Final_Project

Venkata Harsha

2024-03-03

```
#install.packages("tidyverse")
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
#install.packages("gridExtra")
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
#install.packages("readr")
library(readr)
#install.packages("ggmap")
library(ggmap)
```

```
## i Google's Terms of Service: <https://mapsplatform.google.com>
##   Stadia Maps' Terms of Service: <https://stadiamaps.com/terms-of-service/>
##   OpenStreetMap's Tile Usage Policy: <https://operations.osmfoundation.org/policies/tiles/>
## i Please cite ggmap if you use it! Use `citation("ggmap")` for details.
```

```
#install.packages("rworldmap")
library(rworldmap)
```

```
## Loading required package: sp
## ### Welcome to rworldmap ###
## For a short introduction type :   vignette('rworldmap')
```

```r
#install.packages("arules")
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
##
## Attaching package: 'arules'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```r
#install.packages("arulesViz")
library(arulesViz)
```

```r
#install.packages("ggpubr")
library(ggpubr)
```

```r
#install.packages("car")
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:arules':
##
##     recode
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```r
#install.packages("caret")
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
#install.packages("forecast")
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:ggpubr':
##
##     gghistogram
```

```r
#install.packages("zoo")
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
#install.packages("ggfortify")
library(ggfortify)
```

```
## Registered S3 methods overwritten by 'ggfortify':
##   method                 from
##   autoplot.Arima         forecast
##   autoplot.acf           forecast
##   autoplot.ar            forecast
##   autoplot.bats          forecast
##   autoplot.decomposed.ts forecast
##   autoplot.ets           forecast
##   autoplot.forecast      forecast
##   autoplot.stl           forecast
##   autoplot.ts            forecast
##   fitted.ar              forecast
##   fortify.ts             forecast
##   residuals.ar           forecast
```

##0.1 Database information

We're examining the Global Terrorism Database (GTD), an open-source dataset covering terrorist incidents worldwide from 1970 to 2016, comprising over 170,000 cases. (https://www.kaggle.com/START-UMD/gtd) The database is maintained and updated by researchers at the National Consortium for the Study of Terrorism and Responses to Terrorism (START) at the University of Maryland. For any unclear variable meanings, a quick definition will be provided in this report http://start.umd.edu/gtd/downloads/Codebook.pdf. Additionally, we'll utilize the United Nations population forecasts dataset to analyze terrorism trends alongside population metrics (https://esa.un.org/unpd/wpp/Download/Standard/Population/). Notably, the terrorism data has one year missing, which we've chosen to disregard as it doesn't affect the overall analysis.

Our analysis will be guided by specific research questions, with data manipulation conducted prior to addressing each question in the corresponding code section.

We'll begin by importing the data, reducing its size, and renaming variables for clarity.

##0.2 Importing dataset

```
fulldf <- read_csv("/Users/venkataharshapedada/Downloads/globalterrorismdb_0718dist.csv")
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
## Rows: 181691 Columns: 135
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (55): approxdate, resolution, country_txt, region_txt, provstate, city, ...
## dbl (75): eventid, iyear, imonth, iday, extended, country, region, latitude,...
## lgl  (5): gsubname3, weaptype4, weaptype4_txt, weapsubtype4, weapsubtype4_txt
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#variable_names <- names(fulldf)
```

```
#variable_names
#str(fulldf)
```

```
fullpop <- read_csv("/Users/venkataharshapedada/Downloads/WPP2022_TotalPopulationBySex.csv")
```

```
## Rows: 550866 Columns: 18
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (6): Notes, ISO3_code, ISO2_code, LocTypeName, Location, Variant
## dbl (12): SortOrder, LocID, SDMX_code, LocTypeID, ParentID, VarID, Time, Mid...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
column_names <- names(fullpop)
column_names
```

```
##  [1] "SortOrder"   "LocID"       "Notes"       "ISO3_code"   "ISO2_code"
##  [6] "SDMX_code"   "LocTypeID"   "LocTypeName" "ParentID"    "Location"
## [11] "VarID"       "Variant"     "Time"        "MidPeriod"   "PopMale"
## [16] "PopFemale"   "PopTotal"    "PopDensity"
```

```r
#str(fullpop)
```

```r
library(dplyr)
```

```r
pop <- fullpop %>%
  select(-MidPeriod, -PopMale, -PopFemale, -VarID)

pop <- pop %>%
  filter(Time > 1969 & Variant == 'Medium' & Time < 2017) %>%
  select(-Variant, -LocID)

futurepop <- fullpop %>%
  filter(Time >2016 & Variant == 'Medium') %>%
  select(-Variant, -LocID, -MidPeriod, -PopMale, -PopFemale, -VarID)
```

##0.3 Data Cleaning

We'll streamline our analysis by eliminating unnecessary variables and renaming others for clarity. We'll concentrate on a subset of variables to ensure the clarity of our analysis.

###0.3.1 Variables of importance

1. iyear
2. imonth
3. iday
4. country_txt
5. region_txt
6. city
7. latitude
8. longitude
9. summary - event summary, what happened? when etc.
10. multiple - was the attack part of a multiple attack event?
11. attacktype1_txt
12. targtype1_txt
13. targsubtype1_txt
14. gname - perpetrator group name
15. weaptype1_txt
16. nkill - confirmed fatalities of event
17. nwound - number of non-fatal wounded of event
18. nkillter - fatalities of perpetrator(s)

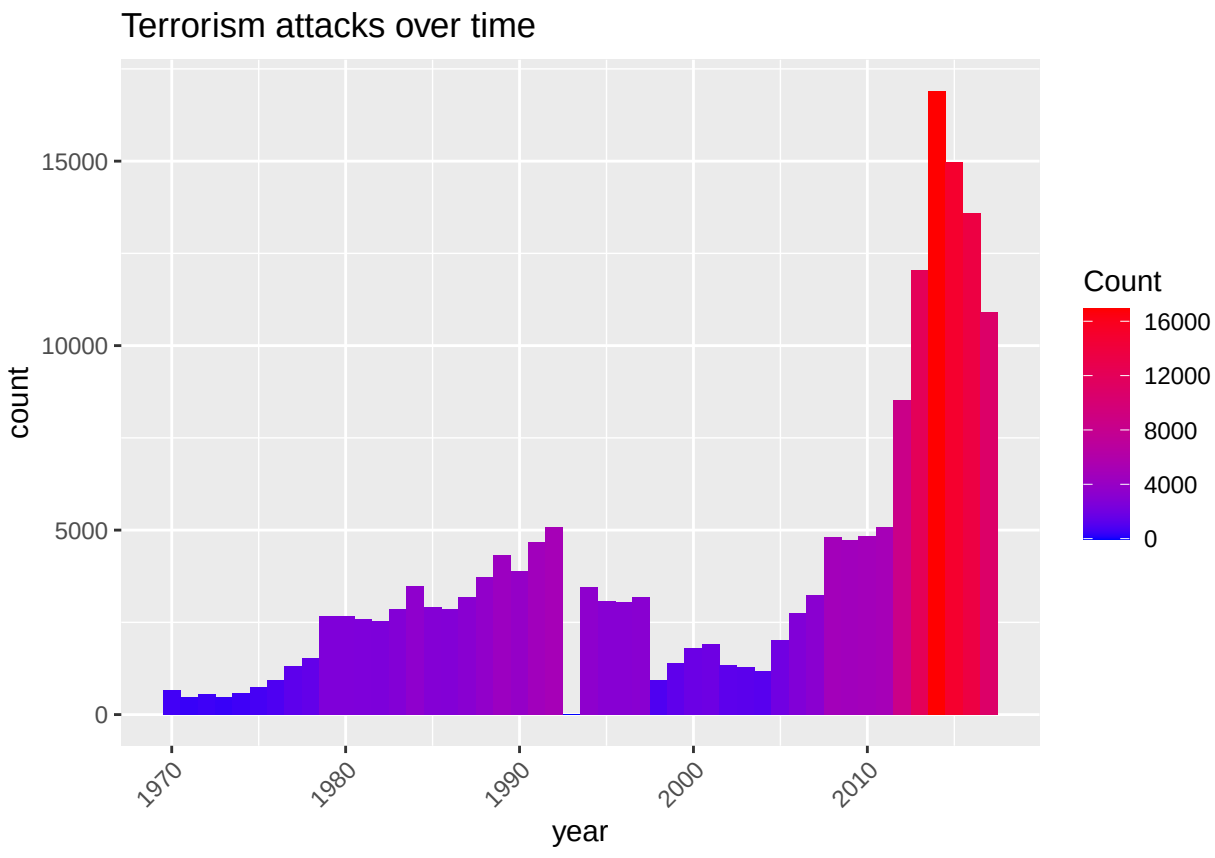
###0.3.2 selecting vars of importance and renaming

```r
df <- fulldf %>%
  select(iyear, imonth, iday, country_txt, region_txt, city, latitude, longitude, summary, multiple, at

df <- df %>%
  rename(year = iyear, month = imonth, day = iday, country = country_txt, region = region_txt, multiple

df <- df %>%
  mutate(decade =
           ifelse(year<1980, '70s',
                  ifelse(year < 1990, '80s',
                         ifelse(year < 2000, '90s',
                                ifelse( year < 2010, '2000s', '2010s')))))

df$decade <- factor(df$decade, levels=c("70s", "80s", "90s", "2000s", "2010s"))
```

1. Data Overview

##1.1 Number of Terrorist Attacks

```r
ggplot(data = df, aes(x = year, fill = after_stat(count))) +
  geom_histogram(binwidth = 1) +  # Adjust binwidth as needed
  scale_fill_gradient(low = "blue", high = "red") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = 'Terrorism attacks over time', fill = "Count")
```



There have been over 170,000 recorded terrorist attacks, and it appears that their frequency has increased over time.

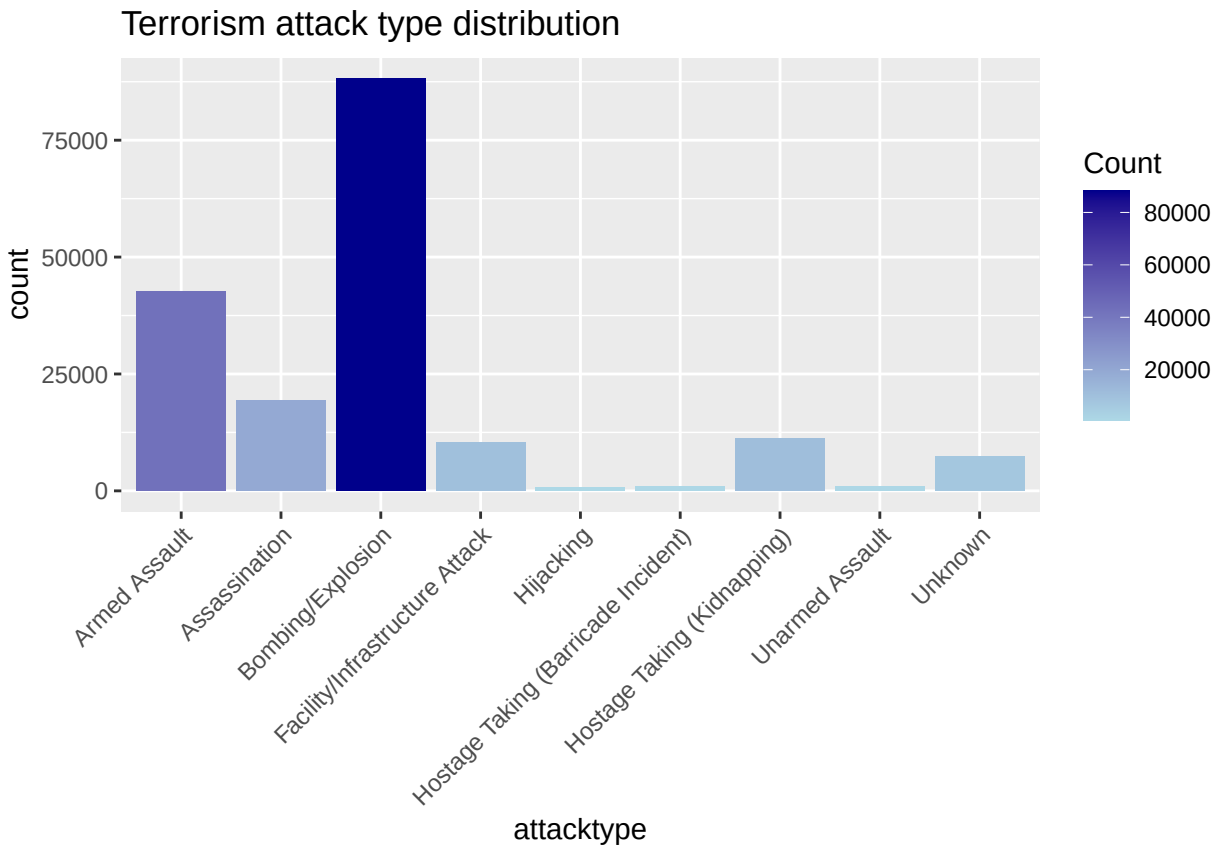
```
df %>%
  summarise(nr_of_attacks = n())
```

```
## # A tibble: 1 x 1
##   nr_of_attacks
##           <int>
## 1        181691
```

##1.2 Attack type Distribution

```
ggplot(data = df, aes(x = attacktype, fill = after_stat(count))) +
  geom_histogram(stat = "count") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +  # Change the color gradient as needed
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = 'Terrorism attack type distribution', fill = "Count")
```

```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## `binwidth`, `bins`, and `pad`
```



Bombings account for more than 80,000 incidents, making them the most prevalent type of attack. The second largest category is armed assault, with approximately 40,000 attacks.

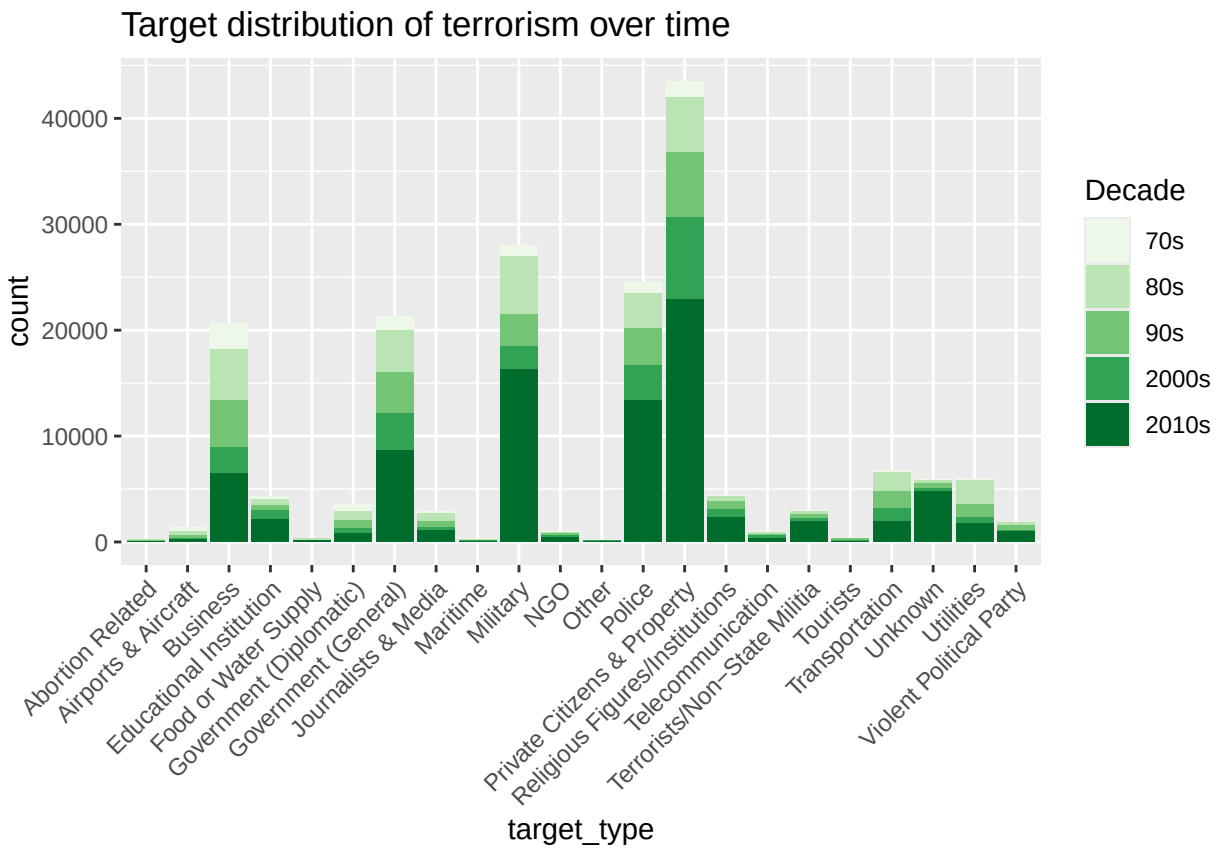
##1.3 Target Distribution

Let's get an idea of what kind of targets terrorists hit.

```
ggplot(data = df, aes(x = target_type, fill = decade)) +
  geom_histogram(stat = 'count') +
  scale_fill_brewer(palette = "Set4") +  # Change the color palette as needed
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = 'Target distribution of terrorism over time', fill = 'Decade')
```

```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## `binwidth`, `bins`, and `pad`
```

```
## Warning: Unknown palette: "Set4"
```




```
df %>%
  group_by(target_type) %>%
  summarise(nr_of_attacks = n()) %>%
  arrange(desc(nr_of_attacks)) %>%
  head(n=10)
```

```
## # A tibble: 10 x 2
##    target_type                  nr_of_attacks
##    <chr>                                <int>
##  1 Private Citizens & Property          43511
```

```
##  2 Military                              27984
##  3 Police                                24506
##  4 Government (General)                  21283
##  5 Business                              20669
##  6 Transportation                         6799
##  7 Utilities                              6023
##  8 Unknown                                5898
##  9 Religious Figures/Institutions         4440
## 10 Educational Institution                4322
```

It appears that private citizens have become increasingly targeted. We'll delve deeper into this trend in question 2.6 for a more thorough analysis.

#1.4 location of terrorism (region/country/city?)

