APENDICE.- *MODELADO DE LA ESTELA DE UN AEROGENERADOR EN PARQUE MARINO*

Tabla 1.- Valores de diseño y de ajuste para c

r	c Optimo	c Diseño PoliMatlab	Diferencias
30,55	0,0000	0,02	-0,02
29,02	0,8860	0,77	0,11
27,50	0,9654	1,02	-0,06
25,97	1,0295	1,06	-0,03
24,44	1,0937	1,06	0,03
22,91	1,1640	1,10	0,07
21,39	1,2434	1,18	0,06
19,86	1,3320	1,31	0,02
18,33	1,4389	1,46	-0,02
16,80	1,5581	1,60	-0,04
15,28	1,7016	1,73	-0,03
13,75	1,8727	1,87	0,01
12,22	2,0805	2,03	0,05
10,69	2,3310	2,27	0,06
9,16	2,6426	2,61	0,03
7,64	3,0336	3,06	-0,03
6,11	3,5071	3,59	-0,08
4,58	4,0448	4,08	-0,04
3,05	4,4359	4,31	0,12

Tabla 2.- Valores de diseño y de ajuste para β

r	β Optimo	β Diseño Ajuste PoliMatlab	Diferencias
30,55	-2,00	-2,02	0,02
29,02	-1,53	-1,40	-0,14
27,50	-0,96	-0,94	-0,03
25,97	-0,54	-0,55	0,01
24,44	-0,13	-0,16	0,03
22,91	0,28	0,26	0,02
21,39	0,74	0,74	0,00
19,86	1,25	1,29	-0,03
18,33	1,84	1,73	0,11
16,80	2,54	2,60	-0,06
15,28	3,36	3,39	-0,04
13,75	4,35	4,34	0,01
12,22	5,57	5,50	0,07
10,69	7,10	7,01	0,10
9,16	9,08	9,01	0,07
7,64	11,71	11,75	-0,04
6,11	15,34	15,52	-0,18
4,58	20,54	20,70	-0,17
3,05	28,21	27,78	0,43
1,53	37,17	37,34	-0,16

Tabla 3.- Valores de diseño y de ajuste para CL

Alfa	CL Diseño NACA 4418	CL Diseño Poli-nomio	Diferencias
0	0,3833	0,0000	0,38
4	0,7944	0,7795	0,01
6	1,0000	1,0139	-0,01
8	1,1750	1,2117	-0,04
10	1,3500	1,3544	0,00
12	1,4500	1,4375	0,01
13	1,5000	1,4579	0,04
14	1,5000	1,4656	0,03
16	1,4333	1,4490	-0,02
20	1,3000	1,3327	-0,03
24	1,1667	1,1868	-0,02
26	1,1000	1,1258	-0,03
30	1,1000	1,0503	0,05
40	1,1000	1,0809	0,02
50	1,0000	1,0440	-0,04
60	0,8300	0,7753	0,05
70	0,5533	0,5511	0,00
80	0,2767	0,2279	0,05
90	0,0000	-0,0664	0,07

Tabla 4.- Valores de diseño y de ajuste para CD

Alfa	CD Diseño NACA 4418	CD Diseño Polinomio	Diferencias
0	0,0070	0,0000	0,0070
4	0,0077	0,0151	-0,0074
6	0,0090	0,0077	0,0013
8	0,0115	0,0040	0,0075
10	0,0140	0,0086	0,0054
12	0,0180	0,0234	-0,0054
13	0,0320	0,0348	-0,0028
14	0,0460	0,0487	-0,0027
16	0,0873	0,0835	0,0038
20	0,1700	0,1750	-0,0050
24	0,2820	0,2821	-0,0001
26	0,3380	0,3369	0,0011
30	0,4500	0,4422	0,0078
40	0,6500	0,6601	-0,0101
50	0,8500	0,8411	0,0089
60	1,0250	1,0286	-0,0036
70	1,2000	1,1956	0,0044
80	1,3000	1,2955	0,0045
90	1,3000	1,2919	0,0081

