Universidad Zaragoza
1542

# Proyecto Fin de Carrera

## METODOLOGÍA DE TRABAJO PARA LA OBTENCIÓN DE SERIES TEMPORALES (P,v) A PARTIR DE SERIES TEMPORALES DE VELOCIDAD DE VIENTO MEDIANTE EL PROGRAMA FAST


Autor

Juan Ajona Marcén

Director

Julio Javier Melero Estela

Departamento de Ingeniería Eléctrica
Escuela de Ingeniería y Arquitectura


2016

# METODOLOGÍA DE TRABAJO PARA LA OBTENCIÓN DE SERIES TEMPORALES (P,v) A PARTIR DE SERIES TEMPORALES DE VELOCIDAD DE VIENTO MEDIANTE EL PROGRAMA FAST

## RESUMEN

El objeto del presente trabajo trata sobre la utilización del programa FAST para generar series temporales (P,v). Debido a la necesidad de disponer de datos reales de parques eólicos para poder realizar nuevos desarrollos y aplicaciones prácticas, y el difícil acceso a los mismos por su pertenecía a los fabricantes o a los promotores de los parques, se hace necesario la utilización de programas de simulación para su obtención. Tras estudiar la metodología de trabajo con el programa, se han realizado los ajustes necesarios en los archivos primarios de entrada. Se han seleccionado los datos de entrada y elegido los datos de salida necesarios, en este caso únicamente tiempo de simulación, velocidad del viento horizontal a la altura del buje y potencia eléctrica generada. El modelo de aerogenerador se ha elegido entre los casos ya desarrollados por NREL. Es un modelo tripala de 70 metros de diámetro de rotor, paso fijo, y una potencia nominal de 1,5 MW.

Tomando como datos de entrada la información contenida en un conjunto de series de velocidad de viento, obtenidas con medidas sobre el terreno, se ejecuta FAST. El tiempo de simulación son 48 horas divididas en dos días distintos, y 9 series de 9999 segundos superpuestas para evitar errores por la inercia inicial del aerogenerador, cada uno. Una vez realizada la simulación se comprueba que la cantidad de datos obtenidos es muy grande y que al representarlos las curvas que se generan no son determinísticas. Es decir, para cada velocidad de viento no se obtiene un único resultado de potencia. Esto es debido que la velocidad del viento cambia mucho y las palas del aerogenerador tienen una inercia que impide que su velocidad varíe con la misma frecuencia. Generalmente no es un problema puesto que la curva de potencia se calcula con promedios de 10 minutos, obteniéndose una curva útil para calcular potencias medias acumuladas a lo largo de un periodo largo. No obstante en este caso se busca calcular en periodos de 0,1 segundos la potencia generada y cómo se comporta instantáneamente debido a las variaciones en la velocidad del viento. Por ello se hace necesario el uso de la curva de potencia dinámica.

La curva de potencia dinámica se obtiene al aplicar una serie de complejos cálculos estadísticos y matemáticos a los datos obtenidos, teniéndose en cuenta todos y cada uno de los binomios potencia-velocidad. Por definición, la curva dinámica necesita medidas a alta frecuencia (0.01 - 0.1 Hz) que cubran todo el espectro de trabajo del aerogenerador. Por eso se ha mantenido el mismo aerogenerador pero no se han utilizado los datos de las series de velocidad de viento del apartado anterior. Para obtener las series de datos para la curva dinámica, se ha utilizado el programa TURBSIM para generar 21 series, a distintas velocidades, compuestas por 60 casos cada una de 600 segundos de duración. A continuación se procesan con FAST para obtener el binomino P-v para, finalmente, calcular la curva de potencia dinámica del aerogenerador.

# CONTENIDO

# 1. INTRODUCCIÓN

## 1.1. Contexto

La investigación en el campo de la energía eólica: aerogeneradores, sus distintas mejoras o evoluciones, parques eólicos, su rendimiento, rentabilidad o situación; requiere datos reales lo más precisos posibles para poder confrontarlos con los resultados teóricos. Las series de velocidad de viento de un lugar pueden ser de acceso público, o se puede instalar un anemómetro para obtener dicha información. No obstante los datos de producción de los aerogeneradores generalmente pertenecen tanto a las empresas desarrolladoras y fabricantes como a los promotores de parques eólicos, no permitiéndose acceso público ni su uso en investigación por los acuerdos de confidencialidad a los que se ven sujetos. Se hace entonces necesaria la obtención de datos no confidenciales o bien la transformación de los datos confidenciales de forma que mantengan sus propiedades fundamentales a la par que se hace imposible su reconocimiento, es decir consiguen hacerse anónimos.

El tipo de datos mayoritariamente usados son las series temporales de potencia y velocidad de uno o más aerogeneradores, dicho binomio es usado para la construcción de la curva de potencia. La transformación de estos datos puede hacerse manteniendo la serie temporal de velocidad de viento y transformando la potencia. Para ello pueden usarse curvas de potencia de otros tipos de aerogeneradores obteniendo así nuevas series temporales de potencia. El principal problema de éste método es debido a que las curvas potencia se obtienen a partir de valores promediados (generalmente 10-minutales) por lo que las series de potencia obtenidas no reflejan variaciones rápidas del viento.

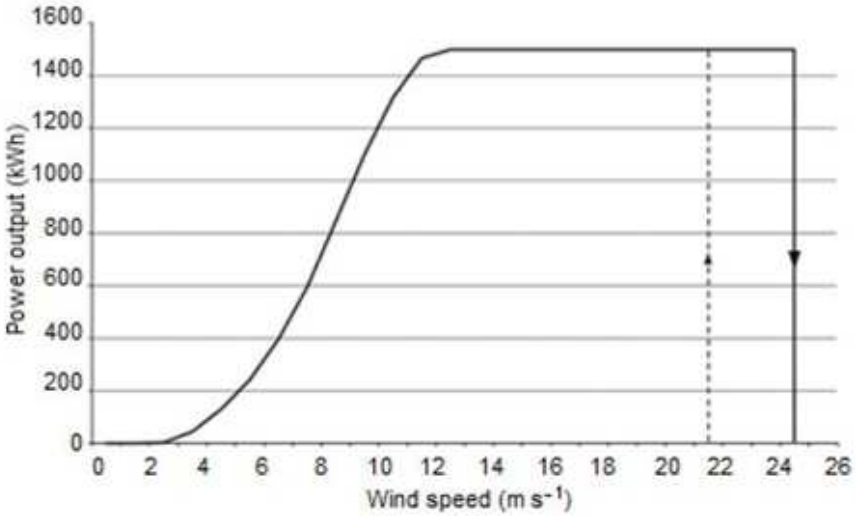


**Figura 1**. *Curva de potencia típica de un aerogenerador de 1.5 MW*

Otra posibilidad es la utilización de programas de simulación que permiten utilizar series temporales de viento para obtener la respuesta de modelos de aerogeneradores lo suficientemente detallados (generalmente modelos aerolásticos), pudiéndose obtener series temporales de potencia (entre otros muchos resultados). Estas series temporales de potencia

reflejan las variaciones rápidas del viento incidente dado que los cálculos para obtener la respuesta del aerogenerador se realizan con modelos muy complejos y avanzados.

Este proyecto se basa en la segunda metodología presentada para la obtención de series temporales (P,v) de alta frecuencia (entre 10 y 15 Hz). El programa de simulación elegido es FAST (Fatigue, Aerodynamics, Structures, and Turbulence) desarrollado por NREL (National Renewable Energy Laboratory). Se ha elegido este programa por ser software libre y estar verificados y reconocidos sus resultados por la comunidad científica. El objetivo fundamental de proyecto es el desarrollo de un flujo de trabajo, mediante el programa FAST, para conseguir obtener series de datos (P,v) a partir de series temporales de viento. La complejidad del software hace necesario un estudio detallado del mismo para comprender tanto su base de funcionamiento como sus mecanismos de operación y distintas variables de entrada por parte del usuario. Las series de datos de velocidad del viento se tomarán de emplazamientos reales.

## 1.2 Objetivos principales

Los objetivos principales de este trabajo son los siguientes:

- Aprendizaje de la metodología de trabajo del programa FAST.
- Determinación de las variables y formatos de entrada del programa.
- Selección del modelo de aerogenerador a utilizar en las simulaciones.
- Obtención de series temporales (P,v) a partir de series de velocidad de viento de emplazamientos reales.
- Cálculo de la curva de potencia dinámica del modelo de aerogenerador elegido, mediante datos generados por TURBSIM y FAST. Y comparación con un modelo previo.

## 1.3 Estructura del trabajo

La estructura de la memoria se ha organizado de la siguiente manera:

- En el Capítulo 1 se ha elaborado una introducción al tema que se va a tratar en el trabajo, especificando los objetivos principales y alcance del mismo.
- En el Capítulo 2 se ha procedido a la presentación del programa FAST, las posibilidades que ofrece y la metodología de trabajo utilizada.
- El Capítulo 3 presenta las variables seleccionadas, los datos de entrada, el modelo de aerogenerador escogido y las razones que han llevado a ello. También está dedicado a los datos generados por el programa FAST tras la introducción de las series de velocidad de viento y el análisis de los mismos.
- En el Capítulo 4 se presenta el concepto de curva de potencia dinámica y se resume el método de cálculo, mostrando el método usado para la generación de las series temporales de viento requeridas y los resultados obtenidos al calcular la curva dinámica de potencia.
- Por último, en el Capítulo 5 se presentan las conclusiones a partir de los datos obtenidos y se desglosa y valoran las posibles líneas de investigación futuras en relación con el tema tratado en el presente estudio.
- Además se han incluido las referencias bibliográficas que han servido de apoyo para la realización de este proyecto, los índices de tablas y figuras, y varios anexos con información complementaria.

# 2. PROGRAMA DE SIMULACION FAST

## 2.1. Presentación FAST

Los grandes aerogeneradores actuales son unas máquinas muy complejas que combinan técnicas de distintas áreas de la ingeniería para su funcionamiento. En la fase de diseño es necesario evaluar su respuesta tanto desde el punto de vista eléctrico, para producción de energía, como mecánico al ser un sistema sometido a grandes cargas sobre su estructura. Para ello, se han desarrollado herramientas avanzadas que permiten simular dicha respuesta. Una de ellas es el código FAST (Fatigue, Aerodynamics, Structures and Turbulence modeling) desarrollado gracias a la colaboración entre NREL (National Renewable Energy Laboratory, es decir, el Laboratorio Nacional de Energías Renovables de EEUU) y la Universidad de Oregón (Oregon State University). Como complemento, NREL desarrolla otros códigos agrupados bajo la denominación de Aerodyn, estos códigos se encargan de la simulación de la respuesta aerodinámica de los aerogeneradores [1].

El programa FAST es un simulador integral que permite el cálculo de múltiples variables de un aerogenerador sometido a una serie de velocidad de viento. Algunas de ellas son las cargas en los extremos de las palas, la fatiga en las palas y en la estructura, la potencia generada, la velocidad de los ejes, la orientación de la góndola, etc. El código FAST es el resultado de la unión de tres códigos distintos: FAST2 para dos palas, FAST3 para tres palas y las subrutinas aerodinámicas Aerodyn para turbinas de tipo HAWTs (Horizontal-axis wind turbines). En las últimas actualizaciones se han centrado en la mejora del cálculo de aerogeneradores offshore permitiendo la simulación de turbinas de mayor tamaño (hasta 5MW) y los movimientos producidos en la plataforma que sostiene el aerogenerador debido a la influencia del medio marino [1].

Aerodyn es un programa de cálculo aerodinámico utilizado para obtener la respuesta dinámica de aerogeneradores frente a series temporales de velocidad de viento. Se base en simulaciones aero-elásticas de aerogenradores de eje horizontal mediante las que se calculan las fuerzas soportadas por las palas debidas al viento incidente, utilizando las velocidades y posiciones de los elementos del aerognerador dadas por las rutinas de análisis dinámico, y el viento incidente en el sistema en forma de series temporales. Aunque Aerodyn es un programa independiente, también puede utilzarse (y se utiliza) integrado en FAST con igual finalidad [1].

El programa FAST completo es realmente complejo y en él existen varias áreas de modelado de la turbina, códigos auxiliares y generadores de archivos de entrada. En la **¡Error! No se encuentra el origen de la referencia.** se puede ver cómo interactúan entre ellos. Este esquema representa el flujo de información entre la mayoría de los códigos de diseño de aerogeneradores que integran FAST con objeto de calcular la potencia de salida, las cargas de diseño y el comportamiento del sistema de control. Este trabajo se centrará en la parte izquierda del esquema, no utilizando ni la simulación de ADAMS ni la linealización para conseguir matrices de estado periódicas [1].

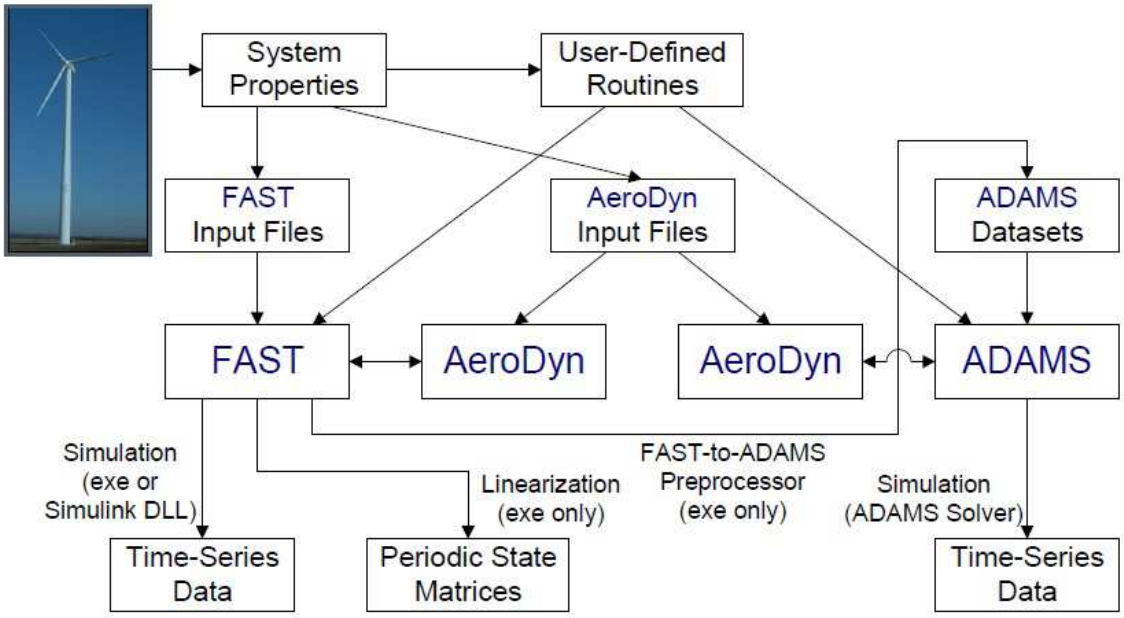


**Figura 2**. *Esquema de códigos usados por FAST.*

FAST tiene dos formas de trabajar en los análisis.

- El Modo Analógico usa un archivo primario de entrada para el control de la simulación, trabajando en función del tiempo de funcionamiento de las ecuaciones no lineales de movimiento. Durante la simulación se determinan en el tiempo la potencia de la turbina y la respuesta estructural a las condiciones de viento en la entrada permitiéndose la implementación de controles activos durante la simulación para determinar numerosos aspectos de la operación. Las salidas obtenidas de la simulación incluyen series temporales de datos de las variables comentadas anteriormente.
- La segunda forma de análisis que utiliza FAST es la linealización, permitiendo obtener representaciones linealizadas del sistema aero-elástico completo (no lineal). Este método de trabajo es útil para desarrollar matrices de estado en aerogeneradores y añadir controles de diseño en el análisis del aerogenerador. También resulta adecuado para determinar todos los modos del sistema de una turbina HAWT tanto en modo transitorio como estacionario a través de un "sencillo" análisis de valores propios. [2].

FAST presenta una ventaja fundamental frente a herramientas similares disponibles en el mercado, es de uso gratuito. Sin embargo, posee un inconveniente imporante que es la carencia de una interfaz gráfica que permita un uso sencillo y eficiente a nivel de usuario, permitiendo que accedan al código todos los ingenieros y diseñadores que lo deseen. Este problema supone una barrera de entrada al software impidiendo que se aproveche al máximo su principal ventaja, la gratuidad.

Trabajando en modo analógico FAST permite dos posibilidades de uso, la primera es a través de un ejecutable en MS-DOS, llamando el programa a los distintos códigos y archivos de entrada, que deberán estar generados, estructurados y completados adecuadamente para

evitar errores. La segunda opción consiste en usar una pseudointerfaz en el entorno de trabajo SIMULINK (de MATLAB), activando controles que pueden ser implementados mediante diagramas de bloques. No obstante los archivos de entrada tipo texto con extensión ".txt", las variaciones en los mismos y los archivos de salida generados son iguales a los del ejecutable en MS-DOS, es decir, no es sencillo trabajar con ellos. En este proyecto utilizaremos SIMULINK por ser un poco más sencillo y, fundamentalmente, por las posibilidades que ofrece MATLAB a la hora de realizar bucles y operaciones más complejas que servirán para completar los pasos necesarios en la ejecución del software.

## 2.2. Metodología de trabajo.

FAST, a través de SIMULINK, requiere numerosos archivos de entrada, muchos de ellos generados por códigos de apoyo. En este proyecto no se va a desarrollar el modelo de un aerogenerador desde cero puesto que excede y mucho el alcance de este trabajo. Sería necesario un conocimiento muy profundo de todo el código (y todos los subcódigos interconectados), programar en FORTRAN y escribir desde cero gran parte de los archivos de entrada. Por tanto se ha utilizado uno de los modelos de aerogeneradores "tipo" que contiene el propio programa y se ha modificado según las necesidades del trabajo. Así se aprovecha que estos modelos de aerogeneradores están certificados y no van a dar errores al simular con ellos.

Los datos y programas con los que trabaja FAST en esta metodología de trabajo se presentan en un esquema en la **Figura 3**.

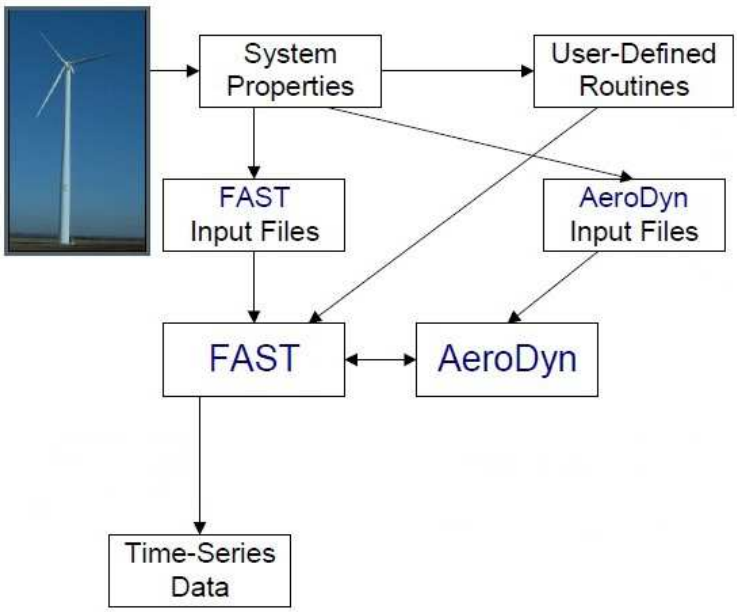


**Figura 3**. *Metodología de operación en el trabajo.*

- El modelo de aerogenerador seleccionado y el emplazamiento del mismo fija la mayor parte de las propiedades del sistema. En este proyecto no se parte de un modelo comercial concreto, por las razones presentadas anteriormente y además porque es difícil acceder a todas sus especificaciones técnicas, sino del modelo de aerogenerador de 1.5MW de potencia certificado de la librería de FAST.

- Parte de propiedades del sistema también se diseñan en función del tipo de estudio buscado. Por ejemplo las condiciones iniciales variaran dependiendo de si se quiere analizar como arranca el aerogenerador, como se para, o su funcionamiento estacionario.
- El usuario puede definir ciertas rutinas. Optimizando los resultados de FAST gracias a la inclusión de controladores externos, como un controlador de orientación de la góndola frente al viento incidente. O mejorando las presentes, como realizar pequeños cambios en el controlador del paso de las palas o el freno en el rotor.
- Los archivos de entrada de FAST son todos los necesarios para su ejecución, varían en función del modelo seleccionado (o desarrollado) y deben ser modificados según las necesidades del estudio. Algunos de ellos son el archivo principal de entrada, donde están presentes las propiedades del sistema, o el que controla el ángulo de paso de las palas.
- Los archivos de entrada de AERODYN pueden venir fijados por las series de velocidad de viento del emplazamiento seleccionado o ser generados por otros programas como TurbSim o IECWind en función de los requerimientos del estudio. Además hay otros archivos de las bibliotecas del propio software necesarios para la ejecución de AERODYN.
- Con toda esa información el software de FAST y de AERODYN trabajan conjuntamente para obtener las tablas de series de datos temporales buscadas. Las variables que se requieran habrán de ser fijadas en el archivo principal de entrada.

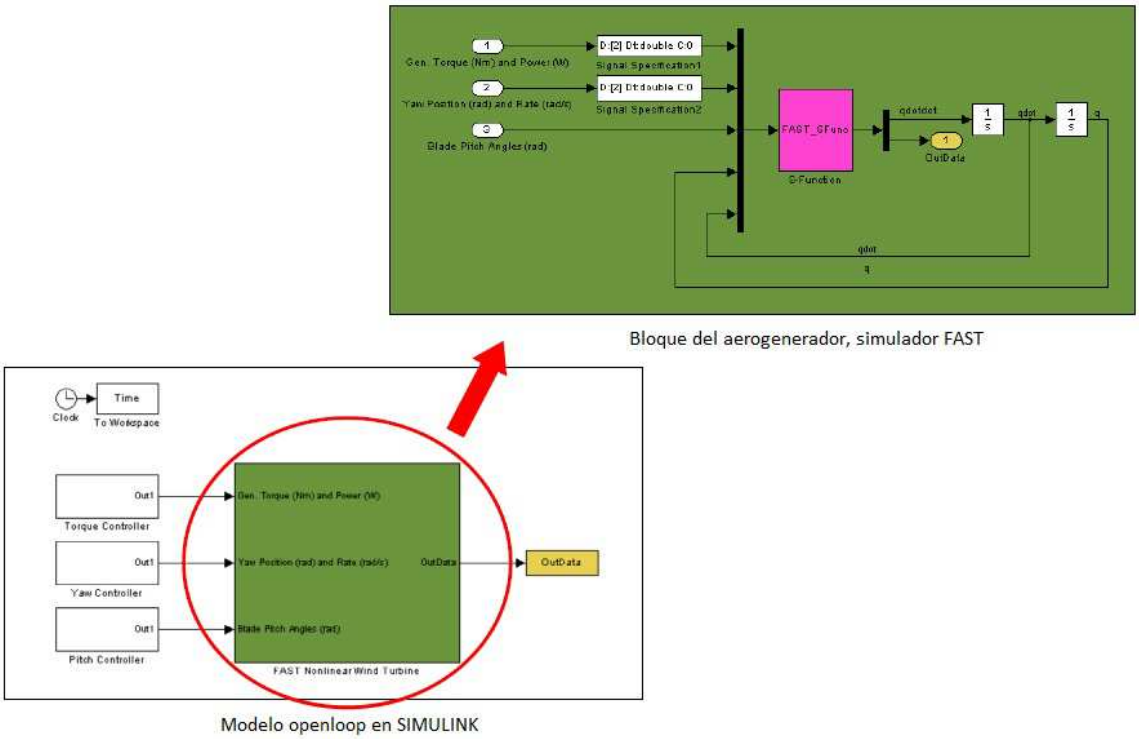


**Figura 4**. *Esquema trabajo FAST en Simulink.*

En la **¡Error! No se encuentra el origen de la referencia.** se puede ver el esquema de trabajo de Simulink. Este programa ofrece una "interfaz" que permite incorporar los archivos de entrada de una manera más sencilla. En el bloque general actúan los controladores de paso

de las palas, de giro del rotor y de orientación de la góndola y se obtiene la salida deseada en forma de tabla de series temporales de datos. Si se entra en detalle en el bloque del aerogenerador se introducen las características del sistema, la situación inicial y las especificaciones, además de dos etapas de retroalimentación que ajustan el bucle de trabajo. La función FAST_SFunc tiene una labor fundamental puesto que se encarga de todos los cálculos relativos al modelo. En el Anexo A se encuentra un desarrollo detallado "paso a paso" sobre el uso del programa.

***Tabla 1**. Ejemplo de evolución temporal de algunas de las variables de salida*

These predictions were generated by FAST (v7.02.00d-bjj, 20-Feb-2013)-Compiled as S-Function for Simulink on 20-Oct-2015 at 13:18:54.
The aerodynamic calculations were made by AeroDyn (v13.00.02a-bjj, 20-Feb-2013).

