MATH 8670 Machine Learning Competition

Predicting the location and speed of a flight

The goal of this contest is to impute the missing values for location and speed in the ADS–B flight data.

Background. ADS–B data can be automatically collected from the aircraft's navigation system (https://en.wikipedia.org/wiki/Automatic_dependent_surveillance_%E2%80%93_broadcast). It provides the basic information of a flight, including the location (latitude, longitude, & altitude) and speed (ground speed & vertical speed). However, ADS-B data is not recorded by every second. In most ADS-B data sets, every two consecutive records can have a time gap of 15-30 seconds. It is a challenge to find out what happens during the time gap. Therefore, our task is to predict the location and speed by second for the ADS-B data.

Kaggle website

I will work on this. Not sure if Kaggle will handle such a complex submission and evaluation method. If it doesn't work, I will figure out a different way to process your submission, for example, by email.

Description of variables

- ID: Flight ID. Has no meaning.
- Time: by second. Converted from the local time. Time point 1 of a flight is arbitrarily taken before or during taking-off.
- Latitude: latitude (degree) of the airplane.
- Longitude: longitude (degree) of the airplane.
- AltB: Barometric altitude (feet) of the airplane.
- GndSpd: Ground speed by knot (1 knot = 1.15 mph). Non-negative numeric value.
- VSpd: Vertical speed (feet per minute). The numeric value can be positive (for climbing), negative (for descending), or zero (for level).

Data

Here is a quick look at the training set. The training data contains 30 flight profiles. The five parameters (lat, long, alt, ground speed, vertical speed) are recorded by second.

```
train = read.csv("data/train.csv")
unique(train$ID)
```

