MAE 579 Wind Energy Fall 2021

HW3 Velocity Azimuth Display

Wind Analysts: Rajesh Ramesh, Pranav Pawar

Arizona State University

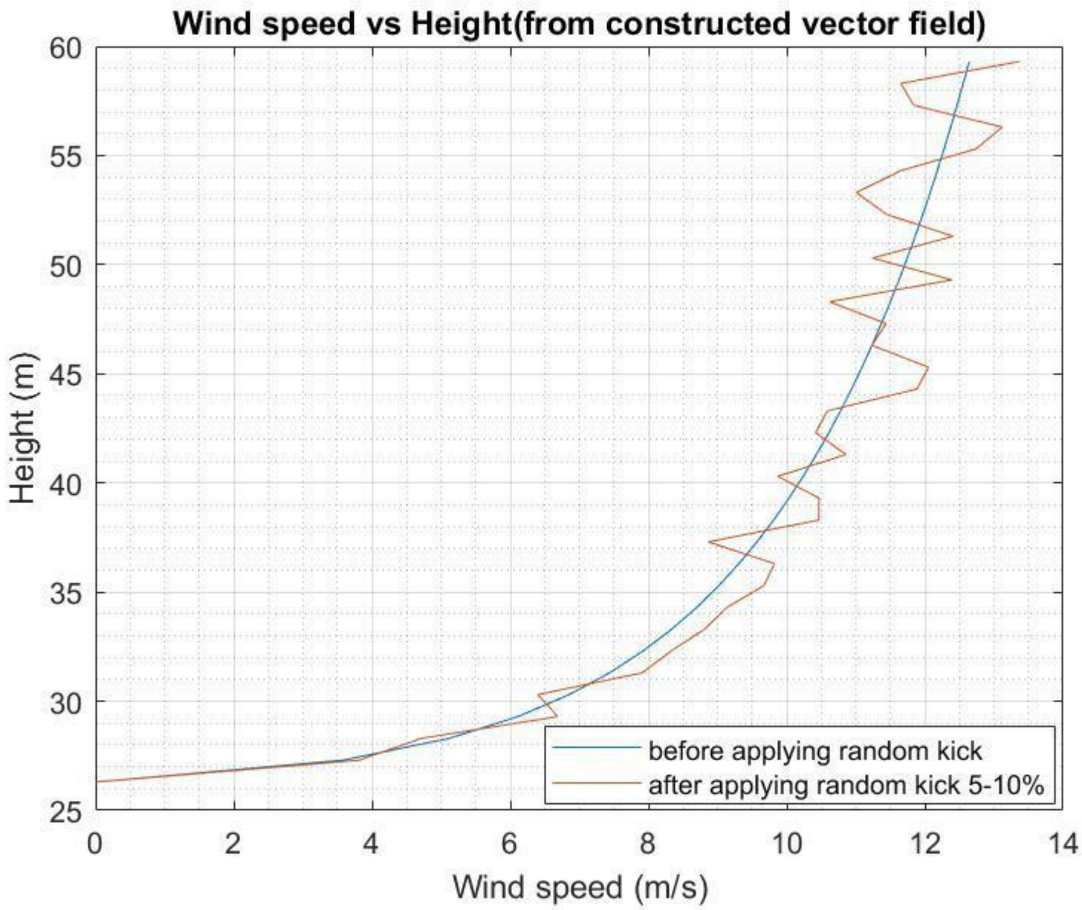

Figure 1. Horizontal velocity vectors (Wind speed) Vs Height (Constructed vector field)

