Airplane crashes between Numbers and Regulations

Mohammad Saleh

***Abstract*—Airplane accidents, still the most dangerous life threatening accident. The main reason for this believe or fearness is the number of survivor across each accident. Nowadays, it is a fact that nobody will survive an airplane accident. The global aviation administrations, mainly FAA (Federal aviation administration) in USA, regularly release new rules to maintain and decrease the number of airplane crashes. Many people show the statistics for these crashes but the results shows that this number is slightly decreasing over the years because they didn't take into account the number of flights yearly. In this letter, we will present the latest analysis for the number of crashes per year globally and among the nations taking into account the increasing number in the flights yearly. We will highlight the main causes of the crashes and the proposed solutions for the fulfillment of the final project of the machine learning course.**

***Index Terms*—crashes, paper, analysis, airplane.**

I. Introduction

AIRPLANE is one the greatest human inventions. Nowadays, the technology for the new inventions in the aerospace field is growing fast, mainly in the drones technology. Lately, the FAA in the USA released a new regulations regarding drone operations to maintain the safety of the commercial airplanes. In this analysis, we wanted to check how effective the regulations and safety on the airplane crashes and the number of survivors. It is very well known that the FAA is one of the main administrations for airplanes safety. We will start with general overview of the crashes yearly and the main reasons and check the proportionally of airplane crashes number in the USA.

II. Analysis

A. Crashes per year

Many people have worked on this analysis in terms of showing the number of crashes yearly without taking into account the number of flights yearly. This analysis has been forked from previous one at the Kaggle website. Figure 1 shows the number of crashes yearly normalized by the number of passengers yearly as we don't have direct access to the number of flights yearly. The results show the number of crashes till 2009 because of the limitation of the dataset. The numbers on the y-axis are up to a constant value, the most important thing is the general trend in the number of crashes. It shows that the number of crashes decrease as times go, except couple years (1988, 1991) that had some reasons for having large number of crashes. In the following paragraphs, we will look in the causes of the crashes and the main reason for this decrease in the number of crashes.

B. Cause of crashes

Figure 2 shows the main reason for the crashes where the y-axis is in log-scale for comparison. The highest contribution is coming form the engine failure then bad weather and many different reasons shown in the figure. Principal component analysis (PCA) using the first two features scatter plot for the six different main causes shows that the engine failure is slightly correlated with other causes which is not surprising as engine failure is mostly from the engine mechanics. Engine failure is also not correlated with the FAA regulations and safety. To investigate more in the matter of the main cause of airplane causes, we looked for the number of crashes across the nations without normalizing these number of crashes. This investigation will show us the effect of FAA regulations on the number of crashes in the USA yearly.

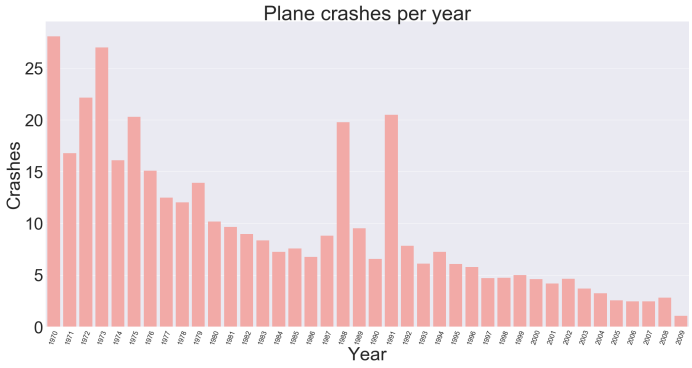

Fig. 1. Proportion of the number of crashes of airplanes (commercial and non-commercial) yearly from 1996 to 2004

C. Crashes in USA

Figure 3 shows the number for crashes yearly across different nations. The y-axis plotted in log-scale for comparison. It shows that the USA has the largest number of accidents followed by Russia, whereas the USA is one of the leaders in the aviation regulations and safety and the third world country doesn't show much contribution in the figure. The large number of accidents in the USA is directly correlated to the number of flights. It is clear that this large number is directly related to the number of flights and engine failures as the cause of the accidents. Moreover, the crashes are mainly from single-engine airplanes which doesn't have emergency plans for engine failure. Even for multi-engine airplanes, the emergency plans for passengers still not effective these days, but this raise more wide research in terms of what new designs of airplanes that can lead to larger number of survivors incase of airplane crashes, many proposal are done but they still far from reality, the difficulty in this subject is due to the high altitude of the airplane.