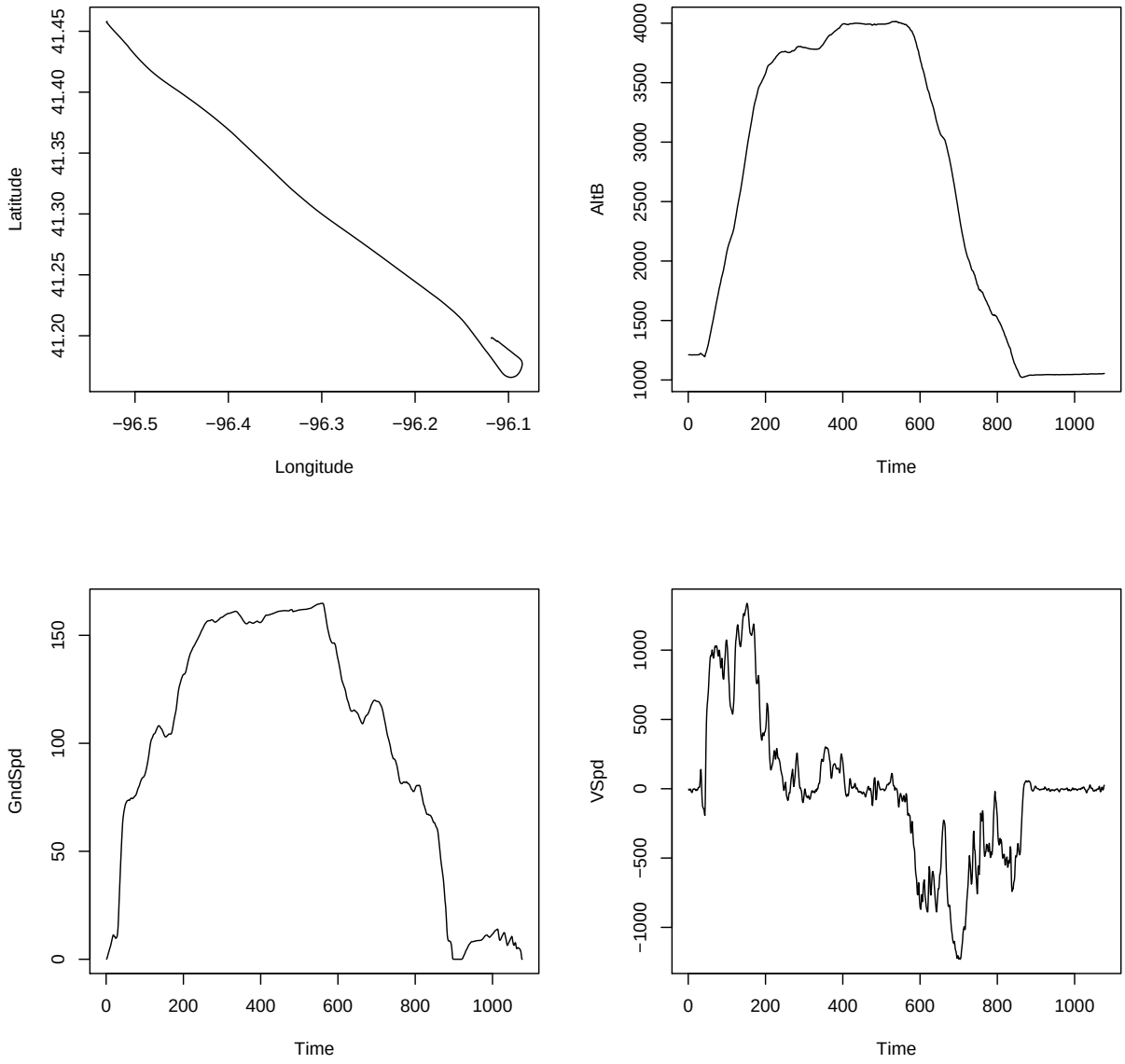
```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28 29 30
```

```
head(train, 10)
```

```
##    ID Time Latitude Longitude   AltB GndSpd   VSpd
## 1   1    1 41.45808 -96.52979 1211.9   0.02  -2.93
## 2   1    2 41.45807 -96.52979 1210.9   0.33  -9.46
## 3   1    3 41.45807 -96.52979 1211.9   0.93 -10.99
## 4   1    4 41.45806 -96.52981 1211.9   1.61  -4.02
```

```
## 5   1    5 41.45806 -96.52983 1211.9   2.27  -1.95
## 6   1    6 41.45805 -96.52984 1211.9   2.94  -3.59
## 7   1    7 41.45803 -96.52985 1210.9   3.58 -10.83
## 8   1    8 41.45803 -96.52988 1210.9   4.18 -21.27
## 9   1    9 41.45801 -96.52990 1209.9   4.76 -27.42
## 10  1   10 41.45800 -96.52993 1210.9   5.35 -26.80
```

```
par(mfrow=c(2,2))
plot(Latitude ~ Longitude, data = train[train$ID == 1,], type='l')
plot(AltB ~ Time, data = train[train$ID == 1,], type='l')
plot(GndSpd ~ Time, data = train[train$ID == 1,], type='l')
plot(VSpd ~ Time, data = train[train$ID == 1,], type='l')
```



Now let us look at the test set. The test data contains 80 flight profiles. The five parameters are recorded by a fixed time gap of 15 seconds.

```
test = read.csv("data/test.csv")
unique(test$ID)
```

```
##  [1]  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49
## [20]  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
## [39]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87
## [58]  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106
## [77] 107 108 109 110
```

```
head(test, 20)
```

```
##    ID Time Latitude Longitude   AltB GndSpd   VSpd
## 1  31   57 41.26664 -95.75749 1240.1  44.46 -71.44
## 2  31   58       NA        NA     NA     NA     NA
## 3  31   59       NA        NA     NA     NA     NA
## 4  31   60       NA        NA     NA     NA     NA
## 5  31   61       NA        NA     NA     NA     NA
## 6  31   62       NA        NA     NA     NA     NA
## 7  31   63       NA        NA     NA     NA     NA
## 8  31   64       NA        NA     NA     NA     NA
## 9  31   65       NA        NA     NA     NA     NA
## 10 31   66       NA        NA     NA     NA     NA
## 11 31   67       NA        NA     NA     NA     NA
## 12 31   68       NA        NA     NA     NA     NA
## 13 31   69       NA        NA     NA     NA     NA
## 14 31   70       NA        NA     NA     NA     NA
## 15 31   71       NA        NA     NA     NA     NA
## 16 31   72       NA        NA     NA     NA     NA
## 17 31   73 41.26193 -95.75766 1280.1  73.30 346.98
## 18 31   74       NA        NA     NA     NA     NA
## 19 31   75       NA        NA     NA     NA     NA
## 20 31   76       NA        NA     NA     NA     NA
```

