ME40343 Helicopter Dynamics: Coursework
The Inverse Problem: The Wind Turbine

The purpose of this coursework is to design an 'optimum' wind turbine. This is the design that produces the maximum power for a given wind profile. The design will be produced through gradually building complexity divided into two stages: A. BEMT Model and B. Optimisation.

Specification

An optimal wind turbine is one that produces electrical power at the lowest unit cost. This requires knowledge of the cost function (in £s) for every single component including the manufacture, installation and operating cost. In the absence of this detailed information the goal in this coursework is instead to produce a *technically optimal turbine*, i.e., the turbine that extracts the most energy possible within the limits of the technical bounds set.

The turbine is to be fixed pitch (stall regulated) with a fixed rotational speed of ω = 30rpm and B = 3 blades. Other quantities you may require are shown in Table 1 and Figure 1.

Table 1. *Constants and limits.*

	Quantity	**Symbol**	**Value**	**Units**
DIMENSIONS	Hub Height	H_{hub}	35	m
	Hub-Tower Separation	d_{hub}	3	m
	Root Radius	R_{min}	1	m
	Maximum Radius	R_{max}	20	m
	Mean Chord	c_{mean}	1	m
FORCES	Maximum Root Bending Moment	$M_{root,max}$	0.5	MNm
	Maximum Vertical Hub Force	$F_{Y,max}$	70,000	N
QUANTITIES	Blade Density	ρ_{blade}	2,000	kg/m^3
	Blade Stiffness	$(EI)_{blade}$	-	GPa
	Cut-In Speed (assumed)	V_{min}	5	m/s
	Cut-Out Speed (assumed)	V_{max}	25	m/s
	Weibull Coefficient	A	7	-
	Weibull Coefficient	k	1.8	-
	Rotational Speed	ω	30	rpm

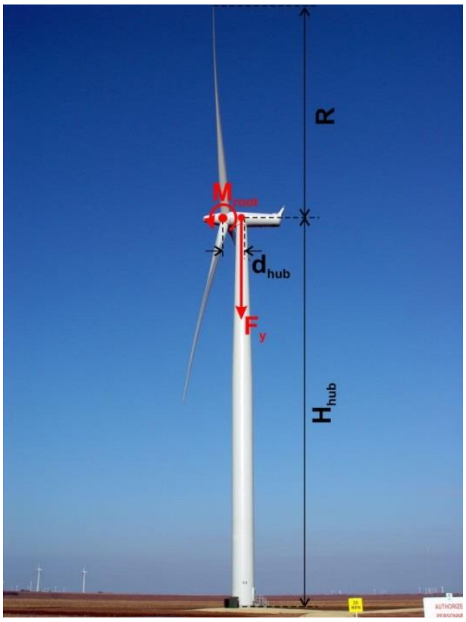

Figure 1: *Wind turbine dimensions.*

The blade stiffness (important for avoiding the blade striking the tower) is a function of the second moment of area. For a first stage design this can be approximated as:

$$(EI)_{blade} \sim 40 * 10^9 * \frac{c(0.2c)^3}{12}$$

Part A: BEMT Model.

Outline: The first stage is to build a BEMT model of the wind turbine which will later be combined with MATLAB optimisation routines. It should therefore be developed in MATLAB. The BEMT model for a wind turbine is very similar to a helicopter rotor, see Chapter 6 of Hansen (available on Moodle). The major difference is that the effect of rotation on the wake (swirl) can be significant for a wind turbine, see Chapter 4 of Hansen. This means we must include two induction factors: *a* represents the proportion of freestream velocity absorbed through the rotor (WT equivalent of induced velocity); a' represents the proportion of rotational velocity added to the freestream flow (swirl).

The solution of annual power from a wind turbine can be divided into three hierarchical levels starting from level 1 at the bottom going to level 3 at the top, see Figure 2. The optimisation for part B then operates above this. The MATLAB functions should be similarly divided (empty function files supplied on Moodle). At this stage assume reasonable parameters for the rotor ($\theta_0 = 12^0$, $\theta_{tw} = -0.4^0$ and $c_{grad} = 0$). After each level is complete you should check your results approximately agree with the results available in the file *ValidationData.xls* before proceeding.