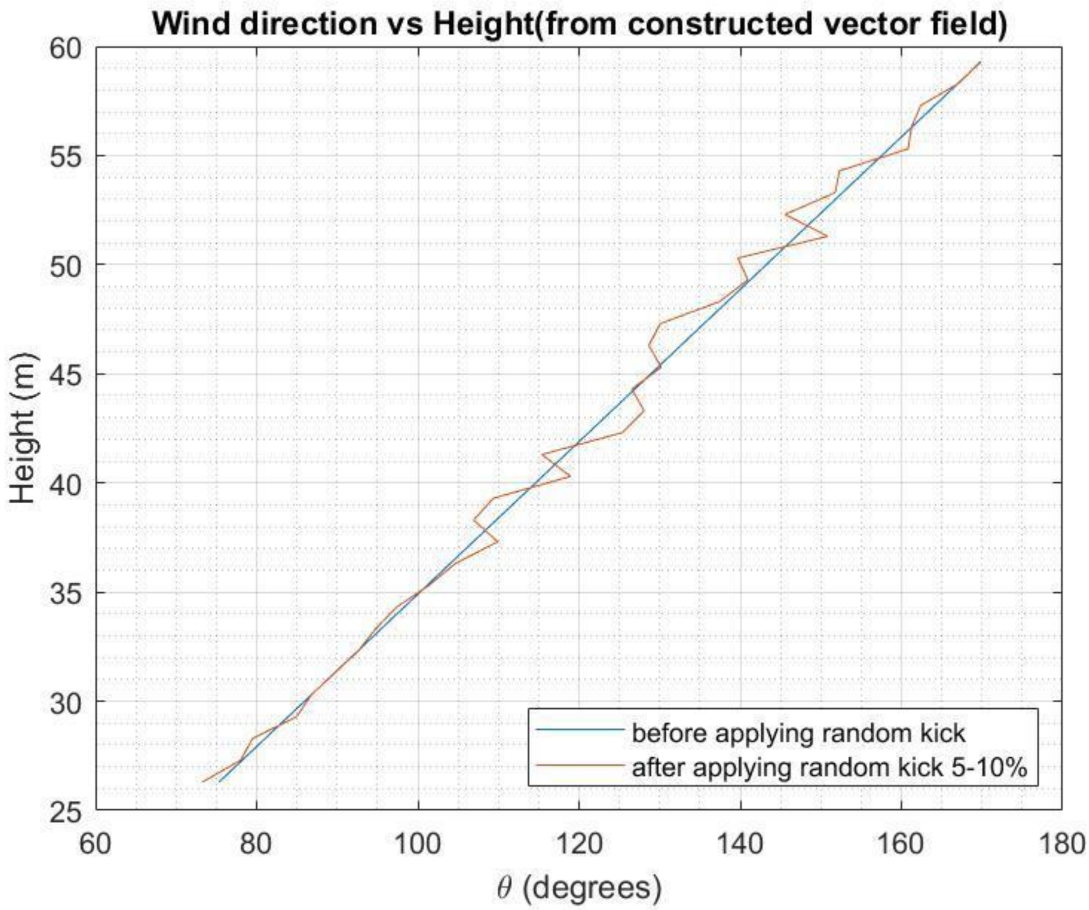

Figure 2. Wind direction Vs Height (Constructed vector field)