FAST certification Test #12: WindPACT 1.5 MW Baseline with many DOFs with VS and VP and ECD wind.

| Time (s) | HorWindV (m/s) | HorWndDir (deg) | OoPDefl3 (m) | IPDefl3 (m) | TipALxb3 (m/s^2) | TipALyb3 (m/s^2) |
|---|---|---|---|---|---|---|
| 5.043 | 1.180E+01 | 0.000E+00 | 1.467E+00 | -2.350E-01 | 6.501E+01 | -6.010E+00 |
| 5.115 | 1.180E+01 | 0.000E+00 | 1.668E+00 | -2.787E-01 | 1.885E+01 | 9.644E+00 |
| 5.188 | 1.180E+01 | 0.000E+00 | 1.895E+00 | -2.575E-01 | 2.182E+01 | 3.256E+00 |
| 5.263 | 1.180E+01 | 0.000E+00 | 2.130E+00 | -2.100E-01 | 1.871E+01 | -5.869E+00 |
| 5.335 | 1.180E+01 | 0.000E+00 | 2.319E+00 | -2.147E-01 | 3.743E-01 | -6.042E-01 |
| 5.410 | 1.180E+01 | 0.000E+00 | 2.375E+00 | -2.206E-01 | 7.785E+00 | 2.979E+00 |
| 5.483 | 1.180E+01 | 0.000E+00 | 2.324E+00 | -1.826E-01 | 2.771E+01 | -2.501E+00 |
| 5.558 | 1.180E+01 | 0.000E+00 | 2.262E+00 | -1.428E-01 | 3.093E+01 | -5.003E+00 |
| 5.630 | 1.180E+01 | 0.000E+00 | 2.214E+00 | -1.308E-01 | 2.749E+01 | -1.398E+00 |
| 5.703 | 1.180E+01 | 0.000E+00 | 2.167E+00 | -1.181E-01 | 3.162E+01 | -2.753E-01 |
| 5.778 | 1.180E+01 | 0.000E+00 | 2.140E+00 | -9.572E-02 | 3.448E+01 | -2.311E+00 |
| 5.850 | 1.180E+01 | 0.000E+00 | 2.151E+00 | -8.740E-02 | 2.997E+01 | -1.665E+00 |
| 5.925 | 1.180E+01 | 0.000E+00 | 2.180E+00 | -9.175E-02 | 2.669E+01 | 2.607E-01 |
| 5.998 | 1.180E+01 | 0.000E+00 | 2.211E+00 | -9.095E-02 | 2.752E+01 | -8.017E-01 |
| 6.068 | 1.180E+01 | -1.013E-02 | 2.246E+00 | -9.012E-02 | 2.746E+01 | -2.634E+00 |
| 6.140 | 1.181E+01 | -3.300E-02 | 2.290E+00 | -1.035E-01 | 2.585E+01 | -2.176E+00 |
| 6.215 | 1.182E+01 | -7.125E-02 | 2.334E+00 | -1.278E-01 | 2.546E+01 | -9.322E-01 |
| 6.288 | 1.183E+01 | -1.256E-01 | 2.373E+00 | -1.497E-01 | 2.637E+01 | -1.355E+00 |
| 6.360 | 1.185E+01 | -1.986E-01 | 2.414E+00 | -1.721E-01 | 2.709E+01 | -2.159E+00 |
| 6.435 | 1.187E+01 | -2.883E-01 | 2.460E+00 | -2.016E-01 | 2.721E+01 | -1.990E+00 |
| 6.508 | 1.190E+01 | -3.883E-01 | 2.505E+00 | -2.333E-01 | 2.725E+01 | -1.868E+00 |
| 6.583 | 1.193E+01 | -5.113E-01 | 2.546E+00 | -2.650E-01 | 2.782E+01 | -2.424E+00 |
| 6.655 | 1.196E+01 | -6.473E-01 | 2.579E+00 | -2.964E-01 | 2.912E+01 | -2.723E+00 |
| 6.728 | 1.200E+01 | -7.963E-01 | 2.603E+00 | -3.291E-01 | 3.017E+01 | -2.484E+00 |
| 6.803 | 1.204E+01 | -9.643E-01 | 2.620E+00 | -3.611E-01 | 3.078E+01 | -2.372E+00 |
| 6.875 | 1.208E+01 | -1.148E+00 | 2.625E+00 | -3.877E-01 | 3.128E+01 | -2.492E+00 |
| 6.950 | 1.213E+01 | -1.352E+00 | 2.614E+00 | -4.103E-01 | 3.241E+01 | -2.523E+00 |
| 7.023 | 1.218E+01 | -1.563E+00 | 2.591E+00 | -4.265E-01 | 3.361E+01 | -2.589E+00 |
| 7.098 | 1.224E+01 | -1.796E+00 | 2.560E+00 | -4.411E-01 | 3.260E+01 | -2.233E+00 |
| 7.170 | 1.230E+01 | -2.041E+00 | 2.521E+00 | -4.514E-01 | 3.203E+01 | -2.457E+00 |
| 7.243 | 1.237E+01 | -2.298E+00 | 2.467E+00 | -4.578E-01 | 3.227E+01 | -3.037E+00 |
| 7.318 | 1.243E+01 | -2.578E+00 | 2.400E+00 | -4.589E-01 | 3.206E+01 | -2.907E+00 |
| 7.390 | 1.250E+01 | -2.865E+00 | 2.329E+00 | -4.555E-01 | 3.001E+01 | -1.972E+00 |
| 7.465 | 1.258E+01 | -3.178E+00 | 2.256E+00 | -4.445E-01 | 2.985E+01 | -1.430E+00 |
| 7.538 | 1.266E+01 | -3.493E+00 | 2.191E+00 | -4.258E-01 | 3.121E+01 | -1.998E+00 |
| 7.610 | 1.274E+01 | -3.822E+00 | 2.136E+00 | -4.082E-01 | 3.027E+01 | -1.765E+00 |
| 7.685 | 1.283E+01 | -4.178E+00 | 2.089E+00 | -3.870E-01 | 2.880E+01 | -1.734E+00 |
| 7.758 | 1.292E+01 | -4.537E+00 | 2.051E+00 | -3.639E-01 | 2.639E+01 | -8.784E-01 |
| 7.830 | 1.301E+01 | -4.907E+00 | 2.018E+00 | -3.370E-01 | 2.712E+01 | -1.080E+00 |
| 7.900 | 1.310E+01 | -5.276E+00 | 1.989E+00 | -3.144E-01 | 2.685E+01 | -8.753E-01 |

Una vez obtenidas las tablas de resultados, como el ejemplo de la **Tabla 1,** se procede a realizar un gráfico que representa la potencia generada frente a velocidad del viento, la llamada curva de potencia o curva P(v). No obstante existe un problema, dado que el sistema simulado posee una inercia considerable a los cambios rápidos en la velocidad o dirección del viento incidente, estos no se ven reflejados instantáneamente en la potencia, sino que la potencia generada en cada instante dependerá, no sólo de la velocidad actual de viento sino también de la situación anterior del aerogenerador. Es decir, si el viento para bruscamente las palas seguirán girando a la misma velocidad el primer instante e irán desacelerando lentamente ya que la inercia de giro de la parte móvil (rotor, palas…) es muy grande. Por tanto la gráfica P(v) obtenida no será determinista, es decir, para la misma velocidad de viento se generan múltiples valores de potencia (una para cada distinto estado previo del sistema). Gráficamente esto se traduce en una nube de puntos en torno a una curva de potencia teórica exacta. Para obtener una curva que represente estas variaciones rápidas del viento de la forma

más precisa posible es necesario promediar los resultados o bien realizar el cálculo de la denominada curva de potencia dinámica (Capítulo 4).

# 3. OBTENCIÓN DE SERIES TEMPORALES

## 3.1. Variables de salida seleccionadas

La mayor parte de los parámetros de configuración están reflejados en el archivo principal de entrada (Anexo B), por lo que es importante centrarnos en él. Como tiene múltiples funciones se presenta cada uno de sus apartados junto con algunos ejemplos de las posibilidades de configuración más relevantes dentro del mismo:

- Controlar la simulación, todo lo relacionado con la configuración de los parámetros de la simulación: el tiempo de trabajo, el tiempo de cada uno de los pasos de integración.
- Controlar el aerogenerador, los sistemas internos y rutinas externas (en caso de existir) que controlan físicamente el aerogenerador: orientación de la góndola, posiciones iniciales de las palas del aerogenerador, tipo de frenos en eje de alta velocidad…
- Condiciones medioambientales: La presión (y por tanto la densidad del aire), depende de la altura a la que este situada la instalación, los resultados son distintos en una instalación a nivel del mar que en las crestas de una zona montañosa.
- Activación y desactivación de ciertos parámetros relacionados con el sistema, como ruidos, flexibilidad del eje de transmisión.
- Condiciones iniciales, variarán dependiendo del tipo de estudio a desarrollar: ángulo de la góndola, velocidad de rotación del eje lento.
- Configuración de la turbina, las características técnicas del aerogenerador seleccionado: altura de la torre, distancia desde el vértice del rotor a la raíz de la pala, distancia desde el eje del rotor a la punta del pala (radio de las palas).
- Masa e inercia: peso de la góndola, peso del eje.
- Transmisión: rendimiento de la caja de cambios, rendimiento del generador, torsión del resorte de transmisión.
- Generador de inducción simple o generador de inducción con circuito equivalente-Thevenin: se puede seleccionar que modelo de generador de los dos se quiere usar.
- Plataforma, en offshore llama al archivo que caracteriza la plataforma, al utilizar un emplazamiento terrestre en este proyecto no se utiliza.
- Torre, llama al archivo que contiene las propiedades de la torre
- Balanceo del rotor y freno de ayuda
- Pala, llama al archivo que caracteriza cada una de las palas.
- Aerodyn, llama al archivo que contiene los parámetros de entrada de AERODYN
- Ruido, llama al archivo con los parámetros del ruido
- ADAMS, llama al archivo que contiene los parámetros de entrada de ADAMS
- Control de linealización, llama al archivo que contiene los parámetros necesarios para la linealización en FAST.
- Salidas: tiempo de simulación a partir del cual empieza a escribir la tabla con los datos de salida, tiempo de los pasos de la tabla de salida respecto a los pasos de integración.

También permite la selección de cuáles de todas las variables que calcula el programa se quieren visualizar en las tablas de salida. (Anexo C)

Puesto que el proyecto se va a centrar en la curva de potencia, las únicas variables de salida necesarias serán la potencia y la velocidad del viento horizontal y perpendicular al área barrida por las palas. Además el tiempo de simulación siempre estará presente por tratarse de series temporales. Hay otras variables con interés como la velocidad del eje lento, la potencia antes de las pérdidas en la caja de cambios y en el generador o la dirección del viento que no necesitan ser presentadas pero que han sido útiles para los primeros cálculos, pruebas y comprobaciones.

## 3.2. Modelo de aerogenerador

Como se ha explicado anteriormente, desarrollar el modelo del aerogenerador excede el alcance de un proyecto de este tipo por lo que se ha elegido uno de los aerogeneradores certificados que incluye el programa. Se dispone de un total de 17 test en los que se combinan diversos tipos de aerogeneradores con distintos tipos de configuración de régimen viento, numero de palas y condiciones iniciales. Hay 6 modelos de aerogeneradores: 5MW, 1.5MW, 175kW, 50kW, 20kW, 10kW. También se utilizan distintas posibilidades de régimen de viento: estable, turbulento, con rachas, en el que se produzcan diversos eventos…; y de condiciones iniciales: modelo flexible, rígido, control de velocidad variable, paso fijo… (Para más detalles, se puede consultar el Anexo D)

El modelo seleccionado es un aerogenerador de paso variable de 1.5 MW, 3 palas, 70 metros de diámetro de rotor y situado en tierra. La altura de la torre son 82.39 metros, la altura de buje es 84 metros y su potencia nominal se alcanza cuando la velocidad del viento es aproximadamente 11.2 m/s [3]. Las razones de la elección del aerogenerador de 1.5MW son las siguientes. Una potencia de 1.5MW es comparable a la de los aerogeneradores que se están instalando actualmente para explotación comercial (actualmente entre 2 y 2.5 MW). Los modelos de menor potencia son demasiado pequeños puesto que la horquilla de potencias que abarcan va desde 10kW hasta 175kW, quedando descartados para el estudio realista de un parque eólico actual. El único aerogenerador más grande entre los modelos certificados tiene una potencia de 5MW. Este modelo es de reciente incorporación al programa y NREL aún está haciendo algunas pruebas con él, además hay que tener en cuenta que un aerogenerador de este tamaño principalmente está pensado para hacer ser instalado en emplazamientos offshore. Por último, el modelo de potencia curva dinámica que se obtiene en el capítulo 4 para verificar los resultados, está basado en un artículo científico en el que se utiliza este mismo modelo de aerogenerador.[4]

## 3.3. Datos de entrada

El archivo principal de entrada se ha ajustado para el objetivo principal del proyecto que, como se ha mencionado previamente, consiste en obtener una serie de datos P(v) a partir tanto de una serie experimental como de una serie sintética de velocidad de viento. Los parámetros más importantes son los siguientes. El tiempo de simulación se ha situado en 9999 segundos (aproximadamente 2.7 horas). Este es el tiempo máximo puesto que el código del programa al ser diseñado y compilado en FORTRAN no ha previsto mayor espacio en las variables para este concepto. Sin embargo, este tiempo máximo resulta insuficiente para, por

ejemplo, realizar los cálculos necesarios para la elaboración de la curva de potencia dinámica. En este caso, se han simulado más de 210 horas de funcionamiento del aerogenerador. Para ello, se ha creado una secuencia lógica en MatLab que hace trabajar en bucle a FAST las veces necesarias hasta cubrir el tiempo de trabajo. El tiempo de paso de integración se mantiene con los valores por defecto puesto que garantiza unos resultados precisos sin alargar en exceso el tiempo de simulación. Las 210 horas de simulación necesitaron aproximadamente 74 horas reales de cálculo con un ordenador portátil con procesador Intel Core i5 M450 2.40GHz, 4GB de memoria RAM, sistema operativo Windows 7 de 64 bits.

Cuando el aerogenerador trabaja a potencia nominal su eje lento gira a 20 revoluciones por minuto. No es necesario cambiar la velocidad inicial del eje lento para cada caso puesto que se deja un margen de tiempo para la estabilización del sistema y evitar los errores debidos al arranque. No obstante, si la velocidad inicial seleccionada es similar a la que tendría en estado estacionario a la velocidad dominante del viento incidente (contando con que no hubiera turbulencias), siempre es mejor. Como las distintas series simuladas comprenden velocidades de viento entre 5 m/s y 15 m/s, se ha elegido una velocidad de viento intermedia (10m/s) que corresponde a una velocidad inicial del eje lento del aerogenerador de 18 rpm. Esta es la velocidad con la que arrancará en cada una de las series que sean calculadas. El resto de configuraciones de la torre, turbina, la masa e inercia, transmisión, generador, góndola, frenos, ruidos y palas se han mantenido igual dado que es el mismo aerogenerador.

El archivo que contiene los parámetros de entrada de AERODYN no ha sido modificado, pero si el archivo de régimen de viento al que llama. Primero se generaron regímenes de viento sencillos con el programa IEC WIND (también desarrollado por NREL). Este programa permite generar regímenes de viento con velocidad constante y hacer que se produzcan diversos eventos (vientos cortantes, ráfagas…) en ellos durante un tiempo determinado. Después se utilizaron regímenes de viento constantes para estudiar la respuesta y posibilidades del programa, puesto que es mucho más fácil hacer comparaciones entre una simulación y otra cuando no hay ningún factor de aleatoriedad en los datos de entrada.

A la hora de realizar las simulaciones largas se han usado series de velocidad de viento reales, medidas a una frecuencia de 10 Hz. Estas series han sido proporcionadas por el área AIRE (Análisis Integral de Recursos Energéticos) de CIRCE y se extraen de una base de datos en formato ".csv" por lo que han debido de tratarse adecuadamente para poder ser utilizadas por FAST. Por último, como se verá en el capítulo 4, se ha utilizado el programa TURBSIM (NREL) para generar y controlar los regímenes de viento turbulentos necesarios para el cálculo de la curva de potencia dinámica.

Como se ha comentado previamente, ha sido necesario crear un sistema de iteración con MATLAB para solventar la limitación del tiempo máximo de simulación y, posteriormente, solapar una simulación con otra automáticamente y así unificar datos, para poder construir las series temporales y sus gráficas correspondientes. Como el modelo utilizado por FAST tiene una inercia tanto en el arranque como en la parada, es necesario realizar un solapamiento de las series de 9999 segundos durante un tiempo por determinar. Para conocer esta tiempo, se han realizado varias pruebas que han consistido en hacer arrancar el aerogenerador (velocidad eje palas 0 rpm) a distintas velocidades de viento (pero constantes) y comprobar cuánto

tiempo tardaba en estabilizarse y llegar al estado estacionario. Las velocidades de viento elegidas están entre 5 y 20 m/s siendo 20m/s la que tiene el tiempo más rápido de estabilización, 40 segundos, mientras que arrancando con una velocidad de 5m/s el sistema es considerablemente más lento y necesita 180 segundos para estabilizarse completamente. Esto es debido a que con una velocidad de viento de 5m/s es mucho más costoso y lento empezar a mover toda la masa giratoria del aerogenerador, y pese a que la velocidad estacionaria a alcanzar es mucho menor que con una velocidad de viento de 20m/s, el tiempo necesario es mayor. Por tanto, se ha elegido un tiempo de solapamiento en las simulaciones igual a 199 segundos (no menor que el mayor de los tiempos obtenidos en las pruebas) de tal forma que por cada 9999 segundos que se simulan se aprovechan 9800.

No obstante hay que reseñar que es una elección muy conservadora dado que no es necesaria la estabilización a la centésima porque en la realidad habrá variaciones continuas de la velocidad del viento y de su dirección. Por ejemplo en el artículo "*Langevin power curve analysis for numerical wind energy converter models with new insights on high frequency power performance"* [44] no utilizan para los cálculos los datos de los 100 primeros segundos, se simulan 700 segundos para una serie de 10 minutos. La razón es que es que hay que alcanzar un compromiso entre precisión y tiempo, y que además existen múltiples factores como la dirección del viento que influyen en mayor medida en el resultado. Además ninguno de los casos reales es tan extremo como partir de una posición inicial de paro y arrancando con una velocidad de viento muy pequeña. Las simulaciones con series temporales de viento reales se hicieron con 199 segundos de solapamiento entre serie y serie, pero solo había 8 solapamientos y era razonable sobre-asegurar, pero los 100 segundos utilizados en el artículo son suficientes [4].

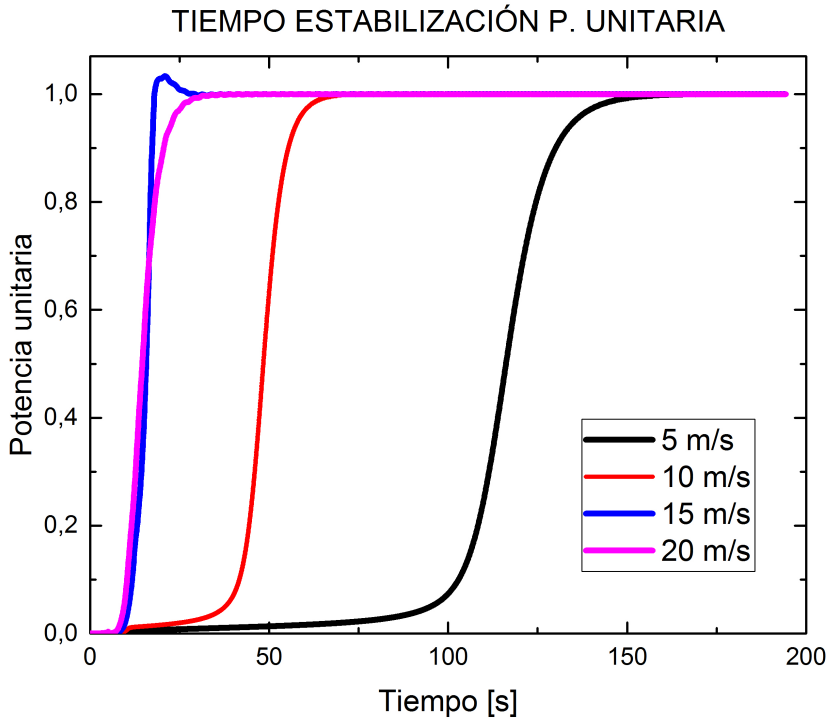


**Figura 5**. *Tiempo de estabilización según la velocidad del viento, potencias unitarias.*

En la **¡Error! No se encuentra el origen de la referencia.** se presentan los distintos tiempos de estabilización, respecto a la potencia máxima unitaria que el aerogenerador es capaz de generar, y que necesita el sistema al arrancar según las velocidades del viento incidente. Las cifras dadas anteriormente son mayores que las presentadas en el gráfico pero eso es debido a que son cifras conservadoras y a que en la figura no se llega a percibir el resultado con un error de un vatio. El tiempo de estabilización es menor cuanto mayor es la velocidad hasta 15 m/s, luego se mantiene constante. La razón es la explicada anteriormente, para empezar a mover toda la masa de las palas del aerogenerador, una velocidad de viento baja lo tiene mucho más difícil y necesita más tiempo para alcanzar el estado estacionario. El aerogenerador seleccionado para los test utiliza la tecnología de paso variable para regular la potencia generada. En los aerogeneradores de paso fijo la aerodinámica de las palas está diseñada para que cuando la velocidad nominal (11.2 m/s) sea sobrepasada el perfil aerodinámico de las palas genere turbulencias para controlar la potencia generada. Es decir, a partir de la velocidad nominal las palas no están diseñadas para obtener el máximo posible de energía del viento, sino para disipar gradualmente parte de la misma. No obstante en los aerogeneradores de paso variable el ángulo de de las palas cambia en función del viento incidente, a partir de la velocidad nominal del viento el ángulo cambia para desaprovechar gradualmente parte de la energía del viento incidente en pos de mantener una potencia generada controlada y constante. Esta puede ser una de las razones para generar extraños como en el caso de 15 m/s donde el sistema sigue un modelo de amortiguamiento de segundo orden (con dos polos), sobreoscilando antes de llegar a situación estacionaria. En las simulaciones con series temporales de viento reales ha ocurrido algo similar, cuando había un incremento importante por encima de la velocidad del viento con la que el aerogenerador

desarrolla una potencia nominal, se han generado breves sobreoscilaciones hasta que el control de paso variable de las palas ha devuelto al aerogenerador a su potencia nominal. Para asegurar que no se trata de un error de cálculo se ha realizado un barrido entre 10m/s y 20m/s para comprobar cómo evolucionan los tiempos de estabilización y si se produce también sobreoscilación en el proceso de estabilización del sistema. Los resultados se presentan en la **Figura 6**.

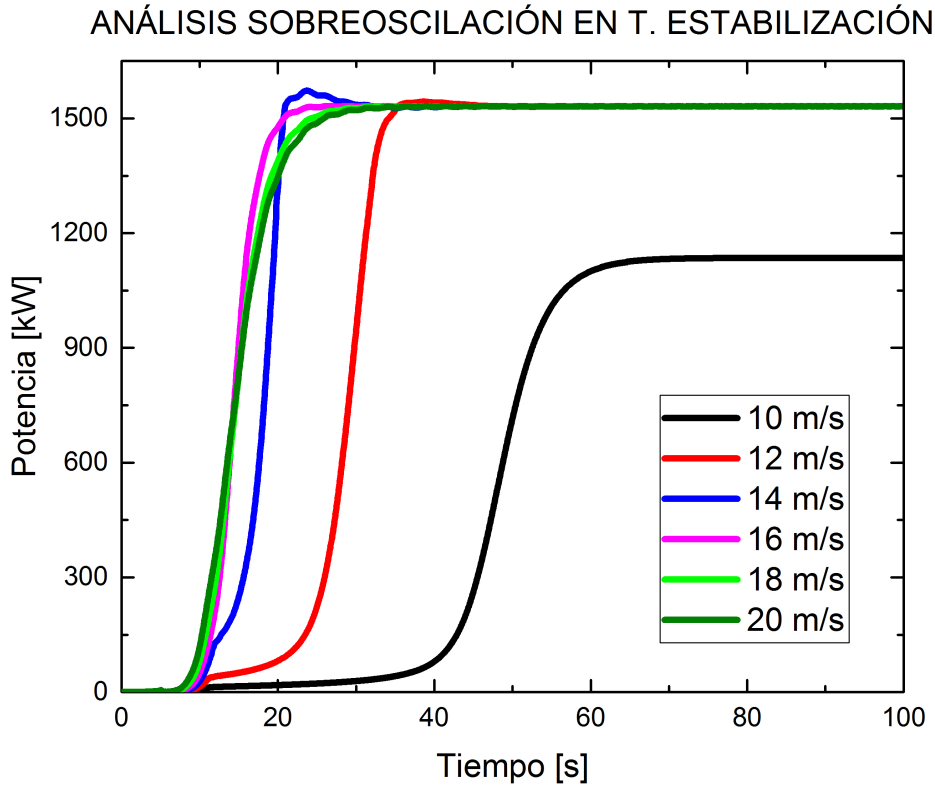


**Figura 6**. *Barrido entre 10m/s y 20m/s para Tiempo de estabilización.*

Como se puede comprobar en la **Figura 6** la sobreoscilación se produce en la estabilización del sistema para velocidades del viento entre 14m/s y 15m/s, pero los resultados respecto al tiempo requerido son coherentes. Entre 5m/s y 20m/s la diferencia de potencia generada es de un orden de magnitud, para apreciarlo más claramente se presenta en la **Figura 7**.

