Momentum Theory: Forward Flight Inflow Ratio

Helicopter Theory
Project Report

Serkan Burak Örs*

*orss19@itu.edu.tr

Contents

1 Introduction

In this project, inflow ratio calculation with using momentum theory in forward flight performed for Aerospatiale AS 365N helicopter. For inflow ratio calculation, fixed point iteration method and Newton-Raphson method are used and their convergence rates are compared. The properties of helicopter can be seen in the table below.

Data	Unit	Value
Gross Takeoff Mass, $GTOW$	kg	4000
Main Rotor Diameter, d	m	11.93
Rotor Angular Velocity, Ω	rpm	300
Angle of Attack, α	degree	-2,0,2,4,6,8
Density, ρ	kg/m^3	1.225
Limit Forward Velocity, V_{max}	m/sec^2	90

2 Momentum Theory in Forward Flight

Under forward flight conditions, thrust generated by rotor tilted forward and from forward component of this thrust force, propulsive force can be obtained for forward flight. And simple momentum theory used for hover can be extended and used for forward flight condition. Forward flight case can be seen in Figure[1] from Leishman(2006).

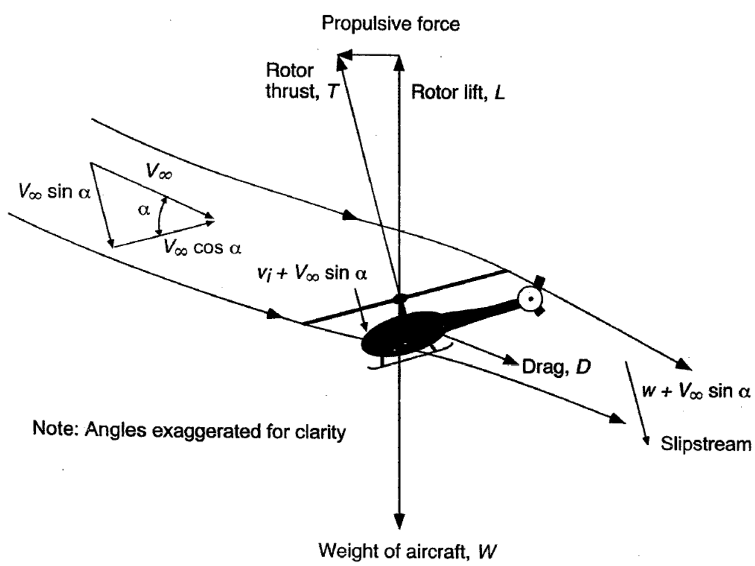

Figure 1: Glauert's flow model for the momentum theory in forward flight from Leishman(2006)

In forward flight, thrust force can be obtained as,

$$T = 2\dot{m}\nu_i = 2\left(\rho AU\right)\nu_i = 2\rho A\nu_i\sqrt{\left(V_\infty cos\alpha\right)^2 + \left(V_\infty sin\alpha + \nu_i\right)^2} \tag{2.1}$$

Recall equation[2.2]

$$\nu_h^2 = \frac{T}{2\rho A} \tag{2.2}$$

that for hovering flight, then the induced velocity in forward flight can be obtained as,

$$\nu_i = \frac{\nu_h^2}{\sqrt{\left(V_\infty cos\alpha\right)^2 + \left(V_\infty sin\alpha + \nu_i\right)^2}} \tag{2.3}$$

Advance ratio,μ is defined as,

$$\mu = \frac{V_\infty cos\alpha}{V_{tip}} = \frac{V_\infty cos\alpha}{\omega R} \tag{2.4}$$

and inflow ratio is defined as,

$$\lambda = \frac{V_\infty sin\alpha + \nu_i}{\omega R} \tag{2.5}$$

And using equation [2.4] and [2.5], leads the the expression,

$$\lambda = \frac{V_\infty sin\alpha}{\omega R} + \frac{\nu_i}{\omega R} = \mu tan\alpha + \lambda_i \tag{2.6}$$