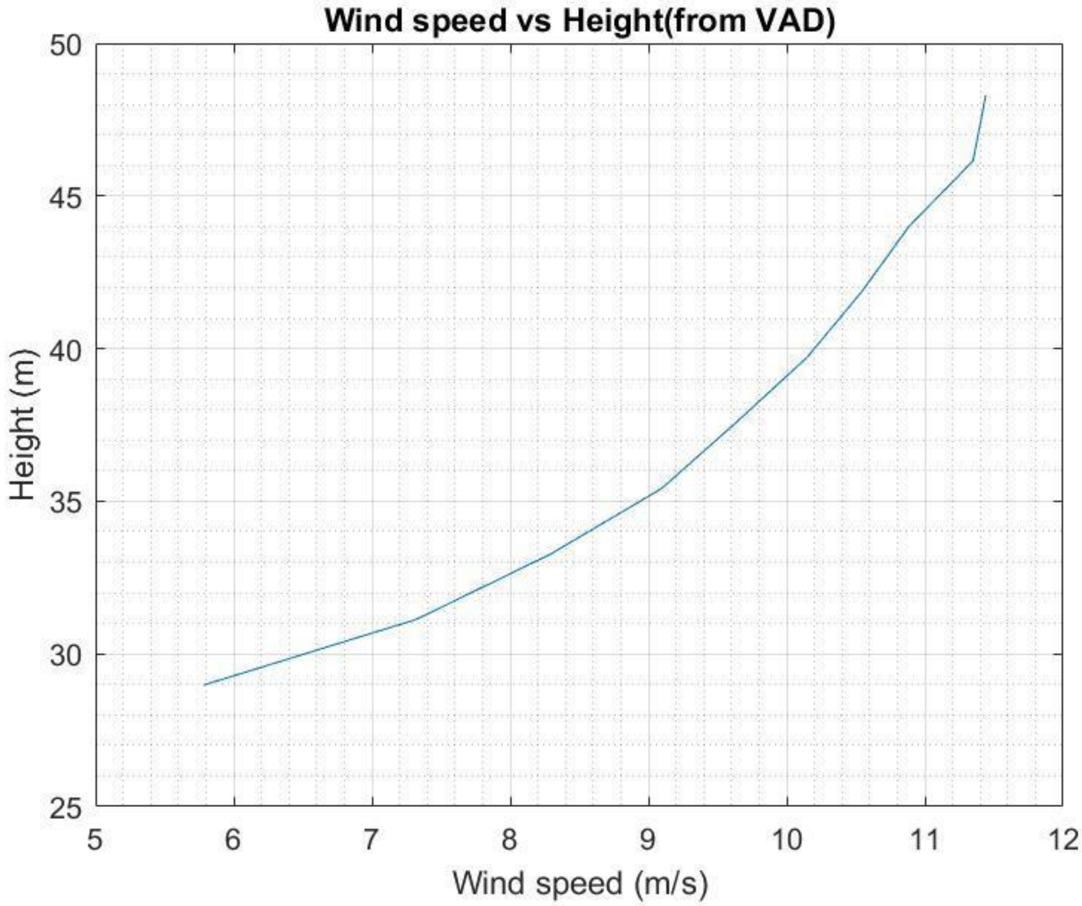

Figure 3. Wind speed from VAD

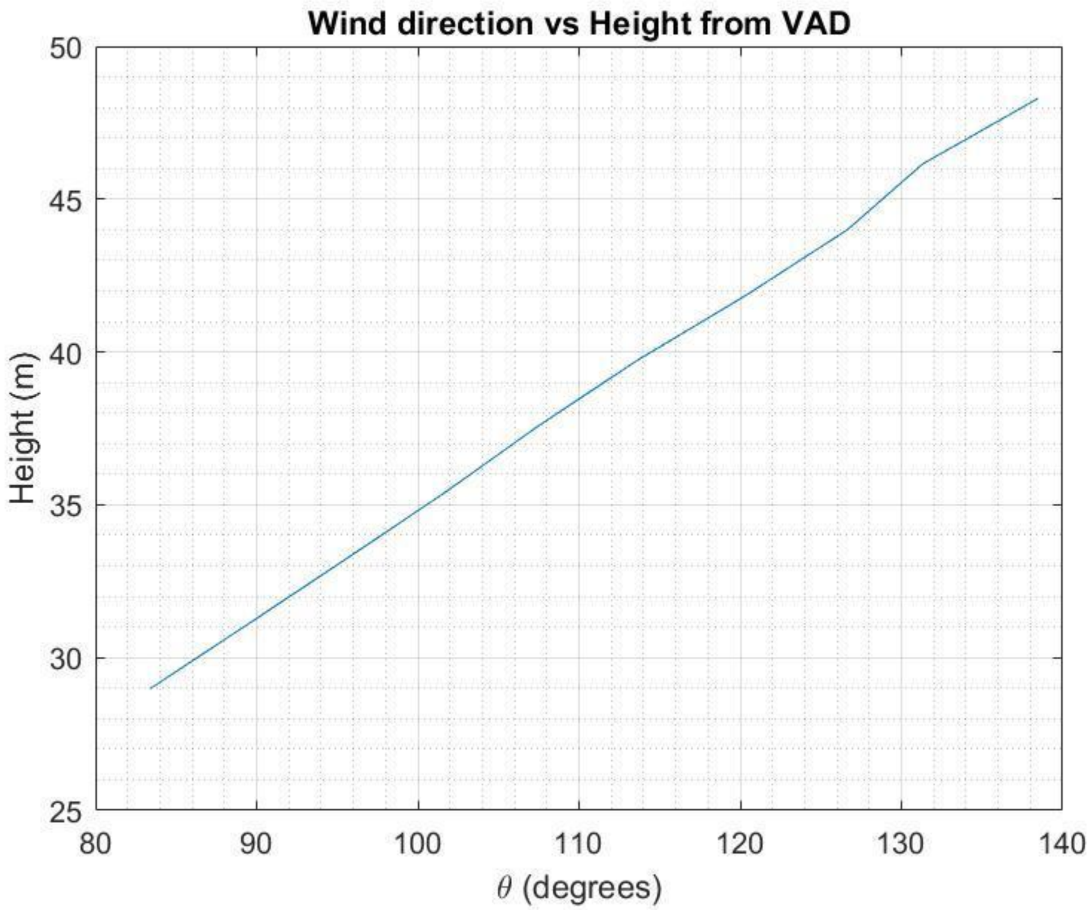

Figure 4. Wind direction from VAD

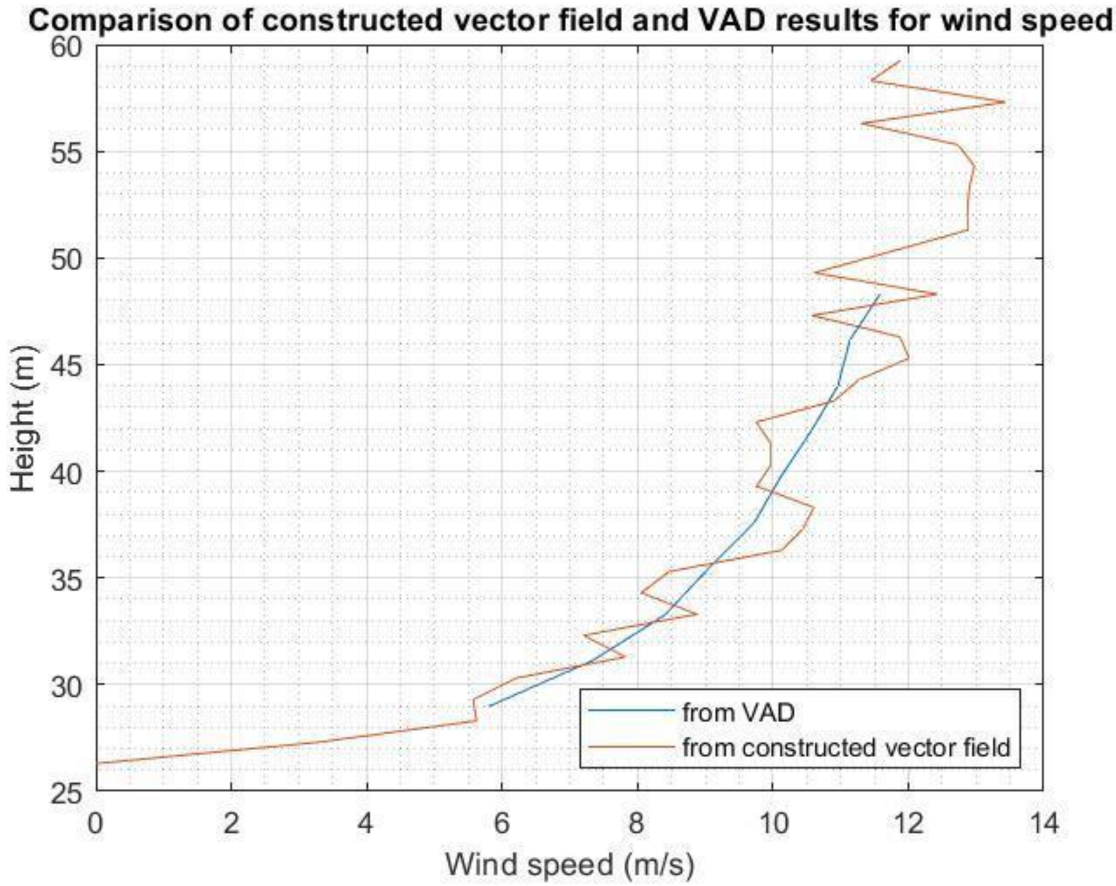

Figure 5(a). Comparison of constructed vector field and VAD results