Linear interpolation

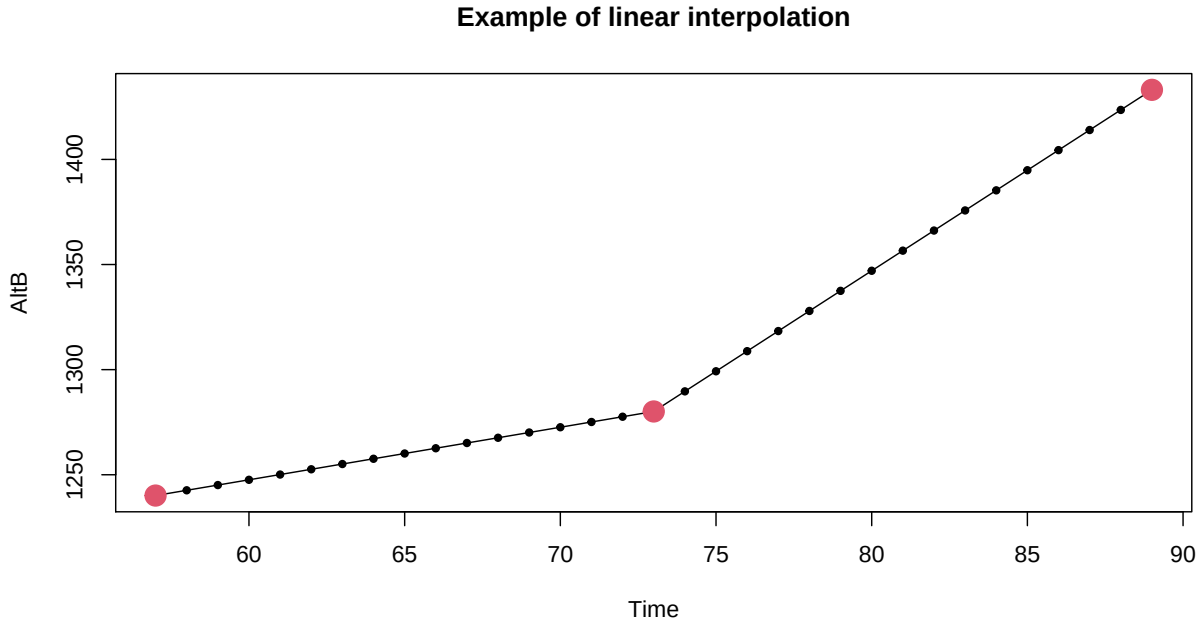
We can use the linear interpolation to impute the missing values. The following code can do the imputation and generate an example of submission.

```
example = test
for (i in unique(test$ID)){
  for (j in 3:7){
    parameter = example[example$ID == i, j]
    complete_parameter = na.omit(parameter)
    nRecords = length(complete_parameter)
    locRecords = which(parameter %in% complete_parameter)[-nRecords]
    for (k in 1:15){
      parameter[locRecords+k] = (1-k/16)*complete_parameter[-nRecords] +
          (k/16)*complete_parameter[-1]
    }
    example[example$ID == i, j] = parameter
  }
}
head(example, 20)
```

```
##    ID Time Latitude Longitude     AltB   GndSpd      VSpd
```

```
## 1  31   57 41.26664 -95.75749 1240.100 44.46000 -71.44000
## 2  31   58 41.26634 -95.75750 1242.600 46.26250 -45.28875
## 3  31   59 41.26605 -95.75751 1245.100 48.06500 -19.13750
## 4  31   60 41.26575 -95.75752 1247.600 49.86750   7.01375
## 5  31   61 41.26546 -95.75753 1250.100 51.67000  33.16500
## 6  31   62 41.26516 -95.75754 1252.600 53.47250  59.31625
## 7  31   63 41.26487 -95.75756 1255.100 55.27500  85.46750
## 8  31   64 41.26458 -95.75757 1257.600 57.07750 111.61875
## 9  31   65 41.26428 -95.75758 1260.100 58.88000 137.77000
## 10 31   66 41.26399 -95.75759 1262.600 60.68250 163.92125
## 11 31   67 41.26369 -95.75760 1265.100 62.48500 190.07250
## 12 31   68 41.26340 -95.75761 1267.600 64.28750 216.22375
## 13 31   69 41.26311 -95.75762 1270.100 66.09000 242.37500
## 14 31   70 41.26281 -95.75763 1272.600 67.89250 268.52625
## 15 31   71 41.26252 -95.75764 1275.100 69.69500 294.67750
## 16 31   72 41.26222 -95.75765 1277.600 71.49750 320.82875
## 17 31   73 41.26193 -95.75766 1280.100 73.30000 346.98000
## 18 31   74 41.26157 -95.75768 1289.662 73.60375 366.57000
## 19 31   75 41.26122 -95.75770 1299.225 73.90750 386.16000
## 20 31   76 41.26086 -95.75772 1308.787 74.21125 405.75000
```

```
plot(AltB ~ Time, data = example[1:33,],type='o',pch=20,
     main="Example of linear interpolation")
points(AltB ~ Time, data = test[1:33,], pch=19, col=2, cex=2)
```