Hence, equation[2.3] becomes

$$\lambda_i = \frac{\lambda_h^2}{\sqrt{\mu^2 + \lambda^2}} \tag{2.7}$$

And with using equation[2.8]

$$\lambda_h = \sqrt{\frac{C_T}{2}} \tag{2.8}$$

into equation[2.7], we obtained that

$$\lambda_i = \frac{C_T}{2\sqrt{\mu^2 + \lambda^2}} \tag{2.9}$$

Finally, the solution for inflow ratio in forward flight yields

$$\lambda = \mu tan\alpha + \frac{C_T}{2\sqrt{\mu^2 + \lambda^2}} \tag{2.10}$$

While the equation[2.10] can be solved analytically, in practice there are several numerical approachs can be used for solving inflow ratio in forward flight.

3 Numerical Solutions

Equation[2.10] can be solved numerically as stated before. There are two common approaches for this solution as:

- Fixed Point Iteration Method
- Newton-Raphson Method

3.1 Fixed Point Iteration Method

The solution procedure with using fixed point iteration method is simple. It requires to compute iteratively. Hence, equation[2.10] can be written as a iterative equation as,

$$\lambda_{k+1} = \mu tan\alpha + \frac{C_T}{2\sqrt{\mu^2 + \lambda_k^2}} \tag{3.1}$$

where k is the iteration number.
The starting value for $\lambda_{k=0}$ can be taken as the inflow ratio in hover condition which is,

$$\lambda_{k=0} = \lambda_h = \sqrt{\frac{C_T}{2}} \tag{3.2}$$

For estimating error, equation[3.3] can be used as,

$$\epsilon = \left| \frac{\lambda_{k+1} - \lambda_k}{\lambda_{k+1}} \right| \tag{3.3}$$

Generally, it is assumed the convergence occures if $\epsilon < 0.0005$. Normally fixed point iteration method converges the result with 10 or 15 iterations. However under some conditions, necessary iteration number may be higher.

3.2 Newton-Raphson Method

The solution procedure with using Newton-Raphson method requires differentiation. And the convergence rate of Newton-Raphson method is higher.
For Newton-Raphson method, the iteration method is

$$\lambda_{k+1} = \lambda_k - \left[\frac{f(\lambda)}{f'(\lambda)}\right]_k \tag{3.1}$$

where k is the iteration number.
Equation[2.10] can be rearranged for this method in the form of $f(\lambda) = 0$ as

$$f(\lambda) = \lambda - \mu tan\alpha - \frac{C_T}{2\sqrt{\mu^2 + \lambda^2}} \tag{3.2}$$

And differentiating this expression yields that,

$$f'(\lambda) = 1 + \frac{C_T}{2} \left(\mu^2 + \lambda_2\right)^{-\frac{3}{2}} \lambda \tag{3.3}$$

In most cases, the inflow ratio at hover condition can be used for initial value as $\lambda_{k=0} = \lambda_h$

4 Results and Discussion

4.1 Inflow Ratio Results

Results for the inflow ratio, $\frac{\lambda}{\lambda_h}$ are computed iteratively with both methods and plotted into figure [2] and [3] with changing angle of attack and advance ratio values.

Results of Fixed Point Iteration:

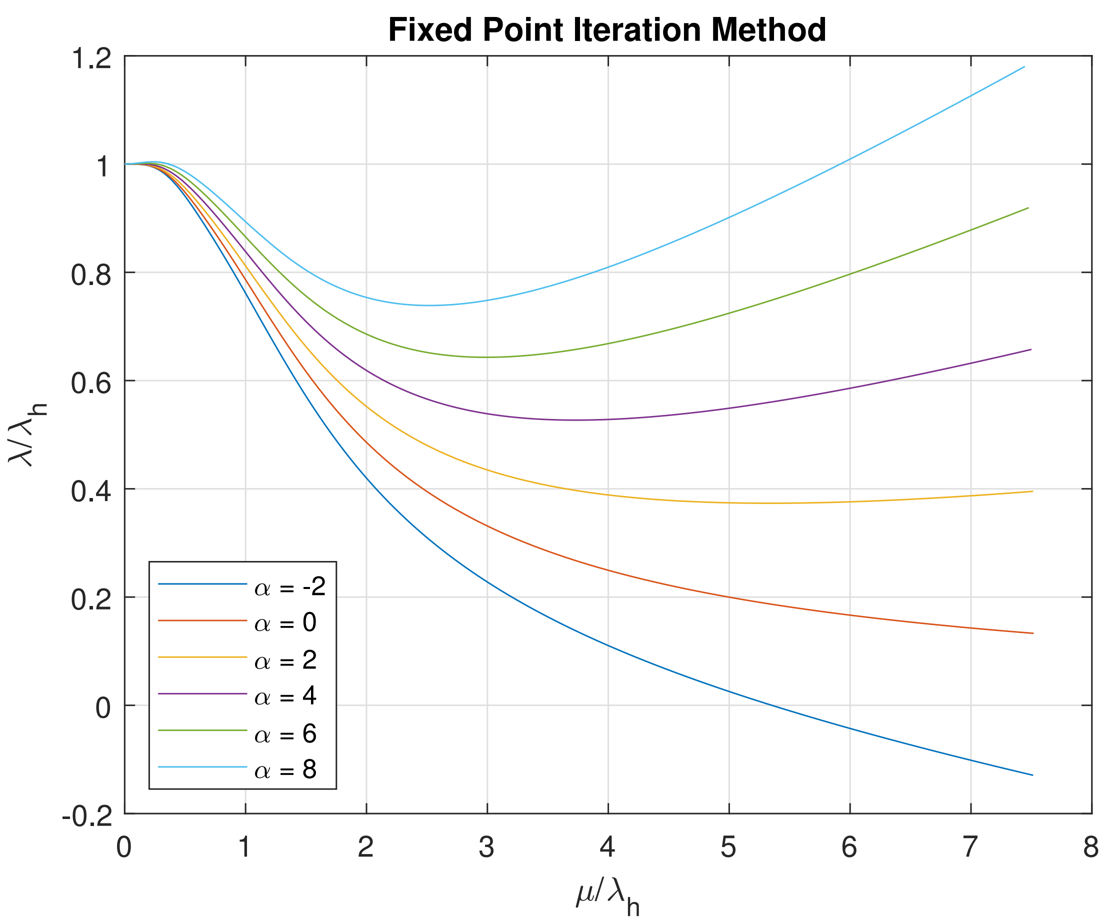

Figure 2: Inflow ratio as a function of forward speed ratio for several angle of attack calculated with Fixed Point Iteration

Results of Newton-Raphson Method:

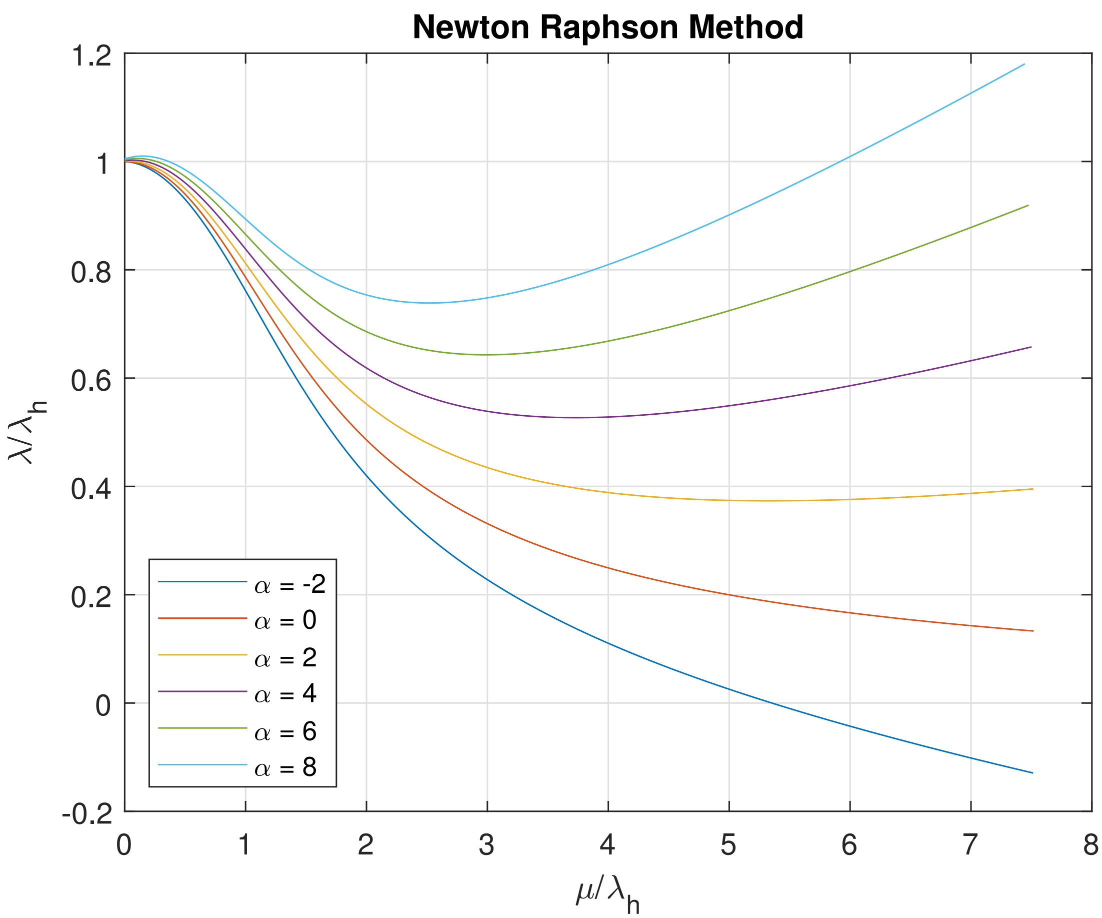

Figure 3: Inflow ratio as a function of forward speed ratio for several angle of attack calculated with Newton-Raphson Method

4.2 Discussion on Results

From figure [2] and [3], these can be deduced as:

- The inflow ratio increases with the increase of the actuator disk tilt angles.
- Notice that the induced part of the total inflow decreases with increasing advance ratio and the total inflow becomes dominated by the $\mu tan\alpha$ term at higher advance ratios.
- Notice also that negative tilt angles ultimately produce a negative inflow which is upflow through actuator disk, which means that the rotor is approaching a zero power required state which is autorotational conditions.

For observing convergence rate, $V_\infty = 90\ m/s^2$ and $\alpha = 8°$ case is chosen.
Generally Newton-Raphson method converges rapidly than Fixed Point Iteration. But for our application. They converged to values at same iteration number.

5 References

[1] Leishman G., *Principles of Helicopter Aerodynamics*, 2nd Ed., Cambridge University Press, 2006.

[2] Chapra, S. and Canale, R.P., *Numerical Methods for Engineers*, 7th Ed., 2015.

[3] Type-Certificate Data Sheet for DAUPHIN, No. EASA.R.105, 01 , European Aviation Safety Agency, 2014

[4] Type-Certificate Data Sheet for SA 365 / AS 365 / EC 155, No. EASA.R.105, 07, European Aviation Safety Agency, 2021

[5] Özdemir Ö., *Istanbul Technical University UCK459E - Helicopter Theory Lecture Notes*, 2022.