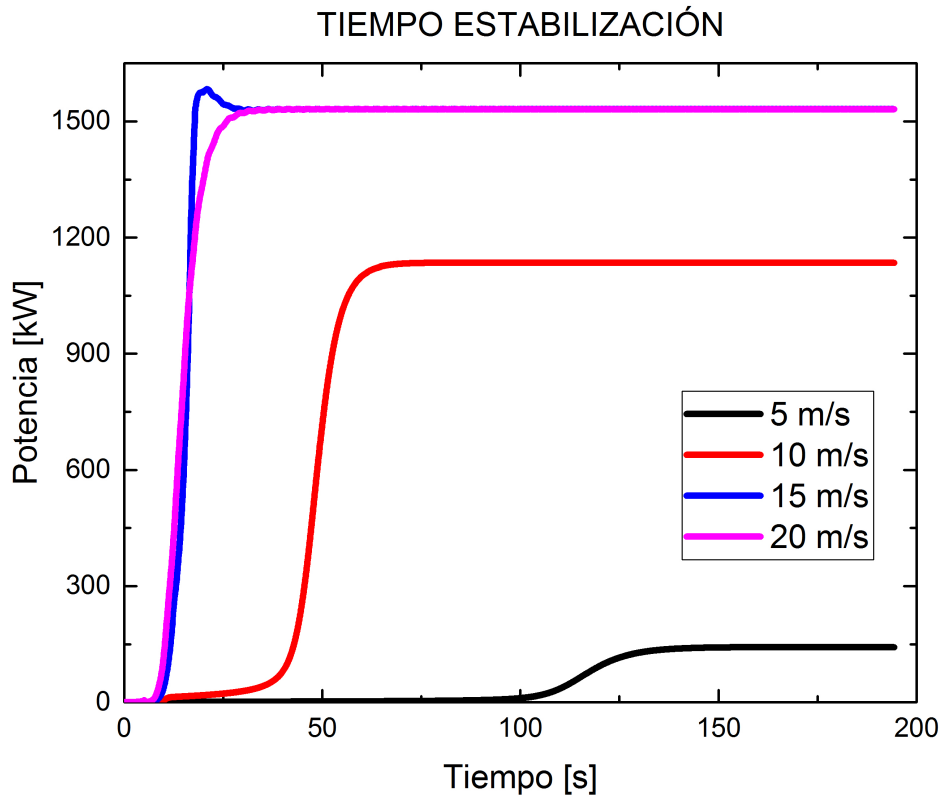


**Figura 7**. *Tiempo de estabilización según la velocidad del viento, potencias absolutas.*

## 3.4. Margen dentro de la curva de potencia

En la **¡Error! No se encuentra el origen de la referencia.** se puede observar el arranque y la parada del aerogenerador, y la gran influencia de la inercia del sistema. Ambas curvas representan la potencia generada por el aerogenerador en Kilovatios frente a la velocidad del viento en m/s. La curva de subida parte desde una situación de parada tanto del eje como del viento, que se va incrementando paulatinamente, si se le diera más tiempo llegaría a una situación estacionaria, por ejemplo antes se ha visto que con un viento de 5 m/s en 180 segundos el aerogenerador arrancaba y la situación se estabilizaba. Para evitar una simulación demasiado larga no se ha ido tan lento. Una vez el aerogenerador llega a una situación de potencia nominal se procede a reducir paulatinamente la velocidad del viento hasta llegar a pararlo, en este caso la propia inercia de las palas favorece sostener el ratio de giro haciendo que la curva de bajada quede considerablemente por encima de la primera. La **¡Error! No se encuentra el origen de la referencia. a**demás ayuda a visualizar de una manera sencilla las diferencias de potencia generada que se pueden producir para una misma velocidad del viento en función de la inercia del sistema rotatorio, es decir, para la misma velocidad del viento incidente existen potencias generadas muy dispares.

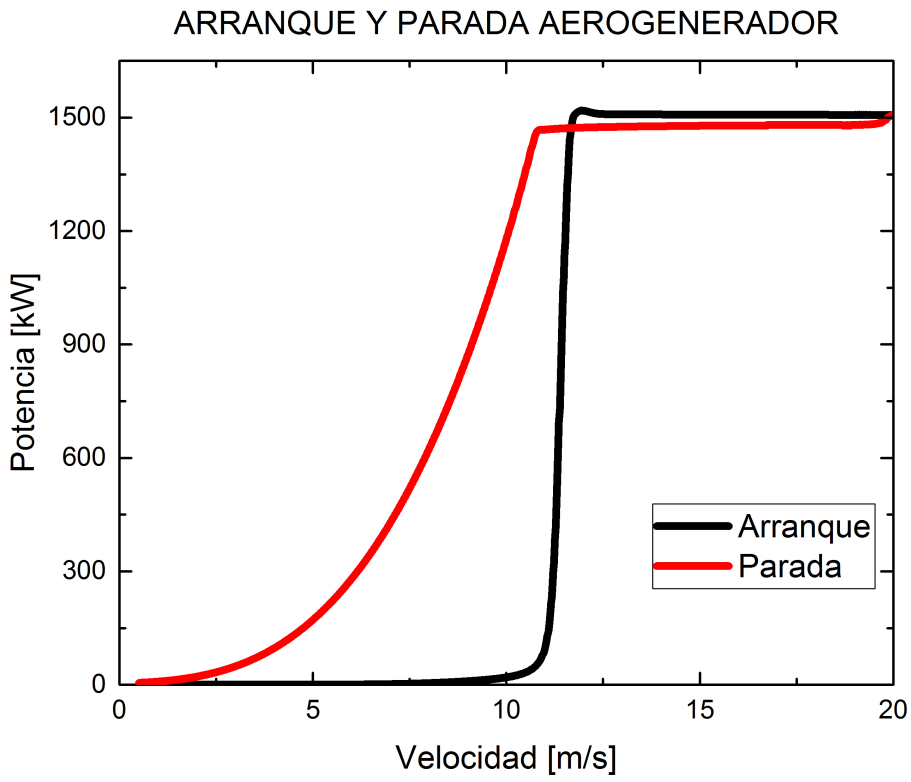


**Figura 8**. *Curva de potencia del aerogenerador, arrancando y parándose.*

## 3.5. Simulación con series de velocidad viento reales

En este apartado se han utilizado como datos de entrada de viento las series de velocidad de viento reales de la base de datos de AIRE mientras que el resto de parámetros se han mantenido constantes. El principal problema encontrado ha sido el manejo de volúmenes tan grandes de datos. El anemómetro mide y guarda la velocidad del viento cada 0,096 segundos, es decir en un día (86400 segundos) se generan 900.000 líneas de datos, y la base de datos contiene varios meses. No obstante una hoja Excel soporta aproximadamente un millón de líneas, y un bloc de notas muchas menos.

Las series de velocidad de viento se encuentran en una base de datos en formato “csv” y han tenido que ser tratados con el programa CSVed, para dividirlas en grupos de aproximadamente 900.000 líneas [5]. Estos conjuntos más pequeños a su vez han sido abiertos con MATLAB para adecuar su tamaño y formato a los archivos de viento con los que es capaz de trabajar FAST. Por diseño, el tiempo máximo de simulación de FAST son 9999 segundos por lo que para cada 900.000 líneas (24 horas de muestreo y cerca del límite de Excel [6]) es necesario realizar 9 simulaciones distintas.

Para conseguir estos resultados ha sido necesario desarrollar un programa en MATLAB que permita adecuar las series de velocidad de viento a los archivos de entrada de FAST y que éste sea ejecutado en bucle automáticamente (Anexo E). En total se han simulado 24 horas, es

decir 1 día. Teniendo en cuenta que la frecuencia de escritura en las series temporales de datos de salida (P y v) aproximadamente es de 10 Hz, se han obtenido 240*3600=864.000 binomios "potencia generada – velocidad de viento incidente" útiles. No obstante para conseguir el solapamiento entre las distintas series generadas se requieren 199 segundos más por serie 9*199*10=17.910, es decir, en total se generan 864.000+17.910=881.910 binomios P-V. La **Tabla 2** es una muestra de una pequeña parte de una de las muchas tablas de series temporales obtenidas.

***Tabla 2**. Ejemplo de tabla de salida*

These predictions were generated by FAST
The aerodynamic calculations were made

FAST certification Test #13: WindPACT 1.5

| Time (s) | HorWindV (m/s) | GenPwr (kW) |
|---|---|---|
| 0.100 | 1.047E+01 | 9.876E+02 |
| 0.195 | 1.047E+01 | 1.023E+03 |
| 0.285 | 1.047E+01 | 1.016E+03 |
| 0.375 | 1.047E+01 | 1.017E+03 |
| 0.463 | 1.029E+01 | 1.040E+03 |
| 0.550 | 1.025E+01 | 1.052E+03 |
| 0.637 | 1.039E+01 | 1.064E+03 |
| 0.725 | 1.047E+01 | 1.070E+03 |
| 0.810 | 1.037E+01 | 1.068E+03 |
| 0.897 | 1.025E+01 | 1.077E+03 |
| 0.985 | 1.025E+01 | 1.091E+03 |
| 1.090 | 1.025E+01 | 1.090E+03 |
| 1.205 | 1.037E+01 | 1.099E+03 |
| 1.317 | 1.047E+01 | 1.113E+03 |
| 1.427 | 1.047E+01 | 1.113E+03 |
| 1.542 | 1.047E+01 | 1.121E+03 |
| 1.657 | 1.041E+01 | 1.134E+03 |
| 1.768 | 1.025E+01 | 1.140E+03 |
| 1.880 | 1.025E+01 | 1.148E+03 |
| 1.992 | 1.025E+01 | 1.162E+03 |
| 2.105 | 1.025E+01 | 1.167E+03 |
| 2.220 | 1.047E+01 | 1.171E+03 |
| 2.335 | 1.040E+01 | 1.179E+03 |
| 2.450 | 1.036E+01 | 1.179E+03 |
| 2.565 | 1.017E+01 | 1.177E+03 |
| 2.678 | 1.023E+01 | 1.178E+03 |
| 2.795 | 1.028E+01 | 1.172E+03 |
| 2.908 | 1.047E+01 | 1.167E+03 |
| 3.025 | 1.036E+01 | 1.165E+03 |
| 3.138 | 1.025E+01 | 1.158E+03 |
| 3.255 | 1.025E+01 | 1.152E+03 |
| 3.368 | 1.025E+01 | 1.147E+03 |
| 3.485 | 1.032E+01 | 1.141E+03 |
| 3.598 | 1.046E+01 | 1.135E+03 |
| 3.712 | 1.047E+01 | 1.133E+03 |
| 3.828 | 1.047E+01 | 1.131E+03 |
| 3.942 | 1.069E+01 | 1.130E+03 |
| 4.060 | 1.069E+01 | 1.132E+03 |
| 4.185 | 1.069E+01 | 1.134E+03 |
| 4.310 | 1.069E+01 | 1.137E+03 |
| 4.433 | 1.043E+01 | 1.140E+03 |
| 4.558 | 1.035E+01 | 1.140E+03 |
| 4.683 | 1.030E+01 | 1.138E+03 |

## 3.6. Análisis de las series temporales

Una vez obtenidas las tablas de resultados, hay que unirlos de nuevo mediante MATLAB, para poder realizar las gráficas que faciliten (permitan) su visualización. Para ello se ha desarrollado otro programa en MATLAB, más sencillo que el anterior, que une y ordena en

un tabla Excel los archivos de generados por FAST (Anexo F). Tras tener todos los datos generados adecuadamente presentados y ordenados (hay que tener en cuenta la superposición de 199 segundos entre ellos, para evitar la influencia de la inercia) es posible proceder a su análisis.

Para poder realizar un análisis de tal cantidad de datos obtenidos es necesario representarlos en una gráfica potencia-velocidad, debido a que es imposible sacar ninguna conclusión solo con las tablas aisladamente, **Figura 9**. Como se ha explicado anteriormente en el Capítulo 2 la gráfica obtenida no es determinística, es decir, para la misma velocidad de viento se generan múltiples valores de potencia (una para cada distinto estado previo del sistema). El gráfico es una nube de puntos que permite suponer donde aproximadamente se sitúa el comportamiento del aerogenerador, pero que realmente no permite su uso para ningún cálculo teórico posterior puesto que tiene unos rangos de variación relativamente grandes. Se hace necesario de este modo necesario un post-procesamiento de los datos para llevar a cabo un análisis concreto. La manera de obtener resultados fiables del comportamiento del aerogenerador, teniendo en cuenta todas las posibilidades de trabajo del mismo es calcular la curva de potencia dinámica. No obstante para calcular de una manera más fiable la curva de potencia dinámica se han generado en el siguiente capítulo unas series temporales de velocidad de viento, siguiendo el método del artículo de la universidad de Oldemburg [4], cubriendo todo el espectro de trabajo del aerogenerador y presentando una cantidad suficiente de datos para asegurar un cálculo correcto de la misma.

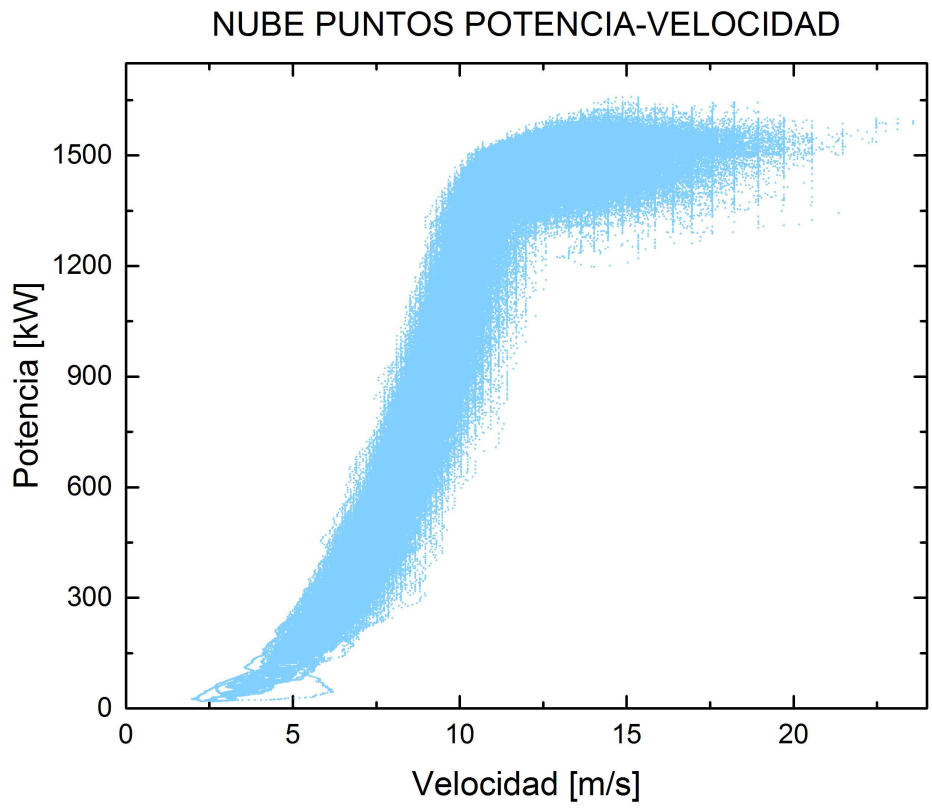


**Figura 9**. Nube de puntos potencia generada – velocidad de viento incidente

En la **Figura 9** se han representado los 864.000 binomios P-V. Para obtener una comparativa frente a las simulaciones realizadas en el Capítulo 3 se ha representado la **Figura 10** en la que se superpone a las mediciones de la **Figura 9** el comportamiento del aerogenerador definido por su curva de potencia dinámica ponderada cuyo cálculo se llevará a cabo en el capítulo 4. La **Figura 9** se presenta para tener una visión sencilla de cómo la nube de puntos P-V generados se encuentran en las proximidades de la CDP. También sirve para hacerse una idea sobre como la inercia de la parte móvil del aerogenerador influye en los resultados de la potencia instantánea obtenida en función de la velocidad del viento.

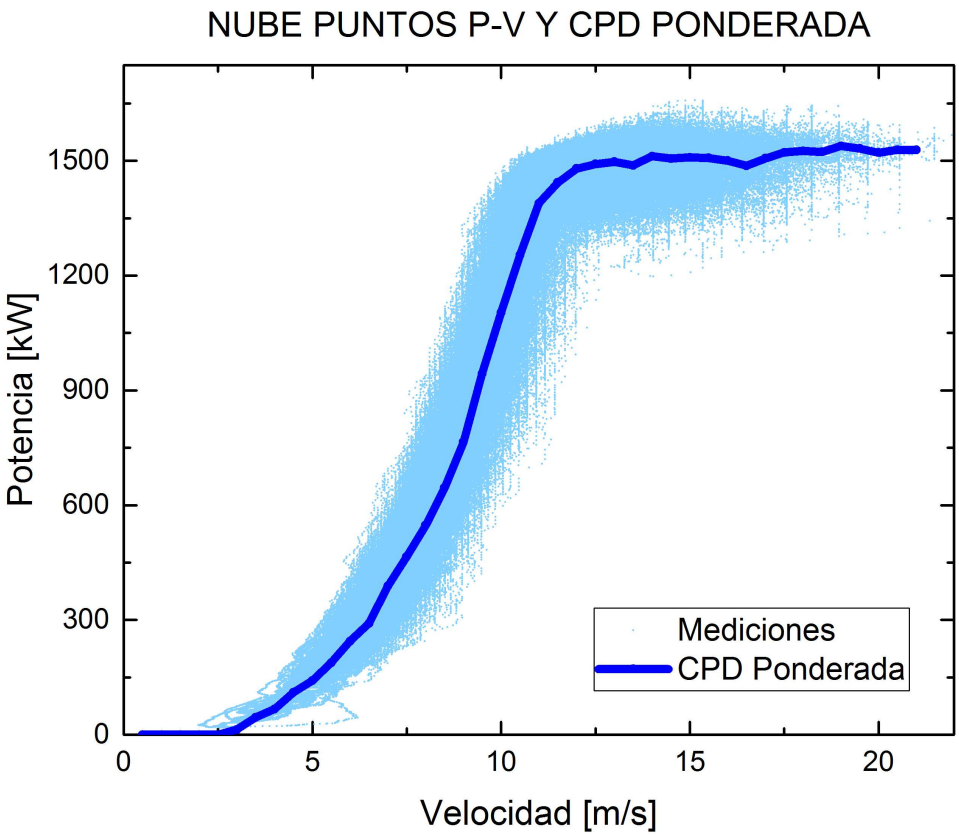


**Figura 10**. *Nube de puntos P-V y curva de parada.*

Esta metodología de trabajo permite la generación de datos anonimizados P-V para su uso en posteriores estudios, sin necesidad de depender de los datos de compañías privadas, con el problema de propiedad intelectual que supone. Lo cual era uno de los objetivos principales del proyecto.

# 4. CURVA DE POTENCIA DINÁMICA

## 4.1. Introducción

Debido al gran incremento de energía eólica generada y vertida a la red eléctrica, cada vez es más necesario un conocimiento preciso de cuál va a ser la cantidad de energía eólica que va a entrar en la red eléctrica. Para ello es necesario saber aproximadamente las velocidades de viento esperadas en cada zona y la relación entre dicha velocidad y la potencia generada por las turbinas, es decir la curva de potencia de los aerogeneradores. Dicho conocimiento es primordial para un diseño óptimo de una red de centrales eléctricas que satisfaga las necesidades de un país.

Generalmente la estimación de la producción anual de energía de los aerogeneradores se realiza mediante promedios de medidas estándar con promedios de 10 minutos según la normativa de IEC (International Electrotechnical Commission) IEC 61400-12-1 [7]. Estas mediciones se realizan obteniendo cada 10 minutos la media de la velocidad del viento a la altura del buje y la media de la potencia generada por el aerogenerador durante dicho periodo. La estimación de la producción anual de energía eólica en la red eléctrica llevada a cabo mediante medidas diezminutales es un método sencillo y ágil si se dispone de la enorme cantidad de datos necesarios para su realización, pero no representa con precisión la potencia característica generada respecto a la velocidad en varios aspectos:

- No representa adecuadamente la potencia generada cuando se producen fluctuaciones rápidas en la velocidad del viento o fuertes turbulencias. Es decir, sirve para obtener una media adecuada de la potencia generada pero no sirve para ver el comportamiento del aerogenerador cuando hay cambios bruscos en la velocidad o dirección del viento [8].
- El modelo tampoco es bueno para representar la dependencia entre el rendimiento del aerogenerador y la orografía del terreno [9]. El campo de viento turbulento está fuertemente influenciado por el tipo de terreno y la localización del anemómetro, haciendo que generalmente las estimaciones con el procedimiento de medida de promedio estándar muestren unos niveles de incertidumbre entre el 10% y el 20% para emplazamientos con orografía compleja [8].

Por todo esto se han desarrollado métodos de cálculo de curva de potencia que reflejen de una manera mucho más precisa el comportamiento de los aerogeneradores, las curvas dinámicas de potencia. Una de las mayores dificultades es debida a que no existe linealidad de la potencia frente a la velocidad: $P \propto u^3$. Por lo que se genera la siguiente desigualdad $P(\langle u \rangle) \neq \langle P(u) \rangle$, es decir la potencia de la media de velocidades no es igual a la media de las potencias para cada una de las velocidades [8].

Diversos autores han propuesto distintos métodos lineales o no-lineales adicionales para tener en cuenta las turbulencias, fluctuaciones del viento y no linealidad entre potencia y velocidad. Entre ellos Albers y Hinsck [10] propusieron el uso de las series de Taylor para la potencia media obtenida. También Keiser [11] y Van Radecke [12] trabajaron y avanzaron en el mismo sentido. No obstante a pesar de los progresos obtenidos sigue siendo necesaria una

mejora en pos de una curva de potencia que represente el comportamiento real de una forma más avanzada y detallada. Especialmente en aspectos como la capacidad de representación de cambios bruscos en la velocidad de viento, el error de estimación de las curvas de potencia dependiendo del lugar de instalación, o la enorme cantidad de datos necesarios para el cálculo según la norma IEC-61400-12 [8].

## 4.2. Procedimiento de cálculo

El objetivo del cálculo de la curva de potencia es doble, el primero es construir la curva de potencia dinámica del aerogenerador usado en el proyecto, para tener una idea clara sobre su comportamiento respecto al viento incidente. El segundo es la verificación del método de cálculo de la curva de potencia dinámica desarrollado por el CIRCE. Para ello se seguirá el procedimiento de obtención de los datos de entrada del artículo referencia, pero se utilizara el sistema de cálculo de CIRCE.

El método utilizado en este trabajo está basado en el artículo: "Markovian Power Curves for Wind Turbines" del "Center for Wind Energy Resarch, University of Oldemburg, Germany" [8]. Este artículo afronta el cálculo de la curva de potencia dinámica a partir de un nuevo método basado en el análisis estocástico de las series temporales. Es decir, la memoria que tiene el sistema frente a las fluctuaciones del viento incidente, que genera unas curvas potencia - velocidad no determinísticas, pasa a ser considerada como una variable sometida al azar permitiendo un tratamiento estadístico de la misma y simplificando los cálculos necesarios. El análisis estocástico permite una estimación de la deriva y difusión de los coeficientes de los sistemas complejos no lineales directamente a partir del conjunto de datos (P-v). Una explicación más sencilla, la potencia generada por el aerogenerador $[P(t)]$ se divide en una parte determinista que se puede calcular de manera convencional $[P_s(u)]$ siendo el estado estacionario de la potencia generada y en la parte estocástica (también llamada ruido) $[p(t)]$ que resume todas las interacciones microscópicas instantáneas y permite una descripción macroscópica del complejo sistema abierto que se trata. La ecuación resultante es la siguiente $P(t) = P_s(u) + p(t)$ [9].

Éste sistema de trabajo ya ha sido ampliamente utilizado en campos tan distintos como los flujos de tráfico, análisis de conducción de ruido, datos financieros, rugosidad superficial o circuitos eléctricos caóticos. El artículo desarrolla un procedimiento que caracteriza el rendimiento de la potencia del aerogenerador a partir de puntos fijos en dinámicas deterministas de la relajación de la potencia de salida [8].

