966667     5.4     0.9     0
## 76   650.983333     6.3     1.5     0
## 77   658.966667     6.3     1.5     0
## 78   666.966667     4.9     0.8     0
## 79   674.983333     5.0     0.8     0
## 80   682.966667     5.0     0.8     0
## 81   690.983333     4.7     0.4     0
## 82   698.983333     5.1     1.0     0
## 83   706.966667     5.1     1.0     0
```

```
## 84   714.966667     4.0     0.4     0
## 85   722.983333     2.6     0.4     0
## 86   730.966667     2.6     0.4     0
## 87   738.966667     2.7     0.4     0
## 88   754.966667     3.6     1.0     0
## 89   762.966667     2.9     0.4     0
## 90   778.966667     2.7     0.5     0
## 91   787.000000     2.7     0.4     2
## 92   794.983333     3.6     0.9     2
## 93   802.966667     3.9    -0.3     2
## 94   810.966667     3.2     0.3     2
## 95   818.983333     3.1     0.4     2
## 96   826.966667     3.2     0.4     2
## 97   834.966667     3.2     0.3     2
## 98   842.983333     3.6     1.0     2
## 99   850.983333     3.6     0.8     2
## 100  858.966667     3.2     0.3     2
## 101  866.966667     3.6     0.7     2
## 102  874.966667     3.5     0.8     2
## 103  882.983333     3.2     0.3     2
## 104  891.000000     3.8     0.9     2
## 105  898.966667     3.9     1.0     2
## 106  906.966667     2.9     0.2     2
## 107  914.983333     2.6     0.3     2
## 108  922.966667     2.7     0.2     2
## 109  930.966667     2.9     0.2     2
## 110  939.000000     2.7     0.2     2
## 111  946.966667     2.7     0.2     2
## 112  954.966667     2.9     0.2     2
## 113  970.966667     3.0     0.4     2
## 114  994.950000     4.5     0.7     2
## 115 1002.966667     4.9     0.8     2
## 116 1018.966667     5.2     0.9     2
## 117 1042.950000     4.9     1.1     2
## 118 1050.950000     6.4     1.4     2
## 119 1058.950000     6.5     1.4     2
## 120 1066.950000     6.5     1.5     2
## 121 1074.933333     8.2     0.5     0
## 122 1082.950000     6.6     1.7     0
## 123 1090.933333     7.1     3.1     0
## 124 1098.933333     6.4     2.1     0
## 125 1114.933333     6.5     1.7     0
## 126 1138.916667     5.2     0.9     0
## 127 1146.883333     4.3     0.7     0
## 128 1162.883333     4.3     0.8     0
```

```
## 129 1170.883333     4.3     0.8     0
## 130 1178.916667     4.3     0.9     0
## 131 1186.916667     4.3     0.8     0
## 132 1194.883333     4.3     0.8     0
## 133 1202.916667     4.3     0.7     0
## 134 1210.883333     4.4     1.1     0
## 135 1218.900000     4.3     0.7     0
## 136 1226.900000     4.3     0.7     0
## 137 1234.900000     3.7     1.1     0
## 138 1242.916667     4.3     0.7     0
## 139 1250.900000     4.3     1.1     0
## 140 1258.916667     4.2     1.4     0
## 141 1266.883333     4.3     0.7     0
## 142 1274.900000     4.3     0.7     0
## 143 1282.883333     4.1     1.3     0
## 144 1290.916667     3.9     0.4     0
## 145 1306.883333     5.2     1.0     0
## 146 1322.850000     5.1     0.9     0
## 147 1330.833333     5.2     0.9     0
## 148 1338.833333     6.4     1.6     0
## 149 1346.866667     6.4     1.9     0
## 150 1354.833333     6.5     1.9     0
## 151 1362.850000     8.0     0.4     0
## 152 1370.866667    10.3    -1.0     0
## 153 1378.833333    10.4    -1.1     0
## 154 1386.850000     9.1    -3.7     0
## 155 1394.866667     9.3    -5.2     0
## 156 1402.833333     8.9    -5.5     0
## 157 1410.850000     9.4    -3.6     0
## 158 1418.833333     9.1    -3.6     0
## 159 1426.833333     9.1    -3.7     0
## 160 1434.833333     7.2    -1.5     0
## 161 1442.850000     7.1    -1.8     0
## 162 1450.833333     7.0    -2.4     0
## 163 1458.866667     7.2    -1.5     0
## 164 1466.850000     7.1    -1.8     0
## 165 1474.833333     7.2    -1.5     0
## 166 1482.833333     7.2    -1.5     0
## 167 1490.866667     7.2    -1.5     0
## 168 1498.833333     7.2    -1.5     0
## 169 1506.833333     6.6    -1.1     0
## 170 1514.833333     6.2    -1.3     0
## 171 1522.850000     6.2    -1.3     0
## 172 1530.833333     6.4    -0.9     0
## 173 1538.850000     5.8    -0.6     0
```

```
## 174 1546.866667     6.0    -0.5     0
## 175 1554.833333     6.1    -0.4     0
## 176 1562.866667     5.7    -0.8     0
## 177 1570.833333     5.7    -0.8     0
## 178 1578.833333     3.7    -1.1     0
## 179 1586.850000     1.5     0.9     0
## 180 1594.850000     1.5     0.9     0
## 181 1602.833333     3.0     0.0     0
## 182 1610.850000     3.4     0.5     0
## 183 1618.850000     3.4     0.5     0
## 184 1626.850000     3.3     0.8     0
## 185 1634.850000     3.7     1.0     0
## 186 1642.833333     3.6     1.1     0
## 187 1650.833333     2.5    -0.2     0
## 188 1658.833333     2.4     0.2     0
## 189 1666.833333     2.4     0.2     0
## 190 1674.833333     1.0     1.1     0
## 191 1682.866667     0.9     1.6     0
## 192 1690.833333     1.6     0.8     0
## 193 1698.833333     2.0     0.3     0
## 194 1706.850000     2.4     0.2     0
## 195 1714.850000     2.4     0.2     0
## 196 1722.833333     2.0    -0.1     0
## 197 1730.850000     2.3     0.6     0
## 198 1738.833333     2.3     0.6     0
## 199 1746.850000     1.4     0.1     0
## 200 1754.850000     1.5     0.9     0
```