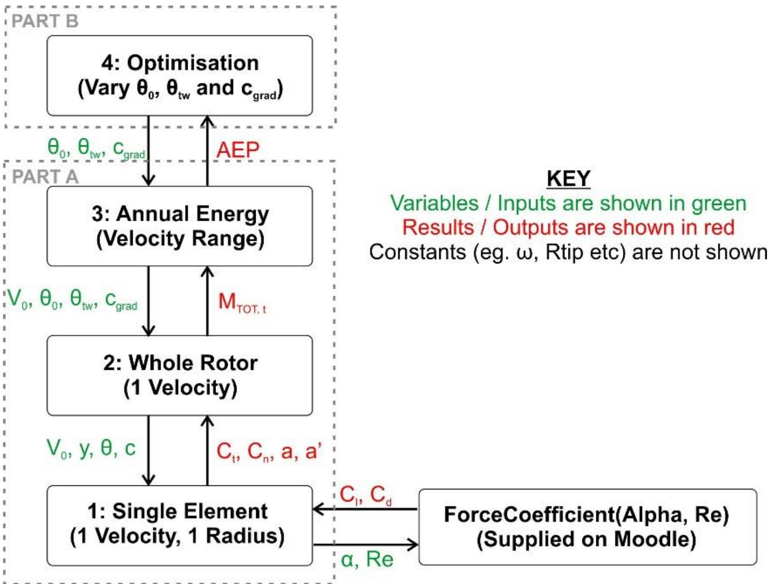

Figure 2: *Suggested MATLAB program architecture. Each box represents a separate script / function.*

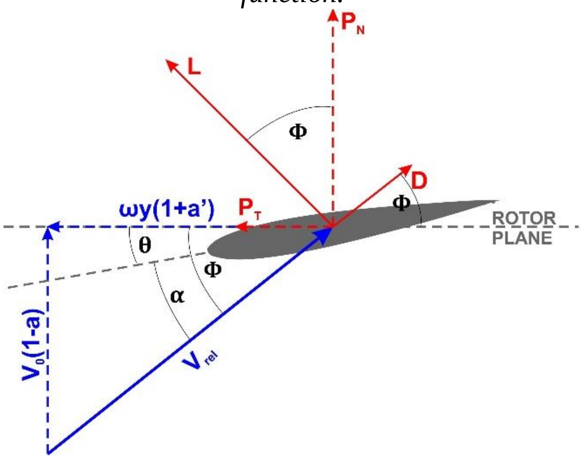

Figure 3: *Velocities and forces for a single element.*

Level 1 - Single Element: the calculation of the forces and induced velocities for a single element. It is an iterative solution because *a* (the proportion of the wind velocity absorbed across the rotor, see Figure 3) and *a'* (the proportion of rotation in the wake) influence the thrust / torque, and the thrust / torque influence a and a'. The solution for each element / radius has to be solved iteratively (all angles are in radians):

1. Initial guess for a and a', typically a = a' = 0 at the start and then the values from the last element.
2. Calculate flow angle ϕ

$$tan\phi = \frac{(1-a)V_0}{(1+a')\omega y}$$

3. Calculate angle α

$$\alpha = \phi - \theta$$

4. Calculate Reynolds number:

$$Re = \frac{\rho V_{REL} c}{\mu}$$

   where V_{REL} is the velocity that the airfoil sees (includes freestream and rotational components):

$$V_{REL} = [(V_o(1-a))^2 + (\omega y(1+a'))^2]^{1/2}$$

5. Use MATLAB function (in Moodle) to get Cl(α, Re) and Cd(α, Re):

```
[Cl, Cd] = ForceCoefficient(Alpha, Re)
```

6. Convert to normal and tangential force coefficients:

$$C_n = C_l cos\phi + C_d sin\phi \qquad C_t = C_l sin\phi - C_d cos\phi$$

7. Calculate a and a' (note that the solidity is local: $\sigma(y) = \frac{Bc}{2\pi y}$)

$$a = \frac{1}{\frac{4sin^2\phi}{\sigma C_n} + 1} \qquad a' = \frac{1}{\frac{4sin\phi cos\phi}{\sigma C_t} - 1}$$

8. If a and a' are outside a specified tolerance, loop back to step 2 with these new values. The error can be defined by (for this error definition a good tolerance is 0.0001):

```
Error = abs(aOUT-aIN)+abs(a'OUT-a'IN);
```

N.B. This is an accurate BEMT model that includes swirl in the wake (*a'*). This means that the iteration has to solve two values simultaneously. This makes the iterative process prone to being unstable. To minimise the risk of instability the values input to the induced factor must be reasonable, otherwise it will go off at a tangent, and a relaxation factor (k ~ 0.1), so that the change between iterations is reduced e.g. $a_{out}=k*(a_{out} - a_{in}) + a_{in}$, should be introduced into step 8. It will slow the iteration down but improve stability. Even with the relaxation factor there are times when the loop will just not converge, it will continue indefinitely. This is because the two values can bifurcate and the iteration becomes stuck. To keep a check on this, add a counter that counts the number of loops. If this counter exceeds 100 you can assume it has gone unstable. Break the loop and create a new one where you solve just for *a* (adapt stages 1 through 8 to remove any mention of *a'*) with *a'* assumed to be zero. It is not as accurate but it is stable and will converge.

