46310 Projects in Wind Turbine Aeroelasticity

Assignment 1

Unsteady aerodynamic loads

Giulio Fauri - s172424 - Workload: 50%
Stefano Ribaudo - s171701 - Workload: 50%

Technical University of Denmark

March 2018

Contents

Introduction

The purpose of this report is to computing the unsteady aerodynamic loads on a horizontal axis wind turbine.

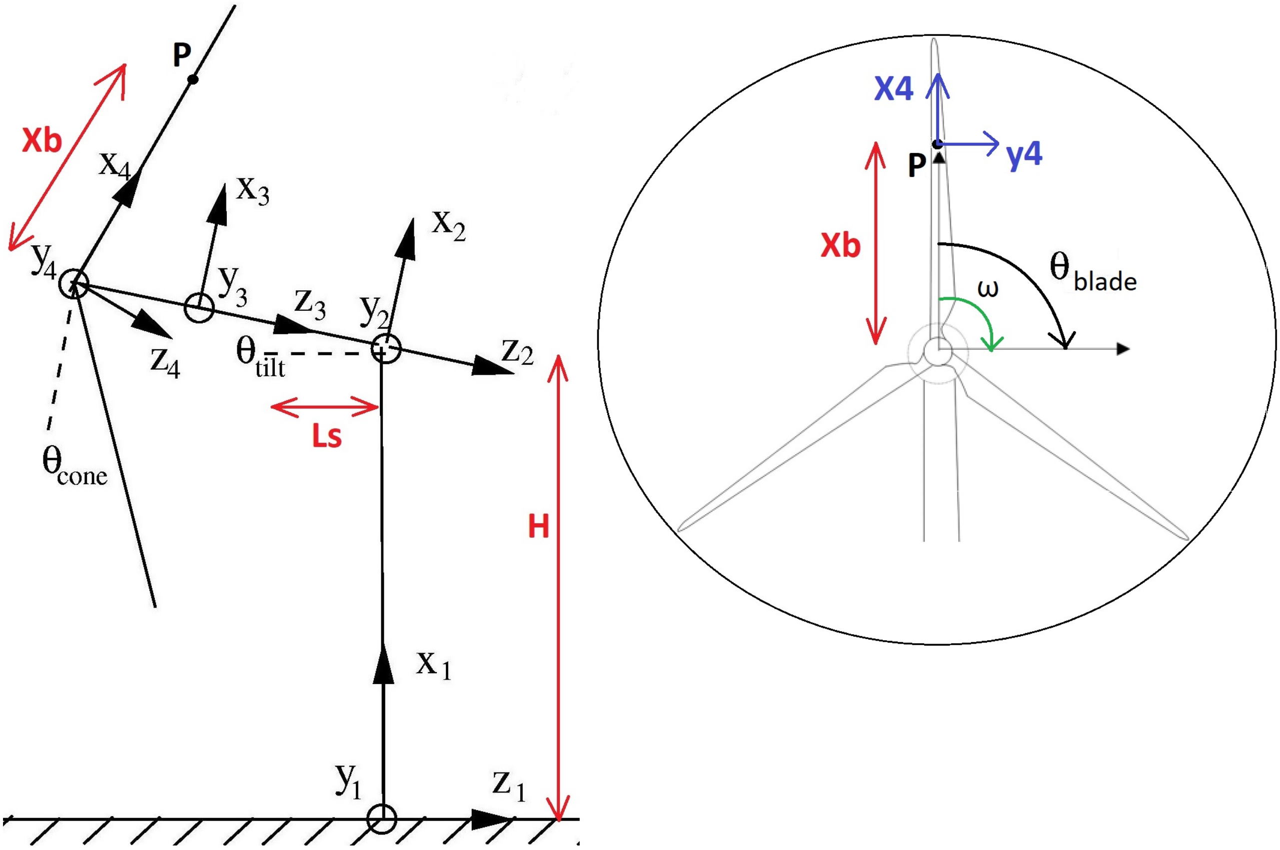


Figure 1: *Definition of the system of axes for HAWT wind turbine*

Before starting with the computation of the Unsteady Blade Element Momentum (BEM) code, we define the reference system 4 as the coordinate system constructed from the observation point of a rotating wind blade. Referring to figure 1, the construction of the coordinate system from the blade point of view can be divided into 3 main steps: tower, shaft and blade.

System 1 - Coordinate system fixed to the ground (not moving). The center is at the base of the tower; x-axis is pointing up following the tower's height; z is positive in the downwind direction; y complete right hand coordinate system.

1 → 2 (Tower) - Translation of H (tower height) along the x axis. Referring to figure 2, the rotation of the system following the angle of yaw on the y-z plane, the angle of tilt on the x-z plane (the roll rotation is not considered in this assignment).

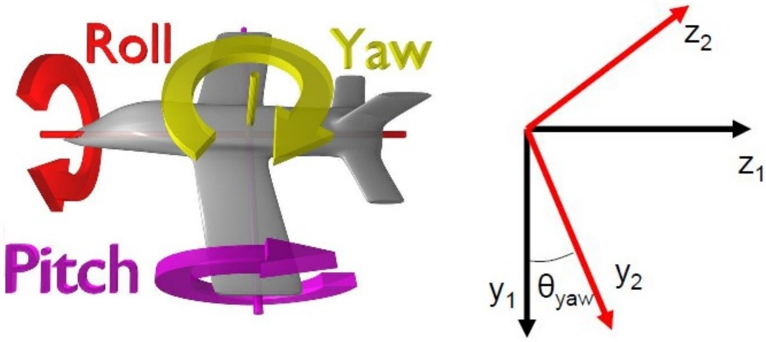


Figure 2: *Yaw angle: going from system 1 to 2 by rotating system 2 about x-axis*

System 2 - Coordinate system fixed to the top of the tower. The x-axis is rotate by the tilt angle with respect to the tower's vertical height; z-axis follows the nacelle axis, but with opposite direction; y complete right hand coordinate system as before.

2 → 3 (Shaft) - Translation of -Ls (nacelle shaft) along the z axis (the negative sign is due to the opposite direction with respect to the z axis); rotation of the system for each blade following the angle of wing θ_{blade} on the y-z plane (the angle of displacement between the three blades of the wind turbine is 120 degrees).

System 3 - Coordinate system fixed to the hub (moving following the rotation of each blade). The x-axis is pointing to the outside following the blade's length; y is perpendicular to x and it's following the blade's rotation as well; z is centered on the hub and it is positive in the downwind direction.

3 → 4 (Blade) - Translation of x_b (distance between the point P and the hub) along the x axis; rotation of the system following the angle of cone on the x-z plane.

System 4 - Coordinate system fixed to the point P on the blade (moving following the rotation of each blade). The coordinate system is similar to system 3 but with an inclination of θ_{cone} over the x-z field.

Q1) Unsteady BEM code

The reason why in this assignment is adopted a reference system anchored (fixed) to the blades of the wind turbine (and therefore "mobile") is because we want to analyze how the wind impacts on the rotor. In this way it will be easier to understand how the wind turbine should adapt its aerodynamics settings to the different conditions of the incoming wind.

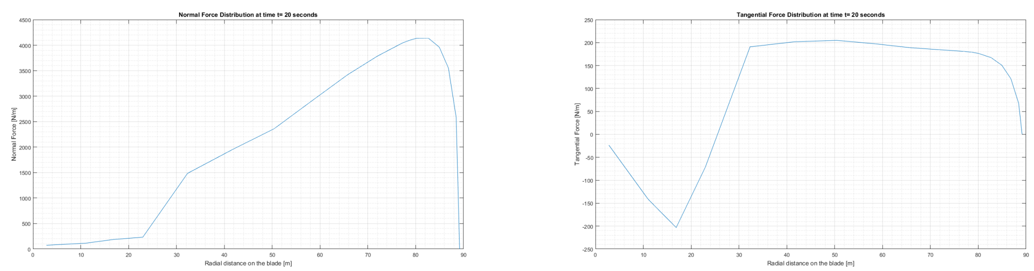
In this assignment we assume a constant rotational speed of $\omega = 0.673$ rad/s, a constant wind speed of $V_0 = 8$ m/s, $\theta_{pitch} = 0$ and the air density $\rho_{air} = 1.225$ kg/m3.

In this first question we calculated the load distributions, thrust and power of the wind turbine referring to the new coordinates system and we considered only steady results, so dynamic wake and dynamic stall models were not taking in account.

To create the code that allows you to calculate the aerodynamic loads (and consequently thrust and power) of the wind turbine, first of all we loaded the "airfoil" files containing all the datas necessary for computation. Then we calculated the exact impact of the wind from the point of view of the blade (coordinates system 4). Through several "for" cycles, this step has been carried out in order to obtain the results for the entire length of the blade for all 3 blades of our wind turbine.

Therefore, through interpolation for 2-D gridded data in the mesh grid format (command *interp*2 in MATLAB), we obtained the results of the coefficients for the lift and drag force (c_l,c_d) that allowed us to obtain the values of the aerodynamic loads on the blades (p_z, p_y), of thrust (T) and of the power produced (P).

As previously mentioned, these results were obtained for all points of the length of a blade for all wind turbine blades. These results are shown below.



(a) *Normal force (p_z)*   **(b)** *Tangential force (p_y)*

Figure 3: *Aerodynamic loads of blade 1 at time t=20s*

The above graphs show the aerodynamic loads (p_z, p_y) for blade 1 in vertical position ($\theta_b = 0\ ^{circ}$) depending on the distance from the rotor (blade radius). As you can see, the normal force (p_z) increases proportionally to the radius blade up to the maximum point ($p_{z,max} = 4.135[kN/m]$) just before the tip, point in which the wind "detaches" from the blade and loses its strength.