Format of submission

Your submission file should be in the csv format with 208288 rows and 7 columns: `ID`, `Time`, `Latitude`, `Longitude`, `AltB`, `GndSpd` and `VSpd`. Example of the submission can be found from the above linear interpolation example. Note that in the 208288 rows, the given time points are also included.

Evaluation metrics

We will use the MSE from the linear interpolation as a baseline. The corresponding MSE's for the five variables are given as follows.

```
##                      MSE
## Latitude  2.483012e-08
## Longitude 4.137023e-08
## AltB      1.284327e+02
## GndSpd    1.793412e+00
## VSpd      2.214808e+04
```

The evaluation metric will be calculated using the following formula.

$$\text{Total Error} = \frac{1}{5}\sum_{j=1}^{5}\frac{MSE_j^{(submission)}}{MSE_j^{(linear)}}$$

where $j = 1, 2, 3, 4, 5$ indicates the five parameters: `Latitude`, `Longitude`, `AltB`, `GndSpd` and `VSpd`. The total error is 1 if the submission is same as linear interpolation. A better prediction can decrease the total error.

Task

1. Create the most accurate classifier that you can for the data, as measured by the test data.
2. Write a 10-15 page slides summarizing your approach to
   (a) formulating the model design matrix,
   (b) building the model and tuning parameters,
   (c) validating the model by training & validation sets, or other approach,
   (d) results from all attempts,
   (e) your findings from the data.

NOTE: This is an individual competition. You may not share your code to anyone except the instructor.

Deadlines

- November 23 (11:59 pm): Final prediction submission.
- November 24 (4 pm): Presentations. Each presentation should be around 10 minutes.
- November 25 (11:59 pm): Slides and code submission.

Grading

- Total points: 15
  - Accuracy of the model: 6
    - You will receive a score from 6.0, 5.5, 5.0, 4.5, 4.0, 3.5, based on your rank in class.
  - Progress made from multiple submissions: 2
    - Progress in accuracy: 1
    - Multiple submissions: 1
  - Presentation: 6
    - Design matrix: 1
    - Model selection: 1
    - Parameter tuning: 1
    - Model validation: 1
    - Results: 1
    - Presentation & slides: 1
  - Met the deadlines: 1