Tabla 5- Cálculos de potencia y fuerza por sector circular y totales

radio (m)	area celda (m²)	Area total sector (m²)	Vz entrada (m/s)	Vz salida (m/s)	Vt entrada (m/s)	Vt salida (m/s)
4,54405	1,18657	85,43304	6,68679	5,87708	-0,352039	-2,73785
7,51031	1,92209	138,39048	6,24207	5,80378	-0,207516	-1,94177
10,41750	2,61304	188,13888	6,23283	5,78426	-0,145666	-1,40736
13,26680	3,26147	234,82584	6,18842	5,72966	-0,112406	-1,10499
16,05940	3,86938	278,59536	6,13717	5,60064	-0,0944888	-0,915798
18,79640	4,43865	319,58280	6,14223	5,57122	-0,0792347	-0,771926
21,47880	4,97110	357,91920	6,20145	5,63448	-0,0665897	-0,656915
24,10790	5,46847	393,72984	6,20175	5,47062	-0,0604427	-0,585574
26,68460	5,93242	427,13424	6,13395	4,91149	-0,0600675	-0,550807
29,21000	6,36465	458,25480	6,54905	5,35643	-0,0475081	-0,455483

Aire densidad Kg/m³	**1,225**

Pot. Viento	**1261285,2**
Limite Betz	**747428,269**

Presion estatica entrada (pascal)	Presion estatica salida (pascal)	Fuerza entrada (N)	Fuerza salida (N)	Momento Entrada (Nm)	Momento Salida (Nm)
20,2663	-16,8443	1731,411619	-1439,059756	-1119,473	-7652,024
24,6026	-13,6592	3404,765623	-1890,303244	-1649,227	-14348,579
24,3893	-13,0057	4588,575586	-2446,877832	-2179,825	-19544,797
24,5335	-12,2301	5761,099746	-2871,943506	-2654,712	-24162,138
24,6513	-11,8728	6867,737798	-3307,70699	-3178,248	-28111,022
24,397	-11,3635	7796,861572	-3631,579148	-3581,258	-31646,096
23,7725	-10,765	8508,634182	-3853,000188	-3888,941	-34857,330
23,4177	-10,3266	9220,247274	-4065,890566	-4358,649	-37248,787
23,1821	-10,0734	9901,868665	-4302,694053	-5144,474	-37772,340
19,4072	-8,02246	8893,442555	-3676,330803	-5101,767	-40005,737
		66674,64462	-31485,38609		-275348,849
	Variacion Presion		**-98160,0307**	**Momento**	**275348,849**
	Variacion Cantidad Movimiento		**-15818,35846**	**Potencia**	**644729.3299**
	Fuerza Empuje		**-113978,3892**		

Presion Total Entrada (pascal)	Presion Total Salida (pascal)	Fuerza Total entrada (N)	Fuerza Total salida (N)	Potencia entrada (w)	Potencia salida (w)
48,0686	9,20589	4106,646627	786,4871686	27460,2836	4622,248009
48,6492	9,31412	6732,58614	1288,985538	42025,274	7480,988483
48,3941	8,8483	9104,811773	1664,709252	56748,744	9629,111137
48,2548	8,81593	11331,47394	2070,208168	70123,92	11861,58893
48,0885	8,15102	13397,23297	2270,836351	82221,0963	12718,1369
48,028	8,45742	15348,92272	2702,845964	94276,6136	15058,14949
48,0378	9,59134	17193,65095	3432,92474	106625,567	19342,74579
47,9002	9,07381	18859,73808	3572,629759	116963,381	19544,49981
47,569	6,06624	20318,34866	2591,098812	124631,735	12726,1559
47,6884	11,7217	21853,4382	5371,525289	143119,259	28772,1992
		138246,8501	25752,25104	864195,873	141755,8237
		Empuje	**112494,599**	**Potencia**	**-722440,0492**