On the other hand, the tangential force (p_y) along the length of the blade, shows a different behavior. In fact, initially of negative value, it decreases up to the minimum point ($p_{y,min} = -0.203[kN/m]$) at the distance $r = 16.87[m]$, and then increases dramatically to a value average of about $p_{y,max} = 190[kN/m]$ which is constant until the end of the blade length (tip) in which the wake of the wind comes off and the tangential force returns to a null value.

The aerodynamic loads determine the thrust and power value of the wind turbine. Specifically: thrust (T) is defined as the underlying area (integral) of normal force (p_z), while power (P) is defined as the torque times the arm, where torque is defined as the underlying area (integral) of the tangential force (p_y).

Table 1: *Thrust and power in steady conditions*

Output	Value	at time [s]
Thrust	533.38 [kN]	20
Power	1.2226 [MW]	20

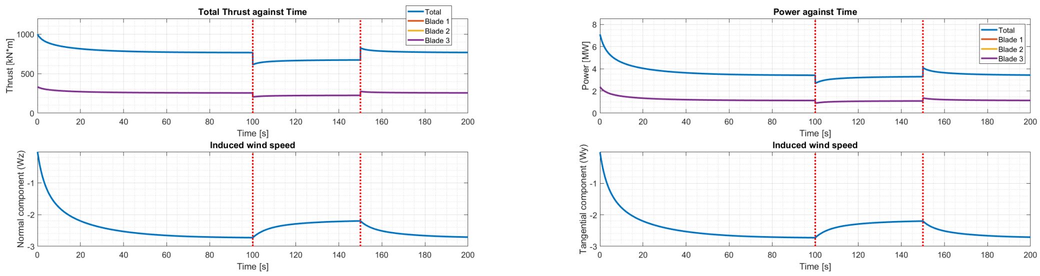
In particular we can assume that having a small load near the hub does not influence much the power, so it is possible that the twist angles used are optimized for different wind speeds. This assumption would explain the negative values of the tangential force in the vicinity of the rotor.

Q2) Pitch variation

In this second task we wanted to focus more on the pitch regulation. To do so, we modified the code in order to obtain the same outputs, but with some constrains (step changes) on the pitch angle:

$$\Theta_p(t) = 0^\circ \quad \leftrightarrow \quad t < 100s$$
$$\Theta_p(t) = 2^\circ \quad \leftrightarrow \quad 100s \leq t \leq 150s$$
$$\Theta_p(t) = 0^\circ \quad \leftrightarrow \quad t > 150s$$

How is possible to see, the pitch angle is almost always zero, but in the period between 100 and 150 seconds, it has a value of 2 degrees. What we are going to do now is analyze how this little change will modify the characteristics of the aerodynamic loads and thus how thrust and power will change.



(a) *Thrust as a function of time* **(b)** *Power as a function of time*

Figure 4: *Time history for the thrust, power and induced wind speed at r=65.75 m*

In addition to the variation of pitch, in this part of the report, we also consider the dynamic wake / inflow and the dynamic stall effects. This two models are easily observable from the plots.

When the operating conditions of the wind turbine are changed, for example a sudden variation in the pitch angle, the flow field can not adapt instantaneously to the new flow state due to the inertia of the air mass. This results in a time delay in the changes of the induced velocity and, as a consequence, in the not instantaneous variations of the thrust and power (Dynamic wake/inflow effect).

By varying the pitch angle, the angle of attack also varies. The effect of changing the AoA will not appear instantaneously on the loads, but will take place with a time delay. The delay in flow separation and the lag in the flow re-attachment process results in a hysteresis variation (dynamic stall effect). We chose to use the Stig Øye dynamic stall model, because it has trailing edge stall, it is very simple and it works fine for wind turbine. Moreover the beauty of this model is that the dynamic stall model is represented with only one function, f_s. In fact, for the trailing edge stall, the degree of stall is described via f_s which is separation function.

$$C_l(t) = f_s(t)C_{i,inv}(\alpha) + (1 - f_s(t))C_{l,f_s}(\alpha) \tag{1}$$

Q3) Atmospheric turbulence

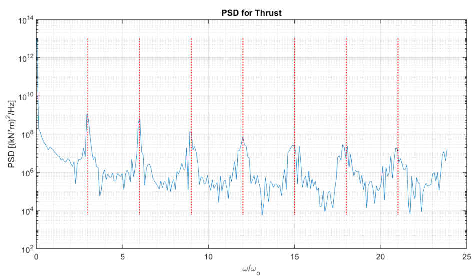
In this last question, we want to study the response of the wind turbine to a more realistic wind. In this regard we consider both the dynamic wake and the dynamic stall effects combined with the turbulence.

Turbulence is produced as air flowing across the Earth's surface encounters objects, such as trees or buildings. Turbulence also causes vibration and unequal forces on the wind turbine, especially the blades, that may weaken and damage the machine.

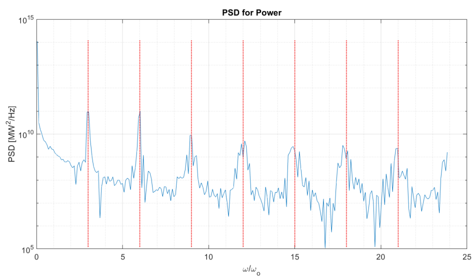
"Turbulence is to a wind machine like potholes to your car." - Robert Preus, Abundant Renewable Energy

To take the turbulence into account, we interpolated linearly the turbulence values obtained from the simulation "*IEC Turbulence Simulator*" to obtain the turbulence for each rotating blade element at any given time. Then, the turbulence is added to a mean wind speed of 8 m/s always considering the three components x, y, z defined according to the coordinate system 4 (explained in the introduction section). Therefore the wind speed used to obtain the aerodynamic loads (and therefore the thrust and power values), is a combination that includes the ideal wind behaviour (without wind shear and without tower effect), dynamic models (dynamic inflow dynamic stall) and the turbulence.

The power spectrum density (PSD) used in this task, allows us to analyze the spectrum of the frequencies of the aerodynamic loads and therefore of the thrust and of the power.



(a) *PSD of Thrust*



(b) *PSD of Power*

Figure 5: *Power spectrum density of total thrust and total power*

As can be seen from the above graphs, both signals show a strongly irregular decreasing trend, generated mainly by turbulence that adds different fluctuations to the aerodynamic loads seen by the blades. In particular, it is observed that in both cases there is a repetition of regular peaks every three frequency indices ($\frac{\omega}{\omega_0}$) (vertical red lines) due to the sum of the signal behaviour of each of the 3 blades.

Conclusion

The purpose of this report was to analyze the response of the wind turbine output signals (aerodynamic, thrust and power loads) to varying wind conditions. Precisely we tried to make changes to the wind characteristic considering models and effects closer and closer to the real wind behavior in general environmental conditions.
Furthermore, the whole study has been dealt with considering a "mobile" coordinate reference system, anchored to one of the wind turbine blades (and therefore rotating), rather than anchored to the ground.

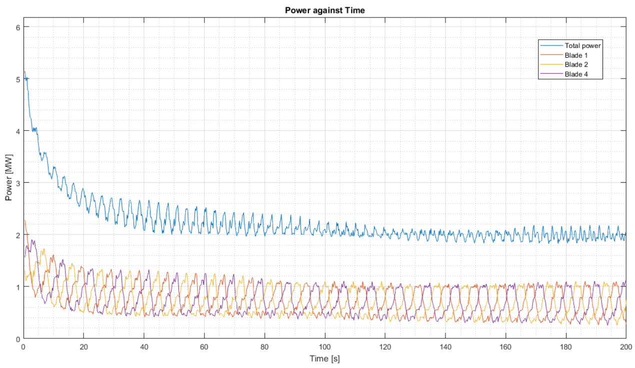


Figure 6: *Power with turbulent wind as function of time*

This last graph clearly shows how with turbulence, the power output from a wind turbine is rapidly influenced by these changes in wind speed. With the information gathered through this work we can say that although the induced wind reacts slowly to a change in external conditions (pitch or wind speed/direction, Fig. 4) the wind turbine is rapidly affected (Figs. 4 and 6). This behaviour can be clarified by noticing that the induced wind is not the only way for the turbine blade to feel a difference in wind speed/direction, but such changes are generated also by turbulence or pitch. Reminding that we also considered the dynamic stall effect, we can conclude that a significant change in lift and drag coefficients is noted as the external conditions changes.

