Ndim_coeff_articulated_rotor

The aim of the following script is to evaluate the angle of attack (α), the rotor inflow ratio (λ) and the non-dimensional coefficients of an articulated rotor:

- Tc Thrust coefficient
- Hc Rotor drag force coefficient
- Yc Lateral force coefficient
- Qc Torque coefficient
- Pc Power coefficient

In order to determine the non-dimensional coefficients, the following hypothesis are considered: constant rotor rotational speed and rotor velocity with respect to the air; small rotor disk plane angle of attack and flap angles; blade lag angle and hinge offset are equal to 0; the rotor inflow ratio is uniform. The last hypothesis leads to the following formula:

$$\lambda = \mu \tan\alpha + \frac{T_c}{2\sqrt{\mu^2 + \lambda^2}} \tag{1}$$

The calculation has been done with an iterative method based on the procedure shown in reference [**1**].

Syntax

The default syntax of the function is shown below; all inputs are scalar values.

```
[Tc,Hc,Yc,Qc,Pc,alfa,lambda] = Ndim_Coeff_Articulated_Rotor(V_inf,h,Lock,f,X)
```

Where:

INPUT

V_{inf}	Airspeed $[m/s]$
h	Altitude $[m]$
Lock	Lock number
f	Equivalent drag area of helicopter fuselage $[m^2]$
X	Angle of climb [°]

OUTPUT

Tc	Thrust coefficient
Hc	Rotor drag force coefficient
Yc	Lateral force coefficient
Qc	Torque coefficient
Pc	Power coefficient
alfa	Rotor angle of attack [°]
lambda	Rotor induced velocity

Within the function are also determined the parasite and induced coefficients, although the function doesn't return these parameters as output values.

Algorithm Description

The first iteration begins, assuming the angle of attack equal to 0, by evaluating the values of the rotor advance ratio and λ_i in order to determine the rotor inflow ratio with the following formula:

$$\lambda = \mu \tan \alpha + \lambda_i. \tag{2}$$

Since λ_i divergences for low values of μ, two different equations have been used [**2**]:

$$\begin{cases} \lambda_i = \dfrac{\sqrt{-\dfrac{V^2}{2} + \sqrt{\dfrac{V^4}{4} + \left(\dfrac{W}{2\rho A}\right)^2}}}{\Omega R}, & \textit{for } \mu \leq 0.1 \\ \lambda_i = \dfrac{T_c}{2\mu}, & \textit{for } \mu > 0.1 \end{cases} \tag{3}$$

The value of λ is used to compute the collective pitch angle (θ_0) and the flap angles (β_0, β_{1c}, β_{1s}), in order to compute the non-dimensional coefficients [**1**].

```
beta_0    = Lock*(theta_0/8*(1 + mu^2) + theta_tw/10*(1 + 5/6*mu^2)...
            -lambda/6);
beta_1c   = -2*mu*(4/3*theta_0+theta_tw-lambda)/(1-mu^2/2);
beta_1s   = -4/3*mu*beta_0/(1+mu^2/2);

Hci       = sigma*Cl_alpha*0.5*( theta_0*( -1/3*beta_1c + ...
            1/2*mu*lambda ) + theta_tw*(-1/4*beta_1c + 1/4*mu*lambda) ...
            +3/4*lambda*beta_1c + 1/6*beta_0*beta_1s + ...
            1/4*mu*(beta_0^2+ beta_1c^2));
Hc0       = sigma*Cd_mean*mu/4;
Hc        = Hci + Hc0;
Qc        = Pc;
Yc        = -sigma*Cl_alpha*0.5*(theta_0*(3/4*mu*beta_0 + ...
            1/3*beta_1s*(1 + 3/2*mu^2)) + theta_tw*(1/2*mu*beta_0 ...
            +1/4*beta_1s*(1 + mu^2)) -3/4*lambda*beta_1s + ...
            beta_0*beta_1c*(1/6 - mu^2)-3/2*mu*lambda*beta_0 - ...
            1/4*beta_1c*beta_1s);
```

Then, λ and α are updated with the following equations:

$$\begin{cases} \lambda = \lambda_i + \lambda_c + \mu \dfrac{H_c}{T_c} + \mu \dfrac{D_{fus}}{W} \\ \alpha = \arctan \left(\dfrac{\lambda - \dfrac{T_c}{2\sqrt{\mu^2 + \lambda^2}}}{\mu} \right) \end{cases} \tag{4}$$

At the end of each iteration, the error is evaluated as the absolute value of the difference between two consecutive values of λ, with a tolerance of 10^{-5}. The updated values of λ and α are used to compute, into the next iterations, the new value of λ_i as follows:

$$\lambda_i = \frac{T_c}{2\sqrt{\mu^2 + \lambda^2}}, for\ i \geq 1 \tag{5}$$

and the new value of μ.