La teoría del método matemático para el cálculo de la curva de potencia dinámica es bastante complejo y en él se aplican funciones de densidad de probabilidad, derivadas y cálculos estadísticos como la varianza y la desviación típica entre otros a una gran cantidad de datos lo que hace necesario el desarrollo de un programa de ordenador que lo ponga en práctica. Por ejemplo en este trabajo se han utilizado 7.560.000 binomios Potencia-velocidad, que significan 210 horas de simulación en FAST. Obviamente no es posible hacerlo a mano para obtener la curva de potencia a partir de tablas de series temporales P-v. En todo caso la teoría matemática se presenta más adelante para presentar como funciona el programa de cálculo.

Como se ha comentado antes, para cumplir el segundo objetivo es necesario generar una estructura equivalente de datos de entrada, velocidad y orientación del viento incidente, y el posterior cálculo del binomino P-V a través de FAST. De esta forma se puede realizar una comparación entre la curva dinámica de potencia del articulo "Markovian Power Curves for Wind Turbines" y la construida ex profeso para este proyecto. En otras palabras, da la oportunidad de realizar una verificación del método de cálculo de la curva de potencia dinámica de CIRCE.

El objetivo es reconstruir la ecuación de Langevin a partir de datos experimentales y numéricos. Se supone que la potencia de salida responde a un proceso de Markov dinámico, siendo una de sus propiedades que las dinámicas del proceso no tengan memoria, sólo depende del valor anterior y de los datos de entrada obviamente, pudiéndose describir el proceso mediante el desarrollo de Kramers-Moyal.

$$\frac{\partial}{\partial t}W(P,t|P',t) = \sum_{n=1}^{\infty}\frac{1}{n!}\left(-\frac{\partial}{\partial P}\right)^n D^{(n)}(P,t)W(P,t|P',t')$$

siendo $D^{(n)}$ los coeficientes Kramers-Moyal que se pueden definir con:

$$D^{(n)}(P) = \lim_{\tau\to 0}\frac{1}{\tau}M^{(n)}(P,\tau)$$

Y el momento condicional $M^{(n)}$ caracteriza las probabilidades de que la potencia P o t se vean condicionados o afectados.

$$M^{(n)}(P,\tau) = \langle\left(\tilde{P}(t+\tau) - P(t)\right)^n\rangle|_{P(t)=P}$$

Se realizan una serie de cálculos matemáticos y simplificaciones: considerando que el ruido gaussiano presente en el proceso de Markov se desvanece para $D^{(4)}$ y de acuerdo con el "Pawula's theorem" los coeficientes $D^{(4)} = 0, \forall n \geq 3$. Con esas condiciones se obtiene, a partir del desarrollo de Kramers-Moyal, la ecuación simplificada de "Fokker-Planck". Que no es otra que la ecuación generalizada de la ecuación estocástica de Langevin, definida como [8]:

$$\frac{d}{dt}P(t) = D^{(1)}(P) + \sqrt{D^{(2)}(P)}\cdot\Gamma(t)$$

$P(t)$ es la potencia generada por la turbina. Los términos $D^{(1)}(P)$ y $D^{(2)}(P)$ se definen como coeficientes de deriva y difusión, y describen la relajación determinística y la evolución temporal del ruido estocástico, respectivamente. El término $\sqrt{D^{(2)}(P)}$ define la amplitud del ruido dinámico. Y por último $\Gamma(t)$ ha sido determinado anteriormente y es el ruido blanco Gaussiano independiente con una media de cero asociado a la respuesta del sistema.

El primer paso es calcular los coeficientes de Kramers – Moyal para ello hay que calcular primero el momento condicional. Dicho momento condicional se obtiene de la simplificación de la ecuación del momento condicional presentada anteriormente, reformulándola como una serie de Taylor:

$$M^{(n)}(P,\tau) = \tau D^{(n)}(P) + O(\tau^2)$$

Para obtener los coeficientes de Kramers – Moyal hay que aplicar $\lim_{\tau\to 0} M^{(n)}(P,\tau)$. Una vez obtenidos los coeficientes se presentan en una gráfica M frente a $\tau$, siendo $\tau$ el tiempo pasado desde el instante inicial y el momento a evaluar, es decir el paso temporal utilizado en la toma de datos. Hay que hacer adimensionales $M^{(1)}$ y $D^{(1)}$ por lo que se sustituye $P(t)$ por $P(t)/P_r$ siendo $P_r$ la potencia nominal para esa velocidad. Como se puede observar en la **Figura 11** una vez creada la gráfica con la ayuda de la varianza de todos los datos que se tengan para esa velocidad, hay que buscar una zona lineal, dibujar la recta y prolongarla hasta $\tau = 0$ obteniéndose en el punto que corta con la escala de $M^{(n)}$ el valor del coeficiente de deriva $D^{(1)}$ de dicha grafica [8].

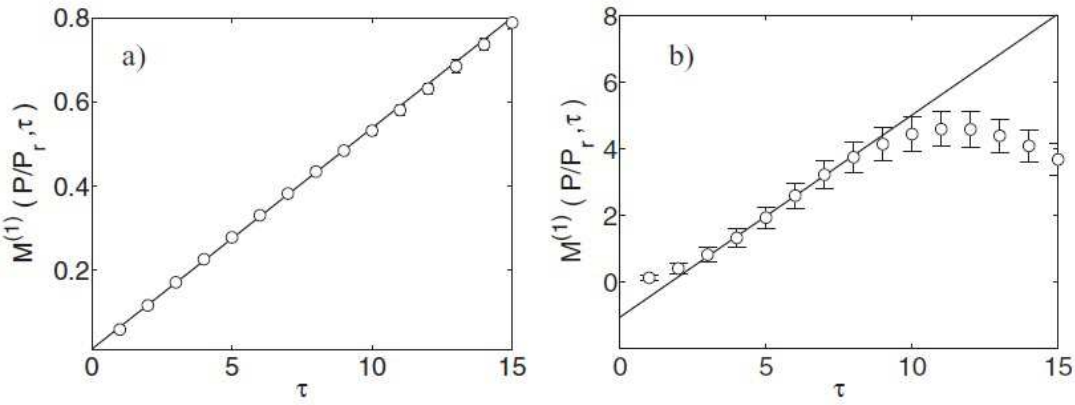


**Figura 11**. *Procedimiento de estimación de* $\boldsymbol{D^{(1)}}$ *a partir de* $\boldsymbol{M^{(1)}}$.

Se puede ver la diferencia entre la gráfica teórica, con lo resultados del modelo numérico y qué ocurre si se toman datos reales. En este caso los datos reales son las simulaciones desarrolladas a partir de los datos de viento generados. Hay que repetir el procedimiento para cada dato de potencia existente para esa velocidad, obteniéndose una gráfica $D^{(1)}\left(\frac{P}{P_r}\right)$ frente a $\left(\frac{P}{P_r}\right)$ como la **Figura 12** [8].

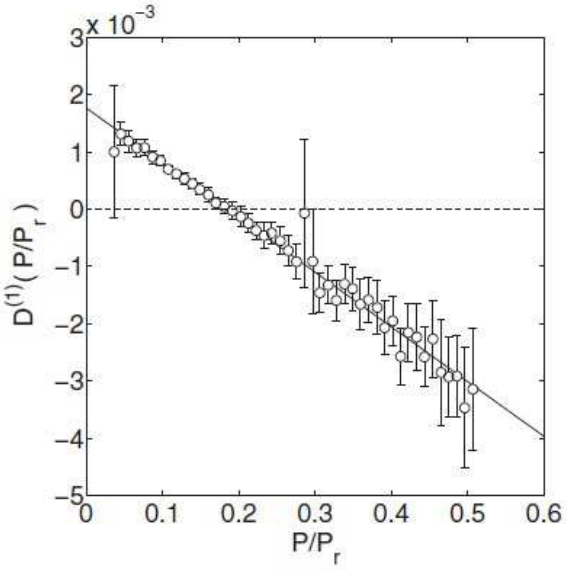


**Figura 12**. *Estimación de uno de los puntos fijos de la gráfica definitiva.*

En el punto en el que corte la aproximación lineal de la curva con cero, hay que recordar que se trata de valores adimensionales, se encontrará uno de los puntos fijos de la gráfica definitiva P-v. Para ello hay que tomar el valor $\left(\frac{P}{P_r}\right)$ obtenido en el corte con cero, que estará asociado a la velocidad de viento incidente seleccionada para realizar la gráfica. Hay que repetir este procedimiento para cada una de las velocidades objeto del estudio. En el caso de este proyecto el rango de velocidades va desde 5m/s hasta 15m/s con pasos de 0.5m/s [8]. Obteniéndose una curva dinámica de potencia equivalente a la de **Figura 13**.

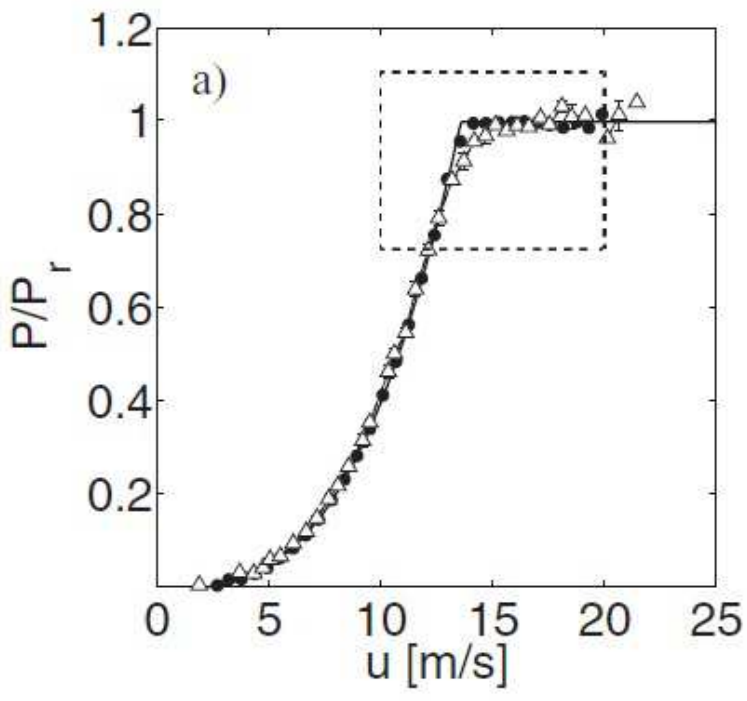


**Figura 13**. *Representación de todos los puntos fijos de la curva dinámica de potencia.*

Esta curva será la curva de potencia dinámica del aerogenerador estudiado. Hay otras posibilidades de resolución, o pueden existir problemas de estabilidad, pero esa información ampliada se encuentra en el artículo de investigación base que se ha utilizado [8].

## 4.3. Generación de datos de entrada

Para calcular la curva de potencia dinámica con una mayor precisión es necesario generar una serie de datos de entrada dentro de un espectro de velocidades controladas y dentro de un margen de turbulencia definido. Este trabajo ha seguido las indicaciones del artículo de investigación "Langevin power curve analysis for numerical wind energy converter models with new insights on high frecuency power performance" también desarrollado por el "Center for Wind Energy Research" de la Universidad de Oldenburg, en Alemania [4]. Los binomios P-v utilizados para el cálculo de la curva se han obtenido mediante la simulación con FAST con ciertos ajustes como se ha explicado anteriormente. Las series de velocidad de viento de entrada requeridas por FAST se han simulado con el programa TURBSIM bajo las condiciones especificadas en el artículo de la Universidad de Oldemburg.

Al igual que FAST, TurbSim es un programa desarrollado por NREL y su función es similar a la del programa IEC Wind: proporcionar las entradas de viento empleadas en los modelos con los que trabaja FAST. Como su propio nombre indica la principal diferencia entre IEC Wind y TURBSIM es que el primero está desarrollado para generar vientos constantes o con cambios de velocidad progresivos y que durante un periodo de tiempo sufren algún fenómeno meteorológico (vientos cortantes, ráfagas muy fuertes…), según las normas IEC, mientras que TurbSim permite generar vientos turbulentos y controlar algunas de las variables que los caracterizan [10].

Se complementa esta información, con una descripción más detallada sobre el programa y sus posibilidades en el Anexo G; y un ejemplo de su archivo principal de entrada en el Anexo H.

Se han modificado numerosas variables del archivo de entrada principal como crear un mallado de 100x100 metros y 9x9 nodos, o usar un modelo turbulento tipo “Kaimal”. El principal problema es que el artículo usa una turbulencia del 15% según las normas IEC, no obstante las posibilidades que da el programa son las mostradas en la **Figura 14** [13]:

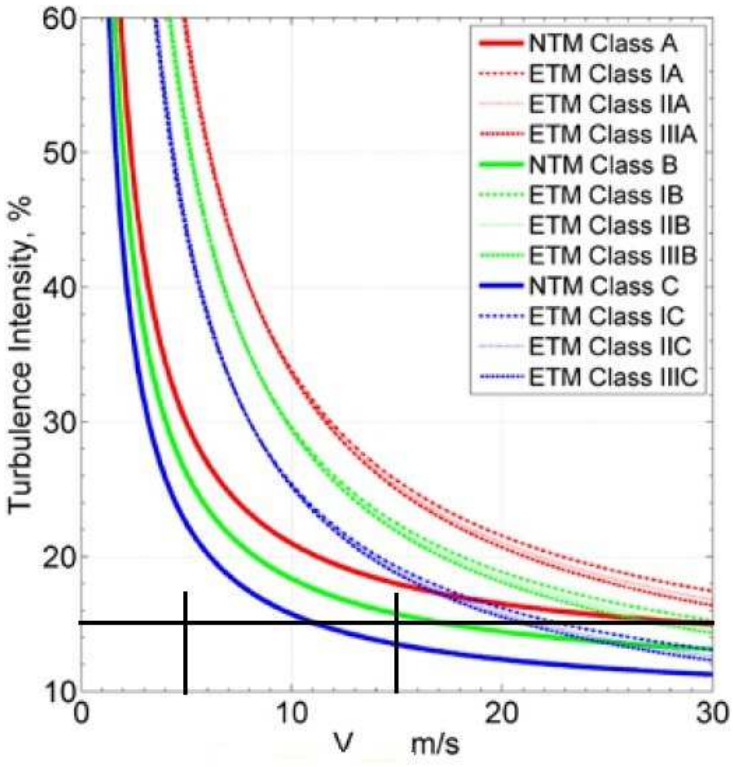


**Figura 14**. *Posibilidades elección de la intensidad de la turbulencia.*

Debido a que el espectro de velocidades en el que se va a trabajar va de 5 a 15m/s la curva más aproximada a la turbulencia del 15% será la NTM Class C, por lo que será la elegida. No obstante eso significará que la turbulencia del viento no será constante en todas las pruebas sino que variará en función de la velocidad del viento.

Para calcular adecuadamente la curva de potencia dinámica son necesarios una gran cantidad de datos: 21 velocidades medias del viento distintas, de 5 m/s hasta 15 m/s en pasos

de 0,5 m/s; y para cada velocidad media, 60 series de 600 segundos. Esto hace un total de 756.000 segundos de muestras, o lo que es lo mismo 210 horas. En realidad se necesitan bastantes más puesto que en cada serie se desechan los primeros 100 segundos para asegurar que no interfieran las condiciones iniciales del aerogenerador, llegando a un total de 245 horas. Obviamente no se pueden calcular a mano, por lo que ha sido necesario sistematizar las simulaciones y la clasificación de los archivos, primero para generar los archivos de viento (Anexo I), y después para ejecutar FAST (Anexo J).

## 4.4. Resultados de curva de potencia dinámica.

Una vez generados todos los archivos, se ha procedido al cálculo de la curva de potencia dinámica mediante el software desarrollado por el área AIRE del CIRCE. El programa aplica y sistematiza el método matemático presentado en el apartado 4.2. Las curvas de potencia dinámica del aerogenerador de 1,5 MW usado son las de la **Figura 15** y **Figura 16**.

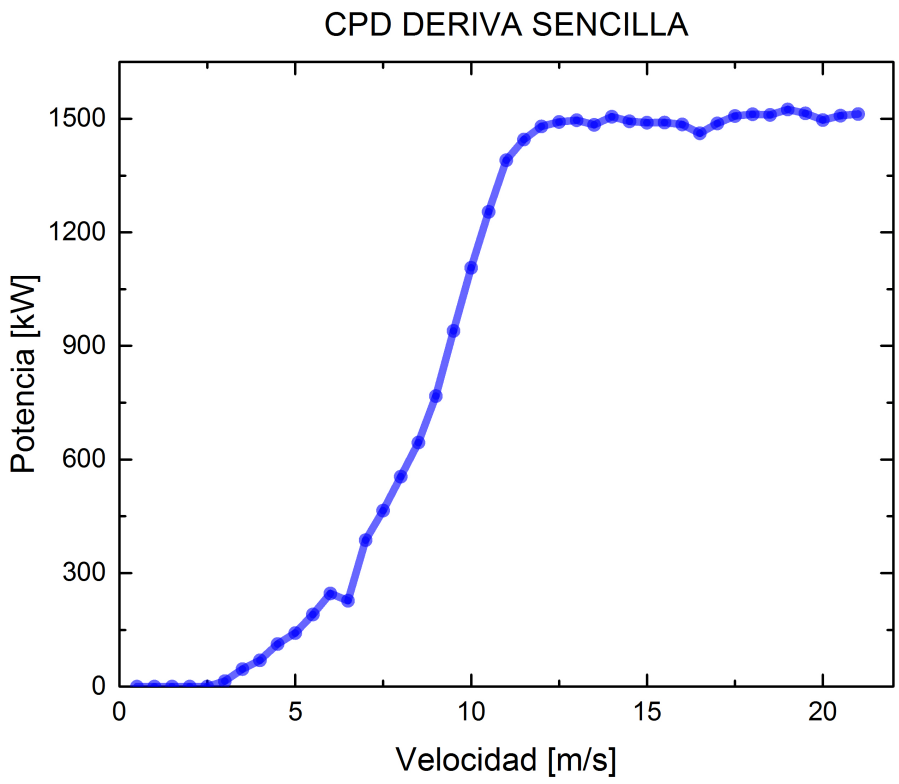


**Figura 15**. *Curva de potencia dinámica con deriva Sencilla.*

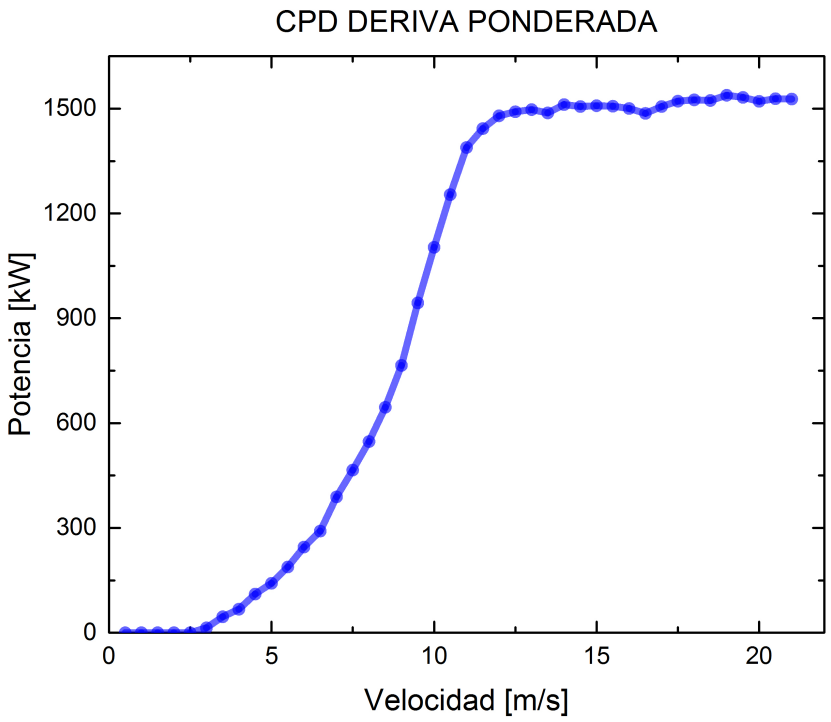


**Figura 16**. *Curva de potencia dinámica con deriva ponderada.*

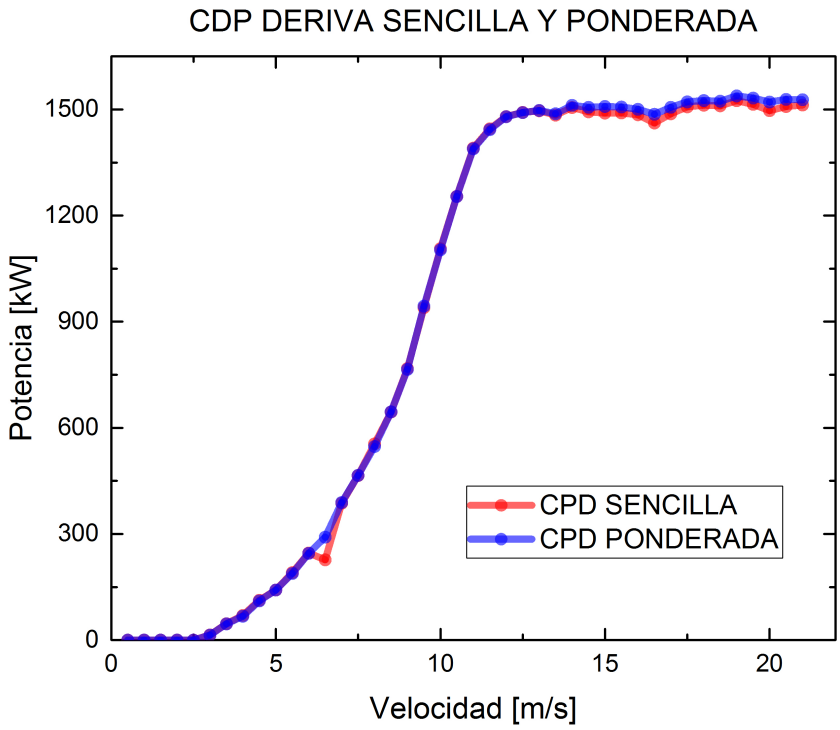


**Figura 17**. *Comparativa de curva de potencia dinámica con deriva sencilla y ponderada.*

Las curvas de potencia dinámica obtenidas tienen la misma forma que el resto de curvas de potencia de aerogeneradores, sean o no dinámicas. Como se puede comprobar en la **Figura 17** la principal diferencia entre la curva dinámica de potencia sencilla y la ponderada es que la segunda aplica una corrección en la detección de los puntos fijos.. Es decir aplica métodos matemáticos para tratar de unificar la progresión de los puntos de la curva, puesto que debido a que cada punto fijo se obtiene por separado, puede haber pequeñas variaciones que afectan al resultado final y a la continuidad de la misma al unirlos. Comienzan en parado, hay un periodo de arranque y a la velocidad nominal de viento incidente (11,2 m/s) se alcanza una estabilidad relativa, el paro automático por vientos demasiado fuertes fuera del rango de trabajo no se ha reflejado puesto que el estudio únicamente llega hasta 15m/s de velocidad de viento incidente.