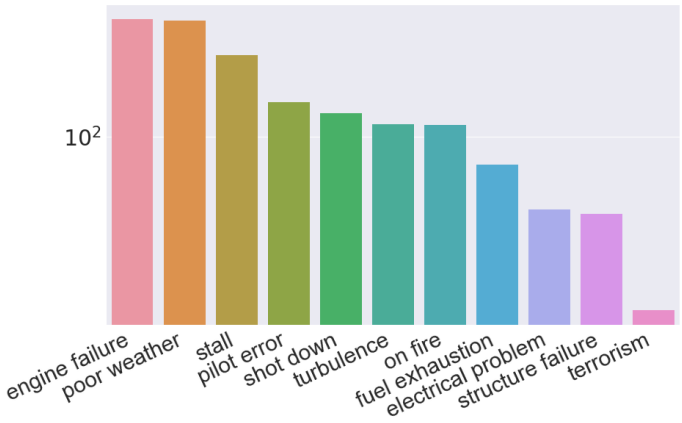

Fig. 2. THe main causes of airplane crashes

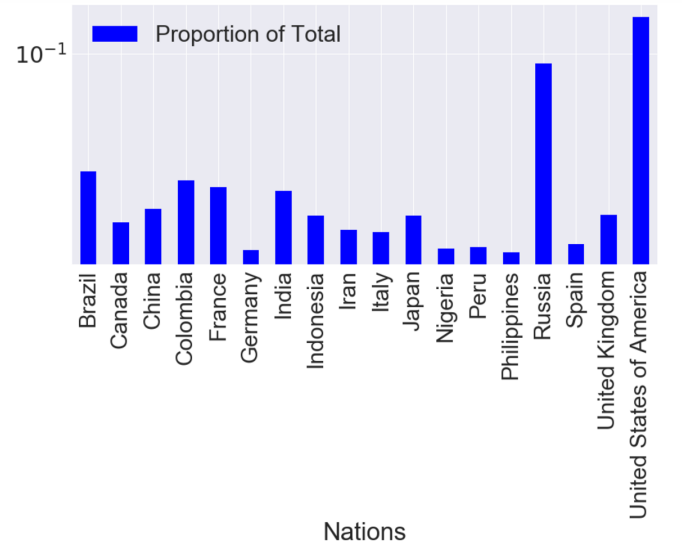

Fig. 3. The main causes of airplane crashes

III. Conclusion

We have shown the numbers of accidents and the main causes. The number of accidents are mainly from technology side (engine). The main cause lead to very small number of survivor. The best approach for increasing the number of survivors is also not effective due to the safety followed these days. To decrease the number of crashes and increase the number of survivor, the rules should be regarding airplanes manufacturing and emergency plans that can be followed by passengers other than the current ones. Lastly, to take you far from airplane fearness, the probability to die from car accident is much smaller than the probability to die from airplane accident.