Appendix

MATLAB CODE

46310 - Projects in wind turbine aeroelasticity

Assignment 1 - Unsteady aerodynamic loads on a horizontal axis wind turbine

Contents

Part 1 (Unsteady BEM code)

```
 clear all

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% VARIABLE PART
% DTU 10MW wind turbine datas

r_elem=[2.8,11,16.87,22.96,32.31,41.57,50.4,58.53,65.75,71.97,77.19,78.71,80.14,82
c=[5.38,5.45,5.87,6.18,6.02,5.42,4.7,4,3.4,2.91,2.54,2.43,2.33,2.13,1.9,1.63,1.18,
beta=-[-14.5,-14.43,-12.55,-8.89,-6.38,-4.67,-2.8866,-1.21,0.13,1.11,1.86,2.08,2.2
toverc=[100,100,86.05,61.1,43.04,32.42,27.81,25.32,24.26,24.1,24.1,24.1,24.1,24.1,

Prated=10000;       % Rated power [kW]
Cutin=4;            % Cut in speed [m/s]
cutout=25;          % Cut out speed [m/s]
T=20;               % Period suddivision [s]
H=119;              % Tower height [m]
Ls=7.1;             % Shaft length [m]
R=89.17;            % Blade length [m]
%r=70;              % Point distance on the blade (xb) [m]
rho=1.225;          % Air density [Kg/m]
theta_T=0; %-5;     % Tilt angle (upwards) [degree]
theta_C=0;          % Cone angle [degree]
theta_Y1=0;         % Yaw angle (first case) [degree]
theta_Y2=20;        % Yaw angle (second case) [degree]
```

```
w=0.673;%           % rotational speed (omega) [rad/s]
delta_t=0.02;       % time step [s]
B=3;                % Number of blades
v=0; %0.2;          % Wind shear [-]
% V0(x)=V0(H)*(x/H)^v;
V_H=6;%8;           % Mean wind speed at hub height [m/s] = V0(H)
% For the point P on the blade 1 at theta_B1=0 and x=xb=r=70m, so the V0(x) is:
% V0_x=V_H*(r/H)^v;

% User control input for tower effect
flag=0;
while flag==0
    flag_tower = input('Do you want to include the tower effect? (1=yes , 0=no)');
    if flag_tower==1 || flag_tower==0
        flag=1;
    else
        disp("Wrong input, try again")
    end
end

% User control input for induced wind
flag=0;
while flag==0
    flag_W = input('Do you want to include the induced wind? (1=yes , 0=no)');
    if flag_W==1 || flag_W==0
        flag=1;
    else
        disp("Wrong input, try again")
    end
end

% Transformation matrix
a1_1=[1 0 0; 0 cosd(theta_Y1) sind(theta_Y1); 0 -sind(theta_Y1) cosd(theta_Y1)];
a1_2=[1 0 0; 0 cosd(theta_Y2) sind(theta_Y2); 0 -sind(theta_Y2) cosd(theta_Y2)];
a2=[cosd(theta_T) 0 -sind(theta_T); 0 1 0; sind(theta_T) 0 cosd(theta_T)];
a3=[1 0 0; 0 1 0; 0 0 1];

a12_yaw1=a1_1*a2*a3;        % Yaw rotation (1 -> 2)
a12_yaw2=a1_2*a2*a3;
a21_1=a12_yaw1';
```

```
a21_2=a12_yaw2';

% theta_W1=w*t;                % Angle of wing 1 [degree]
% theta_W2=theta_W1+2*pi/3;    % Angle of wing 2 [degree]
% theta_W3=theta_W1+4*pi/3;    % Angle of wing 3 [degree]
% a23=[cosd(theta_W) sind(theta_W) 0; -sind(theta_W) cosd(theta_W) 0; 0 0 1]; % Wi

a34=[cosd(theta_C) 0 -sind(theta_C); 0 1 0; sind(theta_C) 0 cosd(theta_C)]; % Cone

% Coordinates of a point on a blade (P)
% r_t1=(H 0 0);
% r_s1=a21*(0 0 -Ls);
% r_b1=a41*(xb 0 0);
% r=(x y z)=r_t1+r_s1+r_b1;
% V0=(Vx Vy Vz)=a34*a23*a12*V1=a14*V1;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% COMPUTATIONAL PART

%Loading file data matrix
thick=[241,301,360,480,600,1000];
for i=1:length(thick)
    filename=strcat("FFA-W3-",num2str(thick(i)),"_ds.txt");
    if thick(i)==1000
        filename="cylinder_ds.txt";
    end
    file=load(filename);
    airfoil.thickness(i)=thick(i);
    airfoil.alpha=file(:,1)';
    airfoil.Clstat(i,:)=file(:,2);
    airfoil.Cdstat(i,:)=file(:,3);
    airfoil.Cmstat(i,:)=file(:,4);
    airfoil.fstat(i,:)=file(:,5);
    airfoil.Clinv(i,:)=file(:,6);
    airfoil.Clfs(i,:)=file(:,7);
end

%Finding the used thicknesses for which we'll interpolate
counter =0;
```

```
thicknesses(1)=0;
for j=1:length(toverc)
    flag=0;
    for k=1:length(thicknesses)
        if toverc(j)*10==thicknesses(k)
            flag=1;
        end
    end
    if flag==0
        counter=counter+1;
        thicknesses(counter)=toverc(j)*10;
    end
end

VCl=zeros(length(thicknesses),length(airfoil.alpha));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VCl(j,i)=interp1(thick,airfoil.Clstat(:,i),thicknesses(j),'linear'); %,line
    end
end
VCd=zeros(length(thicknesses),length(airfoil.alpha));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VCd(j,i)=interp1(thick,airfoil.Cdstat(:,i),thicknesses(j),'linear');
    end
end

clearvars file filename thick i j counter k flag

% x2=a12*x1=a3*a2*a1*x1;
in_len=length(1:ceil(T/delta_t))+1;

% Initialization
theta=zeros(in_len,B);% theta_B2=zeros(1,in_len)+rad2deg(2*pi/3); theta_B3=zeros(1
theta(1,2)=rad2deg(2*pi/3);
theta(1,3)=rad2deg(4*pi/3);
V0=zeros(in_len,3,B,length(r_elem));% V0_2=zeros(in_len,3); V0_3=zeros(in_len,3);
V4=zeros(in_len,3,B,length(r_elem));% V4_2=zeros(in_len,3); V4_3=zeros(in_len,3);
W=zeros(in_len,3,B,length(r_elem));% W_2=zeros(in_len,3); W_3=zeros(in_len,3);
Vrel=zeros(in_len,3,B,length(r_elem));% Vrel_2=zeros(in_len,3); Vrel_3=zeros(in_len
```

```
alpha=zeros(in_len,B,length(r_elem));
theta_p=zeros(in_len,B);
phi=zeros(in_len,B,length(r_elem));
a23=zeros(3,3,B);
a41=zeros(3,3,B);
a41_2=zeros(3,3,B);
r_b=zeros(3,B);
r_B=zeros(3,B);
absVrel=zeros(in_len,B,length(r_elem));
Cl=zeros(in_len,B,length(r_elem));
Cd=zeros(in_len,B,length(r_elem));
L=zeros(in_len,B,length(r_elem));
D=zeros(in_len,B,length(r_elem));
p=zeros(in_len,3,B,length(r_elem));
a=zeros(in_len,B,length(r_elem));
thrust_b=zeros(in_len,B);
Thrust=zeros(1,in_len);
torque_b=zeros(in_len,B);
Torque=zeros(1,in_len);
Power=zeros(1,in_len);
theta_tower=zeros(B,length(r_elem));
flagt=zeros(B,length(r_elem));

clearvars in_len

% Computational cycle for Thrust and Power
for n=1:(T/delta_t)
    for r=1:length(r_elem)
        for b=1:B
    time=n*delta_t;
    % wing angle (rotational position)
    theta(n+1,b)=theta(n,b)+rad2deg(w*delta_t);

    % update a23 for each blade
    a23(:,:,b)=[cosd(theta(n+1,b)) sind(theta(n+1,b)) 0; -sind(theta(n+1,b)) cosd(t

    % Yaw case 1
    a41(:,:,b)=(a34*a23(:,:,b)*a12_yaw1)';

    % Yaw case 2
```

```
    a41_2(:,:,b)=(a12_yaw2*a23(:,:,b)*a34)';

