WT_PERF USER'S GUIDE

by

Marshall L. Buhl, Jr.
National Wind Technology Center
National Renewable Energy Laboratory
Golden, Colorado

December 17, 2004

Introduction

WT_Perf uses blade-element momentum (BEM) theory to predict the performance of wind turbines. It is a descendent of the PROP code originally developed by Oregon State University decades ago. Over the intervening years, many people from many organizations put their marks on the code and it has had several names. More recently, the staff at the NWTC rewrote the program, modernized it, and added new functionality and algorithms.

Retrieving Files from the Archive

Create a folder for your WT_Perf files. You can then download the WT_Perf archive from our server at http://wind.nrel.gov/designcodes/simulators/wtperf. The file is named something such as "wtp_v300.exe," depending upon the version number. Create a WT_Perf folder somewhere on your file system and put this file there. You can double-click on it from Windows Explorer or by entering the file name (currently "wtp_v310") at a command prompt, with the WT_Perf folder as the current directory. This will create some files and folders.

Distributed Files

The files in the WT_Perf archive are:

AlphaChangeLog.txt	The list of changes to WT_Perf for the various alpha versions.
ArcFiles.txt	The list of files that are written to the archive.
Archive.bat	The batch file that creates the archive.
ChangeLog.txt	The list of changes to WT_Perf.
WT_Perf.exe	The WT_Perf executable file.
WT_Perf.pdf	This user's guide in PDF format.
CertTest*.*	Sample input, output, and verification files.
Source*.*	The source-code files for WT_Perf.

Using WT_Perf

The syntax for WT_Perf is

```
WT_Perf <input file>
```

If you do not enter the input-file argument, WT_Perf will display the syntax to remind you. All output files use the same root name as the input file, but will have different extensions. The extensions are as follows:

bed – the blade-element data
ech – the echo of the input data
oup – the primary output file

Creating the WT_Perf Input File

To creating an input file, copy and edit one of the example *.wtp files from the CertTest folder. Do not add or remove any lines, except for the variable-length tables, such as the blade layout, the list of airfoil file names, or the list of combined cases. Do not depend on the values found in the sample input files to be accurate representations of the real turbines—many were modified for convenience. A section-by-section description of an input file follows. In it, variable names use the `Letter Gothic` typeface. All "flag" variables have values of either true or false.

Header

The first line of the file states the type of file. You may change the line, but do not remove it or add additional lines.

Job Title

You have two lines to describe the turbine model. WT_Perf copies the first of the two lines in the header to the output files.

Input Configuration

The **`Echo`** flag tells WT_Perf whether or not to echo the input data to the file "echo.out." If you set it to true, WT_Perf will write out the input values next to their descriptions. If WT_Perf crashes as a results of an input error, checking this file will help you figure out what caused the crash.

If set to true, the **`DimenInp`** flag tells WT_Perf to expect dimensional input parameters. If set to false, some parameters, such as the chord, are assumed to be nondimensional. If you want to use nondimensional input, divide parameters, such as the chord, by the rotor radius. The input-file comments tell you which parameters can be nondimensional. If a parameter can be normalized by the rotor radius, the comment states this in the units for the parameter. For example, the units for the hub radius are "[length or div by radius]," where the word "length" means the parameter has units of length if dimensional, or it is divided by the radius of the rotor if normalized.

The **`Metric`** flag tells whether or not English units are used or if metric units are used. This parameter does not apply to the wind speed. The setting for wind speed is not affected by the **`Metric`** flag. Units used are as follows:

Measurement	**Metric**	**English**
time	seconds	seconds
length	meters	feet
mass	kilogram	slugs
force	newtons	pounds
angle	degrees	degrees

Model Configuration

`NumSect` is the number of pie-wedge sectors around the rotor disk that are used in the calculations. If you set the **`Tilt`**, **`Yaw`**, and **`ShearExp`** (wind-shear exponent) to zero, you need only one sector. This is true because all calculations in all sectors are the same. WT_Perf ignores this parameter in this situation and analyzes one sector. If any of those three parameters are not zero, WT_Perf will use a minimum of four sectors in the analysis. There is virtually no upper limit for the number of sectors.

`MaxIter` limits the number of iteration attempts in the BEM induction loop. First, the code assumes there is no induction effect. It then calculates the angle of attack and looks up the lift and drag for that angle. Next, it uses the lift and drag to compute the induced velocity. Using the new values for the induction, it repeats the process. When the induced velocity changes less than a given amount from one iteration to the next, it exits the induction loop. If, after **`MaxIter`** iterations, the induced velocity changes by more than the given amount, the code terminates the induction loop and sets all output values to a negative number with all nines (for example, the power is given as -999.999 kW).