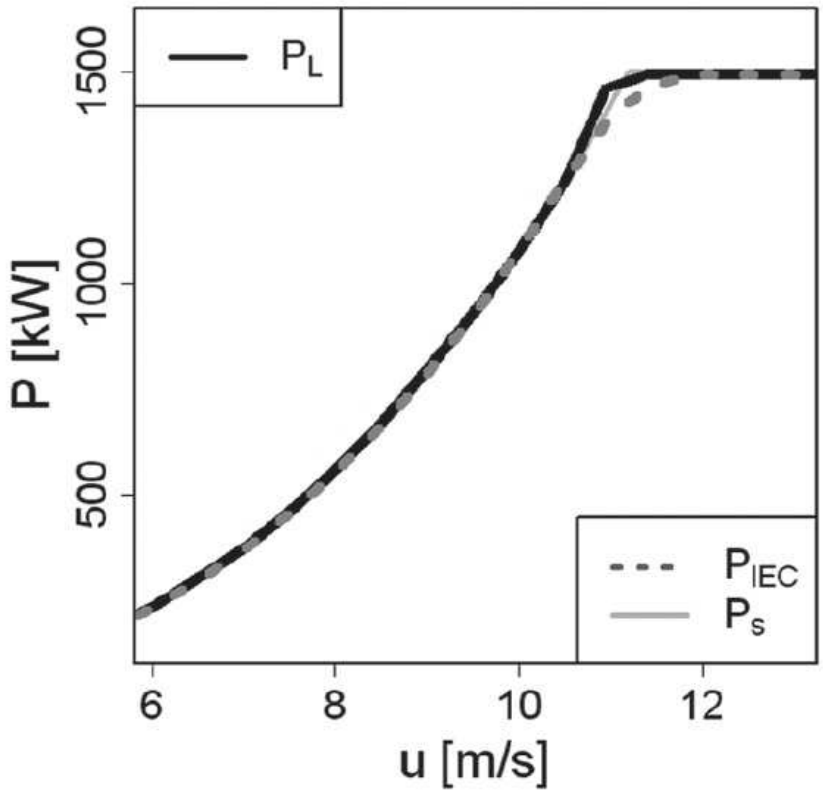


**Figura 18**. *Curva de potencia dinámica generada en el articulo [8].*

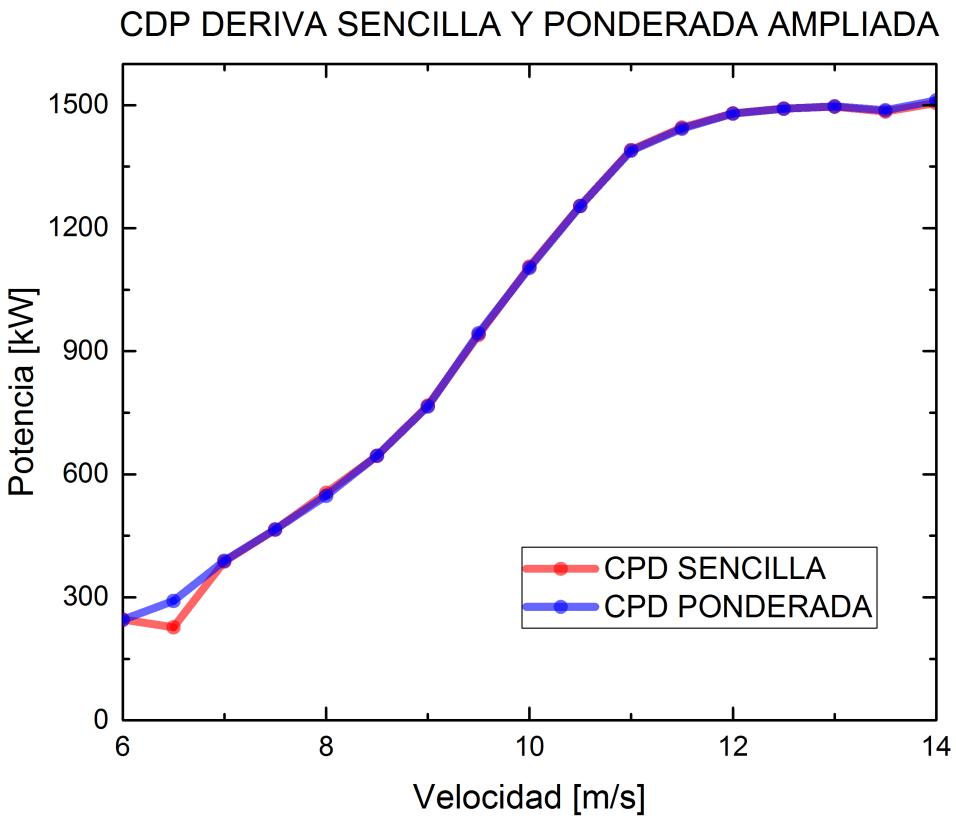


**Figura 19**. *Curva de potencia dinámica con deriva ponderada recortada.*

En la **Figura 18** se presenta la curva de potencia dinámica del articulo "*Markovian Power Curves for Wind Turbines*". La información más importante que aporta es la Curva de potencia según la normativa IEC ($P_{IEC}$) y la curva dinámica de potencia obtenida ($P_L$) según el método Langevin. La comparativa entre las curvas generadas, tomando la **Figura 19** con la $P_L$ de la **Figura 18** es uno de los objetivos de este trabajo. Ambas curvas siguen una estructura similar, no obstante las CDP generadas requieren una velocidad del viento incidente ligeramente superior para alcanzar la potencia nominal, en el último tramo de subida (entre 10m/s y 11m/s) tiene menos pendiente. Las razones para ello, que se amplían en las conclusiones, son debidas a la fuerte turbulencia de los vientos y la gran variabilidad de la orientación del viento incidente. Estas circunstancias hacen que el aerogenerador no disponga de un sistema de orientación suficientemente bueno/rápido para hacer una rotación de góndola que aproveche al máximo la energía del viento.

# 5. CONCLUSIONES Y LÍNEAS FUTURAS DE TRABAJO

## 5.1. Conclusiones

La realización de este trabajo me ha permitido familiarizarme con conceptos y herramientas básicos en energía eólica así como el desarrollo de un sistema útil para la obtención de series de datos (p,v) anonimizados que permitan la realización de estudios sin problemas de confidencialidad.

Las conclusiones finales del trabajo son las siguientes:

- Se ha estudiado a fondo el programa FAST y todas las herramientas de cálculo asociadas como IECwind o TurbSim determinando su estructura y metodología de uso. También las posibilidades de cálculo que ofrece el programa y las limitaciones del mismo respecto a variables de salida, amplitud de la simulación, solapamientos o influencia de la inercia del sistema en los resultados, y de sus herramientas de cálculo asociadas.
- Se han analizado los archivos de entrada requeridos, cómo modificarlos y cómo elegir las variables de salida deseadas. Así mismo se ha seleccionado un modelo de aerogenerador para las simulaciones, presentado sus características y justificado los motivos de su elección.
- Se han llevado a cabo diversas simulaciones partiendo de vientos incidentes uniformes y estables generados mediante IECwind. Estas simulaciones han servido para comprobar la potencia nominal del aerogenerador seleccionado, los tiempos de estabilización del sistema en función de la velocidad del viento incidente y la influencia de la inercia del sistema en movimiento sobre el resultado final. Se trata de simulaciones con vientos uniformes y sin turbulencias por lo que el comportamiento del aerogenerador es fácil de prever, por ejemplo la potencia nominal se alcanza con la velocidad de viento incidente teórica 11.2 m/s.
- La obtención de resultados a partir de series de velocidad de viento reales conllevan más problemas puesto que el viento sufre una variación muy grande y la inercia de la parte móvil del sistema (palas, buje, engranajes, rotor del alternador…) tiene gran influencia sobre el resultado final. Debido a esa gran inercia, la potencia generada no se adapta instantáneamente a las variaciones rápidas del viento incidente, sino que el sistema sigue girando casi a la misma velocidad. La consecuencia más visible es que para una potencia generada concreta, el rango de velocidades de viento que puede haber es enorme. Por ejemplo si un viento medio de 12m/s se para abruptamente, las palas seguirán girando a la misma velocidad los primeros instantes por lo que la curva potencia-velocidad será una nube de puntos y marcará que a 0m/s de velocidad de viento incidente la potencia generada es 1,5MW. Que el resultado sea una curva no determinística, una nube de puntos P-v, supone un problema de análisis de datos puesto que es imposible discernir realmente cual es el comportamiento del aerogenerador frente al viento más allá de las potencias generadas en cada instante.
- Por todo esto ha sido necesario calcular curva de potencia dinámica, ya que una vez desarrollada presenta de forma clara la respuesta que tendrá el aerogenerador en todo su rango de velocidades de trabajo. En el trabajo se ha presentado y explicado

paso a paso el procedimiento de cálculo de la CDP. Los vientos turbulentos incidentes para la simulación han sido generados con el programa TurbSim. Para generarlos se ha seguido una metodología que abarca numerosas variables y un rango de velocidades entre 5 y 15m/s. Una vez generados los vientos incidentes se ha desarrollado un programa en Matlab para automatizar todo el sistema de simulación, esto es debido a las limitaciones de tamaño de FAST y la gran cantidad de tablas distintas necesarias. El siguiente paso ha sido el cálculo de la curva dinámica de potencia con el programa de cálculo previamente desarrollado por el área AIRE. Siendo el resultado satisfactorio, una forma muy similar a otras curvas dinámicas de potencia de modelos parecidos: 1.5MW y palas de paso variable.

- Por último se ha comparado la curva de potencia dinámica generada con la del artículo de referencia "*Markovian Power Curves for Wind Turbines*" [4]. Observando la gran similitud y validando al menos para este modelo el sistema de cálculo de curvas de potencia dinámica del área AIRE. La única pequeña diferencia observada es la curva de potencia dinámica se ve ligeramente afectada siendo más evidente en el tramo entre 10m/s y 12m/s haciendo que tenga una pendiente un poco inferior y que el modelo requiera una velocidad del viento incidente ligeramente mayor que 11.2 m/s para alcanzar la potencia nominal. Las razones para ello son que los vientos incidentes tienen una orientación muy variable haciendo que la velocidad de seguimiento del aerogenerador, giro de la góndola, no sea suficientemente rápida para aprovechar al máximo toda la energía asociada al viento. Esto es debido a que el programa FAST tiene asociado un módulo de control muy rudimentario, que se está mejorando pero cuando se empezó este proyecto no estaba operativo todavía. Otra posibilidad es que el usuario programe el suyo propio mediante (FORTRAN), trabajo que excedía este proyecto, o implemente uno ya diseñado previamente por otros. Las limitaciones de dicho módulo junto con la orientación tan variable de incidencia del viento han afectado ligeramente la curva de potencia obtenida.

## 5.2. Líneas futuras de trabajo

El ámbito de la energía eólica es un campo en crecimiento en todo el mundo y cada vez se harán más estudios y trabajos relacionados con la materia. La escasez de software libre convierte a FAST en un programa muy interesante a pesar de la falta de una interfaz que facilite el trabajo al usuario, puesto que permite un enorme abanico de posibilidades de cálculo. Se puede pensar en numerosas líneas futuras de trabajo para continuar desarrollando este proyecto o tratar de profundizar en alguno de sus apartados. Las presentadas a continuación sólo son algunas de ellas.

Se podría profundizar en FAST y desarrollar un archivo de entrada que no fuera un generador de los ejemplos certificados que presenta el propio programa. También se podrían variar pequeños aspectos de la geometría, por ejemplo incrementar el radio y ver cómo varían la curvas de Potencia en función de ello. Existe la posibilidad de relacionar las potencias obtenidas, los vientos incidentes y las fuerzas que sufre la estructura en sus puntos más críticos.

Otro ámbito de trabajo serían las instalaciones offshore, las posibilidades que ofrecen son por un lado el análisis de los factores nuevos que le afectan: la plataforma flotante, el

movimiento, las características de los vientos marinos…; y por otro lado el poder utilizar modelos más grandes como el de 5MW que aún no ha sido usado en tierra más allá de prototipos.

En la línea de profundizar en el código de programa volver a compilar el archivo de cálculo para eliminar la limitación de 9999 segundos ampliaría sus posibilidades de trabajo. Esto es debido a que haría mucho más sencilla la simulación de las series largas, normalmente usadas en el ámbito de investigación de la energía eólica, al evitar todos los pasos intermedios y bucles ahora necesarios. Además sería interesante el desarrollo de herramientas adicionales del programa como el sistema de seguimiento y orientación de la góndola, que podría ser programado en FORTRAN y después fácilmente implementado dentro diagrama de bloques de SIMULINK.

Probar las interfaces ya existentes y desarrollar una metodología de trabajo con ellas también abriría puertas a nuevos proyectos y facilitaría el acceso de nuevos usuarios.

Respecto a la curva de potencia dinámica sería interesante comparar los resultados obtenidos con los de otros métodos de cálculo. También existe la posibilidad de ver cómo afectan los distintos métodos de optimización de la curva y si realmente suponen una mejora significativa.

# BIBLIOGRAFÍA


[1] José E. Gutierrez et al. *FAST Lognoter: Integración de herramientas para el cálculo de aerogeneradores 'Offshore'*. 2011.

[2] Milton Fabricio Pérez Gutiérrez. *Modelado y simulación de generadores eólicos*. Septiembre 2011.

[3] Jason M. Jonkman, Marshall L. Buhl Jr. NREL. *FAST User's Guide*. August 2005.

[4] Tanja A. mücke, Matthias Wächter, Patric Milan, Joachim Peinke. ForWind. *Langevin power curve analysis for numerical wind energy converter models with new insights on high frequency power performance*. July 2014.

[5] John Repici, *How To: The Comma Separated Value (CSV) File format* [en línea]. [Ultimo acceso: 04 Enero 2016]. Disponible: http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm#FileFormat

[6] *Límites y especificaciones de Excel* [En línea]. 25 mayo 2014. [Consulta: 15 Enero 2016]. Disponible: http://clasesexcel.com/index.php/component/k2/item/24-limites-y-especificaciones-de-excel.html.

[7] IEC 61400-1. *Wind turbines. Part 1: Design requirements.*

[8] E. Anahua, St. Barth and J.Peinke. *Markovian Power Curves for Wind Turbines*. September 2007.

[9] Julia Gottschall and Joachim Peinke. *How to improve the estimation of power curves for wind turbines*. January 2008.

[10] Albers A, Hinsch Ch. *Influence of different meteorological conditions on the power performance of large WECS*. *DEWI Magazin* 1996; 9: 40–49.

[11] Keiser K, Langreder W, Hohlen H, Hϕjstrup W. *Turbulence corrections for power curves. Wind Energy*, Peinke J, Schaumann P, Barth St (eds), Springer, Berlin, 2007; 159–162.

[12] Van Radecke H. *Turbulence correction of power curves*. *DEWI Magazin* 2004; 24: 56–62.

[13] B. J. Jonkman, L. Kilcher. NREL. *TurbSim User's Guide: Version 1.06.00*. September 2012.

# ÍNDICE DE FIGURAS E ÍNDICE DE TABLAS

# ANEXOS

## Anexo A. Metodología de trabajo paso a paso con FAST

La metodología de trabajo con el programa FAST es la siguiente: En una carpeta necesitas tener el archivo principal de entrada, todos a los que llame dentro de su código y el archivo "Pitch.IPT" que siempre es. El segundo archivo de entrada más importante viene marcado como "_AD", y llama a su vez a una serie de archivos guardados en las carpeta "aerodata" y "Wind". El archivo de la carpeta "Wind" es muy relevante puesto que será el que modificaremos para introducir las series de velocidad de viento para que FAST trabaje con ellas. Además necesitaras los archivos que te permitan ejecutar FAST en el entorno SIMULINK: "Simsetup", "Read_Fast_Input", "OpenLoop" y "Fast_SFunc". Todos estos archivos de entrada se relacionan entre sí según el esquema de la **Figura 10**.

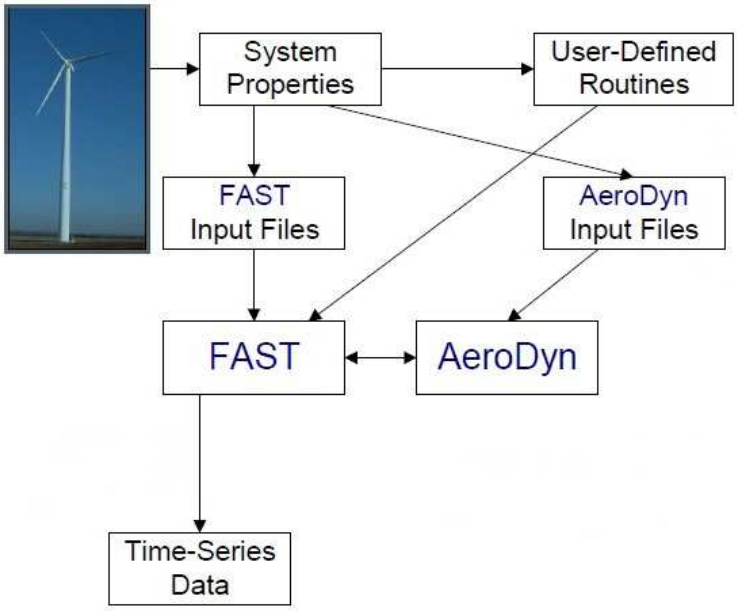


***Figura 3**. Metodología de operación en el trabajo.*

Una vez que tienes todos los archivos necesarios en la misma carpeta (o subcarpetas) Los pasos a seguir para la ejecución de FAST son los siguientes:

- Se abre Matlab y se escribe el comando "Simsetup",
- Simsetup se encarga de llamar al archivo "Read_Fast_Input" que pide el nombre del archivo principal de entrada, lo lee y carga sus variables en el Matlab.
- Se ejecuta "OpenLoop" con doble click o escribiéndolo en la ventana de comandos, abriéndose un diagrama de bloques de SIMULINK.
- Si se hace correr el diagrama de bloques éste recoge toda la información de los distintos ficheros de entrada a los que accede a través del archivo principal de entrada y ejecuta por último el archivo "Fast_SFunc". Este archivo es el que contiene todas las operaciones matemáticas que permiten la simulación del modelo y al ser puesto a trabajar genera las series con las variables de salida que han sido fijadas anteriormente. Es muy importante que si el ordenador tiene una estructura de 64 bits,

este archivo esté preparada para ello (por defecto el archivo es el adecuado para una estructura de 32 bits)

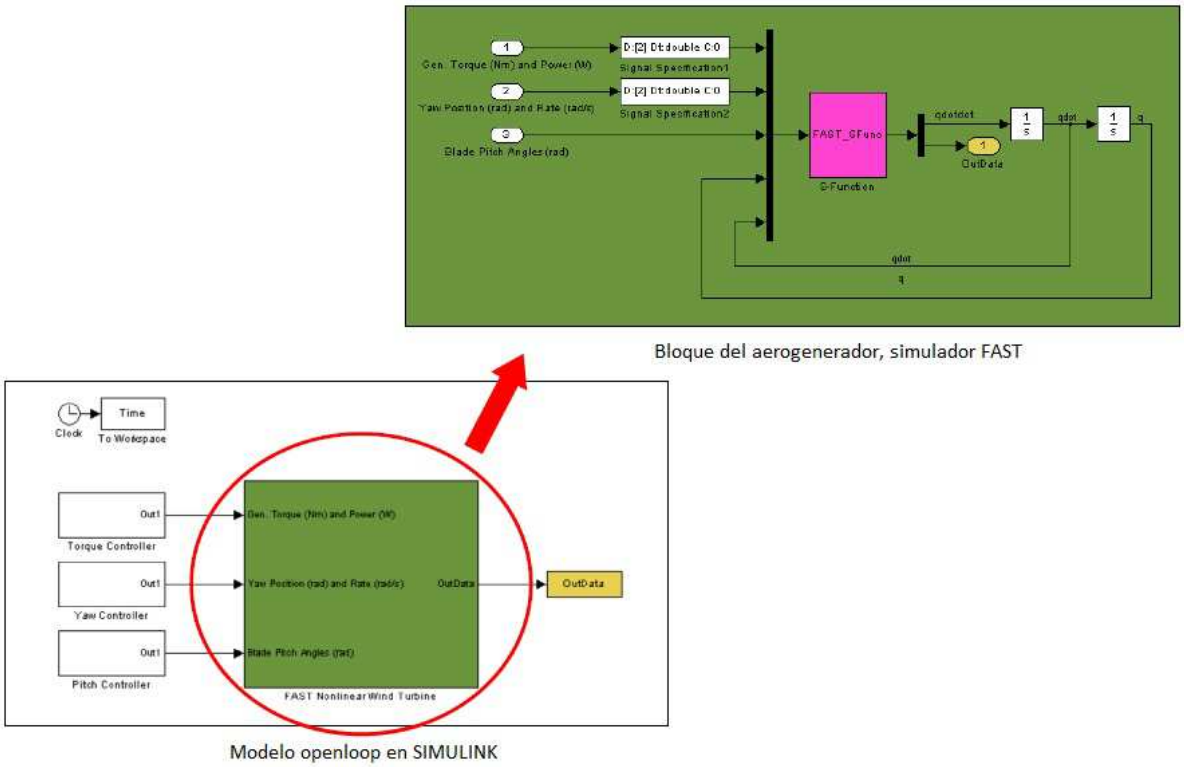


**Figura 4**. *Esquema trabajo FAST en Simulink.*

- Independientemente de las variables de salida que se elijan “Fast_SFunc” genera 3 archivos con el mismo nombre que el archivo principal de entrada añadiéndole “_SFunc” para indicar que ya han sido tratados, se abren y editan con el bloc de notas. Mantienen el mismo nombre pero se diferencian por el tipo de archivo: “.FSM” presenta información sobre las características de la turbina, de las propiedades de la torre y de otros elementos del conjunto, “.OPT” da información de los archivos aerodinámicos utilizados y por último “.OUT” incluye las series temporales junto a las variables elegidas para ser mostradas en ellas.