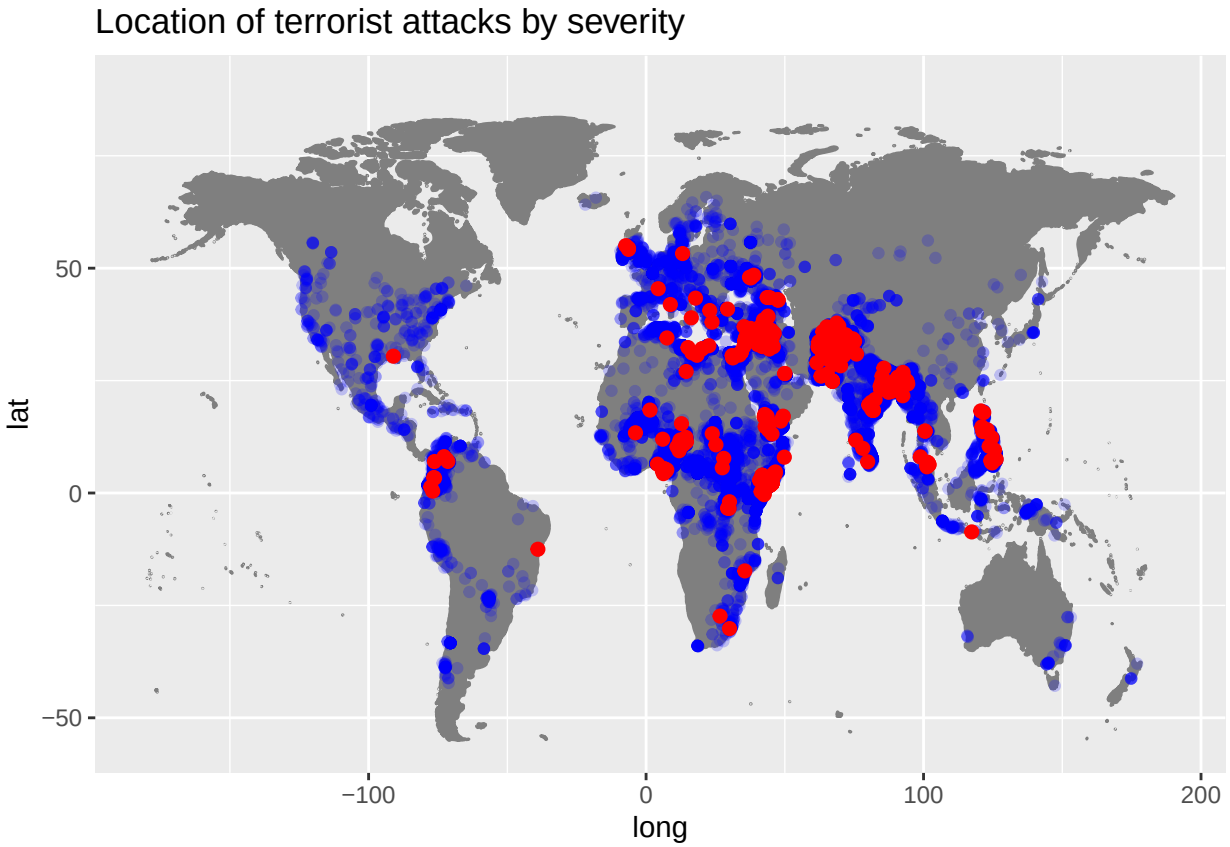
We aim to visualize the geographic distribution of terrorist attacks worldwide. To achieve this, we'll utilize the ggmap package.

```
df2000 <- df %>%
  filter(year > 2006)

world <- borders("world", colour="gray50", fill="gray50")
```

```
worldmap <- ggplot() + world + scale_y_continuous(limits=c(-55, 90))

worldmap +
  geom_point(aes(x=df2000$longitude[df$nkill<51], y=df2000$latitude[df$nkill<51]), col='blue', alpha= 0
  geom_point(aes(x=df2000$longitude[df$nkill>50], y=df2000$latitude[df$nkill>50]), col='red', size=2) +
  labs(title='Location of terrorist attacks by severity')
```

```
## Warning: Removed 88925 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

```
## Warning: Removed 10602 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Red dots will represent locations where terrorist attacks resulted in more than 50 deaths, while blue dots will indicate locations with fewer than 51 deaths.

Additionally, we'll examine the top 10 locations for terrorist attacks, categorized by region, country, and city.

```
df %>%
  group_by(region) %>%
  summarise( nr_of_attacks = n()) %>%
  arrange(desc(nr_of_attacks)) %>%
  head(n=10)
```

```
## # A tibble: 10 x 2
##    region                       nr_of_attacks
##    <chr>                                <int>
##  1 Middle East & North Africa           50474
##  2 South Asia                           44974
##  3 South America                        18978
##  4 Sub-Saharan Africa                   17550
##  5 Western Europe                       16639
##  6 Southeast Asia                       12485
##  7 Central America & Caribbean          10344
##  8 Eastern Europe                        5144
##  9 North America                         3456
## 10 East Asia                              802
```

```r
df %>%
  group_by(country) %>%
  summarise( nr_of_attacks = n()) %>%
  arrange(desc(nr_of_attacks)) %>%
  head(n=10)
```

```
## # A tibble: 10 x 2
##    country        nr_of_attacks
##    <chr>                  <int>
##  1 Iraq                   24636
##  2 Pakistan               14368
##  3 Afghanistan            12731
##  4 India                  11960
##  5 Colombia                8306
##  6 Philippines             6908
##  7 Peru                    6096
##  8 El Salvador             5320
##  9 United Kingdom          5235
## 10 Turkey                  4292
```

```r
df %>%
  filter(city != 'Unknown') %>%
  group_by(city) %>%
  summarise( nr_of_attacks = n()) %>%
  arrange(desc(nr_of_attacks)) %>%
  head(n=10)
```

```
## # A tibble: 10 x 2
##    city         nr_of_attacks
##    <chr>                <int>
##  1 Baghdad               7589
##  2 Karachi               2652
##  3 Lima                  2359
##  4 Mosul                 2265
##  5 Belfast               2171
##  6 Santiago              1621
##  7 Mogadishu             1581
##  8 San Salvador          1558
##  9 Istanbul              1048
## 10 Athens                1019
```

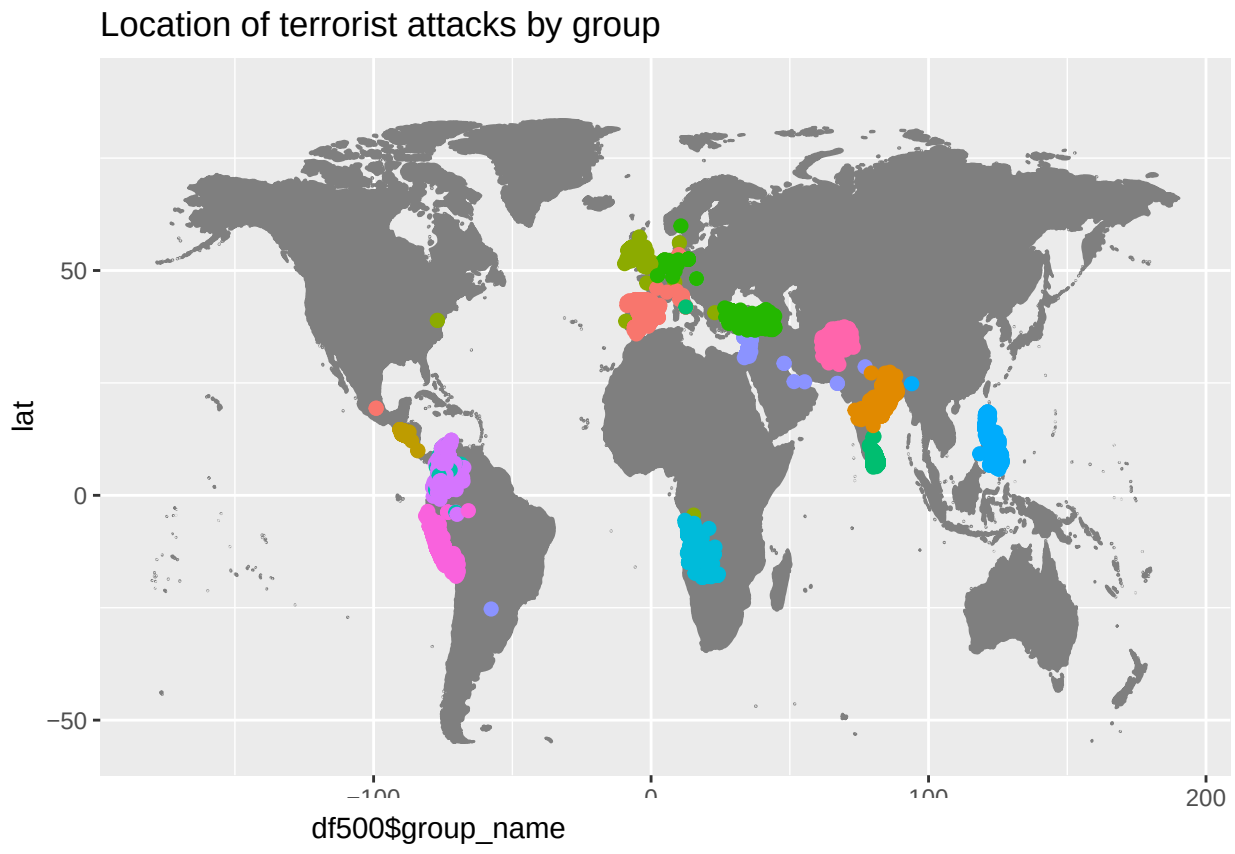
We'll start by reducing the dataset's size by grouping it by decade and selecting only the top 10,000 points for each decade. Additionally, to improve visibility, we'll further filter the dataset to include only those points with more than 300 attacks. Subsequently, we'll examine the distribution of terrorist groups worldwide.

```r
df500 <- df %>%
  select(decade, latitude, longitude, group_name) %>%
  group_by(decade) %>%
  slice(1:10000)

df500 <- df500 %>%
  group_by(group_name) %>%
```

```
  filter(n() >= 300 & group_name != "Unknown")

worldmap +
  geom_point(aes(x=df500$longitude, y=df500$latitude, col=df500$group_name), size=2, position = 'jitter
  labs(title='Location of terrorist attacks by group') +
  theme(legend.position=c(0.5, -0.5))
```

```
## Warning: A numeric ‘legend.position‘ argument in ‘theme()‘ was deprecated in ggplot2
## 3.5.0.
## i Please use the ‘legend.position.inside‘ argument of ‘theme()‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Warning: Removed 585 rows containing missing values or values outside the scale range
## (‘geom_point()‘).
```



The geographical spread is quite evident, especially concerning well-known terrorist groups.

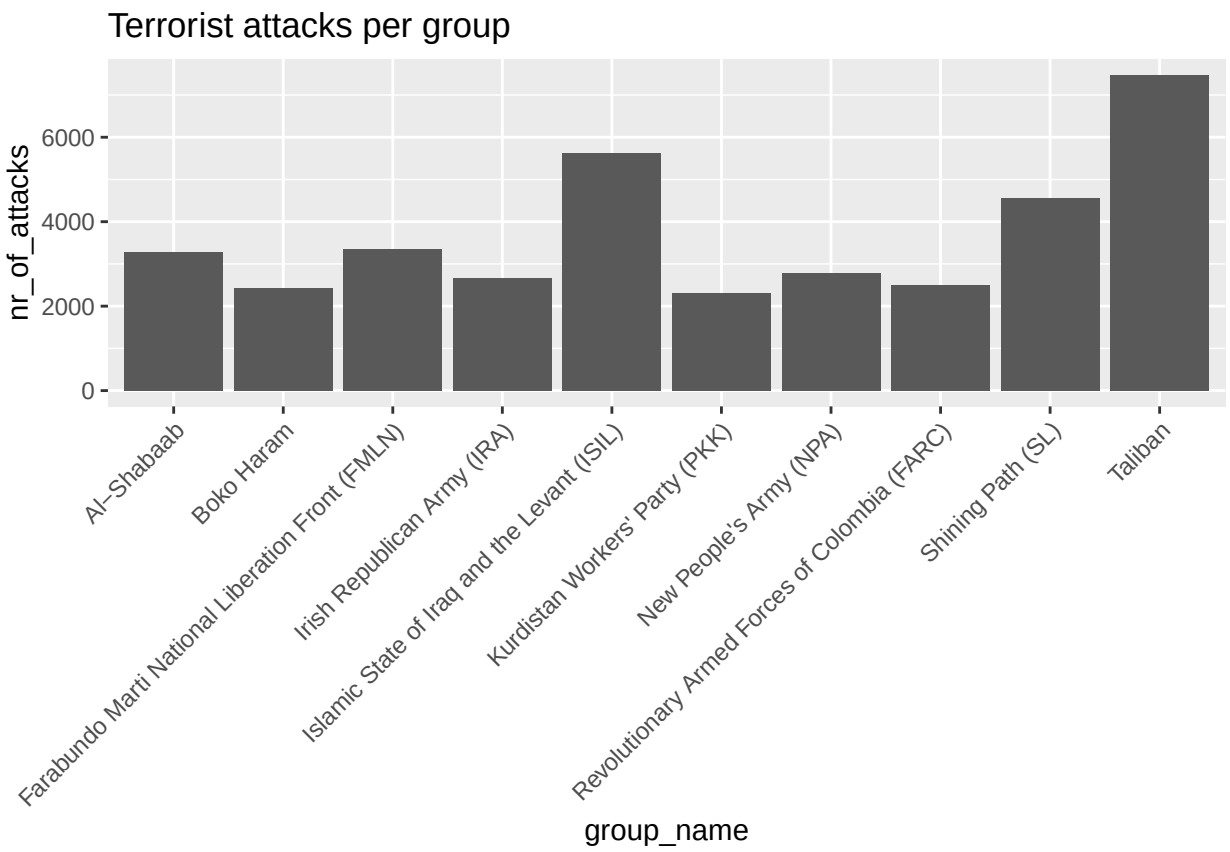
#1.5 distribution of terrorist groups

who are doing these attacks

```
#table
top10_groups <- df %>%
  filter(group_name != "Unknown") %>%
```

```
  group_by(group_name) %>%
  summarise(nr_of_attacks = n()) %>%
  arrange(desc(nr_of_attacks)) %>%
  head(n=10)

#visual
ggplot(data=top10_groups) +
  stat_summary(aes(x=group_name, y=nr_of_attacks), geom="bar") +
  theme(axis.text.x= element_text(angle=45, hjust=1)) +
  labs(title='Terrorist attacks per group')
```

```
## No summary function supplied, defaulting to 'mean_se()'
```



#2. Trends in terorrism

##2.1 Has terrorism gone up?

###2.1.1 Terrorism growth

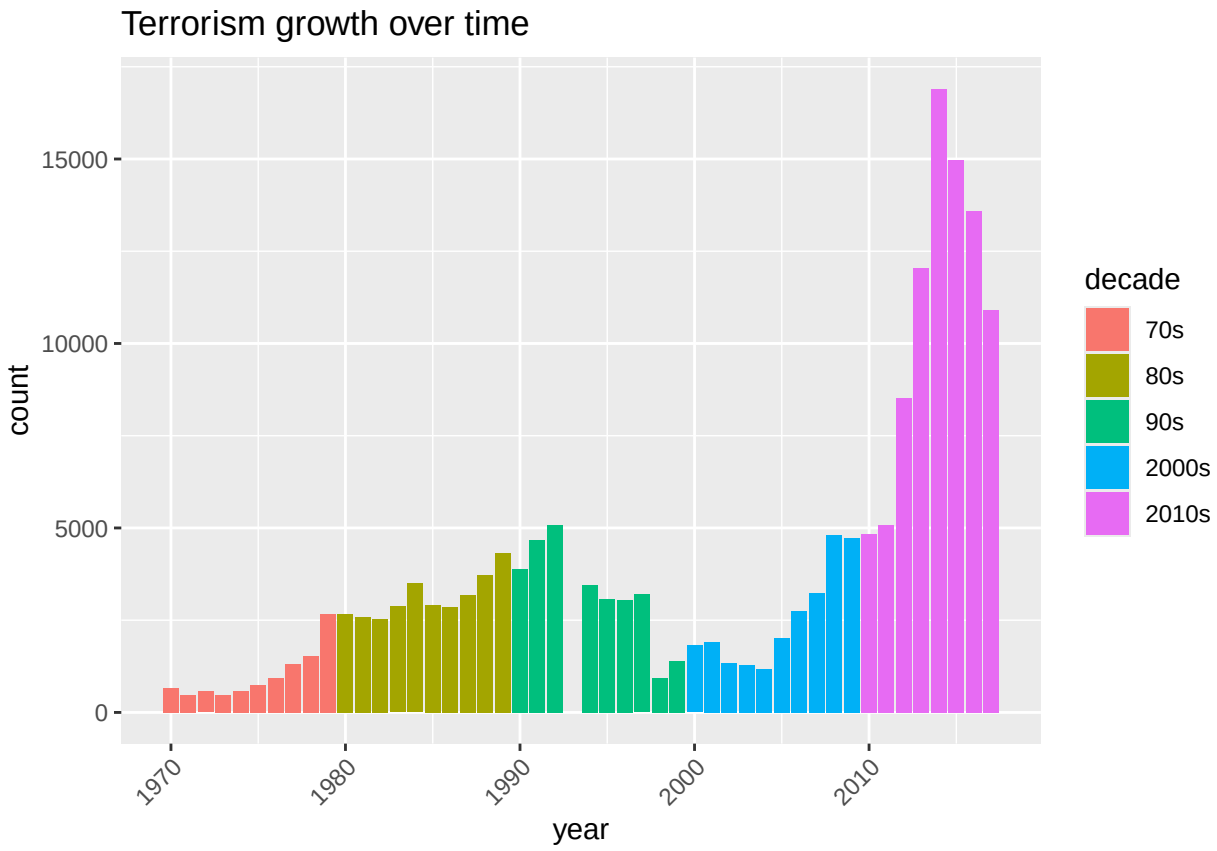
see below for decade breakdown - significant increase since 2010

```
df %>%
  group_by(decade) %>%
  summarise(nr_of_attacks = n()) %>%
  arrange(desc(nr_of_attacks)) %>%head(n=10)
```

```
## # A tibble: 5 x 2
##   decade nr_of_attacks
##   <fct>          <int>
## 1 2010s          86815
## 2 80s            31160
## 3 90s            28762
## 4 2000s          25040
## 5 70s             9914
```

```
#visual
ggplot(data=df, aes(x=year, fill=decade)) +
  geom_histogram(stat='count') +
  theme(axis.text.x= element_text(angle=45, hjust=1)) +
  labs(title='Terrorism growth over time')
```