6 Appendix: MATLAB Codes

```
clc; clear; close all;
g=9.81; % Gravitional Acceleation [m/sec^2]
rho= 1.225; % Density of Air [kg/m^3]
GTOW=4000; % Aerospatiale AS 365N Gross Take Off Mass [kg];
d = 11.93; % Diameter of Main Rotor [m]
A=pi*(d/2)^2; % Actuator Disk Area [m^2];
T = GTOW*g; % Thrust [N]
V_inf = 0:0.01:90; % Forward Flight Speed [m/s^2], must be less than 90 ...
    m/sec^2
omega = 300*(1/60)*(2*pi); % Main Rotor Angular Speed [rad/sec]
Vtip = omega*(d/2); % Tip Speed of Main Rotor
Ct = T/(rho*A*Vtip^2); % Thrust Coefficient in Hover Condition
lambda_h = sqrt(Ct/2); % Initial Inflow Ratio
lambdafp(1) = sqrt(Ct/2);
alphaValues = -2:2:8; % Angle of Attack Values
error = 0.0005; % Permitted Error Percent
N = 1000; % Iteration Number
count1 = 1; % Counting Variable for Fixed Point Iteration Method
count2 = 1; % Counting Variable for Newton-Raphson Method


%% Fixed Point Iteration Method
for i=1:length(alphaValues)
alpha = alphaValues(i);
for j = 1:length(V_inf)
Vinf = V_inf(j);
Tinf = T/cosd(alpha);
Ct_inf = Tinf/(rho*A*Vtip^2);
y = sqrt(Ct_inf/2);
mufp = (Vinf*cosd(alpha))/Vtip;
x0 = lambda_h;
lambda = x0;
Xold = x0;
for k = 0:7
lambda = (mufp*tand(alpha))+(Ct_inf/(2*sqrt(mufp^2+lambda^2)));
ifp(k+1) = k;
if abs((lambda-Xold)/lambda) < error
break
end
```

```
Xold = lambda;
end
a = sqrt(Ct/2);
lambdaa(count1) = lambda/a;
muu(count1) = mufp/a;
count1 = count1+1;
end
figure(1)
plot(muu,lambdaa);
hold on;
grid on;
count1 = 1;
end
title('Fixed Point Iteration Method');
xlabel('\mu/\lambda_{h}');
ylabel('\lambda/\lambda_{h}');
legend('\alpha = -2','\alpha = 0','\alpha = 2','\alpha = 4','\alpha = ...
    6','\alpha = 8');

%% Newton-Raphson Method
for i=1:length(alphaValues)
alpha = alphaValues(i);
for j = 1:length(V_inf)
Vinf2 = V_inf(j);
Tinf2 = T/cosd(alpha);
Ct_inf2 = Tinf2/(rho*A*Vtip^2);
y12 = sqrt(Ct_inf2/2);
mu2 = (Vinf2*cosd(alpha))/Vtip;
f = @(lambda2) lambda2 - mu2*tand(alpha) - ...
    Ct_inf2/(2*sqrt(mu2^2+lambda2^2));
df=@(lambda2) 1+(Ct_inf2/2)*(mu2^2+lambda2^2)^(-3/2)*lambda2;
for k = 0:N
f0 = f(lambda_h);
f0_diff = df(lambda_h);
y12 = lambda_h - (f0/f0_diff);
ins(k+1)=k;
if abs((y12-lambda_h)/y12) < error
break
end
lambda_h = y12;
end
a2 = sqrt(Ct/2);
lambdaa2(count2) = y12/a2;
```

```
muu2(count2) = mu2/a2;
count2 = count2+1;
end
figure(2)
plot(muu2,lambdaa2);
hold on;
grid on;
count2 = 1;
end
title('Newton Raphson Method');
xlabel('\mu/\lambda_{h}');
ylabel('\lambda/\lambda_{h}');
legend('\alpha = -2','\alpha = 0','\alpha = 2','\alpha = 4','\alpha = ...
    6','\alpha = 8');

fprintf('Inflow Ratio is converged at %d iteration number with applying ...
    Fixed Point Iteration Method\n', ifp(end));
fprintf('Inflow Ratio is converged at %d iteration number with applying ...
    Newton-Raphson Method\n', ins(end));
```