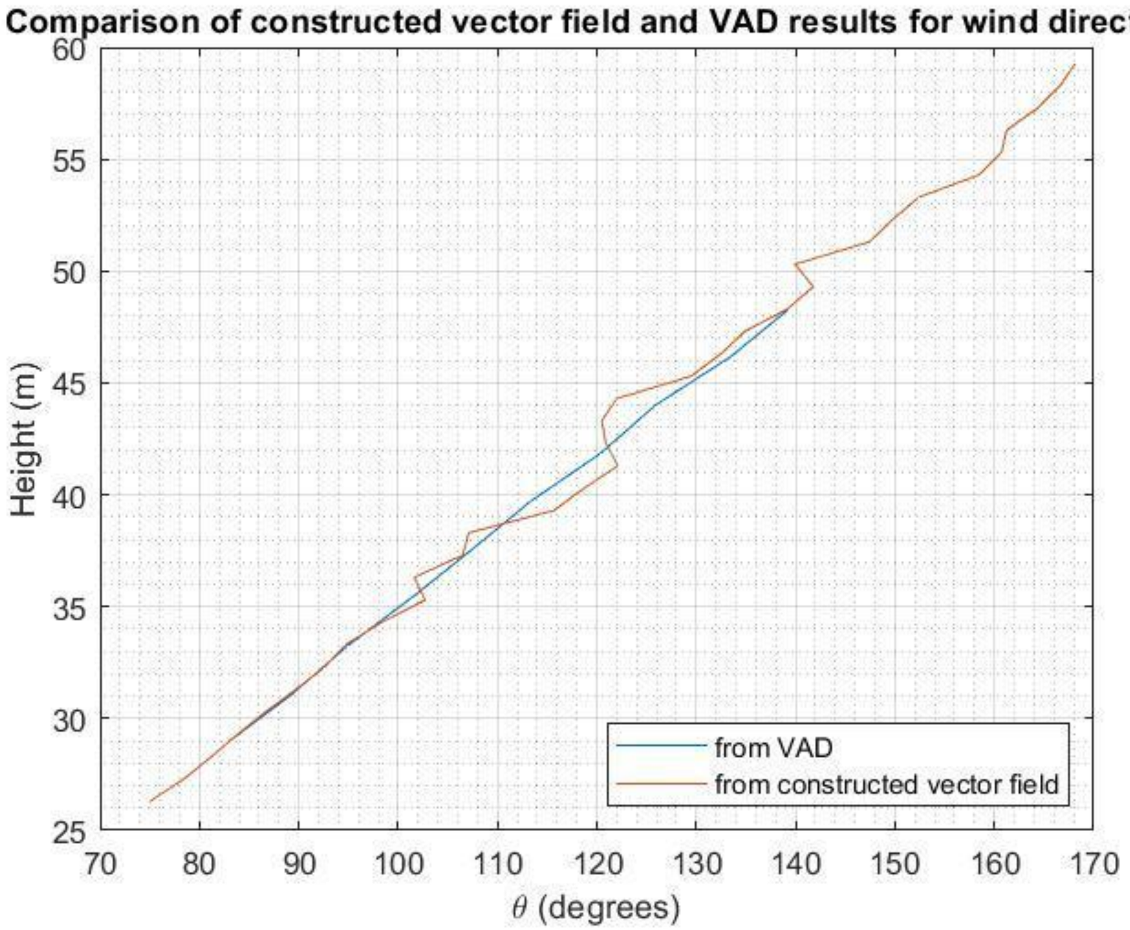

Figure 5(b). Comparison of constructed vector field and VAD results

2.

a. VAD can capture the real mean profiles, but it fails to capture the vector field variations (random kick) because VAD assumes vectors to be homogenous along the horizontal plane.