Appendix A
Code

```
# coding: utf-8

# ##### The plan is to normalize these accidents by gathering datset
      for number of among nations yearly, these dataset are not
      directly avialable and we are working on gathering thses data
      from the  link provide later.


# In[1]:

import numpy as np
import pandas as pd
import matplotlib
from sklearn.linear_model import LinearRegression
from matplotlib import pyplot as plt
get_ipython().magic('matplotlib␣inline')
from collections import Counter
import seaborn as sns
import re
import operator


# In[2]:

frame=pd.read_csv('Airplane_Crashes_and_Fatalities_Since_1908_1.csv',sep=',')
flights=pd.read_csv('flights.csv') # my own dataset for
      normalization, more details will be later


# In[3]:

frame.head()
flights.head()


# In[4]:

frame['Date'] = pd.to_datetime(frame['Date'])
flights_year = flights['YearFlight']
flights_year_crashes = flights['Flights']


# In[5]:

frame['year'] = frame['Date'].dt.year
flights_year.head()
flights_year_norm=flights_year.tolist()
flights_crashes_norm=flights_year_crashes.tolist()
print(flights_crashes_norm)


# In[6]:

frame['Date'] = pd.to_datetime(frame['Date'])
frame['Day'] = frame['Date'].map(lambda x: x.day)
frame['Year'] = frame['Date'].map(lambda x: x.year)
frame['Month'] = frame['Date'].map(lambda x: x.month)


# In[7]:

crashes_per_year = Counter(frame['Year'])
#crashes_per_year = Counter(frame[frame['Year']>2000])
years = list(crashes_per_year.keys())
yearsnew=list()
crashes_year = list(crashes_per_year.values())
crashes_yearnew=list()
for i in range(len(years)):
        if (years[i]) > 1969:
          #  print(years[i])
            yearsnew.append(years[i])
            crashes_yearnew.append(crashes_year[i])


# In[8]:

def get_season(month):
    if month >= 3 and month <= 5:
        return 'spring'
    elif month >= 6 and month <= 8:
        return 'summer'
    elif month >= 9 and month <= 11:
        return 'autumn'
    else:
        return 'winter'

frame['Season'] = frame['Month'].apply(get_season)


# In[9]:

crashes_per_season = Counter(frame['Season'])
seasons = list(crashes_per_season.keys())
crashes_season = list(crashes_per_season.values())


# In[10]:

fig = plt.figure(figsize=(30, 30))

sub1 = fig.add_subplot(211)
sns.barplot(x=yearsnew, y=crashes_yearnew, color='r', ax=sub1)
sub1.set(ylabel="Crashes", xlabel="Year", title="Plane␣crashes␣per␣
      year")
plt.setp(sub1.patches, linewidth=0)
plt.setp(sub1.get_xticklabels(), rotation=70, fontsize=20)

sub2 = fig.add_subplot(223)
sns.barplot(x=seasons, y=crashes_season, color='g', ax=sub2)
```

```
texts = sub2.set(ylabel="Crashes", xlabel="Season", title="Plane␣
      crashes␣per␣season")

plt.tight_layout(w_pad=4, h_pad=1)


# The above plots shows the number of airplanes crashes per year and
      season wise. This plots many people have shown these. My main
      concern that thses plots as it is now, doesnot show much useful
      information unless it will be normalized by the number of
      flights yearly. Please take a look at the following plots for
      this mattter

# In[11]:

for i in range(len(yearsnew)):
    for j in range(len(flights_year_norm)):
        if yearsnew[i] == flights_year_norm[j]:
            crashes_yearnew[i]=
                crashes_yearnew[i]/flights_crashes_norm[i]
sns.set_color_codes("pastel")

fig = plt.figure(figsize=(30, 30))
sns.set(font_scale=3)
sub1 = fig.add_subplot(211)
sns.barplot(x=yearsnew, y=crashes_yearnew, color='r', ax=sub1)
sub1.set(ylabel="Crashes", xlabel="Year", title="Plane␣crashes␣per␣
      year")
plt.setp(sub1.patches, linewidth=0)
plt.setp(sub1.get_xticklabels(), rotation=70, fontsize=20)

#sub2 = fig.add_subplot(223)
#sns.barplot(x=seasons, y=crashes_season, color='g', ax=sub2)
#texts = sub2.set(ylabel="Crashes", xlabel="Season", title="Plane
      crashes per season")

plt.tight_layout(w_pad=4, h_pad=1)