## Anexo B. Archivo principal de entrada FAST.

```
------------------------------------------------------------------------
--------- FAST INPUT FILE -----------------------------------------------

FAST certification Test #13: WindPACT 1.5 MW Baseline with many DOFs with VS
and VP and FF turbulence.

Model properties from "InputData1.5A08V07adm.xls" (from C. Hansen) with bugs
removed.  Compatible with FAST v7.02.00.

---------------------- SIMULATION CONTROL --------------------------------
--

False       Echo        - Echo input data to "echo.out" (flag)

   3        ADAMSPrep   - ADAMS preprocessor mode {1: Run FAST, 2: use FAST as
a preprocessor to create an ADAMS model, 3: do both} (switch)

   1        AnalMode    - Analysis mode {1: Run a time-marching simulation, 2:
create a periodic linearized model} (switch)

   3        NumBl       - Number of blades (-)

  700.0      TMax        - Total run time (s)

   0.005    DT          - Integration time step (s)

---------------------- TURBINE CONTROL -----------------------------------
--

   0        YCMode      - Yaw control mode {0: none, 1: user-defined from
routine UserYawCont, 2: user-defined from Simulink/Labview} (switch)

9999.9      TYCOn       - Time to enable active yaw control (s) [unused when
YCMode=0]

   1        PCMode      - Pitch control mode {0: none, 1: user-defined from
routine PitchCntrl, 2: user-defined from Simulink/Labview} (switch)

   5.0      TPCOn       - Time to enable active pitch control (s) [unused when
PCMode=0]

   1        VSContrl    - Variable-speed control mode {0: none, 1: simple VS,
2: user-defined from routine UserVSCont, 3: user-defined from
Simulink/Labview} (switch)

1800.0      VS_RtGnSp   - Rated generator speed for simple variable-speed
generator control (HSS side) (rpm) [used only when VSContrl=1]

8376.58     VS_RtTq     - Rated generator torque/constant generator torque in
Region 3 for simple variable-speed generator control (HSS side) (N-m) [used
only when VSContrl=1]

   0.002585 VS_Rgn2K    - Generator torque constant in Region 2 for simple
variable-speed generator control (HSS side) (N-m/rpm^2) [used only when
VSContrl=1]

9999.9E-9   VS_SlPc     - Rated generator slip percentage in Region 2 1/2 for
simple variable-speed generator control (%) [used only when VSContrl=1]

   1        GenModel    - Generator model {1: simple, 2: Thevenin, 3: user-
defined from routine UserGen} (switch) [used only when VSContrl=0]

True        GenTiStr    - Method to start the generator {T: timed using
TimGenOn, F: generator speed using SpdGenOn} (flag)
```

```
True        GenTiStp    - Method to stop the generator {T: timed using
TimGenOf, F: when generator power = 0} (flag)

9999.9      SpdGenOn    - Generator speed to turn on the generator for a
startup (HSS speed) (rpm) [used only when GenTiStr=False]

   0.0      TimGenOn    - Time to turn on the generator for a startup (s)
[used only when GenTiStr=True]

9999.9      TimGenOf    - Time to turn off the generator (s) [used only when
GenTiStp=True]

   1        HSSBrMode   - HSS brake model {1: simple, 2: user-defined from
routine UserHSSBr, 3: user-defined from Labview} (switch)

9999.9      THSSBrDp    - Time to initiate deployment of the HSS brake (s)

9999.9      TiDynBrk    - Time to initiate deployment of the dynamic generator
brake [CURRENTLY IGNORED] (s)

9999.9      TTpBrDp(1)  - Time to initiate deployment of tip brake 1 (s)

9999.9      TTpBrDp(2)  - Time to initiate deployment of tip brake 2 (s)

9999.9      TTpBrDp(3)  - Time to initiate deployment of tip brake 3 (s)
[unused for 2 blades]

9999.9      TBDepISp(1) - Deployment-initiation speed for the tip brake on
blade 1 (rpm)

9999.9      TBDepISp(2) - Deployment-initiation speed for the tip brake on
blade 2 (rpm)

9999.9      TBDepISp(3) - Deployment-initiation speed for the tip brake on
blade 3 (rpm) [unused for 2 blades]

9999.9      TYawManS    - Time to start override yaw maneuver and end standard
yaw control (s)

9999.9      TYawManE    - Time at which override yaw maneuver reaches final
yaw angle (s)

   0.0      NacYawF     - Final yaw angle for yaw maneuvers (degrees)

9999.9      TPitManS(1) - Time to start override pitch maneuver for blade 1
and end standard pitch control (s)

9999.9      TPitManS(2) - Time to start override pitch maneuver for blade 2
and end standard pitch control (s)

9999.9      TPitManS(3) - Time to start override pitch maneuver for blade 3
and end standard pitch control (s) [unused for 2 blades]

9999.9      TPitManE(1) - Time at which override pitch maneuver for blade 1
reaches final pitch (s)

9999.9      TPitManE(2) - Time at which override pitch maneuver for blade 2
reaches final pitch (s)

9999.9      TPitManE(3) - Time at which override pitch maneuver for blade 3
reaches final pitch (s) [unused for 2 blades]

   7.5      BlPitch(1)  - Blade 1 initial pitch (degrees)

   7.5      BlPitch(2)  - Blade 2 initial pitch (degrees)

   7.5      BlPitch(3)  - Blade 3 initial pitch (degrees) [unused for 2
blades]
```

```
   2.6      BlPitchF(1) - Blade 1 final pitch for pitch maneuvers (degrees)

   2.6      BlPitchF(2) - Blade 2 final pitch for pitch maneuvers (degrees)

   2.6      BlPitchF(3) - Blade 3 final pitch for pitch maneuvers (degrees)
[unused for 2 blades]

---------------------- ENVIRONMENTAL CONDITIONS -----------------------------
--

   9.80665  Gravity     - Gravitational acceleration (m/s^2)

---------------------- FEATURE FLAGS ----------------------------------------
--

True        FlapDOF1    - First flapwise blade mode DOF (flag)

True        FlapDOF2    - Second flapwise blade mode DOF (flag)

True        EdgeDOF     - First edgewise blade mode DOF (flag)

False       TeetDOF     - Rotor-teeter DOF (flag) [unused for 3 blades]

True        DrTrDOF     - Drivetrain rotational-flexibility DOF (flag)

True        GenDOF      - Generator DOF (flag)

False       YawDOF      - Yaw DOF (flag)

True        TwFADOF1    - First fore-aft tower bending-mode DOF (flag)

True        TwFADOF2    - Second fore-aft tower bending-mode DOF (flag)

True        TwSSDOF1    - First side-to-side tower bending-mode DOF (flag)

True        TwSSDOF2    - Second side-to-side tower bending-mode DOF (flag)

True        CompAero    - Compute aerodynamic forces (flag)

False       CompNoise   - Compute aerodynamic noise (flag)

---------------------- INITIAL CONDITIONS -----------------------------------
--

   0.0      OoPDefl     - Initial out-of-plane blade-tip displacement (meters)

   0.0      IPDefl      - Initial in-plane blade-tip deflection (meters)

   0.0      TeetDefl    - Initial or fixed teeter angle (degrees) [unused for
3 blades]

   0.0      Azimuth     - Initial azimuth angle for blade 1 (degrees)

  18.0      RotSpeed    - Initial or fixed rotor speed (rpm)

   0.0      NacYaw      - Initial or fixed nacelle-yaw angle (degrees)

   0.0      TTDspFA     - Initial fore-aft tower-top displacement (meters)

   0.0      TTDspSS     - Initial side-to-side tower-top displacement (meters)

---------------------- TURBINE CONFIGURATION --------------------------------
--

  35.0      TipRad      - The distance from the rotor apex to the blade tip
(meters)
```

```
   1.75     HubRad      - The distance from the rotor apex to the blade root
(meters)

   1        PSpnElN     - Number of the innermost blade element which is still
part of the pitchable portion of the blade for partial-span pitch control [1
to BldNodes] [CURRENTLY IGNORED] (-)

   0.0      UndSling    - Undersling length [distance from teeter pin to the
rotor apex] (meters) [unused for 3 blades]

   0.0      HubCM       - Distance from rotor apex to hub mass [positive
downwind] (meters)

  -3.3      OverHang    - Distance from yaw axis to rotor apex [3 blades] or
teeter pin [2 blades] (meters)

  -0.1449   NacCMxn     - Downwind distance from the tower-top to the nacelle
CM (meters)

   0.0      NacCMyn     - Lateral  distance from the tower-top to the nacelle
CM (meters)

   1.3890   NacCMzn     - Vertical distance from the tower-top to the nacelle
CM (meters)

  82.39     TowerHt     - Height of tower above ground level [onshore] or MSL
[offshore] (meters)

   1.61     Twr2Shft    - Vertical distance from the tower-top to the rotor
shaft (meters)

   0.0      TwrRBHt     - Tower rigid base height (meters)

  -5.0      ShftTilt    - Rotor shaft tilt angle (degrees)

   0.0      Delta3      - Delta-3 angle for teetering rotors (degrees) [unused
for 3 blades]

   0.0      PreCone(1)  - Blade 1 cone angle (degrees)

   0.0      PreCone(2)  - Blade 2 cone angle (degrees)

   0.0      PreCone(3)  - Blade 3 cone angle (degrees) [unused for 2 blades]

   0.0      AzimB1Up    - Azimuth value to use for I/O when blade 1 points up
(degrees)

---------------------- MASS AND INERTIA ----------------------------------
--

   0.0      YawBrMass   - Yaw bearing mass (kg)

  51.170E3  NacMass     - Nacelle mass (kg)

  15.148E3  HubMass     - Hub mass (kg)

   0.0      TipMass(1)  - Tip-brake mass, blade 1 (kg)

   0.0      TipMass(2)  - Tip-brake mass, blade 2 (kg)

   0.0      TipMass(3)  - Tip-brake mass, blade 3 (kg) [unused for 2 blades]

  49.130E3  NacYIner    - Nacelle inertia about yaw axis (kg m^2)

  53.036    GenIner     - Generator inertia about HSS (kg m^2)

  34.600E3  HubIner     - Hub inertia about rotor axis [3 blades] or teeter
axis [2 blades] (kg m^2)
```

```
---------------------- DRIVETRAIN --------------------------------------------
--

 100.0      GBoxEff     - Gearbox efficiency (%)

  95.0      GenEff      - Generator efficiency [ignored by the Thevenin and
user-defined generator models] (%)

  87.965    GBRatio     - Gearbox ratio (-)

False       GBRevers    - Gearbox reversal {T: if rotor and generator rotate
in opposite directions} (flag)

9999.9      HSSBrTqF    - Fully deployed HSS-brake torque (N-m)

9999.9      HSSBrDT     - Time for HSS-brake to reach full deployment once
initiated (sec) [used only when HSSBrMode=1]

"unused"    DynBrkFi    - File containing a mech-gen-torque vs HSS-speed curve
for a dynamic brake [CURRENTLY IGNORED] (quoted string)

   5.6E9    DTTorSpr    - Drivetrain torsional spring (N-m/rad)

   1.0E7    DTTorDmp    - Drivetrain torsional damper (N-m/(rad/s))

---------------------- SIMPLE INDUCTION GENERATOR ----------------------------
--

9999.9      SIG_SlPc    - Rated generator slip percentage (%) [used only when
VSContrl=0 and GenModel=1]

9999.9      SIG_SySp    - Synchronous (zero-torque) generator speed (rpm)
[used only when VSContrl=0 and GenModel=1]

9999.9      SIG_RtTq    - Rated torque (N-m) [used only when VSContrl=0 and
GenModel=1]

9999.9      SIG_PORt    - Pull-out ratio (Tpullout/Trated) (-) [used only when
VSContrl=0 and GenModel=1]

---------------------- THEVENIN-EQUIVALENT INDUCTION GENERATOR ---------------
--

9999.9      TEC_Freq    - Line frequency [50 or 60] (Hz) [used only when
VSContrl=0 and GenModel=2]

9998        TEC_NPol    - Number of poles [even integer > 0] (-) [used only
when VSContrl=0 and GenModel=2]

9999.9      TEC_SRes    - Stator resistance (ohms) [used only when VSContrl=0
and GenModel=2]

9999.9      TEC_RRes    - Rotor resistance (ohms) [used only when VSContrl=0
and GenModel=2]

9999.9      TEC_VLL     - Line-to-line RMS voltage (volts) [used only when
VSContrl=0 and GenModel=2]

9999.9      TEC_SLR     - Stator leakage reactance (ohms) [used only when
VSContrl=0 and GenModel=2]

9999.9      TEC_RLR     - Rotor leakage reactance (ohms) [used only when
VSContrl=0 and GenModel=2]

9999.9      TEC_MR      - Magnetizing reactance (ohms) [used only when
VSContrl=0 and GenModel=2]
```

```
---------------------- PLATFORM ----------------------------------------------
--

   0        PtfmModel   - Platform model {0: none, 1: onshore, 2: fixed bottom
offshore, 3: floating offshore} (switch)

"unused"    PtfmFile    - Name of file containing platform properties (quoted
string) [unused when PtfmModel=0]

---------------------- TOWER -------------------------------------------------
--

  10        TwrNodes    - Number of tower nodes used for analysis (-)

"Baseline_Tower.dat"    TwrFile - Name of file containing tower properties
(quoted string)

---------------------- NACELLE-YAW -------------------------------------------
--

   0.0      YawSpr      - Nacelle-yaw spring constant (N-m/rad)

   0.0      YawDamp     - Nacelle-yaw damping constant (N-m/(rad/s))

   0.0      YawNeut     - Neutral yaw position--yaw spring force is zero at
this yaw (degrees)

---------------------- FURLING -----------------------------------------------
--

False       Furling     - Read in additional model properties for furling
turbine (flag)

"unused"    FurlFile    - Name of file containing furling properties (quoted
string) [unused when Furling=False]

---------------------- ROTOR-TEETER ------------------------------------------
--

   0        TeetMod     - Rotor-teeter spring/damper model {0: none, 1:
standard, 2: user-defined from routine UserTeet} (switch) [unused for 3
blades]

   0.0      TeetDmpP    - Rotor-teeter damper position (degrees) [used only
for 2 blades and when TeetMod=1]

   0.0      TeetDmp     - Rotor-teeter damping constant (N-m/(rad/s)) [used
only for 2 blades and when TeetMod=1]

   0.0      TeetCDmp    - Rotor-teeter rate-independent Coulomb-damping moment
(N-m) [used only for 2 blades and when TeetMod=1]

   0.0      TeetSStP    - Rotor-teeter soft-stop position (degrees) [used only
for 2 blades and when TeetMod=1]

   0.0      TeetHStP    - Rotor-teeter hard-stop position (degrees) [used only
for 2 blades and when TeetMod=1]

   0.0      TeetSSSp    - Rotor-teeter soft-stop linear-spring constant (N-
m/rad) [used only for 2 blades and when TeetMod=1]

   0.0      TeetHSSp    - Rotor-teeter hard-stop linear-spring constant (N-
m/rad) [used only for 2 blades and when TeetMod=1]

---------------------- TIP-BRAKE ---------------------------------------------
--
```

```
   0.0      TBDrConN    - Tip-brake drag constant during normal operation,
Cd*Area (m^2)

   0.0      TBDrConD    - Tip-brake drag constant during fully-deployed
operation, Cd*Area (m^2)

   0.0      TpBrDT      - Time for tip-brake to reach full deployment once
released (sec)

---------------------- BLADE -----------------------------------------------
--

"Baseline_Blade.dat"    BldFile(1) - Name of file containing properties for
blade 1 (quoted string)

"Baseline_Blade.dat"    BldFile(2) - Name of file containing properties for
blade 2 (quoted string)

"Baseline_Blade.dat"    BldFile(3) - Name of file containing properties for
blade 3 (quoted string) [unused for 2 blades]

---------------------- AERODYN ---------------------------------------------
--

"CurvaDinamica_AD.ipt"         ADFile     - Name of file containing AeroDyn
input parameters (quoted string)

---------------------- NOISE -----------------------------------------------
--

"unused"    NoiseFile   - Name of file containing aerodynamic noise input
parameters (quoted string) [used only when CompNoise=True]

---------------------- ADAMS -----------------------------------------------
--

"Baseline_ADAMS.dat"    ADAMSFile  - Name of file containing ADAMS-specific
input parameters (quoted string) [unused when ADAMSPrep=1]

---------------------- LINEARIZATION CONTROL -------------------------------
--

"Baseline_Linear.dat"   LinFile    - Name of file containing FAST
linearization parameters (quoted string) [unused when AnalMode=1]

---------------------- OUTPUT ----------------------------------------------
--

True        SumPrint    - Print summary data to "<RootName>.fsm" (flag)

1           OutFileFmt  - Format for tabular (time-marching) output file(s)
(1: text file [<RootName>.out], 2: binary file [<RootName>.outb], 3: both)
(switch)

True        TabDelim    - Use tab delimiters in text tabular output file?
(flag)

"ES10.3E2"  OutFmt      - Format used for text tabular output (except time).
Resulting field should be 10 characters. (quoted string)  [not checked for
validity!]

  0.0      TStart      - Time to begin tabular output (s)

 13.5        DecFact     - Decimation factor for tabular output {1: output
every time step} (-)

   1.0      SttsTime    - Amount of time between screen status messages (sec)
```

```
   0.0      NcIMUxn     - Downwind distance from the tower-top to the nacelle
IMU (meters)

   0.0      NcIMUyn     - Lateral  distance from the tower-top to the nacelle
IMU (meters)

   0.0      NcIMUzn     - Vertical distance from the tower-top to the nacelle
IMU (meters)

   0.99     ShftGagL    - Distance from rotor apex [3 blades] or teeter pin [2
blades] to shaft strain gages [positive for upwind rotors] (meters)

   2        NTwGages    - Number of tower nodes that have strain gages for
output [0 to 9] (-)

  4,7       TwrGagNd    - List of tower nodes that have strain gages [1 to
TwrNodes] (-) [unused if NTwGages=0]

   0        NBlGages    - Number of blade nodes that have strain gages for
output [0 to 9] (-)

   0        BldGagNd    - List of blade nodes that have strain gages [1 to
BldNodes] (-) [unused if NBlGages=0]

            OutList     - The next line(s) contains a list of output
parameters.  See OutList.xlsx for a listing of available output channels, (-)

"HorWindV"              - Velocidad horizontal viento

"GenPwr"                - Potencia eléctrica generada por el aerogenerador

END of FAST input file (the word "END" must appear in the first 3 columns of
this last line).

---------------------------------------------------------------------------
```

# Anexo C. Ejemplos de posibilidades de variables de salida.

| | | | | | |
|---|---|---|---|---|---|
| Hub and Rotor Loads | | | | | |
| | LSShftFxa | LSShftFxs, LSSGagFxa, LSSGagFxs, RotThrust | Low-speed shaft thrust force (this is constant along the shaft and is equivalent to the rotor thrust force) | Directed along the xa- and xs-axes | (kN) |
| | LSShftFya | LSSGagFya | Rotating low-speed shaft shear force (this is constant along the shaft) | Directed along the ya-axis | (kN) |
| | LSShftFza | LSSGagFza | Rotating low-speed shaft shear force (this is constant along the shaft) | Directed along the za-axis | (kN) |
| | LSShftFys | LSSGagFys | Nonrotating low-speed shaft shear force (this is constant along the shaft) | Directed along the ys-axis | (kN) |
| | LSShftFzs | LSSGagFzs | Nonrotating low-speed shaft shear force (this is constant along the shaft) | Directed along the zs-axis | (kN) |
| | LSShftMxa | LSShftMxs, LSSGagMxa, LSSGagMxs, RotTorq, LSShftTq | Low-speed shaft torque (this is constant along the shaft and is equivalent to the rotor torque) | About the xa- and xs-axes | (kN·m) |
| | LSSTipMya | | Rotating low-speed shaft bending moment at the shaft tip (teeter pin for 2-blader, apex of rotation for 3-blader) | About the ya-axis | (kN·m) |
| | LSSTipMza | | Rotating low-speed shaft bending moment at the shaft tip (teeter pin for 2-blader, apex of rotation for 3-blader) | About the za-axis | (kN·m) |

| Category | Name | Other Name(s) | Description | Convention | Units |
|---|---|---|---|---|---|
| | RotCq | LSShftCq | Rotor torque coefficient (this is equivalent to the low-speed shaft torque coefficient) | N/A | (-) |
| | RotCp | LSShftCp | Rotor power coefficient (this is equivalent to the low-speed shaft power coefficient) | N/A | (-) |
| | RotCt | LSShftCt | Rotor thrust coefficient (this is equivalent to the low-speed shaft thrust coefficient) | N/A | (-) |
| Shaft Strain Gage Loads | | | | | |
| | LSSGagMya | | Rotating low-speed shaft bending moment at the shaft's strain gage (shaft strain gage located by input ShftGagL) | About the ya-axis | (kN·m) |
| | LSSGagMza | | Rotating low-speed shaft bending moment at the shaft's strain gage (shaft strain gage located by input ShftGagL) | About the za-axis | (kN·m) |
| | LSSGagMys | | Nonrotating low-speed shaft bending moment at the shaft's strain gage (shaft strain gage located by input ShftGagL) | About the ys-axis | (kN·m) |
| | LSSGagMzs | | Nonrotating low-speed shaft bending moment at the shaft's strain gage (shaft strain gage located by input ShftGagL) | About the zs-axis | (kN·m) |
| Generator and High-Speed Shaft Loads | | | | | |
| | HSShftTq | | High-speed shaft torque (this is constant along the shaft) | Same sign as LSShftTq / RotTorq / LSShftMxa / LSShftMxs / LSSGagMxa / LSSGagMxs | (kN·m) |
| | HSShftPwr | | High-speed shaft power | Same sign as HSShftTq | (kW) |
| | HSShftCq | | High-speed shaft torque coefficient | N/A | (-) |
| | HSShftCp | | High-speed shaft power coefficient | N/A | (-) |
| | GenTq | | Electrical generator torque | Positive reflects power extracted and negative represents a motoring-up | (kN·m) |

| | | | | | |
|---|---|---|---|---|---|
| Wind Motions | | | | | |
| | WindVxi | uWind | Nominally downwind component of the hub-height wind velocity | Directed along the xi-axis | (m/s) |
| | WindVyi | vWind | Cross-wind component of the hub-height wind velocity | Directed along the yi-axis | (m/s) |
| | WindVzi | wWind | Vertical component of the hub-height wind velocity | Directed along the zi-axis | (m/s) |
| | TotWindV | | Total hub-height wind velocity magnitude | N/A | (m/s) |
| | HorWindV | | Horizontal hub-height wind velocity magnitude | In the xi- and yi-plane | (m/s) |
| | HorWndDir | | Horizontal hub-height wind direction. Please note that FAST uses the opposite sign convention that AeroDyn uses. Put a "-", "_", "m", or "M" character in front of this variable name if you want to use the AeroDyn convention. | About the zi-axis | (deg) |
| | VerWndDir | | Vertical hub-height wind direction | About an axis orthogonal to the zi-axis and the HorWindV-vector | (deg) |
| Blade 1 Tip Motions | | | | | |
| | TipDxc1 | OoPDefl1 | Blade 1 out-of-plane tip deflection (relative to the undeflected position) | Directed along the xc1-axis | (m) |
| | TipDyc1 | IPDefl1 | Blade 1 in-plane tip deflection (relative to the undeflected position) | Directed along the yc1-axis | (m) |
| | TipDzc1 | TipDzb1 | Blade 1 axial tip deflection (relative to the undeflected position) | Directed along the zc1- and zb1-axes | (m) |
| | TipDxb1 | | Blade 1 flapwise tip deflection (relative to the undeflected position) | Directed along the xb1-axis | (m) |

## Anexo D. Posibles CerTest para usar en FAST.

| Test Name | Turbine Name | No. Blades (-) | Rotor Diameter (m) | Rated Power (kW) | Test Description |
|---|---|---|---|---|---|
| Test01 | AWT-27CR2 | 2 | 27 | 175 | Flexible, fixed yaw error, steady wind |
| Test02 | AWT-27CR2 | 2 | 27 | 175 | Flexible, start-up, HSS brake shut-down, steady wind |
| Test03 | AWT-27CR2 | 2 | 27 | 175 | Flexible, free yaw, steady wind |
| Test04 | AWT-27CR2 | 2 | 27 | 175 | Flexible, free yaw, turbulence |
| Test05 | AWT-27CR2 | 2 | 27 | 175 | Flexible, generator start-up, tip-brake shutdown, steady wind |
| Test06 | AOC-15/50 | 3 | 15 | 50 | Flexible, generator start-up, tip-brake shutdown, steady wind |
| Test07 | AOC-15/50 | 3 | 15 | 50 | Flexible, free yaw, turbulence |
| Test08 | AOC-15/50 | 3 | 15 | 50 | Flexible, fixed yaw error, steady wind |
| Test09 | UAE VI downwind | 2 | 10 | 20 | Flexible, yaw ramp, steady wind |
| Test10 | UAE VI upwind | 2 | 10 | 20 | Rigid, power curve, ramp wind |
| Test11 | WP 1.5 MW | 3 | 70 | 1500 | Flexible, variable speed & pitch control, pitch failure, turbulence |
| Test12 | WP 1.5 MW | 3 | 70 | 1500 | Flexible, variable speed & pitch control, ECD event |
| Test13 | WP 1.5 MW | 3 | 70 | 1500 | Flexible, variable speed & pitch control, turbulence |
| Test14 | WP 1.5 MW | 3 | 70 | 1500 | Flexible, stationary linearization, vacuum |
| Test15 | SWRT | 3 | 5.8 | 10 | Flexible, variable speed control, free yaw, tail-furl, EOG01 event |
| Test16 | SWRT | 3 | 5.8 | 10 | Flexible, variable speed control, free yaw, tail-furl, EDC01 event |
| Test17 | SWRT | 3 | 5.8 | 10 | Flexible, variable speed control, free yaw, tail-furl, turbulence |