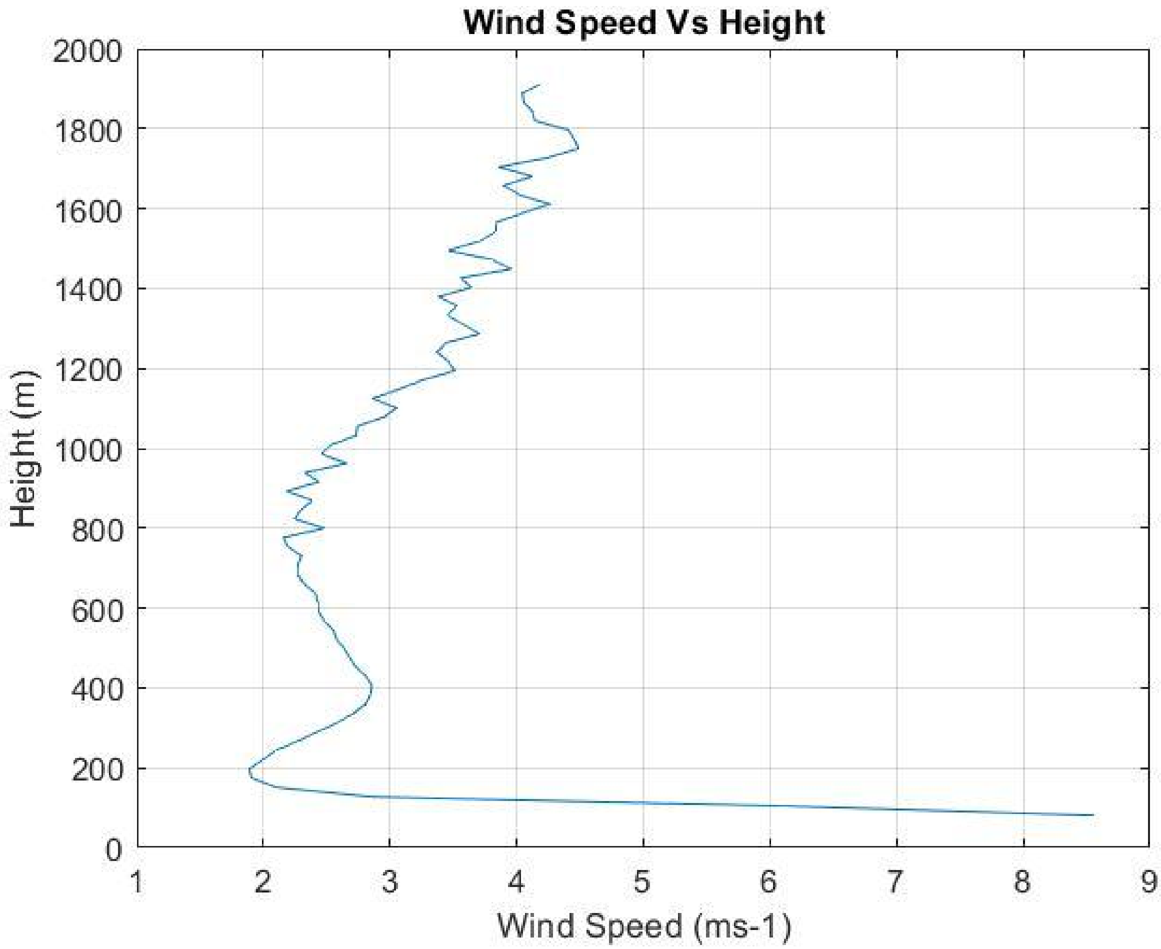

Figure 6. Mean wind speed Vs Height

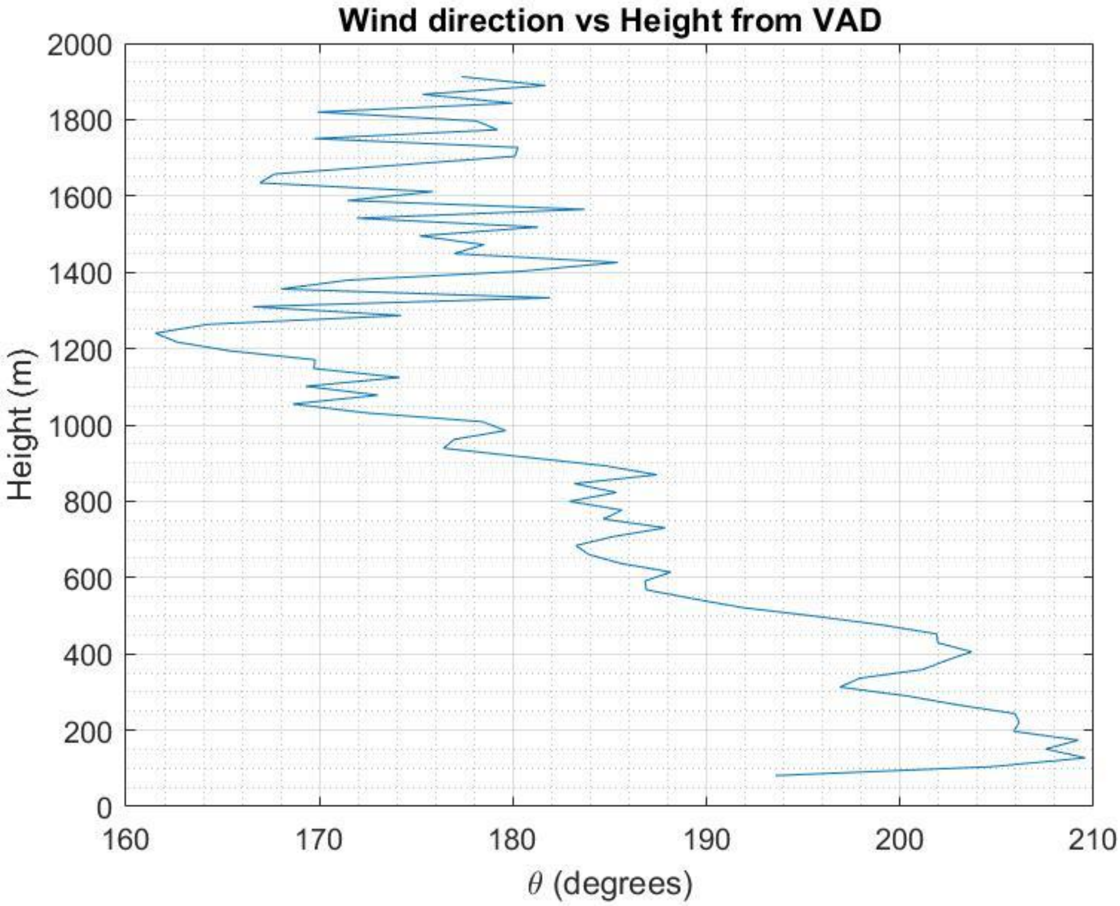

Figure 7. Mean wind direction Vs Height

5. Q&A

1. Nocturnal jets prevalence is seen at night and higher wind intensities are seen because of shear-driven turbulence. These wind regions can assist in generating high wind power.
2. Day time boundary layer has three parts: mixed layer (convective BL), surface layer and cloud layer. And a few centimeters above the ground there is a layer called microlayer or interfacial layer. Clouds in between mixed layers divide it into two halves: cloud layer and sub cloud layer. Towards the ground level, logarithmic profile is followed within the surface layer.
3. Land. Diurnal cycles transformation for land are observed as temperature changes are more and boundary layer profile variations are seen for day night time period. As compared to sea, diurnal cycle changes are less as the sea surface temperature variation is little.
4. Wind direction angles increase steeply as they approach the ground. Wind directions cross isobars at small angles, in a mixed layer at daytime. At night time, wind direction angles vary at large. For the night time period at 4am on 22-10-2013. The wind direction remains constant with height.
5. With an altitude increase, wind speed increases as it suffers less resistance from obstacles. High values of wind shear observed in the potential rotor area captures high wind velocities and in turn generates more wind power.
6. For lower wind directional shear variations, the turbine blades can move to align perpendicular to wind direction. But when wind directional shear variations are large wind profiles become unpredictable and hard to predict wind power.