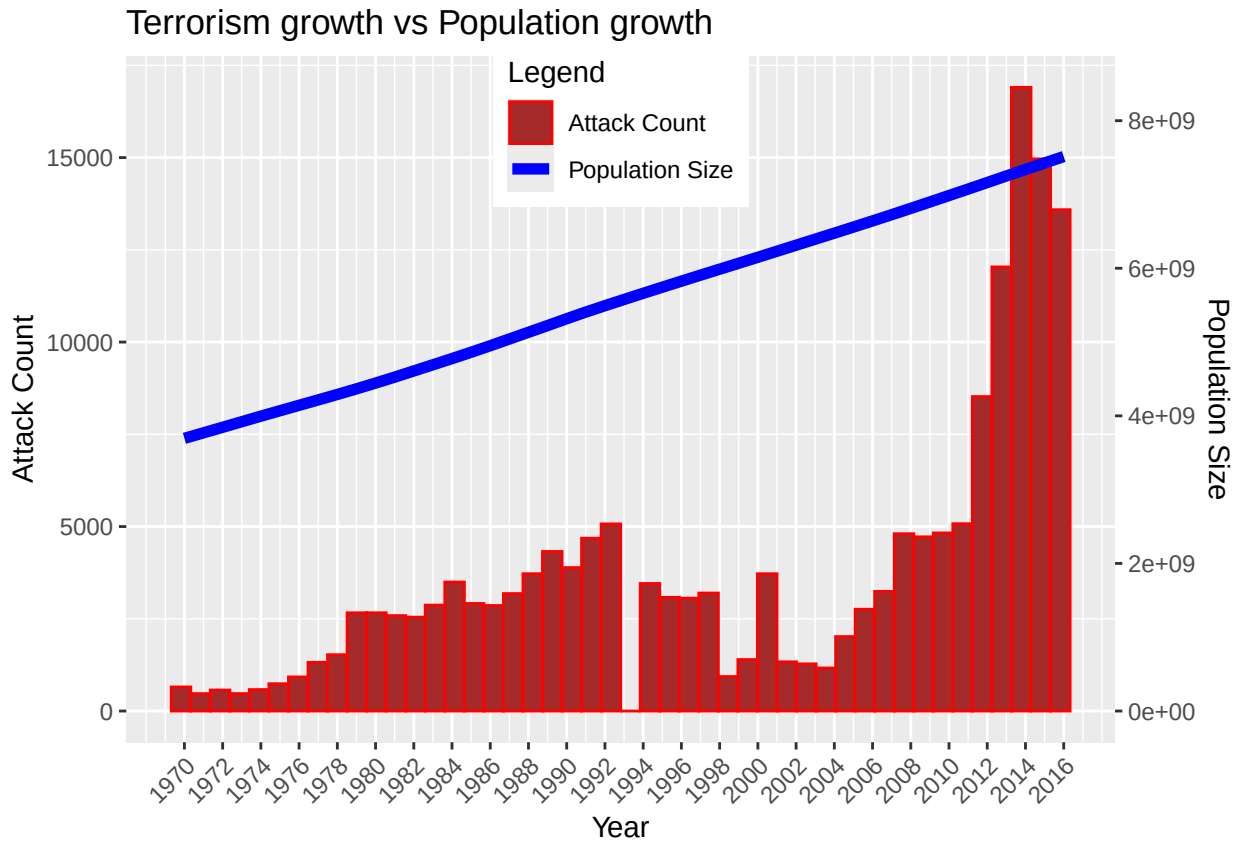
```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## `binwidth`, `bins`, and `pad`
```



##2.1.2 Take into account world population growth

```
#get just the world population by year
popworld <- pop %>%
  filter(Location == "World") %>%
  select(-Location)
```

```r
# Join to the dataframe based on year.
df2 <- inner_join(df, popworld, by = c("year" = "Time"))

# Plot
p1 <- ggplot(data = df2, aes(x = year)) +
  geom_histogram(aes(col = 'Attack Count'), bins = 46, fill = "brown") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_x_continuous(breaks = seq(1970, 2016, 2))

p1 +
  geom_line(aes(y = PopTotal / 500, col = 'Population Size'), linewidth = 2) +
  scale_y_continuous(sec.axis = sec_axis(~ . * 500000, name = "Population Size")) +
  labs(y = "Attack Count", x = "Year", color = "Legend") +
  theme(legend.position = c(0.5, 0.9)) +
  scale_color_manual(values = c("red", "blue")) +  # Change line colors as needed
  labs(title = 'Terrorism growth vs Population growth')
```

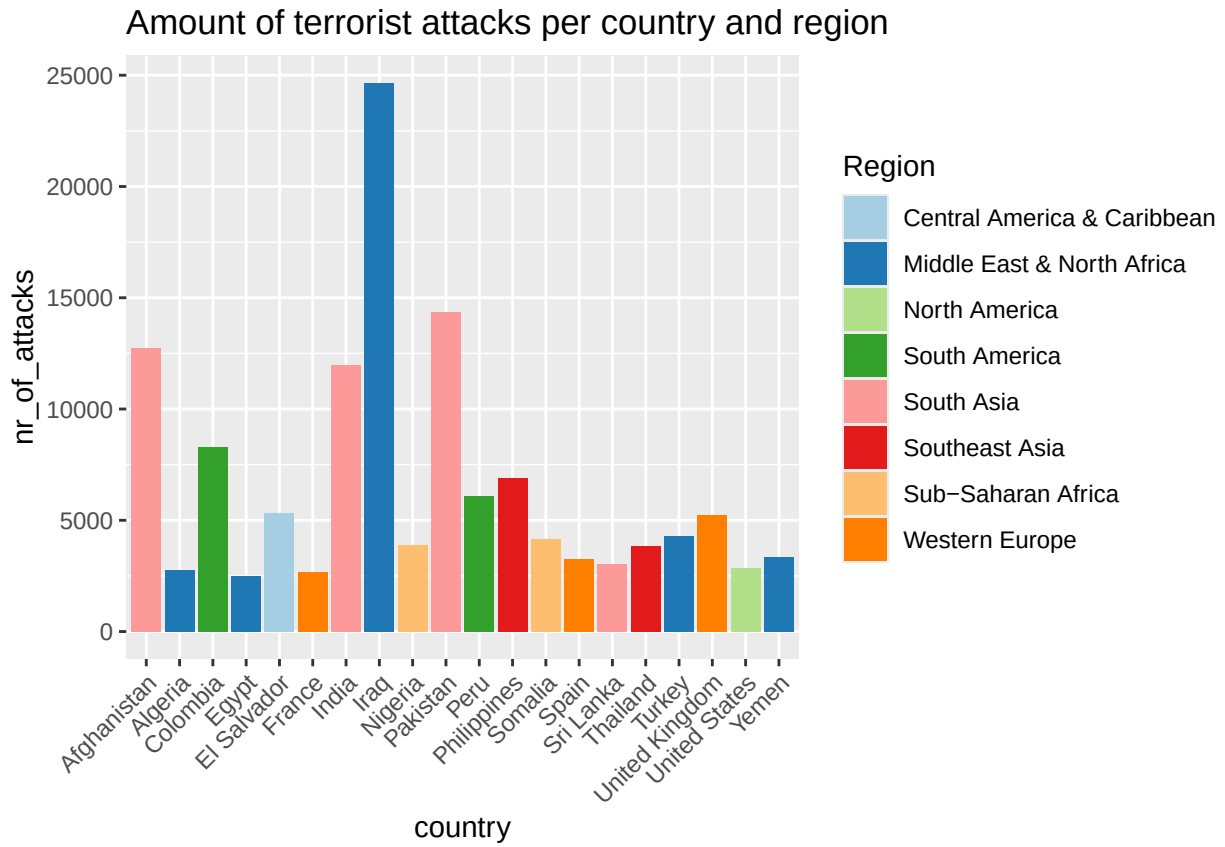


As depicted in the plot, there has been a noticeable surge in terrorism, particularly after 2010.

##2.2 Locations of terrorism

Middle East and N Africa (~27% of total), South Asia (~24%) and S America (11%) are the top three regions in terms of number of attacks. Iraq (~12.9% of total), Pakistan,(~8%), Afhganistan (~6.6%), India(~6.4%) and Colombia (4.7%) are the top five countries in terms of number of attacks.

```r
#table
library(ggplot2)

top20_countries <- df %>%
  group_by(region, country) %>%
  summarise(nr_of_attacks = n(), .groups = "drop_last") %>%
  mutate(percent = nr_of_attacks / sum(nr_of_attacks)) %>%
  arrange(desc(nr_of_attacks)) %>%
  head(n = 20)

# Visual by country
ggplot(data = top20_countries) +
  geom_bar(aes(x = country, y = nr_of_attacks, fill = region), stat = "identity") +
  scale_fill_brewer(palette = "Paired") +  # Change the color palette as needed
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = 'Amount of terrorist attacks per country and region', fill = 'Region')
```



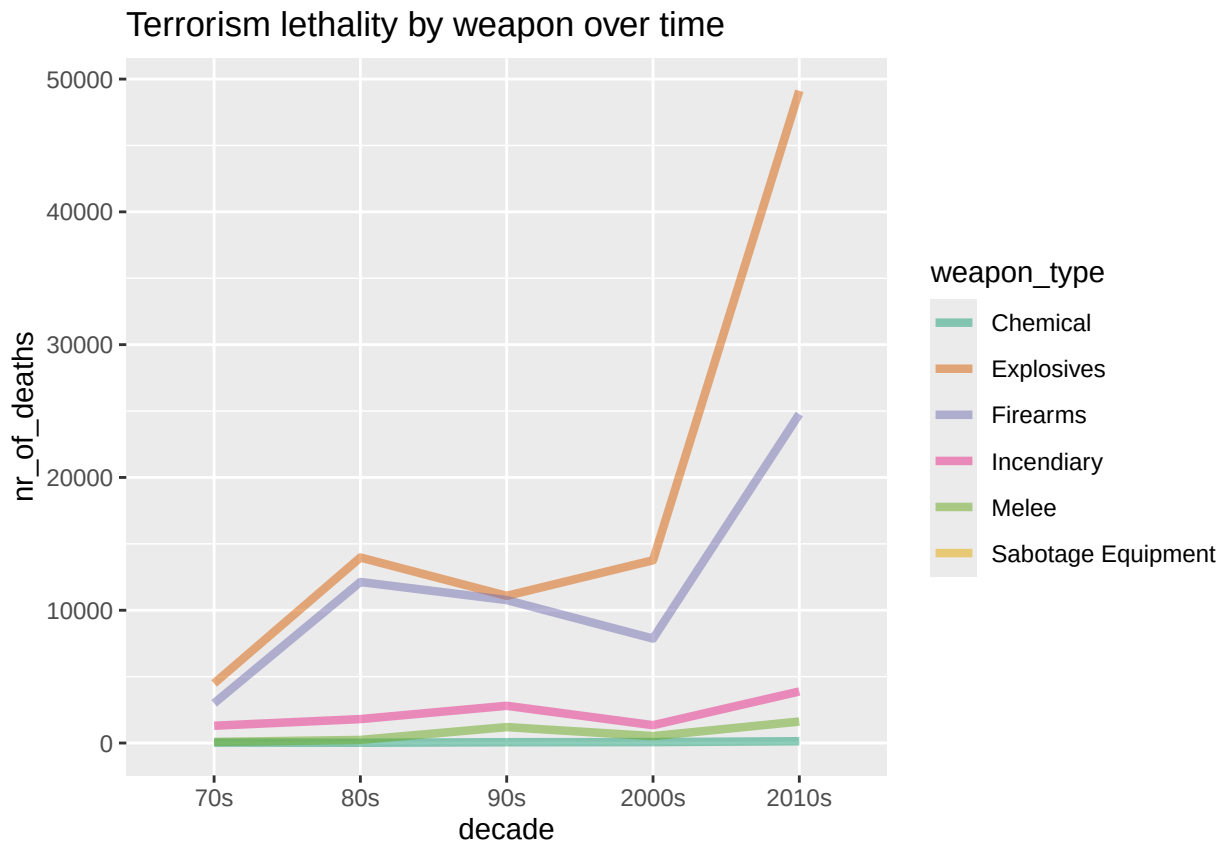
##2.3 Has terrorism become deadlier?

Over half of all deaths by terrorist attack have occurred during bomb attacks. The next deadliest weapon grouping is “Firearms” responsibile for ~32% of all Terror attack deaths.

```r
#table
weapon_lethality <- df %>%
  filter(weapon_type != "Unknown") %>%
  select(decade, weapon_type, nkill) %>%
```

```
  group_by(decade, weapon_type, .drop = FALSE) %>%
  summarise(nr_of_deaths = n(), .groups = "drop_last") %>%
  top_n(n = 5, wt = nr_of_deaths) %>%
  mutate(percent_deaths = (nr_of_deaths / sum(nr_of_deaths) * 100))

# Visual by decade / weapon type
ggplot(data = weapon_lethality, aes(x = decade, y = nr_of_deaths, col = weapon_type, group = weapon_type)) +
  geom_line(size = 1.5, alpha = 0.5) +
  scale_color_brewer(palette = "Dark2") +  # Change the color palette as needed
  labs(title = 'Terrorism lethality by weapon over time')
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



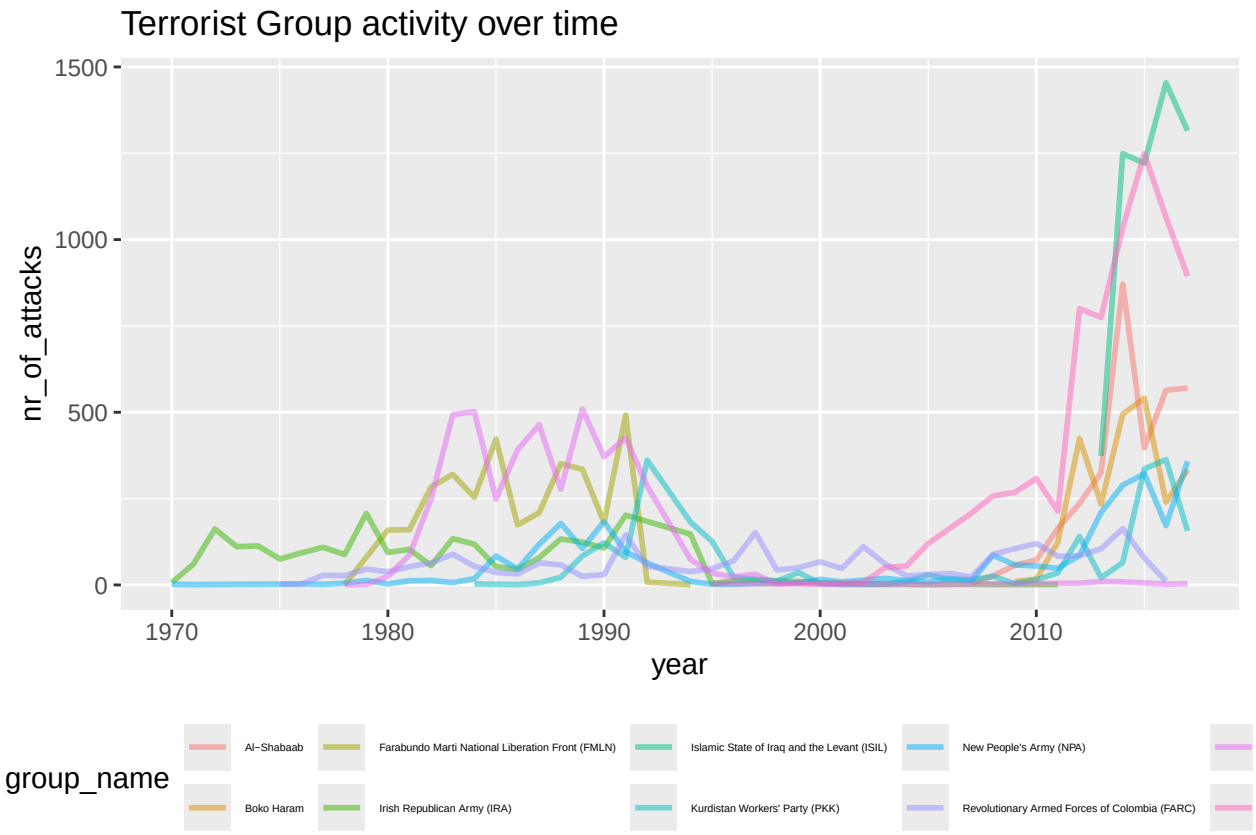
##2.4 Activity of groups over time First we identify the top ten Terror Groups in terms of number of attacks

```
top10_groups <- df %>%
  filter(group_name != "Unknown") %>%
  group_by(group_name) %>%
  summarise(nr_of_attacks = n(), .groups = "drop_last") %>%
  arrange(desc(nr_of_attacks)) %>%
  head(n=10)
```

```
top10_groups
```

```
## # A tibble: 10 x 2
##    group_name                                          nr_of_attacks
##    <chr>                                                       <int>
##  1 Taliban                                                      7478
##  2 Islamic State of Iraq and the Levant (ISIL)                  5613
##  3 Shining Path (SL)                                            4555
##  4 Farabundo Marti National Liberation Front (FMLN)             3351
##  5 Al-Shabaab                                                   3288
##  6 New People's Army (NPA)                                      2772
##  7 Irish Republican Army (IRA)                                  2671
##  8 Revolutionary Armed Forces of Colombia (FARC)                2487
##  9 Boko Haram                                                   2418
## 10 Kurdistan Workers' Party (PKK)                               2310
```

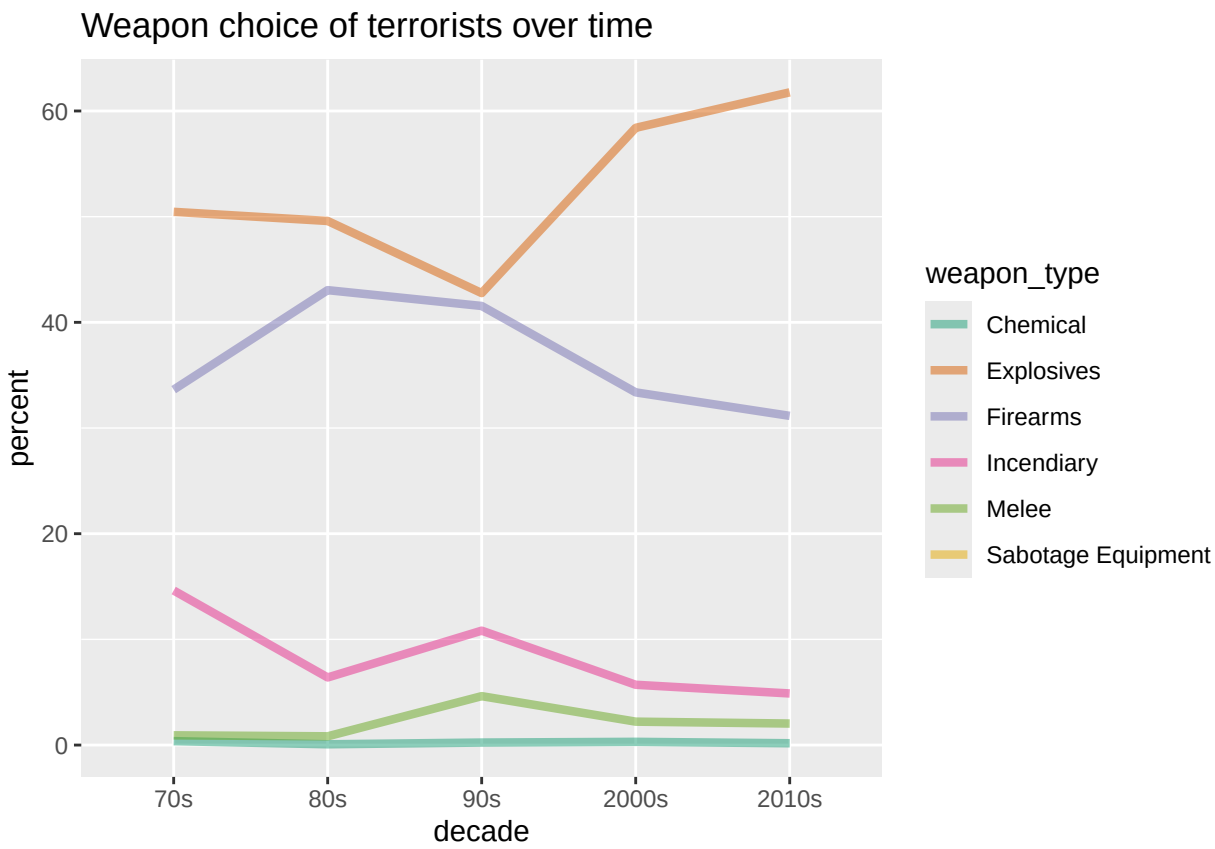
```
#table
top10_groups_activity <- df %>%
filter(df$group_name %in% c("Taliban", "Shining Path (SL)", "Islamic State of Iraq and the Levant (ISIL)
select(year, group_name)%>%
group_by(year, group_name) %>%
  summarise(nr_of_attacks = n(), .groups = "drop_last")%>%
  arrange(desc(nr_of_attacks))%>%
   top_n(n=10, wt=nr_of_attacks)