## Anexo E. Datos entrada CSV, salida series P-v.

```
%Programa para trabajar con los archivos CSV de entrada, con datos
reales de viento, y obtener las series de P-v.

Serie=9;
for i=1:Serie
%Abrimos el input general de FAST y lo guardamos con otro nombre, para
%obtener una salida con nombre distinto para cada caso.
numLines = 198;
name1='TablaViento';
name2=num2str(i);
name3='.fst';
name4='_SFunc.out';
name7='_SFunc.opt';
name8='_SFunc.fsm';
NombreArchivoINP = strcat(name1,name3);
NombreArchivoINP_OUT = strcat(name1,name2,name3);
NombreArchivoOUT = strcat(name1,name2,name4);
NombreArchivosNoUsados1 = strcat(name1,name2,name7);
NombreArchivosNoUsados2 = strcat(name1,name2,name8);
fileID = fopen(NombreArchivoINP,'r');
mydata = cell(1, numLines);
for m = 1:numLines
   mydata{m} = fgetl(fileID);
end
fclose(fileID);

fileID = fopen(NombreArchivoINP_OUT,'w');
fprintf(fileID,'%s\n',mydata{1:numLines-1});
fprintf(fileID,'%s',mydata{numLines});
fclose(fileID);



%Se abre el archivo de entrada _AD, para modificar la entrada del
viento y lo guardamos igual, se irá actualizando cada serie nueva, así
se evita
%tener mil que no nos aportan nada.
replaceLine1 = 10;
numLinesAD = 39;
entradaAD='TablaViento_AD.ipt';
fileAD = fopen(entradaAD,'r');
mydataAD = cell(1, numLinesAD);
for k = 1:numLinesAD
   mydataAD{k} = fgetl(fileAD);
end
fclose(fileAD);

% Para que cambie de nombre el archivo de entrada de viento.
text='"Wind/WP_Baseline/';
textMedio3='TablaViento';
textMedio5=num2str(i);
textFinInput='.WND"        WindFile - Name of file containing wind
data (quoted string)';

rutaViento = strcat(text,textMedio3,textMedio5,textFinInput);

mydataAD{replaceLine1} = rutaViento;
```

```
fileAD = fopen(entradaAD,'w');
fprintf(fileAD,'%s\n',mydataAD{1:numLinesAD-1});
fprintf(fileAD,'%s',mydataAD{numLinesAD});
fclose(fileAD);

%Para crear el archivo viento, con la velocidad para cada uno de los
%tiempos.
nSegundos = 9999;
segundosPorDato=0.096;
%Con la función ceil() se consigue que tome el entero justo por
encima.
nDatos=ceil(nSegundos/segundosPorDato);
entradaExcel='prueba text largo.csv';
velocidades=xlsread(entradaExcel);
tiempo=[0:segundosPorDato:nSegundos];
ceros=zeros(1,nDatos);

name9='.WND';
tViento = strcat(name1,name2,name9);
fileWind = fopen(tViento,'w');
fprintf(fileWind,'! Time        Wind        Wind        Vertical
Horiz.      Pwr.Law     Lin.Vert.   Gust\n');
fprintf(fileWind,'!             Speed       Dir         Speed
Shear       Vert.Shr    Shear       Speed\n');
fprintf(fileWind,'! (sec)       (m/s)       (Deg)       (m/s)
(m/s)\n');

%Para trabajar con la superposición entre las distintas simulaciones,
%hay que restar 199 segundos entre una y otra, teniendo en cuenta que
cada
%paso de iteración son 0,096 segundos: 199/0,096=2073. La primera
iteración
%no se superpone a nada, de ahí el uso del operador IF

if i == 1
   z = 0;
   else
   z = 1;
end


for r=1:nDatos
sup=r+(((i-1)*nDatos)-(z*2073));
fprintf(fileWind,'%9.3f  %9.3f   %9.3f   %9.3f   %9.3f   %9.3f   %9.3f
%9.3f\n',tiempo(r),velocidades(sup),ceros(r),ceros(r),ceros(r),ceros(r
),ceros(r),ceros(r));
end
fclose(fileWind);


%Clasificar archivos Viento para que pueda ejecutarse FAST.
rutaArchivoViento='C:\Users\Juan\Desktop\FAST curva
viento\Wind\WP_Baseline';
movefile(tViento,rutaArchivoViento);

%Para ejecutar Fast en Simulink, hay que saltarse la llamada de
Simsetup que solo es un paso intermedio hacia Read-Fast,
%Que se ejecuta ahora:
```

```
%------------------------------------------------------------------------
-----
% This script reads FAST *.fst files and creates Matlab workspace
variables
% required to run the FAST s-function in Simulink.
% Developed at NREL's National Wind Technology Center
%     modified 10-Mar-2010 by B. Jonkman, NREL
%------------------------------------------------------------------------
-----

%------------------------------------------------------------------------
-----
%open the FAST input file
%------------------------------------------------------------------------
-----
input_fast=NombreArchivoINP_OUT;
fid=fopen(input_fast,'r');
if (fid == -1)
    disp(['Input file:  ',input_fast,' could not be opened.']);
    return
end

for iLine=1:2;                                          % skip the first
two lines
    fgetl(fid);
end
disp( fgetl(fid) )                                      % line 3: input
file description

for iLine=1:5;                                          % skip 5 lines
(through AnalMode)
    fgetl(fid);
end

% read simulation parameters
NumBl= fscanf(fid,'%i',1); fgetl(fid);                  % the S-function
needs this parameter for initialization
TMax = fscanf(fid,'%g',1); fgetl(fid);                  % can be used in
Simulation Parameters menu
DT   = fscanf(fid,'%g',1); fgetl(fid);                  % can be used in
Simulation Parameters menu

for iLine=1:34                                          % skip 34 lines
(12-45)
    fgetl(fid);                                         % (first part of
Turbine control)
end

BlPitch = zeros(3,1);
for iLine=1:3                                           % read lines 46-48
    BlPitch(iLine) = fscanf(fid,'%g',1); fgetl(fid); % initial blade
pitch
end

for iLine=1:20                                          % skip 20 lines
(49-68)
    fgetl(fid);                                         % (Turbine control
through feature switches)
end
```

```
% read initial conditions section
OoPDefl  = fscanf(fid,'%g',1); fgetl(fid);          % read line 69,
initial OoPDefl
IPDefl   = fscanf(fid,'%g',1); fgetl(fid);          % read line 70,
initial IPDefl
TeetDefl = fscanf(fid,'%g',1); fgetl(fid);          % read line 71,
initial TeetDefl
Azimuth  = fscanf(fid,'%g',1); fgetl(fid);          % read line 72,
initial Azimuth
RotSpeed = fscanf(fid,'%g',1); fgetl(fid);          % read line 73,
initial RotSpeed
NacYaw   = fscanf(fid,'%g',1); fgetl(fid);          % read line 74,
initial NacYaw
TTDspFA  = fscanf(fid,'%g',1); fgetl(fid);          % read line 75,
initial TTDspFA
TTDspSS  = fscanf(fid,'%g',1); fgetl(fid);          % read line 76,
initial TTDspSS

    %Turbine Configuration section
for iLine=1:2                                       % skip 2 lines
(77-78)
    fgetl(fid);                                     % (Turbine
configuration)
end
HubRad  = fscanf(fid,'%g',1); fgetl(fid);           % read line 79,
HubRad
for iLine=1:7                                       % skip 7 lines
(80-86)
    fgetl(fid);                                     % (Turbine
configuration parameters)
end
TowerHt = fscanf(fid,'%g',1); fgetl(fid);           % read line 87,
TowerHt
for iLine=1:7                                       % skip 7 lines
(88-94)
    fgetl(fid);                                     % (Turbine
configuration parameters)
end
AzimB1Up = fscanf(fid,'%g',1); fgetl(fid);          % read line 95,
Azimuth when Blade 1 is up

    % MASS AND INERTIA section
for iLine=1:10                                      % skip 10 lines
(96-105)
    fgetl (fid);                                    % (MASS AND
INERTIA parameters)
end

    % DRIVETRAIN
for iLine=1:2                                       % skip 2 lines
(106-107)
    fgetl (fid);                                    % (DRIVETRAIN
parameters)
end
GenEff  = fscanf(fid,'%g',1); fgetl(fid);           % line 108 (used
to initialize
GBRatio = fscanf(fid,'%g',1); fgetl(fid);           % line 109
for iLine=1:6                                       % skip 6 lines
(110-115)
```

```
    fgetl (fid);                                        % (DRIVETRAIN
parameters)
end

    % SIMPLE INDUCTION GENERATOR
for iLine=1:5                                           % skip 5 lines
(106-120)
    fgetl (fid);                                        % (SIMPLE
INDUCTION GENERATOR parameters)
end

    % THEVENIN-EQUIVALENT INDUCTION GENERATOR
for iLine=1:9                                           % skip 5 lines
(121-129)
    fgetl (fid);                                        % (THEVENIN-
EQUIVALENT INDUCTION GENERATOR)
end
    % PLATFORM MODEL
fgetl(fid);                                             % heading, line
130
PtfmModel = fscanf(fid,'%g',1); fgetl(fid);             % read line 131,
PtfmModel
temp_char = fgetl(fid);                                 % read line 132,
PtfmFile line

%-----------------------------------------------------------------------
--
% If we have a platform model, read initial platform displacements:
% Open the platform file (get the name of the file first)
%-----------------------------------------------------------------------
--
if ( PtfmModel >= 1 && PtfmModel <= 3 )
   temp_char = strtrim(temp_char);                      % remove leading
(and trailing spaces)
   if strfind( '"''', temp_char(1) )                    % the string
starts with quotes
      PtfmFile = strtok( temp_char(2:end), temp_char(1) ); %read to
the end of the quote
   else
      PtfmFile = strtok( temp_char );                   % read to the
first white space
   end

   fid2=fopen(PtfmFile);
   if (fid2 == -1)
      disp(['Platform file:  ',PtfmFile,' could not be opened.']);
      return
   end
   for iLine=1:11;
      fgetl(fid2);                                      % skip first 11
lines of PtfmFile
   end
   PtfmSurge = fscanf(fid2,'%g',1); fgetl(fid2);        % Initial platform
surge
   PtfmSway  = fscanf(fid2,'%g',1); fgetl(fid2);        % Initial platform
sway
   PtfmHeave = fscanf(fid2,'%g',1); fgetl(fid2);        % Initial platform
heave
   PtfmRoll  = fscanf(fid2,'%g',1); fgetl(fid2);        % Initial platform
roll
```

```
   PtfmPitch = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
pitch
   PtfmYaw   = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
yaw
   fclose(fid2);
end

for iLine=1:8                                       % skip 8 lines
(133-140) of the FAST file
   fgetl(fid);                                      % (Nacelle-Yaw to
beginning of Furling)
end

Furling   = strtrim( fgetl(fid) );                  % read line 141,
Furling
temp_char = fgetl(fid);                             % read line 142,
FurlFile line

%------------------------------------------------------------------------
--
% If furling, read initial rotor and tail furl values:
% Open the Furling file (get the name of the file first)
%------------------------------------------------------------------------
--
if ( strcmpi(Furling(1), 't') )
   temp_char = strtrim(temp_char);                  % remove leading
(and trailing spaces)
   if strfind( '"''', temp_char(1) )                % the string
starts with quotes
      FurlFile = strtok( temp_char(2:end), temp_char(1) ); % read to
the end of the quote
   else
      FurlFile = strtok( temp_char );               % read to the
first whitespace
   end

   fid2=fopen(FurlFile);
   if (fid2 == -1)
      disp(['Furling file:  ',FurlFile,' could not be opened.']);
      return
   end
   for iLine=1:7;
      fgetl(fid2);                                  % skip first 7
lines of FurlFile
   end
   RotFurl  = fscanf(fid2,'%g',1); fgetl(fid2);     % Initial rotor
furl
   TailFurl = fscanf(fid2,'%g',1); fgetl(fid2);     % Initial tail
furl
   fclose(fid2);
end

for iLine=1:42                                      % skip 42 lines
(143-184) of the FAST file
    fgetl(fid);                                     % (Rotor-Teeter to
beginning of OutList)
end

%------------------------------------------------------------------------
-----
```

```
% get the FAST outputs (OutList) and close the FAST input file
%------------------------------------------------------------------------
-----
%temp_char = strtrim( fgetl(fid) );
NumOuts   = 0;
OutList   = {};                                        % clear this from
subsequent runs
delims    = [' ,;''"' char(9)];                        % allowable
delimiters

while ~feof(fid)

      % get the next line
   temp_char = strtrim( fgetl(fid) );

   if length(temp_char) < 3     %FAST doesn't have 2-character
variables;

            % skip any blank lines
      if isempty(temp_char)
         continue;
      end

   elseif strcmpi( temp_char(1:3), 'end' )
            % we've reached the end
        break;
   end

      % get the equivalent "line" that Fortran would read
   if strfind( '"''', temp_char(1) )                   % the string
starts with quotes
      temp_char = strtok( temp_char(2:end), temp_char(1) ); % read to
the end of the quotes
   else
      temp_char = strtok( temp_char );                 % read to the
first whitespace
   end

      % read all the words on the line
%    while length(temp_char) > 1
   while ~isempty(temp_char)
      NumOuts = NumOuts + 1;
      [OutList{NumOuts,1}, temp_char] = strtok(temp_char,delims);
   end


end %while not END of FILE

fclose(fid);


%------------------------------------------------------------------------
-----
% Set number of DOFs
%------------------------------------------------------------------------
-----
if (NumBl == 2)
    NDOF = 22;
```

```
elseif (NumBl == 3 )
    NDOF = 24;
else
    disp ('NumBl must be 2 or 3')
end

%------------------------------------------------------------------------
-----
% Set DOF indices
%------------------------------------------------------------------------
-----
DOF_Sg   =  1;                                      % DOF index for
platform surge.
DOF_Sw   =  2;                                      % DOF index for
platform sway.
DOF_Hv   =  3;                                      % DOF index for
platform heave.
DOF_R    =  4;                                      % DOF index for
platform roll.
DOF_P    =  5;                                      % DOF index for
platform pitch.
DOF_Y    =  6;                                      % DOF index for
platform yaw.
DOF_TFA1 =  7;                                      % DOF index for
1st tower fore-aft mode.
DOF_TSS1 =  8;                                      % DOF index for
1st tower side-to-side mode.
DOF_TFA2 =  9;                                      % DOF index for
2nd tower fore-aft mode.
DOF_TSS2 = 10;                                      % DOF index for
2nd tower side-to-side mode.
DOF_Yaw  = 11;                                      % DOF index for
nacelle-yaw.
DOF_RFrl = 12;                                      % DOF index for
rotor-furl.
DOF_GeAz = 13;                                      % DOF index for
the generator azimuth.
DOF_DrTr = 14;                                      % DOF index for
drivetrain rotational-flexibility.
DOF_TFrl = 15;                                      % DOF index for
tail-furl.
DOF_BE   = 17 + 3*(0:(NumBl-1))';                   %        1st blade
edge mode--DOFs 17, 20, and 23 for blade 1, 2, and 3, respectively
DOF_BF   = [DOF_BE-1 DOF_BE+1];                     % col 1: 1st blade
flap mode--DOFs 16, 19, and 22 for blade 1, 2, and 3, respectively
                                                    % col 2: 2nd blade
flap mode--DOFs 18, 21, and 24 for blade 1, 2, and 3, respectively
DOF_Teet = 22;                                      % DOF index for
rotor-teeter.

%------------------------------------------------------------------------
-----
% Create initial condition arrays in rad, rad/s
%------------------------------------------------------------------------
-----
q_init    = zeros(1,NDOF);
qdot_init = zeros(1,NDOF);
```

```
%------------------------------------------------------------------------
-----
% Set all initial conditions except initial blade and tower
displacements,
% which are very complicated equations.
%------------------------------------------------------------------------
-----
if (NumBl == 2)
    q_init(DOF_Teet)=TeetDefl*pi/180;
end
if ( strcmpi( Furling(1), 't') )
    q_init(DOF_RFrl) = RotFurl*pi/180;
    q_init(DOF_TFrl) = TailFurl*pi/180;
end
if ((PtfmModel == 1) || (PtfmModel == 2) || (PtfmModel == 3))
    q_init(DOF_Sg  ) = PtfmSurge;
    q_init(DOF_Sw  ) = PtfmSway;
    q_init(DOF_Hv  ) = PtfmHeave;
    q_init(DOF_R   ) = PtfmRoll*pi/180;
    q_init(DOF_P   ) = PtfmPitch*pi/180;
    q_init(DOF_Y   ) = PtfmYaw*pi/180;
end

q_init(DOF_Yaw )=NacYaw*pi/180;                    % convert from deg
to radians
Azim_Initial = rem(Azimuth - AzimB1Up + 270.0 + 360.0, 360); %
Internal position of blade 1.
q_init(DOF_GeAz)=Azim_Initial*pi/180;

qdot_init(DOF_GeAz)=RotSpeed*pi/30;

NacYaw   = q_init(DOF_Yaw );

% RotSpeed = RotSpeed*pi/30;                       % convert from rpm
to rad/s
% HubHt = TowerHt + 0.5*HubRad;                    % calculate hub
height

Initialized = 1;                                   % Tells S-function
if this script ran prior to simulation.  0 = no.

%------------------------------------------------------------------------
-----
% clear variables not needed anymore
%------------------------------------------------------------------------
-----
clear temp_char fid iLine delims
clear AzimB1Up Azim_Initial FTitle
clear PtfmModel PtfmFile PtfmSurge PtfmSway PtfmHeave PtfmRoll
PtfmPitch PtfmYaw
clear Furling FurlFile RotFurl TailFurl fid2
clear OoPDefl IPDefl TeetDefl Azimuth TTDspFA TTDspSS RotSpeed %NacYaw
clear TowerHt HubRad



%========================================================================
========
```

```
sim('OpenLoop');
% orden, devolverá un 0 si esta todo correcto.

%Clasificar archivos INPUT FST creados
rutaArchivoFST='C:\Users\Juan\Desktop\FAST curva viento\Input FST
creados';
movefile(NombreArchivoINP_OUT,rutaArchivoFST);

%Clasificar archivos creados al ejecutar FAST
rutaArchivoOUT='C:\Users\Juan\Desktop\FAST curva viento\Archivos
Salida';
movefile(NombreArchivoOUT,rutaArchivoOUT);

rutaArchivosNoUsados='C:\Users\Juan\Desktop\FAST curva viento\Archivos
no usados';
movefile(NombreArchivosNoUsados1,rutaArchivosNoUsados);
movefile(NombreArchivosNoUsados2,rutaArchivosNoUsados);



end
```

## Anexo F. Adecuar y preparar en series 24h las series P-v

```
%Los resultados de FAST se pasan manualmente a Excel, se borra la
%información que no interesa (el tiempo de cada uno, que se repite) y
se
%unifican en un único archivo para poder hacer una gráfica con el

nSeries=9;
name1='excelTablaViento';
name2='_SFunc.xlsx';
name4='excel';
r=1;
name3=num2str(r);
NombreExcelEntrada = strcat(name1,name3,name2);
ExcelSalida=xlsread(NombreExcelEntrada);
ExcelSalida(:,1)=[];
Junto=ExcelSalida;


for r=1:nSeries
    name3=num2str(r);
    NombreExcelEntrada = strcat(name1,name3,name2);

    %Eliminar columna de los tiempos.
    ExcelSalida=xlsread(NombreExcelEntrada);
    ExcelSalida(:,1)=[];

    %Eliminar los primeros 1660 datos del solapamiento a partir de la
    %segunda serie.
    nLineasBorradas=1660;
    if r==1
       Junto=ExcelSalida;
    else
       %Borrar las filas del solapamiento.
       for i = 1:nLineasBorradas
           ExcelSalida(1,:)=[ ];
       end
       %Sumar los vectores en vertical.
       Junto=[Junto;ExcelSalida];
    end

end

%Crear el archivo Excel con los vectores ya unificados.
nameExcel='excelUnido.xlsx';
status=xlswrite(nameExcel,Junto);



%Crea uno menos de los que debería, en lugar de 887431 crea 887430, no
pasa nada. El tiempo es 100079 pasos, para 9999 segundos = 0,0999111
segundos/paso (para una sola simulación), dando 88664,08 segundos


%Un día son 24x3600=86400 segundos, es decir 86400/0,0999111= 864769
datos
%887430-864769=22661 datos hay que eliminar para redondear al día, se
eliminan en el propio Excel
```

```
%SE EJECUTA POR SEPARADO, LAS VARIABLES NO CONCUERDAN, PORQUE HAY QUE
ASEGURARSE DE QUE EL NÚMERO DE FILAS CREADAS PARA t ES IGUAL A LAS DEL
EXCEL:
Para incluir una columna con el tiempo transcurrido de simulación
real,
%una vez aplicado el solapamiento, se crea un vector con el mismo paso
y nº
%de filas, y se añade a las otras dos columnas.

name='excelUnido'
t=[0:0.099911:88664];
ExcelSalida=xlsread(name);

%Sumar vectores en horizontal.
JuntoTiempo=[t' Junto];

ExcelTodo='excelUnidoTodo.xlsx';
status=xlswrite(ExcelTodo,JuntoTiempo);
```

## Anexo G. Programa TurbSim

Es necesario ejecutar TurbSim a través de MS2 y funciona de una manera similar a FAST: no hay interfaz de trabajo y se controla esencialmente a mediante un archivo principal de entrada (también existen otros secundarios) [10]. Es posible crear un archivo de entrada desde cero o utilizar uno de los ejemplos del programa y variarlo hasta que se adecue a las necesidades del usuario (Anexo H). Algunas de las variables son:

- El número semilla para la generación del viento turbulento aleatorio: si se mantiene todo lo demás constante, a igual semilla, igual viento.
- Los formatos en los que quieres la tabla de viento generada (hay hasta 7 posibilidades). El único formato compatible con FAST es ".HH"
- El número de puntos en el mallado del campo de viento generado, en "Z" e "Y".
- El paso de iteración.
- El tiempo de simulación.
- La altura a la que estará el buje del aerogenerador que usara las series de velocidad de viento.
- La altura y anchura del mallado de viento generado, dependerá del diámetro del aerogenerador.
- Velocidad media del viento generado.
- Tipo de turbulencia e intensidad de la misma.

Cuando TurbSim es ejecutado se obtiene una tabla de viento que se podrá usar directamente como entrada de viento de FAST:

```
! This hub-height wind-speed file was generated by TurbSim (v1.06.00,
! 21-Sep-2012) on 14-Jan-2016 at 23:21:13.
!
! The requested statistics for this data were:
!    Mean Total Wind Speed =    6.500 m/s
!    Turbulence Intensity  =   19.338%
!
!   Time   HorSpd   WndDir   VerSpd   HorShr   VerShr   LnVShr   GstSpd
!  (sec)    (m/s)    (deg)    (m/s)      (-)      (-)      (-)    (m/s)
   0.000     5.78   -22.44     0.03    0.000    0.200    0.000     0.00
   0.050     5.85   -22.70     0.05    0.000    0.200    0.000     0.00
   0.100     5.87   -22.55     0.09    0.000    0.200    0.000     0.00
   0.150     5.60   -20.11     0.09    0.000    0.200    0.000     0.00
   0.200     5.49   -20.52    -0.04    0.000    0.200    0.000     0.00
   0.250     5.29   -20.05     0.07    0.000    0.200    0.000     0.00
   0.300     5.10   -18.25    -0.19    0.000    0.200    0.000     0.00
   0.350     4.94   -19.47    -0.03    0.000    0.200    0.000     0.00
   0.400     5.02   -18.48     0.06    0.000    0.200    0.000     0.00
   0.450     4.91   -16.49    -0.38    0.000    0.200    0.000     0.00
   0.500     4.65   -12.47    -0.22    0.000    0.200    0.000     0.00
   0.550     4.56   -11.25     0.00    0.000    0.200    0.000     0.00
   0.600     4.52   -11.16    -0.09    0.000    0.200    0.000     0.00
   0.650     4.59   -12.46    -0.20    0.000    0.200    0.000     0.00
   0.700     4.83   -16.31    -0.37    0.000    0.200    0.000     0.00
   0.750     4.87   -14.68    -0.23    0.000    0.200    0.000     0.00
   0.800     4.66   -12.02    -0.02    0.000    0.200    0.000     0.00
   0.850     4.53   -15.83     0.04    0.000    0.200    0.000     0.00
   0.900     4.43   -18.59    -0.10    0.000    0.200    0.000     0.00
   0.950     4.52   -23.06     0.09    0.000    0.200    0.000     0.00
   1.000     4.40   -25.52    -0.15    0.000    0.200    0.000     0.00
   1.050     4.21   -23.66    -0.24    0.000    0.200    0.000     0.00
   1.100     4.23   -27.08     0.18    0.000    0.200    0.000     0.00
   1.150     4.61   -22.44     0.45    0.000    0.200    0.000     0.00
   1.200     4.62   -23.48     0.17    0.000    0.200    0.000     0.00
   1.250     4.54   -22.91    -0.08    0.000    0.200    0.000     0.00
   1.300     4.61   -21.49    -0.04    0.000    0.200    0.000     0.00
   1.350     4.64   -21.44     0.38    0.000    0.200    0.000     0.00
   1.400     4.54   -21.25     0.63    0.000    0.200    0.000     0.00
   1.450     4.51   -20.87     0.53    0.000    0.200    0.000     0.00
   1.500     4.44   -23.68     0.48    0.000    0.200    0.000     0.00
   1.550     4.41   -22.54     0.64    0.000    0.200    0.000     0.00
   1.600     4.23   -28.69     0.50    0.000    0.200    0.000     0.00
   1.650     4.19   -30.89     0.42    0.000    0.200    0.000     0.00
   1.700     4.37   -30.61     0.72    0.000    0.200    0.000     0.00
   1.750     4.58   -33.50     1.05    0.000    0.200    0.000     0.00
   1.800     4.72   -29.55     0.79    0.000    0.200    0.000     0.00
   1.850     4.87   -32.00     0.59    0.000    0.200    0.000     0.00
   1.900     4.58   -32.17     0.63    0.000    0.200    0.000     0.00
   1.950     4.38   -29.87     0.74    0.000    0.200    0.000     0.00
   2.000     4.28   -29.84     0.65    0.000    0.200    0.000     0.00
   2.050     4.35   -32.98     0.63    0.000    0.200    0.000     0.00
```