7. Fronts usually divert the wind direction and the meteorological weather conditions in the region can be predicted from the type fronts and their interactions. For 17:11:37 15-10-2013 time period, the wind direction changes are large, signaling sudden changes in meteorological weather conditions.
8. Like Stull's boundary layer, diurnal cycles are captured with nocturnal behavior dominant at night time.

MATLAB Code

```
close all
clear


%Defining constants for Log Law
z0=0.4;   %aerodynamic roughness length  %stull fig9.6 %dense forest
k=0.35;
d=25.9;
u0=1;  %surface stress

%Defining constans for Ekman Spiral
ekman=0.1;

lidar_position_x=10;  %x coordinate of lidar origin
lidar_position_y=10;  %y coordinate of lidar origin


x=[0:1:20];  %x-axis coordinates
y=[0:1:20];  %y-axis coordinates
z=[z0+d+0.01:1:60]; %z-coordinates along height

[X,Y,Z]=meshgrid(x,y,z);  %3D grid coordinates defined by x,y,z

W=0*X; %assuming wind velocity is horizontal
%defining horizontal wind velocity components
V(:,:,:)=(u0/k)*log((Z(:,:,:)-d)/z0).*sin(ekman*(Z(:,:,:)));  %x component of wind
velocity
U(:,:,:)=(u0/k)*log((Z(:,:,:)-d)/z0).*cos(ekman*(Z(:,:,:)));  %y component of wind
velocity

%fetching data for vertical velocity profile
u=U(lidar_position_x,lidar_position_y,:);
v=V(lidar_position_x,lidar_position_y,:);

figure(1)
plot(sqrt(v(:).^2+u(:).^2),z(:));  %plotting wind speed profile at lidar position
(10,10)
hold on


figure(2)
plot(rad2deg(myatan(v(:),u(:))),z(:)); %plotting wind direction profile at lidar
position (10,10)
hold on


a=5/100; %minimum kick of 5 percent   %
b=10/100; %maximum kick of 10 percent %

random_kick_flag="yes";  %whether to apply random kick

if random_kick_flag=="yes"
    %applying random kick to U component
    random_kick_sign=sign(2.*rand(length(x),length(y),length(z))-1);    %randomly
decide whether kick increases or decreases velocity
    random_kick_percentage=((b-a).*rand(length(x),length(y),length(z))+a);  %randomly
decide the kick value between 5-10 percent %
    random_kick=1-random_kick_sign.*random_kick_percentage;
    U=U.*(random_kick);
```

```
    %applying random kick to V component
    random_kick_sign=sign(2.*rand(length(x),length(y),length(z))-1); %randomly decide
whether kick increases or decreases velocity
    random_kick_percentage=((b-a).*rand(length(x),length(y),length(z))+a);  %randomly
decide the kick value between 5-10 percent %
    random_kick=1-random_kick_sign.*random_kick_percentage;
    V=V.*(random_kick);
end

%fetching data with kick for vertical velocity profile
u_kick=U(lidar_position_x,lidar_position_y,:);
v_kick=V(lidar_position_x,lidar_position_y,:);

figure(1)
plot(sqrt(v_kick(:).^2+u_kick(:).^2),z(:));
title('Wind speed vs Height(from constructed vector field)');
xlabel('Wind speed (m/s)');
ylabel('Height (m)');
grid on
grid minor
legend('before applying random kick','after applying random kick 5-
10%','Location','southeast')

figure(2)
plot(rad2deg(myatan(v_kick(:),u_kick(:))),z(:));
title('Wind direction vs Height(from constructed vector field)')
xlabel('\theta (degrees)');
ylabel('Height (m)');
legend('before applying random kick','after applying random kick 5-
10%','Location','southeast')
grid on
grid minor

%range ring
range=linspace(30,50,10); %radial distance from origin of virtual lidar
elevation=75;            %elevation angle in degrees
azimuth=[0:5:355];       %azimuthal angle in degrees



r_z=range*sind(elevation);  % z coordinate of range ring - height
r_x=zeros(length(azimuth),1);
r_y=zeros(length(azimuth),1);


r_u=zeros(length(azimuth),1);
r_v=zeros(length(azimuth),1);

radial_velocity=zeros(length(azimuth),1);




%Loop for constructing lidar data

for ring=1:length(range)
    for i=1:length(azimuth)
        MyData.range(ring,i)=range(ring);   %radial position of points on range rings
        MyData.az(ring,i)=azimuth(i);       %azimuthal position of points on range
rings
        MyData.el(ring,i)=elevation;        %elevation angle of range rings

        %x coordinate of points on a range ring
```

```
        r_x(i)=lidar_position_x+range(ring)*cosd(elevation)*sind(azimuth(i));
        %y coordinate of points on a range ring
        r_y(i)=lidar_position_y+range(ring)*cosd(elevation)*cosd(azimuth(i));

        %interpolate velocity to points on a range ring

        r_u(i)=interp3(X,Y,Z,U,r_x(i),r_y(i),r_z(ring));    %u velocity component at
points on range ring

        r_v(i)=interp3(X,Y,Z,V,r_x(i),r_y(i),r_z(ring));    %v velocity component at
points on range ring


radial_velocity(i)=dot([cosd(elevation)*sind(azimuth(i)),cosd(elevation)*cosd(azimuth(
i)),sind(elevation)],[r_u(i),r_v(i),0]); %virtual lidar
        MyData.rv(ring,i)=radial_velocity(i);        %radial velocity obtained from
virtual lidar
    end
end

Data(1)=MyData;

B=zeros(2,1);

A=zeros(2);

time=1;