    % calculate position of P in system 1 --> r=r_t+r_s+r_b=(x,y,z)
    r_t1=[H 0 0]';
    r_s1=a21_1*[0 0 -Ls]';
    r_b(:,b)=a41(:,:,b)*[r_elem(r) 0 0]';
    r_B(:,b)=r_t1+r_s1+r_b(:,b);    % P(x,y,z) in Blade_b

    % calculate wind velocity in system 1 (wind shear, tower, turbulence)

    % WIND SHEAR
    V0(n+1,:,b,r)=[0,0,V_H*(r_B(1,b)/H)^v];

    % TOWER EFFECT
    if flag_tower==1
        a_bot=8.3;% size_tower_bottom [m]
        a_top=5.5;% size_tower_top [m]
        a_tower=((a_top-a_bot)/H)*r_B(1,b)+a_bot;
        r_tower=sqrt((r_B(3,b)^2)+(r_B(2,b)^2));
        if r_tower>=a_tower && r_B(1,b)<=H
%           if theta_tower(b,r)<0
%               flagt(b,r)=1;
%               theta_tower(b,r)
%           end

        theta_tower(b,r)=asind(-r_B(2,b)/r_tower);
%           if theta_tower(b,r)>0 && flagt(b,r)==1
%               theta_tower(b,r)
%               keyboard
%           end

        Vr=V0(n+1,3,b,r)*(1-(a_tower/r_tower)^2)*cosd(theta_tower(b,r));
        Vtheta=-V0(n+1,3,b,r)*(1+(a_tower/r_tower)^2)*sind(theta_tower(b,r));


            V0(n+1,3,b,r)=Vr*cosd(theta_tower(b,r))-Vtheta*sind(theta_tower(b,r));
            V0(n+1,2,b,r)=-Vr*sind(theta_tower(b,r))-Vtheta*cosd(theta_tower(b,r))
        end
    end
```

```
    % transform the wind velocity for blade system V0_4=a14*V0_1
    V4(n+1,:,b,r)=(trasposta(a41(:,:,b)))* V0(n+1,:,b,r)';     % V0 from the point

    Vrel(n+1,2,b,r)=V4(n+1,2,b,r)+W(n,2,b,r)-w*r_elem(r)*cosd(theta_C);
    Vrel(n+1,3,b,r)=V4(n+1,3,b,r)+W(n,3,b,r);
    phi(n+1,b,r)=atand(Vrel(n+1,3,b,r)/(-Vrel(n+1,2,b,r)));
    alpha(n+1,b,r)=phi(n+1,b,r)-(beta(r)+theta_p(n+1,b));

    %Finding Cl and Cd
%     Cl(n+1,b)=interp2(airfoil.alpha,airfoil.thickness,airfoil.Clstat,alpha(n+1,b
%     Cl(n+1,b)=interpolazione(airfoil.alpha,airfoil.thickness,airfoil.Clstat,alpha
%    Cd(n+1,b)=interp2(airfoil.alpha,airfoil.thickness,airfoil.Cdstat,alpha(n+1,b,r
%     Cd(n+1,b)=interpolazione(airfoil.alpha,airfoil.thickness,airfoil.Cdstat,alpha
    ind=find(thicknesses==toverc(r)*10);

    Cl(n+1,b,r)=interp1(airfoil.alpha,VCl(ind,:),alpha(n+1,b,r),'linear');
    Cd(n+1,b,r)=interp1(airfoil.alpha,VCd(ind,:),alpha(n+1,b,r),'linear');
    %Cl(n+1,b,r)=0.8;
    %Cd(n+1,b,r)=0.01;
    absVrel(n+1,b,r)=sqrt(Vrel(n+1,2,b,r)^2+Vrel(n+1,3,b,r)^2+Vrel(n+1,1,b,r)^2);
%     absVrel(n+1,b,r)=norm(Vrel(n+1,:,b,r));
    if r~=length(r_elem)
        L(n+1,b,r)=0.5*rho*absVrel(n+1,b,r)^2*c(r)*Cl(n+1,b,r);
        D(n+1,b,r)=0.5*rho*absVrel(n+1,b,r)^2*c(r)*Cd(n+1,b,r);
    else
        L(n+1,b,r)=0;
        D(n+1,b,r)=0;
    end
    p(n+1,3,b,r)=L(n+1,b,r)*cosd(phi(n+1,b,r))+D(n+1,b,r)*sind(phi(n+1,b,r));
    p(n+1,2,b,r)=L(n+1,b,r)*sind(phi(n+1,b,r))-D(n+1,b,r)*cosd(phi(n+1,b,r));


    %finding fg
    ac=0.2; %0.053
    %a(n+1,b,r)=(norm(V4(n+1,:,b,r))-norm(V4(n+1,:,b,r)+W(n,:,b,r)))/norm(V4(n+1,:
    a(n+1,b,r)=abs(W(n,3,b,r)/V4(n+1,3,b,r));
    if a(n+1,b,r)<=1/3
    %if a(n+1,b,r)<=ac
        fg=1;
    else
```

```
        %fg=ac/a(n+1,b,r)*(2-ac/a(n+1,b,r));
        fg=1/4*(5-3*a(n+1,b,r));
    end

    ABS=sqrt(V4(n+1,2,b,r)^2+(V4(n+1,3,b,r)+fg*W(n,3,b,r))^2);
    %Prandtl factor
    f=B/2*(R-r_elem(r))/(r_elem(r)*abs(sind(phi(n+1,b,r))));
    F=2/pi*acos(exp(-f));

    %calc W for next step (W(:,:,3)=Wz=Wn , W(:,:,2)=Wy=Wt)
    if r~=length(r_elem) && flag_W==1
        W(n+1,3,b,r)=-B*L(n+1,b,r)*cosd(phi(n+1,b,r))/(4*pi*rho*r_elem(r)*F*ABS);
        W(n+1,2,b,r)=-B*L(n+1,b,r)*sind(phi(n+1,b,r))/(4*pi*rho*r_elem(r)*F*ABS);
    else
        W(n+1,3,b,r)=0;
        W(n+1,2,b,r)=0;
    end

    if r==length(r_elem)
        thrust_b(n+1,b)=trapz(r_elem,squeeze(p(n+1,3,b,:))');
        Thrust(n+1)=Thrust(n+1)+thrust_b(n+1,b);
        torque_b(n+1,b)=trapz(r_elem,r_elem.*squeeze(p(n+1,2,b,:))');
        Torque(n+1)=Torque(n+1)+torque_b(n+1,b);
%         Torque2(n+1)=torque_b(n+1,b)*3;

    end
        end
    end
    Power(n+1)=Torque(n+1)*w;

end

clearvars ABS ac b r n f F fg ind r_tower theta_tower a_top a_bot flag a_tower


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% RESULTS PART
% Position in space in system 1 of the point P(r) on the blade --> (x1,y1,z1)
% Velocity components Vz and Vy on blade 1 with Yaw1 and Yaw2 (2 cases) --> Vz_1, V
```

```
% Thrust and Power results table
ResultsTable=table({'Thrust','Power'}',...
    [Thrust(end),Power(end)]',...
    [time,time]',...
    'VariableNames',{'Output','Value','Time'});
disp(ResultsTable);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PLOTS PART

% Plot for normal force distribution (p_z)
figure
plot(r_elem,squeeze(p(end,3,1,:)))
title(strcat("Normal Force Distribution at time t= ",num2str(T)," seconds"))
ylabel("Normal Force [N/m]")
xlabel("Radial distance on the blade [m]")
grid on
grid minor

% Plot for tangential force distribution (p_y)
figure
plot(r_elem,squeeze(p(end,2,1,:)))
title(strcat("Tangential Force Distribution at time t= ",num2str(T)," seconds"))
ylabel("Tangential Force [N/m]")
xlabel("Radial distance on the blade [m]")
grid on
grid minor

% Plot for thrust against time
figure
plot(delta_t*2:delta_t:T,Thrust(3:end)/1000)
if flag_tower==1
    title("Total Thrust against Time with tower effect")
else
    title("Total Thrust against Time")
end
ylabel("Thrust [kN*m]")
xlabel("Time [s]")
ylim([0,max(Thrust(3:end))*1.2/1000])
xlim([0,T])
```

```
grid on
grid minor
hold on
plot(delta_t*2:delta_t:T,thrust_b(3:end,1)/1000)
plot(delta_t*2:delta_t:T,thrust_b(3:end,2)/1000)
plot(delta_t*2:delta_t:T,thrust_b(3:end,3)/1000)
legend('Total','Blade 1','Blade 2','Blade 3')

% Plot for power against time
figure
plot(delta_t*2:delta_t:T,Power(3:end)/10^6)
if flag_tower==1
    title("Power against Time with tower effect")
else
    title("Power against Time")
end

ylabel("Power [MW]")
xlabel("Time [s]")
ylim([0,max(Power(3:end))*1.2/10^6])
xlim([0,T])
grid on
grid minor
hold on
plot(delta_t*2:delta_t:T,w*torque_b(3:end,1)/1000000)
plot(delta_t*2:delta_t:T,w*torque_b(3:end,2)/1000000)
plot(delta_t*2:delta_t:T,w*torque_b(3:end,3)/1000000)
legend('Total','Blade 1','Blade 2','Blade 3')
```