Code listing

```
function [Tc,Hc,Yc,Qc,Pc,alfa,lambda] = ...
            Ndim_Coeff_Articulated_Rotor(V_inf,h,Lock,f,X)
%% Data
 [~,~,~,rho_inf] = atmosisa(h);
 g               = 9.8195;
 sigma           = N*c/(pi*R);
 W               = W*g;
 A               = pi*R^2;
 D_fus           = f*0.5*rho_inf*V_inf^2;

%% Initialization
 i        = 0;
 err      = 1;
 err_stop = 1e-5;
 alfa     = convang(0,'deg','rad');
 Tc       = W/(rho_inf*(Omega*R)^2*A);
 X        = convang(X,'deg','rad');
 lambda_c = V_inf*sin(X)/(Omega*R);

%% Beginning of the cycle
while abs(err) > err_stop
    mu = V_inf*cos(alfa)/(Omega*R);

if mu <= 0.1
    if i == 0
        lambda_i = sqrt(-V_inf^2/2+sqrt(V_inf^4/4+(W/(2*rho_inf*A))^2))/...
        (Omega*R);
    else
        lambda_i = Tc/(2*sqrt(mu^2 + lambda^2));
    end
end

if mu > 0.1
    if i == 0
        lambda_i = Tc/(2*mu);
    else
        lambda_i = Tc/(2*sqrt(mu^2 + lambda^2));
    end
end
%%
    lambda   = mu*tan(alfa) + lambda_i;
    theta_0  = (2*Tc/(sigma*Cl_alpha) - theta_tw/4*(1 + mu^2)...
               + lambda/2)*3/(1 + 3/2*mu^2);
    Pc0      = sigma*Cd_mean*(1 + 3*mu^2)/8;
    Pc       = lambda_i*Tc + lambda_c*Tc + mu*D_fus*Tc/W + Pc0;

    beta_0   = Lock*(theta_0/8*(1 + mu^2) + theta_tw/10*(1 + 5/6*mu^2)...
               -lambda/6);
    beta_1c  = -2*mu*(4/3*theta_0+theta_tw-lambda)/(1-mu^2/2);
    beta_1s  = -4/3*mu*beta_0/(1+mu^2/2);

    Hci      = sigma*Cl_alpha*0.5*( theta_0*( -1/3*beta_1c + ...
               1/2*mu*lambda ) + theta_tw*(-1/4*beta_1c + 1/4*mu*lambda) ...
               +3/4*lambda*beta_1c + 1/6*beta_0*beta_1s + ...
               1/4*mu*(beta_0^2+ beta_1c^2));
    Hc0      = sigma*Cd_mean*mu/4;
    Hc       = Hci + Hc0;
    Qc       = Pc;
    Yc       = -sigma*Cl_alpha*0.5*(theta_0*(3/4*mu*beta_0 + ...
               1/3*beta_1s*(1 + 3/2*mu^2)) + theta_tw*(1/2*mu*beta_0 ...
               +1/4*beta_1s*(1 + mu^2)) -3/4*lambda*beta_1s + ...
               beta_0*beta_1c*(1/6 - mu^2)-3/2*mu*lambda*beta_0 - ...
               1/4*beta_1c*beta_1s);

    lambda_old = lambda;
    lambda     = lambda_i + lambda_c+mu*Hc/Tc + mu*D_fus/W;
    alfa       = atan((lambda - Tc/(2*sqrt(mu^2 + lambda^2)))/mu);
    err        = abs(lambda - lambda_old);
    i          = i + 1;
end
```

```
alfa = convang(alfa,'rad','deg');

%% Determination of induced and parasite coefficients
% Qc0 = sigma*Cd_mean*(1 + mu^2)/8;
% Pc0 = sigma*Cd_mean*(1 + 3*mu^2)/8;
% Qci = Qc - Qc0;
% Pci = Pc - Pc0;
% Yci = Yc;
% Yc0 = 0;
end
```

Test Cases

In order to validate the function, some test cases have been run. The results obtained by the Ndim_Coeff_Articulated_Rotor function, for two different values of the thrust coefficient, have been compared with the data reference by Ki C. Kim [**3**]. The main physical and aerodynamic characteristics are given in the table 1.

Aircraft gross weight, W	7375 Kg
Number of blades, N	4
Radius, R	8.18 m
Blade chord, c	0.533 m
Solidity, σ	0.083
Lock number, γ	8
Blade airfoil	SC1095
Rotational speed, Ω	27 rad/sec
Nominal lift curve slope	$5.73/rad$
Equivalent drag area	4.20 m^2
Rate of climb	$0°$
Mean drag coefficient	0.0121
Linear twist rate, θ_{tw}	$0°$

Table 1: UH-60A Black Hawk Helicopter main characteristics.