%minimizing cost function using least squares method
 for ring=1:size(Data(time).range,1)
        count=0;
        for i=1:size(Data(time).range,2)
             if ~isnan(Data(time).rv(ring,i))==1
                count=count+1;

B(1)=B(1)+Data(time).rv(ring,i)*cosd(Data(time).el(ring,i))*cosd(Data(time).az(ring,i)
);

B(2)=B(2)+Data(time).rv(ring,i)*cosd(Data(time).el(ring,i))*sind(Data(time).az(ring,i)
);

A(1,1)=A(1,1)+cosd(Data(time).el(ring,i))*sind(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*cosd(Data(time).az(ring,i));

A(1,2)=A(1,2)+cosd(Data(time).el(ring,i))*cosd(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*cosd(Data(time).az(ring,i));

A(2,1)=A(2,1)+cosd(Data(time).el(ring,i))*sind(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*sind(Data(time).az(ring,i));

A(2,2)=A(2,2)+cosd(Data(time).el(ring,i))*cosd(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*sind(Data(time).az(ring,i));
             end
        end


        output_data(time).velocity(ring,:)=pinv(A)*B;
        A=zeros(2);
        B=zeros(2,1);
 end

figure(3)
plot(vecnorm(output_data(time).velocity(:,:),2,2),r_z);
```

```
title('Wind speed vs Height(from VAD)');
xlabel('Wind speed (m/s)');
ylabel('Height (m)');
grid on
grid minor


figure(4)
plot(vecnorm(output_data(time).velocity(:,:),2,2),r_z);
hold on
plot(sqrt(v_kick(:).^2+u_kick(:).^2),z(:));
title('Comparison of constructed vector field and VAD results for wind speed');
xlabel('Wind speed (m/s)');
ylabel('Height (m)');
legend('from VAD','from constructed vector field','Location','southeast');
grid on
grid minor


figure(5)
plot(rad2deg(myatan(output_data(time).velocity(:,2),output_data(time).velocity(:,1))),
r_z);
title('Wind direction vs Height from VAD')
xlabel('\theta (degrees)');
ylabel('Height (m)');
grid on
grid minor


figure(6)
plot(rad2deg(myatan(output_data(time).velocity(:,2),output_data(time).velocity(:,1))),
r_z);
hold on
plot(rad2deg(myatan(v_kick(:),u_kick(:))),z(:));
title('Comparison of constructed vector field and VAD results for wind direction')
xlabel('\theta (degrees)');
ylabel('Height (m)');
legend('from VAD','from constructed vector field','Location','southeast');
grid on
grid minor
```

```
close all
clear

% load matlab HW3 VAD file
load('Data_for_VAD (1).mat')

% AX=B, Sector VAD processing
% Least square method, Minimise L to find u,v
% Cost Function - L = sum(u*r(n)-u(rn))^2 minimize, take derivatie
% Radial unit vector the lidar to point on range ring - rn(cap)=sin(az)*cos(el)*x(cap)
+ cos(az)*cos(el)*y(cap) + sin(el)*z(cap)
% u(rn) - Radial velocity measured by the lidar

% Minimise L wrt u and v
% dl\du = 2*summation*(u*rn(cap)-u(rn))^sin(az)*cos(el)
% dl\dv = 2*summation*(u*rn(cap)-u(rn))^cos(az)*cos(el)
```

```
% dl\du=0, dl\dv=0; 2*2 Linear system, solve for u and v
% solve as AX=B 2*2 linear system


% Preallocate A, B
B=zeros(2,1);
A=zeros(2);

% 512 Data set points
T=512;


%
for time=1:size(Data,2)
% loop over all time intervals
%for time=1:T
    % loop for all range ring
    for ring=1:size(Data(time).range,1)

        count=0;
        % Locate ith column
        for i=1:size(Data(time).range,2)
            % test for nan if NaN dont claculate A,B matrices, ignore
            % unusually high values of radial velocities
             if ~isnan(Data(time).rv(ring,i))==1 && abs(Data(time).rv(ring,i))<15
                count=count+1;

B(1)=B(1)+Data(time).rv(ring,i)*cosd(Data(time).el(ring,i))*cosd(Data(time).az(ring,i)
);

B(2)=B(2)+Data(time).rv(ring,i)*cosd(Data(time).el(ring,i))*sind(Data(time).az(ring,i)
);

A(1,1)=A(1,1)+cosd(Data(time).el(ring,i))*sind(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*cosd(Data(time).az(ring,i));

A(1,2)=A(1,2)+cosd(Data(time).el(ring,i))*cosd(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*cosd(Data(time).az(ring,i));

A(2,1)=A(2,1)+cosd(Data(time).el(ring,i))*sind(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*sind(Data(time).az(ring,i));

A(2,2)=A(2,2)+cosd(Data(time).el(ring,i))*cosd(Data(time).az(ring,i))*cosd(Data(time).
el(ring,i))*sind(Data(time).az(ring,i));
             end
        end

%%%%%%%%        % if a range ring has more than one not a NaN, do Pinv
        if count>1
            % get velocity vector (u,v)
            output_data(time).velocity(ring,:)=pinv(A)*B;

            % remove velocities > 15 to reduce noise in graphs, take in
            % velocities less than 15
            if
sqrt((output_data(time).velocity(ring,1))^2+(output_data(time).velocity(ring,2))^2)<15
                % Calculate resultant vector from u,v components and assign
                % it to output_data structure

output_data(time).wind_speed(ring,1)=sqrt((output_data(time).velocity(ring,1))^2+(outp
ut_data(time).velocity(ring,2))^2);
            else
                % if velocitites greater than 15, set it to NaN
```

```