Level 2 - Whole Rotor: the numerical integration of the many elements / radii to calculate the moment at the blade root in the rotor plane (torque) and out of plane (blade bending) for a given wind-speed. The individual element moment is defined as:

$$M_i = \left(\frac{1}{2}\rho {V_{REL}}^2 c C_?\right)\Delta y * y_i$$

Where $C_?$ is C_t for torque or C_n for root bending moment, Δy is the radial increment and V_{REL} is the velocity that the airfoil sees (defined above). The C_t and C_n values are outputs of the level 1 function. It will require as input the chord length and geometric angle of attack at that particular radius, for simplicity these can be assumed to vary linearly:

$$c(r) = c_{mean} + \left(y - \frac{R}{2}\right) c_{grad} \qquad\qquad \theta(r) = \theta_0 + y\theta_{tw}$$

Where c_{mean} is the mean chord length (set as 1m), c_{grad} is the taper ratio, θ_0 is the geometric angle at the root and θ_{tw} is the twist angle (in deg/rad per metre). The numerical integration to find the root moments is defined as:

$$M_{TOT,?} = \sum_{i=1}^{N} M_i$$

To convert the torque (based on C_t) into power:

$$P = M_{TOT,t} * B * \omega$$

To check the results are realistic compare the power with the Betz limit: $C_p = \frac{16}{27} = \frac{P}{\frac{1}{2}\rho V_0{}^3 A}$. Your results should always be below it but of the same magnitude, see Figure 4a. As a further check, the data for the default blade ($\theta_0 = 12^o$, $\theta_{tw} = -0.4^o$ and $c_{grad} = 0$) at 20m/s should be around $M_T = 9\text{x}10^4$ Nm which equates to a power of around $P = 8\text{x}10^5$ W (details in *ValidationData.xls*).

Level 3 (Annual Energy Production): the combination of the power–velocity data with probability-velocity data to calculate AEP:

$$AEP = \sum_{i=1}^{N} \frac{1}{2}[P(V_i) + P(V_{i+1})] * f(V_i < V_0 < V_{i+1}) * 8760$$

In essence you just multiply the two curves together as shown in Figure 4. The 8760 is to convert it into the standard units of W*hr/year. The frequency distribution of the wind profile is approximated by the Weibull distribution:

$$f(V_i < V_0 < V_{i+1}) = exp\left\{-\left(\frac{V_i}{A}\right)^k\right\} - exp\left\{-\left(\frac{V_{i+1}}{A}\right)^k\right\}$$

The first exponential term represents the probability of a velocity below V_i, the second term the probability of a velocity below V_{i+1}. The difference between the two is therefore the probability of a velocity between V_i and V_{i+1}. The constants A and k describe the wind profile for a particular site (based on a curve fit to experimental data) where A represents the mean wind speed and k the shape of the profile, see Table 1 for values.

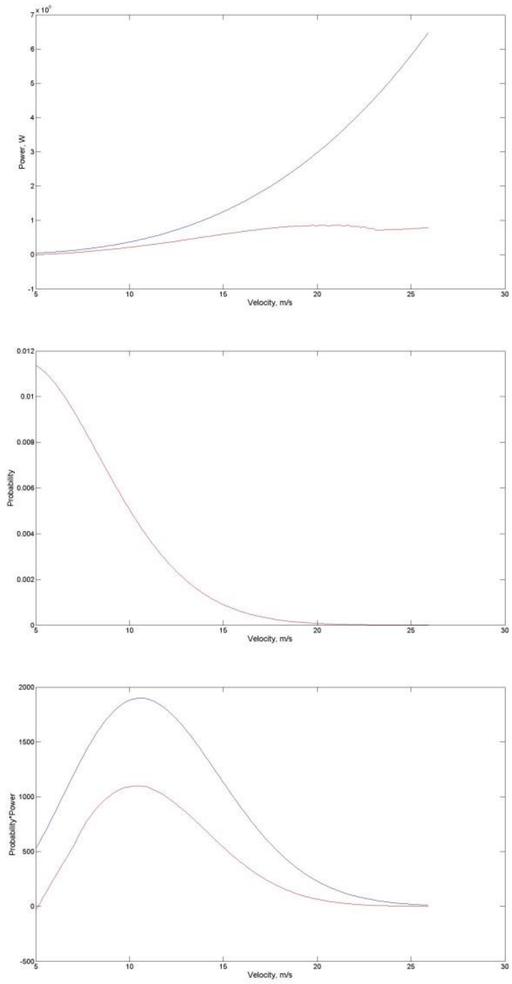

Figure 4: *Calculation of AEP. a) Power extraction against wind speed, b) probability of a particular wind speed, c) power*probability. The total AEP will be 8760 times the numerical integration of c. Red represents the real turbine; blue the theoretical Betz limit of $C_p = 16/27$.*