# The above plots, mainly the crashes per year plot, shows the
      number of crashes per year nomalized by the number of
      passengers yearly as an indication of the number of flight
      yearly, the number of passenger yearly was obtained from this
      link
      http://data.worldbank.org/indicator/IS.AIR.PSGR?end=2015&name_desc=true&start=1970&type=shaded
# The results are from 11970 only as the number of passengers yearly
      dataset is only from 1970 and later

# In[12]:

failures = {
    'pilot␣error': '(pilot|crew)␣(error|fatigue)',
    'engine␣failure': 'engine.*(fire|fail)',
    'structure␣failure': '(structural␣fail)|(fuel␣leak)|(langing␣
         gear)',
    'electrical␣problem': 'electrical',
    'poor␣weather':
        '((poor|bad).*(weather|visibility)|thunderstorm)',
    'stall': 'stall',
    'on␣fire': '(caught␣fire)|(caught␣on␣fire)',
    'turbulence': 'turbulence',
    'fuel␣exhaustion': '(out␣of␣fuel)|(fuel.*exhaust)',
    'terrorism': 'terrorist|terrorism',
    'shot␣down': 'shot␣down',
}

failure_counts = {'other':0}

for s in frame.Summary.dropna():
    other = True
    for failure, exp in failures.items():
        if re.search(exp, s.lower()):
            other = False
            if failure in failure_counts:
                failure_counts[failure] += 1
            else:
                failure_counts[failure] = 1
    if other:
        failure_counts['other'] += 1

nan_counts = len(frame.Summary.isnull())

del failure_counts['other']

sortedcauses = sorted(failure_counts.items(),
      key=operator.itemgetter(1), reverse=True)
for k, v in sortedcauses:
    print(k, v)

plt.figure(figsize=(14, 8))
x, y = zip(*sortedcauses)
sns.barplot(x=x, y=y)
plt.xticks(rotation=25, horizontalalignment='right')
plt.yscale('log')
plt.show()


# In[13]:

#Splitting out the country from the location to see if we can find
      some interesting
#statistics about where the most crashes have taken place.
operator =
      frame[['Operator','Fatalities']].groupby('Operator').agg(['sum','count'])
fatalities =
      operator['Fatalities','sum'].sort_values(ascending=False)
#fig_ops,((ax1, ax2), (ax3, ax4))=plt.subplots(2,2,sharex=True)
#fatalities.head(10).plot(kind='bar',title='Fatalities by
      Operator',ax=ax3,grid=True,rot=90)
totalfatal = fatalities.sum()
s = frame['Location'].str[0:].str.split(',', expand=True)
frame['Country'] = s[3].fillna(s[2]).fillna(s[1]).str.strip()
#I've pulled out all the US states so as to be able to assign them a
      country

usNames = ['Virginia','New␣Jersey','Ohio','Pennsylvania',
      'Maryland', 'Indiana', 'Iowa',
          'Illinois','Wyoming', 'Minnisota', 'Wisconsin', 'Nevada',
               'NY','California',
          'WY','New␣York','Oregon', 'Idaho',
               'Connecticut','Nebraska', 'Minnesota', 'Kansas',
          'Texas', 'Tennessee', 'West␣Virginia', 'New␣Mexico',
               'Washington', 'Massachusetts',
          'Utah', 'Ilinois','Florida', 'Michigan',
               'Arkansas','Colorado', 'Georgia''Missouri',
          'Montana', 'Mississippi','Alaska','Jersey', 'Cailifornia',
               'Oklahoma','North␣Carolina',
          'Kentucky','Delaware','D.C.','Arazona','Arizona','South␣
               Dekota','New␣Hampshire','Hawaii',
          'Washingon','Massachusett','Washington␣
               DC','Tennesee','Deleware','Louisiana',
          'Massachutes', 'Louisana', 'New␣York␣
               (Idlewild)','Oklohoma','North␣Dakota','Rhode␣Island',
          'Maine','Alakska','Wisconson','Califlornia','Virginia','Virginia.','CA','V
          'HI','AK','IN','GA','Coloado','Airzona','Alabama','Alaksa'
          ]

#Decided to try and cleanse the country names.
afNames = ['Afghanstan'] #Afghanistan
anNames = ['off␣Angola'] #Angola
ausNames = ['Qld.␣Australia','Queensland␣␣Australia','Tasmania','off␣
      Australia'] #Australia
argNames = ['Aregntina'] #Argentina