Part 2 (Unsteady BEM code with pitch variation between 100s and 150s)

```
clear all

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% VARIABLE PART
% DTU 10MW wind turbine datas

r_elem=[2.8,11,16.87,22.96,32.31,41.57,50.41,58.53,65.75,71.97,77.19,78.71,80.14,82
c=[5.38,5.45,5.87,6.18,6.02,5.42,4.7,4,3.4,2.91,2.54,2.43,2.33,2.13,1.9,1.63,1.18,0
```

```
beta=-[-14.5,-14.43,-12.55,-8.89,-6.38,-4.67,-2.89,-1.21,0.13,1.11,1.86,2.08,2.28,2
toverc=[100,100,86.05,61.1,43.04,32.42,27.81,25.32,24.26,24.1,24.1,24.1,24.1,24.1,2

Prated=10000;       % Rated power [kW]
Cutin=4;            % Cut in speed [m/s]
cutout=25;          % Cut out speed [m/s]
T=200;              % Period suddivision [s]
H=119;              % Tower height [m]
Ls=7.1;             % Shaft length [m]
R=89.17;            % Blade length [m]
%r=70;              % Point distance on the blade (xb) [m]
rho=1.225;          % Air density [Kg/m]
theta_T= input('Choose tilt angle');   % Tilt angle (upwards) [degree]
theta_C=0;          % Cone angle [degree]
theta_Y1=0;         % Yaw angle (first case) [degree]
theta_Y2=20;        % Yaw angle (second case) [degree]
w=0.673;            % rotational speed (omega) [rad/s]
delta_t=0.02;       % time step [s]
B=3;                % Number of blades
v=0;                % Wind shear [-]

% User control input for tower effect
flag=0;
while flag==0
    flag_tower = input('Do you want to include the tower effect? (1=yes , 0=no)');
    if flag_tower==1 || flag_tower==0
        flag=1;
    else
        disp("Wrong input, try again")
    end
end

% User control input for dynamic stall
flag=0;
while flag==0
    flag_dyn_stall = input('Do you want to include the dynamic stall effect? (1=yes
    if flag_dyn_stall==1 || flag_dyn_stall==0
        flag=1;
    else
        disp("Wrong input, try again")
```

```
    end
end

% V0(x)=V0(H)*(x/H)^v;
V_H=8;          % Mean wind speed at hub height [m/s] = V0(H)
% For the point P on the blade 1 at theta_B1=0 and x=xb=r=70m, so the V0(x) is:
% V0_x=V_H*(r/H)^v;

% Transformation matrix
a1_1=[1 0 0; 0 cosd(theta_Y1) sind(theta_Y1); 0 -sind(theta_Y1) cosd(theta_Y1)];
a1_2=[1 0 0; 0 cosd(theta_Y2) sind(theta_Y2); 0 -sind(theta_Y2) cosd(theta_Y2)];
a2=[cosd(theta_T) 0 -sind(theta_T); 0 1 0; sind(theta_T) 0 cosd(theta_T)];
a3=[1 0 0; 0 1 0; 0 0 1];

a12_yaw1=a1_1*a2*a3;        % Yaw rotation (1 -> 2)
a12_yaw2=a1_2*a2*a3;
a21_1=a12_yaw1';
a21_2=a12_yaw2';

% theta_W1=w*t;               % Angle of wing 1 [degree]
% theta_W2=theta_W1+2*pi/3;   % Angle of wing 2 [degree]
% theta_W3=theta_W1+4*pi/3;   % Angle of wing 3 [degree]
% a23=[cosd(theta_W) sind(theta_W) 0; -sind(theta_W) cosd(theta_W) 0; 0 0 1]; % Wi

a34=[cosd(theta_C) 0 -sind(theta_C); 0 1 0; sind(theta_C) 0 cosd(theta_C)]; % Cone

% Coordinates of a point on a blade (P)
% r_t1=(H 0 0);
% r_s1=a21*(0 0 -Ls);
% r_b1=a41*(xb 0 0);
% r=(x y z)=r_t1+r_s1+r_b1;
% V0=(Vx Vy Vz)=a34*a23*a12*V1=a14*V1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% COMPUTATIONAL PART

%Loading file data matrix
thick=[241,301,360,480,600,1000];
for i=1:length(thick)
    filename=strcat("FFA-W3-",num2str(thick(i)),"_ds.txt");
```

```
    if thick(i)==1000
        filename="cylinder_ds.txt";
    end
    file=load(filename);
    airfoil.thickness(i)=thick(i);
    airfoil.alpha=file(:,1)';
    airfoil.Clstat(i,:)=file(:,2);
    airfoil.Cdstat(i,:)=file(:,3);
    airfoil.Cmstat(i,:)=file(:,4);
    airfoil.fstat(i,:)=file(:,5);
    airfoil.Clinv(i,:)=file(:,6);
    airfoil.Clfs(i,:)=file(:,7);
end

%Finding the used thicknesses for which we'll interpolate
counter =0;
thicknesses(1)=0;
for j=1:length(toverc)
    flag=0;
    for k=1:length(thicknesses)
        if toverc(j)*10==thicknesses(k)
            flag=1;
        end
    end
    if flag==0
        counter=counter+1;
        thicknesses(counter)=toverc(j)*10;
    end
end

VClstat=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VClstat(j,i)=interp1(thick,airfoil.Clstat(:,i),thicknesses(j),'linear','ext
    end
end
VCd=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VCd(j,i)=interp1(thick,airfoil.Cdstat(:,i),thicknesses(j),'linear','extrap
```

```
    end
end
VClinv=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VClinv(j,i)=interp1(thick,airfoil.Clinv(:,i),thicknesses(j),'linear','extra
    end
end
Vfstat=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        Vfstat(j,i)=interp1(thick,airfoil.fstat(:,i),thicknesses(j),'linear','extra
    end
end
VClfs=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VClfs(j,i)=interp1(thick,airfoil.Clfs(:,i),thicknesses(j),'linear','extrap
    end
end

clearvars file filename thick i j counter k flag

%Creating a vector for computing torque
% x2=a12*x1=a3*a2*a1*x1;
in_len=length(1:ceil(T/delta_t))+1;

% Initialization
theta=zeros(in_len,B);                  % theta_B2=zeros(1,in_len)+rad2deg(2*pi/3)
theta(1,2)=rad2deg(2*pi/3);
theta(1,3)=rad2deg(4*pi/3);
V0=zeros(in_len,3,B,length(r_elem));    % V0_2=zeros(in_len,3); V0_3=zeros(in_len,
V4=zeros(in_len,3,B,length(r_elem));    % V4_2=zeros(in_len,3); V4_3=zeros(in_len,
W=zeros(in_len,3,B,length(r_elem));     % W_2=zeros(in_len,3); W_3=zeros(in_len,3)
Wqs=zeros(in_len,3,B,length(r_elem));
Wint=zeros(in_len,3,B,length(r_elem));
Vrel=zeros(in_len,3,B,length(r_elem));  % Vrel_2=zeros(in_len,3); Vrel_3=zeros(in_
alpha=zeros(in_len,B,length(r_elem));
theta_p=zeros(in_len,B);
phi=zeros(in_len,B,length(r_elem));
```

```
a23=zeros(3,3,B);
a41=zeros(3,3,B);
a41_2=zeros(3,3,B);
r_b=zeros(3,B);
r_B=zeros(3,B);
absVrel=zeros(in_len,B,length(r_elem));
Clstat=zeros(in_len,B);
Cl=zeros(in_len,B);
fs=zeros(in_len,B,length(r_elem));
Clinv=zeros(in_len,B);
Clfs=zeros(in_len,B);
fstat=zeros(in_len,B);
Cd=zeros(in_len,B);
L=zeros(in_len,B);
D=zeros(in_len,B);
p=zeros(in_len,3,B,length(r_elem));
a=zeros(in_len,B,length(r_elem));
thrust_b=zeros(in_len,B);
Thrust=zeros(1,in_len);
torque_b=zeros(in_len,B);
Torque=zeros(1,in_len);
Power=zeros(1,in_len);

clearvars in_len


for n=1:(T/delta_t)
    for r=1:length(r_elem)
        for b=1:B
    time=n*delta_t;
    if time>=100 && time<=150
        theta_p(n+1,:)=[2,2,2];
    end
    % wing angle (rotational position)
    theta(n+1,b)=theta(n,b)+rad2deg(w*delta_t);

    % update a23 for each blade
    a23(:,:,b)=[cosd(theta(n+1,b)) sind(theta(n+1,b)) 0; -sind(theta(n+1,b)) cosd(t

    % Yaw case 1
```

```
a41(:,:,b)=(a34*a23(:,:,b)*a12_yaw1)';

% Yaw case 2
a41_2(:,:,b)=(a12_yaw2*a23(:,:,b)*a34)';

% calculate position of P in system 1 --> r=r_t+r_s+r_b=(x,y,z)
r_t1=[H 0 0]';
r_s1=a21_1*[0 0 -Ls]';
r_b(:,b)=a41(:,:,b)*[r_elem(r) 0 0]';
r_B(:,b)=r_t1+r_s1+r_b(:,b);    % P(x,y,z) in Blade_b

% calculate wind velocity in system 1 (wind shear, tower, turbulence)
% WIND SHEAR
V0(n+1,:,b,r)=[0,0,V_H*(r_B(1,b)/H)^v];
% TOWER EFFECT
if flag_tower==1
    r_tower=sqrt((r_B(3,b)^2)+(r_B(2,b)^2));
    theta_tower=asind(-r_B(2,b)/r_tower);
    a_bot=8.3;% size_tower_bottom [m]
    a_top=5.5;% size_tower_top [m]
    a_tower=((a_top-a_bot)/H)*r_B(1,b)+a_bot;

    Vr=V0(n+1,3,b,r)*(1-(a_tower/r_tower)^2)*cosd(theta_tower);
    Vtheta=-V0(n+1,3,b,r)*(1+(a_tower/r_tower)^2)*sind(theta_tower);