Termino fuente

```
#include "udf.h"

#define PiNUMBER 3.141592653 /* declaring constant: Pi number*/
#define OmegaMagnitude 2.3415 /* declaring constant: angular speed */
#define Ro 30.46 /*Tjareborg wind turbine data: RADIO [m]*/
#define B 3 /*Tjareborg wind turbine data: number of blades*/
#define THICK 6 /*dimension along the z-axis [m]*/
#define RHO 1.225 /*air density [kg/m3]*/

DEFINE_SOURCE(SourceTermX, c, t, dS, eqn)
/* DEFINE_SOURCE(SourceTermX, c, t, dS, eqn)*/
/* DEFINE_SOURCE(SourceTermX, c, t, dS, eqn)*/
{
real origin[ND_ND]; /* declaring point: the coordinates of the origin O = (0, 0, 0)
*/
real centroid[ND_ND]; /* declaring point: coordinates of cell centroids P = (X, Y, Z)
*/
real Xcoordenate; /* declaring component: x-component of the position vector = X */
real Ycoordenate; /* declaring component: y-component of the position vector = Y */
real Zcoordenate; /* declaring component: z-component of the position vector = Z */

real NV_VEC(Sposition); /* declaring vector: position vector (S) */
real Smagnitude; /* declaring magnitude: magnitude of the vector position */

real NV_VEC(ProjectionS); /* declaring vector: projection of the position vector onto
rotor axis */
real ProjectionSmagnitude; /* declaring magnitude: scalar projection of the position
vector onto rotor axis */

real NV_VEC(Radialposition); /* declaring vector: Radial position */
real Radius; /* declaring magnitude: Radius */

real theta; /* declaring angle: angular coordinate (polar angle) */

real NV_VEC(Vabsolute); /* declaring vector: absolute velocity */
real Vabsolutetangentialmagnitude; /* declaring magnitude: absolute velocity - tan-
gential component*/
real NV_VEC(Vabsolutetangential); /* declaring vector: absolute velocity - tangential
component*/

real NV_VEC(VabsolutetangentialPlusVz);
real NV_VEC(VabsoluteZ); /* declaring vector: z-componente absolute velocity */

real NV_VEC(Wrelative); /* declaring vector: relative velocity */
real Wrelativemagnitude; /* declaring magnitude: |Wrelative| */

real NV_VEC(Utangential); /* declaring vector: tangential velocity */
real Utangentialmagnitude;

real NV_VEC(Omega); /* declaring angular velocity: blade rotational speed */

real beta; /* declaring angle: pitch angle of the blade*/
real alpha; /* declaring angle: angle of attack */
real phi; /* declaring angle: angle of relative wind to rotor plane */
```

```
real NV_VEC(ZaxisCrossDaxis); /* declaring vector: Zaxis X Daxis */
real WrelativeDotUtangential;

real NV_VEC(Xaxis); /*declaring axis: unit vector in the x-direction (1, 0, 0) */
real NV_VEC(Yaxis); /*declaring axis: unit vector in the y-direction (0, 1, 0) */
real NV_VEC(Zaxis); /*declaring axis: unit vector in the y-direction (0, 0, 1) (rotor
axis) */

real NV_VEC(Raxis);
real NV_VEC(Taxis);

real NV_VEC(Daxis); /*declaring vector:  unit vector in the D-direction */
real NV_VEC(Laxis); /*declaring vector:  unit vector in the L-direction */

real chord; /*chord line [m]*/

real BE[7] = {4.3204e-008, -1.3462e-005, 1.1138e-003, -4.2397e-002, 8.5891e-001, -
9.5563e+000, 5.0075e+001};
real CU[7] = { -6.1595e-007, 6.0942e-005, -2.3676e-003, 4.5373e-002, -4.3320e-001,
1.6280e+000, 2.2782e+000};
real KL[9] ={0.000000000000688,-0.00000000025032,0.000000037,-0.00000283,0.000118,-
0.00252,0.0206,0.0504,0.382};
real KD[9]={-0.0000000000000726,0.0000000000285410,-0.00000000459,0.000000386,-
0.0000179,0.000432,-0.00416,0.0132,0.00552};

real CL, CD; /* drag and lift coefficients */

real NV_VEC(FORCE), NV_VEC(LIFT), NV_VEC(DRAG); /* forces [N]  */

real FLmagnitude, FDmagnitude;
real FXmagnitude, FYmagnitude, FZmagnitude;

real source; /* source term  [N/m3] */