#Visual by Top 10 Terror Group Activity  / decade since 1970
ggplot(data=top10_groups_activity, aes(x=year, y=nr_of_attacks, col=group_name, group= group_name)) +
  geom_line(size=1, alpha=0.5) +
  theme(legend.position="right")+
  labs(title='Terrorist Group activity over time') +
  theme(legend.position="bottom", legend.text=element_text(size=3.5))
```



The current spike in Terror Activity (since 2000) has been maintained primarily by 4 x Main Groups - Taliban - Boko Haram - NPA - ISIL FARC have shown a small spike in activity since 2000 and IRA and Shining Path have shown a decrease in Activity since 2000

##2.5 Weapon choice over time Did technology growth change what weapons terrorist use?

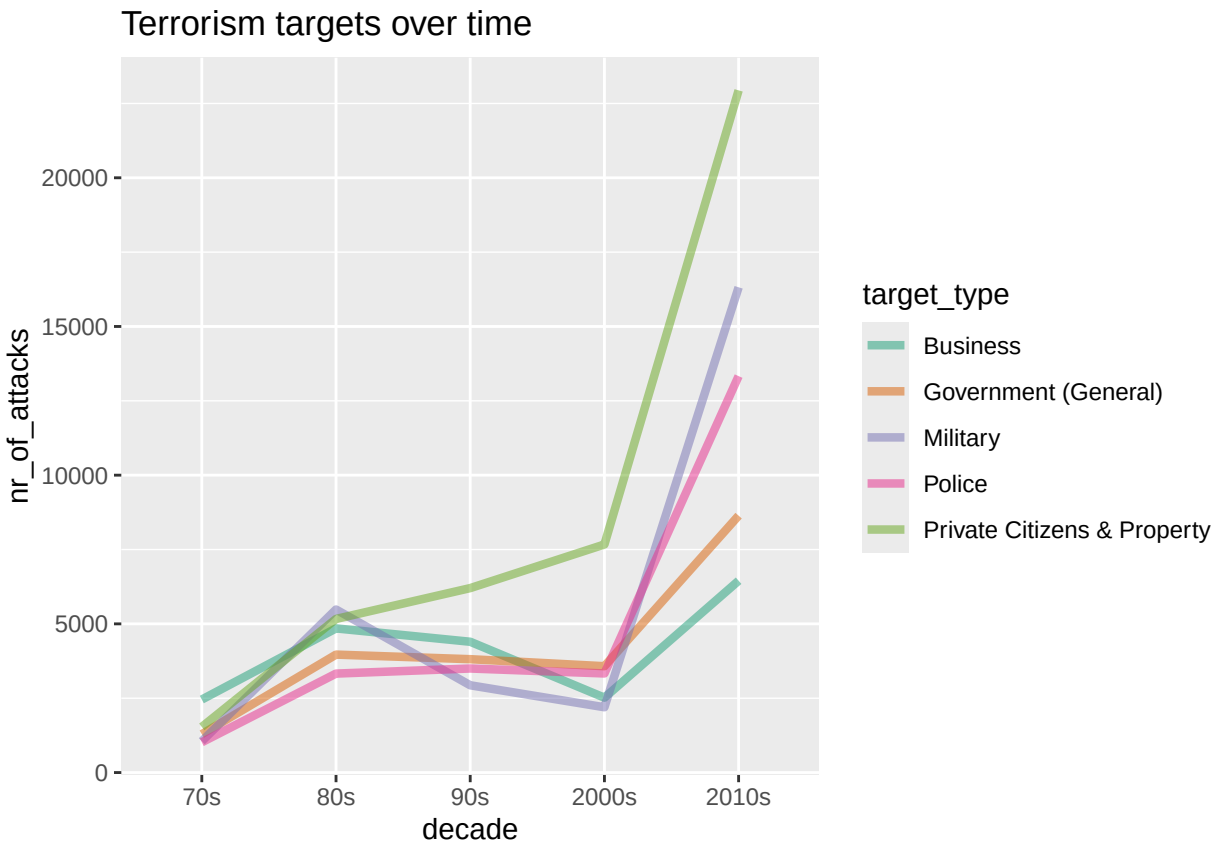
```
dfweapons <- df %>%
  select(year, weapon_type, decade) %>%
  filter(weapon_type != "Unknown")

#table
top15_weapons <- dfweapons %>%
  group_by(decade, weapon_type) %>%
  summarise(nr_of_attacks = n(), .groups = "drop_last") %>%
  top_n(n=5, wt=nr_of_attacks) %>%
  mutate(percent = nr_of_attacks/sum(nr_of_attacks)*100) %>%
  arrange(decade, desc(nr_of_attacks))

#visual

ggplot(data = top15_weapons, aes(x = decade, y = percent, col = weapon_type, group = weapon_type)) +
  geom_line(size = 1.5, alpha = 0.5) +
  scale_color_brewer(palette = "Dark2") +  # Change the color palette as needed
  labs(title = 'Weapon choice of terrorists over time')
```



It seems that explosives and firearmas have been consistently the most popular. The whole top3 of weapon choices seems to have stayed completely consistent over the years.

##2.6 Target choice over time Have the targets changed? Have terrorists changed what targets they use?

```
dftargets <- df %>%
  select(year, target_type, target_sub_type, decade) %>%
  filter(target_type != "Unknown")

#table
dftargetstop <- dftargets %>%
  group_by(decade, target_type) %>%
  summarise(nr_of_attacks = n(), .groups = "drop_last") %>%
  top_n(n=5, wt=nr_of_attacks) %>%
  arrange(decade, desc(nr_of_attacks))

#visual
ggplot(data=dftargetstop, aes(x=decade, y=nr_of_attacks, col=target_type, group= target_type)) +
  geom_line(size=1.5, alpha=0.5)+
  scale_color_brewer(palette = "Dark2") +
  labs(title='Terrorism targets over time')
```

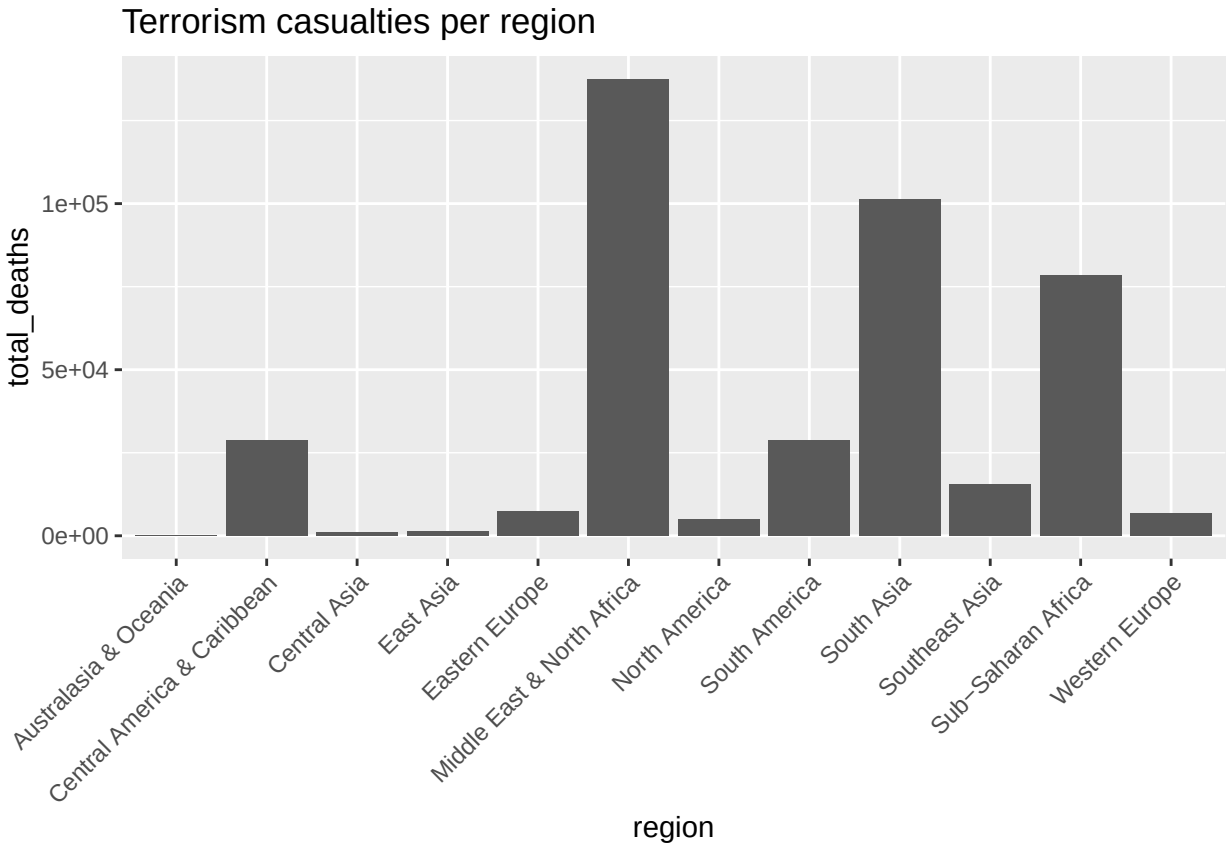


From the data we an see that private citizens have become a way bigger target group than before. It seems that violence has escelated to this innocent group. Besides that the Military has become a bigger target over the decades.

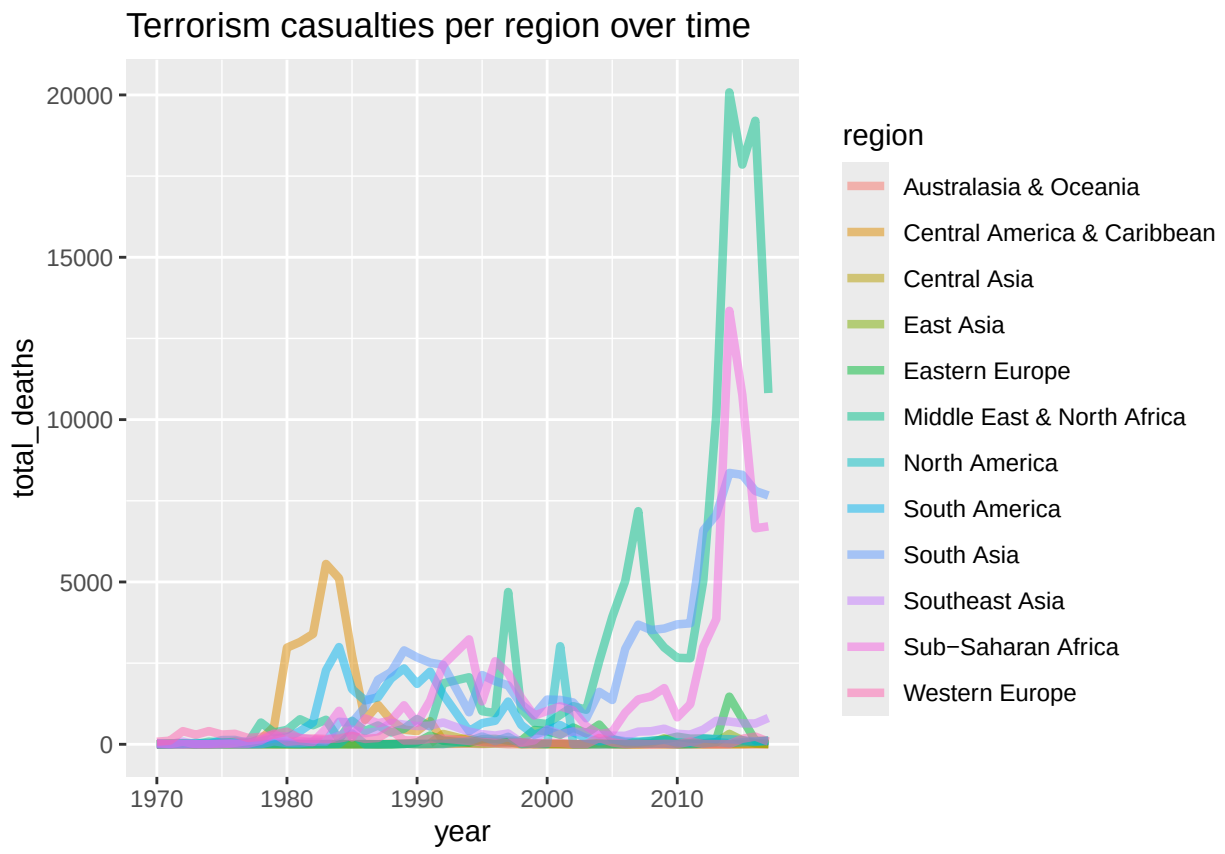
##2.7 Location vs Mortality

```
#Mortality by Region
regionmort <- df %>%
  filter(nkill != 'Unknown') %>%
  select(region, nkill, nwound, year, group_name, decade) %>%
  group_by(region, year) %>%
  summarise(total_deaths = sum(nkill), .groups = "drop_last")

#Raw region amounts
ggplot(data=regionmort, aes(x=region, y=total_deaths)) +
  geom_histogram(stat='identity') +
  theme(axis.text.x= element_text(angle=45, hjust=1))+
  labs(title='Terrorism casualties per region')
```

```
## Warning in geom_histogram(stat = "identity"): Ignoring unknown parameters:
## `binwidth`, `bins`, and `pad`
```




```
#and over time
ggplot(data=regionmort, aes(x=year, y=total_deaths, col=region, group= region)) +
  geom_line(size=1.5, alpha=0.5) +
  labs(title='Terrorism casualties per region over time')
```



The Middle East and North Africa have had an immense increase of deaths from 2003 upwards with links in well with incraesed instability in the region during and after the Iraq war.

#3. Has terrorism gone up over the past few decades - taking into account population growth?

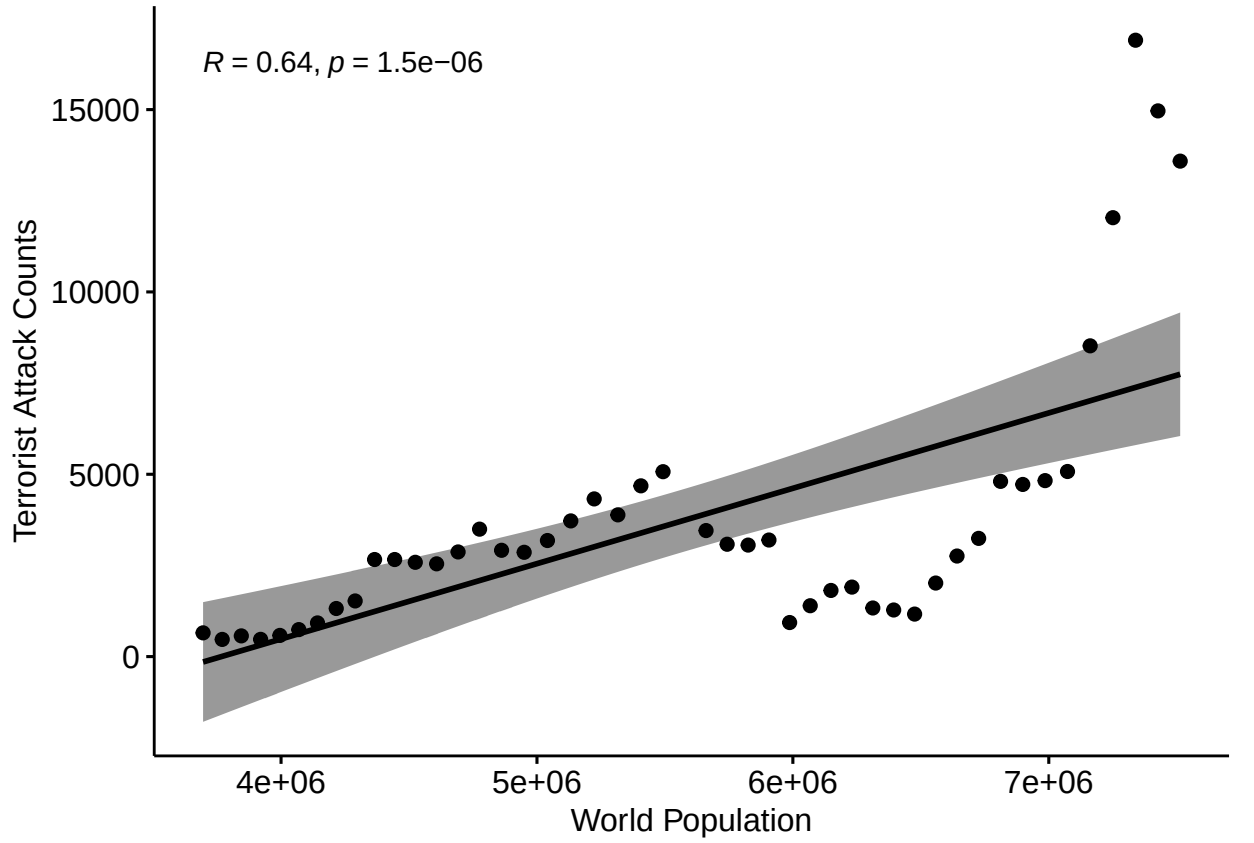
We have to take into account popoulation growth first. To do this we'll first check if population growth and amount of attacks per year are correlated.

##3.1 Correlation analysis.

