MAE 158 Extra Credit

By Sam Hince ID: 19295309 03/02/2021

Directions:

The main project involves developing a tool (code) to calculate aircraft performance given information about the aircraft geometry and flight mission requirements. The plots should be digitized, and the code should be able to run without needing supplemental inputs from the charts, from the user.

You need to develop the ability for calculate performance of the aircraft at any airspeed. You will need to be able to generate plots of 'Thrust required versus airspeed', 'Power required versus airspeed', and 'L/D versus airspeed'. The following page shows and example of the charts.

For the aircraft you will be given:

- Wing geometry
  - Sweep
  - Thickness
  - Span
- Fuselage geometry
  - Length
  - Diameter
  - Wetted area
- Tail geometry
  - Similar to wing
- Additional drag contributions
  - Additional flat-plate drag area

You should be able to calculate: + Total Drag (at any airspeed) + Induced Drag - Coefficient of induced drag + Parasitic Drag - Flat-plate drag area - Coefficient of parasitic drag

---

Part 1

Inputs for the aircraft given in problem 11.1 from Shevell:

```r
# Wing geometry
wing <- list(span = 93.2,
             Sref = 1000,
             tc = 0.106,
             sweep = 24.5,
             taper = 0.2,
             c_root = 17.8,
             S_wet = (1 - 0.17) * 1000 * 1.02 * 2)
```

```r
# Fuselage geometry
fuselage <- list(length = 107,
                 diameter = 11.5,
                 S_wet = 3280)

# Tail geometry
horizontal_tail <- list(S_wet = 261 * 1.02 * 2,
                        tc = 0.09,
                        sweep = 31.6,
                        taper = 0.35,
                        c_root = 11.1)

vertical_tail <- list(S_wet = 161 * 1.02 * 2,
                      tc = 0.09,
                      sweep = 43.5,
                      taper = 0.80,
                      c_root = 15.5)

# Additional items
pylons <- list(S_wet = 117 * 1.02 * 2,
               tc = 0.06,
               sweep = 0,
               taper = 1,
               c_root = 16.2)

nacelles <- list(S_wet = 455,
                 fineness_ratio = 5,
                 length = 16.8)

farirings <- list(delta_f = 0.15)
```

Also given are the conditions in which the jet is operating:

```r
env <- list(alt = 31000,
            P = 601.61,
            mu = 3.10945e-7,
            mach = 0.78,
            temp = 399.67, # converted to R
            W = 89000)

env$rho = env$P / (1718 * env$temp)

env$V <- sqrt(1.4 * 1718 * env$temp) * env$mach
```

Only altitude was given and other parameters were calculated using: https://www.digitaldutch.com/atmoscalc/

First let us perform total drag calculations. I will begin the process of solving for the parasitic drag for winglike surfaces by finding the Reynolds number and Cf for each component:

```r
# Find MAC for each element
c_bar <- function(c_root, taper){
  c_bar <- (2/3) * wing$c_root * (1 + wing$taper - (wing$taper/(1 + wing$taper)))
  return(c_bar)
}

wing$c_bar <- c_bar(wing$c_root, wing$taper)
horizontal_tail$c_bar <- c_bar(horizontal_tail$c_root, horizontal_tail$taper)
vertical_tail$c_bar <- c_bar(vertical_tail$c_root, vertical_tail$taper)
pylons$c_bar <- c_bar(pylons$c_root, pylons$taper)

# Find Re:
wing$Re <- (env$rho * env$V * wing$c_bar) / env$mu
horizontal_tail$Re <- (env$rho * env$V * horizontal_tail$c_bar) / env$mu
vertical_tail$Re <- (env$rho * env$V * vertical_tail$c_bar) / env$mu
pylons$Re <- (env$rho * env$V * pylons$c_bar) / env$mu

# find Cf
c_f <- function(Re){
  c_f <- (0.208 * Re^(-0.27845)) + 0.00101
  return(c_f)
}

wing$c_f <- c_f(wing$Re)
horizontal_tail$c_f <- c_f(horizontal_tail$Re)
vertical_tail$c_f <- c_f(vertical_tail$Re)
pylons$c_f <- c_f(pylons$Re)
```