`ATol` tells WT_Perf how little you want the BEM induction factors to change from one iteration to the next in order to consider it converged.

After BEM convergence for each element, WT_Perf uses the average induction for the entire rotor to compute the skewed wake correction, which it then applies to each element. WT_Perf will recompute the induction for each element using the newly corrected induction values for the initial estimates. It then uses the resulting average induction factor to compute and add the skewed-wake correction to each of those elements. WT_Perf continues this outer iteration until the correction changes by less than **`SWTol`**.

Algorithm Configuration

The **`TipLoss`** and **`HubLoss`** flags tell the code to turn on the Prandtl tip- and hub-loss algorithms. This is usually enabled for non-research work.

The **`SWIRL`** flag tells the code to enable the algorithms for the calculation of the tangential induction factor (swirl). This is generally set to true.

`SkewWake` tells the code to correct the induction factor for a skewed wake. It is ignored if the **`Tilt`** and **`Yaw`** are both zero.

`AdvBrake` invokes the advanced brake state algorithm instead of the classic momentum brake-state model. Most engineers use the advanced brake-state model.

Setting the **`IndProp`** flag to true tells WT_Perf to use its traditional PROP-style induction algorithm. A false setting invokes the algorithm that comes from the PROPX code, which is more similar to the AeroDyn algorithm.

`AIDrag` enables the inclusion of the drag term in the axial-induction algorithm.

`TIDrag` enables the inclusion of the drag term in the tangential-induction algorithm.

Turbine Data

`NumBlade` is the number of blades on the turbine. It must be an integer greater than zero.

`RotorRad` is the rotor radius. It is the distance along the preconed blade, and is therefore a number larger than the swept radius if the precone is not zero.

HubRad is the hub radius entered in either meters or feet if using dimensional data. Otherwise, divide the hub radius by the rotor radius.

The **PreCone** in degrees should be a positive value regardless of whether the turbine is downwind or upwind.

The shaft **Tilt** and nacelle **Yaw** in should be specified in degrees.

HubHt is the hub height entered in either meters or feet if using dimensional data. Otherwise, divide the hub height by the rotor radius..

NumSeg tells WT_Perf how many analysis points there will be along the blade. The input data should be for the centers of the segments.

The next part of this section contains a header followed by **NumSeg** lines defining the distributions of **RElm (**the distance along the blade of the center of the segment from the center of rotation), **Twist**, **Chord**, airfoil file number (**AFfile**), and a flag (**PrntElem**) to tell WT_Perf to generate element data for that segment. If you are entering data in nondimensional form, **RElm** and **Chord** must be normalized by **RotorRad**. Enter **Twist** in degrees.

Aerodynamic Data

The air density (**Rho**) is always entered as a dimensional number. Use either kg/m^3 or slugs/ft^3. For standard temperature and pressure at sea level, use 1.225 kg/m^3 or 0.00238 slugs/ft^3.

For calculating the Reynolds Number, we added the variable **KinVisc**, the kinematic viscosity. For Standard Temperature and Pressure at sea level, use 1.464E-05 m^2/sec or 1.576E-04 ft^2/sec.

ShearExp is the exponent of the power-law wind shear. For the standard 1/7th power law, use 0.143.

The airfoil tables are compatible with AeroDyn. Therefore, they may contain pitching-moment-coefficient data. If so, set the **UseCm** to true.

On the next line, enter the number of unique airfoil-table files (**NumAF**). After that, enter the **NumAF** files on separate lines and enclose the strings (which may include absolute or relative paths) in quotes or apostrophes. The next major section describes the format of the airfoil files.

I/O Settings

Setting the **TabDel** flag to true tells WT_Perf to generate output files with tabs between the columns, instead of using fixed format. Tab-delimited files are best for importing into spreadsheets, while fixed-format files are best for viewing with a text editor or for printing.

By setting the **KFact** flag to true, WT_Perf will output data in the primary results file in “kilo” units. For example, thrust would be in kN or klbf instead of N or lbf, and power would be in kW instead of W.

If **WriteBED** is true, WT_Perf will generate a file containing the blade-element data. Only segments that have their **PrntElem** flag set in the distributed-data block above will be included in the file.

If you enable **InputTSR**, WT_Perf will expect the speed data to be tip-speed ratios (TSR) instead of actual wind speeds. This applies to both combined-case and parametric analyses.