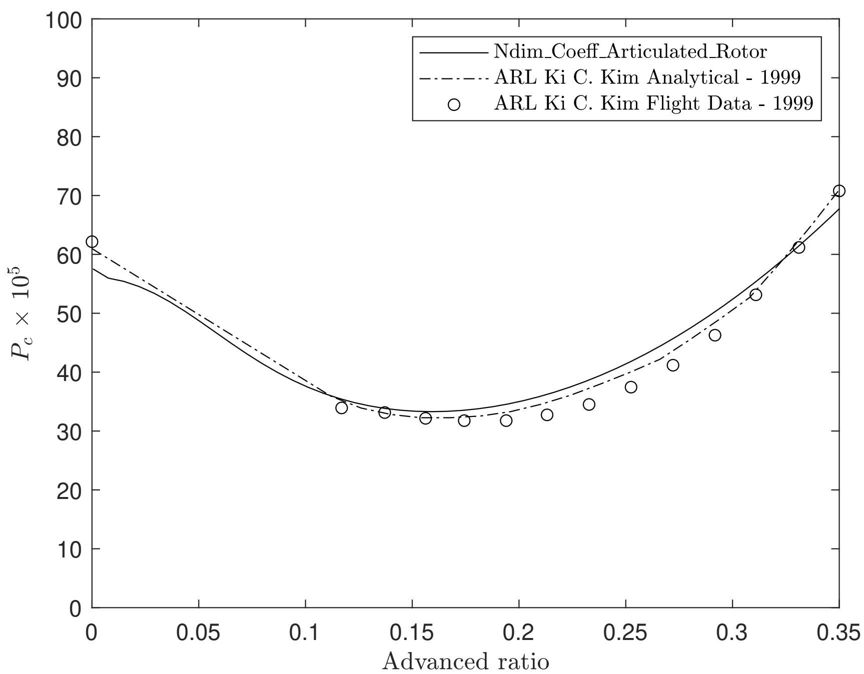

Figure 1: Power coefficient of UH-80A Helicopter, for $T_c = 0.0066$.

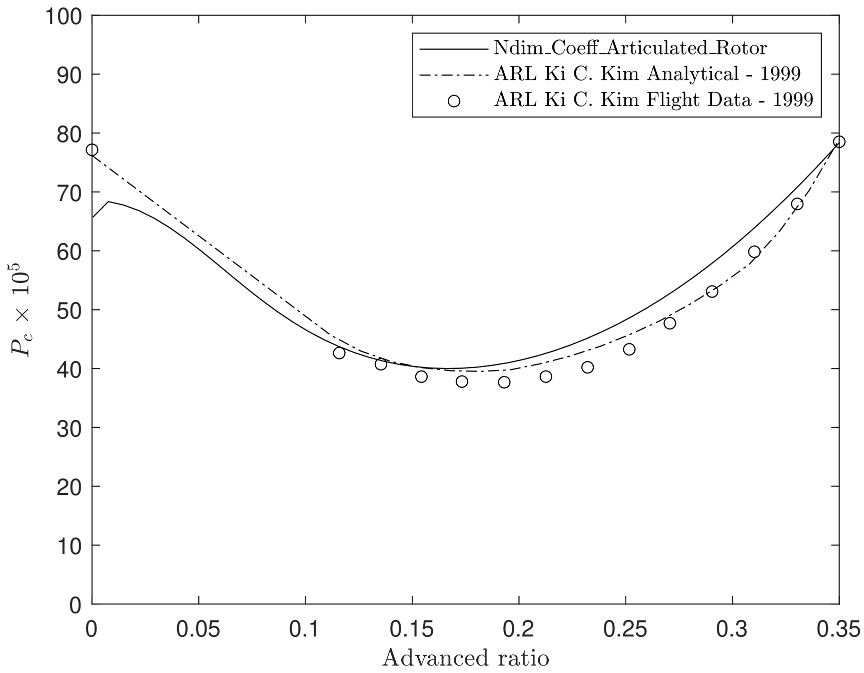

Figure 2: Power coefficient of UH-80A Helicopter, for $T_c = 0.0078$.

The comparisons have been performed considering an empirical factor, to cover non-uniform flow and tip loss in the evaluation of the induced power, equal to 1.15, according to the data reference by Ki C. Kim [**3**].

References

[1] Tognaccini R., Materiale didattico del corso di Aerodinamica dell'ala rotante - Lezioni di AERODINAMICA DELL'ALA ROTANTE, a.a. 2019-2020 - Dipartimento di Ingegneria Industriale, Università degli studi di Napoli Federico II.

[2] Lezioni del corso di Aerodinamica dell'ala rotante a.a. 2019-2020, a cura del Professore Tognaccini R. e dell'Ingegner Di Giorgio G. - Dipartimento di Ingegneria Industriale, Università degli studi di Napoli Federico II.

[3] Ki C. Kim, Analytical Calculations of Helicopter Trque Coefficient (C_Q) and Thrust Coefficient (C_T) Values ofr the Helicopter Perfomance (HELPE) Model. Army Research laboratory - 1999.