In order to find the drag coefficients for each component we need to find the form factor correction k. This is given by a pair of equations:

```r
form_factor <- function(mach, lambda, tc){
  z <- ((2 - (mach^2)) * cos(lambda))/(sqrt(1-((mach^2) * cos(lambda))))
  form_factor <- 1 + (z * tc) + (100 * (tc^4))
  return(form_factor)
}

wing$k <- form_factor(env$mach, wing$sweep, wing$tc)
horizontal_tail$k <- form_factor(env$mach, horizontal_tail$sweep, horizontal_tail$tc)
vertical_tail$k <- form_factor(env$mach, vertical_tail$sweep, vertical_tail$tc)
pylons$k <- form_factor(env$mach, pylons$sweep, pylons$tc)
```

Next a similar series of calculations is needed for the fuselage elements. In this case k was found from a plot. The plot was digitized useing a tool found at: https://apps.automeris.io/wpd/

```r
# Find Re:
fuselage$Re <- (env$rho * env$V * fuselage$length) / env$mu
nacelles$Re <- (env$rho * env$V * nacelles$length) / env$mu

# find Cf
c_f <- function(Re){
  c_f <- (0.208 * Re^(-0.27845)) + 0.00101
  return(c_f)
}

fuselage$c_f <- c_f(fuselage$Re)
```

```r
nacelles$c_f <- c_f(nacelles$Re)

# form factor calculations (using digitized plot)
setwd("/home/sam/Documents/classGitRepos/MAE158")
fusalage_k_plot <- read.csv(file = "fuselage_form.csv", header = TRUE)

fuselage$fineness_ratio <- fuselage$length / fuselage$diameter

fuselage$k <- approx(x = fusalage_k_plot$Fineness.Ratio,
                     y = fusalage_k_plot$k,
                     xout = fuselage$fineness_ratio,
                     method="linear",
                     ties=min)$y
nacelles$k <- approx(x = fusalage_k_plot$Fineness.Ratio,
                     y = fusalage_k_plot$k,
                     xout = nacelles$fineness_ratio,
                     method="linear",
                     ties=min)$y
```

Finally we just need to find Cdp for each component and sum it all up:

```r
# wing components:
wing$delta_f <- wing$k * wing$c_f * wing$S_wet
horizontal_tail$delta_f <- horizontal_tail$k * horizontal_tail$c_f * horizontal_tail$S_wet
vertical_tail$delta_f <- vertical_tail$k * vertical_tail$c_f * vertical_tail$S_wet
pylons$delta_f <- pylons$k * pylons$c_f * pylons$S_wet

wing$delta_Cdp <- wing$delta_f / wing$Sref
horizontal_tail$delta_Cdp <- horizontal_tail$delta_f / wing$Sref
vertical_tail$delta_Cdp <- vertical_tail$delta_f / wing$Sref
pylons$delta_Cdp <- pylons$delta_f / wing$Sref

# fuselage components:
fuselage$delta_f <- fuselage$k * fuselage$c_f * fuselage$S_wet
nacelles$delta_f <- nacelles$k * nacelles$c_f * nacelles$S_wet

fuselage$delta_Cdp <- fuselage$delta_f / wing$Sref
nacelles$delta_Cdp <- nacelles$delta_f / wing$Sref

# and don't forget the fairings
farirings$delta_Cdp <- farirings$delta_f / wing$Sref

# sum it all up:
f <- wing$delta_f +
  horizontal_tail$delta_f +
  vertical_tail$delta_f +
  pylons$delta_f +
  fuselage$delta_f +
  nacelles$delta_f +
  farirings$delta_f
Cdp <- wing$delta_Cdp +
  horizontal_tail$delta_Cdp +
  vertical_tail$delta_Cdp +
  pylons$delta_Cdp +
  fuselage$delta_Cdp +
  nacelles$delta_Cdp +
  farirings$delta_Cdp

print((sprintf("Flat plate drag area: %g", f)))
```

```
## [1] "Flat plate drag area: 18.1217"
```

```r
print((sprintf("Total parasitic drag coeficient: %g", Cdp)))
```

```
## [1] "Total parasitic drag coeficient: 0.0181217"
```

In his discussion notes Colin multiplied these values by 1.10 in order to account for any additional drag contributions not explicitly calculated. I have decided to proceed with the numbers attained above assuming they are complete and accurate.