/************************************************************************************/

NV_S(origin, =, 0.0); /* get origin vector of fluid region: O = (0, 0, 0)*/

NV_D(Xaxis, =, 1.0, 0.0, 0.0); /* get axis of fluid region: Xaxis = (1, 0, 0)*/
NV_D(Yaxis, =, 0.0, 1.0, 0.0); /* get axis of fluid region: Yaxis = (0, 1, 0)*/
NV_D(Zaxis, =, 0.0, 0.0, 1.0); /* get axis of fluid region: Zaxis = (0, 0, 1)*/

C_CENTROID(centroid, c, t); /* P = (X, Y, Z) */

NV_VV(Sposition, =, centroid, -, origin); /* vec(S) = P - O */
ND_SET(Xcoordenate, Ycoordenate, Zcoordenate, Sposition[0], Sposition[1],
Sposition[2]); /* vec(S) = (X, Y, Z) */

theta = atan(fabs(Ycoordenate/Xcoordenate)); /* theta = arc tan( |Y / X|) = arc tan (
|Ry / Rx| ) */

if (Xcoordenate <= 0.0 && Ycoordenate >= 0.0) {
      theta = PiNUMBER - theta; /* II */
} else if (Xcoordenate <= 0.0 && Ycoordenate <= 0.0) {
      theta += PiNUMBER; /* III */
```

```
} else if (Xcoordenate > 0.0 && Ycoordenate < 0.0) {
      theta = 2.0*PiNUMBER - theta; /* IV */
}

ProjectionSmagnitude = NV_DOT(Sposition, Zaxis); /* |vec(S)| = vec(S) · vec(k) */
NV_VS(ProjectionS, =, Zaxis, *, ProjectionSmagnitude); /* vec(Sproj) = |vec(Sproj)| *
vec(k) */

NV_VV(Radialposition, =, Sposition, -, ProjectionS); /* vec(S) = vec(projS) + vec(R)
*/
Radius = NV_MAG(Radialposition); /* |vec(R)| */
NV_VS(Raxis, =, Radialposition, /, Radius);

NV_VS(Omega, =, Zaxis, *, OmegaMagnitude); /* vec(Omega) = |Omega| * vec(k) */

NV_CROSS(Utangential, Omega, Radialposition); /* vec(U) = vec(Omega) X vec(S) */
Utangentialmagnitude = NV_MAG(Utangential);

ND_SET(Vabsolute[0], Vabsolute[1], Vabsolute[2], C_U(c,t), C_V(c,t), C_W(c,t)); /*
vec(V) */

NV_CROSS(Taxis, Zaxis, Raxis);
Vabsolutetangentialmagnitude = NV_DOT(Vabsolute, Taxis);
NV_VS(Vabsolutetangential, =,  Taxis, *, Vabsolutetangentialmagnitude);
NV_VS(VabsoluteZ, =, Zaxis, *, Vabsolute[2]);
NV_VV(VabsolutetangentialPlusVz, =, Vabsolutetangential, +,  VabsoluteZ);


NV_VV(Wrelative, =, Vabsolute, -, Utangential); /* vec(V) = vec(W) + vec(U) */
Wrelativemagnitude = NV_MAG(Wrelative); /* |Wrelative| */

WrelativeDotUtangential = NV_DOT(Wrelative, Utangential);

phi = acos(fabs(WrelativeDotUtangential/Wrelativemagnitude/Utangentialmagnitude)); /*
cos(phi) = vec(W)/|W| · vec(U)/|U|  [radians] */

beta = (BE[0]*pow(Radius,6) + BE[1]*pow(Radius,5) + BE[2]*pow(Radius,4) +
BE[3]*pow(Radius,3) + BE[4]*pow(Radius,2) + BE[5]*pow(Radius,1) +
BE[6])*PiNUMBER/180;
alpha = (phi - beta)*180.0/PiNUMBER; /* [degree] */

CL = KL[0]*pow(alpha,8) + KL[1]*pow(alpha,7) + KL[2]*pow(alpha,6) +
KL[3]*pow(alpha,5) + KL[4]*pow(alpha,4) + KL[5]*pow(alpha,3) + KL[6]*pow(alpha,2) +
KL[7]*alpha + KL[8];
CD = KD[0]*pow(alpha,8) + KD[1]*pow(alpha,7) + KD[2]*pow(alpha,6) +
KD[3]*pow(alpha,5) + KD[4]*pow(alpha,4) + KD[5]*pow(alpha,3) + KD[6]*pow(alpha,2) +
KD[7]*alpha + KD[8];

chord = CU[0]*pow(Radius,6) + CU[1]*pow(Radius,5) + CU[2]*pow(Radius,4) +
CU[3]*pow(Radius,3) + CU[4]*pow(Radius,2) + CU[5]*pow(Radius,1) + CU[6] ;

NV_VS(Daxis, =, Wrelative,/, Wrelativemagnitude);

NV_CROSS(Laxis, Daxis, Raxis);

FLmagnitude = 0.5*RHO*B*chord*pow(Wrelativemagnitude,2.0)*CL;
```

```
NV_VS(LIFT, =, Laxis, *, FLmagnitude);
FDmagnitude = 0.5*RHO*B*chord*pow(Wrelativemagnitude,2.0)*CD;
NV_VS(DRAG, =, Daxis, *, FDmagnitude);

NV_VV(FORCE, =, LIFT, +, DRAG);

FXmagnitude = NV_DOT(FORCE, Xaxis);
FYmagnitude = NV_DOT(FORCE, Yaxis);
FZmagnitude = NV_DOT(FORCE, Zaxis);

source = -FXmagnitude/THICK/2.0/PiNUMBER/Radius;
/* source = -FYmagnitude/THICK/2.0/PiNUMBER/Radius; */
/* source = -FZmagnitude/THICK/2.0/PiNUMBER/Radius; */

return source;}
```