The **SpdUnits** string tells WT_Perf what units are used for wind-speed data. Three possible values are valid: "mps" will tell the code that the wind-speed values are in meters/second, "fps" will indicate that they are in feet/second, and "mph" will indicate that they are in miles/hour. If **InputTSR** is true, this parameter is ignored.

Combined-Case Analysis

The first line in the block is the number of combined cases (**NumCases**) to run. If set to zero, WT_Perf runs no combined cases but performs the old-style parametric analysis.

The second line is the header for the columns in the combined-cases block. It must not be removed from the file.

After that, enter **NumCases** lines containing a combination of speed (wind speed or TSR), rotor speed (**RotSpd**) in rpm, and **Pitch** in degrees.

If **NumCases** is greater than zero, WT_Perf will do the performance analysis for each case and generate a single table containing wind speed, TSR, rotor speed, pitch, power, torque, thrust, flap moment, and power coefficient columns.

Parametric Analysis

If **NumCases** is zero, WT_Perf varies as many as three parameters in each run: rotor speed in rpm, blade pitch in degrees, and wind speed. Enter the wind speed as a tip-speed ratio or an actual wind speed according to the **InputTSR** flag mentioned above. The first three parameters in this section are **ParRow**, **ParCol**, and **ParSht**. They determine how the output data are tabulated for output. If all three parameters are varied, WT_Perf generates multiple tables of data. The **ParRow** parameter determines variation in the table rows. The **ParCol** parameter determines column variation in the tables. The **ParSht** parameter determines which of the parametric values vary from sheet to sheet (table to table).

The next five parameters in this section tell WT_Perf which of the possible output values should be written to the output file. They are rotor power (kW), power coefficient (C_P), rotor torque (N-m or ft-lbf), flap-bending moment at the hub radius (N-m or ft-lbf), and rotor thrust (N or lbf).

This next line tells WT_Perf how to vary the various parameters. The **`PitSt`**, **`PitEnd`**, and **`PitDel`** values define the start, end, and delta pitch angles to use. They are input in degrees. The pitch value is added to the local twist at each segment to determine the angle between the chord line and the plane (or cone) of rotation.

The **`OmgSt`**, **`OmgEnd`**, and **`OmgDel`** parameters define the start, end, and delta rotor speed in rpm.

When specifying the parametric wind speeds, you can either input values in tip-speed ratio (speed of the blade tip divided by the wind speed) or actual wind speeds. If you enable the **`InputTSR`** flag mentioned above, WT_Perf will expect the following line to be tip-speed ratios. **`SpdSt`**, **`SpdEnd`**, and **`SpdDel`** define the start, end, and delta speed. If **`InputTSR`** is false, enter actual wind speeds. The **`Units`** string mentioned above defines the units for actual wind speeds. Three possible values are valid: "mps" will tell the code that the wind-speed values are in meters/second, "fps" will indicate that they are in feet/second, and "mph" will indicate that they are in miles/hour. If **`InputTSR`** is true, this parameter is ignored.

Creating the Airfoil Data Files

The files containing aerodynamic coefficients are compatible with those used by AeroDyn (http://wind.nrel.gov/designcodes/simulators/aerodyn). WT_Perf accepts files formatted to the existing AeroDyn v12 style or the newer, slightly modified style. The differences between the styles are minor. Please see the AeroDyn user's guide for details on the old format of the files.

To tell WT_Perf to assume the file is in the new format, start the first line with the string AERODYN INPUT FILE. The next two lines are for comments. The next line tells WT_Perf how many blocks of data there will be for different Reynolds numbers.

For each block, the first value is the Reynolds number. The next seven lines are for AeroDyn's dynamic-stall model and are ignored by WT_Perf, but you must include them.

A multicolumn table follows. The first column is for the angle of attack in degrees. The values must increase monotonically. The second and third columns are for the lift and drag coefficients. If **`UseCm`** is true, add a fourth column for the pitching-moment data. At the end of the table, add a line containing the string "EOT". The new table format allows one to use different sets of angles of attack for each table.

Known Bugs

None.

Caveats

NREL makes no promises about the usability or accuracy of WT_Perf, which is essentially a beta code. NREL does not have the resources to provide full support for this program. *You may use WT_Perf for evaluation purposes only.*

Acknowledgements

WT_Perf development was funded by the U.S. Department of Energy.

Feedback

Send your comments or bug reports to

Marshall L. Buhl, Jr.
NWTC/3811
National Renewable Energy Laboratory
1617 Cole Blvd.
Golden, CO 80401-3393
United States of America

Web: http://wind.nrel.gov/designcodes/
Email: marshall_buhl@nrel.gov
Voice: (303) 384-6914
Fax: (303) 384-6901