```
#first reframe the data to get it grouped per year
df3 <- df2 %>%
  group_by(year) %>%
  summarise(terrorist_attacks_count = n())

df3 <- inner_join(df3, popworld, by = c("year" = "Time"))

df3 <- df3 %>%
  mutate(decade =
           ifelse(year<1980, '70s',
                  ifelse(year < 1990, '80s',
                         ifelse(year < 2000, '90s',
                                ifelse( year < 2010, '2000s', '2010s')))))


df3$decade <- factor(df3$decade, levels=c("70s", "80s", "90s", "2000s", "2010s"))

cor.test(df3$PopTotal, df3$terrorist_attacks_count, method="pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  df3$PopTotal and df3$terrorist_attacks_count
## t = 5.5646, df = 44, p-value = 1.467e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4332257 0.7862924
## sample estimates:
##       cor
## 0.6426955
```

```
ggscatter(df3, y = "terrorist_attacks_count", x = "PopTotal",
          add = "reg.line", conf.int = TRUE,
          cor.coef = TRUE, cor.method = "pearson",
          ylab = "Terrorist Attack Counts", xlab = "World Population")
```



It seems that there is a medium correlation (r=0.64, p<0.05) for world population and terrorist attack counts. However, they seem to disconnect when the population reaches more than 5.8 billion. Our linear model doesn't explain the variance too well so perhaps we could try seeing if a polynomial model works better.

##3.2 Are the amounts of attacks different over time? Lets do linear regression to see if the variance in amount of attacks can be explained merely by population growth.

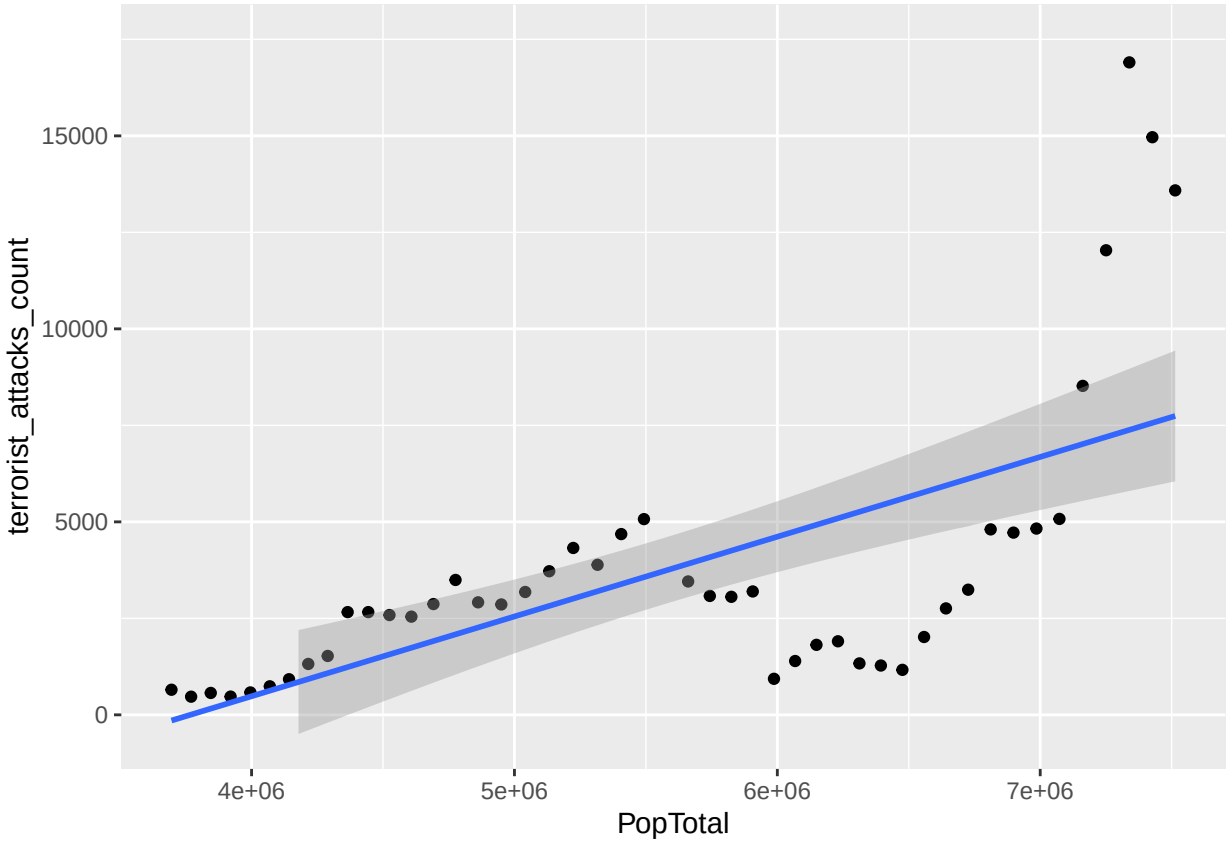
H_0 : variance-terrorist attacks properly explained by population growth
H_a : variance-terrorist attacks not properly explained by population growth

###3.2.1 Linear Model

```
m1 <- lm(data=df3, terrorist_attacks_count ~ PopTotal)
summary(m1)
```

```
##
## Call:
## lm(formula = terrorist_attacks_count ~ PopTotal, data = df3)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -4432  -1755    399   1004   9521
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.785e+03  2.110e+03  -3.690 0.000615 ***
## PopTotal     2.067e-03  3.714e-04   5.565 1.47e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2891 on 44 degrees of freedom
## Multiple R-squared:  0.4131, Adjusted R-squared:  0.3997
## F-statistic: 30.96 on 1 and 44 DF,  p-value: 1.467e-06
```

```
#Adjusted R-squared: 0.397 with p<0.05.

ggplot(data=df3, aes(x=PopTotal, y=terrorist_attacks_count)) +
  geom_point() +
  geom_smooth(method="lm",formula= y ~ x)+
  scale_y_continuous(limits = c(-500, 17500))
```



The Linear Model is significant but explains the variance in terrorist attacks count quite poorly with an adjusted R-squared of 0.397 ($p<0.05$). This means we REJECT H0 as the variance is not properly explained by population growth, meaning there are other factors in play.

###3.2.2 Polynomial regression

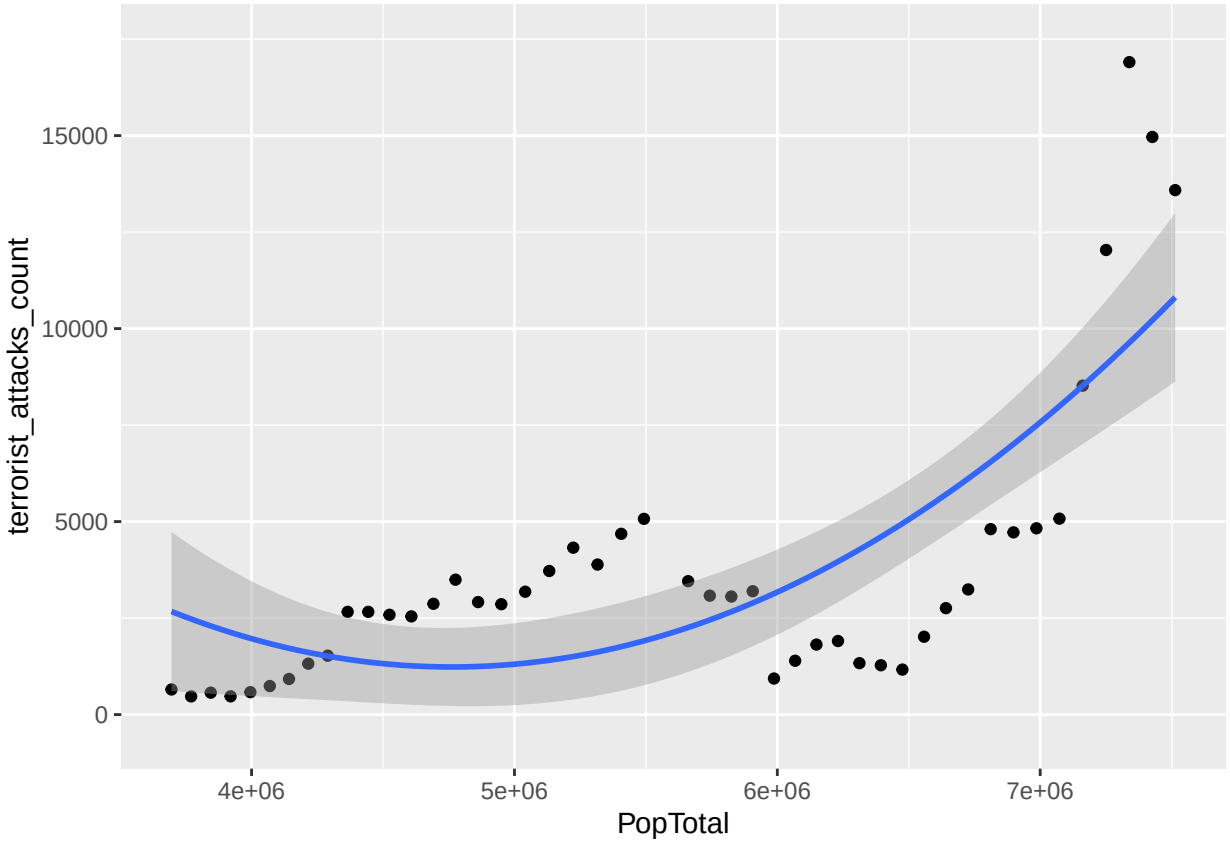
What seemed to be the case in our linear was that after a while there was no linear relationship anymore as most of the points fell outside of the 95% conf interval of the fitted line. Let's see if we can better explain the data with a quadratic term of PopTotal. It should ofcourse be kept in mind that we are heavily overfitting the model by adding these polynomial terms. However, because we suspect a non-linear relationship based on our initial analysis its an interesting thing to see on what kind of order the terrorist attack count has diverted from normal growth through population gains. Beside, that we can try and making some predictions about future amount of terrorist attacks based on our better fitted model since the dataset lacks other interesting variables to predict terrorist attacks.

```
m2 <- lm(data=df3, terrorist_attacks_count ~ PopTotal + I(PopTotal^2))
summary(m2)
```

```
##
## Call:
## lm(formula = terrorist_attacks_count ~ PopTotal + I(PopTotal^2),
##     data = df3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3783.8 -2016.1     7.1  1659.0  7268.4
##
```

```
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.990e+04  9.979e+03   2.996 0.004523 **
## PopTotal      -1.204e-02  3.686e-03  -3.267 0.002141 **
## I(PopTotal^2)  1.264e-09  3.291e-10   3.842 0.000396 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2523 on 43 degrees of freedom
## Multiple R-squared:  0.5631, Adjusted R-squared:  0.5428
## F-statistic: 27.71 on 2 and 43 DF,  p-value: 1.857e-08
```

```
#Adjusted R-squared: 0.536 with p<0.05.

ggplot(data=df3, aes(x=PopTotal, y=terrorist_attacks_count)) +
  geom_point() +
  geom_smooth(method="lm",formula= y ~ x + I(x^2)) +
  scale_y_continuous(limits = c(-500, 17500))
```



This already explains the variance in terrorist attack count much better with an adjusted R-squared of 0.536 ($p<0.05$) but still most points fall outside of the confidence interval.

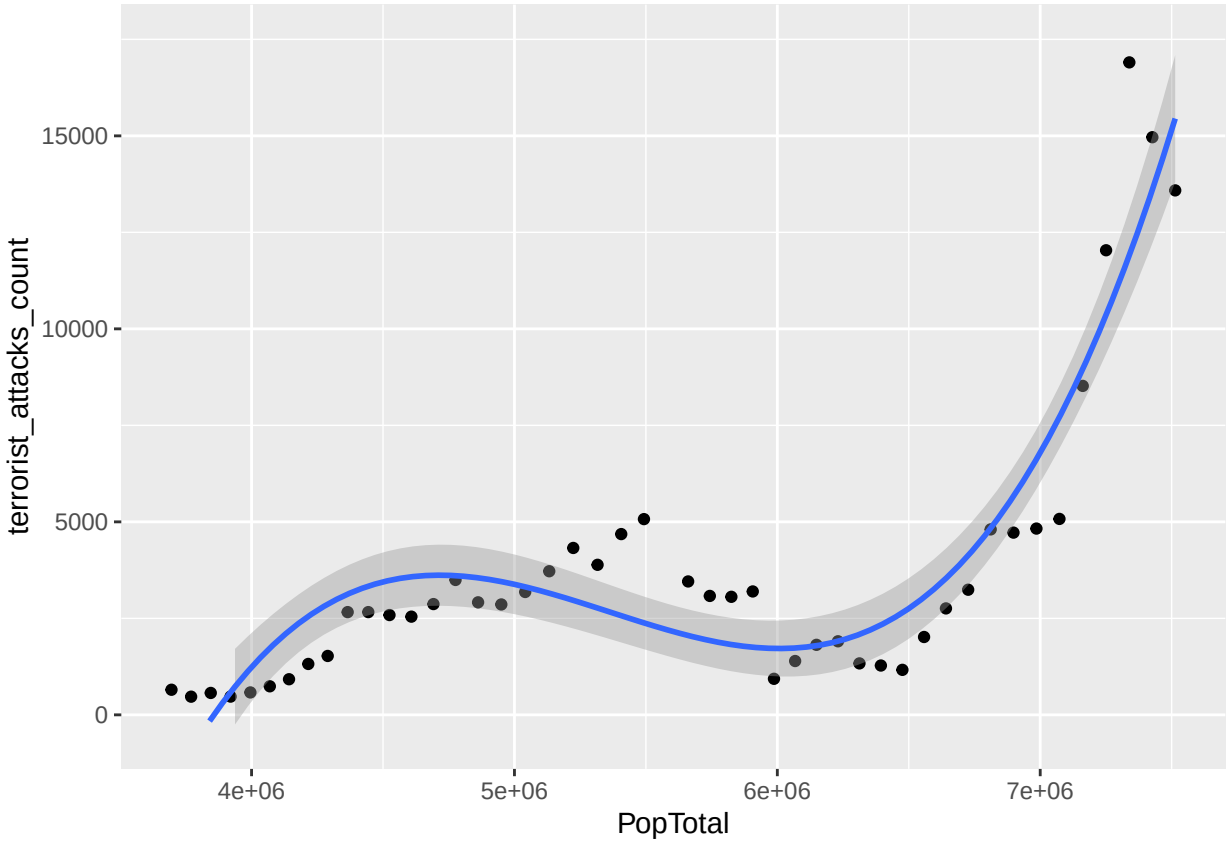
###3.2.3 Third order Polynomial regression Lets see if we can make a better fit with higher order versions of the population variable.

```r
m3 <- lm(data=df3, terrorist_attacks_count ~ PopTotal + I(PopTotal^2) + I(PopTotal^3))
summary(m3)
```

```
##
## Call:
## lm(formula = terrorist_attacks_count ~ PopTotal + I(PopTotal^2) +
##     I(PopTotal^3), data = df3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2638.0  -931.8  -451.8  1147.2  4995.1
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.563e+05  3.192e+04  -8.031 5.07e-10 ***
## PopTotal       1.493e-01  1.782e-02   8.379 1.67e-10 ***
## I(PopTotal^2) -2.827e-08  3.243e-09  -8.716 5.75e-11 ***
## I(PopTotal^3)  1.757e-15  1.927e-16   9.122 1.62e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1479 on 42 degrees of freedom
## Multiple R-squared:  0.8534, Adjusted R-squared:  0.843
## F-statistic: 81.52 on 3 and 42 DF,  p-value: < 2.2e-16
```

```r
#Adjusted R-squared: 0.8356 with p<0.05.

ggplot(data=df3, aes(x=PopTotal, y=terrorist_attacks_count)) +
  geom_point() +
  geom_smooth(method="lm",formula= y ~ x + I(x^2) + I(x^3)) +
  scale_y_continuous(limits = c(-500, 17500))
```

```
## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_smooth()`).
```



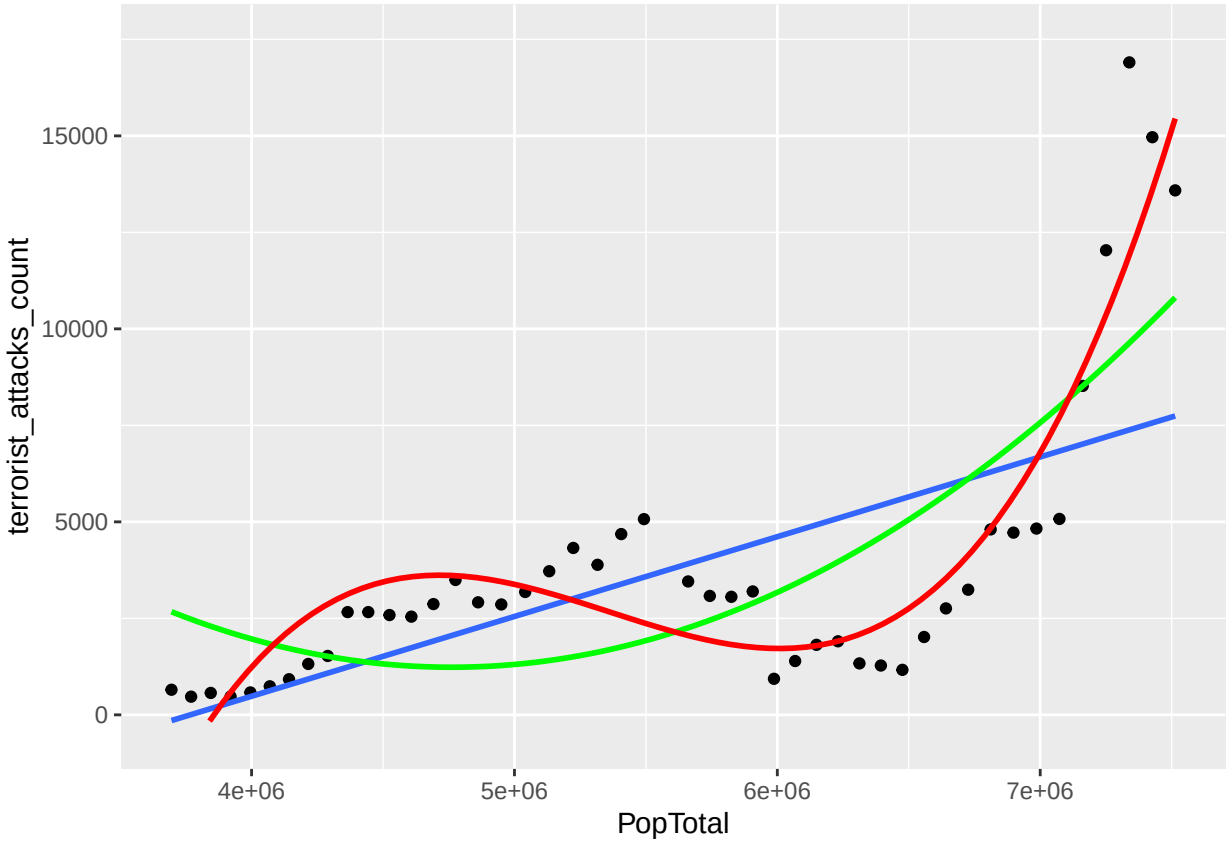
Again, even a better fit is achieved and again, overfitting is incredibly high right now. Let's see if we can see which of these models is better or worse through an ANOVA.

###3.2.4 Model tests Because the above models are all nested we can use ANOVA to see which one is better.

$$H_0 : \text{SSE}_{m1} = \text{SSE}_{m2} = \text{SSE}_{m3}$$
$$H_a : \text{at least two SSE's are different}$$

```
#visual, without confidence interval for visiblity
ggplot(data=df3, aes(x=PopTotal, y=terrorist_attacks_count)) +
  geom_point() +
  geom_smooth(method="lm",formula= y ~ x, se=F) +
  geom_smooth(method="lm",formula= y ~ x + I(x^2), se=F, col='green') +
  geom_smooth(method="lm",formula= y ~ x + I(x^2) + I(x^3), se=F, col='red') +
  scale_y_continuous(limits = c(-500, 17500))
```

```
## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_smooth()`).
```



Which of these models is better? First we compare model 1 and model 2.

