# windmillAnalysis.R

sam

2021-03-13

```r
# Windmill analysis
# Script created by Sam Hince
# 01/11/2021

library(varhandle)
library(rjson)
library(caTools)
library(ggplot2)

rm(list = ls())

##########################################################################

# nomenclature:
# V = velocity of aircraft
# phi = angle
# Omega = angular
# B = number of blade
# r = radius along blade

##########################################################################

convergence_minimum <- 0.00001
feathering <- "none" # thrust, power, none

##########################################################################

## read in data from json
setwd("/home/sam/Documents/classGitRepos/MAE195")
geom <- fromJSON(file = './propSpecs/DesignOutput.json') #
coef <- read.csv(file = './propSpecs/NACA4415_RN500K_NCRIT9.csv', header = TRUE)

##########################################################################

## calculate some constants
mu <- geom$alt$kinematicViscosity * geom$alt$density   # N s/m^2 dynamic viscosity of the fluid
rho <- geom$alt$density                                 # kg/m^3 density of air
B <- geom$blades                                        # number of blades
R <- geom$diameter / 2                                  # radius along the prop
V <- geom$velocity                                      # m / s volocity
Omega <- (2 * pi * geom$RPM) / 60                       # rad / s angular velocity of the propeller
```

```r
##########################################################################

## ass-umptions
beta <- 20 * (pi / 180) # blade angle is this given???

##########################################################################

# equation 1:
eqn_phi <- function(V, a, Omega, r, aprime){
  phi <- atan2((V*(1+a)), (Omega*r*(1-aprime))) # should this be atan2?
  return(phi)
}

eqn_phi_estimate <- function(V, Omega, r){
  phi <- atan2(V, (Omega*r)) # initial phi condition (step 1)
  return(phi)
}

# quations 2:
eqn_alpha <- function(beta, phi){
  alpha <- phi - beta
  return(alpha)
}

eqn_W <- function(V, a, phi){
  W <- (V * (1 + a)) / sin(phi)
  return(W)
}

eqn_Re <- function(rho, W, c, mu){
  Re <- rho * W * (c / mu)
  return(Re)
}

# equation 3:
eqn_Cx <- function(Cl, Cd, phi){
  Cx <- (Cl * sin(phi)) + (Cd * cos(phi))
  return(Cx)
}

eqn_Cy <- function(Cl, Cd, phi){
  Cy <- (Cl * cos(phi)) - (Cd * sin(phi))
  return(Cy)
}

# equation 4:
eqn_a <- function(sigma, varF, Cy, phi){
  a <- ((sigma/(4*varF))*(Cy/(sin(phi)^2))) / (1-(sigma/(4*varF)*(Cy/(sin(phi)^2))))
  return(a)
}

eqn_aprime <- function(sigma, varF, Cx, phi){
  aprime <- ((sigma/(4*varF))*(Cx/(sin(phi)*cos(phi)))) / (1+(sigma/(4*varF)*(Cx/(sin(phi)*cos(phi)))))
```

```
  return(aprime)
}

##########################################################################



### complex plotting loop ###
#pl_df <- data.frame()
#og_thrust <- geom$thrust
#for(i in seq(1,10)){
#  geom$thrust <- og_thrust * (1 + (i * 0.05))

#pl_df <- data.frame()
#for(i in seq(1,30)){
#  geom$RPM <- geom$RPM + 50
#  Omega <- (2 * pi * geom$RPM) / 60

#pl_df <- data.frame()
#V <- V - 40
#og_thrust <- geom$thrust
#for(i in seq(1,10)){
#  V <- V + 10

#pl_df <- data.frame()
#og_B <- B
#for(i in seq(1,10)){
#  B <- og_B + i
#  geom$blades <- og_B + i

keep_feathering <- TRUE
total_d_beta <- 0
while(keep_feathering){ # feathering loop

  df <- data.frame()

  for (station in 1:length(geom$radialStation)){
    #separate our values for this loop
    r <- geom$radialStation[station]
    Xi <- r/R
    c <- geom$chord[station]
    beta <- geom$beta[station] * (pi/180) # convery to rad

    # correction for final blade station:
    if(Xi >= 1){
      Xi <- 1 - 1e-15
    }

    #print(sprintf("Blade station: %g", r))

    #step 1
    # initial assumptions:
    phi <- eqn_phi_estimate(V, Omega, r)
```

```r
a <- 0
aprime <- 0

convergent <- FALSE
loops_to_converge <- 0
while(convergent == FALSE){
  #step 2
  alpha <- eqn_alpha(beta, phi)
  W <- eqn_W(V, a, phi)
  Re <- eqn_Re(rho, W, c, mu)

  #step 3
  alpha_degrees <- alpha * (180/pi)
  Cl <- - 1 * approx(x = coef$ALPHA, y = coef$CL, xout = alpha_degrees, method="linear")$y
  Cd <- approx(x = coef$ALPHA, y = coef$CD, xout = alpha_degrees, method="linear")$y

  #step 4
  Cx <- eqn_Cx(Cl, Cd, phi)
  Cy <- eqn_Cy(Cl, Cd, phi)

  #step 5
  phit <- atan((Xi)*tan(phi))
  f <- (B/2)*((1-(Xi)) / (sin(phit)))
  varF <- (2/pi) * atan((exp(2*f)-1)^(1/2))

  #step 6
  sigma <- (B*c)/(2*pi*r) # local solidity
  a <- eqn_a(sigma, varF, Cy, phi)
  aprime <- eqn_aprime(sigma, varF, Cx, phi)

  # corrections for finite tip chord:
  if((abs(a) > .7) || (abs(aprime) > .7)){
    aprime = 0.4
  }

  #step 7
  phi_new <- eqn_phi(V, a, Omega, r, aprime)

  #step 8
  percent_change <- (abs(phi - phi_new)/phi)

  if(percent_change < convergence_minimum){     # if the percent change has decreased enough
    convergent <- TRUE                          # then assume we have reached convergence
    #print(sprintf("Phi converged in %g loops", loops_to_converge))
  } else {                                      # otherwise increment phi
    phi <- phi + (0.4 * (phi_new - phi))        # 0.4 is variable, recopmended value by liebeck
    loops_to_converge <- loops_to_converge + 1
  }
}

#record data
df <- rbind(df, data.frame(station, r, c, beta * (180/pi), alpha_degrees, phi * (180/pi), Re / 10000
```

```
}

#reformat data
colnames(df) <- c("station", "r", "c", "beta", "alpha", "phi", "Re", "a", "aprime", "f", "F", "W", "C
#print(df)

#step 9
dr <- geom$radialStation[2]-geom$radialStation[1]
n <- geom$RPM / 60

#Ct:
coef_term <- (1/(rho * (n^2) * ((2*R)^4)))

integral <- 0
for(station in 1:(length(geom$radialStation)-1)){
  #separate our values for this loop
  c <- geom$chord[station]
  W <- df$W[station]
  Cy <- df$Cy[station]

  d <- ((1/2) * rho * (W^2) * B * c * Cy * dr)
  integral <- integral + d
}

thrust <- integral
Ct <-  integral * coef_term

#Cp:
coef_term <- (1/(rho * (n^3) * ((2*R)^5)))

integral <- 0
for(station in 1:(length(geom$radialStation)-1)){
  #separate our values for this loop
  r <- geom$radialStation[station] #df$r[station]
  c <- geom$chord[station]
  W <- df$W[station]
  Cx <- df$Cx[station]

  d <- ((1/2) * rho * (W^2) * B * c * Cx * Omega * r * dr) # Omega missing in the notes
  integral <- integral + d
}

power <- integral / 550
Cp <- integral * coef_term

advance_ratio <- V / (n * geom$diameter) # could use geom$J
efficiency <- Ct * advance_ratio / Cp

solidity <- (geom$blades * trapz(geom$radialStation, geom$chord)) / (pi * ((geom$diameter/2)^2))

### feathering stuff ###
if(feathering == "thrust"){
  dif_thrust <- geom$thrust - thrust
```

```r
    delta <- (dif_thrust / abs(dif_thrust)) * 0.01
    print("changing by thrust")
    geom$beta <- geom$beta + delta
    total_d_beta <- total_d_beta + delta

    if((abs(dif_thrust) / geom$thrust) > 0.01){
      keep_feathering <- TRUE
    }else{
      keep_feathering <- FALSE
    }

  }else if(feathering == "power"){
    dif_power <- geom$power - power
    delta <- (dif_power / abs(dif_power)) * 0.01
    print("changing by power")
    geom$beta <- geom$beta + delta
    total_d_beta <- total_d_beta + delta

    if((abs(dif_power) / geom$power) > 0.01){
      keep_feathering <- TRUE
    }else{
      keep_feathering <- FALSE