azNames = ['Azores␣(Portugal)'] #Azores
baNames = ['Baangladesh'] #Bangladesh
bahNames = ['Great␣Inagua'] #Bahamas
berNames = ['off␣Bermuda'] #Bermuda
bolNames = ['Boliva','BO'] #Bolivia
bhNames = ['Bosnia-Herzegovina'] #Bosnia Herzegovina
bulNames = ['Bugaria','Bulgeria'] #Bulgaria
canNames = ['British␣Columbia', 'British␣Columbia␣Canada','Canada2',
           'Saskatchewan','Yukon␣Territory'] #Canada
camNames = ['Cameroons','French␣Cameroons'] #Cameroon
caNames = ['Cape␣Verde␣Islands'] #Cape Verde
chNames = ['Chili'] #Chile
coNames = ['Comoro␣Islands', 'Comoros␣Islands'] #Comoros
djNames = ['Djbouti','Republiof␣Djibouti'] #Djibouti
domNames = ['Domincan␣Republic', 'Dominica'] #Dominican Republic
drcNames = ['Belgian␣Congo','Belgian␣Congo␣(Zaire)','Belgium␣Congo'
           'DRCongo','DemocratiRepubliCogo','DemocratiRepubliCongo',
            'DemocratiRepubliof␣
                 Congo','DemoctratiRepubliCongo','Zaire',
           'Za re'] #Democratic Republic of Congo
faNames = ['French␣Equitorial␣Africa'] #French Equatorial Africa
gerNames = ['East␣Germany','West␣Germany'] #Germany
grNames = ['Crete'] #Greece
haNames = ['Hati'] #Haiti
hunNames = ['Hunary'] #Hungary
inNames = ['Indian'] #India
indNames = ['Inodnesia','Netherlands␣Indies'] #Indonesia
jamNames = ['Jamacia'] #Jamaica
malNames = ['Malaya'] #Malaysia
manNames = ['Manmar'] #Myanmar
marNames = ['Mauretania'] #Mauritania
morNames = ['Morrocco','Morroco'] #Morocco
nedNames = ['Amsterdam','The␣Netherlands'] #Netherlands
niNames = ['Niger'] #Nigeria
philNames = ['Philipines','Philippine␣Sea', 'Phillipines',
           'off␣the␣Philippine␣island␣of␣Elalat'] #Philippines
romNames = ['Romainia'] #Romania
rusNames = ['Russian','Soviet␣Union','USSR'] #Russia
saNames = ['Saint␣Lucia␣Island'] #Saint Lucia
samNames = ['Western␣Samoa'] #Samoa
siNames = ['Sierre␣Leone'] #Sierra Leone
soNames = ['South␣Africa␣(Namibia)'] #South Africa
surNames = ['Suriname'] #Surinam
uaeNames = ['United␣Arab␣Emirates'] #UAE
ukNames = ['England', 'UK','Wales','110␣miles␣West␣of␣Ireland']
      #United Kingdom
uvNames = ['US␣Virgin␣Islands','Virgin␣Islands'] #U.S. Virgin Islands
wkNames = ['325␣miles␣east␣of␣Wake␣Island']#Wake Island
yuNames = ['Yugosalvia'] #Yugoslavia
zimNames = ['Rhodesia', 'Rhodesia␣(Zimbabwe)'] #Zimbabwe

clnames = []
for country in frame['Country'].values:
    if country in afNames:
        clnames.append('Afghanistan')
    elif country in anNames:
        clnames.append('Angola')
    elif country in ausNames:
        clnames.append('Australia')
    elif country in argNames:
        clnames.append('Argentina')
    elif country in azNames:
        clnames.append('Azores')
    elif country in baNames:
        clnames.append('Bangladesh')
    elif country in bahNames:
        clnames.append('Bahamas')
    elif country in berNames:
        clnames.append('Bermuda')
    elif country in bolNames:
        clnames.append('Bolivia')
    elif country in bhNames:
        clnames.append('Bosnia␣Herzegovina')
    elif country in bulNames:
        clnames.append('Bulgaria')
    elif country in canNames:
        clnames.append('Canada')
    elif country in camNames:
        clnames.append('Cameroon')
    elif country in caNames:
        clnames.append('Cape␣Verde')
    elif country in chNames:
        clnames.append('Chile')
    elif country in coNames:
        clnames.append('Comoros')
    elif country in djNames:
        clnames.append('Djibouti')
```

```
    elif country in domNames:
        clnames.append('Dominican Republic')
    elif country in drcNames:
        clnames.append('Democratic Republic of Congo')
    elif country in faNames:
        clnames.append('French Equatorial Africa')
    elif country in gerNames:
        clnames.append('Germany')
    elif country in grNames:
        clnames.append('Greece')
    elif country in haNames:
        clnames.append('Haiti')
    elif country in hunNames:
        clnames.append('Hungary')
    elif country in inNames:
        clnames.append('India')
    elif country in jamNames:
        clnames.append('Jamaica')
    elif country in malNames:
        clnames.append('Malaysia')
    elif country in manNames:
        clnames.append('Myanmar')
    elif country in marNames:
        clnames.append('Mauritania')
    elif country in morNames:
        clnames.append('Morocco')
    elif country in nedNames:
        clnames.append('Netherlands')
    elif country in niNames:
        clnames.append('Nigeria')
    elif country in philNames:
        clnames.append('Philippines')
    elif country in romNames:
        clnames.append('Romania')
    elif country in rusNames:
        clnames.append('Russia')
    elif country in saNames:
        clnames.append('Saint Lucia')
    elif country in samNames:
        clnames.append('Samoa')
    elif country in siNames:
        clnames.append('Sierra Leone')
    elif country in soNames:
        clnames.append('South Africa')
    elif country in surNames:
        clnames.append('Surinam')
    elif country in uaeNames:
        clnames.append('UAE')
    elif country in ukNames:
        clnames.append('United Kingdom')
    elif country in usNames:
        clnames.append('United States of America')
    elif country in uvNames:
        clnames.append('U.S. Virgin Islands')
    elif country in wkNames:
        clnames.append('Wake Island')
    elif country in yuNames:
        clnames.append('Yugoslavia')
    elif country in zimNames:
        clnames.append('Zimbabwe')
    else:
        clnames.append(country)

plt.figure(figsize=(14, 8))
frame['Nations'] = clnames
fatalcountries =
     frame[['Fatalities','Nations']].groupby(['Nations']).agg('sum')
fatalcountries.reset_index(inplace = 'True')
fatalcountries['Proportion of Total'] =
     fatalcountries['Fatalities']/totalfatal

fig_c, (ax1,ax2) = plt.subplots(2,1,sharex = True)

fatalcountries[fatalcountries['Fatalities']>1300].plot(x = 'Nations'
                                                       , y =
                                                            'Fatalities'
                                                       , ax = ax1
                                                       , kind = 'bar'
                                                       , grid = True,
                                                       colormap='winter',
                                                      logy=True,
                                                        figsize=(14,14),
                                                        sort_columns=True)

fatalcountries[fatalcountries['Fatalities']>1300].plot(x = 'Nations'
                                                       , y =
                                                            'Proportion
                                                            of Total'
                                                       , ax = ax2
                                                       , kind = 'bar'
                                                       , grid = True,
                                                        colormap='winter',
                                                         logy=True)


# These two plots was also shown before, what we will be doing next,
     is taking the largest two location for accidents, Russia and
     United State of America and gather dataset to normalize these
     by the number of flights in these two countires.

# ########### The plan is to normalize these accidents by gathering
     datset for number of among nations yearly, these dataset are
     not directly avialable and we are working on gathering thses
     data from the above link provide.
```

REFERENCES

[1] https://www.kaggle.com/saurograndi/airplane-crashes-since-1908.

[2] http://data.worldbank.org/indicator/IS.AIR.PSGR?end=2015