```
anova(m1, m2, test="F")
```

```
## Analysis of Variance Table
##
## Model 1: terrorist_attacks_count ~ PopTotal
## Model 2: terrorist_attacks_count ~ PopTotal + I(PopTotal^2)
##   Res.Df       RSS Df Sum of Sq      F    Pr(>F)
## 1     44 367814724
## 2     43 273805903  1  94008821 14.764 0.0003965 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#p<0.05 for model 2, meaning it's the better model. Is it also better than model 3?

anova(m2, m3, test="F")
```

```
## Analysis of Variance Table
##
## Model 1: terrorist_attacks_count ~ PopTotal + I(PopTotal^2)
## Model 2: terrorist_attacks_count ~ PopTotal + I(PopTotal^2) + I(PopTotal^3)
##   Res.Df       RSS Df Sum of Sq      F   Pr(>F)
## 1     43 273805903
## 2     42  91845622  1 181960282 83.209 1.62e-11 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#p<0.05 for model 3, meaning it's the best model.
```

The ANOVA at the end to compare the models doesn't explain too much as, yes, for this dataset the variance is better explained by the second and third model but we are overfitting the data massively at this point, especially since we only have an n=46 because we are looking at aggregates per year.

Not too much can be inferred from this except that clearly populating growth has decoupled from the occurance of terrorist attacks and some other variable(s) have entered the fold in the past decade or so that have pushed up the amount of terrorist attacks.

Before we start doing some predictions with our model, let's check some assumptions with the help of the 'car' package.

##3.3 Regression Diagnostics

###3.3.1 Outlier Tests One of the main reasons we suspect that population growth has decoupled as a linear predictor for terrorism attacks is because of the data from the last ~10 years. Let's see first what is deemed an outlier with a Bonferroni Outlier Test.

```
outlierTest(m1)
```

```
##    rstudent unadjusted p-value Bonferroni p
## 44 3.948107         0.00028735     0.013218
```

```
outlierTest(m2)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##    rstudent unadjusted p-value Bonferroni p
## 44 3.453797          0.0012752     0.058659
```

```
outlierTest(m3)
```

```
##    rstudent unadjusted p-value Bonferroni p
## 44 4.411949         7.2609e-05      0.00334
```

Model 1 and Model 3 return a p<0.05 outlier which is observation. Let's see that in a bit more detail.

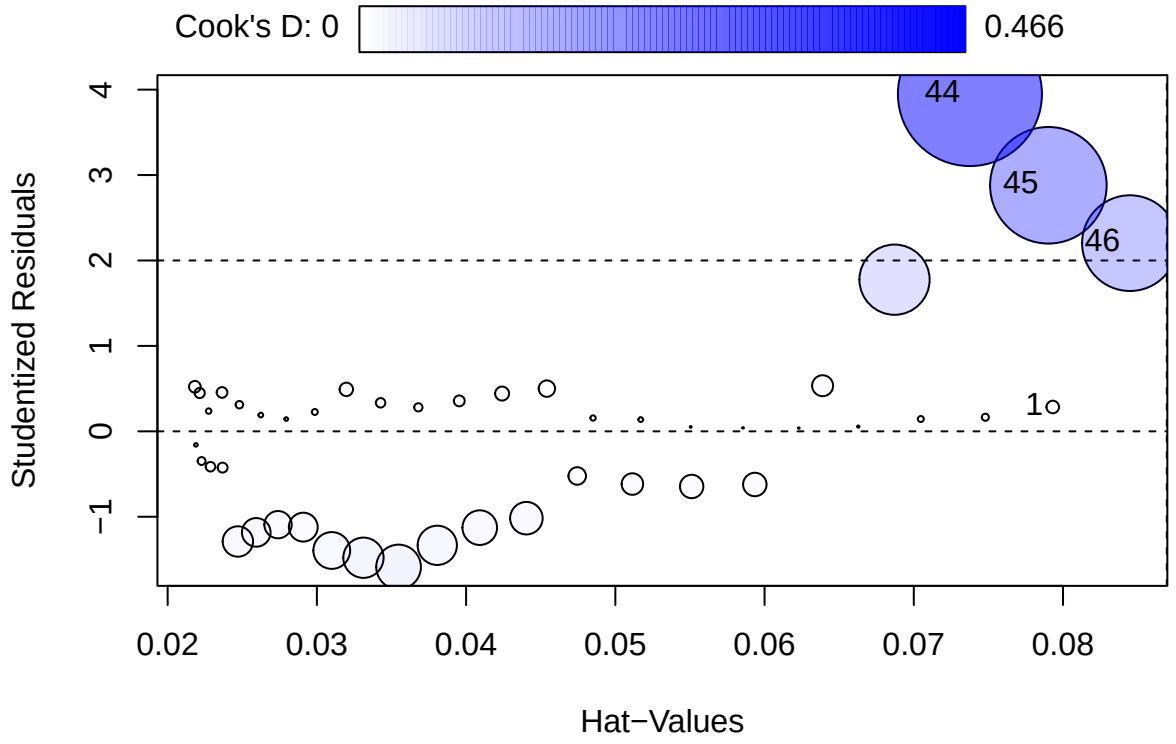
```
df3[44, ]
```

```
## # A tibble: 1 x 13
##    year terrorist_attacks_count SortOrder Notes ISO3_code ISO2_code SDMX_code
##   <dbl>                   <int>     <dbl> <chr> <chr>     <chr>         <dbl>
## 1  2014                   16903         1 <NA>  <NA>      <NA>              1
## # i 6 more variables: LocTypeID <dbl>, LocTypeName <chr>, ParentID <dbl>,
## #   PopTotal <dbl>, PopDensity <dbl>, decade <fct>
```

So the year 2014 is a definite outlier, this is also very clear in the plots but we shouldn't remove this as, since these our yearly aggregates, this is an important datapoint. Let's also view some influence plots to see what years are the most distortionairy in regards to our linear model. Grid arrange seems to not accept influence plots so they are not in 1 plot.

```r
infl1 <- influencePlot(m1, id.method = "noteworthy")
```

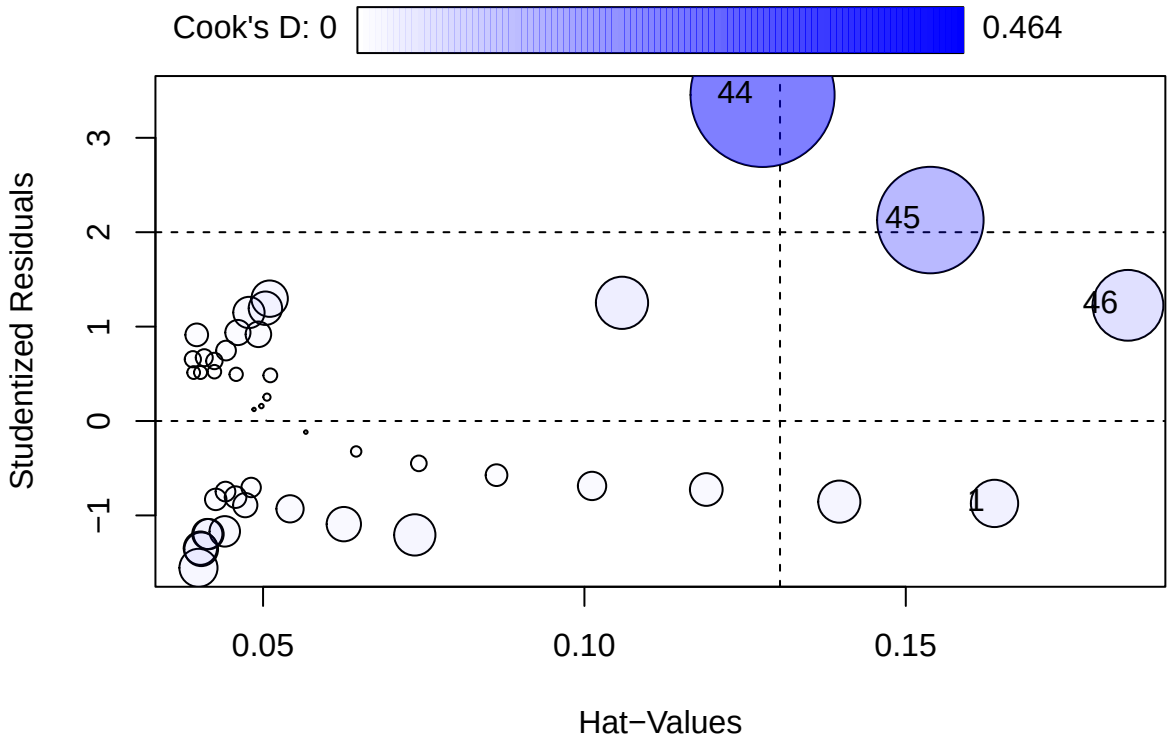
```
## Warning in plot.window(...): "id.method" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "id.method" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is not
## a graphical parameter

## Warning in box(...): "id.method" is not a graphical parameter

## Warning in title(...): "id.method" is not a graphical parameter

## Warning in plot.xy(xy.coords(x, y), type = type, ...): "id.method" is not a
## graphical parameter
```




```r
infl2 <- influencePlot(m2, id.method = "noteworthy")
```

```
## Warning in plot.window(...): "id.method" is not a graphical parameter
```

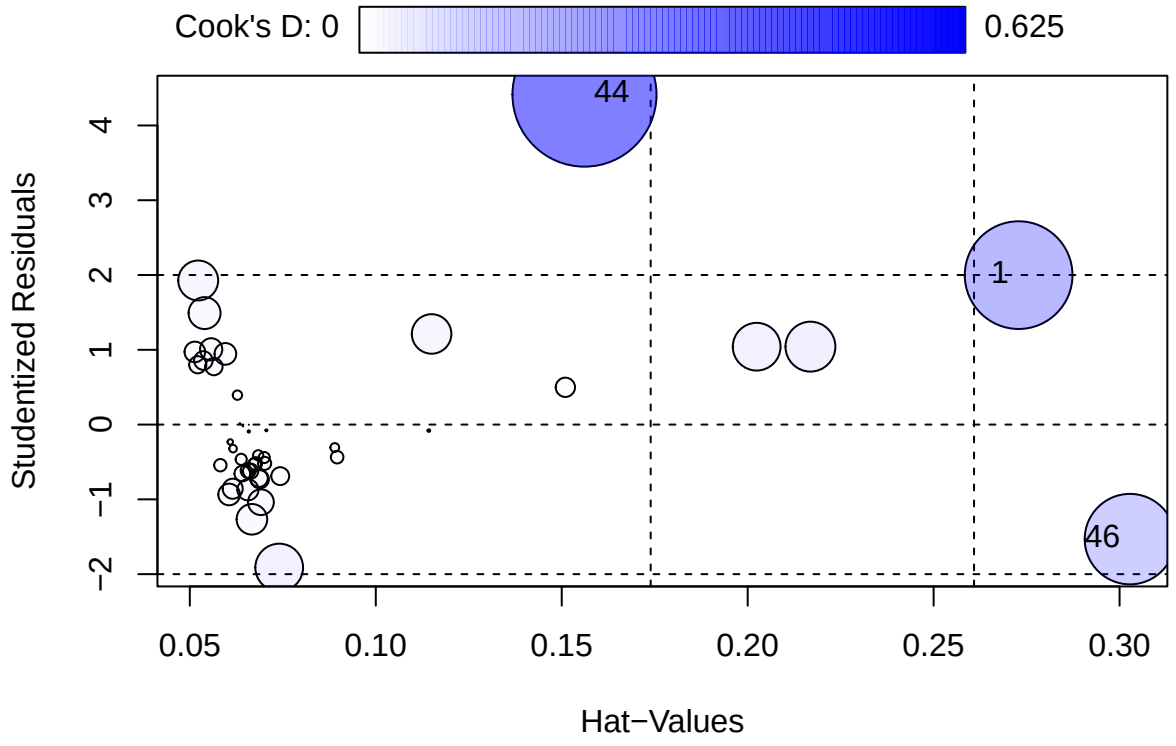
```
## Warning in plot.xy(xy, type, ...): "id.method" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is not
## a graphical parameter

## Warning in box(...): "id.method" is not a graphical parameter

## Warning in title(...): "id.method" is not a graphical parameter

## Warning in plot.xy(xy.coords(x, y), type = type, ...): "id.method" is not a
## graphical parameter
```




```
infl3 <- influencePlot(m3, id.method = "noteworthy")
```

```
## Warning in plot.window(...): "id.method" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "id.method" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "id.method" is not
## a graphical parameter
```

```
## Warning in box(...): "id.method" is not a graphical parameter

## Warning in title(...): "id.method" is not a graphical parameter

## Warning in plot.xy(xy.coords(x, y), type = type, ...): "id.method" is not a
## graphical parameter
```

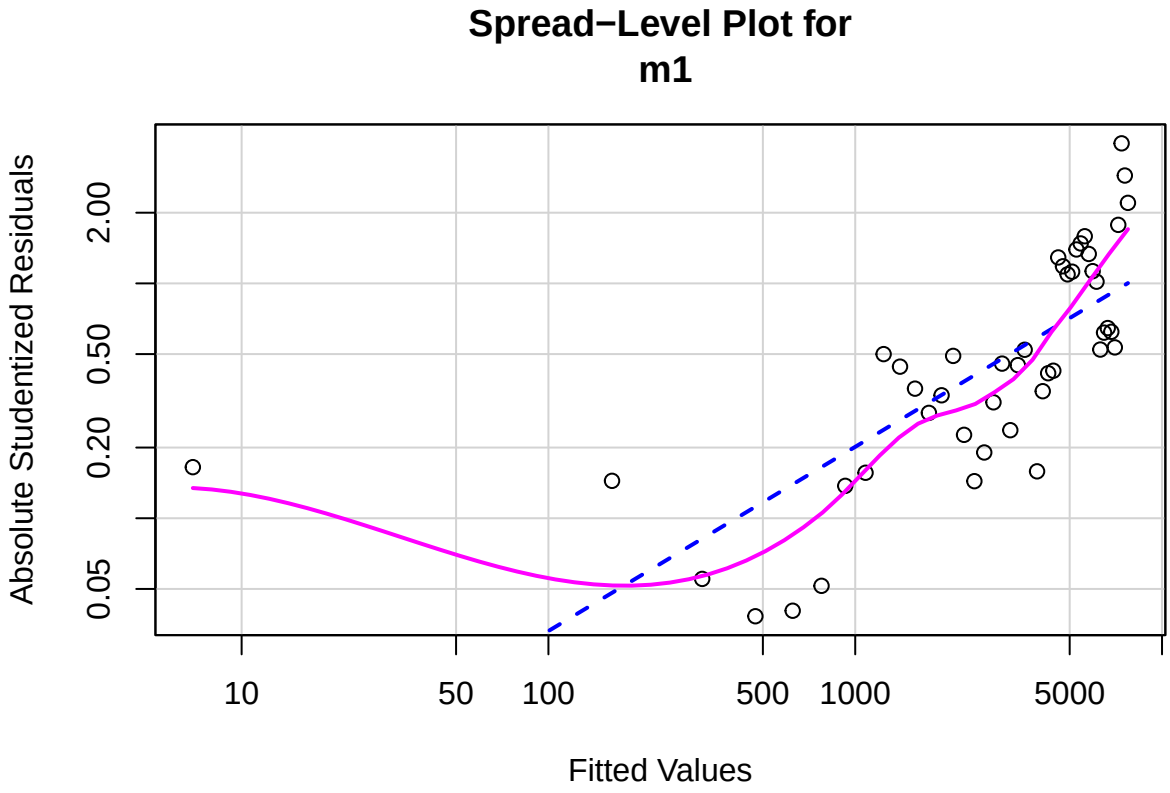


Constant recurring datapoints are 46 and 44, year 2016 and 2014 respectively.

###3.3.2 Homoscedasticity We expect none of the models to fare well under this assumption due to the fact that terrorist attacks seem to have completely decoupled from population levels about ~10 years ago. Next to that our second and third model employ higher order polynomials to model linearly a non-linear relationship between the two variables. Below are a few spread level plots but it's quite obvious this assumption won't be met. Again, grid arrange can't handle plots with a numerical output.

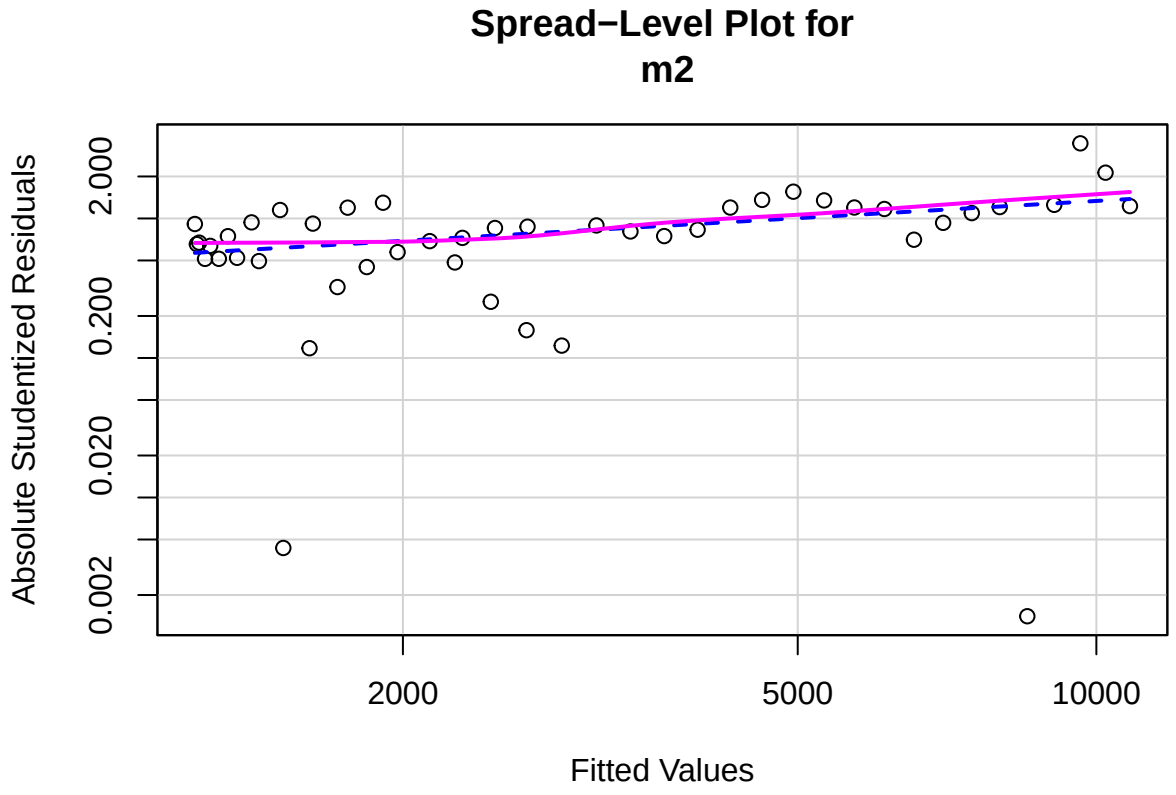
```
spreadLevelPlot(m1)
```

