Contest

Ali Alghaithi

Methods:

1. Linear interpolation
2. Stineman interpolation
3. Modeling Each Point
4. Linear + ML models
5. tslm is used to fit linear models to time series including trend and seasonality components.

linear interpolation

```
test <- read_csv("~/Box/Advance NL Class/Contest1/test.csv")
for (i in unique(test$ID)){
for (j in 3:7){
test[test$ID == i, j] = na.interpolation(test[test$ID == i, j], option = "linear" )
}
}
```

Filter

	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

Stineman interpolation

```
library(imputeTS)
for (i in unique(test$ID)){
for (j in 3:7){
test[test$ID == i, j] = na.interpolation(test[test$ID == i, j], option = "stine" )

}
}
```

stinterp

From stinepack v1.3
by Tomas Johannesson

99.99th
Percentile

A Consistently Well Behaved Method Of Interpolation

Returns the values of an interpolating function that runs through a set of points in the xy-plane according to the algorithm of Stineman (1980).

According to Stineman, the interpolation procedure has "the following properties:

1. If values of the ordinates of the specified points change monotonically, and the slopes of the line segments joining the points change monotonically, then the interpolating curve and its slope will change monotonically.
2. If the slopes of the line segments joining the specified points change monotonically, then the slopes of the interpolating curve will change monotonically.
3. Suppose that the conditions in (1) or (2) are satisfied by a set of points, but a small change in the ordinate or slope at one of the points will result conditions (1) or (2) being not longer satisfied. Then making this small change in the ordinate or slope at a point will cause no more than a small change in the interpolating curve."

Linear and Stineman interpolation

```
library(imputeTS)
for (i in unique(test$ID)){
for (j in 3:7){
test[test$ID == i, j] = na.interpolation(test[test$ID == i, j], option = "stine" )

}
}
test1<- test


test <- read_csv("~/Box/Advance NL Class/Contest1/test.csv")
for (i in unique(test$ID)){
for (j in 3:7){
test[test$ID == i, j] = na.interpolation(test[test$ID == i, j], option = "linear" )
}
}

test2<- test

test <- (test2+ test1) /2
```

Linear	Stineman	Stineman+ Linear	
0.11978	0.12504	0.11943	Kaggle score

Modeling Each Point

- Creating New Variables

Other variables	start...	End...	Avg...
...	...	...	...

- Creating another 15 data sets, each one represent a step, train1 = data of step 1

Linear + ML models

- Make train looks like test (deleting 15 seconds)
- Use Linear interpolation to imputed NA (Imputed train data)

- Imputed train data	- Original train
Predictors	Target Variables
. . .	. . .

Model Design Matrix

Latitude	yLatitude	Longitude	yLongitude	AltB	yAltB	GndSpd	yGndSpd	VSpd	yVSpd
41.45808	41.45808	-96.52979	-96.52979	1211.900	1211.9	0.020000	0.02	-2.930000	-2.93
41.45807	41.45807	-96.52982	-96.52979	1211.757	1210.9	0.644876	0.33	-2.905629	-9.46
41.45806	41.45807	-96.52986	-96.52979	1211.633	1211.9	1.273559	0.93	-2.881014	-10.99
41.45804	41.45806	-96.52989	-96.52981	1211.528	1211.9	1.906124	1.61	-2.856104	-4.02
41.45803	41.45806	-96.52992	-96.52983	1211.443	1211.9	2.542646	2.27	-2.830834	-1.95

```
# model part
```{r}
mLatitude <- lm(yLatitude ~  ., data = x)
mLongitude <- lm( yLongitude~  Latitude+ Longitude+AltB, data = all_train)
mAltB<- lm( yAltB~ Latitude+ Longitude+AltB+GndSpd+VSpd, data = all_train)
mGndSpd <- lm( yGndSpd~ Latitude*Longitude*AltB*GndSpd*VSpd, data = all_train)
mVSpd <- lm( yVSpd~ Latitude*Longitude*AltB*GndSpd*VSpd, data = all_train)

```
```

Kaggle score(only updating flight 32)
0.11940

tsml: time series linear model

- tslm is used to fit linear models to time series including trend and seasonality components.
- Stationarity assumptions

tsml: time series linear model

tslm is largely a wrapper for lm() except that it allows variables "trend" and "season" which are created on the fly from the time series characteristics of the data. The variable "trend" is a simple time trend and "season" is a factor indicating the season (e.g., the month or the quarter depending on the frequency of the data).

```
for (i in unique(train$ID)){
new_train <- train %>% filter(ID==i)

ts_fit = function(new_train) {
  VSpd <- ts(new_train$VSpd,frequency = 16)
  EndLatitude <- new_train$EndLatitude
  startLatitude <- new_train$startLatitude
  EndLongitude <- new_train$EndLongitude
  startLongitude <- new_train$startLongitude
  startAltB <- new_train$startAltB
  EndAltB <- new_train$EndAltB
  AvgLatitude <- new_train$AvgLatitude

  EndVSpd <- new_train$EndVSpd
  startVSpd <- new_train$startVSpd
  EndGndSpd <- new_train$EndGndSpd
  startGndSpd <- new_train$startGndSpd
  AvgVSpd <- new_train$AvgVSpd
  AvgGndSpd <- new_train$AvgGndSpd

  fit <- tslm(VSpd ~ trend  + season    + EndVSpd+ startVSpd +EndGndSpd+startGndSpd+EndLatitude+startLatitude +startLongitude+
EndLongitude+startAltB+EndAltB+AvgVSpd+AvgGndSpd )
  return(fit)
}

model_fit<- ts_fit(new_train)
forecast <- forecast(model_fit, newdata = InData12)

M1<- as.data.frame(forecast$mean)
output[,i] <- M1$x

}
```

V1	V2	V3	V4	V5	V6
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
71.94370424	-5848.031	-223.136857	4.035715e+01	26.4897863	50.355387
70.49340573	-5866.849	-226.016301	4.400755e+01	24.2730331	48.285091
67.40967439	-5885.795	-228.922412	4.259121e+01	23.9409985	45.630353
67.47773409	-5900.573	-233.883894	3.718087e+01	23.6780547	43.854749
68.53325648	-5897.608	-234.647968	3.185213e+01	24.1254573	44.658781
63.25758484	-5894.201	-234.461857	3.172093e+01	24.2979249	47.121674
56.57654006	-5907.109	-233.055746	3.410521e+01	26.3176651	47.627642
57.49265946	-5923.646	-226.705005	3.600333e+01	29.9602625	46.018986
62.99862961	-5915.794	-215.718153	3.719875e+01	33.5064963	44.834051
65.25519678	-5889.079	-204.082227	4.057653e+01	33.1898296	44.005665

1–10 of 833 rows | 1–10 of 30 columns

Kaggle score(changing only VSpd for flight 31) = 0.11969

Validation Method

- Using Kaggle to check for improvement
- Only prediction flight ID = 31
- We will use the MSE from the new interpolation results as a baseline
- Replace the values of Flight 31 from the baseline, and check for improvement.
- In summary, if we see any improvement even small, then we use the method for the other flights

Findings from the data!

- Interpolation is great for other variables comparing to GndSpd & VSpd
- For using ML methods, we will need to create more variables
- Maybe more variables about the aircraft or the training task
- Some flights shorter than others.
- There were not any format for collecting the data in term of timing