## Anexo H. Archivo principal de entrada TurbSim

```
TurbSim Input File. Valid for TurbSim v1.06.00, 21-Sep-2012

---------Runtime Options-----------------------------------
2318573             RandSeed1       - First random seed  (-2147483648 to
                                      2147483647)
RANLUX              RandSeed2       - Second random seed (-2147483648 to
                                      2147483647) for intrinsic pRNG, or an
                                      alternative pRNG: "RanLux" or "RNSNLW"
False               WrBHHTP         - Output hub-height turbulence parameters
in
                                      binary form?  (Generates RootName.bin)
False               WrFHHTP         - Output hub-height turbulence parameters
in
                                      formatted form?  (Generates
RootName.dat)
False               WrADHH          - Output hub-height time-series data in
                                      AeroDyn form?  (Generates RootName.hh)
False               WrADFF          - Output full-field time-series data in
                                      TurbSim/AeroDyn form? (Generates
                                      Rootname.bts)
True                WrBLFF          - Output full-field time-series data in
                                      BLADED/AeroDyn form?  (Generates
                                      RootName.wnd)
False               WrADTWR         - Output tower time-series data?
(Generates
                                      RootName.twr)
False               WrFMTFF         - Output full-field time-series data in
                                      formatted (readable) form?  (Generates
                                      RootName.u, RootName.v, RootName.w)
True                WrACT           - Output coherent turbulence time steps in
                                      AeroDyn form? (Generates RootName.cts)
True                Clockwise       - Clockwise rotation looking downwind?
(used
                                      only for full-field binary files - not
                                      necessary for AeroDyn)
 0                  ScaleIEC        - Scale IEC turbulence models to exact
target
                                      standard deviation? [0=no additional
                                      scaling; 1=use hub scale uniformly;
2=use
                                      individual scales]

--------Turbine/Model Specifications-----------------------
13                  NumGrid_Z       - Vertical grid-point matrix dimension
13                  NumGrid_Y       - Horizontal grid-point matrix dimension
0.05                TimeStep        - Time step [seconds]
600                 AnalysisTime    - Length of analysis time series [seconds]
                                     (program will add time if necessary:
                                      AnalysisTime = MAX(AnalysisTime,
                                      UsableTime+GridWidth/MeanHHWS) )
40                  UsableTime      - Usable length of output time series
                                     [seconds] (program will add
                                      GridWidth/MeanHHWS seconds)
84.2876             HubHt           - Hub height [m] (should be >
0.5*GridHeight)
80.00               GridHeight      - Grid height [m]
80.00               GridWidth       - Grid width [m] (should be >=
                                      2*(RotorRadius+ShaftLength))
0                   VFlowAng        - Vertical mean flow (uptilt) angle
[degrees]
0                   HFlowAng        - Horizontal mean flow (skew) angle
[degrees]

--------Meteorological Boundary Conditions-------------------
"SMOOTH"            TurbModel       - Turbulence model ("IECKAI"=Kaimal,
                                     "IECVKM"=von Karman, "GP_LLJ", "NWTCUP",
```

```
                                    "SMOOTH", "WF_UPW", "WF_07D", "WF_14D",
                                    "TIDAL", or "NONE")
"1-ED3"             IECstandard     - Number of IEC 61400-x standard (x=1,2,
or 3
                                     with optional 61400-1 edition number
(i.e.
                                    "1-Ed2") )
"A"                 IECturbc        - IEC turbulence characteristic ("A", "B",
                                     "C" or the turbulence intensity in
percent)
                                    ("KHTEST" option with NWTCUP model, not
                                     used for other models)
"NTM"               IEC_WindType    - IEC turbulence type ("NTM"=normal,
                                    "xETM"=extreme turbulence,
"xEWM1"=extreme
                                     1-year wind, "xEWM50"=extreme 50-year
wind,
                                     where x=wind turbine class 1, 2, or 3)
default             ETMc            - IEC Extreme Turbulence Model "c"
parameter
                                     [m/s]
default             WindProfileType - Wind profile type
                                    ("JET";"LOG"=logarithmic;"PL"=power
                                     law;"H2L"=Log law for TIDAL spectral
                                     model;"IEC"=PL on rotor disk, LOG
                                     elsewhere; or "default")
84.2876             RefHt           - Height of the reference wind speed [m]
18.2                URef            - Mean (total) wind speed at the reference
                                     height [m/s] (or "default" for JET wind
                                     profile)
default             ZJetMax         - Jet height [m] (used only for JET wind
                                     profile, valid 70-490 m)
default             PLExp           - Power law exponent [-] (or "default")
default             Z0              - Surface roughness length [m] (or
"default")

--------Non-IEC Meteorological Boundary Conditions------------
default             Latitude        - Site latitude [degrees] (or "default")
0.05                RICH_NO         - Gradient Richardson number
default             UStar           - Friction or shear velocity [m/s] (or
                                     "default")
default             ZI              - Mixing layer depth [m] (or "default")
default             PC_UW           - Hub mean u'w' Reynolds stress (or
                                     "default")
default             PC_UV           - Hub mean u'v' Reynolds stress (or
                                     "default")
default             PC_VW           - Hub mean v'w' Reynolds stress (or
                                     "default")
default             IncDec1         - u-component coherence parameters (e.g.
                                     "10.0  0.3e-3" in quotes) (or "default")
default             IncDec2         - v-component coherence parameters (e.g.
                                     "10.0  0.3e-3" in quotes) (or "default")
default             IncDec3         - w-component coherence parameters (e.g.
                                     "10.0  0.3e-3" in quotes) (or "default")
default             CohExp          - Coherence exponent (or "default")

--------Coherent Turbulence Scaling Parameters-------------------
"C:\Users\Juan\Desktop\turbSim\Test\EventData"  CTEventPath     - Name of the
                                                                  path where event data
                                                                  files are located
"Random"            CTEventFile     - Type of event files ("LES", "DNS", or
                                    "RANDOM")
true                Randomize       - Randomize the disturbance scale and
                                     locations? (true/false)
 1.0                DistScl         - Disturbance scale (ratio of wave height
to
                                     rotor disk). (Ignored when Randomize =
                                     true.)
```

```
 0.5                CTLy            - Fractional location of tower centerline
                                      from right (looking downwind) to left
side
                                      of the dataset. (Ignored when Randomize
=
                                      true.)
 0.5                CTLz            - Fractional location of hub height from
the
                                      bottom of the dataset. (Ignored when
                                      Randomize = true.)
30.0                CTStartTime     - Minimum start time for coherent
structures
                                      in RootName.cts [seconds]

================================================
NOTE: Do not add or remove any lines in this file!
================================================
```

## Anexo I. Cambio semilla TurbSim, ejecución y clasificación

```
%Cambio semilla INP, ejecución TurbSim y guardado de output (moverlo)

%Se desarrolla el bucle para las distintas velocidades del viento, de
5 a 15, de 0,5 m/s
%de paso, total 21 velocidades distintas.
numVelocidades=21;
for r=1:numVelocidades

%Crear carpeta en FAST para cada velocidad.
textMedio1=num2str(r/2+4.5);
mkdir('C:\Users\Juan\Desktop\Fast Turbsim
Final\Fast\Wind\WP_Baseline',textMedio1)
%crear carpeta para ordenar archivos creados
mkdir('C:\Users\Juan\Desktop\Fast Turbsim Final\Turbsim\Archivos
usados',textMedio1)

%Bucle para las distintas series de cada velocidad, 60 para cada
serie.
numSeries=60;
for i=1:numSeries
replaceLine1 = 4;
replaceLine2 = 5;
replaceLineV = 37;
numLines = 66;
rand1 = randi([-2147483647,2147483646]);
rand2 = randi([-2147483647,2147483646]);
%seed1='%rand1';
%seed2='%rand2';
fileID = fopen('CurvaDinamica.inp','r');
mydata = cell(1, numLines);
for k = 1:numLines
   mydata{k} = fgetl(fileID);
end
fclose(fileID);

mydata{replaceLine1} = rand1;
mydata{replaceLine2} = rand2;
viento=(r/2+4.5);
mydata{replaceLineV} = viento;

% Para que cambie de nombre el archivo de salida en función de la
serie y de la velocidad del viento.
text='CurvaDinamica';
text2=' CurvaDinamica';% necesita el espacio inicial para la orden de
MS2
textMedio2='caso';
textMedio3=num2str(i);
textFinal='.inp';
textOutput='.hh';
textSobras='.sum';
NombreArchivoINP =
strcat(text,textMedio1,textMedio2,textMedio3,textFinal);
NombreArchivoINPsimbolo =
strcat(text2,textMedio1,textMedio2,textMedio3,textFinal);
NombreArchivoOUT =
strcat(text,textMedio1,textMedio2,textMedio3,textOutput);
```

```
NombreArchivoSobras =
strcat(text,textMedio1,textMedio2,textMedio3,textSobras);

fileID = fopen(NombreArchivoINP,'w');
fprintf(fileID,'%s\n',mydata{1:replaceLine1-1});
fprintf(fileID,'%11d      Seed1\n',mydata{replaceLine1});
fprintf(fileID,'%11d      Seed2\n',mydata{replaceLine2});
fprintf(fileID,'%s\n',mydata{replaceLine2+1:replaceLineV-1});
fprintf(fileID,'%4g            URef            - Mean (total) wind
speed at the reference height [m/s]\n',mydata{replaceLineV});
fprintf(fileID,'%s\n',mydata{replaceLineV+1:numLines-1});
fprintf(fileID,'%s',mydata{numLines});
fclose(fileID);

%Llamada para ejecutar en ms dos el archivo.
ejecucion='turbsim';
nombreINP=NombreArchivoINPsimbolo;
MS2 = strcat(ejecucion,nombreINP);
command=MS2;
status = dos(command);

% Sirve para comprobar si se ha realizado bien la
% orden, devolverá un 0 si esta todo correcto.

%Escribir ruta carpeta WIND en FAST y mover el archivo de salida que
%acaba de ser creado.
ruta='C:\Users\Juan\Desktop\Fast Turbsim
Final\Fast\Wind\WP_Baseline\';
carpetaVviento=textMedio1;
rutaCarpeta= strcat(ruta,carpetaVviento);
movefile(NombreArchivoOUT,rutaCarpeta);

%Mover archivos usados a su carpeta correspondiente, los ordena.
rutaUsados='C:\Users\Juan\Desktop\Fast Turbsim Final\Turbsim\Archivos
usados\';
rutaCarpetaUsados= strcat(rutaUsados,carpetaVviento);
movefile(NombreArchivoINP,rutaCarpetaUsados);
movefile(NombreArchivoSobras,rutaCarpetaUsados);

end
end
```

## Anexo J. Datos de entrada TurbSim, 1260 series salida P-v

```
%Programa para ejecutar FAST para cada una de las series y
posteriormente clasificar en carpetas los archivos creados.


numVelocidades=21;
for r=1:numVelocidades

%Crear carpeta en FAST para cada velocidad.
textMedio1=num2str(r/2+4.5);
mkdir('C:\Users\Juan\Desktop\Fast Turbsim Final\Fast\Archivos
Salida',textMedio1)
%Crear carpeta para ordenar archivos creados que no uso.
mkdir('C:\Users\Juan\Desktop\Fast Turbsim Final\Fast\Archivos no
usados',textMedio1)
%Crear carpeta para ordenar archivos de entrada FST generados.
mkdir('C:\Users\Juan\Desktop\Fast Turbsim Final\Fast\FST Input
creados',textMedio1)


numSeries=60;
for i=1:numSeries
%Abrir el input general de fast y lo guardamos con otro nombre, para
%obtener una salida con nombre distinto para cada caso.
numLines = 198;
name1='CurvaDinamica';
name2=num2str(i);
name3='.fst';
name4='_SFunc.out';
name5=textMedio1;
name6='caso';
name7='_SFunc.opt';
name8='_SFunc.fsm';
NombreArchivoINP = strcat(name1,name3);
NombreArchivoINP_OUT = strcat(name1,name5,name6,name2,name3);
NombreArchivoOUT = strcat(name1,name5,name6,name2,name4);
NombreArchivosNoUsados1 = strcat(name1,name5,name6,name2,name7);
NombreArchivosNoUsados2 = strcat(name1,name5,name6,name2,name8);
fileID = fopen(NombreArchivoINP,'r');
mydata = cell(1, numLines);
for m = 1:numLines
   mydata{m} = fgetl(fileID);
end
fclose(fileID);

fileID = fopen(NombreArchivoINP_OUT,'w');
fprintf(fileID,'%s\n',mydata{1:numLines-1});
fprintf(fileID,'%s',mydata{numLines});
fclose(fileID);



%Abrir el archivo de entrada _AD, para modificar la entrada del viento
y se guarda igual, se irá actualizando cada serie nueva, así se evita
%tener mil que no aportan nada.
replaceLine1 = 10;
numLinesAD = 39;
```

```
entradaAD='CurvaDinamica_AD.ipt';
fileAD = fopen(entradaAD,'r');
mydataAD = cell(1, numLinesAD);
for k = 1:numLinesAD
   mydataAD{k} = fgetl(fileAD);
end
fclose(fileAD);

% Hace que cambie de nombre el archivo de entrada de viento.
text='"Wind/WP_Baseline/';
textMedio2='/';
textMedio3='CurvaDinamica';
textMedio4='caso';
textMedio5=num2str(i);
textFinInput='.hh"        WindFile - Name of file containing wind data
(quoted string)';

rutaViento =
strcat(text,textMedio1,textMedio2,textMedio3,textMedio1,textMedio4,tex
tMedio5,textFinInput);

mydataAD{replaceLine1} = rutaViento;




fileAD = fopen(entradaAD,'w');
fprintf(fileAD,'%s\n',mydataAD{1:numLinesAD-1});
fprintf(fileAD,'%s',mydataAD{numLinesAD});
fclose(fileAD);

%Para ejecutar Fast en Simulink, hay que saltarse la llamada de
Simsetup que solo es un paso intermedio hacia el comandoRead-Fast,
%Que se procede a ejecutar ahora:
%------------------------------------------------------------------------
-----
% This script reads FAST *.fst files and creates Matlab workspace
variables
% required to run the FAST s-function in Simulink.
% Developed at NREL's National Wind Technology Center
%     modified 10-Mar-2010 by B. Jonkman, NREL
%------------------------------------------------------------------------
-----

%------------------------------------------------------------------------
-----
%open the FAST input file
%------------------------------------------------------------------------
-----
input_fast=NombreArchivoINP_OUT;
fid=fopen(input_fast,'r');
if (fid == -1)
    disp(['Input file:  ',input_fast,' could not be opened.']);
    return
end

for iLine=1:2;                                        % skip the first
two lines
    fgetl(fid);
```

```
end
disp( fgetl(fid) )                                      % line 3: input
file description

for iLine=1:5;                                          % skip 5 lines
(through AnalMode)
    fgetl(fid);
end

% read simulation parameters
NumBl= fscanf(fid,'%i',1); fgetl(fid);                  % the S-function
needs this parameter for initialization
TMax = fscanf(fid,'%g',1); fgetl(fid);                  % can be used in
Simulation Parameters menu
DT   = fscanf(fid,'%g',1); fgetl(fid);                  % can be used in
Simulation Parameters menu

for iLine=1:34                                          % skip 34 lines
(12-45)
    fgetl(fid);                                         % (first part of
Turbine control)
end

BlPitch = zeros(3,1);
for iLine=1:3                                           % read lines 46-48
    BlPitch(iLine) = fscanf(fid,'%g',1); fgetl(fid); % initial blade
pitch
end

for iLine=1:20                                          % skip 20 lines
(49-68)
    fgetl(fid);                                         % (Turbine control
through feature switches)
end

% read initial conditions section
OoPDefl  = fscanf(fid,'%g',1); fgetl(fid);              % read line 69,
initial OoPDefl
IPDefl   = fscanf(fid,'%g',1); fgetl(fid);              % read line 70,
initial IPDefl
TeetDefl = fscanf(fid,'%g',1); fgetl(fid);              % read line 71,
initial TeetDefl
Azimuth  = fscanf(fid,'%g',1); fgetl(fid);              % read line 72,
initial Azimuth
RotSpeed = fscanf(fid,'%g',1); fgetl(fid);              % read line 73,
initial RotSpeed
NacYaw   = fscanf(fid,'%g',1); fgetl(fid);              % read line 74,
initial NacYaw
TTDspFA  = fscanf(fid,'%g',1); fgetl(fid);              % read line 75,
initial TTDspFA
TTDspSS  = fscanf(fid,'%g',1); fgetl(fid);              % read line 76,
initial TTDspSS

    %Turbine Configuration section
for iLine=1:2                                           % skip 2 lines
(77-78)
    fgetl(fid);                                         % (Turbine
configuration)
end
```

```
HubRad  = fscanf(fid,'%g',1); fgetl(fid);           % read line 79,
HubRad
for iLine=1:7                                       % skip 7 lines
(80-86)
    fgetl(fid);                                     % (Turbine
configuration parameters)
end
TowerHt = fscanf(fid,'%g',1); fgetl(fid);           % read line 87,
TowerHt
for iLine=1:7                                       % skip 7 lines
(88-94)
    fgetl(fid);                                     % (Turbine
configuration parameters)
end
AzimB1Up = fscanf(fid,'%g',1); fgetl(fid);          % read line 95,
Azimuth when Blade 1 is up

    % MASS AND INERTIA section
for iLine=1:10                                      % skip 10 lines
(96-105)
    fgetl (fid);                                    % (MASS AND
INERTIA parameters)
end

    % DRIVETRAIN
for iLine=1:2                                       % skip 2 lines
(106-107)
    fgetl (fid);                                    % (DRIVETRAIN
parameters)
end
GenEff  = fscanf(fid,'%g',1); fgetl(fid);           % line 108 (used
to initialize
GBRatio = fscanf(fid,'%g',1); fgetl(fid);           % line 109
for iLine=1:6                                       % skip 6 lines
(110-115)
    fgetl (fid);                                    % (DRIVETRAIN
parameters)
end

    % SIMPLE INDUCTION GENERATOR
for iLine=1:5                                       % skip 5 lines
(106-120)
    fgetl (fid);                                    % (SIMPLE
INDUCTION GENERATOR parameters)
end

    % THEVENIN-EQUIVALENT INDUCTION GENERATOR
for iLine=1:9                                       % skip 5 lines
(121-129)
    fgetl (fid);                                    % (THEVENIN-
EQUIVALENT INDUCTION GENERATOR)
end
    % PLATFORM MODEL
fgetl(fid);                                         % heading, line
130
PtfmModel = fscanf(fid,'%g',1); fgetl(fid);         % read line 131,
PtfmModel
temp_char = fgetl(fid);                             % read line 132,
PtfmFile line
```

```
%------------------------------------------------------------------------
--
% If we have a platform model, read initial platform displacements:
% Open the platform file (get the name of the file first)
%------------------------------------------------------------------------
--
if ( PtfmModel >= 1 && PtfmModel <= 3 )
   temp_char = strtrim(temp_char);                  % remove leading
(and trailing spaces)
   if strfind( '"''', temp_char(1) )                % the string
starts with quotes
      PtfmFile = strtok( temp_char(2:end), temp_char(1) ); %read to
the end of the quote
   else
      PtfmFile = strtok( temp_char );               % read to the
first white space
   end

   fid2=fopen(PtfmFile);
   if (fid2 == -1)
      disp(['Platform file:  ',PtfmFile,' could not be opened.']);
      return
   end
   for iLine=1:11;
      fgetl(fid2);                                  % skip first 11
lines of PtfmFile
   end
   PtfmSurge = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
surge
   PtfmSway  = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
sway
   PtfmHeave = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
heave
   PtfmRoll  = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
roll
   PtfmPitch = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
pitch
   PtfmYaw   = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial platform
yaw
   fclose(fid2);
end

for iLine=1:8                                       % skip 8 lines
(133-140) of the FAST file
   fgetl(fid);                                      % (Nacelle-Yaw to
beginning of Furling)
end

Furling   = strtrim( fgetl(fid) );                  % read line 141,
Furling
temp_char = fgetl(fid);                             % read line 142,
FurlFile line

%------------------------------------------------------------------------
--
% If furling, read initial rotor and tail furl values:
% Open the Furling file (get the name of the file first)
%------------------------------------------------------------------------
--
if ( strcmpi(Furling(1), 't') )
```

```
   temp_char = strtrim(temp_char);                  % remove leading
(and trailing spaces)
   if strfind( '"''', temp_char(1) )                % the string
starts with quotes
      FurlFile = strtok( temp_char(2:end), temp_char(1) ); % read to
the end of the quote
   else
      FurlFile = strtok( temp_char );              % read to the
first whitespace
   end

   fid2=fopen(FurlFile);
   if (fid2 == -1)
      disp(['Furling file:  ',FurlFile,' could not be opened.']);
      return
   end
   for iLine=1:7;
      fgetl(fid2);                                 % skip first 7
lines of FurlFile
   end
   RotFurl  = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial rotor
furl
   TailFurl = fscanf(fid2,'%g',1); fgetl(fid2);    % Initial tail
furl
   fclose(fid2);
end

for iLine=1:42                                      % skip 42 lines
(143-184) of the FAST file
    fgetl(fid);                                     % (Rotor-Teeter to
beginning of OutList)
end

%------------------------------------------------------------------------
-----
% get the FAST outputs (OutList) and close the FAST input file
%------------------------------------------------------------------------
-----
%temp_char = strtrim( fgetl(fid) );
NumOuts   = 0;
OutList   = {};                                     % clear this from
subsequent runs
delims    = [' ,;''"' char(9)];                     % allowable
delimiters

while ~feof(fid)

      % get the next line
   temp_char = strtrim( fgetl(fid) );

   if length(temp_char) < 3     %FAST doesn't have 2-character
variables;

            % skip any blank lines
      if isempty(temp_char)
         continue;
      end

   elseif strcmpi( temp_char(1:3), 'end' )
```

```
            % we've reached the end
         break;
   end

      % get the equivalent "line" that Fortran would read
   if strfind( '"''', temp_char(1) )               % the string
starts with quotes
      temp_char = strtok( temp_char(2:end), temp_char(1) ); % read to
the end of the quotes
   else
      temp_char = strtok( temp_char );             % read to the
first whitespace
   end

      % read all the words on the line
%    while length(temp_char) > 1
   while ~isempty(temp_char)
      NumOuts = NumOuts + 1;
      [OutList{NumOuts,1}, temp_char] = strtok(temp_char,delims);
   end


end %while not END of FILE

fclose(fid);


%------------------------------------------------------------------------
-----
% Set number of DOFs
%------------------------------------------------------------------------
-----
if (NumBl == 2)
    NDOF = 22;
elseif (NumBl == 3 )
    NDOF = 24;
else
    disp ('NumBl must be 2 or 3')
end

%------------------------------------------------------------------------
-----
% Set DOF indices
%------------------------------------------------------------------------
-----
DOF_Sg   =  1;                                     % DOF index for
platform surge.
DOF_Sw   =  2;                                     % DOF index for
platform sway.
DOF_Hv   =  3;                                     % DOF index for
platform heave.
DOF_R    =  4;                                     % DOF index for
platform roll.
DOF_P    =  5;                                     % DOF index for
platform pitch.
DOF_Y    =  6;                                     % DOF index for
platform yaw.
DOF_TFA1 =  7;                                     % DOF index for
1st tower fore-aft mode.
```

```
DOF_TSS1 =  8;                                          % DOF index for
1st tower side-to-side mode.
DOF_TFA2 =  9;                                          % DOF index for
2nd tower fore-aft mode.
DOF_TSS2 = 10;                                          % DOF index for
2nd tower side-to-side mode.
DOF_Yaw  = 11;                                          % DOF index for
nacelle-yaw.
DOF_RFrl = 12;                                          % DOF index for
rotor-furl.
DOF_GeAz = 13;                                          % DOF index for
the generator azimuth.
DOF_DrTr = 14;                                          % DOF index for
drivetrain rotational-flexibility.
DOF_TFrl = 15;                                          % DOF index for
tail-furl.
DOF_BE   = 17 + 3*(0:(NumBl-1))';                       %        1st blade
edge mode--DOFs 17, 20, and 23 for blade 1, 2, and 3, respectively
DOF_BF   = [DOF_BE-1 DOF_BE+1];                         % col 1: 1st blade
flap mode--DOFs 16, 19, and 22 for blade 1, 2, and 3, respectively
                                                        % col 2: 2nd blade
flap mode--DOFs 18, 21, and 24 for blade 1, 2, and 3, respectively
DOF_Teet = 22;                                          % DOF index for
rotor-teeter.

%------------------------------------------------------------------------
-----
% Create initial condition arrays in rad, rad/s
%------------------------------------------------------------------------
-----
q_init    = zeros(1,NDOF);
qdot_init = zeros(1,NDOF);


%------------------------------------------------------------------------
-----
% Set all initial conditions except initial blade and tower
displacements,
% which are very complicated equations.
%------------------------------------------------------------------------
-----
if (NumBl == 2)
    q_init(DOF_Teet)=TeetDefl*pi/180;
end
if ( strcmpi( Furling(1), 't') )
    q_init(DOF_RFrl) = RotFurl*pi/180;
    q_init(DOF_TFrl) = TailFurl*pi/180;
end
if ((PtfmModel == 1) || (PtfmModel == 2) || (PtfmModel == 3))
    q_init(DOF_Sg  ) = PtfmSurge;
    q_init(DOF_Sw  ) = PtfmSway;
    q_init(DOF_Hv  ) = PtfmHeave;
    q_init(DOF_R   ) = PtfmRoll*pi/180;
    q_init(DOF_P   ) = PtfmPitch*pi/180;
    q_init(DOF_Y   ) = PtfmYaw*pi/180;
end

q_init(DOF_Yaw )=NacYaw*pi/180;                         % convert from deg
to radians
```

```
Azim_Initial = rem(Azimuth - AzimB1Up + 270.0 + 360.0, 360); %
Internal position of blade 1.
q_init(DOF_GeAz)=Azim_Initial*pi/180;

qdot_init(DOF_GeAz)=RotSpeed*pi/30;

NacYaw   = q_init(DOF_Yaw );

% RotSpeed = RotSpeed*pi/30;                          % convert from rpm
to rad/s
% HubHt = TowerHt + 0.5*HubRad;                       % calculate hub
height

Initialized = 1;                                      % Tells S-function
if this script ran prior to simulation.  0 = no.

%------------------------------------------------------------------------
-----
% clear variables not needed anymore
%------------------------------------------------------------------------
-----
clear temp_char fid iLine delims
clear AzimB1Up Azim_Initial FTitle
clear PtfmModel PtfmFile PtfmSurge PtfmSway PtfmHeave PtfmRoll
PtfmPitch PtfmYaw
clear Furling FurlFile RotFurl TailFurl fid2
clear OoPDefl IPDefl TeetDefl Azimuth TTDspFA TTDspSS RotSpeed %NacYaw
clear TowerHt HubRad




%========================================================================
========

sim('OpenLoop');
% Orden, devolverá un 0 si esta todo correcto.

%Clasificar archivos INPUT FST creados.
rutaFST1='C:\Users\Juan\Desktop\Fast Turbsim Final\Fast\FST Input
creados\';
rutaArchivoFST= strcat(rutaFST1,textMedio1);
movefile(NombreArchivoINP_OUT,rutaArchivoFST);

%Clasificar archivos creados al ejecutar FAST.
rutaOUT1='C:\Users\Juan\Desktop\Fast Turbsim Final\Fast\Archivos
Salida\';
rutaArchivoOUT= strcat(rutaOUT1,textMedio1);
movefile(NombreArchivoOUT,rutaArchivoOUT);

rutaNoUsado1='C:\Users\Juan\Desktop\Fast Turbsim Final\Fast\Archivos
no usados\';
rutaArchivosNoUsados= strcat(rutaNoUsado1,textMedio1);
movefile(NombreArchivosNoUsados1,rutaArchivosNoUsados);
movefile(NombreArchivosNoUsados2,rutaArchivosNoUsados);
```

```
end
end
```