```
## Warning in spreadLevelPlot.lm(m1):
## 1 negative fitted value removed
```




```
##
## Suggested power transformation:  0.2157472
```

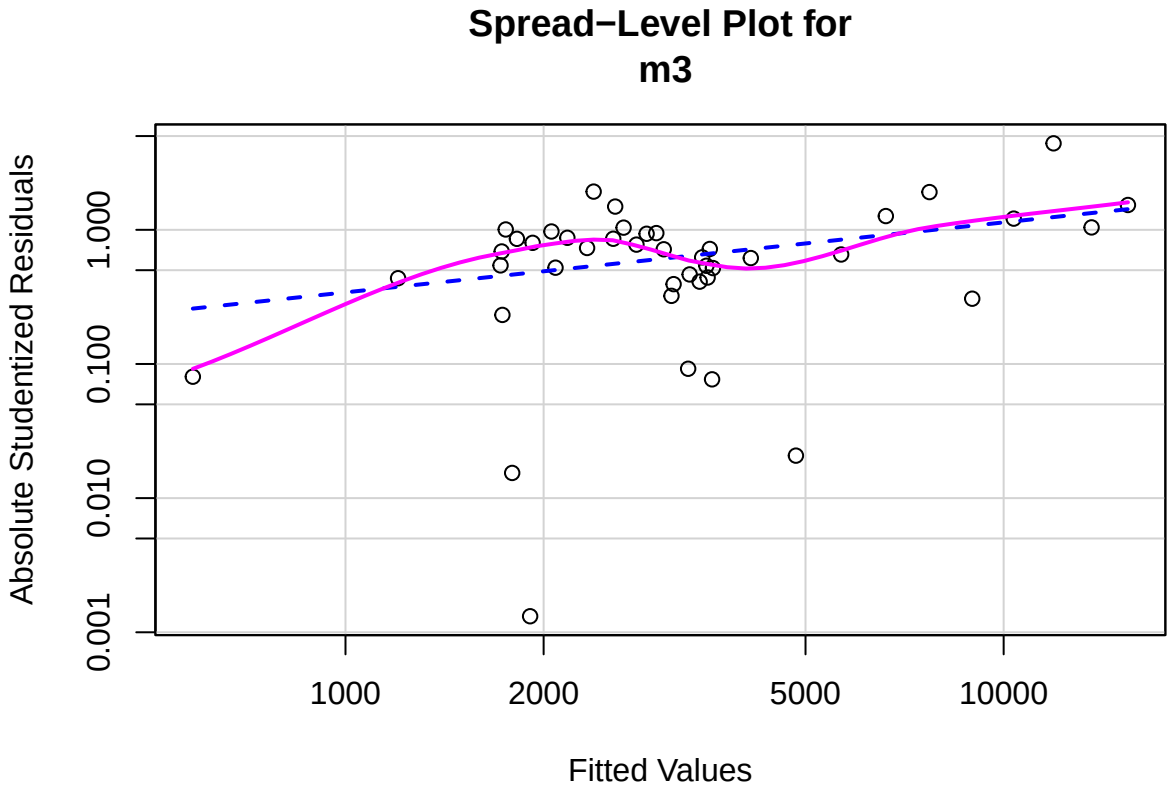
```
spreadLevelPlot(m2)
```




```
##
## Suggested power transformation:  0.5907935
```

```
spreadLevelPlot(m3)
```

```
## Warning in spreadLevelPlot.lm(m3):
## 3 negative fitted values removed
```

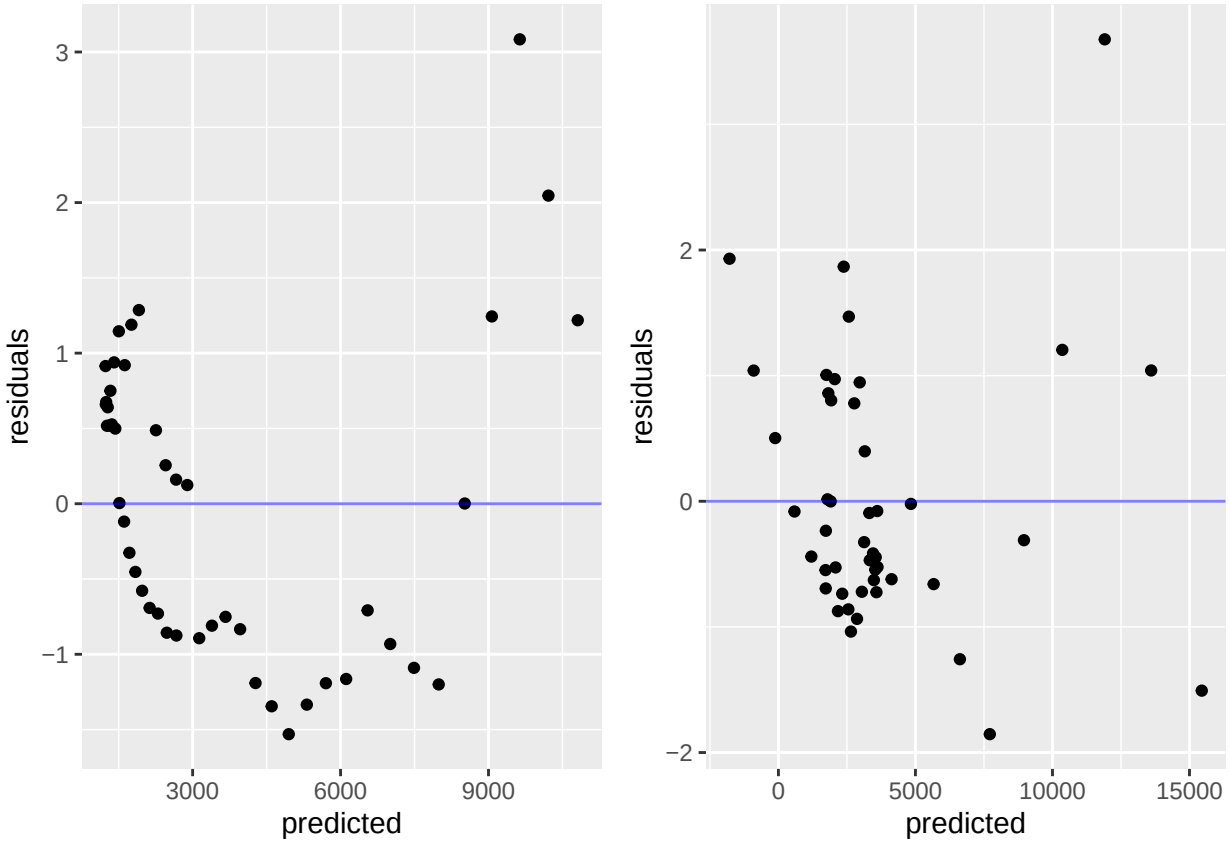



```
##
## Suggested power transformation:  0.4783159
```

###3.3.3 Linearity

It was already quite clear from our original correlation plot that linearity is absent, this is why we added in nested polynomials, however it might still be interesting to see how the points are distributed in the two models with more than 1 independent variable.

```
library(gridExtra)

# Create data frames for plotting
data_m2 <- data.frame(predicted = predict(m2), residuals = rstandard(m2))
data_m3 <- data.frame(predicted = predict(m3), residuals = rstandard(m3))

# Create plots using ggplot
lin2 <- ggplot(data = data_m2, aes(x = predicted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, colour = "blue", alpha = 0.5)

lin3 <- ggplot(data = data_m3, aes(x = predicted, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, colour = "blue", alpha = 0.5)

# Arrange plots using grid.arrange
grid.arrange(lin2, lin3, nrow = 1)
```



As we can see they both lack linearity which is exactly as expected as we added in non-linear elements through the nested polynomials purely because we spotted a partly non-linear relationship in the first model.

###3.3.4 Independence of residuals Let's check the independence of residuals with the help of a Durbin Watson test. Unfortunately since our data is essentially 'panel data', specifically Time Series, since it's measurements over time from the same group (the world) the standard car packaged DWtest is not going to be useful. It will always tell us that the errors are autocorrelated.

```
durbinWatsonTest(m1)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1        0.860651      0.184112       0
##  Alternative hypothesis: rho != 0
```

```
durbinWatsonTest(m2)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.8610159     0.2349111       0
##  Alternative hypothesis: rho != 0
```

```
durbinWatsonTest(m3)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1        0.610455     0.6768258       0
##  Alternative hypothesis: rho != 0
```

As expected the we reject the H0 three times with p=0.

###3.3.5 multicollinearity

Since we are working with nested polynomial models this diagnostic is also going to be less relevant as multicollinearity will be a thing.

```
vif(m2)
```

```
##      PopTotal I(PopTotal^2)
##       129.316       129.316
```

```
vif(m3)
```

```
##      PopTotal I(PopTotal^2) I(PopTotal^3)
##      8802.804     36566.600      9694.577
```

As expected the variance inflation factors are really high and normally this would mean dropping the extra variables. But since this is the rule only fro *linear combinations* of variables our model is exempt.

##3.4 Model validation and forecasting

###3.4.1 k-fold cross-validation We can validate our models to see whether adding the polynomials of world population improved our predictive power. We'll be using k-fold validation to do so. First we'll set a seed for reproducibility, we won't divy up the data in a test and train set for now since this is all the *real data* we have.

```
set.seed(42)
```

Now let's use Caret for training the model with cross validation.

```
trainmethod <- trainControl(method="cv", number=5, returnData=TRUE, returnResamp='all')

model1 <- train(data=df3, terrorist_attacks_count ~ PopTotal, method='lm', trControl=trainmethod)
model2 <- train(data=df3, terrorist_attacks_count ~ PopTotal + I(PopTotal^2), method='lm', trControl=tra
model3 <- train(data=df3, terrorist_attacks_count ~ PopTotal + I(PopTotal^2) + I(PopTotal^3), method='l
```

Let's analyse our obtained models. We don't use summarise() on the cross validated models because it seems that base-r function can't read the cross-validation information and just outputs the original r-squared values.

```
model1$resample
```

```
##       RMSE  Rsquared      MAE intercept Resample
## 1 2384.677 0.4610351 1878.605      TRUE    Fold1
## 2 2154.446 0.2661987 1749.946      TRUE    Fold2
## 3 3221.007 0.3690876 2342.012      TRUE    Fold3
## 4 2805.990 0.6438384 2017.555      TRUE    Fold4
## 5 3633.035 0.4451938 2401.546      TRUE    Fold5
```

```
model1
```

```
## Linear Regression
##
## 46 samples
##  1 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 36, 37, 37, 38, 36
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   2839.831  0.4370707  2077.933
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
model2$resample
```

```
##       RMSE   Rsquared      MAE intercept Resample
## 1 2009.643 0.05712051 1759.780      TRUE    Fold1
## 2 2185.218 0.69153383 1922.647      TRUE    Fold2
## 3 2585.501 0.54738826 2367.472      TRUE    Fold3
## 4 3217.114 0.80693551 2152.354      TRUE    Fold4
## 5 2670.368 0.38707576 2544.958      TRUE    Fold5
```

```
model2
```

```
## Linear Regression
##
## 46 samples
##  1 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 37, 38, 35, 38, 36
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   2533.569  0.4980108  2149.442
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
model3$resample
```

```
##        RMSE  Rsquared      MAE intercept Resample
## 1 2321.2051 0.8887417 1587.201      TRUE    Fold1
## 2  877.2866 0.9908034  735.669      TRUE    Fold2
## 3 1483.6305 0.2393518 1360.295      TRUE    Fold3
## 4 1534.4014 0.8304601 1336.389      TRUE    Fold4
## 5 1701.9966 0.7297134 1403.458      TRUE    Fold5
```

```
model3
```

```
## Linear Regression
##
## 46 samples
##  1 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 36, 38, 37, 36, 37
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   1583.704  0.7358141  1284.602
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

model1: Ajusted R-squared of 0.57 model2: Ajusted R-squared of 0.50 *model3: Ajusted R-squared of 0.79

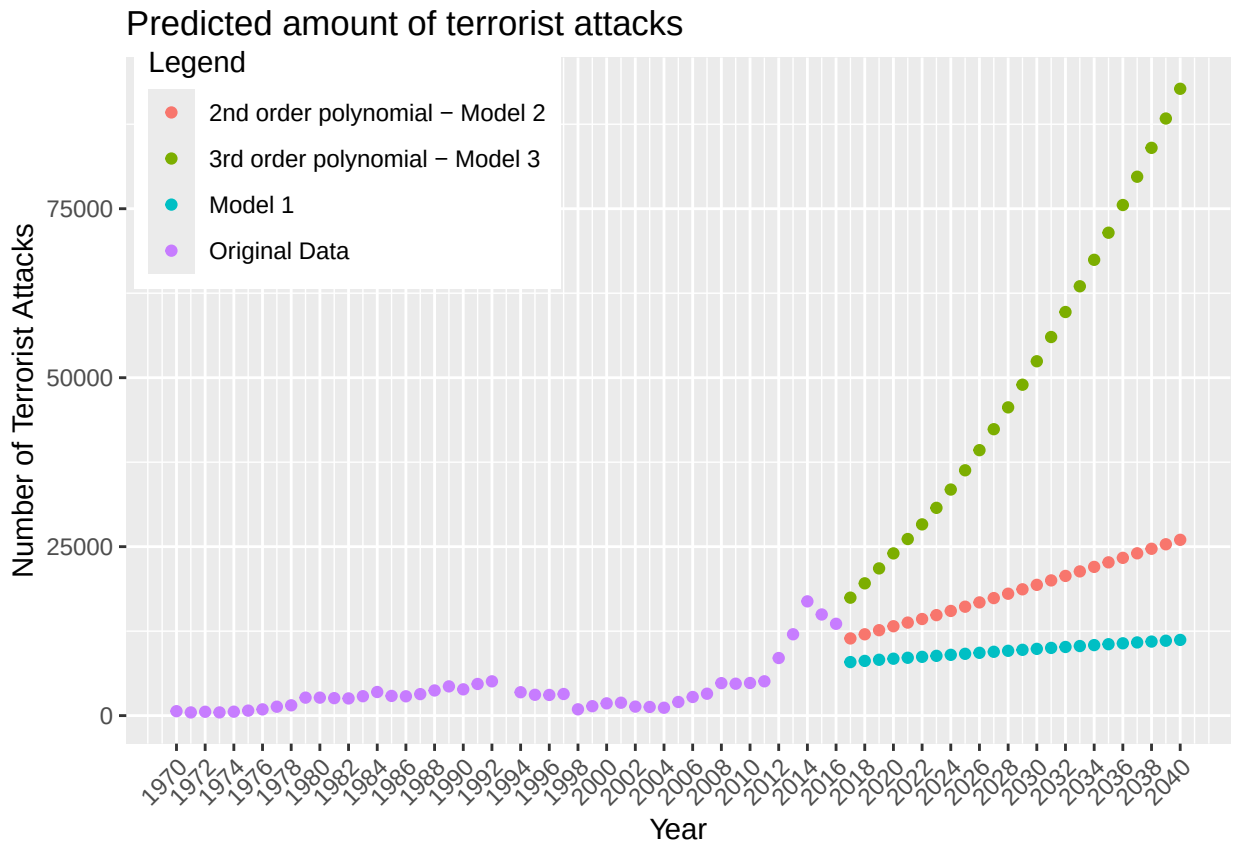
The adjust r-squared has shifted up for model 1 (0.4 -> 0.49) while model 2 and model 3 shifted down (0.54 -> 0.50 and 0.84 -> 0.79 respectively)

###3.4.2 standard forecasting

We can try using our slightly overfitted models to predict the amount of future terrorism attacks based on future PopTotal given by the United Nations predicted population dataset.

```
#First make a dataframe with empty terrorist values with the project UN populations
futurepopworld <- futurepop %>%
  filter(Location == "World") %>%
  select(-Location)

futurepopworld$terrorist_attacks_count <- NA

#duplicate for three models
fpopworld1 <- futurepopworld
fpopworld2 <- futurepopworld
fpopworld3 <- futurepopworld

#predict new values of terrorist_attacks_count based on the three models.
fpopworld1$terrorist_attacks_count <- predict(object=m1, newdata=fpopworld1)

fpopworld2$terrorist_attacks_count <- predict(object=m2, newdata=fpopworld2)

fpopworld3$terrorist_attacks_count <- predict(object=m3, newdata=fpopworld3)

#filter until 2040
fpopworld1 <- filter(fpopworld1, Time < 2041)
fpopworld2 <- filter(fpopworld2, Time < 2041)
fpopworld3 <- filter(fpopworld3, Time < 2041)

#visual
ggplot() +
  geom_point(data=df3, aes(x=year, y=terrorist_attacks_count, col='Original Data')) +
  geom_point(data=fpopworld1, aes(x=Time, y=terrorist_attacks_count, col='Model 1')) +
```

```
  geom_point(data=fpopworld2, aes(x=Time, y=terrorist_attacks_count, col='2nd order polynomial - Model 1
  geom_point(data=fpopworld3, aes(x=Time, y=terrorist_attacks_count, col='3rd order polynomial - Model 3
  labs(title='Predicted amount of terrorist attacks') +
  theme(legend.position = c(0.2, 0.85)) +
  labs(x= 'Year', y= 'Number of Terrorist Attacks', colour = 'Legend') +
  scale_x_continuous(breaks = seq(1970, 2040, 2)) +
  theme(axis.text.x= element_text(angle=45, hjust=1))
```



As can be confirmed in the graph the 2nd order polynomial seems to be giving the most accurate prediction based on the upswing from the past decades but assuming that's just a blip in the data the first model gives the best prediction. To test this correctly we would have to compare it with new data coming in for the coming years. However, with ISIL currently being 'defeated' we suspect that the Model 1 prediction will be most correct.