Next we have been asked to find max range and the corresponding max airspeed. In order to perform there calculations we need to input some more information about the aircraft specifications

```r
specs <- list(Wi = 97000,
              Wf = 82000,
              ct = 0.82,          # specific fuel consumption in lb/lb-h
              fuel_weight = 6/7) # lb / gal

specs$Cdp <- Cdp
```

Next we need valued for Cl and Cdi. Lets begin with Cl

```r
env$q <- (1.4 / 2) * env$P * (env$mach^2)

specs$Cl <- env$W / (env$q * wing$Sref)
```

To find Cdi I will be taking Colin's recommendation and finding e for both 30 and 0 degrees of sweep then interpolating between these values.

```r
# find k for each component
e_func <- function(AR, lambda){
  e_nosweep <- (1.78 * (1 - (0.045 * (AR^0.68)))) - 0.64
  e_30deg <- (4.61 * (1 - (0.045 * (AR^0.68))) * ((cos(lambda))^0.15)) - 3.1

  # interpolate between
  e_func <- e_nosweep + ((lambda/30) * (e_30deg - e_nosweep))

  return(e_func)
}

wing$AR <- wing$span^2 / wing$Sref
```

```r
e <- e_func(wing$AR, wing$sweep)

specs$Cdi <- ((specs$Cl)^2) / (pi * wing$AR * e)

specs$Cd <- specs$Cdp + specs$Cdi
```

Now we can find the maximum range and corresponding velocity. See Lecture 13 for equations and explanation. For the airspeed to maximize range, this will obviously vary over the course of the flight so I have used the average weight. Integrating and finding the true average may be more accurate over a full flight.

```r
specs$max_range <- (2 / specs$ct) * sqrt(2 / (env$rho * wing$Sref)) * (sqrt(specs$Cl) / (specs$Cd)) * (s

specs$W_avg <- (specs$Wi + specs$Wf) / 2
k <- 1 / (pi * wing$AR * e)
specs$V_max_range <- sqrt((2/env$rho) * sqrt((3 * k)/specs$Cdp) * (specs$W_avg/wing$Sref))

print((sprintf("Maximum range: %g Nautical miles", specs$max_range)))
```

```
## [1] "Maximum range: 2077.58 Nautical miles"
```

```r
print((sprintf("Velocity to achieve maximum range: %g ft/s", specs$V_max_range)))
```

```
## [1] "Velocity to achieve maximum range: 825.479 ft/s"
```

Next we perform calculations for endurance and airspeed to achieve maximum endurance. The airspeed corresponds with L/D max. I found a video that was helpful for reviewing this topic: https://www.youtube.com/watch?v=RyCcKV8puSE. For calculating the velocity I once again used the average velocity for simplicity.

```r
specs$E <- (1/specs$ct) * (specs$Cl/specs$Cd) * log(specs$Wi/specs$Wf)

specs$V_max_E <- sqrt(((2 * specs$W_avg) / (env$rho * wing$Sref)) * sqrt(k / specs$Cd))

print((sprintf("Maximum endurance: %g Hours", specs$E)))
```

```
## [1] "Maximum endurance: 2.71304 Hours"
```

```r
print((sprintf("Velocity to achieve maximum endurance: %g ft/s", specs$V_max_E)))
```