    if r_tower>=a_tower && r_B(1,b)<=H
        V0(n+1,3,b,r)=Vr*cosd(theta_tower)-Vtheta*sind(theta_tower);
        V0(n+1,2,b,r)=-Vr*sind(theta_tower)-Vtheta*cosd(theta_tower);
    end
end

% transform the wind velocity for blade system V0_4=a14*V0_1
V4(n+1,:,b,r)=(trasposta(a41(:,:,b)))* V0(n+1,:,b,r)';     % V0 from the point

Vrel(n+1,2,b,r)=V4(n+1,2,b,r)+W(n,2,b,r)-w*r_elem(r)*cosd(theta_C);
Vrel(n+1,3,b,r)=V4(n+1,3,b,r)+W(n,3,b,r);

phi(n+1,b,r)=atand(Vrel(n+1,3,b,r)/(-Vrel(n+1,2,b,r)));
alpha(n+1,b,r)=phi(n+1,b,r)-(beta(r)+theta_p(n+1,b));
```

```
    %Finding Cl and Cd
%     Cl(n+1,b)=interp2(airfoil.alpha,airfoil.thickness,airfoil.Clstat,alpha(n+1,b
%     Cl(n+1,b)=interpolazione(airfoil.alpha,airfoil.thickness,airfoil.Clstat,alpha
%    Cd(n+1,b)=interp2(airfoil.alpha,airfoil.thickness,airfoil.Cdstat,alpha(n+1,b,r
%     Cd(n+1,b)=interpolazione(airfoil.alpha,airfoil.thickness,airfoil.Cdstat,alpha
    ind=find(thicknesses==toverc(r)*10);

    Clstat(n+1,b)=interp1(airfoil.alpha,VClstat(ind,:),alpha(n+1,b,r));
    Clinv(n+1,b)=interp1(airfoil.alpha,VClinv(ind,:),alpha(n+1,b,r));
    Clfs(n+1,b)=interp1(airfoil.alpha,VClfs(ind,:),alpha(n+1,b,r));
    fstat(n+1,b)=interp1(airfoil.alpha,Vfstat(ind,:),alpha(n+1,b,r));
    Cd(n+1,b)=interp1(airfoil.alpha,VCd(ind,:),alpha(n+1,b,r));
    absVrel(n+1,b,r)=sqrt(Vrel(n+1,2,b,r)^2+Vrel(n+1,3,b,r)^2+Vrel(n+1,1,b,r)^2);
    if n==1
        fs(n,b,r)=fstat(n+1,b);
    end
    %Dynamic stall model
    tau=4*c(r)/norm(V4(n+1,:,b,r));
    fs(n+1,b,r)=fstat(n+1,b)+(fs(n,b,r)-fstat(n+1,b))*exp(-delta_t/tau);
    Cl(n+1,b)=fs(n+1,b,r)*Clinv(n+1,b)+(1-fs(n+1,b,r))*Clfs(n+1,b);


    if flag_dyn_stall==0
        Cl(n+1,b)=Clstat(n+1,b);
    end
    if r~=length(r_elem)
        L(n+1,b)=0.5*rho*absVrel(n+1,b,r)^2*c(r)*Cl(n+1,b);
        D(n+1,b)=0.5*rho*absVrel(n+1,b,r)^2*c(r)*Cd(n+1,b);
    else
        L(n+1,b)=0;
        D(n+1,b)=0;
    end
    p(n+1,3,b,r)=L(n+1,b)*cosd(phi(n+1,b,r))+D(n+1,b)*sind(phi(n+1,b,r));
    p(n+1,2,b,r)=L(n+1,b)*sind(phi(n+1,b,r))-D(n+1,b)*cosd(phi(n+1,b,r));

    %finding fg
    ac=0.2;
    a(n+1,b,r)=(norm(V4(n+1,:,b,r))-norm(V4(n+1,:,b,r)+W(n,:,b,r)))/norm(V4(n+1,:,b

    %a(n+1,b,r)=abs(W(n,3,b,r)/V4(n+1,3,b,r));
```

```
%if a(n+1,b,r)<=ac
if a(n+1,b,r)<=1/3
    fg=1;
else
    %fg=ac/a(n+1,b,r)*(2-ac/a(n+1,b,r));
    fg=1/4*(5-3*a(n+1,b,r));
end

%ABS=sqrt(V4(n+1,2,b,r)^2+(V4(n+1,3,b,r)+fg*W(n,3,b,r))^2);
ABS=norm(V4(n+1,:,b,r)+[0,0,fg*W(n,3,b,r)]);

%Prandtl factor
f=B/2*(r_elem(end)-r_elem(r))/(r_elem(r)*abs(sind(phi(n+1,b,r))));
F=2/pi*acos(exp(-f));

%calc W for next step (W(:,:,3)=Wz=Wn , W(:,:,2)=Wy=Wt)
if r~=length(r_elem)
    Wqs(n+1,3,b,r)=-B*L(n+1,b)*cosd(phi(n+1,b,r))/(4*pi*rho*r_elem(r)*F*ABS);
    Wqs(n+1,2,b,r)=-B*L(n+1,b)*sind(phi(n+1,b,r))/(4*pi*rho*r_elem(r)*F*ABS);
else
    Wqs(n+1,3,b,r)=0;
    Wqs(n+1,2,b,r)=0;
end

%Dynamic wake model
if a(n+1,b,r)<=0.5
    tau1=1.1/(1-1.3*a(n+1,b,r))*R/sqrt(V4(n+1,2,b,r)^2+V4(n+1,3,b,r)^2);
    tau2=tau1*(0.39-0.26*(r_elem(r)/R)^2);
    H_din=Wqs(n+1,:,b,r)+0.6*tau1*(Wqs(n+1,:,b,r)-Wqs(n,:,b,r))/delta_t;
    Wint(n+1,:,b,r)=H_din+(Wint(n,:,b,r)-H_din)*exp(-delta_t/tau1);
    W(n+1,:,b,r)=Wint(n+1,:,b,r)+(W(n,:,b,r)-Wint(n+1,:,b,r))*exp(-delta_t/tau2
else
    W(n+1,:,b,r)=Wqs(n+1,:,b,r);
end
if r==17 && b==1 && W(n+1,3,b,r)==0
    keyboard
end

if r==length(r_elem)
    thrust_b(n+1,b)=trapz(r_elem,squeeze(p(n+1,3,b,:))');
```

```
        Thrust(n+1)=Thrust(n+1)+thrust_b(n+1,b);
        torque_b(n+1,b)=trapz(r_elem,r_elem.*squeeze(p(n+1,2,b,:))');
        Torque(n+1)=Torque(n+1)+torque_b(n+1,b);
    end
        end
    end
    Power(n+1)=Torque(n+1)*w;
end

clearvars ABS ac b r n f F fg ind r_tower theta_tower a_top a_bot flag a_tower

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% RESULTS PART
% Position in space in system 1 of the point P(r) on the blade --> (x1,y1,z1)
% Velocity components Vz and Vy on blade 1 with Yaw1 and Yaw2 (2 cases) --> Vz_1, V

% Thrust and Power results table
ResultsTable=table({'Thrust','Power'}',...
    [Thrust(end),Power(end)]',...
    [time,time]',...
    'VariableNames',{'Output','Value','Time'});
disp(ResultsTable);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PLOTS PART
%Plot for normal force distribution
figure
plot(r_elem,squeeze(p(end,3,1,:)))
title(strcat("Normal Force Distribution at time t= ",num2str(T)," seconds"))
ylabel("Normal Force [N/m]")
xlabel("Radial distance on the blade [m]")
grid on
grid minor

%Plot for tangential force distribution
figure
plot(r_elem,squeeze(p(end,2,1,:)))
title(strcat("Tangential Force Distribution at time t= ",num2str(T)," seconds"))
ylabel("Tangential Force [N/m]")
xlabel("Radial distance on the blade [m]")
```

```
grid on
grid minor

%Plot for thrust against time
figure
subplot(2,1,1)
plot(delta_t*2:delta_t:T,Thrust(3:end)/1000,'LineWidth',3)
if flag_tower==1
    title("Total Thrust against Time with tower effect")
else
    title("Total Thrust against Time")
end
ylabel("Thrust [kN*m]")
xlabel("Time [s]")
ylim([0,max(Thrust(3:end))*1.2/1000])
xlim([0,T])
grid on
grid minor
hold on
plot(delta_t*2:delta_t:T,thrust_b(3:end,1)/1000,'LineWidth',3)
plot(delta_t*2:delta_t:T,thrust_b(3:end,2)/1000,'LineWidth',3)
plot(delta_t*2:delta_t:T,thrust_b(3:end,3)/1000,'LineWidth',3)
plot([100,100],[0,max(Thrust(3:end))*1.2/1000],':r','LineWidth',3)
plot([150,150],[0,max(Thrust(3:end))*1.2/1000],':r','LineWidth',3)
set(gca,'fontsize',15)
legend('Total','Blade 1','Blade 2','Blade 3')
subplot(2,1,2)
plot(delta_t*2:delta_t:T,W(3:end,3,1,9),'LineWidth',3)
hold on
plot([100,100],[min(W(3:end,3,1,9))*1.1,max(W(3:end,3,1,9))*1.1],':r','LineWidth',3
plot([150,150],[min(W(3:end,3,1,9))*1.1,max(W(3:end,3,1,9))*1.1],':r','LineWidth',3
ylabel("Normal component (Wz)")
xlabel("Time [s]")
xlim([0,T])
ylim([min(W(3:end,3,1,9))*1.1,max(W(3:end,3,1,9))*1.1])
grid on
grid minor
title('Induced wind speed')
set(gca,'fontsize',15)
```

```
%Plot for power against time
figure
subplot(2,1,1)
plot(delta_t*2:delta_t:T,Power(3:end)/10^6,'LineWidth',3)
if flag_tower==1
    title("Power against Time with tower effect")
else
    title("Power against Time")
end
ylabel("Power [MW]")
xlabel("Time [s]")
ylim([0,max(Power(3:end))*1.2/10^6])
xlim([0,T])
grid on
grid minor
hold on
plot(delta_t*2:delta_t:T,w*torque_b(3:end,1)/1000000,'LineWidth',3)
plot(delta_t*2:delta_t:T,w*torque_b(3:end,2)/1000000,'LineWidth',3)
plot(delta_t*2:delta_t:T,w*torque_b(3:end,3)/1000000,'LineWidth',3)
plot([100,100],[0,max(Power(3:end))*1.2/10^6],':r','LineWidth',3)
plot([150,150],[0,max(Power(3:end))*1.2/10^6],':r','LineWidth',3)
set(gca,'fontsize',15)
legend('Total','Blade 1','Blade 2','Blade 3')
subplot(2,1,2)
plot(delta_t*2:delta_t:T,W(3:end,3,1,9),'LineWidth',3)
hold on
plot([100,100],[min(W(3:end,3,1,9))*1.1,max(W(3:end,3,1,9))*1.1],':r','LineWidth',
plot([150,150],[min(W(3:end,3,1,9))*1.1,max(W(3:end,3,1,9))*1.1],':r','LineWidth',
ylabel("Tangential component (Wy)")
xlabel("Time [s]")
xlim([0,T])
ylim([min(W(3:end,3,1,9))*1.1,max(W(3:end,2,1,9))*1.1])
grid on
grid minor
title('Induced wind speed')
set(gca,'fontsize',15)
```