    }

  }else{
    break
  }
}

### print output ###
print((sprintf("Cp: %g", Cp)))
```

```
## [1] "Cp: -0.0034363"
```

```r
print((sprintf("Ct: %g", Ct)))
```

```
## [1] "Ct: -0.00826219"
```

```r
print((sprintf("Efficiency: %g", efficiency)))
```

```
## [1] "Efficiency: 1.11621"
```

```r
print((sprintf("Advance Ratio: %g", advance_ratio)))
```

```
## [1] "Advance Ratio: 0.464241"
```

```r
print((sprintf("Power (Hp): %g", power)))
```

```
## [1] "Power (Hp): -6.59271"
```

```r
print((sprintf("Thrust (lbs): %g", thrust)))
```

```
## [1] "Thrust (lbs): -110.358"
```

```r
print((sprintf("RPM: %g", geom$RPM)))
```

```
## [1] "RPM: 158"
```

```r
print((sprintf("Solidity: %g", solidity)))
```

```
## [1] "Solidity: 0.0160697"
```

```r
minidf <- data.frame(df$station, df$c, df$beta)
colnames(minidf) <- c("station", "chord", "beta")

print(minidf)
```

```
##    station      chord      beta
## 1        1 0.16553886 64.792235
## 2        2 0.32070743 51.846256
## 3        3 0.40508783 42.101411
## 4        4 0.43004139 34.888754
## 5        5 0.42294978 29.492929
## 6        6 0.40192387 25.371770
## 7        7 0.37622429 22.152123
## 8        8 0.35009630 19.582385
## 9        9 0.32530616 17.491633
## 10      10 0.30245077 15.761661
## 11      11 0.28158404 14.308982
## 12      12 0.26250440 13.073363
## 13      13 0.24487699 12.010468
## 14      14 0.22827149 11.087047
## 15      15 0.21214964 10.277735
## 16      16 0.19581176  9.562883
## 17      17 0.17828988  8.927053
## 18      18 0.15813332  8.357963
## 19      19 0.13287410  7.845725
## 20      20 0.09700042  7.382296
## 21      21 0.00000000  6.961070
```

```r
print((sprintf("Total change in beta: %g deg", total_d_beta)))
```

```
## [1] "Total change in beta: 0 deg"
```

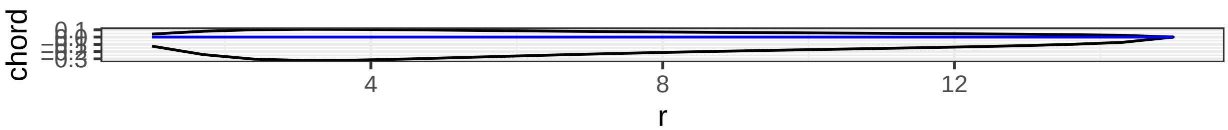
```r
################################################################################
# plotting code

prop_geom_le <- data.frame(chord = (df$c * (1/4)), r = df$r)
prop_geom_te <- data.frame(chord = (df$c * (-3/4)), r = df$r)
```

```
prop_geom_te <- prop_geom_te[seq(dim(prop_geom_te)[1],1),]

prop_geom <- rbind(prop_geom_le, prop_geom_te)

geom_plot <- ggplot(prop_geom, aes(x = r, y = chord)) + geom_path() + coord_fixed() + theme_bw() +
  geom_line(data = data.frame(chord = rep(0, length(df$r)), r = df$r), colour = "blue")
print(geom_plot)
```




```
###########################################################################

#  #record
#  pl_df <- rbind(pl_df, data.frame(Cp, Ct, eta = efficiency, J = advance_ratio, B = B))
#  pl_df <- rbind(pl_df, data.frame(T = geom$thrust, J = advance_ratio, velocity = V, degrees = total_d
#
#} # end advanced plotting loop

#coef <- 6
#pl <- ggplot(pl_df, aes(x = B)) +
#  geom_path(aes(y = Cp, colour = "Cp")) +
#  geom_point(aes(y = Cp, colour = "Cp")) +
#  geom_path(aes(y = Ct, colour = "Ct")) +
#  geom_point(aes(y = Ct, colour = "Ct")) +
#  geom_path(aes(y = eta/coef, colour = "eta")) +
#  geom_point(aes(y = eta/coef, colour = "eta")) +
#  scale_y_continuous(
```

```
#    # Features of the first axis
#    name = "Cp, Ct",

#    # Add a second axis and specify its features
#    sec.axis = sec_axis(~.*coef, name="Eta")) +
#  scale_colour_manual("",
#                          breaks = c("Cp", "Ct", "eta"),
#                          values = c("blue", "red", "green")) +
#  theme_bw()
#print(pl)

#pl <- ggplot(pl_df, aes(x = J, y = degrees)) +
#  geom_path() +
#  geom_point() +
#  #ggtitle("Feathering with Colocity") + # for the main title
#  xlab("J") + # for the x axis label
#  ylab("Feathing angle") + # for the y axis label
#  theme_bw()
#print(pl)
```