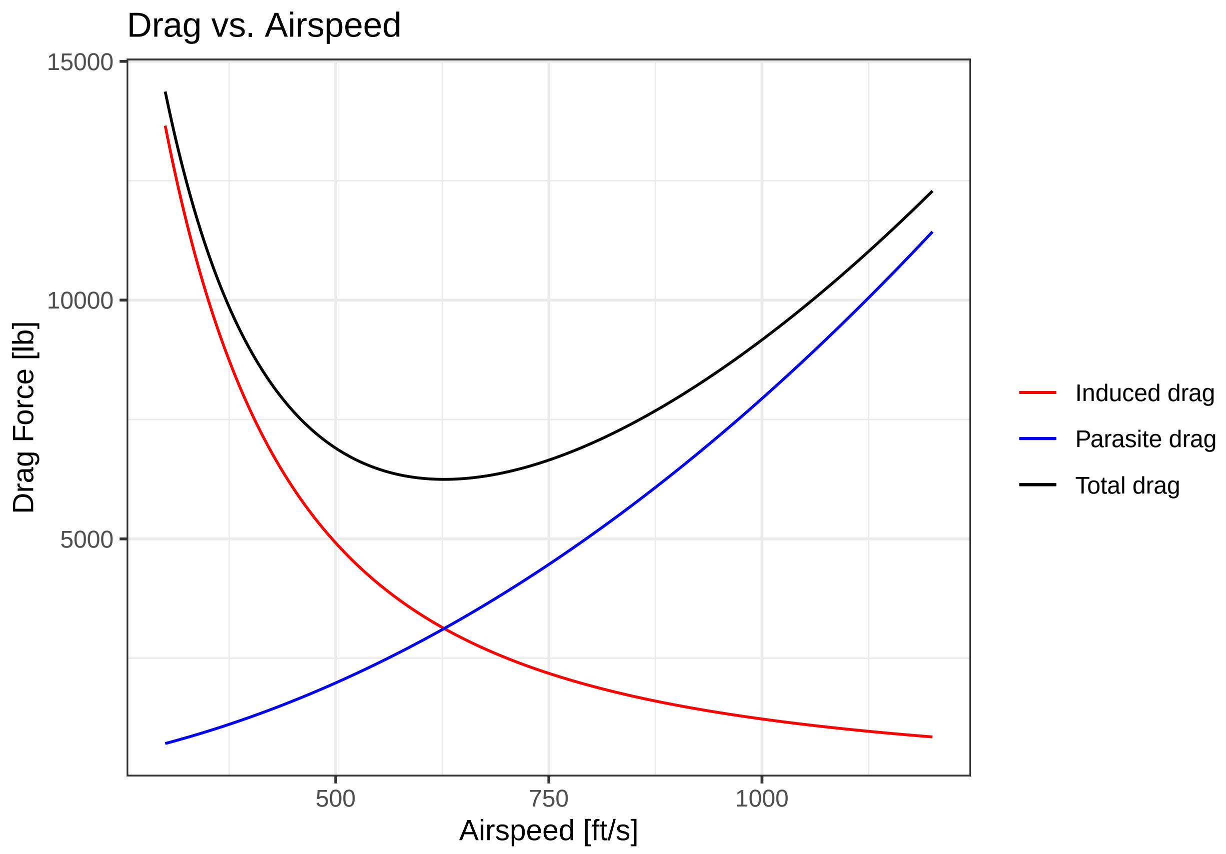
```
## [1] "Velocity to achieve maximum endurance: 571.84 ft/s"
```

Finally we must create the desired plots. For this we will use the R package ggplot2. Let us begin with a plot of drag vs. airspeed:

```r
library(ggplot2)

df <- data.frame(airspeed = 300:1200)
df$q <- (1/2) * env$rho * (df$airspeed^2)
df$Cl <- specs$W_avg / (df$q * wing$Sref)
df$Cdi <- (df$Cl^2) * k
df$inducedDrag <- df$q * wing$Sref * df$Cdi
df$parasiteDrag <- df$q * wing$Sref * specs$Cdp
df$totalDrag <- df$parasiteDrag + df$inducedDrag

pl <- ggplot(df, aes(x = airspeed)) +
  geom_path(aes(y = inducedDrag, color = "Induced drag")) +
  geom_path(aes(y = parasiteDrag, color = "Parasite drag")) +
  geom_path(aes(y = totalDrag, color = "Total drag")) +
  scale_colour_manual("",
                      breaks = c("Induced drag", "Parasite drag", "Total drag"),
                      values = c("red", "blue", "black")) +
  ggtitle("Drag vs. Airspeed") + # for the main title
  xlab("Airspeed [ft/s]") + # for the x axis label
  ylab("Drag Force [lb]") + # for the y axis label
  theme_bw()
print(pl)
```



Note that this analysis does not account for compressibility.

Finally, lets make the plot of Power required, Power Available, and Rate of Climb over a series of airspeeds. We will use a fresh dataframe in order to be able to adjust the plot bound independently of the previous plot.

Colin sent an email indicating that thrust should be 14500 for the aircraft we are analysing.