Part 3 (PSD of thrust and power with turbulence)

```
clear all
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% VARIABLE PART
% DTU 10MW wind turbine datas

r_elem=[2.8,11,16.87,22.96,32.31,41.57,50.41,58.53,65.75,71.97,77.19,78.71,80.14,8
c=[5.38,5.45,5.87,6.18,6.02,5.42,4.7,4,3.4,2.91,2.54,2.43,2.33,2.13,1.9,1.63,1.18,0
beta=-[-14.5,-14.43,-12.55,-8.89,-6.38,-4.67,-2.89,-1.21,0.13,1.11,1.86,2.08,2.28,2
toverc=[100,100,86.05,61.1,43.04,32.42,27.81,25.32,24.26,24.1,24.1,24.1,24.1,24.1,2

Prated=10000;        % Rated power [kW]
Cutin=4;             % Cut in speed [m/s]
cutout=25;           % Cut out speed [m/s]
T=200;               % Period suddivision [s]
H=119;               % Tower height [m]
Ls=7.1;              % Shaft length [m]
R=89.17;             % Blade length [m]
%r=70;               % Point distance on the blade (xb) [m]
rho=1.225;           % Air density [Kg/m]
theta_T= input('Choose tilt angle');   % Tilt angle (upwards) [degree]
theta_C=0;           % Cone angle [degree]
theta_Y1=0;          % Yaw angle (first case) [degree]
theta_Y2=20;         % Yaw angle (second case) [degree]
w=0.673;             % rotational speed (omega) [rad/s]
delta_t=T/1024;      % time step [s]
B=3;                 % Number of blades
v=0;                 % Wind shear [-]

% User control input for tower effect
flag=0;
while flag==0
    flag_tower = input('Do you want to include the tower effect? (1=yes , 0=no)');
    if flag_tower==1 || flag_tower==0
        flag=1;
    else
        disp("Wrong input, try again")
    end
end

% User control input for induced wind
flag=0;
```

```
while flag==0
    flag_W = input('Do you want to include the induced wind? (1=yes , 0=no)');
    if flag_W==1 || flag_W==0
        flag=1;
    else
        disp("Wrong input, try again")
    end
end

% User control input for dynamic stall
flag=0;
while flag==0
    flag_dyn_stall = input('Do you want to include the dynamic stall effect? (1=yes
    if flag_dyn_stall==1 || flag_dyn_stall==0
        flag=1;
    else
        disp("Wrong input, try again")
    end
end

% V0(x)=V0(H)*(x/H)^v;
V_H=8;          % Mean wind speed at hub height [m/s] = V0(H)
% For the point P on the blade 1 at theta_B1=0 and x=xb=r=70m, so the V0(x) is:
% V0_x=V_H*(r/H)^v;

% Transformation matrix
a1_1=[1 0 0; 0 cosd(theta_Y1) sind(theta_Y1); 0 -sind(theta_Y1) cosd(theta_Y1)];
a1_2=[1 0 0; 0 cosd(theta_Y2) sind(theta_Y2); 0 -sind(theta_Y2) cosd(theta_Y2)];
a2=[cosd(theta_T) 0 -sind(theta_T); 0 1 0; sind(theta_T) 0 cosd(theta_T)];
a3=[1 0 0; 0 1 0; 0 0 1];

a12_yaw1=a1_1*a2*a3;        % Yaw rotation (1 -> 2)
a12_yaw2=a1_2*a2*a3;
a21_1=a12_yaw1';
a21_2=a12_yaw2';

% theta_W1=w*t;                % Angle of wing 1 [degree]
% theta_W2=theta_W1+2*pi/3;    % Angle of wing 2 [degree]
% theta_W3=theta_W1+4*pi/3;    % Angle of wing 3 [degree]
% a23=[cosd(theta_W) sind(theta_W) 0; -sind(theta_W) cosd(theta_W) 0; 0 0 1]; % Wi
```

```
a34=[cosd(theta_C) 0 -sind(theta_C); 0 1 0; sind(theta_C) 0 cosd(theta_C)]; % Cone

% Coordinates of a point on a blade (P)
% r_t1=(H 0 0);
% r_s1=a21*(0 0 -Ls);
% r_b1=a41*(xb 0 0);
% r=(x y z)=r_t1+r_s1+r_b1;
% V0=(Vx Vy Vz)=a34*a23*a12*V1=a14*V1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% COMPUTATIONAL PART

%Loading file data matrix
thick=[241,301,360,480,600,1000];
for i=1:length(thick)
    filename=strcat("FFA-W3-",num2str(thick(i)),"_ds.txt");
    if thick(i)==1000
        filename="cylinder_ds.txt";
    end
    file=load(filename);
    airfoil.thickness(i)=thick(i);
    airfoil.alpha=file(:,1)';
    airfoil.Clstat(i,:)=file(:,2);
    airfoil.Cdstat(i,:)=file(:,3);
    airfoil.Cmstat(i,:)=file(:,4);
    airfoil.fstat(i,:)=file(:,5);
    airfoil.Clinv(i,:)=file(:,6);
    airfoil.Clfs(i,:)=file(:,7);
end

%Finding the used thicknesses for which we'll interpolate
counter =0;
thicknesses(1)=0;
for j=1:length(toverc)
    flag=0;
    for k=1:length(thicknesses)
        if toverc(j)*10==thicknesses(k)
            flag=1;
        end
```

```
    end
    if flag==0
        counter=counter+1;
        thicknesses(counter)=toverc(j)*10;
    end
end

VClstat=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VClstat(j,i)=interp1(thick,airfoil.Clstat(:,i),thicknesses(j),'linear','ext
    end
end
VCd=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VCd(j,i)=interp1(thick,airfoil.Cdstat(:,i),thicknesses(j),'linear','extrap
    end
end
VClinv=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VClinv(j,i)=interp1(thick,airfoil.Clinv(:,i),thicknesses(j),'linear','extra
    end
end
Vfstat=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        Vfstat(j,i)=interp1(thick,airfoil.fstat(:,i),thicknesses(j),'linear','extra
    end
end
VClfs=zeros(length(thicknesses),length(thick));
for j=1:length(thicknesses)
    for i=1:length(airfoil.alpha)
        VClfs(j,i)=interp1(thick,airfoil.Clfs(:,i),thicknesses(j),'linear','extrap
    end
end

clearvars file filename thick i j counter k flag
```

```
%Creating a vector for computing torque
% x2=a12*x1=a3*a2*a1*x1;
in_len=length(1:ceil(T/delta_t))+1;

% Initialization
theta=zeros(in_len,B);% theta_B2=zeros(1,in_len)+rad2deg(2*pi/3); theta_B3=zeros(1
theta(1,2)=rad2deg(2*pi/3);
theta(1,3)=rad2deg(4*pi/3);
V0=zeros(in_len,3,B,length(r_elem));% V0_2=zeros(in_len,3); V0_3=zeros(in_len,3);
V4=zeros(in_len,3,B,length(r_elem));% V4_2=zeros(in_len,3); V4_3=zeros(in_len,3);
W=zeros(in_len,3,B,length(r_elem));% W_2=zeros(in_len,3); W_3=zeros(in_len,3);
Wqs=zeros(in_len,3,B,length(r_elem));
Wint=zeros(in_len,3,B,length(r_elem));
Vrel=zeros(in_len,3,B,length(r_elem));% Vrel_2=zeros(in_len,3); Vrel_3=zeros(in_le
alpha=zeros(in_len,B,length(r_elem));
theta_p=zeros(in_len,B);
phi=zeros(in_len,B,length(r_elem));
a23=zeros(3,3,B);
a41=zeros(3,3,B);
a41_2=zeros(3,3,B);
r_b=zeros(3,B);
r_B=zeros(3,B);
absVrel=zeros(in_len,B,length(r_elem));
Clstat=zeros(in_len,B);
Cl=zeros(in_len,B);
fs=zeros(in_len,B,length(r_elem));
Clinv=zeros(in_len,B);
Clfs=zeros(in_len,B);
fstat=zeros(in_len,B);
Cd=zeros(in_len,B);
L=zeros(in_len,B);
D=zeros(in_len,B);
p=zeros(in_len,3,B,length(r_elem));
a=zeros(in_len,B,length(r_elem));
thrust_b=zeros(in_len,B);
Thrust=zeros(1,in_len);
torque_b=zeros(in_len,B);
Torque=zeros(1,in_len);
Power=zeros(1,in_len);
```

```
clearvars in_len

%Loading turbolence
n1=1024;
n2=64;
n3=64;
filenames={'turb_12u.bin','turb_12v.bin','turb_12w.bin'};
turb.u=zeros(n1,n2,n3);
turb.v=zeros(n1,n2,n3);
turb.w=zeros(n1,n2,n3);
fields = fieldnames(turb);
for l=1:numel(fields)
    fid=fopen(filenames{l});
    uraw=fread(fid,'single');
    itael=0;
    for i=1:n1
        for j=1:n2
            for k=1:n3
                itael=itael+1;
                turb.(fields{l})(i,j,k)=uraw(itael);
            end
        end
    end
end

clearvars n1 n2 n3 fid uraw itael filenames fields i j k l


% Computational cycle for Thrust and Power
for n=1:(T/delta_t)
    for r=1:length(r_elem)
        for b=1:B
    time=n*delta_t;

    % wing angle (rotational position)
    theta(n+1,b)=theta(n,b)+rad2deg(w*delta_t);

    % update a23 for each blade