The inputs at this stage are the linear twist (defined by the variables θ_0 and θ_{tw}) and the linear chord distribution (defined by the constant c_{mean} and the variable c_{grad}). At this stage these must be assumed, good starting values are: $\theta_0 = 12^\circ$, $\theta_{tw} = -0.4^\circ$ and $c_{grad} = 0$. For comparison the AEP for these values should be around 7×10^8 Whr/year (see *ValidationData.xls*).

Part B: Optimisation.

Stage 1 produces a code to predict the AEP from a rotor with assumed: θ_0, θ_{tw}, and c_{grad}. These values now need to be optimised. The MATLAB function required to optimise these is *fminsearchbnd*. This is a very useful function applicable to a very wide range of design / numerical problems. To use it you need to devise a cost function that it will minimise by adjusting the specified parameters within the limits of the bounds you set. It is generally very stable and if it is not converging it is most likely due to a poorly defined cost function (the slightest error in defining the problem can cause it to fail) or poor initial guess (finding local instead of global optima). The core function is (more detail can be found in the function help files):

```
x=FMINSEARCHBND(fun,x0,LB,UB)
```

fun – is the name of the cost function you are trying to minimise, e.g.:

```
function [out1, out2…, outN] = fun(in1, in2, …, inN)
…
end
```

x0 – is a matrix of initial guesses for the input parameters (eg. [in1, in2,…]) used by the cost function. To get good convergence and solve quickly these should be quite close to the final values.

LB / UB – are matrices of lower / upper bounds for the parameters, that stop it from exceeding the limits of reality or the assumptions made in the derivation of the cost function. It will converge much faster and more accurately if these are set as tight as possible, think about what are reasonable / practical limits on the variables θ_0, θ_{tw}, and c_{grad}. For example the specified mean chord length places practical upper and lower bounds on c_{grad} because it means that if c_{grad} is too negative the chord length will be negative (non-existent) at the tip, and if c_{grad} is too far positive the chord length will be negative at the root (it cannot attach to the hub).

x – is a matrix of the parameters output once the optimal solution is found (the breakout conditions / tolerances are defined in the help files).

N.B. Note that x0, LB, UB, and x will all be matrices with a length equal to the number of parameters in the cost function, i.e., [1, 2, …, N]. There is an example of its use on Moodle.

In the current case the parameters to optimise are θ_0, θ_{tw} and c_{grad}. The cost function to minimise is the difference between the current AEP and the ideal AEP, i.e., the AEP for the Betz limit of C_p = 16/27. This needs to operate above level 3 (AEP) and therefore requires a separate function / script file.

Analysis: use the optimisation routine to find the optimum turbine for the case described in the specification. Include a direct comparison with the ideal wind turbine and a simple diagram of the optimal turbine blade.

Discussion: critically assess the work you have undertaken so far to address two questions:

(i) Model Limitations: why might this not be an optimum design? Describe the principle limitations of the model including possible improvements.

(ii) Design Limitation: how could you improve the design further? This could include aspects outside the scope of the optimisation / model like current, emerging and future technologies or changes to the design constraints.

Time permitting, you can also demonstrate the above effects / improvements.

Assessment

You are required to submit a report of maximum length 2,000 words (excluding references and appendices) with no page limit. The report should include the standard sections with 90% of the marks awarded for content (distribution shown in brackets next to each stage) and 10% for the quality of the report.

Table 2: Assessment criteria

	Mark (%)	Description
Summary	5	
Introduction	5	Brief: give motivation (WT sector), technical detail and aims / objectives.
Methodology	5	Brief: can reuse figures from slides / handout (with reference).
Results & Discussion	45	Majority of report, two sub-sections: A. Optimal Design. Compare with Betz limit and explain basic shape of curves. B. Limitations. Discuss the model or design limitations, if implemented demonstrate and explain the effect.
Conclusions	5	Be concise.
References & Appendices	25	Include your final MATLAB code here.
Report Quality	10	Clarity. The goal is to convey your meaning as clearly and succinctly as possible. Use figures, tables and bullet points / lists to help you.

You should submit one electronic copy on moodle.

Deadline: 4pm Thursday 15th November