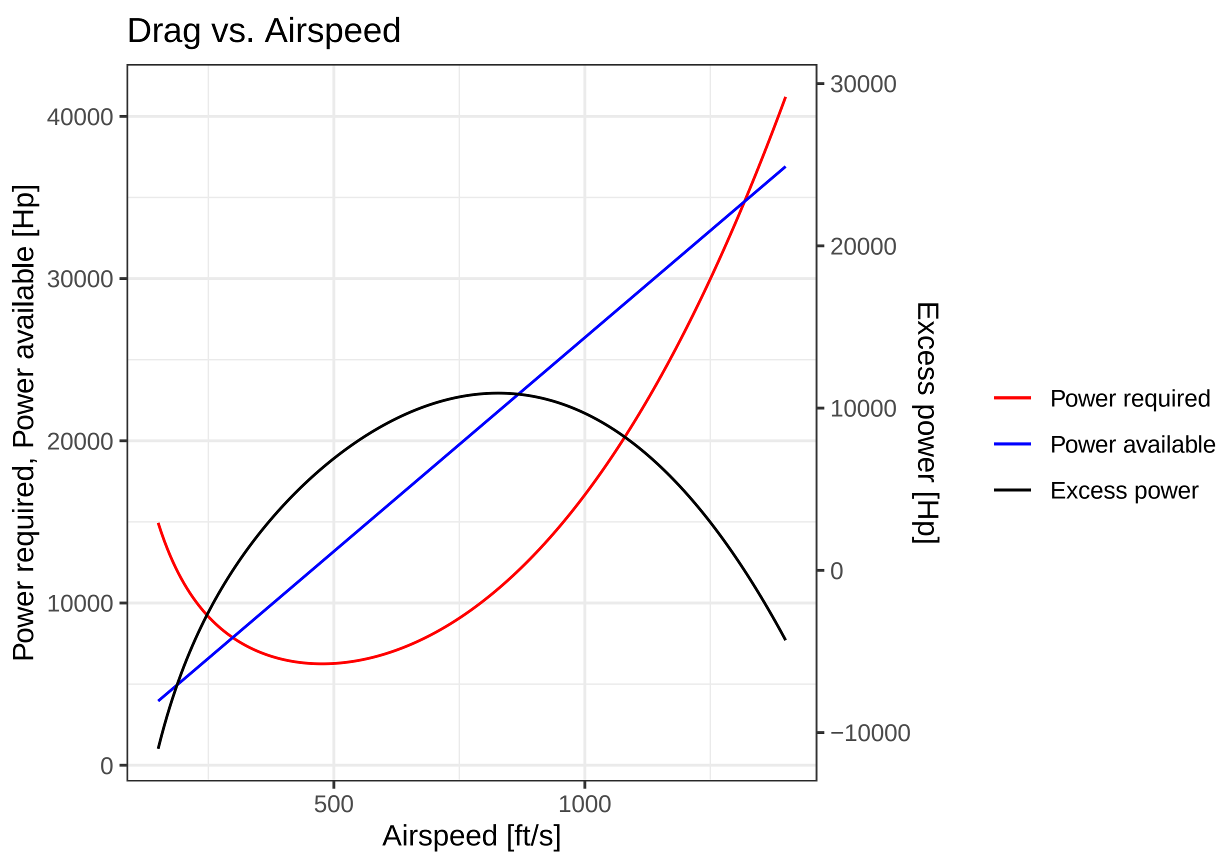
```r
specs$T_static <- 14500 #* (env$rho / 0.002377) # air density ratios

df <- data.frame(airspeed = 150:1400)
df$q <- (1/2) * env$rho * (df$airspeed^2)
df$Cl <- specs$W_avg / (df$q * wing$Sref)
df$Cdi <- (df$Cl^2) * k
df$Cd <- specs$Cdp + df$Cdi
df$Pr <- sqrt((2 * specs$W_avg^3) / (env$rho * wing$Sref)) * (1 / ((df$Cl^(3/2)) / df$Cd)) / 550
df$Pa <- (specs$T_static*df$airspeed) / 550
df$Ep <- df$Pa - df$Pr

coef <- 12000

pl <- ggplot(df, aes(x = airspeed)) +
  geom_path(aes(y = Pr, color = "Power required")) +
  geom_path(aes(y = Pa, color = "Power available")) +
  geom_path(aes(y = Ep + coef, color = "Excess power")) +
  scale_y_continuous(
    # Features of the first axis
    name = "Power required, Power available [Hp]",

    # Add a second axis and specify its features
    sec.axis = sec_axis(~.-coef, name="Excess power [Hp]")) +
  scale_colour_manual("",
                      breaks = c("Power required", "Power available", "Excess power"),
                      values = c("red", "blue", "black")) +
  ggtitle("Drag vs. Airspeed") + # for the main title
  xlab("Airspeed [ft/s]") + # for the x axis label
  ylab("Drag Force [lb]") + # for the y axis label
  theme_bw()
print(pl)
```