                output_data(time).wind_speed(ring,1)=NaN;
            end

            % assign structure array to height to output_data, height = range * sin
(el)

output_data(time).height(ring,1)=Data(time).range(ring,1)*sind(Data(time).el(ring,1));

            %vecnorm calculatess reultant
            %wind_speed(ring,1)=vecnorm(velocity(ring,:));

            % assign output_data structure array to wind direction
            % myantan converts tan (negative angle) values to postive
            % To get linear graph over zigzag

output_data(time).wind_direction(ring,1)=rad2deg(myatan(output_data(time).velocity(rin
g,2),output_data(time).velocity(ring,1)));

            % Reset A,B when did Pinc for one ring, before moving to next
            % range ring set to zero
            A=zeros(2);
            B=zeros(2,1);
        else
            % Set the radial velocties (u,v) calculated to NaN instead of
            % zero
            output_data(time).velocity(ring,:)=[NaN NaN];
            output_data(time).wind_speed(ring,1)=NaN;
            output_data(time).wind_direction(ring,1)=NaN;

output_data(time).height(ring,1)=Data(time).range(ring,1)*sind(Data(time).el(ring,1));

            A=zeros(2);
            B=zeros(2,1);
        end
    end
end



%plot(sqrt(output_data(t).velocity(:,1).^2+output_data(t).velocity(:,2).^2),Data(t).ra
nge(:,1).*sind(Data(t).el(:,1)))

%hold on
% Preallocate
wind_speed_mean=zeros(size(Data(1).range,1),1);
wind_direction_mean=zeros(size(Data(1).range,1),1);
% notnancount
notnancount=zeros(size(Data(time).range,1),1);

% loop for all rings
for ring=1:size(Data(time).range,1)

    % loop for all time
    for time=1:T

        % Check if NaN, then store in output
        if ~isnan(output_data(time).wind_speed(ring,1))==1
            % Get mean wind speed and direction from output_data

wind_speed_mean(ring)=wind_speed_mean(ring)+output_data(time).wind_speed(ring,1);
```

```
wind_direction_mean(ring)=wind_direction_mean(ring)+output_data(time).wind_direction(r
ing,1);
            % counts number of data points where wind speed value is not NaN
            notnancount(ring)=notnancount(ring)+1;
        end

    end
    % Set zero calculated zero values to NaN, wind velocity unknown cant be
    % zero
    if notnancount(ring)==0
        wind_speed_mean(ring)=NaN;
        wind_direction_mean(ring)=NaN;
    end
end

% Create movie for all wind velocities and directions
% for all
% get figure
h = figure;
title('Wind speed profile')
xlabel('Wind speed (m/s)');
ylabel('Height (m)');
grid on
grid minor
% get axis of current figure
axis([0 30 0 2000])
ax = gca;
% acts like clf,clear fig slate
ax.NextPlot = 'replaceChildren';

loops = T;  %time
% Preallocate M loop array
M(loops) = struct('cdata',[],'colormap',[]);

h.Visible = 'off';
v = VideoWriter('speed_profile.avi');
v.FrameRate=15;
open(v);

% Captures frame lopp
for j = 1:loops
    % plot wind speed V height for all movie frames
    plot(output_data(j).wind_speed(:),output_data(j).height(:));
    date_time=Data(j).name(1:4)+"-"+Data(j).name(5:6)+"-"+ Data(j).name(7:8)+ " "+
Data(j).name(10:11)+":"+Data(j).name(12:13)+":"+Data(j).name(14:15);


    legend(date_time);
    % draws plots
    drawnow
    % get each frames
     M(j) = getframe(gcf);
    writeVideo(v,M(j));
end

%h.Visible = 'on';
% play movie
%movie(M,1,12);
close(v);
```

```
h = figure;
title('Wind direction vs Height from VAD')
xlabel('\theta (degrees)');
ylabel('Height (m)');
grid on
grid minor
% get axis of current figure
axis([0 400 0 2000])
ax = gca;
% acts like clf,clear fig slate
ax.NextPlot = 'replaceChildren';

loops = T;  %time
% Preallocate M loop array
M(loops) = struct('cdata',[],'colormap',[]);

h.Visible = 'off';
v = VideoWriter('direction_profile.avi');
v.FrameRate=15;
open(v);

% Captures frame lopp
for j = 1:loops
    % plot wind speed V height for all movie frames
    plot(output_data(j).wind_direction(:),output_data(j).height(:));
    date_time=Data(j).name(1:4)+"-"+Data(j).name(5:6)+"-"+ Data(j).name(7:8)+ " "+
Data(j).name(10:11)+":"+Data(j).name(12:13)+":"+Data(j).name(14:15);
    legend(date_time);
    % draws plots
    drawnow
    % get each frames
     M(j) = getframe(gcf);
    writeVideo(v,M(j));
end

%h.Visible = 'on';
% play movie
%movie(M,1,12);
close(v);




close all

% Calculate mean wind speed and direction profile at 20th time step
%plot(output_data(20).wind_speed(:),Data(20).range(:,1).*sind(Data(20).el(:,1)))

figure(1)
plot(wind_speed_mean./notnancount,Data(1).range(:,1).*sind(Data(1).el(:,1)));
title('Wind speed vs Height from VAD')
xlabel('Wind speed (m/s)');
ylabel('Height (m)');
grid on
grid minor
figure(2)
plot(wind_direction_mean./notnancount,Data(1).range(:,1).*sind(Data(1).el(:,1)));
title('Wind direction vs Height from VAD')
xlabel('\theta (degrees)');
ylabel('Height (m)');
grid on
```

```
grid minor
```