Besides all that it should also be noted that due to the fact that the three models are all descriptive regression models made with the goal of obtaining a best fit, *all* predictions based on data outside of the range of the original data are pure extrapolation. This can be useful for short term prediction but in actuallity this falls apart quite fast with extremely wide confidence intervals. Due to this we decided to use the 'forecast' package to forecast the data as pure time-series data to forecast percentage changes instead of raw numbers.

###3.4.3 Time-Series Forecasting

To do this we first have to transform the dataset into a time-serie frame. Let's also impute the missing year (1993) with an average value of the preceding and following year so that we can convert to time-series.

```
#for the missing value
df3[df3$year == 1992 | df3$year == 1994, ]
```

```
## # A tibble: 2 x 13
##    year terrorist_attacks_count SortOrder Notes ISO3_code ISO2_code SDMX_code
##   <dbl>                   <int>     <dbl> <chr> <chr>     <chr>         <dbl>
## 1  1992                    5071         1 <NA>  <NA>      <NA>              1
## 2  1994                    3456         1 <NA>  <NA>      <NA>              1
## # i 6 more variables: LocTypeID <dbl>, LocTypeName <chr>, ParentID <dbl>,
## #   PopTotal <dbl>, PopDensity <dbl>, decade <fct>
```

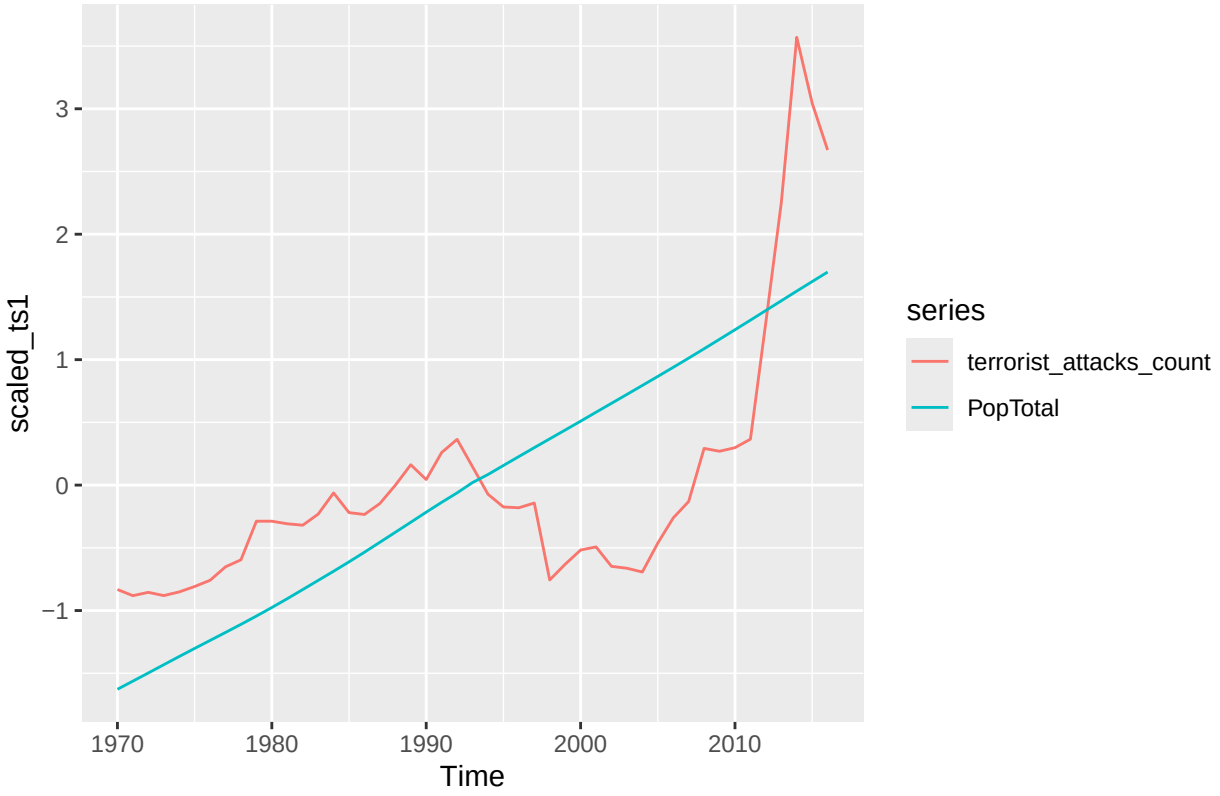
```
newrow <- data_frame(year=1993, terrorist_attacks_count=(5073+3458)/2-0.5, PopTotal=(5504401+5670320)/2)
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## i Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

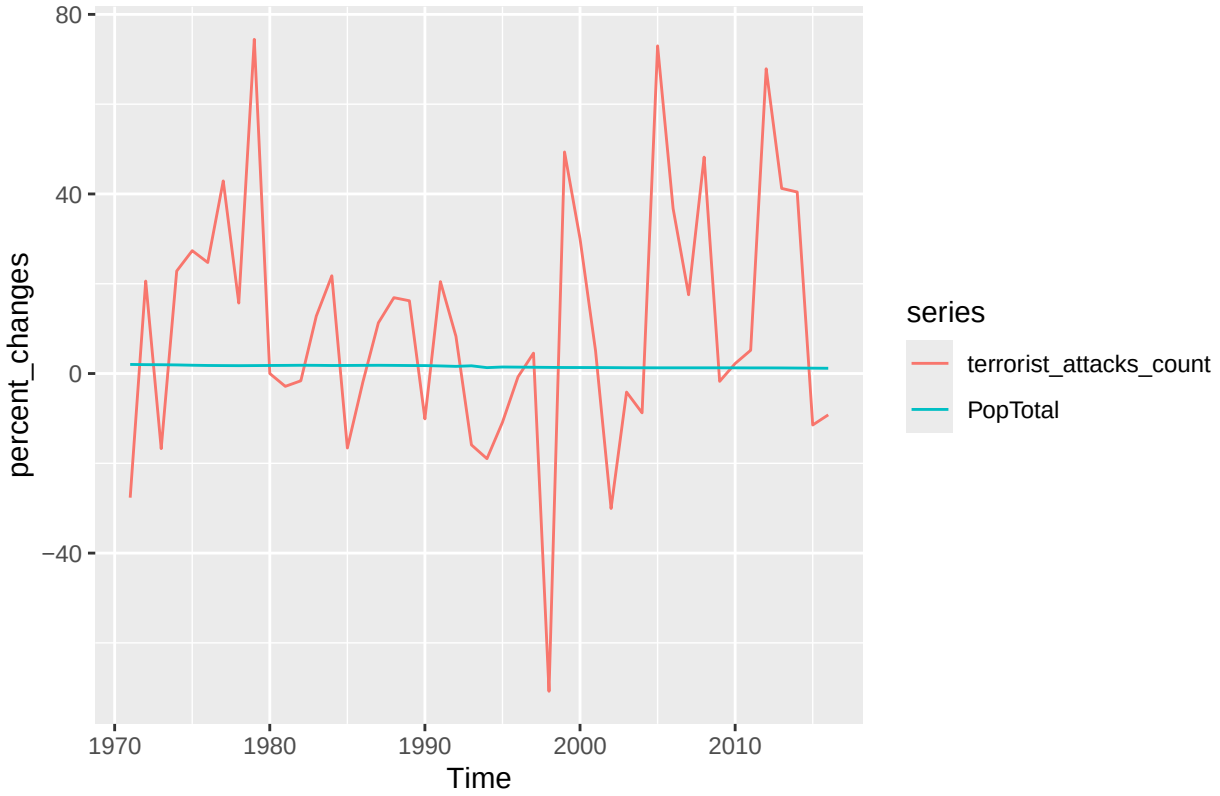
```
df4 <- df3 %>%
  select(-decade) %>%
  bind_rows(newrow)

df4 <- df4 %>%
  arrange(year)

values <- df4[, c(2,11)]

ts1 <- ts(values, start=1970,end=2016,frequency=1)

scaled_ts1 <- scale(ts1)

percent_changes <- diff(ts1)/ts1[-nrow(ts1),] * 100
```

So now we have a time-series data frame with percent changes. Let's visualize how that looks. Because Ggplot can't out of the box handle ts objects we're using ggfortify.

```
autoplot(scaled_ts1)
```

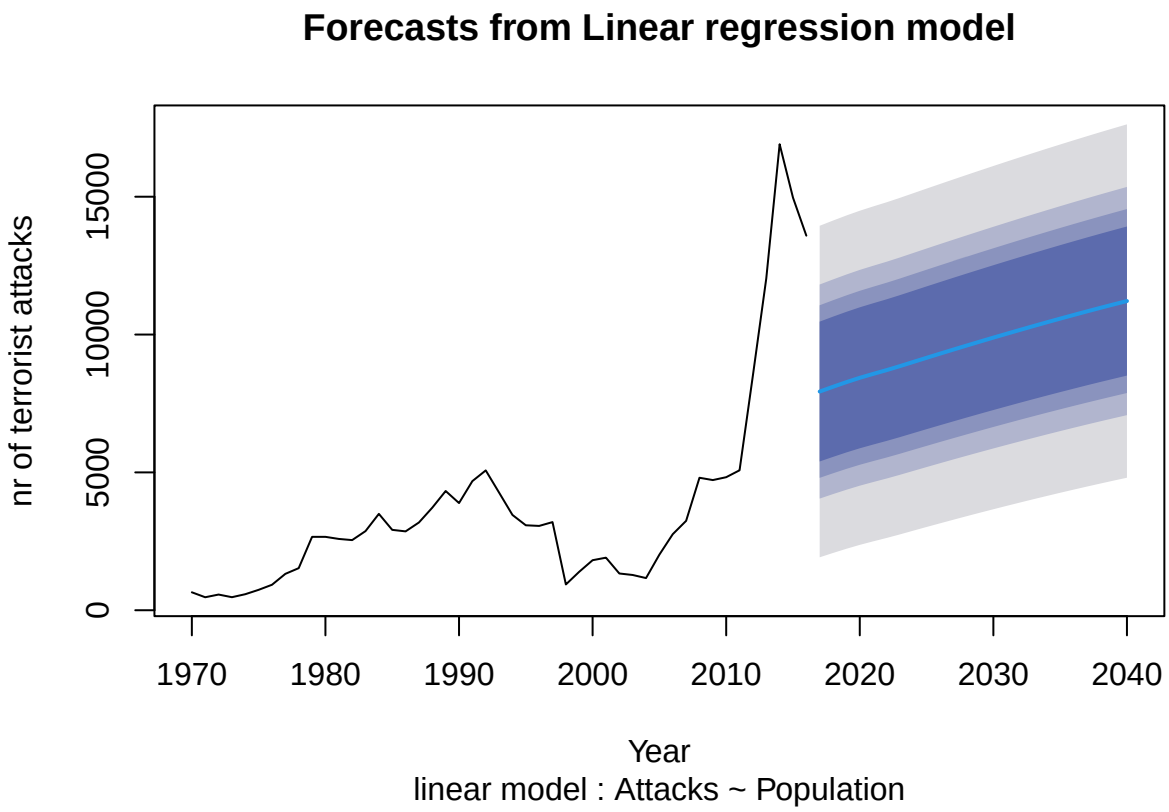



```
autoplot(percent_changes)
```

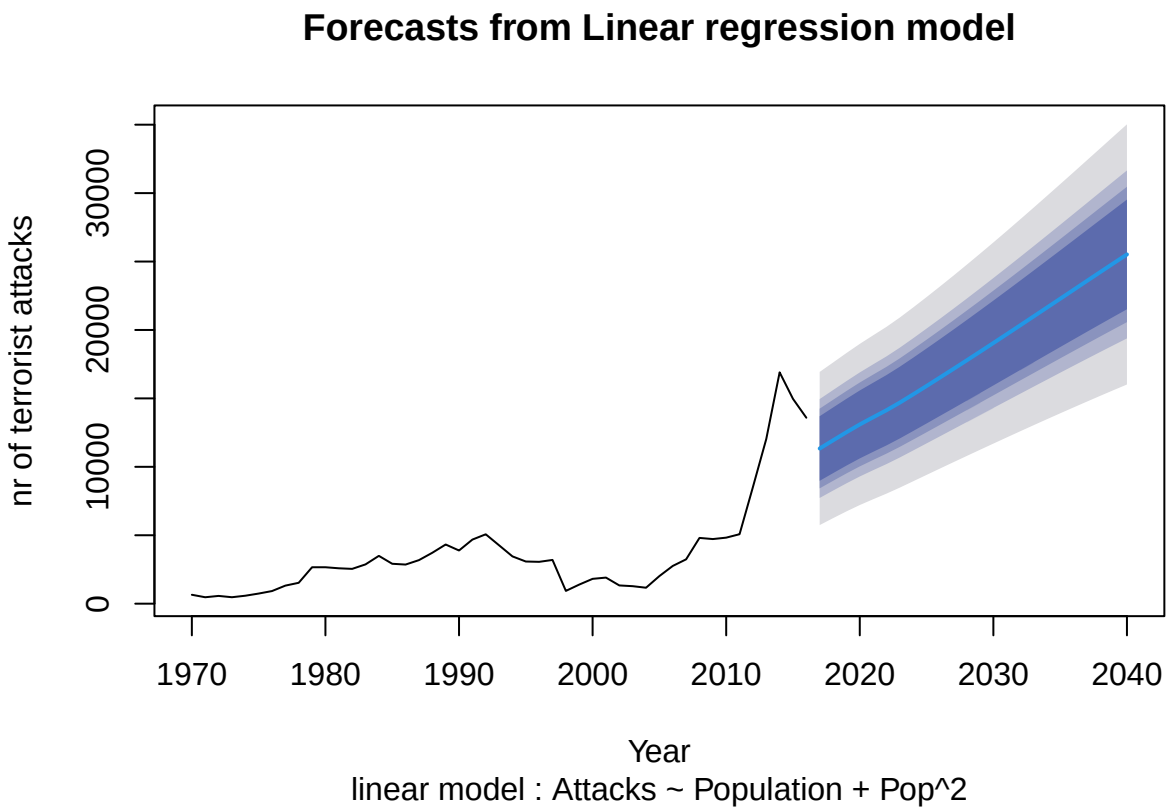


Let's now use the forecast package to forecast this time series. First we need to build a time series linear model.

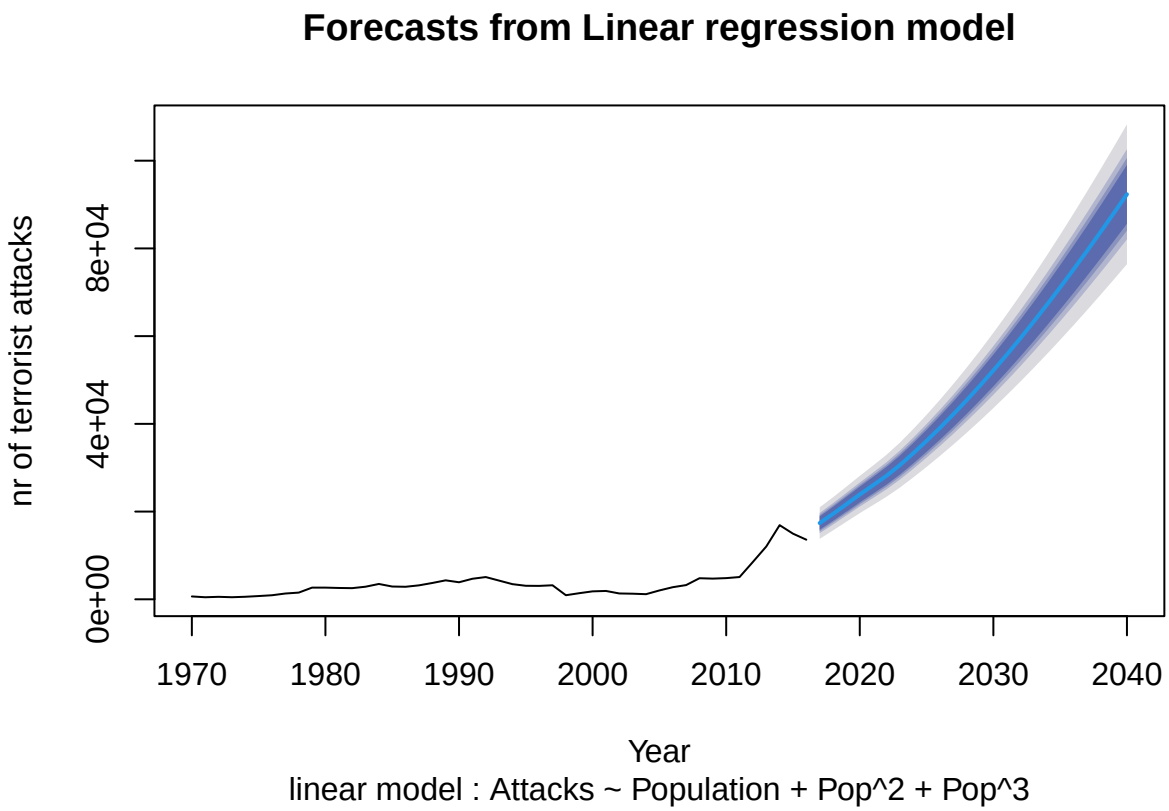
```
tm1 <- tslm(terrorist_attacks_count ~ PopTotal, data=ts1)

tm2 <- tslm(terrorist_attacks_count ~ PopTotal + I(PopTotal^2), data=ts1)

tm3 <- tslm(terrorist_attacks_count ~ PopTotal + I(PopTotal^2) + I(PopTotal^3), data=ts1)

futurepopworld4 <- futurepopworld %>%
  filter(Time < 2041)

f1 <- forecast(tm1, newdata=futurepopworld4, level=c(60, 70, 80, 95))
f2 <- forecast(tm2, newdata=futurepopworld4, level=c(60, 70, 80, 95))
f3 <- forecast(tm3, newdata=futurepopworld4, level=c(60, 70, 80, 95))

plot(f1, ylab = 'nr of terrorist attacks', xlab = 'Year', sub = 'linear model : Attacks ~ Population')
```




```
plot(f2, ylab = 'nr of terrorist attacks', xlab = 'Year', sub = 'linear model : Attacks ~ Population + I
```




```
plot(f3, ylab = 'nr of terrorist attacks', xlab = 'Year', sub = 'linear model : Attacks ~ Population + P
```



These three forecasts show the difficulty and issues that arise when forecasting outside of the fitted model. The 4 different colored confidence intervals show how unsure the forecast is. From the outside in the confidence intervals for the colored bands are: 95%, 80%, 70% and 60%.