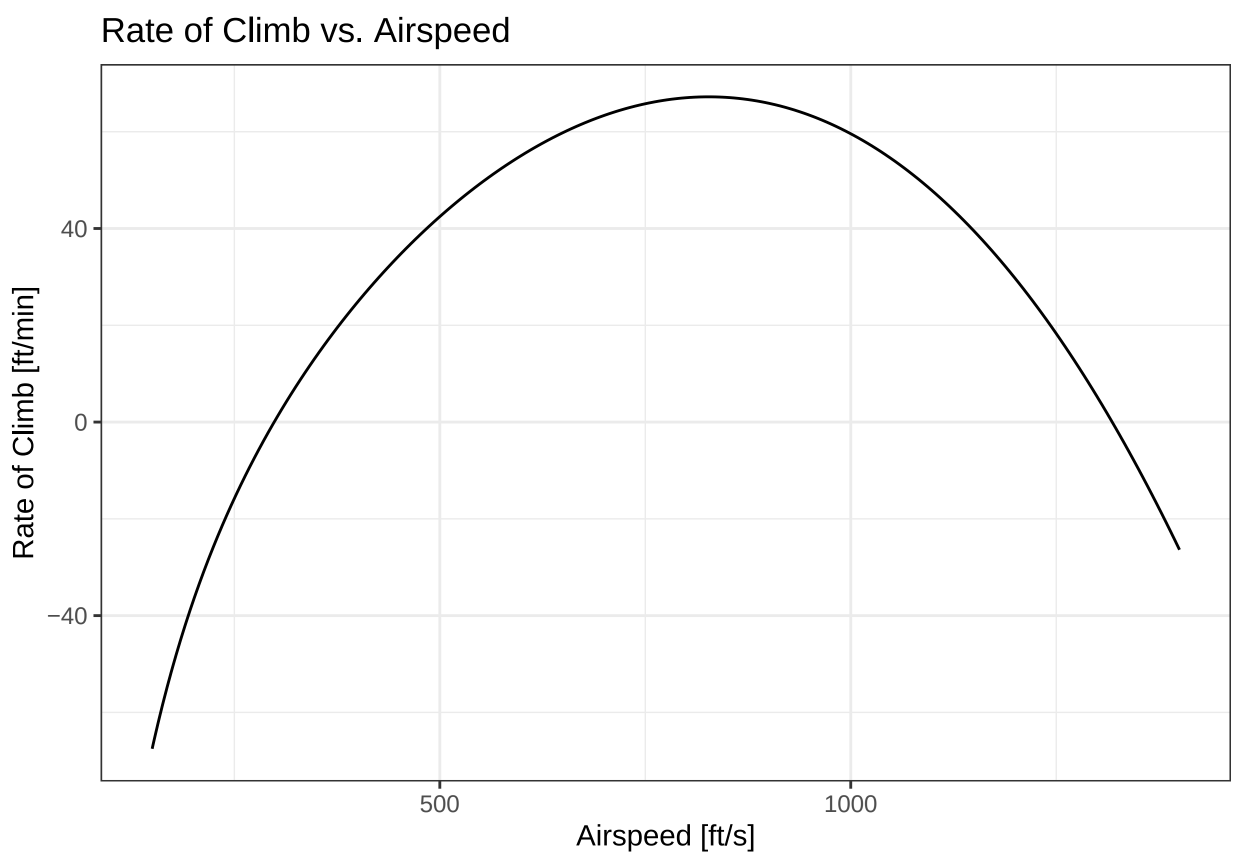


Power equations can be found in lecture 11.

Building from the previous plot we can plot rate of climb at different airspeeds

```
df$ROC <- (df$Ep * 550) / specs$W_avg

pl <- ggplot(df, aes(x = airspeed)) +
  geom_path(aes(y = ROC)) +
  ggtitle("Rate of Climb vs. Airspeed") + # for the main title
  xlab("Airspeed [ft/s]") + # for the x axis label
  ylab("Rate of Climb [ft/min]") + # for the y axis label
  theme_bw()
print(pl)
```



Keep in mind that this plot takes place at 31000 ft, therefore the relatively limited excess power and rate of climb makes logical sense.

This concludes part 1 of my report.