    a23(:,:,b)=[cosd(theta(n+1,b)) sind(theta(n+1,b)) 0; -sind(theta(n+1,b)) cosd(t
```

```
% Yaw case 1
a41(:,:,b)=(a34*a23(:,:,b)*a12_yaw1)';

% Yaw case 2
a41_2(:,:,b)=(a12_yaw2*a23(:,:,b)*a34)';

% calculate position of P in system 1 --> r=r_t+r_s+r_b=(x,y,z)
r_t1=[H 0 0]';
r_s1=a21_1*[0 0 -Ls]';
r_b(:,b)=a41(:,:,b)*[r_elem(r) 0 0]';
r_B(:,b)=r_t1+r_s1+r_b(:,b);    % P(x,y,z) in Blade_b


% calculate wind velocity in system 1 (wind shear, tower, turbulence)

% WIND SHEAR
V0(n+1,:,b,r)=[0,0,V_H*(r_B(1,b)/H)^v];

% TOWER EFFECT
if flag_tower==1
    r_tower=sqrt((r_B(3,b)^2)+(r_B(2,b)^2));
    theta_tower=asind(-r_B(2,b)/r_tower);
    a_bot=8.3;% size_tower_bottom [m]
    a_top=5.5;% size_tower_top [m]
    a_tower=((a_top-a_bot)/H)*r_B(1,b)+a_bot;

    Vr=V0(n+1,3,b,r)*(1-(a_tower/r_tower)^2)*cosd(theta_tower);
    Vtheta=-V0(n+1,3,b,r)*(1+(a_tower/r_tower)^2)*sind(theta_tower);

    if r_tower>=a_tower && r_B(1,b)<=H
        V0(n+1,3,b,r)=Vr*cosd(theta_tower)-Vtheta*sind(theta_tower);
        V0(n+1,2,b,r)=-Vr*sind(theta_tower)-Vtheta*cosd(theta_tower);
    end
end

%Turbulence
xturb=linspace(-90,90,64);
yturb=linspace(119-90,119+90,64);
```

```
    Vturb(1)=interp2(xturb,yturb,squeeze(turb.w(n,:,:)),r_B(2,b),r_B(1,b)); %Vx tur
    Vturb(2)=interp2(xturb,yturb,squeeze(turb.v(n,:,:)),r_B(2,b),r_B(1,b)); %Vy tur
    Vturb(3)=interp2(xturb,yturb,squeeze(turb.u(n,:,:)),r_B(2,b),r_B(1,b)); %Vz tur
    V0(n+1,:,b,r)=V0(n+1,:,b,r)+Vturb;

    % transform the wind velocity for blade system V0_4=a14*V0_1
    V4(n+1,:,b,r)=(trasposta(a41(:,:,b)))* V0(n+1,:,b,r)';     % V0 from the point

    Vrel(n+1,2,b,r)=V4(n+1,2,b,r)+W(n,2,b,r)-w*r_elem(r)*cosd(theta_C);
    Vrel(n+1,3,b,r)=V4(n+1,3,b,r)+W(n,3,b,r);

    phi(n+1,b,r)=atand(Vrel(n+1,3,b,r)/(-Vrel(n+1,2,b,r)));
    alpha(n+1,b,r)=phi(n+1,b,r)-(beta(r)+theta_p(n+1,b));

    %Finding Cl and Cd
%     Cl(n+1,b)=interp2(airfoil.alpha,airfoil.thickness,airfoil.Clstat,alpha(n+1,b
%     Cl(n+1,b)=interpolazione(airfoil.alpha,airfoil.thickness,airfoil.Clstat,alpha
%    Cd(n+1,b)=interp2(airfoil.alpha,airfoil.thickness,airfoil.Cdstat,alpha(n+1,b,r
%     Cd(n+1,b)=interpolazione(airfoil.alpha,airfoil.thickness,airfoil.Cdstat,alpha
    ind=find(thicknesses==toverc(r)*10);

    Clstat(n+1,b)=interp1(airfoil.alpha,VClstat(ind,:),alpha(n+1,b,r));
    Clinv(n+1,b)=interp1(airfoil.alpha,VClinv(ind,:),alpha(n+1,b,r));
    Clfs(n+1,b)=interp1(airfoil.alpha,VClfs(ind,:),alpha(n+1,b,r));
    fstat(n+1,b)=interp1(airfoil.alpha,Vfstat(ind,:),alpha(n+1,b,r));
    Cd(n+1,b)=interp1(airfoil.alpha,VCd(ind,:),alpha(n+1,b,r));
    absVrel(n+1,b,r)=sqrt(Vrel(n+1,2,b,r)^2+Vrel(n+1,3,b,r)^2+Vrel(n+1,1,b,r)^2);
    if n==1
        fs(n,b,r)=fstat(n+1,b);
    end


    %Dynamic stall model
    tau=4*c(r)/norm(V4(n+1,:,b,r));
    fs(n+1,b,r)=fstat(n+1,b)+(fs(n,b,r)-fstat(n+1,b))*exp(-delta_t/tau);
    Cl(n+1,b)=fs(n+1,b,r)*Clinv(n+1,b)+(1-fs(n+1,b,r))*Clfs(n+1,b);

    if flag_dyn_stall==0
        Cl(n+1,b)=Clstat(n+1,b);
    end
```

```
if r~=length(r_elem)
    L(n+1,b)=0.5*rho*absVrel(n+1,b,r)^2*c(r)*Cl(n+1,b);
    D(n+1,b)=0.5*rho*absVrel(n+1,b,r)^2*c(r)*Cd(n+1,b);
else
    L(n+1,b)=0;
    D(n+1,b)=0;
end
p(n+1,3,b,r)=L(n+1,b)*cosd(phi(n+1,b,r))+D(n+1,b)*sind(phi(n+1,b,r));
p(n+1,2,b,r)=L(n+1,b)*sind(phi(n+1,b,r))-D(n+1,b)*cosd(phi(n+1,b,r));

%finding fg
ac=0.2;
a(n+1,b,r)=(norm(V4(n+1,:,b,r))-norm(V4(n+1,:,b,r)+W(n,:,b,r)))/norm(V4(n+1,:,

%a(n+1,b,r)=abs(W(n,3,b,r)/V4(n+1,3,b,r));
%if a(n+1,b,r)<=ac
if a(n+1,b,r)<=1/3
    fg=1;
else
    %fg=ac/a(n+1,b,r)*(2-ac/a(n+1,b,r));
    fg=1/4*(5-3*a(n+1,b,r));
end

%ABS=sqrt(V4(n+1,2,b,r)^2+(V4(n+1,3,b,r)+fg*W(n,3,b,r))^2);
ABS=norm(V4(n+1,:,b,r)+[0,0,fg*W(n,3,b,r)]);

%Prandtl factor
f=B/2*(r_elem(end)-r_elem(r))/(r_elem(r)*abs(sind(phi(n+1,b,r))));
F=2/pi*acos(exp(-f));

%calc W for next step (W(:,:,3)=Wz=Wn , W(:,:,2)=Wy=Wt)
if r~=length(r_elem)
    Wqs(n+1,3,b,r)=-B*L(n+1,b)*cosd(phi(n+1,b,r))/(4*pi*rho*r_elem(r)*F*ABS);
    Wqs(n+1,2,b,r)=-B*L(n+1,b)*sind(phi(n+1,b,r))/(4*pi*rho*r_elem(r)*F*ABS);
else
    Wqs(n+1,3,b,r)=0;
    Wqs(n+1,2,b,r)=0;
end

%Dynamic wake model
```

```
    if a(n+1,b,r)<=0.5
        tau1=1.1/(1-1.3*a(n+1,b,r))*R/sqrt(V4(n+1,2,b,r)^2+V4(n+1,3,b,r)^2);
        tau2=tau1*(0.39-0.26*(r_elem(r)/R)^2);
        H_din=Wqs(n+1,:,b,r)+0.6*tau1*(Wqs(n+1,:,b,r)-Wqs(n,:,b,r))/delta_t;
        Wint(n+1,:,b,r)=H_din+(Wint(n,:,b,r)-H_din)*exp(-delta_t/tau1);
        W(n+1,:,b,r)=Wint(n+1,:,b,r)+(W(n,:,b,r)-Wint(n+1,:,b,r))*exp(-delta_t/tau2
    end
%     if r==17 && b==1 && W(n+1,3,b,r)==0
%         keyboard
%     end

    if r==length(r_elem)
        thrust_b(n+1,b)=trapz(r_elem,squeeze(p(n+1,3,b,:))');
        Thrust(n+1)=Thrust(n+1)+thrust_b(n+1,b);
        torque_b(n+1,b)=trapz(r_elem,r_elem.*squeeze(p(n+1,2,b,:))');
        Torque(n+1)=Torque(n+1)+torque_b(n+1,b);
    end
        end
    end
    Power(n+1)=Torque(n+1)*w;
end

clearvars ABS ac b r n f F fg ind r_tower theta_tower a_top a_bot flag a_tower


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% RESULTS PART
% Thrust and Power results table
ResultsTable=table({'Thrust','Power'}',...
    [Thrust(end),Power(end)]',...
    [time,time]',...
    'VariableNames',{'Output','Value','Time'});
disp(ResultsTable);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PLOTS PART

%Plot for normal force distribution
figure
plot(r_elem,squeeze(p(end,3,1,:)))
```

```
title(strcat("Normal Force Distribution at time t= ",num2str(T)," seconds"))
ylabel("Normal Force [N/m]")
xlabel("Radial distance on the blade [m]")

%Plot for tangential force distribution
figure
plot(r_elem,squeeze(p(end,2,1,:)))
title(strcat("Tangential Force Distribution at time t= ",num2str(T)," seconds"))
ylabel("Tangential Force [N/m]")
xlabel("Radial distance on the blade [m]")

%Plot for thrust against time
figure
plot(2*delta_t:delta_t:T,Thrust(3:end)/1000)
if flag_tower==1
    title("Total Thrust against Time with tower effect")
else
    title("Total Thrust against Time")
end
ylabel("Thrust [kN*m]")
xlabel("Time [s]")
ylim([0,max(Thrust(3:end))*1.2/1000])
xlim([0,T])

%Plot for power against time
figure
plot(2*delta_t:delta_t:T,Power(3:end)/10^6)
if flag_tower==1
    title("Power against Time with tower effect")
else
    title("Power against Time")
end

ylabel("Power [MW]")
xlabel("Time [s]")
ylim([0,max(Power(3:end))*1.2/10^6])
xlim([0,T])
hold on
plot(2*delta_t:delta_t:T,w*torque_b(3:end,1)/1000000)
plot(2*delta_t:delta_t:T,w*torque_b(3:end,2)/1000000)
```

```
plot(2*delta_t:delta_t:T,w*torque_b(3:end,3)/1000000)

f_low=1/T;
f_high=0.5/delta_t;
fs=1/delta_t;
f=f_low:0.01:f_high;

%PSD Power
[Pyy,f]=pwelch(Power(2:end),1024,64,f,fs);
figure
semilogy(2*pi*f/w,Pyy)
title("PSD for Power")
grid on
xlabel('\omega/\omega_{o}','FontSize',14)
ylabel('PSD [MW^{2}/Hz]','FontSize',14)
set(gca,'FontSize',14)

%PSD Thrust
[Pyy,f]=pwelch(Thrust(2:end),1024,64,f,fs);
figure
semilogy(2*pi*f/w,Pyy)
title("PSD for Thrust")
grid on
xlabel('\omega/\omega_{o}','FontSize',14)
ylabel('PSD [(kN*m)^{2}/Hz]','FontSize',14)
set(gca,'FontSize',14)
```